CLUSTER AND LANDMARK ATTRIBUTES-INFUSED GRAPH NEURAL NETWORKS FOR LINK PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning positional information of nodes in a graph is important for link prediction tasks. We propose a simple representation of positional information using a set of representative nodes called landmarks. The position of a node is represented as a vector of its distances to the landmarks, where the landmarks are selected from the nodes with high degree centrality. We justify this selection strategy by analyzing well-known models of random graphs, and deriving closed-form bounds on the average path lengths involving landmarks. In a model for scale-free networks, we show that the distances to landmarks provide asymptotically accurate information on inter-node shortest distances. Our result is consistent with small-world phenomenon, i.e., landmarks can provide short paths between nodes as hubs. We apply theoretical insights to practical networks, and propose Cluster and Landmark Attributes-iNfused graph neural networks (CLAN). CLAN combines graph clustering and landmark selection, in which the graph is partitioned into densely connected clusters, and local nodes with the maximum degree is selected as landmarks. In addition, CLAN encodes the distances to landmarks using cluster-specific embedding in order to extract locality among the nodes in the common cluster. Experiments demonstrate that CLAN achieves state-of-the-art performances on various datasets in terms of HIT@K, MRR, and AUC. The improvement in those metrics averaged over datasets ranges from 8.6% to 35.2% relative to previous GNN-based methods.

1 INTRODUCTION

Graph Neural Networks are foundational methods for various graph related tasks such as node classification Kipf & Welling (2017); Veličković et al. (2018), link prediction Adamic & Adar (2003); Kipf & Welling (2016); Zhang & Chen (2018), graph classification Xu et al. (2019), and graph clustering Parés et al. (2017). In this paper, we focus on the task of link prediction using GNNs.

Message passing GNNs Gilmer et al. (2017); Kipf & Welling (2017); Hamilton et al. (2017); Veličković et al. (2018) have been successful in learning structural node representations through neighborhood aggregation. Recent approaches have proposed to incorporate *distance* attributes in the embeddings. SEAL Zhang & Chen (2018) proposed to learn from subgraph structures by extracting enclosing subgraphs, and computing relative distances associated with target nodes. DE-GNN Li et al. (2020) proposed to compute and aggregate distances from target node sets whose representation are to be learned. While the distance attributes in both works are utilized as node labels, Position-aware GNN (P-GNN) proposed by You et al. (2019) computes low-dimensional embedding of distance information to reference points called anchor nodes. Through the aggregation over anchor nodes, P-GNN is shown to effectively capture position of nodes in the graph. Especially in the link prediction, positional information between nodes is important, since two nodes which are close to each other tend to have a high probability of link formation You et al. (2019).

In this paper, we propose a simple and effective method of representing positional information of nodes. We select a set of representative nodes called *landmarks*. Each node computes the distances to the landmarks, and uses the vector of these distances to represent its position in the network. The question is, how do we select the landmarks, and how many of them, so that the distance vector is a good representation? A natural choice for a landmark is the node with *high degree* where the degree of node is often used as a measure of importance or centrality. In network models with the

property of preferential attachment (PA) Barabási & Albert (1999), nodes with very high degrees, called *hubs*, are likely to appear. Preferential attachment is a process such that, if a new node joins the graph, it is more likely to connect to nodes with higher degrees. Thus the degree of hubs tends to be big and has power-law distribution, and the network exhibits scale-free property Barabási & Albert (1999). Such hubs are abundant in social/citation networks and World Wide Web.

In this paper, we provide a theoretical justification of degree-based landmark selection for a well-known class of random graphs Erdős et al. (1960); Barabási & Albert (1999); Fronczak et al. (2004). By analyzing the average distances in random networks with preferential attachment, we show that the strategy of choosing high-degree nodes as landmarks is asymptotically optimal in the following sense. We compare the average distance between two nodes and the distances of *detour* via landmarks. We show that, the minimum distances among the detour via landmarks is asymptotically equal to the shortest path distance. This proves that the hub-type landmarks offer short paths to nodes in networks with preferential attachment, manifesting *small-world phenomenon* for scale-free networks Barabási & Albert (1999). We show that the optimality is achieved even with a small number of landmarks relative to network size. By contrast, we show that in other models where big hubs are absent, the distance reduction can be achieved by selecting higher number of landmarks.

Motivated by the theory, we propose ***C**luster and **L**andmark **A**ttributes-i**N**fused Graph Neural Networks (**CLAN**)*. Instead of simply choosing nodes with the highest degree as landmarks, CLAN first partitions graph into *clusters* which are locally dense, and appoints the node with the highest degree in the cluster as the landmark. Our intention is to bridge gap between theory and practice: hubs may not be present in practical networks, thus it is important to distribute the landmarks evenly over the network so that nodes can access nearby/local landmarks. CLAN also adopts cluster-specific embedding, which uses encoders specific to local clusters in order to capture graph structures and attributes specific to clusters. We perform experiments on various datasets, and show that CLAN achieves the state-of-the-art performance in most cases. In particular, CLAN outperforms baseline methods on dense and/or large graphs, demonstrating its effectiveness and robustness.

Our contributions are summarized as follows: 1) we propose CLAN, a simple and effective method for representing nodes' positional information using landmarks combined with graph clustering; 2) we derive closed-form bounds on average distances of paths via landmarks for well-known random networks; 3) we conduct extensive experiments on a variety of datasets, and show that CLAN outperforms existing methods in most cases.

2 Analysis of Random Graphs with Landmarks

2.1 Notation

We consider undirected graph $G = (V, E)$ where V and $E \subseteq V \times V$ denote the set of vertices and edges, respectively. Let N denote the number of nodes in the graph, or $N = |V|$. $\mathbf{A} \in R^{N \times N}$ denotes the adjacency matrix of G. For nodes $u, v \in V$, $d(v, u)$ denotes the geodesic (shortest-path) distance between v and u. The node attributes are defined as $X = \{x_1, ..., x_N\}$ where $x_i \in \mathbb{R}^n$ denotes the feature vector of node i. We consider methods of embedding node feature X into latent space $Z = \{z_1, ..., z_V\}, z_i \in \mathbb{R}^m$. We study the node-pair-level task $y(z_v, z_u)$ of predicting the link probability between node embedding z_v and z_u.

2.2 Representation of Distances using Landmarks

The distances between nodes can provide a complete description of the graph structure. For example, a connected and undirected graph can be represented by a finite metric space with its vertex set and the inter-node geodesic distances. Compact representations of distance information have been actively studied, e.g., the seminal work by Bourgain (1985) proposed the embedding of a metric space on low-dimensional Euclidean spaces with a small factor of error in distances.

We consider a representation of distance information using K representative nodes called *landmarks* denoted by $\lambda_1, \cdots, \lambda_K$. For node v, we define K-dimensional vector of distances to landmarks:

$$D(v) := (d(v, \lambda_1), d(v, \lambda_2), \cdots, d(v, \lambda_K)) \tag{1}$$

We propose to encode $D(\cdot)$ as part of node attributes into the node embedding. Note $D(\cdot)$ provides an approximate description of the position of the node in the graph. Moreover, finding shortest

paths for all pairs of nodes for a complete description requires high computational complexity, up to $O(N^3)$. Instead, each node only needs to compute the distances only to K landmarks.

The key question is, how much information $D(\cdot)$ has on the inter-node distances in the graph. From triangle inequality, we have that

$$d(u,v) \leq \min_{i=1,\dots,K}[d(u,\lambda_i) + d(\lambda_i, v)] \tag{2}$$

Eq. 2 states that the *detour* via landmarks, i.e., from u to λ_i to v, is longer than $d(u,v)$, but the shortest detour, or the minimum component of $D(u) + D(v)$, may provide a good estimate of $d(u,v)$. The key design questions are: how to select "good" landmarks, and how many of them? In the following sections, we analyze well-known random graph models, derive the average path lengths associated with landmarks, and glean design insights from the analysis.

2.3 Path lengths via Landmarks in random networks

The framework by Fronczak et al. (2004) provides a useful tool for analyzing path lengths for a wide range of classes of random networks. The asymptotic expressions for average path lengths in various random networks were derived. Following the framework, the probability of the existence of an edge for node i and j is defined as

$$q_{ij} = \frac{h_i h_j}{\beta} \tag{3}$$

where β is some constant depending on the network model. h_i is *tag* information of node i, and is called the *hidden variable* Boguná & Pastor-Satorras (2003). h is related to the connectivity or degree of the node, as will be specified according to the underlying model. From the continuum approximation Albert & Barabási (2002), h is regarded as continuous random variable (RV) with distribution $\rho(\cdot)$. Below we denote the mean of RVs by $\langle\cdot\rangle$, whereas $\langle\cdot\rangle_Q$ denotes the expectation with respect to some distribution Q.

Theorem 1 *Let L_{ij} denote the random variable representing the minimum path length from node i to j among the detour via $K(N)$ landmarks. The landmarks are chosen i.i.d. according to distribution Q. Asymptotically in N, we have that*

$$P(L_{ij} > s) = \exp\left[-\frac{h_i h_j}{\beta N \langle h^2\rangle} \cdot \langle h^2\rangle_Q K(N) \cdot (s-1)\left(\frac{\langle h^2\rangle N}{\beta}\right)^{s-1}\right], \quad s = 1, 2, \cdots \tag{4}$$

In Eq. 4, the design parameters are $\langle h^2\rangle_Q$ and $K(N)$ which are related to what kind of landmarks are chosen, and how many of them, respectively. Next, we bound the average of the minimum path length among the detours via landmarks.

Theorem 2 *Assume $K(N) = o(N)$ and $K(N) \to \infty$ as $N \to \infty$. The mean of the minimum path length among the detour via landmarks, denoted by $\bar{l}$, is bounded above as,*

$$\bar{l} \leq \frac{-2\langle \log h\rangle - \log\left(\langle h^2\rangle_Q K(N)\right) + \log(N\beta\langle h^2\rangle) + \log\log\left(\frac{N\langle h^2\rangle}{\beta}\right) - \gamma}{\log N + \log\langle h^2\rangle - \log\beta} + \frac{1}{2} \tag{5}$$

where $\gamma \approx 0.5772$ is the Euler's constant.

The assumption $K(N) = o(N)$ implies that the number of landmarks are chosen to be not too large compared to N. The proofs of Theorem 1 and 2 are based on the framework by Fronczak et al. (2004), and are provided in the Appendix A. We apply the results to some well-known random graph models.

2.4 Erdős-Rényi Model

The Erdős-Rényi (ER) model is a classical random graph in which every node pair is connected with a common probability. We have that

$$\beta = \langle k\rangle N, \quad \rho(h) = \delta(h - \langle k\rangle). \tag{6}$$

where $\langle k \rangle$ denotes the mean degree of nodes. By applying Eq. 6 to Eq. 3, we have $q_{ij} = \langle k \rangle / N$, i.e., the edge probability is constant. The node degree follows the Poisson distribution with mean $\langle k \rangle$ for large N. Assume $\langle k \rangle$ is a finite constant. From Eq. 5, the mean of the minimum of path length via landmarks in ER network, denoted by $\bar{l}_{\text{ER}}$, is bounded above as

$$\bar{l}_{\text{ER}} \leq \frac{2 \log N - \log K(N)}{\log \langle k \rangle} \tag{7}$$

asymptotically in N. From Fronczak et al. (2004), the average length of the shortest paths without landmarks, denoted by $\bar{l}^*_{\text{ER}}$, is given by

$$\bar{l}^*_{\text{ER}} = \frac{\log N}{\log \langle k \rangle} \tag{8}$$

We observe that, by comparing Eq. 7 and 8, the detour via landmarks incurs the overhead of at most factor 2. This is because nodes in ER graphs appear *homogeneous*, and thus the path length to and from landmarks are on average similar to the inter-node distance. Thus the distance of a detour will be twice the direct distance. The cost can be reduced by using a large number of landmarks, say $K(N) = N^{1-\varepsilon}$ for some $\varepsilon \in (0, 1)$. Then

$$\bar{l}_{ER} \leq \frac{(1+\varepsilon) \log N}{\log \langle k \rangle} = (1+\varepsilon) \cdot \bar{l}^*_{\text{ER}} \tag{9}$$

For example, selecting $\sqrt{N}$ landmarks guarantees a 1.5 factor approximation of the optimal distance. By making ε close to 0, then we get arbitrarily close to the shortest path distance.

Discussion. Due to having Poisson distribution, the degrees in ER graphs are highly concentrated on mean $\langle k \rangle$. There seldom are nodes with very large degrees, i.e., most nodes look alike. Thus the design question should be on *how many* rather than on *what kind* of landmarks. We benefit from choosing a large number of landmarks, e.g., $K(N) = N^{1-\varepsilon}$. There however is a tradeoff: the computational overhead of managing $K(N)$-dimensional vector $D(v)$ will be high.

2.5 Barabási-Albert (BA) Model

The Barabási-Albert (BA) model is a simple random graph with preferential attachment. The BA model is characterized by continuous growth over time with preferential attachment. Initially there are m nodes, and new nodes arrive to the network over time. The preferential attachment in BA networks is such that, the probability of the connection of the existing node to the newly arriving node is *proportional* to its degree. In Boguná & Pastor-Satorras (2003), it is shown that

$$q_{ij} = \frac{m}{2} \frac{1}{\sqrt{t_i t_j}}$$

where t_i is the time of arrival of node i. Thus we have $h_i = 1/\sqrt{t_i}$ which gives $\beta = \frac{m}{2}$. It was shown in Boguná & Pastor-Satorras (2003) that the probability of a newly arriving node connecting to node i is proportional to h_i. Thus, the degree of nodes with large h_i is likely to be high. The asymptotic expression of the distribution of hidden variables was derived in Fronczak et al. (2004)

$$\rho(h) = \frac{2}{N} h^{-3}, \; h \in \left[\frac{1}{\sqrt{N}}, 1 \right].$$

By applying $\rho(\cdot)$ to Eq. 5, we get the following bound on the average path length with landmarks denoted by $\bar{l}_{\text{BA}}$:

$$\bar{l}_{\text{BA}} \leq \frac{\log N - \log\left(\langle h^2 \rangle_Q K(N)\right) + \log \log N + \log \log \log N + \log \left[2 \log(m/2)/m\right]}{\log \log N + \log(m/2)} + \frac{1}{2} \tag{10}$$

We show that, unlike ER graphs, there exists a landmark selection strategy which achieves the asymptotically optimal distance, despite using a small number of landmarks relative to the network size.

Selecting Landmarks with Large h. We consider distribution of landmark selection:

$$Q(t) = \rho(t | h \geq \frac{1}{\sqrt{M(N)}}) = \rho(t) \cdot \mathbf{1}\left(t \in \left[\frac{1}{\sqrt{M(N)}}, 1 \right] \right) \Big/ P\left(h \in \left[\frac{1}{\sqrt{M(N)}}, 1 \right] \right) \tag{11}$$

where $M(N) = g(N) \cdot K(N)$ for some slowly increasing function $g(N)$, say $g(N) = \log N$. Importantly, distribution $Q(\cdot)$ implies that, we select landmarks from $M(N)$ nodes with highest values of h. Specifically $K(N)$ landmarks are chosen at random from a pool of $M(N) = g(N) \cdot K(N)$ nodes with the largest h. Then we get

$$\bar{l}_{\text{BA}} \leq \frac{\log N - \log\log(g(N)K(N)) + \log g(N) + \log\log N + \log\log\log N}{\log\log N} \tag{12}$$

where we assume $m = O(1)$, and drop the related terms as well as the constant from Eq. 10. The derivation of Eq. 12 is provided in Appendix A.3. The numerator of Eq. 12 is $\approx \log N$ for large N. The average distance in BA networks is given by Cohen & Havlin (2003); Fronczak et al. (2004)

$$\bar{l}^{*}_{\text{BA}} = \frac{\log N}{\log\log N}$$

Thus, $\bar{l}_{\text{BA}}$ is asymptotically equal to $\bar{l}^{*}_{\text{BA}}$. This implies that, the shortest detour via landmarks has the same length on average as the shortest direct path in the asymptotic sense.

Discussion. The node degree in BA networks is known to follow power-law distribution, which predicts the presence of big hubs. The inter-node distance can be drastically reduced due to the presence of hubs, which is known as *small-world phenomenon* Barabási & Albert (1999). In conclusion, the shortest path between two nodes are well-approximated by the detour via landmarks chosen from high-degree nodes. Notably, this is achieved even without large number of landmarks, say we let $K(N) = \log N$ in Eq. 12. Finally note that, the landmark selection strategy can affect the performance bound. Suppose landmarks are selected at random. Then $\langle h^2 \rangle_Q = \langle h^2 \rangle$, and

$$\bar{l}_{BA} \leq \frac{2\log N - \log(K(N)) + \log\log\log N}{\log\log N}$$

where the results are similar to the ER model, e.g., 2-factor approximation, etc.

2.6 Design Insights from Theory

The key design parameters in our method are $\langle h^2 \rangle_Q$ and $K(N)$ in Eq. 5.

Landmark Selection and Graph Clustering. In order to make $\langle h^2 \rangle_Q$ large, one may choose landmarks with as large h, e.g., high-degree nodes, as possible. In practice, however, such high-degree nodes may not always provide short paths, unlike scale-free networks. Suppose all the hubs are located at one end of the network. The nodes at the other end of the network have to make a long detour via landmarks, even in order to reach nodes in local neighborhoods.

In order to better capture local graph structures, we propose to partition the network into *clusters* such that the nodes within a cluster tend to be densely connected, i.e., close to one another. Then we pick the node with the highest degree within each cluster as the landmark, as suggested by the analysis. Each landmark represents the associated cluster, and the distance between two nodes in remote clusters can be captured by distance between respective landmarks. Also, the distances within the cluster can be estimated by the distances to the local landmark. We empirically find that such combination of clustering and landmark selection yields improved results.

Number of Landmarks. Although large number of landmarks $K(N)$ appears preferable, our analysis show that $K(N)$ does *not* drastically reduce distances, unless $K(N)$ is very large, e.g., $N^{1-\epsilon}$. A large number of landmarks can severely hamper the scalability. Thus, we will use only a moderate number of landmarks, and empirically find that setting $K(N) = \log N$ suffices to yield good results.

3 Proposed Method

Below we describe **C**luster and **L**andmark **A**ttributes-i**N**fused GNN (**CLAN**). CLAN consists of graph clustering, landmark selection, distance computation, and cluster-specific embedding.

3.1 Graph Clustering

We partition the graph into K-disjoint clusters $C_1, \cdots, C_K \subseteq V$ using FluidC graph clustering algorithm proposed in Parés et al. (2017). FluidC initially selects K random central nodes, assigns

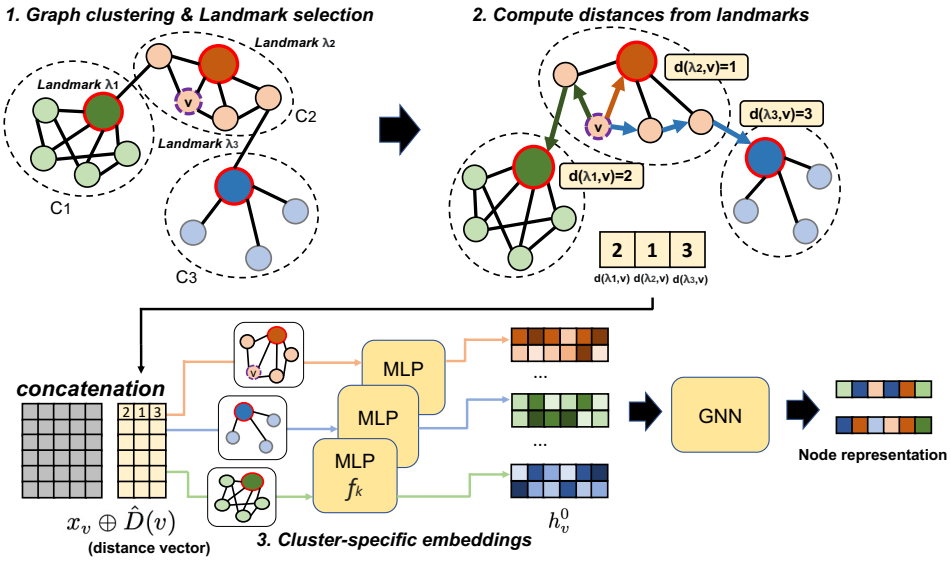

Figure 1: Overview of CLAN when $K = 3$. First, CLAN partitions the graph into K-clusters. Second, CLAN selects the highest-degree node as a landmark in each cluster and computes distances among selected nodes and other nodes to concatenate distance vectors with node features. Third, CLAN utilizes cluster-specific embeddings to extract structure and attributes of local clusters.

each node to $C_1, \cdots, C_K$, and iteratively assigns nodes to clusters according to the following rule. Node u is assigned to cluster $C_{k^*(u)}$ such that

$$k^*(u) = \underset{k=1,\ldots,K}{\arg\max} \frac{|\{u, \mathcal{N}(u)\} \cap C_k|}{|C_k|} \tag{13}$$

where $\mathcal{N}(u)$ denotes the neighbors of u. This assignment rule prefers community candidates of small sizes (denominator), which results in well-balanced cluster sizes. The rule also prefers communities already containing many neighbors of u (numerator), which makes clusters locally dense. As a result, FluidC generates densely-connected cohesive clusters of relatively even sizes.

In addition, FluidC is highly scalable as compared to existing algorithms Parés et al. (2017). FluidC adopts an efficient propagation method, and has the time complexity linear in $|E|$. Moreover, in FluidC, one can specify the number of clusters to be detected. This is important, because the number of clusters or K is a hyperparameter in our method. As mentioned in Section 2.6, we let $K = \log N$.

3.2 Landmark Selection and Approximated Distances

For each cluster, we select the node with the highest degree as the landmark.

$$\lambda_k = \text{Hub}(C_k) \tag{14}$$

where λ_k denotes the landmark of cluster C_k, and $\text{Hub}(C_k)$ denotes the highest-degree node in cluster C_k. For each node, we compute the distances to landmarks $\lambda_1, \cdots, \lambda_K$ to yield K-dimensional vector of distances. Since G may not be a connected graph, we define the following:

$$\hat{d}(v, \lambda_k) = \begin{cases} d(v, \lambda_k), & \text{if a path exists from } v \text{ to } \lambda_k, \\ d_{\max} + 1, & \text{otherwise.} \end{cases} \tag{15}$$

where $d_{\max}$ is defined as follows. For each node v, we compute the distances to the landmarks within the connected component containing v and set $d_{\max}$ to the maximum among all the computed distances. Finally, the distance vector of node v, denoted by $\hat{D}(v)$ is given by

$$\hat{D}(v) = (\hat{d}(v, \lambda_1), \hat{d}(v, \lambda_2) ..., \hat{d}(v, \lambda_K)) \tag{16}$$

3.3 CLUSTER-SPECIFIC EMBEDDING

We will combine node features and $\hat{D}(v)$ to compute the node embeddings. Specifically, for node v, feature x_v and $\hat{D}(v)$ are concatenated and input to an encoder. We propose *cluster-specific embedding* as follows. The embedding of a node in a cluster is computed by the encoder specific to the cluster. Since there are K clusters, we use K separate encoders, one for each cluster.

The motivation behind cluster-specific embedding is as follows. If nodes u and v are closely located, $\hat{D}(u)$ and $\hat{D}(v)$ will look similar. Thus, if an encoder is used to embed $\hat{D}(\cdot)$ of nodes sampled from a small region, the variation in $\hat{D}(\cdot)$ will be relatively small. The encoder will be relatively easier to train because its input data has lower variability, as compared to the case where a single encoder is used to embed $\hat{D}(\cdot)$ sampled from the entire graph. In addition, using a separate encoder per local region may facilitate capturing attributes specific to the local graph structure, which is important for link prediction tasks. The proposed heuristics use one encoder per local cluster.

We encode the node attributes as

$$z_v = f_k(x_v \oplus \hat{D}(v)), \quad x_v \in C_k, \tag{17}$$

where x_v is the input node features, and $\oplus$ denotes concatenation. f_k denotes the encoder associated with the cluster C_k. We use a simple Multi-Layer Perceptron (MLP) for f_k. The output embedding is input to the GNN layers as follows:

$$h_v^l = \text{GNN}(h_v^{l-1}, \mathbf{A}), \quad h_v^0 = z_v, \quad l = 1, 2, \cdots. \tag{18}$$

where h_v^l denotes hidden embedding vector of l-th layer of the GNN.

3.4 COMPLEXITY ANALYSIS

Firstly we consider the time complexity of computing $\hat{D}(v)$ for all $v \in V$. There are $\log N$ landmarks, and for each landmark, computing the distances from all $v \in V$ to the landmark requires $O(|E| + N \log N)$ using Fibonacci heap. Thus the overall complexity is $O(|E| \log N + N \log^2 N)$. The computation of $\hat{D}(v)$ is done once, and thus can be considered as a preprocessing step.

Next, cluster-specific embedding uses MLP with input dimension of $(n + \log N)$ or the length of $x_v \oplus \hat{D}(v)$. There are a total of $\log N$ MLPs, which adds to the space complexity of our model. The resulting model seems to have a manageable size, and in our experiments, the overall size of MLPs is up to 1.6M parameters in the worst case. The size overhead can be reduced by letting the multiple clusters share a single MLP, although it was unnecessary in our experiments.

The complexity of the rest of training phase depends on the type of GNN used in the last embedding step, where we use GCN as the default implementation. Overall, the computational complexity of CLAN is reasonable, and our experiments shows that CLAN handles large or dense graphs well.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETTING

Datasets. Experiments were conducted on 7 datasets widely used for evaluating link prediction. For experiments on small graphs, we used PubMed, Cora, Citeseer, and Facebook. For experiments on dense or large graphs, we chose DDI, COLLAB, and CITATION2 provided by OGB Hu et al. (2020). Detailed statistics and metrics are illustrated in Table 5 in Appendix C.

Baseline models. We compared CLAN with a heuristic method (AA) Adamic & Adar (2003), Matrix Factorization (MF) Koren et al. (2009), Node2Vec Grover & Leskovec (2016), GCN Kipf & Welling (2017), GraphSAGE Hamilton et al. (2017), GAT Veličković et al. (2018), P-GNN You et al. (2019), and plug-in type approaches such as JKNet Xu et al. (2018b), SEAL Zhang & Chen (2018), GCN+DE Li et al. (2020), LGLP Cai et al. (2021), and LRGA Puny et al. (2021). All methods except for AA and GAE are computed by the same decoder, which is a 2-layer MLP. For a fair comparison, we only use GCN in all plug-in type approaches such as SEAL, GCN+DE, GAE, JKNet, GCN+LRGA, and CLAN. We set GCN+DE and GCN+LRGA as the same environment proposed by Zhang et al. (2021).

Baselines	Avg. (H.M.)	CITATION2	COLLAB	DDI	PubMed	Cora	Citeseer	Facebook
Adamic Adar	65.74 (50.65)	76.12 ± 0.00	53.00 ± 0.00	18.61 ± 0.00	66.89 ± 0.00	77.22 ± 0.00	68.94 ± 0.00	99.41 ± 0.00
MF	54.06 (42.65)	53.08 ± 4.19	38.74 ± 0.30	17.92 ± 3.57	58.18 ± 0.01	51.14 ± 0.01	50.54 ± 0.01	98.80 ± 0.00
Node2Vec	64.01 (51.17)	53.47 ± 0.12	41.36 ± 0.69	21.95 ± 1.58	80.32 ± 0.29	84.49 ± 0.49	80.00 ± 0.68	86.49 ± 4.32
GCN (GAE)	76.35 (66.67)	84.74 ± 0.21	44.14 ± 1.45	37.07 ± 5.07	95.80 ± 0.13	88.68 ± 0.40	85.35 ± 0.60	98.66 ± 0.04
GCN (MLP)	75.05 (66.43)	84.79 ± 0.24	44.29 ± 1.88	39.31 ± 4.87	95.83 ± 0.80	90.25 ± 0.53	71.47 ± 1.40	99.43 ± 0.02
GraphSAGE	78.51 (71.48)	82.64 ± 0.01	48.62 ± 0.87	44.82 ± 7.32	96.58 ± 0.11	90.24 ± 0.34	87.37 ± 1.39	99.29 ± 0.01
GAT	-	OOM	44.14 ± 5.95	29.53 ± 5.58	85.55 ± 0.23	82.59 ± 0.14	87.29 ± 0.11	99.37 ± 0.00
JKNet	-	OOM	48.84 ± 0.83	57.98 ± 7.68	96.58 ± 0.23	89.05 ± 0.67	88.58 ± 1.78	99.43 ± 0.02
P-GNN	-	OOM	OOM	1.14 ± 0.25	87.22 ± 0.51	85.92 ± 0.33	90.25 ± 0.42	93.13 ± 0.21
GCN+DE	72.48 (60.04)	60.30 ± 0.61	53.44 ± 0.29	26.63 ± 6.82	95.42 ± 0.08	89.51 ± 0.12	86.49 ± 0.11	99.38 ± 0.02
GCN+LRGA	78.42 (74.47)	65.05 ± 0.22	52.21 ± 0.72	62.30 ± 9.12	93.53 ± 0.25	88.83 ± 0.01	87.59 ± 0.03	99.42 ± 0.05
LGLP	-	OOM	OOM	OOM	OOM	91.30 ± 0.05	89.41 ± 0.13	98.51 ± 0.02
SEAL	77.08 (62.88)	85.26 ± 0.98	53.72 ± 0.95	26.25 ± 8.00	95.86 ± 0.28	92.55 ± 0.50	85.82 ± 0.44	**99.60 ± 0.02**
CLAN	**84.50 (80.83)**	**85.64 ± 0.48**	**54.47 ± 0.75**	**67.02 ± 6.85**	**96.62 ± 0.21**	**93.25 ± 0.15**	**95.02 ± 0.13**	99.53 ± 0.00

Table 1: Link prediction results on various datasets. All baselines and our method were evaluated for 10 repetitions. **Bold** denotes the best performance. We used a single NVIDIA RTX 3090 with 24GB memory on all datasets except CITATION2 and A100 GPU with 40GB memory on CITATION2. OOM indicates 'out-of-memory'. For SEAL and GCN+DE, we trained 2% of training data and evaluated 1% of both validation and test set respectively on CITATION2. We also trained 15% of training data but evaluated all of the validation and test sets on COLLAB. Both implementations have followed the guideline on the official GitHub of SEAL-OGB. 'Avg.' denotes the average of performance metrics, and 'H.M' indicates their harmonic mean. (-) means that we do not report the average and harmonic mean due to OOM.

Evaluation metrics. Link prediction was evaluated based on the ranking performance of positive edges in the test data over negative ones. For COLLAB and DDI, we ranked all positive and negative edges in the test data, and computed the ratio of positive edges which are ranked in top-k. We did not utilize validation edges for computing node embeddings when we predicted test edges on COLLAB. In CITATION2, we computed all positive and negative edges, and calculated the reverse of the mean rank of positive edges. However, due to high complexity when evaluating SEAL, we only trained 2% of training set edges and evaluated 1% of validation and test set edges respectively, as recommended in the official GitHub of SEAL. For Cora, Citeseer, PubMed, and Facebook, we utilized Area Under ROC Curve (AUC). If applicable, we calculated the average and the harmonic mean of the measurements. The harmonic mean severely penalizes the model for very low scores, thus is a useful indicator of robustness.

Hyperparameters. We used GCN as our base GNN encoder. In the ablation study in Appendix B, we provide the performance with different encoders such as GraphSAGE or GAT. MLP is used in both cluster-specific embeddings and decoders, except GAE. CLAN does not use edge weights. In training our model, we used learning rates of 0.0005–0.001, batch size of 65,536, and the Adam optimizer Kingma & Ba (2015). Detailed hyperparameters are provided in Appendix D.

4.2 RESULTS

Experimental results are summarized in Table 1. CLAN outperformed the baselines on most datasets. CLAN achieved large performance gains over GAE combined with GCN on all datasets, which are 80.8% on DDI, 11.3% on Citeseer, 23.4% on COLLAB, 5.2% on Cora, and 1.1% on CITATION2. CLAN showed superior performance over SEAL, achieving gains of 155% on DDI, and 10.7% on Citeseer. We compare CLAN with other distance-based methods. Compared to GCN+DE which encodes distances from a target node set whose representations are to be learned, or to P-GNN which uses distances to random anchor sets, CLAN achieved higher performance gains by a large margin. The results show that approximate inter-node distances via landmarks can be effective for representing positional information of nodes.

GAT, JKNet, LGLP, and P-GNN suffered from out-of-memory (OOM) on large graphs such as CITATION2 and COLLAB, due to the high memory usage from storing a large number of shortest paths, attention weights, or aggregation of hidden embedding vectors, etc. SEAL performed poorly on DDI which is a highly dense graph. Since the nodes of DDI have a large number of neighbors, the enclosing subgraphs are both very dense and large, and the model struggles with learning local structural representations. By contrast, CLAN achieved the best performance on DDI, demonstrating its effectiveness on densely connected graphs as well.

Finally, we computed the average and harmonic means of measurements except for the methods with OOM problems. Although the averages are taken over heterogeneous metrics, and thus the values do not represent specific performance metrics, they are presented for comparison purposes. In summary, CLAN achieved the best average and harmonic mean of performance measurements on the entire datasets, demonstrating both its effectiveness and robustness.

Ablation Study. Ablation study on the components of our method: graph clustering, landmark selection, and cluster specific embedding, is provided in Appendix B.

5 Related Work

Graph Neural Networks. GNN methods Kipf & Welling (2017); Hamilton et al. (2017); Veličković et al. (2018); Xu et al. (2018a;b) extract node features and perform neighborhood aggregation to compute node embeddings. Such node representation methods provide inductive bias for representing graphs, and can be applied to various graph-related downstream tasks. The aforementioned methods utilize the message-passing architecture Gilmer et al. (2017).

Link Prediction. Katz (1953); Adamic & Adar (2003); Zhou et al. (2009) proposed link prediction heuristics using manually designed formulas. Liben-Nowell & Kleinberg (2003) extensively conducted empirical studies on various heuristics, and Sarkar et al. (2011) provided their theoretical justification. Kipf & Welling (2016) proposed graph auto-encoder which reconstructs adjacency matrices combined with GNNs. Zhang & Chen (2018) proposed structural link representation by extracting enclosing subgraphs and learning structural patterns of those subgraphs. The authors demonstrated that higher-order heuristics can be approximately represented by lower-order enclosing subgraphs thanks to γ-decaying heuristic. Cai & Ji (2020) proposed the approach of multi-scale link learning, which learns enclosing subgraphs at various scales. Cai et al. (2021) proposed the line graph transformation prior to GNN layers for link prediction. Pan et al. (2022) proposed a link prediction method by learning subgraph structure based on random walks. Zhao et al. (2022) proposed counterfactual learning framework for link prediction to learn causal relationships between nodes.

Distance and Position based GNN models. You et al. (2019) proposed position-aware GNN (P-GNN) based on distances for injecting positional information into node embeddings. P-GNN focuses on realizing Bourgain's embedding Bourgain (1985) guided by Linial's method Linial et al. (1995), and performs message computation and aggregation based on distances to random subset of nodes. By contrast, we judiciously select representative nodes (landmarks) in combination with graph clustering, and use the associated distances as input attributes to nodes. Li et al. (2020) proposed distance encoding for node labels and theoretical analysis regarding the expressive power of GNN. Zhang et al. (2021) analyzed the effects of various node labeling tricks using distances. However, these two methods do not utilize distances as positional information.

Networks with Landmarks. Ng & Zhang (2002) proposed to estimate inter-node distances using detours via landmarks for Internet systems. Kleinberg et al. (2004) analyzed the theory of approximating distances between nodes utilizing landmarks. However, they consider randomly selected landmarks, where we also consider the distance under a judicious selection strategy. Potamias et al. (2009) proposed to use vectors of distances to landmarks to estimate inter-node distances.However, the work did not provide theoretical analysis on the distances under the detour via landmarks.

6 Conclusion

We proposed to use landmark-based distance vectors to capture nodes' positional information for link prediction. We provided theoretical analysis of the average distances of detours via landmarks for well-known random graphs. From the analysis, we obtained design guidelines on the type and number of landmarks to be selected, and proposed CLAN which effectively infuses cluster and landmark attributes for the link prediction on real-world graphs. CLAN achieved state-of-the-art performance as compared to existing methods on various graph datasets of diverse sizes and densities. A limitation of our work is that, CLAN performs the positional node embeddings through transductive learning, because it learns from the preprocessed graph attributes. Thus, it is hard to apply CLAN to inductive learning tasks. In the future, we plan to study various topological attributes for link prediction, and propose to combine these attributes with inductive learning methods.

REFERENCES

Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286 (5439):509–512, 1999.

Marián Boguná and Romualdo Pastor-Satorras. Class of correlated random networks with hidden variables. *Physical Review E*, 68(3):036112, 2003.

Jean Bourgain. On lipschitz embedding of finite metric spaces in hilbert space. *Israel Journal of Mathematics*, 52(1):46–52, 1985.

Lei Cai and Shuiwang Ji. A multi-scale approach for graph link prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 3308–3315, 2020.

Lei Cai, Jundong Li, Jie Wang, and Shuiwang Ji. Line graph neural networks for link prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Reuven Cohen and Shlomo Havlin. Scale-free networks are ultrasmall. *Physical review letters*, 90 (5):058701, 2003.

Paul Erdős, Alfréd Rényi, et al. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Agata Fronczak, Piotr Fronczak, and Janusz A Hołyst. Average path length in random networks. *Physical Review E*, 70(5):056110, 2004.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. PMLR, 2017.

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*, pp. 855–864, 2016.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL `http://arxiv.org/abs/1412.6980`.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=SJU4ayYgl`.

Jon Kleinberg, Aleksandrs Slivkins, and Tom Wexler. Triangulation and embedding using small sets of beacons. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pp. 444–453. IEEE, 2004.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

Pan Li, Yanbang Wang, Hongwei Wang, and Jure Leskovec. Distance encoding: Design provably more powerful neural networks for graph representation learning. *Advances in Neural Information Processing Systems*, 33:4465–4478, 2020.

David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pp. 556–559, 2003.

Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

TS Eugene Ng and Hui Zhang. Predicting internet network distance with coordinates-based approaches. In *Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 1, pp. 170–179. IEEE, 2002.

Liming Pan, Cheng Shi, and Ivan Dokmanić. Neural link prediction with walk pooling. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=CCu6RcUMwK0.

Ferran Parés, Dario Garcia Gasulla, Armand Vilalta, Jonatan Moreno, Eduard Ayguadé, Jesús Labarta, Ulises Cortés, and Toyotaro Suzumura. Fluid communities: A competitive, scalable and diverse community detection algorithm. In *International conference on complex networks and their applications*, pp. 229–240. Springer, 2017.

Michalis Potamias, Francesco Bonchi, Carlos Castillo, and Aristides Gionis. Fast shortest path distance estimation in large networks. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 867–876, 2009.

Omri Puny, Heli Ben-Hamu, and Yaron Lipman. Global attention improves graph networks generalization, 2021. URL https://openreview.net/forum?id=H-BVtEaipej.

Purnamrita Sarkar, Deepayan Chakrabarti, and Andrew W Moore. Theoretical justification of popular link prediction heuristics. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ. accepted as poster.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018a.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pp. 5453–5462. PMLR, 2018b.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Jiaxuan You, Rex Ying, and Jure Leskovec. Position-aware graph neural networks. In *International conference on machine learning*, pp. 7134–7143. PMLR, 2019.

Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018.

Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin. Labeling trick: A theory of using graph neural networks for multi-node representation learning. *Advances in Neural Information Processing Systems*, 34:9061–9073, 2021.

Tong Zhao, Gang Liu, Daheng Wang, Wenhao Yu, and Meng Jiang. Learning from counterfactual links for link prediction. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 26911–26926. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/zhao22e.html.

Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630, 2009.

A PROOFS

A.1 PROOF OF THEOREM 1.

We follow the proof technique in Fronczak et al. (2004), and briefly describe their technique. We first state a key lemma from Fronczak et al. (2004):

Lemma 1 *If $A_1, \cdots, A_n$ are mutually independent events and their probabilities fulfill relations $\forall_i P(A_i) \leq \varepsilon$,*

$$P\left(\cup_{k=1}^{n} A_k\right) = 1 - \exp\left(-\sum_{k=1}^{n} P(A_k)\right) - R$$

where $0 \leq R < \sum_{j=0}^{n+1} (n\varepsilon)^j / j! - (1+\varepsilon)^n$.

where it can be shown that R vanishes in the limit $n \to \infty$. The authors estimate the probability

$$1 - \exp\left[-\sum_{v_1=1}^{N} \cdots \sum_{v_{s-1}=1}^{N} q_{iv_1} q_{v_1 v_2} \cdots q_{v_{s-1} j}\right] \tag{19}$$

where q_{ij} is defined in Eq. 3. This represents the probability that there is at least one walk (i.e., revisiting a node is allowed) from i to j in s steps by applying Lemma 1. An event A_k in the lemma corresponds to a walk starting at i ends at j, and the expression in Eq. 19 counts all the possible paths and sums up the probability. This expression is asymptotically accurate: although the same edge may participate between different A_k's and induce correlation, it is argued in Fronczak et al. (2004) that the fraction of such correlations vanishes when $s \ll N$.

We extend the arguments to the walks visiting a given landmark. Consider node $i, j \in V$ and landmark $\lambda \in V$. Consider the collection of walks of length s which visits λ during the walk. Let $p_{ij}^{\lambda}(s)$ denote the probability that the length of path between i and j via λ is less than or equal to s for $s = 1, 2, \cdots$. Using Lemma 1, $p_{ij}^{\lambda}(s)$ is given by

$$p_{ij}^{\lambda}(s) = 1 - \exp\left[-\left\{\sum_{v_1=1}^{N} \cdots \sum_{v_{s-1}=1}^{N} q_{iv_1} q_{v_1 v_2} \cdots q_{v_{s-1} j} - \sum_{\substack{v_1=1 \\ v_1 \neq \lambda}}^{N} \cdots \sum_{\substack{v_{s-1}=1 \\ v_{s-1} \neq \lambda}}^{N} q_{iv_1} q_{v_1 v_2} \cdots q_{v_{s-1} j}\right\}\right]$$

where q_{ij} is defined in Eq. 3, and the error in $p_{ij}^{\lambda}(s)$ vanishes as $N \to \infty$ from the lemma. The expression in the bracket considers all the walks from i to j in s steps, and visiting λ at least once during the walk. We have

$$\sum_{v_1=1}^{N} \cdots \sum_{v_{s-1}=1}^{N} q_{iv_1} q_{v_1 v_2} \cdots q_{v_{s-1} j} = h_i h_j \frac{\left(N\langle h^2 \rangle\right)^{s-1}}{\beta^s}$$

whereas

$$\sum_{\substack{v_1=1 \\ v_1 \neq \lambda}}^{N} \cdots \sum_{\substack{v_{s-1}=1 \\ v_{s-1} \neq \lambda}}^{N} q_{iv_1} q_{v_1 v_2} \cdots q_{v_{s-1} j} = h_i h_j \frac{\left(N\langle h^2 \rangle - h_\lambda^2\right)^{s-1}}{\beta^s}$$

Thus the subtraction in the bracket is given by

$$\begin{aligned}
\frac{h_i h_j}{\beta^s}\left[\left(N\langle h^2 \rangle\right)^{s-1} - \left(N\langle h^2 \rangle - h_\lambda^2\right)^{s-1}\right] &= \frac{h_i h_j}{\beta^s}\left(N\langle h^2 \rangle\right)^{s-1}\left[1 - \left(1 - \frac{h_\lambda^2}{N\langle h^2 \rangle}\right)^{s-1}\right] \\
&\approx \frac{h_i h_j}{\beta^s}\left(N\langle h^2 \rangle\right)^{s-1}(s-1)\frac{h_\lambda^2}{N\langle h^2 \rangle} \\
&= \frac{h_i h_j h_\lambda^2}{\beta N\langle h^2 \rangle}(s-1)\left(\frac{N\langle h^2 \rangle}{\beta}\right)^{s-1}
\end{aligned}$$

Let random variable $L_{ij}(\lambda)$ denote the path length visiting landmark λ, and

$$F_\lambda(s) := P(L_{ij}(\lambda) > s).$$

We have that

$$F_\lambda(s) = \exp\left[-\frac{h_i h_j h_\lambda^2}{\beta N \langle h^2 \rangle}(s-1)\left(\frac{\langle h^2 \rangle N}{\beta}\right)^{s-1}\right],\ s = 1, 2, \cdots$$

Consider the minimum distance among the routes via landmarks λ_k, $k = 1, \cdots, K(N)$ where the landmarks are chosen i.i.d. from distribution $\sim Q$. Let L denote the minimum distance among the routes visiting the landmarks. Then $L_{ij} := \min[L_{ij}(\lambda_1), L_{ij}(\lambda_2), \cdots, L_{ij}(\lambda_{K(N)})]$ where

$$\begin{aligned}
P(L_{ij} > s) &= P(\min[L_{ij}(\lambda_1), L_{ij}(\lambda_2), \cdots, L_{ij}(\lambda_{K(N)})] > s) \\
&= P(L_{ij}(\lambda_1) > s, L_{ij}(\lambda_2) > s, \cdots, L_{ij}(\lambda_{K(N)}) > s) \\
&= \prod_{k=1}^{K(N)} P(L_{ij}(\lambda_k) > s) \\
&= \exp\left[-\frac{h_i h_j}{\beta N \langle h^2 \rangle}\left(\sum_{k=1}^{K(N)} h_{\lambda_k}^2\right)(s-1)\left(\frac{\langle h^2 \rangle N}{\beta}\right)^{s-1}\right] \\
&= \exp\left[-\frac{h_i h_j}{\beta N \langle h^2 \rangle} K(N) \cdot \frac{1}{K(N)}\left(\sum_{k=1}^{K(N)} h_{\lambda_k}^2\right)(s-1)\left(\frac{\langle h^2 \rangle N}{\beta}\right)^{s-1}\right] \\
&= \exp\left[-\frac{h_i h_j}{\beta N \langle h^2 \rangle} K(N) \cdot \langle h^2 \rangle_Q \cdot (s-1)\left(\frac{\langle h^2 \rangle N}{\beta}\right)^{s-1}\right]
\end{aligned}$$

where $\langle \cdot \rangle_Q$ denotes the expectation of the hidden variables of landmarks chosen according to $\sim Q$ assuming $K(N)$ is sufficiently large.

A.2 Proof of Theorem 2.

Let L_{ij} be the mean of the minimum path length among the detour via landmarks from i to j. We have that

$$l_{ij} = \sum_{s=1}^{\infty} P(L_{ij} > s) = \sum_{s=0}^{\infty} \exp\left[-\frac{h_i h_j}{\beta N \langle h^2 \rangle} \cdot \langle h^2 \rangle_Q K(N) \cdot s\left(\frac{\langle h^2 \rangle N}{\beta}\right)^{s}\right]$$

We utilize the Poisson summation formula:

$$l_{ij} = \frac{1}{2} f(0) + \int_0^\infty f(t)\, dt + 2\sum_{n=1}^{\infty} \int_0^\infty f(t) \cos(2\pi n t)\, dt \tag{20}$$

where

$$f(t) = \exp\left[-atb^t\right], \tag{21}$$

$$a := \langle h^2 \rangle_Q K(N) \cdot \frac{h_i h_j}{\beta N \langle h^2 \rangle}, \tag{22}$$

$$b := \frac{\langle h^2 \rangle N}{\beta} \tag{23}$$

Firstly we have $f(0) = 1$. Next, we evaluate the second term of Eq. 20:

$$\int_0^\infty \exp\left(-atb^t\right) dt = \int_0^\infty \exp\left(-ate^{t \log b}\right) dt = (\log b)^{-1} \int_0^\infty \exp\left(-\frac{a}{\log b} t e^t\right) dt \tag{24}$$

Let $u = te^t$, then we have

$$dt = \frac{du}{u + e^{W(u)}}$$

where $W(\cdot)$ is the Lambert W function which is the inverse of te^t for $t \geq 0$. Thus Eq. 24 is equal to

$$(\log b)^{-1} \int_0^\infty \frac{\exp\left(-\frac{a}{\log b}u\right)}{u + e^{W(u)}} du$$

Since $W(u) \geq 0$ for $u \geq 0$, Eq. 24 is bounded above by

$$(\log b)^{-1} \int_0^\infty \frac{\exp\left(-\frac{a}{\log b}u\right)}{u+1} du = (\log b)^{-1} \exp\left(\frac{a}{\log b}\right) \int_1^\infty \frac{\exp\left(-\frac{a}{\log b}u\right)}{u} du \tag{25}$$

$$= -\exp\left(\frac{a}{\log b}\right) \frac{\mathrm{Ei}\left(-\frac{a}{\log b}\right)}{\log b} \tag{26}$$

where $\mathrm{Ei}(\cdot)$ denotes the exponential integral. Consider the assumption of the theorem:

$$K(N) = o(N), \tag{27}$$

i.e., the number of landmarks is not too large compared to N. Under this assumption, one can verify that $a/\log b$ is at most $o(N)/N$ which tends to 0 as $N \to \infty$. Thus the exponential term of Eq. 26 can be approximated to 1. Moreover,

$$-\frac{\mathrm{Ei}\left(-\frac{a}{\log b}\right)}{\log b} = \frac{-\gamma - \log a + \log\log b}{\log b} \tag{28}$$

$$= \frac{-\log(h_i h_j) - \log\left(\langle h^2\rangle_Q K(N)\right) + \log(N\beta\langle h^2\rangle) + \log\log\left(\frac{N\langle h^2\rangle}{\beta}\right) - \gamma}{\log N + \log\langle h^2\rangle - \log\beta} \tag{29}$$

In Eq. 28, we used

$$-\mathrm{Ei}\left(-\frac{a}{\log b}\right) \approx -\gamma - \log\left(\frac{a}{\log b}\right)$$

where the error term vanishes because $a/\log b$ is small, and $\gamma \approx 0.5772$ is the Euler's constant.

Finally, similar to the derivation in Appendix B of Fronczak et al. (2004), one can show that the last term of Eq. 20 vanishes, by using generalized mean value theorem. By averaging Eq. 29 over all $i, j \in V$, we get

$$\bar{l} \leq \frac{-2\langle\log h\rangle - \log\left(\langle h^2\rangle_Q K(N)\right) + \log(N\beta\langle h^2\rangle) + \log\log\left(\frac{N\langle h^2\rangle}{\beta}\right) - \gamma}{\log N + \log\langle h^2\rangle - \log\beta} + \frac{1}{2} \tag{30}$$

from Eq. 20.

A.3 Derivation of Eq. 12

From Eq. 11, we have that

$$\langle h^2\rangle_Q = \frac{\left\langle h^2 \mathbf{1}\left(h \in \left[\frac{1}{\sqrt{M(N)}}, 1\right]\right)\right\rangle}{P\left(h \in \left[\frac{1}{\sqrt{M(N)}}, 1\right]\right)}$$

We have

$$P\left(h \in \left[\frac{1}{\sqrt{M(N)}}, 1\right]\right) = \int_{\frac{1}{\sqrt{M(N)}}}^1 \rho(h)\, dh = \frac{2}{N} \int_{h=\frac{1}{\sqrt{M(N)}}}^1 h^{-3}\, dh \approx \frac{M(N)}{N}$$

and

$$\left\langle h^2 \mathbf{1}\left(h \in \left[\frac{1}{\sqrt{M(N)}}, 1\right]\right)\right\rangle = \int_{h=\frac{1}{\sqrt{M(N)}}}^1 h^2 \rho(h)\, dh = \frac{2}{N} \int_{h=\frac{1}{\sqrt{M(N)}}}^1 h^{-1}\, dh \approx \frac{\log M(N)}{N}$$

Thus, we have

$$\langle h^2\rangle_Q = \frac{\log M(N)}{M(N)}$$

Applying the result to Eq. 10, and using $M(N) = g(N)K(N)$, we obtain Eq. 12.

B ABLATION STUDY

We present an ablation study to investigate the advantages of components of CLAN. Table B show the performance on OGB-DDI and OGB-COLLAB where

- "Distance Vector" column indicates the usage of the vector of distances to landmarks,
- "Cluster specific Embeddings" column indicates the usage of cluster-specific MLPs for encoding node attributes.

We observe that, by adding either of the components "Distance Vector" and "Cluster-specific Embedding" of CLAN, the prediction performance is improved. As a result, CLAN obtained 70.5% higher performance on OGB-DDI, and 23.0% better performance on OGB-COLLAB than basic GCN.

Distance Vectors	Cluster-specific Embeddings	DDI (HIT@20)	COLLAB (HIT@50)
✗	✗	39.31 ± 4.87	44.29 ± 1.88
✔	✗	46.86 ± 9.91	53.31 ± 0.60
✗	✔	54.00 ± 8.90	52.34 ± 0.39
✔	✔	$\mathbf{67.02 \pm 6.85}$	$\mathbf{54.47 \pm 0.75}$

Table 2: Ablation study on OGB-DDI and OGB-COLLAB.

In Table 3, we investigate the effects of graph clustering and landmark selection.

- In "Graph Clustering" column, 'Random' means that the graph is randomly partitioned, and 'FluidC' means that the graph clustering is done using FluidC algorithm.
- In "Landmark Selection" column, 'Random' means that landmarks is randomly selected, and 'Hub' means that the node with largest degree within the cluster is selected as the landmark.

The results show that, only performing a proper selection of landmarks, or only performing a proper graph clustering is not effective. The performance is significantly improved only if graph clustering and landmark selection are properly combined. One possible interpretation is that, FluidC is the density-based propagation algorithm, and thus the resulting clusters are locally dense and are likely to include nodes with high degrees which may be chosen as a "good" landmark. Thus, we conclude that graph clustering and landmark selection have synergistic effect on the performance.

Graph Clustering	Landmark selection	HIT@20
Random	Random	59.15 ± 7.68
FluidC	Random	59.01 ± 6.38
Random	Hub	60.97 ± 8.36
FluidC	Hub	$\mathbf{67.02 \pm 6.85}$

Table 3: Ablation study with graph clustering and landmark selection strategies on OGB-DDI.

In addition, we experimented with CLAN combined with three different GNN encoders. Table 4 shows that, CLAN generally enhances the performance of various types of GNNs.

Dataset	CLAN w/ GraphSAGE	CLAN w/ GAT	CLAN w/ GCN
PubMed	$96.47\ (-0.11) \pm 0.29$	$91.42\ (+5.87) \pm 0.24$	$\mathbf{96.62}\ (+0.79) \pm 0.21$
Cora	$92.68\ (+2.44) \pm 0.16$	$89.22\ (+6.63) \pm 0.15$	$\mathbf{93.25}\ (+3.00) \pm 0.15$
Citeseer	$94.32\ (+6.95) \pm 0.15$	$93.26\ (+5.97) \pm 0.12$	$\mathbf{95.02}\ (+4.77) \pm 0.13$
Facebook	$99.41\ (+0.12) \pm 0.01$	$98.76\ (-0.53) \pm 0.00$	$\mathbf{99.53}\ (+0.10) \pm 0.00$

Table 4: Ablation study on GNN types for CLAN. + denotes the performance gain over the default encoder for each GNN, i.e., without CLAN.

C DATASETS

Dataset	# Nodes	# Edges	$\frac{\#\text{Edges}}{\#\text{Nodes}}$	Avg. node deg	Density	Split ratio	Metric
Cora	2,708	7,986	2.95	5.9	0.0021%	70/10/20	AUC
Citeseer	3,327	7,879	2.36	4.7	0.0014%	70/10/20	AUC
PubMed	19,717	64,041	3.25	6.5	0.00033%	70/10/20	AUC
Facebook	4,039	88,234	21.85	43.7	0.0108%	70/10/20	AUC
OGB-DDI	4,267	1,334,889	312.84	500.5	14.67%	80/10/10	Hits@20
OGB-CITATION2	2,927,963	30,561,187	10.81	20.7	0.00036%	98/1/1	MRR
OGB-COLLAB	235,868	1,285,465	5.41	8.2	0.0046%	92/4/4	Hits@50

Table 5: Dataset statistics.

D HYPERPARAMETERS

Hyperparameter	**Value**
Encoder of all plug-in methods	GCN
Batch size	65,536
Learning rate	0.001, 0.0005
Hidden dimension	256
Number of GNN layers	2, 3
Number of Decoder layers	2, 3
max hop of GCN+DE	3
Negative sampling	Uniformly Random sampling
Dropout	0.2, 0.5
Negative sample rate	1
JKNet mode	Concatenation
Activation function	ReLU (GNNs), LeakyReLU (f_k)
Cluster-specific embedding layer	MLP
Loss function	BCE Loss
Use edge weights	False (only binary edge weights)
The number of landmarks	$\log(\#\text{ nodes})$
Optimizer	Adam

Table 6: Detailed hyperparameters. We easily reproduced and experimented thanks to Fey & Lenssen (2019); Hu et al. (2020); Zhang et al. (2021).

E NODE CENTRALITY

For each cluster, the most "central" node should be selected as the landmark. We have used Degree centrality in this paper; however, there are other types of centrality such as Betweenness and Closeness. Betweenness centrality is a measure of how often a given node is included in the shortest paths between node pairs. Closeness centrality is the reciprocal of the sum-length of shortest paths to the other nodes.

Table 7 shows the experimental results comparing Degree, Betweenness and Closeness centralities. The results show that the performances are similar among the centralities. Thus, all the centralities are effective measures for identifying "important" nodes. Degree centrality, however, was slightly better than the other choices.

More importantly, Betweenness and Closeness centralities require full information on inter-node distances, which incurs high computational overhead. In Table 7, we excluded datasets CITATION2 and COLLAB which are too large graphs to compute Betweenness and Closeness centralities. Scalability is crucial for link prediction methods. Thus we conclude that, from the perspective of scalability and performance, Degree centrality is the best choice.

Centrality	DDI	PubMed	Cora	Citeseer	Facebook
Degree	$\mathbf{67.02 \pm 6.85}$	$\mathbf{96.62 \pm 0.21}$	$\mathbf{93.25 \pm 0.15}$	$\mathbf{94.75 \pm 0.13}$	$\mathbf{99.53 \pm 0.00}$
Betweeness	66.41 ± 6.77	96.36 ± 0.19	92.69 ± 0.12	94.44 ± 0.08	99.32 ± 0.00
Closeness	65.56 ± 6.65	95.92 ± 0.20	92.43 ± 0.10	94.65 ± 0.12	99.35 ± 0.00

Table 7: Link prediction results from landmark selection with different centrality.

F $(1+\epsilon)$-BOUND FOR REAL-WORLD GRAPHS

In Sec. 2.4, we proved that the average path length via landmarks is bounded above by $(1+\epsilon)$ times the average distance in ER graphs, if the number of landmarks is $N^{1-\epsilon}$: see Eq. 9. Below we provide experimental results to check the bound for real-world graphs.

Table 8 shows the average path length via randomly selected landmarks measured from benchmark datasets. The number of landmarks is $N^{1-\epsilon}$ with $\epsilon = 0.3, 0.5$, and 0.8. Interestingly, the average path length indeed does not exceed $(1+\epsilon)$-factor of the actual distance in all cases.

Our interpretation is that, the factor of $(1+\epsilon)$ for ER graphs may serve as a conservative bound for real-world graphs. In ER graphs, every node has a similar degree, and thus it is unlikely to choose landmarks that provide short paths like hubs. By contrast, in real-world graphs, the variance of node degree can be higher. Thus, there are chances that several high-degree nodes are selected as landmarks, which can reduce the detour distances.

# of landmarks	**COLLAB**	**DDI**	**PubMed**	**Cora**	**Citeseer**	**Facebook**	**1+ϵ**
$N^{0.3}$	9.01 (117%)	4.53 (145%)	11.90 (163%)	11.41 (157%)	14.99 (144%)	7.26 (136%)	170%
$N^{0.5}(\sqrt{N})$	8.79 (115%)	3.90 (125%)	10.31 (141%)	9.61 (132%)	13.04 (126%)	6.76 (127%)	150%
$N^{0.8}$	7.92 (104%)	3.44 (110%)	8.31 (114%)	8.03 (110%)	10.93 (105%)	5.77 (108%)	120%
actual distances	7.64 (100%)	3.12 (100%)	7.29 (100%)	7.27 (100%)	10.38 (100%)	5.32 (100%)	100%

Table 8: Comparison of average path lengths via landmarks and actual distances in benchmark datasets.

G NUMBER OF MODEL PARAMETERS

We compare the numbers of parameters in the models. The number of parameters changes with the size of graph datasets. We chose the DDI dataset, since all the models, including CLAN, have the largest parameter count for the DDI dataset. Table 9 shows the parameter count for CLAN and baseline models. We observe that CLAN has the largest number of parameters. The number, however, can be lowered by reducing the number of cluster-specific encoders as follows.

	CLAN	GCN+LRGA	LGLP	P-GNN	GCN+DE	GraphSAGE
# parameters	1,812,672	1,229,537	894,721	462,854	519,425	262,656

Table 9: Comparison of the number of parameters.

Table 10 shows the performance by varying the number of clusters per encoder. For example, if the number of clusters per encoder is 2, every two clusters share one encoder. Thus, the total number of cluster-specific encoders is halved.

The results show that, we achieve better performance with more encoders. However, the best tradeoff is achieved when the number of clusters per encoder is 2. Compared to the default model, the performance decreases only slightly, but the number of parameters is reduced by 46%. In this case, the number of parameters is less than GCN+LRGA, and is on par with LGLP.

# clusters per encoder	# parameters	DDI	COLLAB	PubMed	Cora	Citeseer	Facebook
1 (default)	1,812,672	**67.02 ± 6.85**	**54.47 ± 0.75**	**96.62 ± 0.21**	**93.25 ± 0.15**	**95.02 ± 0.13**	**99.53 ± 0.00**
2	973,152	66.71 ± 6.62	53.71 ± 0.57	96.02 ± 0.11	93.02 ± 0.14	94.92 ± 0.03	99.41 ± 0.00
4	**553,392**	54.44 ± 4.31	53.70 ± 0.60	94.06 ± 0.13	87.16 ± 0.04	93.21 ± 0.08	99.31 ± 0.00

Table 10: Performance with different numbers of cluster-specific encoders.