# Modeling Context Pair Interaction for Pairwise Tasks on Graphs

Zhen Wang
The Ohio State University
Columbus, OH, USA
wang.9215@osu.edu

Bo Zong
NEC Laboratories, America
Princeton, NJ, USA
bzong@nec-labs.com

Huan Sun
The Ohio State University
Columbus, OH, USA
sun.397@osu.edu

## ABSTRACT

Predicting pairwise relationships between nodes in graphs is a fundamental task in data mining with many real-world applications, such as link prediction on social networks, relation prediction on knowledge graphs, etc. A dominating methodology is to first use advanced graph representation methods to learn generic node representations and then build a pairwise prediction classifier with the target nodes' vectors concatenated as input. However, such methods suffer from low interpretability, as it is difficult to explain why certain relationships are predicted only based on their prediction scores. In this paper, we propose to model the pairwise interactions between neighboring nodes (i.e., contexts) of target pairs. The new formulation enables us to build more appropriate representations for node pairs and gain better model interpretability (by highlighting meaningful interactions). To this end, we introduce a unified framework with two general perspectives, node-centric and pair-centric, about how to model context pair interactions. We also propose a novel pair-centric context interaction model and a new pre-trained embedding, which represents the pair semantics and shows many attractive properties. We test our models on two common pairwise prediction tasks: link prediction task and relation prediction task, and compare them with graph feature-based, embedding-based, and Graph Neural Network (GNN)-based baselines. Our experimental results show the superior performance of the pre-trained pair embeddings and that the pair-centric interaction model outperforms all baselines by a large margin.

## KEYWORDS

Graph Pairwise Prediction; Context Pair Interaction; Interpretable Graph Learning

**ACM Reference Format:**
Zhen Wang, Bo Zong, and Huan Sun. 2021. Modeling Context Pair Interaction for Pairwise Tasks on Graphs. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441744

## 1 INTRODUCTION

Pairwise prediction, which aims to predict relationships between two nodes in a graph, is a fundamental problem in data mining and machine learning, and has a wide range of practical applications. For example, friend recommendation in social networks [40] recommends potential friends to a user by predicting his/her links with other users. Other examples include multi-relational link prediction in knowledge graphs [41], weakly-supervised relation extraction in entity co-occurrence graphs [28], etc.

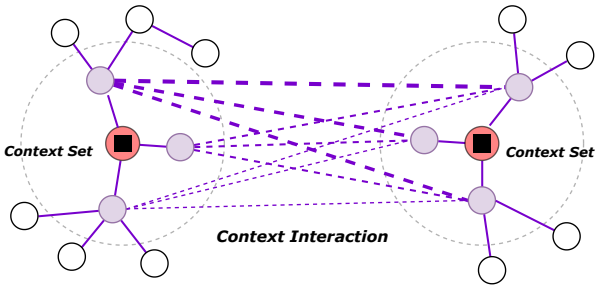



**Figure 1: Intuition Illustration. To predict the potential relationships between target nodes $u$ and $v$, we model the pairwise interactions between their context nodes (purple ones within the circle) and aggregate important interaction information (dashed lines) for the final prediction. The thickness of a line reflects how important the interaction is.**

Given a graph and a pair of target nodes $(u, v)$, many powerful representation learning methods [10, 11, 16, 25, 29, 34, 35, 37, 39] have been proposed to encode various graph information into node vectors, $\vec{\boldsymbol{v}}_u$, $\vec{\boldsymbol{v}}_v$, and predict their relationships based on these vectors. By preserving the graph structure and properties into a low-dimensional latent vector space, these representation learning methods have obtained state-of-the-art results in many pairwise prediction tasks, such as link prediction [45], relation prediction [28], etc. However, such methods suffer from low transparency and interpretability and it is difficult for users to understand and trust the prediction decisions. This is because, various graph information, e.g., graph structure, context information, is all encoded into a latent vector space *implicitly*, and the pairwise prediction is usually made based on the similarity metrics of node vectors. For example, when predicting whether two users are friends in social networks, users may want to know how the decision is made and the model would be less likely to be convincing to users if the provided explanation is that two user vectors have high cosine similarity.

In this paper, we propose to *explicitly* model the interactions between neighborhoods[1] of target nodes for the pairwise prediction. Taking a toy example for illustrating our intuition, as shown in Figure 1, the interaction links between target nodes could be indicative for predicting that node $u$ is connected with $v$ (e.g., some context nodes of node $u$ are very similar or hold special relations with context nodes of node $v$). Learning such meaningful patterns explicitly among context interaction links in graphs is expected

[1] In this paper, we use context and neighborhood interchangeably.


*WSDM '21, March 8–12, 2021, Virtual Event, Israel*

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441744

to shed some light on interpreting the model's decision (by highlighting important interaction links for the prediction). In addition, most of previous methods learn generic embeddings from the graph, which might be suboptimal for pairwise relationship predictions. In contrast, modeling context pair interactions enables us build pair-specific representations to further improve the model performance.

One straightforward way to manipulate the contexts of target nodes for the interaction is to compute heuristic features between their contexts, such as common neighbors, Jaccard similarity [20]. Such features provide competitive performance in some pairwise prediction tasks, such as link prediction, and intuitive explanations for justifying the model prediction. However, they are also known for the lack of generalizability [17]. For example, though common neighbors feature works reasonably well in link prediction, Kovács et al. [17] find that linked proteins do not necessarily share many neighbors in the Protein-Protein Interaction (PPI) network.

Thus, in this paper, we propose to combine the representation learning with context interaction and take the first step towards explicitly exploring Context Pair Interaction for pairwise prediction with a general deep learning framework, ConPI ("con-$\pi$"). Our framework essentially computes an interaction score for the pair of contexts (i.e., pairwise interaction) jointly with the pairwise prediction task, as shown in Figure 1. We provide two perspectives of aggregating the pair interaction information, node-centric and pair-centric. From the node-centric perspective (referred to as ConPI-node model), the model learns interaction-aware node representations for target nodes whose contexts dynamically influence each other. The final prediction is made based on the simple combination of new node representations as traditional embedding methods do [10]. On the other hand, from a relatively new perspective of pair-centric (referred to as ConPI-pair), we novelly propose to directly model pair representations for each interacting pair and aggregate them together for the final prediction. Such a new perspective further motivates us to propose a new type of pair embedding (which represents the semantics of each context pair) and inject it back into the ConPI-pair model for more efficient learning. Finally, our ConPI framework offers a certain amount of model interpretability that can generate instance-level explanations, i.e., meaningful and important context links, which could be easily understood by users and thus, obtain their trust better.

Our contributions are summarized as follows:

- We highlight the importance of context interactions for pairwise predictions. To the best of our knowledge, we are the first to systematically study how to model context pair interactions in pairwise tasks. Our study aims to inspire more graph-based models to study the interaction mechanisms.
- We propose a general framework ConPI with two perspectives, node-centric and pair-centric, and build a novel model for the pair-centric view that aggregates context pair representations directly for the final prediction. Our framework considers the mutual influence between nodes via context interactions and enhances the interpretability by highlighting important context pairs that lead to the model decision.
- We also propose a new type of pair embedding on homogeneous graph to capture semantics of any node pairs and inject it back to ConPI-pair model to show its effectiveness.
- We conduct extensive experiments on two types of pairwise prediction tasks, link prediction and relation prediction with totally 6 datasets. In comparison with strong baselines from different categories, our framework can achieve very competitive performance and more importantly, much better interpretability.

## 2 RELATED WORK

**Pairwise Tasks on Graphs**. Given a graph, there are generally three types of features that can be leveraged effectively for pairwise predictions, heuristic features (e.g., common neighbors), latent features (e.g., node embeddings) and explicit features (e.g., node attributes). In this paper, as we mainly utilize the graph information for the interaction, we do not focus on modeling explicit features at this moment. For the heuristic features, they can be directly extracted from the graph, which is computationally efficient and could be competitive baselines for some tasks, such as link prediction. But they may lose the generalizability across different datasets [17]. On the other hand, latent features, i.e., low-dimensional embedding vectors learned from the graph, can be optimized via various approaches, e.g., matrix factorization [4, 26], random walk [10, 25], neural networks [34, 39]. However, modeling pairwise prediction in the latent space may suffer several disadvantages as we mentioned previously, such as lacking a certain amount of interpretability or learning node representation independently from the target pairwise prediction task. Our framework ConPI deals with such issues by explicitly modeling context pair interactions between nodes.

**Neighborhood Aggregation.** Our framework takes as input neighborhoods from the graph and aggregates them for the pairwise prediction, which is closely related to the line of recent researches in neighborhood aggregation for graph learning. For example, Hamilton et al. [11] propose to aggregate features of neighboring nodes by different functions, e.g., Mean, LSTM, and Pooling. Veličković et al. [37] leverage the multi-head self-attention mechanism to aggregate neighborhood information for node representations. Sun et al. [33] introduce gated multi-hop neighborhood aggregation to align nodes from knowledge graphs. All the aforementioned methods inspire us to thoroughly explore neighborhoods for pairwise predictions. Instead of aggregating neighborhoods independently, we propose that we need to consider the interactions between contexts for a better aggregation. The ConPI-node method models the context pair interactions with a similar mechanism as mutual attention module that has been applied in many other tasks [19, 30, 36, 42] and our contribution for ConPI-node here is to unify it into a framework for systemically modeling the context pair interactions.

**Graph Representation Learning.** Representation learning on graphs has been extensively studied and mostly focuses on nodes [5, 10, 25, 29, 39], i.e., projecting nodes to a low-dimensional vector space while preserving the graph structure [29, 39] and properties [46]. The learned embeddings can be reused for a variety of downstream tasks, such as the pairwise prediction tasks as mentioned above. More recently, while being far less mature compared with node embeddings, learning edge semantics and representing edges beyond nodes have received increasing attention [1, 3, 22, 31, 38, 48]. First of all, there are some recent works trying to learn representations of connected edges in the graph [3, 48], which cannot be generalized to unobserved pairs of nodes, e.g., the context

pairs in our interaction module. To learn embeddings for any pair of nodes, Abu-El-Haija et al. [1] model the asymmetry property between two nodes and define an edge function over node vectors to produce a score for the pair rather than a pair vector. In addition, edge embeddings [31, 38] are studied in heterogeneous networks to learn task-relevant representations for specific pairs, e.g., author-paper pairs in citation networks. In this paper, we aim to pre-train a general-purpose pair embedding on homogeneous graphs and incorporate it into our context pair interaction model for more general pairwise prediction tasks. More relevantly, Joshi et al. [14] propose to learn word pair embeddings from a large text corpus to incorporate text patterns into the pair embeddings while our ConPI-pair model (Section 4.3) focuses on learning node pair embeddings from the graph structure.

## 3 PRELIMINARIES

In this section, we first introduce our problem definition and, then describe how we can obtain contexts for our framework.

### 3.1 Task Definition

**Notations.** We denote an undirected graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the the set of existing edges. For any node $x \in \mathcal{V}$, we denote its neighbor[2] set as $\mathcal{N}(x)$.

**Problem Definition.** In this paper, we consider the pairwise prediction tasks on graphs with a generalized definition in which the label to predict between nodes can by any pairwise supervisions, e.g., link label, relation label. Specifically, given the graph $\mathcal{G}$ and target pairs $\mathcal{S}$ consisting of a few positive training labels $\mathcal{S} = \{(u_i, v_i)\}_{i=1}^{N}, v, u \in \mathcal{V}$. We aim to leverage the training pairs to make prediction for unseen pairs, $P(y|(u'_j, v'_j))$ where $y$ indicates whether the pair holding the target label. We formally define the pairwise prediction as follows:

**Definition 3.1. (Pairwise Prediction)**. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a set of target pairs $\mathcal{S} = \{(u_i, v_i)\}_{i=1}^{N}$, we aim to estimate the possibility of a new node pair holding the target label, $P(y|(u'_j, v'_j))$ and make a binary prediction for the new pair[3].

### 3.2 Context Acquisition.

As our framework explicitly utilizes contexts for the pairwise prediction, we always assume the neighbor set of each node is given for the rest of the paper. As the background, we give a brief introduction about how we can acquire contexts in order to apply our framework. For the simplest case, we can uniformly sample a fix-number of neighbors [11, 37]. For example, for the neighbor set $\mathcal{N}(u)$, we sample from the set $\{v \in \mathcal{V}, (u, v) \in \mathcal{E}\}$.

Moreover, in some graphs, the degree distribution has a long tail in which for some nodes, we may not able to sample enough contexts or even zero contexts (i.e., out-of-vocabulary issue). Under such circumstances, we can leverage some context prediction methods to recover sufficient high-potential contexts for our following context interaction framework [12, 42]. Instead of recovering the context graph by GNNs [12], we need discrete contexts for our framework. That is, we can estimate a context conditional probability $p(c|u)$ representing the likelihood that context node $c$ is connected with node $u$:

$$p(c|u) = \frac{\exp(\boldsymbol{\nu}_c^T \cdot \text{Encoder}(u))}{\sum_{i=1}^{|\mathcal{V}|} \exp(\boldsymbol{\nu}_{c_i}^T \cdot \text{Encoder}(u))} \tag{1}$$

where $\boldsymbol{\nu}_c$ is the context embedding for node $c$ and $\text{Encoder}(\cdot)$ is used to encode the node features into a vector, for instance, we can adopt Recurrent Neural Networks (RNNs) to encode the article content from citation networks into a single vector. There are multiple options to optimize such probability [24, 34]. For example, we can employ the cross-entropy loss to minimize the distance between the estimated probability with the empirical probability from the original graph. We refer to Tang et al. [34] for more theoretical details. After pre-training such a conditional probability on the graph, given a node $v$, we can select the top-$L_v$ entities from $p(\cdot|v)$ as $v$'s neighbors for the subsequent modeling.

## 4 CONPI FRAMEWORK

In this section, we introduce our ConPI framework for modeling context pair interactions in pairwise tasks. We first give an overview and then introduce ConPI-node and ConPI-pair models as well as how to pre-train and utilize pair embeddings.

### 4.1 Framework Overview

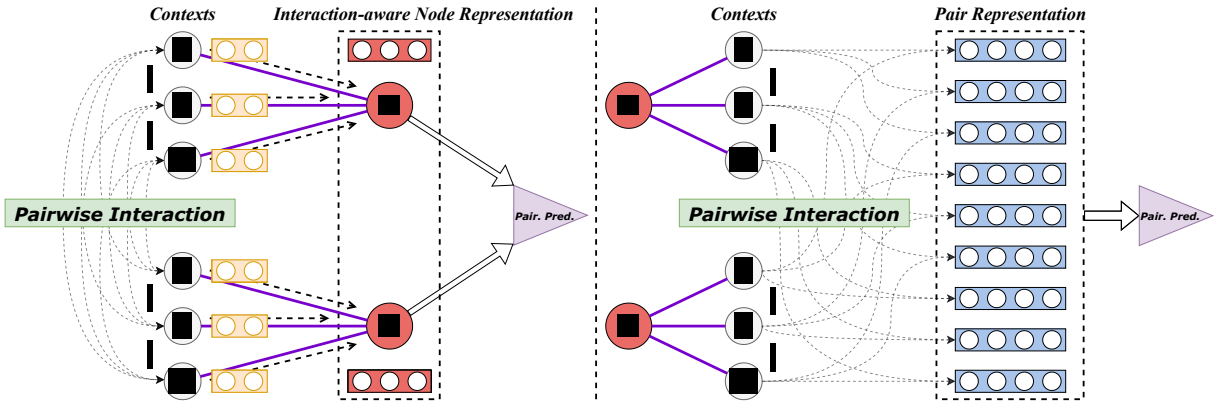
Current state-of-the-art pairwise prediction models usually focus on learning high-quality node representations with advanced node embedding methods [10, 29, 39] or graph neural networks [11, 16, 37], which perform reasonably well across different tasks. However, there are two challenges that our framework tries to solve. Firstly, though preserving rich graph information, the learned latent representation is hard to be comprehended by humans, and thus, has a lack of interpretability. Secondly, when learning node representations, most models project each node to a latent vector space independently by preserving as much graph information as possible, but ignore the mutual influence between the pair when applying the learned vectors for the pairwise prediction task.

To solve the challenges as mentioned above, our proposed framework ConPI, as shown in Figure 2, presents two different perspectives for conducting context pair interactions for pairwise prediction. For the node-centric context interaction (Figure 2 (a)), interaction-aware node representations are firstly learned by aggregating context information with the consideration of the pairwise interaction. By doing so, node representations of target nodes will mutually affect each other for a better pairwise prediction. Moreover, instead of learning node representations, we model the context pairs directly for the pairwise prediction (Figure 2 (b)). Specifically, for each context pair, we formulate a pair representation (either training a pair encoder function taking as input the node representations or applying pre-trained pair embeddings) and then aggregate them for the pairwise prediction. In next parts, we will introduce model details and show how to obtain interpretable results for pairwise prediction by our framework.

### 4.2 Node-centric Context Interaction

The goal of the node-centric context interaction model (ConPI-node) is to first form an interaction-aware representation for each

[2]We consider 1-hop neighbor in this paper.
[3]Note that our task can be easily extended to multi-class setting.




*(a) Node-centric Context Interaction* *(b) Pair-centric Context Interaction*

**Figure 2: Framework Overview: (a) ConPI-node; (b) ConPI-pair.**

target node and make the pairwise prediction based on the new representations. Inspired by previous graph neighborhood aggregation methods that aggregate contexts with a self-attention layer [37], we propose to aggregate the context information for each node in such a way that each context node is weighed based on its interaction links with contexts on the other side. Thereby each node representations in the pairwise prediction task would influence each other via the interaction links between their contexts.

Formally, as shown in Figure 2 (a), given two target nodes in a graph, $(u, v) \in V$, we obtain their neighbor set, $\mathcal{N}(u) = \{c_u^1, ..., c_u^{L_u}\}$ and $\mathcal{N}(v) = \{c_v^1, ..., c_v^{L_v}\}$ as mentioned in Section 3.2. Without losing the generalizability, we assign a feature vector to each context node and we get two sets of vectors, $\mathbf{\Psi}_u = \{\boldsymbol{\nu}_u^i\}_{i=1}^{L_u}, \mathbf{\Psi}_v = \{\boldsymbol{\nu}_v^j\}_{j=1}^{L_v}$ for target nodes $u, v$, as the input for ConPI-node model.

There are several neighborhood aggregation mechanisms that have been tried for mapping a set of context vectors to a node representation, e.g., Mean, LSTM, Pooling [11] or Multi-head Self-attention [37]. However, none of them consider the factor of context interactions with other nodes for the pairwise prediction when aggregating the contexts. Thus, we leverage the attention-based approach Santos et al. [30], Tu et al. [36] to weigh each context with the consideration of its interactions with contexts on the other side and aggregate the weighed contexts for target node $u$ as follows:

$$\boldsymbol{h}_u = \sum_{i=1}^{L_u} \alpha_u^i \cdot \boldsymbol{\nu}_u^i \tag{2}$$

$$\alpha_u^i = \frac{\exp(S_{\text{pool}}(\boldsymbol{\nu}_u^i, \mathbf{\Psi}_v))}{\sum_{k=1}^{L_u} \exp(S_{\text{pool}}(\boldsymbol{\nu}_u^k, \mathbf{\Psi}_v))} \tag{3}$$

$$S_{\text{pool}}(\boldsymbol{\nu}_u^i, \mathbf{\Psi}_v) = \text{Pool}\left(\left\{\eta(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j), \forall \boldsymbol{\nu}_v^j \in \mathbf{\Psi}_v\right\}\right) \tag{4}$$

$$\eta(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j) = \sigma\left(\boldsymbol{\nu}_u^{i\,T} \cdot \mathbf{W}_{\text{pool}} \cdot \boldsymbol{\nu}_v^j\right) \tag{5}$$

where $\mathbf{W}_{\text{pool}}$ is the weight matrix for the pooling function. $S_{\text{pool}}$ is a pooling function (e.g., mean or max function) to measure the interaction score between one context vector with a set of context vectors in which the (pairwise) interaction between each pair of contexts is measured by a non-linearity function $\eta$ (Eqn.5).

To this end, the representation, $\boldsymbol{h}_u$, for target node $u$ incorporates the interaction information from the other side, and with a similar operation in the opposite direction, we can obtain the interaction-aware representation for node $v$ as $\boldsymbol{h}_v$. Finally, the pairwise prediction can be made by applying a classifier on a binary operator for $\boldsymbol{h}_u, \boldsymbol{h}_v$ as what the previous works do [8, 10].

**Interpretability.** ConPI-node model generates explanations for model decisions in a two-steps process as follows: after the prediction is made, we first trace back to the most important contexts for each target nodes by attention weights, $\alpha_u^i, i \in [1, L_u]$. Then, for each context node $c_u^i$, we further trace back to its pair interaction scores, $\eta(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j), j \in [1, L_v]$ to obtain the most influential interaction pairs, $\{(c_u^i, c_v^j), \eta(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j) > \epsilon\}$ as the explanations where $\epsilon$ is a pre-defined threshold.

## 4.3 Pair-centric Context Interaction

In contrast to the *node-centric context interaction*, as shown in Figure 4 (b), we also propose a new approach of the pair-centric perspective for modeling the pairwise prediction in graphs by the *pair-centric context interaction*. Instead of aggregating context features for each target node individually, ConPI-pair model directly works on context pairs for the final prediction. Our motivation is that to infer the pairwise relationships between two nodes, we are encouraged to model the pairwise context interactions by the nature of the task. In other words, the relationships between contexts could be more indicative for predicting the relationships for their target nodes, as shown in Figure 1 in the Introduction. Next, we introduce the details of our ConPI-pair model.

Same as ConPI-node model, we have two neighbor sets, $\mathbf{\Psi}_u$ and $\mathbf{\Psi}_v$. As shown in Figure 2 (b), the essential idea of the ConPI-pair model is to formulate a representation for each pair of context nodes, and them aggregate all pair for the final prediction. For simplicity, we encode the pair representation as a compositional function for its given two node vectors, $(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j)$ as:

$$g_p(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j) = f_p([\boldsymbol{\nu}_i; \boldsymbol{\nu}_j; \boldsymbol{\nu}_i \oplus \boldsymbol{\nu}_j]) \tag{6}$$

where $\oplus$ is the element-wise product for two vectors, $[;]$ represents vector concatenation. $f_p(\cdot)$ is a fully-connected network that produces a low-dimensional vector for representing the pair. Such a pair representation will help capture the relationship for a pair of

nodes during the pairwise prediction task. Next, we aggregate all pairs with an attention-based mechanism for the final prediction:

$$z = \sum_{i}^{L_u} \sum_{j}^{L_v} \beta_{ij} \cdot g_p(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j) \tag{7}$$

$$\beta_{ij} = \frac{\exp(S_p(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j))}{\sum_{m=1}^{L_u} \sum_{n=1}^{L_v} \exp(S_p(\boldsymbol{\nu}_u^m, \boldsymbol{\nu}_v^n))} \tag{8}$$

where $S_p$ is a similarity function for each pair of context vectors (e.g., bilinear similarity). $z$ will be used for making the final prediction with a classification layer.

**Interpretability.** The attention distribution, $\beta_{ij}$, in Eqn.8 is the normalized pairwise interaction scores and will be used to retrieve the most important pairs for explanations after the prediction. And, the explanation question that the ConPI-pair model can answer is which important context interactions lead to current prediction. Apparently, for interpretability, ConPI-pair selects context pairs in a one-step process, which is more straightforward and easier to be understood by users than ConPI-node model.

Though the model architecture for our ConPI-pair is relatively simple, the perspective of modeling pairs for pairwise prediction is new, and we also show the superior performance of ConPI-pair model with comprehensive experiments later. As one of our main contributions, shifting the perspective from node-centric to pair-centric model provides us more promising directions to go for modeling pairwise prediction tasks. Next, we show a new type of pair embeddings, which can be naturally injected into ConPI-pair.

## 4.4 Pretraining and Injecting Pair Embedding

To expand the idea of modeling pair interactions directly for pairwise prediction tasks, we further propose to leverage the graph structure to pre-train embeddings of node pairs, and inject them back to our ConPI-pair model with fewer parameters. Our intuition is that the pre-trained pair embedding would incorporate more prior knowledge for node pairs than combining their node vectors straightforwardly as in previous ConPI-pair model. In this part, we introduce learning general-purpose embeddings for node pairs based on the graph structure to facilitate context pair interactions.

**Pair Representation Learning.** Following the distributional hypothesis [25], we encourage the pair embeddings of two nodes to be similar if they are likely to co-occur with similar context nodes. Given a pair of node $(u, v)$ and a context node $c$, we first embed them to vectors, $\boldsymbol{\nu}_u, \boldsymbol{\nu}_v, \boldsymbol{\nu}_c$ via two embedding matrix, $\boldsymbol{E}_p$ for $u, v$, $\boldsymbol{E}_c$ for $c$. Note that the node pairs are not necessarily to be connected in the graph, and it would be $O(|\mathcal{V}|^2)$ complexity to build embeddings for all pairs, which is computationally expensive. Thus, we define a deep compositional function taking the input as representations for nodes, $(u, v)$ to generate a fixed-length vector for the pair:

$$\mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v) = f_{\text{MLP}}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v, \boldsymbol{\nu}_u \oplus \boldsymbol{\nu}_v) \tag{9}$$

where $f_{\text{MLP}}$ is a fully-connected multi-layer neural networks with the same input as Eqn.6. With such a compositional function, we are able to generate embeddings for any pairs in the graph efficiently.

After encoding the node pair $(u, v)$ and its context nodes $c$, we can define the following conditional probability representing how likely the context is observed around the pair [34] and optimize it with a negative log-likelihood objective function:

$$p(c|(u, v)) = \frac{\exp(\boldsymbol{\nu}_c^T \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v))}{\sum_{c' \in \mathcal{V}} \exp(\boldsymbol{\nu}_{c'}{}^T \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v))} \tag{10}$$

Minimizing the negative log-likelihood for $p(c|(u, v))$ would be computationally costly, which leads to the adoption of popular negative sampling technique [21] for efficient training. The goal of negative sampling is to encourage the similarity between the pair and context if they appear together (co-occur) in the graph, and the dissimilarity between the pair and randomly sampled contexts. That is, for a valid pair-context sample, we fix the pair $(u, v)$ and randomly sample contexts $c^N$ as the distractor.

However, pair embeddings are different from node embeddings in which a pair consists of two nodes that are both changeable. In other words, we are able to fix the left pair-node $u$ and context $c$, sample the right pair-node $v^N$, and vice versa. By doing so, we can expose the pair embedding to noisier environments to make it more robust. Therefore, we introduce the objective function of the negative sampling for training our pair embeddings as follows:

$$\begin{aligned}\mathcal{L}_{\text{pair}} = \log \sigma(\boldsymbol{\nu}_c^T \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v)) + \sum_{i=1}^{K_c} \log \sigma(-\boldsymbol{\nu}_{c_i}^{N\,T} \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v)) \\ + \sum_{j=1}^{K_u} \log \sigma(-\boldsymbol{\nu}_c^T \cdot \mathcal{P}(\boldsymbol{\nu}_{u_j}^N, \boldsymbol{\nu}_v)) + \sum_{k=1}^{K_v} \log \sigma(-\boldsymbol{\nu}_c^T \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_{v_k}^N))\end{aligned} \tag{11}$$

where $K_c, K_u, K_v$ are the number of random samples for contexts, left pair-nodes, right pair-nodes, and $\sigma$ here is the sigmoid function.

Our objective function is similar to the multivariate objective function in pair2vec [14], but differs in that we optimize node pair embeddings with graph structure and represent contexts as nodes, while they try to encode a short span of words as the context to learn word pair embeddings from a text corpus.

**Pair Representation Injection.** Once the node pair embeddings are learned, we are ready to inject them back into our ConPI-pair model for the pairwise prediction task. We keep the injection as simple and generalizable as possible to show the effectiveness of our pair embeddings. We simply replace the pair encoder function $g_p(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j)$ in Eqn.6 by pre-trained pair embedding vectors $\mathcal{P}(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j)$ in Eqn.9 and keep the rest parts as the same in ConPI-pair model. With such a simple injection, we do not need the fully-connected network in the pair encoder (Eqn.6), and thus, reduce the number of parameters compared with ConPI-pair model. Note that although we can produce a pair embedding for the pair of target nodes and use it for the final prediction, directly making a prediction based on the pair embedding does not involve the context interaction, which is not the focus of this paper.

## 4.5 Model Optimization

**Pairwise Prediction Task.** In this paper, we consider a setting of binary pairwise prediction to test our framework for simplicity and define a binary cross entropy loss as follows:

$$\begin{aligned}\mathcal{L}_{pred} = -\sum_{i=1}^{M} (y_i \cdot \log(p(r|u_i, v_i)) \\ + (1 - y_i) \cdot \log(1 - p(r|u_i, v_i)))\end{aligned} \tag{12}$$

where $M$ is the total number of samples, $y_i$ is the ground-truth label indicating whether $u_i, v_i$ holds a certain relation. $p(r|u_i, v_i)$ is a binary classifier with the sigmoid function with the input as

interaction-aware node representations in ConPI-node or aggregated pair representations in ConPI-pair.

**Complexity Analysis.** Our pairwise context interaction takes $O(L^2)$ computations ($L$ is the maximal size of contexts), which could be costly when $L$ is very large. In our preliminary experiments, we observe that the performance does not further improve when we set $L$ at a reasonably large number (e.g., 100). Thus, we uniformly sample a limited-size set of contexts for each node to perform efficient interactions. For future work, we could select the contexts more smartly, e.g., adopting some graph sparsification techniques [47], to further decrease the complexity without significantly sacrificing the performance.

**Training Pair Embeddings.** To optimize the objective function (Eqn.11) for pair embeddings with negative sampling, we need to obtain positive samples and randomly sample negatives ones. Our training algorithm adopts the random walks to provide contexts for pairs of nodes. Specifically, given a sequence of nodes generated by random walks [10, 25], we define a pair window to sample node pairs and then, for each pair, we define a context window to the left and right of the pair as well as all nodes in the middle of it to sample positive contexts. We randomly sample contexts, left pair-nodes, right pair-nodes for training the Eqn.11.

# 5 EXPERIMENTS

In this section, to show the generalizability of our framework, we test ConPI on two types of pairwise prediction tasks , link prediction and (weakly-supervised) relation prediction.

## 5.1 Link Prediction Task

The link prediction task is a well-studied task in graph domain, which is to predict whether two nodes in a graph have a link. It has many meaningful applications in various kinds of graphs, e.g., predicting friendship links in social networks [20], predicting author identification in citation networks [22], etc.

*5.1.1* ***Datasets.*** We collect a number of commonly-used publicly-available real-world graphs for link prediction shown in Table 1.

- **PPI**: A Protein-Protein Interaction (PPI) graph for Home Sapiens that is used in Grover and Leskovec [10]. Edges represent the interaction relationships between proteins.
- **Pubmed**: A citation network for papers from PubMed used in Kipf and Welling [15]. Edges represent the citation relationship between publications.
- **BlogCatalog**: A social network for bloggers from the BlogCatalog website that is used on Grover and Leskovec [10]. Edges represent friendship links between bloggers.
- **DrugBank DDI**: A Drug-Drug Interaction graph crawled from DrugBank that is used in Yue et al. [44]. Edges represent the interaction relationships between drugs.

*5.1.2* ***Experimental Setup.*** To evaluate all methods fairly, we follow the experiment strategy that is commonly used by previous methods [8, 10, 45]. We first randomly split all links of the original graph into training, validation, and testing sets with the requirement of keeping the graph in training set connected. The training graph is used for extracting features, training embeddings, and obtaining context sets. For prediction tasks, we also randomly sample the equal number of non-existent links as negative samples. We split all edges to train/validation/test sets with a ratio of 70/15/15.

**Table 1: Statistics of datasets for link prediction task.**

| Dataset | # Nodes | # Edges | Avg. # Degrees |
|---|---|---|---|
| PPI | 3,890 | 38,739 | 19.9172 |
| Pubmed | 19,717 | 44,327 | 4.4963 |
| BlogCatalog | 10,312 | 333,983 | 64.7756 |
| DrugBank DDI | 2,191 | 242,027 | 220.9283 |

*5.1.3* ***Compared Methods.*** We compare our models with the following three types of baselines for the link prediction tasks.

**Graph feature-based methods.** We test the traditional graph feature-based methods for link prediction tasks [2, 20, 45], which calculate some heuristics based on the neighborhood of nodes in the graph. We consider: *Common Neighbors* that calculates the number of shared neighborhoods, *Jaccard Coefficient* that measures the Jaccard similarity between two neighbor sets, and *Adamic Adar* for the number of shared links between two nodes.

**Embedding-based methods.** We consider several state-of-the-art embedding-based methods, which take the training graph as input and produce an embedding for every node in the graph: *Laplacian* eigenmaps [4] is a matrix factorization method that factorizes the graph Laplacian matrix to the lowest eigenvectors as embeddings; *DeepWalk* [25] is a random walk-based method that learns node embeddings with the skip-gram algorithm on random walks generated from the graph. *LINE* [34] utilizes the first and second order proximity to optimize node embeddings with edge sampling. *node2vec* [10] learns the embedding by performing biased random walks on the training graph by minimizing the skip-gram objective.

**Graph Neural Networks-based methods.** We compare against a popular GNN-based method, *GAE* [15], which uses graph auto-encoders model with graph convolutional network encoder to learn efficient node embeddings from the adjacent matrix.

In the testing phase, for all embedding and GNN based baselines, we follow the procedure in previous work [10, 45] to learn a classifier based on positive training samples and an equal number of negative training samples, and test the classifier on the testing set. We train a binary logistic regression model on the Hadamard product of node embeddings, using the scikit-learn library [23].

**Variants of our ConPI framework.** *ConPI-node* model makes the prediction based on the interaction-aware node representations of target nodes. *ConPI*-pair model makes the prediction based on the aggregated pair representations with an attention-based module. *ConPI-pair-emb* is one variant of ConPI-pair model that simply replaces the pair encoder function with pre-trained pair embedding and has fewer parameters than ConPI-pair.

Note that in this paper, we focus on modeling context interaction based on graph structure, and thus, we mainly consider baseline methods that leverage graph structure for a fair comparison, e.g., we do not feed node attribute features in the GAE model.

**Combination with more advanced GNN methods.** We compare ConPI mainly with various embedding-based methods to show its effectiveness by adopting node embeddings (e.g., LINE) as feature vectors. In fact, our framework can be built on top of any graph representation methods, including those more complex GNN methods, such as GCN[16], GAT[37], HGCN[7]. Specifically, we can replace current feature vectors by these advanced GNN methods as more powerful feature encoders for our ConPI framework. We

**Table 2: Results on Link Prediction Task**

| Method | PPI | | | Pubmed | | | BlogCatalog | | | DrugBank DDI | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AUC | AP | F1 | AUC | AP | F1 | AUC | AP | F1 | AUC | AP | F1 |
| Common Neighbors | 0.8290 | 0.8250 | - | 0.6317 | 0.6313 | - | 0.9396 | 0.9357 | - | 0.9440 | 0.9451 | - |
| Jaccard Coefficient | 0.8099 | 0.7835 | - | 0.6316 | 0.6292 | - | 0.7748 | 0.7130 | - | 0.9299 | 0.9043 | - |
| Adamic Adar | 0.8325 | 0.8391 | - | 0.6318 | 0.6317 | - | 0.9444 | 0.9456 | - | 0.9459 | 0.9478 | - |
| Laplacian [4] | 0.5355 | 0.5742 | 0.3269 | 0.6976 | 0.7657 | 0.5616 | 0.7157 | 0.7711 | 0.6938 | 0.6828 | 0.7862 | 0.6590 |
| DeepWalk [25] | 0.7919 | 0.8079 | 0.4630 | 0.9131 | 0.9323 | 0.6643 | 0.8051 | 0.8007 | 0.6909 | 0.9235 | 0.9117 | 0.8377 |
| LINE [34] | 0.8153 | 0.8364 | 0.7205 | 0.8107 | 0.8318 | 0.6996 | 0.9153 | 0.9143 | 0.8382 | 0.9222 | 0.9169 | 0.8449 |
| node2vec [10] | 0.7513 | 0.7625 | 0.4116 | 0.9230 | 0.9343 | 0.6572 | 0.6400 | 0.5873 | 0.5527 | 0.8949 | 0.8884 | 0.8030 |
| GAE [15] | 0.7056 | 0.6029 | 0.7203 | 0.7904 | 0.8250 | 0.7162 | 0.7547 | 0.6496 | 0.6968 | 0.7516 | 0.7170 | 0.7500 |
| ConPI-node | 0.7970 | 0.7748 | 0.7332 | 0.8550 | 0.8140 | 0.7439 | 0.9419 | 0.9307 | 0.8745 | 0.9395 | 0.9334 | 0.8641 |
| ConPI-pair-emb | 0.8450 | 0.8291 | 0.7009 | 0.8736 | 0.8987 | 0.7542 | 0.9388 | 0.9177 | 0.8449 | 0.9663 | 0.9644 | 0.9003 |
| ConPI-pair | **0.9004** | **0.8986** | **0.8208** | **0.9375** | **0.9362** | **0.8437** | **0.9684** | **0.9658** | **0.9117** | **0.9842** | **0.9823** | **0.9364** |

do not include them as baselines in this paper as they suffer from the same low interpretability problem as other baselines and it is suffice for us to demonstrate ConPI's interpretability using simpler embedding methods. We leave the exploration of combining our framework with more complex GNN methods to future work.

*5.1.4* ***Experimental Results and Analysis.*** The main results for comparing all methods in link prediction task are shown in Table 2. For graph feature-based baselines, we report Area Under ROC curve (AUC) and Average Precision (AP) scores on their ranking of positive and negative samples in the testing set. For the rest of all classification methods, we report AUC, AP, and F1 scores in the testing set for their performance comparison.

We first compare with graph feature-based methods. Similar with our ConPI framework, those heuristic features also explicitly utilize neighborhood information but based on deterministic rules. As shown in Table 2, such heuristic features obtain very competitive performance for the link prediction task (e.g., Common Neighbors feature gets 0.9396 AUC score in BlogCatalog, 0.9440 AUC score in DrugBank DDI graph.), which shows the necessity of explicitly modeling neighborhood nodes for link prediction. However, heuristic feature baselines do not perform consistently in those four datasets indicating their bad generalizability. We observe that our ConPI-pair model outperforms them in a large margin. This is because ConPI incorporates the semantic representation of context pairs and models their interactions with more parameters.

Then, we compare with the embedding and GNN based methods. The ConPI-node model outperforms all baselines in BlogCatalog, and most of the baselines in other datasets, which indicates that making the pairwise prediction based on two node embeddings learned independently cannot fully capture the pairwise information. Furthermore, we can observe that our ConPI-pair model consistently outperforms all these methods in a relatively large margin, which is the core contribution in this paper. These results confirm our hypothesis that directly modeling pairwise interactions fits more the nature of pairwise prediction task.

Finally, we also compare variants in the ConPI framework. All models that directly model pair representations outperform the node model that learns (interaction-aware) node representation by a large margin. Note that ConPI-pair-emb model with injecting pre-trained pair embeddings has the same number of parameters with ConPI-node, and it beats the ConPI-node model in most datasets. This shows the effectiveness of our pre-trained pair embeddings, which encourages us to further explore the pair representations in the future. All in all, these results indicate that for pairwise prediction tasks, it is necessary to model the pairwise interaction and directly design pair representations for the prediction.

## 5.2 Relation Prediction Task

Relation prediction is another well-known pairwise prediction task that aims to predict relation labels between entities, and can be leveraged to facilitate many downstream applications dealing with graphs, e.g., knowledge base completion [41], hypernymy detection [32], synonyms discovery [27]. In contrast to link prediction task, this task considers the setting where the labels are not connected links from the graph, and the graph is mainly used to offer the distributionally semantic information for the task. Same as the previous setting in the link prediction task, we do not consider other task-specific information (e.g., text patterns), but only take as input the graph structure for evaluating all methods fairly.

*5.2.1* ***Datasets.*** We consider medical relation prediction tasks that infers relations between medical terms based on a medical term-term co-occurrence graph. We employ a publicly available dataset from Finlayson et al. [9] in which medical terms are extracted from 20 million clinical notes, and the edges are weighted by the co-occurrence counts based on the frequency that two terms co-occur in a temporal bin. We select the 1-day per-bin graph that has the most number of terms/nodes (see more details of the original graph in Finlayson et al. [9]). For preprocessing, we convert the co-occurrence counts into PPMI value, subsample the graph to remove meaningless terms [21], and filter edges with a PPMI value less than 2. Finally, we obtain a large medical term-term co-occurrence graph with 48,651 nodes and 1,659,249 edges.

*5.2.2* ***Experimental Setup.*** For relations, we select Clinically Associated With (Caw) and IsA relations that are two of the most common relations in the dataset. The first one indicates a clinically salient relationship between medical terms, while the second one represents a hierarchical relationship meaning that the first term has a more specific meaning than the second one. To obtain the labels on a large scale, we follow the procedure of *weakly supervision* for relation extraction to automatically retrieve supervisions from knowledge bases (KBs) [27, 28]. Specifically, we first collect positive samples between concepts in the KB where each concept has a

**Table 3: Results on Relation Prediction Task**

| Method | CAW | | IsA | |
|---|---|---|---|---|
| | AUC | F1 | AUC | F1 |
| Common Neighbors | 0.5631 | - | 0.7084 | - |
| Jaccard Coefficient | 0.5632 | - | 0.7109 | - |
| Adamic Adar | 0.5634 | - | 0.7097 | - |
| Laplacian [4] | 0.5547 | 0.5163 | 0.5452 | 0.4896 |
| DeepWalk [25] | 0.7309 | 0.6512 | 0.8303 | 0.7292 |
| word2vec [21] | 0.6842 | 0.5855 | 0.8083 | 0.6845 |
| LINE [34] | 0.7426 | 0.6746 | 0.8209 | 0.7336 |
| node2vec [10] | 0.7458 | 0.6653 | 0.8443 | 0.7477 |
| ConPI-node | 0.7852 | 0.7072 | 0.8434 | 0.7556 |
| ConPI-pair-emb | 0.8177 | 0.7401 | 0.8807 | 0.7917 |
| ConPI-pair | **0.8807** | **0.8085** | **0.8945** | **0.8173** |

number of string mentions (terms) as $C_A = \{t_i\}_{i=1}^{m}, C_B = \{t_j\}_{j=1}^{n}$. Then we obtain the positive labels between terms as $\{(t_i, t_j), t_i \in C_A, t_j \in C_B\}$. The term-to-concept mapping is provided by the dataset [9], and we use UMLS (Unified Medical Language System) as the KB. Finally, we have 132,716 positive samples for CAW relation, 85,283 positive samples for IsA relation, and we sample an equal number of negative samples by randomly pairing one argument of the positive pair with a random term for the classification task. Then we split each dataset into train/validation/test sets with a ratio of 70/15/15. We use the full co-occurrence graph as the input for all methods, which is utilized for training all embedding methods and for extracting the graph features.

*5.2.3* ***Compared Methods.*** We keep most of baseline methods the same as the link prediction task. We remove the GAE method as it cannot process a huge graph like our co-occurrence graph with the out-of-memory (OOM) issue. We also compare another representative embedding method in NLP domain, *word2vec* [21] by conducting SVD over the shifted PPMI co-occurrence matrix [18].

*5.2.4* ***Results and Analysis.*** The results of relation prediction task are shown in Table 3. We observe similar performance comparison as what we have observed in link prediction task. Additionally, compared with IsA relation, the CAW is relatively difficult to classify as it requires the understanding of the complex semantic of *clinical association*. Even though our ConPI framework beats all baseline methods, and ConPI-pair model obtains the best performance among all methods. Also, in IsA relation, we observe that the ConPI-pair-emb model gets very competitive performance with ConPI model with fewer parameters.

## 5.3 Interpretability Analysis

There are some recently-proposed interpretability techniques on graphs, such as GraphLIME [13] and GNNExplainer [43], which can provide different types of explanations. Most of them try to analyze well-trained graph models post-hoc and instead, our model makes the prediction directly based on the explanations. To demonstrate ConPI's interpretability for pairwise predictions, we conduct case study by visualizing the interactions scores, i.e., attention weights, calculated between context pairs.

We conduct the case study on the medical term-term co-occurrence graph and choose an easily-understood relation, IsA to interpret.

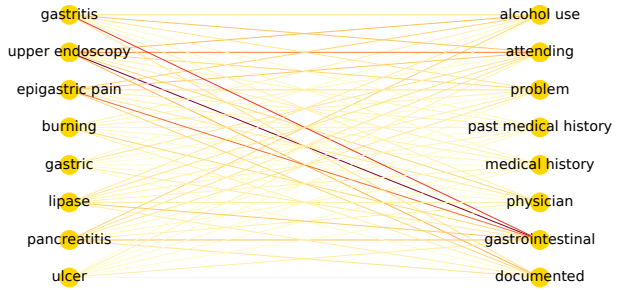



**Figure 3: Interpretability Visualization for our ConPI-pair model (best view with colors). Contexts (left) of "*burning epigastric pain*" interact with contexts (right) of "*pain epigastric*" to make the pairwise prediction. The line color indicates interaction score (the redder, the larger).**

As described in Section 4, the model decision is made based on interaction scores for context pairs and we recover such information to explore whether these scores can faithfully explain the model decision. Due to the limitation of the space, we show a testing-set example of our best model, ConPI-pair, for a correct prediction of "*burning epigastric pain*" is a type of "*pain epigastric*" in Figure 3. The figure visualizes the interactions between two sets of contexts for target nodes. The nodes on left (right) side are contexts for "*burning epigastric pain*" ("*pain epigastric*"). The color of the line indicates the strength of interaction (the redder, the stronger).

There are two major findings from Figure 3. Firstly, by observing the nodes, we can see that our ConPI-pair model nicely downweighs contexts that are irrelevant for the pairwise prediction, such as "problem", "medical history", etc. Secondly, by observing the pairs, the model successfully highlights three pairs, "gastritis", "upper endoscopy", "epigastric pain" with "gastrointestinal", which are strongly relevant with the pairwise prediction and can be treated as explanations. We also see that some interaction scores are not perfect yet, for example, pairs with a meaningless node, "attendig", get relatively high scores. To further enhance the interpretability of our model, we may adopt some post-hoc explainable techniques to further prune some untrustworthy explanations and we leave this to future work. Nevertheless, based on the case study, we can clearly see that our ConPI model can provide informative and faithful explanations for the pairwise prediction, and such interpretability would be helpful to convince users to better trust the model decision, especially on those high-stake domains (e.g., medicine, finance, etc).

## 6 CONCLUSION AND FUTURE WORK

In this paper, we study modeling context pair interactions for pairwise prediction tasks on graphs to capture the pairwise relationships between nodes better, and provide a certain amount of interpretability by selecting influential interaction links. We propose a general framework ConPI with node-centric and pair-centric perspectives and further propose a new pair-centric context interaction model, ConPI-pair, to first formulate a pair representation and then attentively aggregate all pairs for the final prediction. To capture the node pairwise relationships in embedding space, we also propose a new type of pair embeddings in homogeneous graphs and show how to inject them back to the ConPI-pair model.

We demonstrate the effectiveness of our framework in two pairwise prediction tasks across a variety of real-world datasets as well as the model interpretability by the case study.

There are many promising directions to explore for explicitly modeling context interactions. First, more sophisticated interaction mechanisms can be developed by considering high-order relationships among contexts. Secondly, combining such interaction mechanisms with existing embedding learning methods would also be an inspiring direction. Last but not lease, the power of pair embedding can be further extended into a wide range of applications and the interplay between it with node embedding would be interesting.

## ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their helpful comments. This research was sponsored in part by the Patient-Centered Outcomes Research Institute Funding ME-2017C1-6413, the Army Research Office under cooperative agreements W911NF-17-1-0412, NSF Grant IIS1815674, NSF CAREER #1942980, and Ohio Supercomputer Center [6]. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S.Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice herein.

## REFERENCES

[1] Sami Abu-El-Haija, Bryan Perozzi, and Rami Al-Rfou. 2017. Learning edge representations via low-rank asymmetric projections. In *CIKM*.
[2] Lada A Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social networks* 25, 3 (2003), 211–230.
[3] Sambaran Bandyopadhyay, Anirban Biswas, MN Murty, and Ramasuri Narayanam. 2019. Beyond Node Embedding: A Direct Unsupervised Edge Representation Framework for Homogeneous Networks. *arXiv preprint arXiv:1912.05140* (2019).
[4] Mikhail Belkin and Partha Niyogi. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*. 585–591.
[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep neural networks for learning graph representations. In *AAAI*.
[6] Ohio Supercomputer Center. 1987. Ohio Supercomputer Center. http://osc.edu/ark:/19495/f5s1ph73
[7] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In *NeurIPs*. 4869–4880.
[8] Alessandro Epasto and Bryan Perozzi. 2019. Is a single embedding enough? learning node representations that capture multiple social contexts. In *WWW*.
[9] Samuel G Finlayson, Paea LePendu, and Nigam H Shah. 2014. Building the graph of medicine from millions of clinical narratives. *Scientific data* 1 (2014), 140032.
[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*.
[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*.
[12] Weihua Hu*, Bowen Liu*, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2020. Strategies for Pre-training Graph Neural Networks. In *ICLR*.
[13] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. 2020. GraphLIME: Local interpretable model explanations for graph neural networks. *arXiv preprint arXiv:2001.06216* (2020).
[14] Mandar Joshi, Eunsol Choi, Omer Levy, Daniel Weld, and Luke Zettlemoyer. 2019. pair2vec: Compositional Word-Pair Embeddings for Cross-Sentence Inference. In *NAACL*.
[15] Thomas N Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. *NIPS Workshop on Bayesian Deep Learning* (2016).
[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.
[17] István A Kovács, Katja Luck, Kerstin Spirohn, Yang Wang, Carl Pollis, Sadie Schlabach, Wenting Bian, Dae-Kyum Kim, Nishka Kishore, Tong Hao, et al. 2019. Network-based prediction of protein interactions. *Nature communications* 10, 1 (2019), 1–8.
[18] Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *NeurIPS*.
[19] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. 2019. Graph matching networks for learning the similarity of graph structured objects. In *ICML*.
[20] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58, 7 (2007), 1019–1031.
[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*.
[22] Chanyoung Park, Donghyun Kim, Qi Zhu, Jiawei Han, and Hwanjo Yu. 2019. Task-Guided Pair Embedding in Heterogeneous Network. In *CIKM*.
[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[24] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.
[25] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*.
[26] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *WSDM*.
[27] Meng Qu, Xiang Ren, and Jiawei Han. 2017. Automatic synonym discovery with knowledge bases. In *KDD*.
[28] Meng Qu, Xiang Ren, Yu Zhang, and Jiawei Han. 2018. Weakly-supervised relation extraction by pattern-enhanced embedding learning. In *WWW*.
[29] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *KDD*.
[30] Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609* (2016).
[31] Yu Shi, Qi Zhu, Fang Guo, Chao Zhang, and Jiawei Han. 2018. Easing embedding learning by comprehensive transcription of heterogeneous information networks. In *KDD*.
[32] Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving Hypernymy Detection with an Integrated Path-based and Distributional Method. In *ACL*.
[33] Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020. Knowledge Graph Alignment Network with Gated Multi-hop Neighborhood Aggregation. In *AAAI*.
[34] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*.
[35] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. Verse: Versatile graph embeddings from similarity measures. In *WWW*.
[36] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. Cane: Context-aware network embedding for relation modeling. In *ACL*.
[37] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.
[38] Janu Verma, Srishti Gupta, Debdoot Mukherjee, and Tanmoy Chakraborty. 2019. Heterogeneous Edge Embedding for Friend Recommendation. In *ECIR*.
[39] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *KDD*.
[40] Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou. 2015. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences* 58, 1 (2015), 1–38.
[41] Yanjie Wang, Rainer Gemulla, and Hui Li. 2018. On multi-relational link prediction with bilinear models. In *AAAI*.
[42] Zhen Wang, Xiang Yue, Soheil Moosavinasab, Yungui Huang, Simon Lin, and Huan Sun. 2019. SurfCon: Synonym Discovery on Privacy-Aware Clinical Data. In *KDD*.
[43] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in neural information processing systems*. 9244–9255.
[44] Xiang Yue, Zhen Wang, Jingong Huang, Srinivasan Parthasarathy, Soheil Moosavinasab, Yungui Huang, Simon Lin, Wen Zhang, Ping Zhang, and Huan Sun. 2019. Graph Embedding on Biomedical Networks: Methods, Applications, and Evaluations. *Bioinformatics (Oxford, England)* (2019).
[45] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *NeurIPS*.
[46] Zhen Zhang, Hongxia Yang, Jiajun Bu, Sheng Zhou, Pinggang Yu, Jianwei Zhang, Martin Ester, and Can Wang. 2018. ANRL: Attributed Network Representation Learning via Deep Neural Networks.. In *IJCAI*.
[47] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. 2020. Robust Graph Representation Learning via Neural Sparsification. In *ICLR*.
[48] Yang Zhou, Sixing Wu, Chao Jiang, Zijie Zhang, Dejing Dou, Ruoming Jin, and Pengwei Wang. 2018. Density-adaptive local edge representation learning with generative adversarial network multi-label edge classification. In *ICDM*.