Abstract

Missing node attributes is a common problem in real-world graphs. Graph neural networks have been demonstrated powerful in graph representation learning, however, they rely heavily on the completeness of graph information. Few of them consider the incomplete node attributes, which can bring great damage to the performance in practice. In this paper, we propose an innovative node representation learning framework, Wasserstein graph diffusion (WGD), to mitigate the problem. Instead of feature imputation, our method directly learns node representations from the missing-attribute graphs. Specifically, we extend the message passing schema in general graph neural networks to a Wasserstein space derived from the decomposition of attribute matrices. We test WGD in node classification tasks under two settings: missing whole attributes on some nodes and missing only partial attributes on all nodes. In addition, we find WGD is suitable to recover missing values and adapt it to tackle matrix completion problems with graphs of users and items. Experimental results on both tasks demonstrate the superiority of our method.

1. Introduction

Many real-world networks are attributed networks, where nodes are not only connected with other nodes, but also associated with features, e.g., social network users with profiles or keywords showing interests, Internet Web pages with content information, etc. However, observed node attributes are usually partially absent and some are even entirely inaccessible. For instance, in the case of social networks like Facebook and Twitter, users tend to selectively (or entirely not) provide their personal information for privacy concerns. In this paper, we focus on such missing-attribute graphs and mainly consider two missing cases: 1) Entirely missing: missing entire attributes on some nodes, 2) Partially missing: missing partial attributes on all nodes.

Learning node representations underlie various downstream graph-based learning tasks and have attracted much attention (Perozzi et al., 2014; Grover & Leskovec, 2016; Pimentel et al., 2017; Duarte et al., 2019). In attributed networks, node representations are expected to express node-attributed and graph-structured information. Random walk based graph embedding approaches (Perozzi et al., 2014; Grover & Leskovec, 2016) exploit graph structure information to preserve pre-specified node similarities in the embedding space and have proven successful in various applications based on plain graphs. But they can not take informative node attributes into account. Message passing schema (Gilmer et al., 2017) based methods, such as many graph neural networks, aggregate information from neighborhoods and allow us to incorporate attribute and structure information effectively. On missing-attribute graphs, a message passing schema, however, have limited compensation for the incompleteness of information by collecting observed information from neighbors.

As most existing methods can’t directly make a high-quality node representation on missing-attribute graphs, it is worth considering how to make the most of the limited incomplete node attributes and generate powerful node representations efficiently. A straightforward strategy would be to leverage matrix imputation techniques (Troyanskaya et al., 2001; Hastie et al., 2015) to estimate missing values before learning. But it is hard to obtain high-quality estimation with extremely rare observations. In this paper, we propose an ingenious missing-attribute graph learning framework based on a generalized message passing schema called Wasserstein graph diffusion (WGD) to mitigate this problem. The general architecture is shown in Figure 1. Our method does not rely on an accurate imputation method but directly learn node representations from incomplete data. The key idea is to implicitly enrich node latent information through the WGD process and preserve node differentiation at the same time.

There are several keys that free us from imputing missing attributes and the first one is adopting matrix factorization techniques. WGD follows the low-rank matrix assump-
tion in most matrix imputation algorithms. We ingeniously transform node features into discrete distributions in a lower-dimensional Wasserstein space instead of Euclidean space and propose an innovative message passing schema, the Wasserstein graph diffusion. Here Wasserstein Barycenter is used to update node distributional representations by aggregating distributions from neighbors. As our experiments will show, compared with common aggregation approaches like mean-aggregation, WGD indeed reduces information distortion caused by attributes missing and greatly boosts the performance. Since no parameter is introduced during the WGD process, we can effortlessly pull node distributions back to the original Euclidean space and generate new Euclidean representations. In fact, if we combine WGD with an additional multi-layer perceptron layer, the framework can also be seen as a new graph neural network.

After learning the node representations, our framework can be applied to various downstream tasks. The first task is node classification on graph with incomplete attributes. To comprehensively investigate the representation ability, we examine our framework on node classification concerning two missing-attribute cases: partially missing and entire missing.

In addition, although our framework is not originally designed for predicting missing attribute values, it can be naturally adapted for the matrix completion task by adding reconstruction constrains in the step of Euclidean representations generation. Compared with the state of art matrix completion algorithms (Rao et al., 2015; Monti et al., 2017; Hartford et al., 2018; Zhang & Chen, 2020), our method relies on much fewer parameters and has competitive performance.

Contributions. Overall, our contribution can be summarized as follows: 1. We develop a novel non-parametric missing-attribute graph learning framework called WGD which allows us to elegantly generate powerful node representations without data imputation. Our performance is far surpassing that of baselines. 2. We develop a generalized message passing schema in Wasserstein space which can reduce information distortion and improve the distinguishable ability of node representations. 3. We extend WGD on multi-graph and adapt it for matrix completion with content of users and items and achieve competitive results of state of the art algorithms with much fewer parameters.

2. Background and Related Work

Graph representation learning. In this paper, we focus on learning node representations on attributed graphs. There are many effective graph embedding approaches, such as DeepWalk (Bojchevski & Gümennann, 2018), node2vec (Grover & Leskovec, 2016), GenVetor (Duarte et al., 2019), which embed nodes into a lower-dimension Euclidean space and preserve graph structure while most of them disregard node informative attributes. So far, there is little attention paid to attribute information (Yang et al., 2015; Gao & Huang, 2018; Hong et al., 2019). The advent of graph neural networks (Bruna et al., 2014; Kipf & Welling, 2017; Hamilton et al., 2017; Veličković et al., 2017; Gilmer et al., 2017; Klicpera et al., 2019a; b) fills the gap to some extent, by defining graph convolutional operations in spectral domain or aggregation schemes in spatial domain. Although they are successfully applied in many graph-based tasks such as node classification, they highly rely on the completeness and adequacy of attribute information.

Machine learning with missing data. To handle missing data, most machine learning methods rely on data imputation. There is a variety of missing value imputation (MVI) techniques such as mean-filling, KNN imputation (Troyanskaya et al., 2001), softimpute (Hastie et al., 2015) with SVD decomposition, multivariate imputation (Van Buuren, 2007; Buuren & Groothuis-Oudshoorn, 2010). Also, many deep learning methods are proposed to perform the imputation tasks (Gondara & Wang, 2017; Yoon et al., 2018; Spinelli et al., 2020). The “imputing before learning” strategy has an important limitation: the performance of models is inherently constrained by the reconstruction ability of the used imputation methods. However, these imputation methods would not always work especially in the extreme missing cases.

Recently, some advanced models have been developed to directly handle missing data targeting at specified tasks. GRAPE (You et al., 2020) tackles missing data problems for label prediction and feature imputation. Unlike our work, the missing data is not originally on the graph nodes, but GRAPE represents their two tasks as graph-based problems by leveraging a created bipartite graph. Another recent work, SAT (Chen et al., 2020), models link prediction and node attribute imputation on missing-attribute graphs with shared-latent space assumption. Different from these works, our WGD is a graph embedding framework that focus on learning node representations with incomplete attribute matrix as input without imputation processing; and it can be adapted to various downstream graph-based tasks.

3. Wasserstein graph diffusion (WGD) framework

In this paper, we propose a Wasserstein graph diffusion (WGD) framework on missing-attribute graphs to generate node representations directly. The WGD framework (depicted in Fig 1) consists of three main components: 1. space transformation - to embed nodes into a Wasserstein space; 2. Wasserstein graph diffusion - to update node distributional representations; 3. inverse transformation - to pull nodes
Figure 1. In the WGD framework, we adapt an innovative message-passing schema based on the generalized MEAN aggregation function, called Barycenter Update, in a discrete Wasserstein space (right). To embed nodes into the Wasserstein space, we first derive the principal component matrix $U_k^0$ and basis matrix $V_k$ from the low-rank SVD decomposition on feature matrix $X$ (left). We represent nodes as generalized histograms with rows of $U_k^0$ viewed as frequencies and columns of $V_k$ viewed as bins. Then we can transform these histograms representation to standard discrete distribution as the input of WGD (middle). Through rounds of message passing, WGD generate node distributional embeddings. After that, we pull nodes back to the Euclidean space whose dimension is the same as the original feature space, to generate Euclidean embeddings $X$.

back to the Euclidean space.

3.1. Preliminary: Wasserstein distance

Wasserstein distance is an optimal transport metric, which measures the distance traveled in transporting the mass in one distribution to match another. The $p$-Wasserstein distance between two distributions $\mu$ and $\nu$ over a metric space $\mathcal{X}$ is defined as

$$W_p(\mu, \nu) = \left( \inf_{\Pi(\mu, \nu)} \int_{\mathcal{X} \times \mathcal{X}} d(x, y)^p d\pi(x, y) \right)^{1/p}$$

where $\Pi(\mu, \nu)$ is the set of all possible couplings $\pi$ on $(\mu, \nu)$, $d(x, y)$ is a ground metric on $\mathcal{X}$. In this paper, we take $p = 2$. The Wasserstein space is a metric space that endows probability distributions with the Wasserstein distance.

3.2. The space transformation

Space transformation is the first step of our WGD framework, which attempts to transform node features to discrete distributions endowed with the Wasserstein metric.

Matrix factorization. A common assumption for matrix completion is that the matrix is low-ranked, i.e. the features lie in a smaller subspace, and the missing features can be recovered from this space. Inspired by Alternating Least Square (ALS) Algorithm, a well known missing value imputation method which follows this assumption and uses SVD to factorize matrix into low-ranked submatrices, we first decompose the feature matrix $X \in \mathbb{R}^{n \times m}$ into a principal component matrix $U$, an singular value matrix $\Lambda$, and an orthogonal basis matrix $V$, i.e. $X = U\Lambda V^T$. For dimensionality reduction, we only account for the first $k$ singular vectors of $V$:

$$U_k, \Lambda_k, V_k = \text{SVD}(X, k),$$

where $U_k \in \mathbb{R}^{n \times k}, V_k \in \mathbb{R}^{m \times k}$ and $\Lambda_k \in \mathbb{R}^{k \times k}$. In this way, we map nodes into a low-dimensional principal components space in which $U_k$ is the feature matrix.

Discrete Wasserstein space as embedding space. It is worth noting that such node representations have strong semantic information: each feature dimension is a basis vector from $V_k$. More precisely, we can further express nodes as histograms. In this scenario, we allow the existence of negative frequencies and take principal components (the rows of $U_k$, noted $R(U_k)$) and basis vectors (the columns of $V_k$, noted $C(V_k)$) as frequencies and bins respectively. While most existing aggregation methods fail to take advantage of rich information from $V_k$ stemming from the limited representation capacity of Euclidean embedding. To breakout the limitation, we adopt discrete Wasserstein space as the embedding space in which nodes are represented by discrete distributions.

Transformation formulation. We need to transform nodes to standard discrete distributions leveraging a reversible
positive function \( \varphi \) and the L_1 normalization operation:

\[
\hat{U}_k := \phi(U_k) = L_1 \text{Normalize}(\varphi(U_k)). \tag{2}
\]

Here, \( \varphi \) can be arbitrary reasonable reversible positive functions depending on data. In our experiments, we use the exponential function for implementation. Each row of \( \hat{U}_k \) is a discrete distribution with the form \( u_i = \sum_j a_{ij} \delta_{v_j} \), where \( a_{ij} \) are weights summing to 1, and \( v_j \) is a column of \( V_k \) as well as a support point.

**Distance matrix and Wasserstein distance.** Columns of \( V_k \) form the common support points of nodes noted as \( \text{supp}(U_k) \). Then we define distance metric \( d \) in \( \text{supp}(U_k) \):

\[
d(v_i, v_j) = \| X(v_i - v_j) \|_2^2 = |A_{ii}^2 - A_{jj}^2|. \tag{3}
\]

Here, \( v_i \) and \( v_j \) refer to the \( i \)-th and \( j \)-th support points which are mutual orthogonal unit vectors. In the meantime, we equip node discrete distributions \( U_k \) with Wasserstein metric:

\[
W_2^2(\hat{u}_i, \hat{u}_j|D) = \min_{T\in O(n)} \text{tr}(D T^T) \quad \text{s.t} \quad T \mathbb{1} = \hat{u}_i, \quad T^T \mathbb{1} = \hat{u}_j.
\]

Here \( D \in \mathbb{R}^{k \times k} \) refers to the underlying distance matrix with \( D_{ij} = d(v_i, v_j) \).

### 3.3. The Wasserstein graph diffusion process

The Wasserstein graph diffusion process boils down to a generalized aggregation function, which is specified for producing discrete distributional representations. During the process, valid semantic information from different nodes can be shared, which can alleviate information distortion caused by missing attributes.

**Generalized aggregation function.** Through the space transformation \( \phi \), we obtain the initial node distributional representations \( \hat{U}_k(0) \). At each WGD layer \( l \), the aggregation function takes the distribution embedding \( \hat{U}_k(l-1) \) and the distance matrix \( D \) as the input:

\[
\hat{U}_k(0) = \phi(U_k) \text{ with fixed supp}(U_k); \quad \hat{U}_k(l) = \text{Barycenter\_Update}(\hat{U}_k(l-1), D). \tag{5}
\]

Barycenter\_Update is a generalized MEAN(·) aggregation function in Wasserstein space that made considering Wasserstein barycenter. We update the node embedding \( \hat{u} \) by smoothing node distributions over its local neighborhood \( \mathcal{N}(u) \) (with self-loop):

\[
\hat{u} = \text{Barycenter\_Update}(\mathcal{N}(u), D)
:= \arg \inf_{\hat{u}'} \frac{1}{|\mathcal{N}(u)|} \sum_{u \in \mathcal{N}(u)} W_2^2(p, \hat{u}' |D), \tag{6}
\]

Here \( |\mathcal{N}(u)| \) is the degree of \( u \) and \( \hat{u}' \) is the distributional embedding of node \( u' \). In practice, we use Iterative Bregman Projection (IBP) algorithm (Benamou et al., 2015) to compute the discrete Wasserstein barycenter (see Algorithm 1).

**Support-sharing in Wasserstein diffusion.** Recall that the set of support points of barycenter \( S_u \), contains all possible combinations of the common support points \( \text{supp}(U_k) \):

\[
S_u = \left\{ \frac{1}{|\mathcal{N}(u)|} \sum_{i=1}^{\mathcal{N}(u)} x_i | x_i \in \text{supp}(U_k) \right\}. \tag{7}
\]

As the diffusion process goes on, \( S(u) \) will be larger and larger and differ from node to node. Therefore, we adopt the support sharing trick to reduce computation complexity. That is, no matter how many times we update node distributions, they always share the common support points \( \text{supp}(U_k) \) as well as the distance matrix \( D \).

### 3.4. The inverse transformation

Matrix factorization separates the observed attribute information into two parts and stores in \( \hat{U}_k \) and \( \hat{V}_k \) respectively. To take a full advantage of rich information in \( \hat{V}_k \), we need to pull nodes back to the Euclidean space, which has the same dimension as the original feature space.

**Inverse transformation formulation.** We first convert the updated Wasserstein embeddings \( \hat{U}_k \) to a principal component matrix \( \hat{U}_k \), which is orthogonal and unitary, then generate the Euclidean embedding \( \hat{X} \):

\[
\hat{U}_k = \text{Gram-Schmidt\_Ortho}(\varphi^{-1}(\hat{U}_k)) \tag{8}
\]

\[
\hat{X} = \hat{U}_k A_k \hat{V}_k^T.
\]

where \( \text{Gram-Schmidt\_Ortho} \) is the Gram-Schmidt orthogonalization processing, \( \varphi^{-1} \) is the inverse function of the given \( \varphi \) in space transformation (2). In our implementation, \( \varphi^{-1}(\cdot) = \text{log}((\cdot)) \). Interestingly, empirical results show that orthogonalizing node embeddings can efficiently alleviate over-smoothing problem.

**Feature reconstruction for matrix completion.** Note that this \( \hat{X} \) is not a matrix completion for the original \( X \), since the elements in \( X \) do not remain the same. However, WGD framework can be adapted to recover missing values with
additional neural networks (like MLP) and a reconstruction constrains. More details are provided in Section 4.2.

4. Empirical Study

In this section, we adapt the WGD framework to two tasks: node classification on missing-attribute graphs and matrix completion with side information of users and items.

4.1. Node classification on missing-attribute graphs

In node classification tasks, we apply WGD for node representation generation and then use a multi-layer perceptron (MLP) for prediction. Algorithm 2 summarizes the WGD_MLP architecture. Each WGD layer involves twice space transformation between principal components space and Wasserstein space and $h$ times Wasserstein diffusion. To avoid over-smoothing, we take the mean of output from each layer as the updated principal component matrix.

4.1.1. Experimental Setup

Datasets. We conduct experiments on three Citation datasets: CORA, CITESEER and PUBMED (summarized in Table 1) in two missing-attribute cases:

- **Partially Missing.** Given full attribute matrix $X \in \mathbb{R}^{n \times m}$, we generate a random mask matrix $M \in \{0, 1\}^{n \times m}$ with $p(M_{ij} = 0) = r_p$, the partially missing rate $r_p \in \{0.1, \ldots, 0.9\}$.

- **Entirely Missing.** Given full attribute matrix $X \in \mathbb{R}^{n \times m}$, we generate a random mask vector $m \in \{0, 1\}^n$ with $p(m_i = 0) = r_e$, the entirely missing rate $r_e \in \{0.1, \ldots, 0.9\}$.

Baselines. Since no graph embedding methods are available for missing-attribute graphs, we first impute the data and then produce node embedding based on the imputed attribute matrix. We apply four commonly used imputation approaches: zero-filling, mean-filling, soft-impute (Mazumder et al., 2010), and KNN-impute (Batista et al., 2002) for the data preprocessing. soft-impute is based on low-rank SVD decomposition. Working with the well-known graph neural network: GCN, we obtain four baselines: Zero_GCN, Mean_GCN, Soft_GCN, and KNN_GCN. We also compare our model against another graph neural network GAT with missing-attribute. GAT works with two imputation methods, i.e. Zero_GAT and Soft_GAT. Moreover, we compare with two additional methods that only take structures into consideration: the Label Propagation Algorithm (LP) and GCN_NoFeat (GCN with identity matrix as input).

**Model configurations.** For all experiments, we train models using Adam optimizer with 0.01 learning rate. We early stop the model training process with patience 100, select the best performing models based on validation set accuracy, and report the mean accuracy for 10 runs as the final results. We apply two-layer MLP with 128 hidden units for node prediction with fixed $k = 64$, $\varphi(\cdot) = \exp(\cdot)$. The optimal $L$, $h$ and the layers of GCN and GAT differ in different missing level. For partially missing, $L = 14$ and $h = 2, 4, 6$ when $r_p \in [0.1, 0.3]$, $r_p \in [0.4, 0.6]$, $r_p \in [0.7, 0.9]$ respectively. We use 2-layer GCN and GAT when $r_p \in [0.1, 0.6]$ and 4-layers when $r_p \in [0.7, 0.9]$. For whole missing, when $r_e \in [0.1, 0.5]$, $L = 10$, $h = 2$; when $r_e \in [0.6, 0.9]$, $L = 14$, $h = 6$. We use 2-layer GCN and GAT when $r_e \in [0.1, 0.3]$ and 4-layer when $r_e \in [0.4, 0.9]$. For more experimental details please refer to our codes: https://anonymous.4open.science/r/3507bfe0-b3b1-4d18-a7b2-eb3643ceedb1.

**Algorithm 2 WGD_MLP for node classification**

**Input:** attribute matrix $X \in \mathbb{R}^{n \times m}$ with initial values, $k$

Apply $k$-rank SVD to $X$: $U_k, \Lambda_k, V_k = \text{SVD}(X, k)$

Initialize $U^{(0)}_k \leftarrow U_k$

for $l = 1 \text{ to } L$

space transformation:

$\hat{U}^{(l-1)}_k = L_1 \cdot \text{Normalize}(\varphi(U^{(l-1)}_k))$

$\tilde{U}_k^{(0)} \leftarrow \hat{U}^{(l-1)}_k$

for $i = 1 \text{ to } h$

Wasserstein diffusion:

$\hat{U}_k^{(i)} = \text{Barycenter_Update}(\tilde{U}^{(l-1)}_k, D)$

end for

inverse transformation:

$U_k^{(i)} = \text{Gram_Schmidt_Ortho}(\varphi^{-1}(\hat{U}^{(i)}_k))$

end for

$\tilde{U}_k = \text{Mean}(U_k^{(1)}, \ldots, U_k^{(L)})$

$X = \tilde{U}_k \Lambda_k V_k^T$

Apply MLP for node classification: $Y = \text{MLP}(\tilde{X})$

4.1.2. Experimental Results

**Results.** As shown in Fig 2, WGD_MLP has the best performance on all datasets across different entirely and partially missing levels. Except for LP and GCN_NoFeat, all methods have comparable performance at low missing level (missing ratio lower than 0.4). As in this case, observed attribute information is adequate for missing value imputation. As the missing ratio increases, matrix imputation becomes harder and harder and noise would be introduced during the pro-

| Dataset | Nodes | Edges | Features | Classes | Label Rate |
|---------|-------|-------|----------|---------|------------|
| Cora    | 2,708 | 5,278 | 1,433    | 7       | 0.052      |
| Citeseer| 3,327 | 4,552 | 3,703    | 6       | 0.036      |
| Pubmed  | 19,717| 44,324| 500      | 3       | 0.003      |

| Dataset   | Nodes | Edges | Features | Classes | Label Rate |
|-----------|-------|-------|----------|---------|------------|
| Citeseer  | 3,327 | 4,552 | 3,703    | 6       | 0.036      |
| Citeseer  | 3,327 | 4,552 | 3,703    | 6       | 0.036      |
| Citeseer  | 3,327 | 4,552 | 3,703    | 6       | 0.036      |
Figure 2. Average accuracy of node classification with different entirely missing (upper) and partially missing (lower) levels. At the highest missing level, WGD_MLP yields 8% higher accuracy in entirely missing case and 5% higher accuracy in partially missing case compared with the best baselines.

4.1.3. SENSITIVE ANALYSIS

Different from GCN layers, each WGD layer aggregates information across $h-hop$ neighbors. In this way, $L$ layers WGD model involves $L*h$ hops neighbors. The results of model tuning are shown in Figure 3. We only report the experimental results on CORA at 0.9 missing ratio on account of the similar performances on other datasets and missing ratios.

Except the curve of $L = 1(h * 5)$, all curves of $L$ have clear trend of increasing as $h$ increases. WGD_MLP has the best performance with $L = 5, h = 9$. In this case, $45 - hop$ neighbors are involved in the graph embedding process. As we know, many GNN models encounter the over-smoothing issue when they go deep. However, our method can efficiently handle over-smoothing. This is due to two strategies: incorporating the output of all WGD layers and orthogonalization which make nodes different. The curve of $L = 1(h * 5)$ provides the experimental illustration: $L = 1$ means that there is only once orthogonalization, the performance would decrease starting from $h = 4 * 5$.

4.1.4. ABLATION STUDY

We test the influence of low-rank SVD decomposition, Wasserstein diffusion and matirx composition on WGD respectively. For most ablation models, we report the experimental results on CORA in the partially missing setting, shown in Figure 4.

**Low-rank SVD decomposition.** We design two ablation models Wasser_GCN and SVD_GCN to verify the benefit and rationality of using decomposition.

- **Wasser_MLP.** At Wasser_MLP layer $l$, we directly transform nodes to standard discrete distribution without SVD decomposition and then apply Wasserstein diffusion. We also leverage the inverse transform to pull nodes back to Euclidean space and use MLP for prediction. Each Wasser_MLP is formulated as follows:

$$X^{(l+1)} = \varphi^{-1}(\text{Barycenter Update}(\phi(X^{(l)})))$$

(9)

where $\phi$ is defined in 2, $\varphi(\cdot) = \log(\cdot)$. Even at 0.1 partially missing ratio, Wasser_MLP only has 0.439
Figure 3. Sensitive analysis on times \( h \) of Wasserstein diffusion in each layer and the number of WGD layers \( L \) on Cora at 0.9 entirely missing (left) and partially missing (right) ratio, respectively. WGD_MLP has better performance with larger \( h \) and \( L \).

accuracy. It illustrates the clear benefit of low-rank decomposition: reduce computation complexity and improve performance.

- **SVD_GCN.** To investigate how the low-rank assumption affects graph learning model, we apply SVD on the attribute matrix before feed it to a two-layer GCN:

\[
U_k, \Lambda_k, V_k = \text{SVD}(X, k) \\
\tilde{X} = \text{GCN}(U_k\Lambda_kV_k^\top)
\]  

We provide all the experimental results of SVD_GCN in Figure 2. Except on Cora with partially missing, Zero_GCN and SVD_GCN have comparable performance across all the experiments. That means low-rank assumption is appropriate for these datasets.

**Wasserstein diffusion.** We adapt GCN and MEAN(·) aggregation function to WGD framework to update \( U_k \) to demonstrate the power of Wasserstein diffusion.

- **U_GCN.** We use GCN to update \( U_k \). The architecture of the GCN-based framework is similar to SVD_GCN which takes reconstructed low-rank attribute matrix as input:

\[
\tilde{X} = \text{GCN}(U_k\Lambda_kV_k^\top)
\]  

As shown in Figure 4, U_GCN has the worst performance. Interestingly, U_Mean has minor fluctuations across all missing ratios. At the missing ratio of 0.9, its performance is close to that of WGD_MLP. A plausible explanation is that, the semantic information provided by \( V_k \) is very limited under extreme missing cases. It verifies our assumption that incorporating semantic information could help node representation be more expressive.

**SVD-based composition.** We compare the expressive capacity of node representations with and without matrix composition.

- **U_pred.** We use \( \bar{U}_k \), the final U-type matrix of WGD framework, for prediction. The curve of U_pred is similar to U_Mean. It confirms the necessary of taking information of \( V_k \) into account.

Figure 4. Ablation Study.

### 4.2. Multi-Graph Matrix completion

An interesting aspect of the WGD framework is that we can reconstruct feature matrix by leveraging simple neural networks with additional reconstruction constrains. In this section, we test the reconstruction ability of WGD on recommendation systems with known pairwise relationship among users and items. Algorithm 3 summarizes the WGD framework adapted for matrix completion tasks.

**Multi-WGD.** Let \( X \) be the rating matrix where \( R(X) \) and \( C(X) \) represent users and items, respectively. Follow the same definition of space transformation in Section 3.2, the corresponding initial distribution matrices of users and items are \( \phi(U_k) \) with \( \text{supp}(U_k) = C(V_k) \) and \( \phi(V_k) \) with
WGD, can be thought of as an overlay of WGDs. In the last multi-WGD for matrix completion (Jamali & Ester, 2010) and advanced matrix completion methods. We compare our Multi-WGD model against five methods provided in Table 2. More Statistics details are provided by (Monti et al., 2017). More Statistics details are provided in Table 2.

Benchmarks. We conduct experiments on two popular matrix completion datasets with multi-graph: FLIXSTER (Jamali & Ester, 2010) and MOVIELENS-100K (Miller et al., 2003) and use the same preprocessed data and splits provided by (Monti et al., 2017). More Statistics details are provided in Table 2.

Baselines. We compare our Multi-WGD model against five advanced matrix completion methods.

- **GRALS** (Rao et al., 2015): A graph regularized model utilizing alternating minimization methods and graph structure for completion.
- **sRMGCNN** (Monti et al., 2017): A geometric matrix completion method applying multi-graph CNNs to graphs of users and items.
- **GC-MC** (Berg et al., 2017): A graph-based method representing matrix completion as link prediction on user-item bipartite graphs.
- **F-EAE** (Hartford et al., 2018): An inductive completion method leveraging exchangeable matrix layers.
- **IGMC** (Zhang & Chen, 2020): The state of the art matrix completion method using a GNN to encoding subgraphs for prediction.

**Experimental Settings and Results.** We follow the experimental setup of (Monti et al., 2017) and take the common metric Root Mean Square Error (RMSE) to evaluate the accuracy of matrix completion. We use 4-layer MLP with 50 hidden units on all datasets. For FLIXSTER, we choose $k = 25$, $L = 3$, $h = 2$. For MOVIELENS-100K, we set $k = 25$, $L = 7$, $h = 1$. We train the model using Adam optimizer with 0.001 learning rate. Table 3 presents the experimental results. As we can see, our Multi-WGD model has comparable performance as the best baseline IGMC. However, we only rely on a MLP, while IGMC needs to train both GCM and MLP. Furthermore, IGMC requires to extract the enclosing subgraph for each target edge, which is extremely computationally expensive.

## 5. Conclusion

Missing-attribute graphs are ubiquitous in the real-world, while most graph learning approaches have limited ability to leverage incomplete information directly. In this work, we propose WGD, a framework to generate node representations with incomplete attributes. By matrix decomposition, we represent nodes as histograms and then develop a generalized message passing schema in Wasserstein space. It allows us to aggregate distributional information from neighbors so that reduce information distortion caused by missing features and incorporate attribute and graph structure. Compared with graph neural networks working with imputation techniques, our framework shows significant improvement on prediction especially with extremely rare features. We further adapt the framework to perform matrix completion with multi-graph. Experiment results on recommendation systems illustrate our capacity to recover missing features.
References

Batista, G. E., Monard, M. C., et al. A study of k-nearest
neighbour as an imputation method. *His*, 87(251-260):
48, 2002.

Benamou, J.-D., Carlier, G., Cuturi, M., Nenna, L., and
Peyré, G. Iterative bregman projections for regularized
transportation problems. pp. A1111–A1138, 2015.

Berg, R. v. d., Kipf, T. N., and Welling, M. Graph convolu-
tional matrix completion. *Computing Research Reposi-
tory*, 2017.

Bojchevski, A. and Günnemann, S. Deep gaussian em-
bedding of graphs: Unsupervised inductive learning via
ranking. *International Conference on Learning Representa-
tions*, 2018.

Bruna, J., Zaremba, W., Szlam, A., and Lecun, Y. Spectral
networks and locally connected networks on graphs. In
*International Conference on Learning Representations
(*ICLR*2014), CBLS, April 2014*, pp. http–openreview,
2014.

Buuren, S. v. and Groothuis-Oudshoorn, K. mice: Multi-
variate imputation by chained equations in r. *Journal of
statistical software*, pp. 1–68, 2010.

Chen, X., Chen, S., Yao, J., Zheng, H., Zhang, Y., and Tsang,
I. W. Learning on attribute-missing graphs. *IEEE Trans-
actions on Pattern Analysis and Machine Intelligence*, 2020.

Duarte, L., Lima, J. S., Maestri, R., Debastiani, V., and
Collevatti, R. G. Genvectors: An integrative analytical
tool for spatial genetics. pp. 330761, 2019.

Gao, H. and Huang, H. Deep attributed network embed-
ing. In *International Joint Conferences on Artificial
Intelligence*, pp. 3364–3370, 2018.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and
Dahl, G. E. Neural message passing for quantum chem-
istry. In *Proceedings of the 34th International Conference
on Machine Learning-Volume 70*, pp. 1263–1272, 2017.

Gondara, L. and Wang, K. Multiple imputation using
depth denoising autoencoders. *arXiv preprint
arXiv:1705.02737*, 2017.

Grover, A. and Leskovec, J. node2vec: Scalable feature
learning for networks. 2016.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive represen-
tation learning on large graphs. In *Advances in Neural
Information Processing Systems*, pp. 1024–1034, 2017.

Hartford, J., Graham, D. R., Leyton-Brown, K., and Ran-
vankanhsh, S. Deep models of interactions across sets.
*International Conference on Machine Learning*, 2018.

Hastie, T., Mazumder, R., Lee, J. D., and Zadeh, R. Matrix
completion and low-rank svd via fast alternating least
squares. pp. 3367–3402, 2015.

Hong, R., He, Y., Wu, L., Ge, Y., and Wu, X. Deep attributed
network embedding by preserving structure and attribute
information. *IEEE Transactions on Systems, Man, and
Cybernetics: Systems*, 2019.

Jamali, M. and Ester, M. A matrix factorization technique
with trust propagation for recommendation in social net-
works. In *Proceedings of the fourth ACM conference on
Recommender systems*, pp. 135–142, 2010.

Kipf, T. N. and Welling, M. Semi-supervised classification
with graph convolutional networks. *International
Conference on Learning Representations*, 2017.

Klicpera, J., Bojchevski, A., and Günnemann, S. Predict
then propagate: Graph neural networks meet personal-
ized pagerank. In *International Conference on Learning
Representations*, 2019a.

Klicpera, J., Weißenberger, S., and Günnemann, S. Diffu-
sion improves graph learning. In *Advances in Neu-
ral Information Processing Systems*, pp. 13354–13366,
2019b.

Mazumder, R., Hastie, T., and Tibshirani, R. Spectral reg-
ularization algorithms for learning large incomplete ma-
trices. *The Journal of Machine Learning Research*, 11:
2287–2322, 2010.

Miller, B. N., Albert, I., Lam, S. K., Konstan, J. A., and
Riedl, J. Movielens unplugged: experiences with an occa-
sionally connected recommender system. In *Proceedings
of the 8th international conference on Intelligent user
interfaces*, pp. 263–266, 2003.

Monti, F., Bronstein, M., and Bresson, X. Geometric matrix
completion with recurrent multi-graph neural networks.
In *Advances in Neural Information Processing Systems*, pp. 3697–3707, 2017.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online
learning of social representations. 2014.

Pimentel, T., Veloso, A., and Ziviani, N. Unsupervised
and scalable algorithm for learning node representations.
*International Conference on Learning Representations*, 2017.

Rao, N., Yu, H.-F., Ravikumar, P. K., and Dhillon, I. S. Col-
laborative filtering with graph information: Consistency
and scalable methods. In Advances in Neural Information Processing Systems, pp. 2107–2115, 2015.

Spinelli, I., Scardapane, S., and Uncini, A. Missing data imputation with adversarially-trained graph convolutional networks. Neural Networks, 129:249–260, 2020.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. Missing value estimation methods for dna microarrays. pp. 520–525, 2001.

Van Buuren, S. Multiple imputation of discrete and continuous data by fully conditional specification. Statistical methods in medical research, 16(3):219–242, 2007.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. Computing Research Repository, 2017.

Yang, C., Liu, Z., Zhao, D., Sun, M., and Chang, E. Y. Network representation learning with rich text information. In International Joint Conferences on Artificial Intelligence, pp. 2111–2117, 2015.

Yoon, J., Jordan, J., and Schaar, M. Gain: Missing data imputation using generative adversarial nets. In International Conference on Machine Learning, pp. 5689–5698. PMLR, 2018.

You, J., Ma, X., Ding, D. Y., Kochenderfer, M., and Leskovec, J. Handling missing data with graph representation learning. arXiv preprint arXiv:2010.16418, 2020.

Zhang, M. and Chen, Y. Inductive matrix completion based on graph neural networks. International Conference on Learning Representations, 2020.