GROMOV-WASSERSTEIN LEARNING FOR GRAPH MATCHING AND NODE EMBEDDING

Hongteng Xu[1,2], Dixin Luo[2], Hongyuan Zha[3], Lawrence Carin[2]

Xuehan

07/27/2021

OVERVIEW

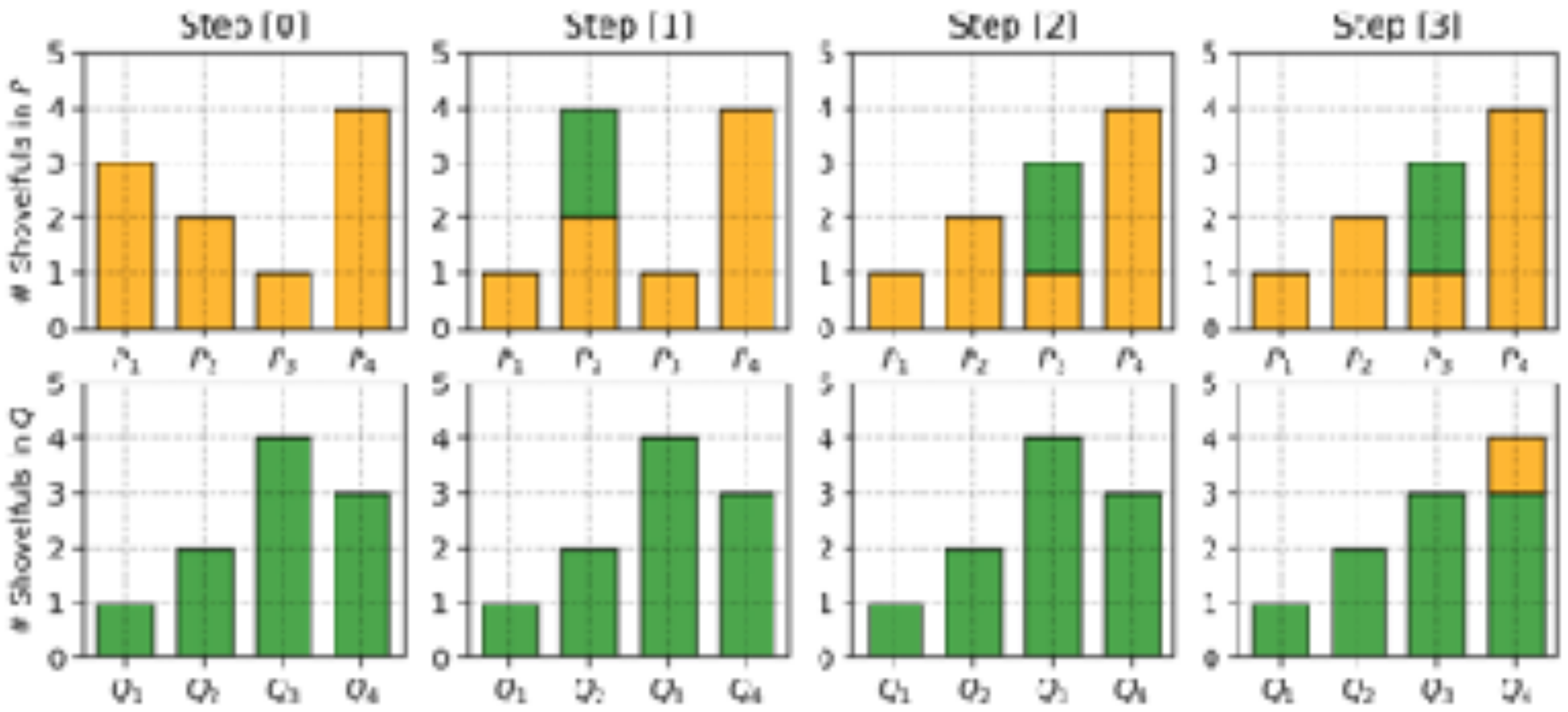
- Introduction
- Gromov-Wasserstein Learning (GWL) framework
- Experiments

WASSERSTEIN DISTANCE

- **K-Wasserstein distance:** $(WD-k) \quad \inf\left\{K(\gamma) := \int_{X\times Y} \|x-y\|_k^k d\gamma(x,y) : \gamma \in \Pi(\mu,\nu)\right\}$

- **Earth mover (1-wasserstein distance) :** $(WD-1) \quad \inf\left\{K(\gamma) := \int_{X\times Y} \|x-y\|_1^1 d\gamma(x,y) : \gamma \in \Pi(\mu,\nu)\right\}$

$$W(\mathbb{P}_p, \mathbb{P}_q) = \inf_{\gamma\in\Pi(\mathbb{P}_p,\mathbb{P}_q)} \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|] = \min_{\mathbf{T}\in\Pi(\mathbf{p},\mathbf{q})} \sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{T}_{ij}\cdot c(\boldsymbol{x}_i, \boldsymbol{y}_j)$$



w=2*d(P1,Q2)+2*d(P2,Q2)+1*d(P3,Q4)=5

P, Q	Q1	Q2	Q3	Q4
P1	0	2	0	0
P2	0	2	0	0
P3	0	0	0	1
P4	0	0	0	0

GROMOV-WASSERSTEIN DISTANCE

Definition 2.1. Let (X, d_X, μ_X) and (Y, d_Y, μ_Y) be two metric measure spaces, where (X, d_X) is a compact metric space and μ_X is a Borel probability measure on X (with (Y, d_Y, μ_Y) defined in the same way). The GromovWasserstein distance $d_{GW}(\mu_X, \mu_Y)$ is defined as:

Better explain the terms one by one and slowly.
Sometimes you need to read some tutorials to understand the terms.

$$\inf_{\pi \in \Pi(\mu_X, \mu_Y)} \iint_{X \times Y, X \times Y} L(x, y, x', y') d\pi(x, y) d\pi(x', y')$$

where $L(x, y, x', y') = |d_X(x, x') - d_Y(y, y')|$ is the loss function and $\Pi(\mu_X, \mu_Y)$ is the set of all probability measures on $X \times Y$ with μ_X and μ_Y as marginals.

PROPOSED METHOD

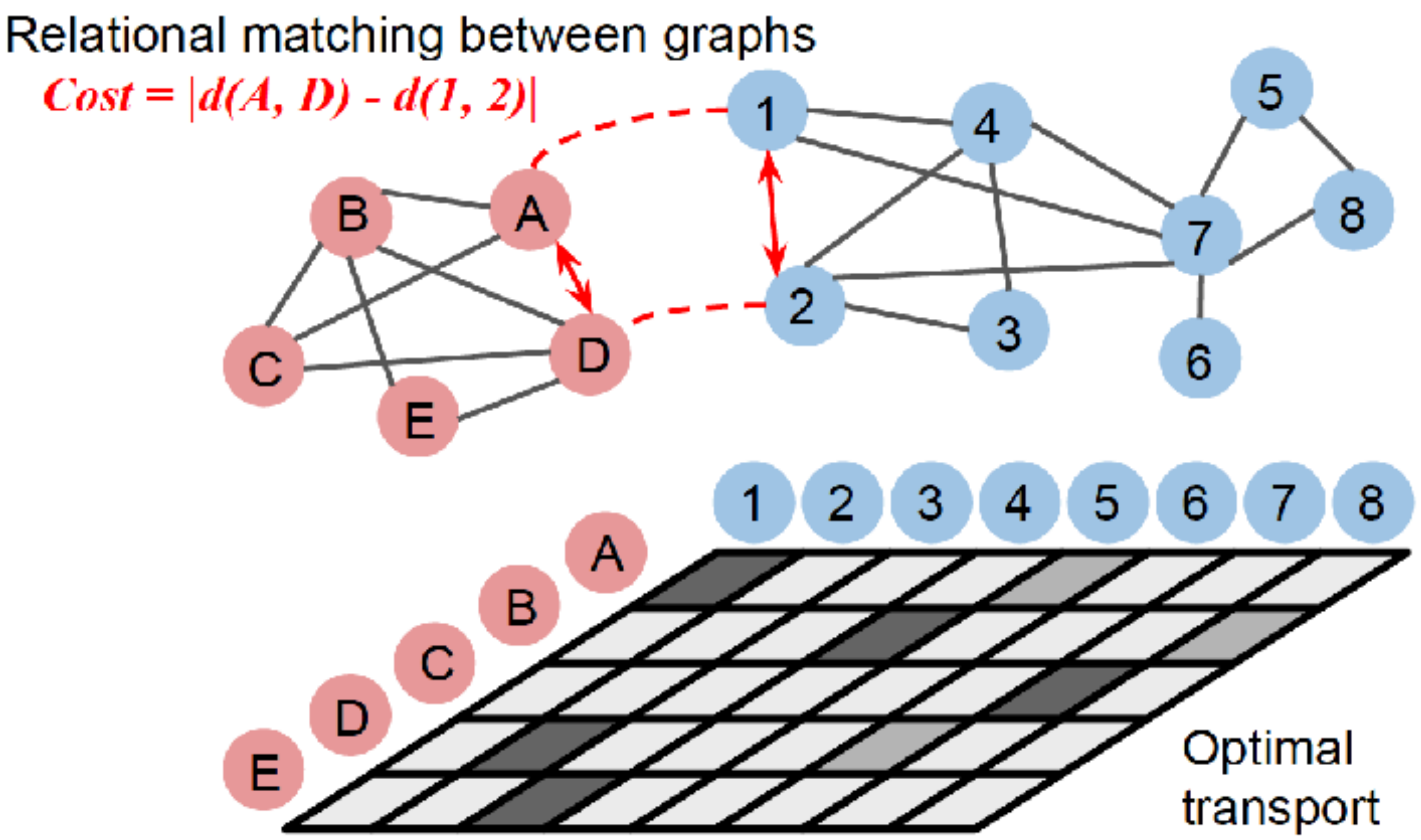
- **Graph matching:** Finding a correspondence between their nodes.
- **Node embedding:** Embedding their nodes in the same space.

Unify them in the **Gromov-Wasserstein Learning (GWL)** framework.

better explain the terms using the example.

$$d_{GW}(G_s, G_t) := \min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu}_s,\boldsymbol{\mu}_t)} \sum_{i,j,i',j'} L(c^s_{ij}, c^t_{i'j'})T_{ii'}T_{jj'} = \min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu}_s,\boldsymbol{\mu}_t)} \langle \boldsymbol{L}(\boldsymbol{C}_s, \boldsymbol{C}_t, \boldsymbol{T}), \boldsymbol{T}\rangle.$$

$$\Pi(\boldsymbol{\mu}_s, \boldsymbol{\mu}_t) = \left\{\boldsymbol{T} \in \mathbb{R}^{|\mathcal{V}_s|\times|\mathcal{V}_t|} \mid \boldsymbol{T}\boldsymbol{1}_{|\mathcal{V}_t|} = \boldsymbol{\mu}_s, \boldsymbol{T}^\top\boldsymbol{1}_{|\mathcal{V}_s|} = \boldsymbol{\mu}_t\right\}$$



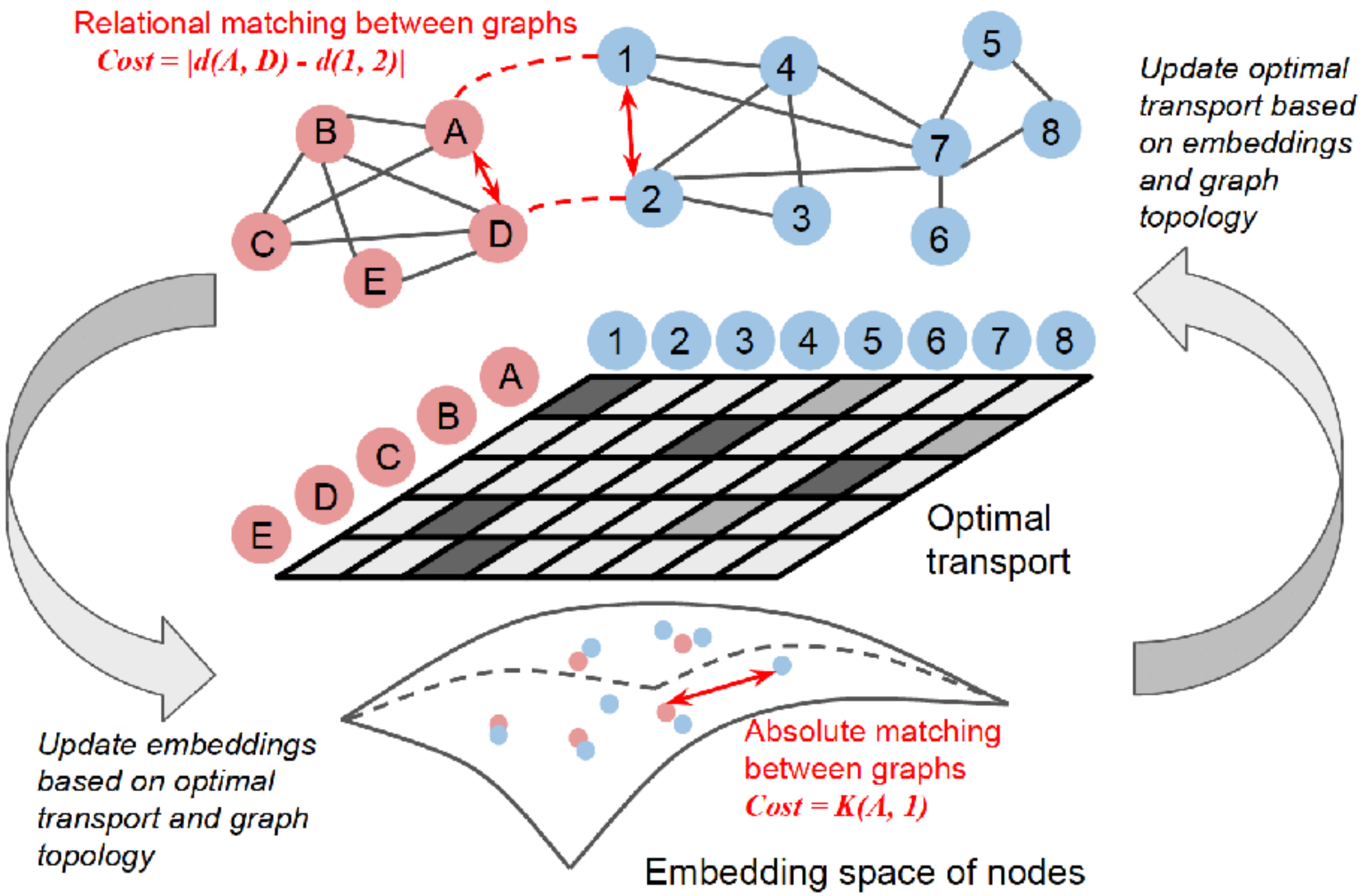
$$L_{jj'} = \sum_{i,i'} L\left(c^s_{ij}, c^t_{i'j'}\right)T_{ii'}$$

$$\boldsymbol{L}(\boldsymbol{C}_s, \boldsymbol{C}_t, \boldsymbol{T}) = \left[L_{jj'}\right] \in \mathbb{R}^{|\mathcal{V}_s|\times|\mathcal{V}_t|}$$

there is a lot of information and math terms.
Better organize them by input-output and objective.

GROMOV-WASSERSTEIN LEARNING

$$\min_{\boldsymbol{X}_s,\boldsymbol{X}_t} \min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu}_s,\boldsymbol{\mu}_t)} \underbrace{\langle \boldsymbol{L}(\boldsymbol{C}_s(\boldsymbol{X}_s), \boldsymbol{C}_t(\boldsymbol{X}_t), \boldsymbol{T}), \boldsymbol{T}\rangle}_{\text{Gromov-Wasserstein discrepancy}} + \underbrace{\alpha\langle \boldsymbol{K}(\boldsymbol{X}_s, \boldsymbol{X}_t), \boldsymbol{T}\rangle}_{\text{Wasserstein discrepancy}} + \underbrace{\beta \sum_{k=s,t} R(\boldsymbol{K}(\boldsymbol{X}_k, \boldsymbol{X}_k), \boldsymbol{C}_k)}_{\text{prior information}}.$$



GROMOV-WASSERSTEIN LEARNING

$$\min_{\boldsymbol{X}_s,\boldsymbol{X}_t}\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu}_s,\boldsymbol{\mu}_t)} \underbrace{\langle \boldsymbol{L}(\boldsymbol{C}_s(\boldsymbol{X}_s),\boldsymbol{C}_t(\boldsymbol{X}_t),\boldsymbol{T}),\boldsymbol{T}\rangle}_{\text{Gromov-Wasserstein discrepancy}} + \underbrace{\alpha\langle \boldsymbol{K}(\boldsymbol{X}_s,\boldsymbol{X}_t),\boldsymbol{T}\rangle}_{\text{Wasserstein discrepancy}} + \underbrace{\beta\sum_{k=s,t} R(\boldsymbol{K}(\boldsymbol{X}_k,\boldsymbol{X}_k),\boldsymbol{C}_k)}_{\text{prior information}}.$$

- $\boldsymbol{C}_k(\boldsymbol{X}_k) = (1-\alpha)\boldsymbol{C}_k + \alpha\boldsymbol{K}(\boldsymbol{X}_k,\boldsymbol{X}_k),\ \text{for } k = s,t$
- $\boldsymbol{K}(\boldsymbol{X}_k,\boldsymbol{X}_k) = \left[\kappa\left(\boldsymbol{x}_i^k,\boldsymbol{x}_j^k\right)\right] \in \mathbb{R}^{|\mathcal{V}_k\times|\mathcal{V}_k|}$ **distance between the node embedding within the same graph**
- $\boldsymbol{K}(\boldsymbol{X}_s,\boldsymbol{X}_t) = \left[\kappa\left(\boldsymbol{x}_i^s,\boldsymbol{x}_j^t\right)\right] \in \mathbb{R}^{|\mathcal{V}_s|\times|\mathcal{V}_t|}$ **distance between nodes of the two graphs**
- $R(\boldsymbol{X}_s,\boldsymbol{X}_t) = \sum_{k=s,t} L(\boldsymbol{K}(\boldsymbol{X}_k,\boldsymbol{X}_k),\boldsymbol{C}_k) + \underbrace{L(\boldsymbol{K}(\boldsymbol{X}_s,\boldsymbol{X}_t),\boldsymbol{C}_{st})}_{\text{optional}}$

GROMOV-WASSERSTEIN LEARNING

- **Updating optimal tranaport:** **m-th outer iteration , n-th inner iteration, based on KL divergence**
- $$\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu}_s,\boldsymbol{\mu}_t)} \left\langle \boldsymbol{L}\left(\boldsymbol{C}_s\left(\boldsymbol{X}_s^{(m)}\right), \boldsymbol{C}_t\left(\boldsymbol{X}_t^{(m)}\right), \boldsymbol{T}\right), \boldsymbol{T}\right\rangle + \alpha\left\langle \boldsymbol{K}\left(\boldsymbol{X}_s^{(m)}, \boldsymbol{X}_t^{(m)}\right), \boldsymbol{T}\right\rangle + \gamma \mathrm{KL}\left(\boldsymbol{T}\|\boldsymbol{T}^{(n)}\right). \quad (6)$$
- $$\mathrm{KL}\left(\boldsymbol{T}\|\boldsymbol{T}^{(n)}\right) = \sum_{ij} T_{ij}\log\frac{T_{ij}}{T^{(n)}} - T_{ij} + T_{ij}^{(n)}$$
- **Parameter α control：** $$\alpha_m = \frac{m}{M}$$
- **Updating embeddings:** $$\min_{\boldsymbol{X}_s,\boldsymbol{X}_t} \alpha_m \langle \boldsymbol{K}(\boldsymbol{X}_s,\boldsymbol{X}_t),\ \widehat{\boldsymbol{T}}^{(m)}\rangle + \beta R(\boldsymbol{X}_s,\boldsymbol{X}_t). \qquad (8)$$

EMBEDDING-BASED DISTANCE

- **Cosine-based distance:** $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 - \exp\left(-\sigma\left(1 - \frac{\boldsymbol{x}_i^\top \boldsymbol{x}_j}{\|\boldsymbol{x}_i\|_2 \|\boldsymbol{x}_j\|_2}\right)\right)$

- **Radial basis function(RBF)-based distance:** $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 - \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma^2}\right)$

ALGORITHM

$$\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu}_s,\boldsymbol{\mu}_t)}\langle \boldsymbol{L}(\boldsymbol{C}_s(\boldsymbol{X}_s^{(m)}),\boldsymbol{C}_t(\boldsymbol{X}_t^{(m)}),\boldsymbol{T}),\boldsymbol{T}\rangle + \alpha\langle \boldsymbol{K}(\boldsymbol{X}_s^{(m)},\boldsymbol{X}_t^{(m)}),\ \boldsymbol{T}\rangle + \gamma\mathrm{KL}(\boldsymbol{T}\|\boldsymbol{T}^{(n)}). \tag{6}$$

$$\min_{\boldsymbol{X}_s,\boldsymbol{X}_t}\alpha_m\langle \boldsymbol{K}(\boldsymbol{X}_s,\boldsymbol{X}_t),\ \widehat{\boldsymbol{T}}^{(m)}\rangle + \beta R(\boldsymbol{X}_s,\boldsymbol{X}_t). \tag{8}$$

Algorithm 1 Gromov-Wasserstein Learning (GWL)

Input: $\{\boldsymbol{C}_s,\boldsymbol{C}_t\}$, $\{\boldsymbol{\mu}_s,\boldsymbol{\mu}_t\}$, β, γ, the dimension D, the number of outer/inner iterations $\{M,N\}$.
Output: $\boldsymbol{X}_s$, $\boldsymbol{X}_t$ and $\widehat{\boldsymbol{T}}$.
Initialize $\boldsymbol{X}_s^{(0)}$, $\boldsymbol{X}_t^{(0)}$ randomly, $\widehat{\boldsymbol{T}}^{(0)} = \boldsymbol{\mu}_s\boldsymbol{\mu}_t^\top$.
For $m = 0 : M-1$
    Set $\alpha_m = \frac{m}{M}$.
    For $n = 0 : N-1$
        Update optimal transport $\widehat{T}^{(m+1)}$ via solving (6).
    Obtain $\boldsymbol{X}_s^{(m+1)}$, $\boldsymbol{X}_t^{(m+1)}$ via solving (8).
$\boldsymbol{X}_s = \boldsymbol{X}_s^{(M)}$, $\boldsymbol{X}_t = \boldsymbol{X}_t^{(M)}$ and $\widehat{\boldsymbol{T}} = \widehat{\boldsymbol{T}}^{(M)}$.
`\\ Graph matching:`
Initialize correspondence set $\mathcal{P} = \emptyset$
For $v_i \in \mathcal{V}_s$
    $j = \arg\max_j \widehat{T}_{ij}$. $\mathcal{P} = \mathcal{P} \cup \{(v_i \in \mathcal{V}_s, v_j \in \mathcal{V}_t)\}$.

SINKHORN-KNOPP ALGORITHM

- $$\mathcal{L}(P,\alpha,\beta) = \sum_{ij} \gamma\, p_{ij} \log p_{ij} + p_{ij}c_{ij} + \alpha^T(P\mathbf{1}_d - a) + \beta^T\left(P^T\mathbf{1}_d - b\right)$$
- $$\frac{\partial \mathcal{L}}{\partial p_{ij}} = 0$$ **, then** $$p_{ij} = e^{-\frac{\alpha_i}{\gamma}} e^{-\frac{C_{ij}}{\gamma}} e^{-\frac{\beta_i}{\gamma}}$$
- $$P = \operatorname{diag}(u) K \operatorname{diag}(v) = \operatorname{diag}(u) e^{-\frac{C_{ij}}{\gamma}} \operatorname{diag}(v)$$
- $$\mathbf{u} \odot (\mathbf{K}\mathbf{v}) = \mathbf{a} \quad \text{and} \quad \mathbf{v} \odot \left(\mathbf{K}^{\mathrm{T}}\mathbf{u}\right) = \mathbf{b}$$
- $$\mathbf{u}^{(\ell+1)} \stackrel{\text{def.}}{=} \frac{\mathbf{a}}{\mathbf{K}\mathbf{v}^{(\ell)}} \quad \text{and} \quad \mathbf{v}^{(t+1)} \stackrel{\text{def.}}{=} \frac{\mathbf{b}}{\mathbf{K}^{\mathrm{T}}\mathbf{u}^{(\ell+1)}}$$

PROXIMAL POINT METHOD

$$\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu}_s,\boldsymbol{\mu}_t)}\langle \boldsymbol{C}^{(m,n)} - \gamma \log \boldsymbol{T}^{(n)}, \boldsymbol{T}\rangle + \gamma H(\boldsymbol{T}), \quad (7)$$

where $\boldsymbol{C}^{(m,n)} = \boldsymbol{L}(\boldsymbol{C}_s, \boldsymbol{C}_t, \boldsymbol{T}^{(n)}) + \alpha \boldsymbol{K}\left(\boldsymbol{X}_s^{(m)}, \boldsymbol{X}_t^{(m)}\right) + \gamma,$

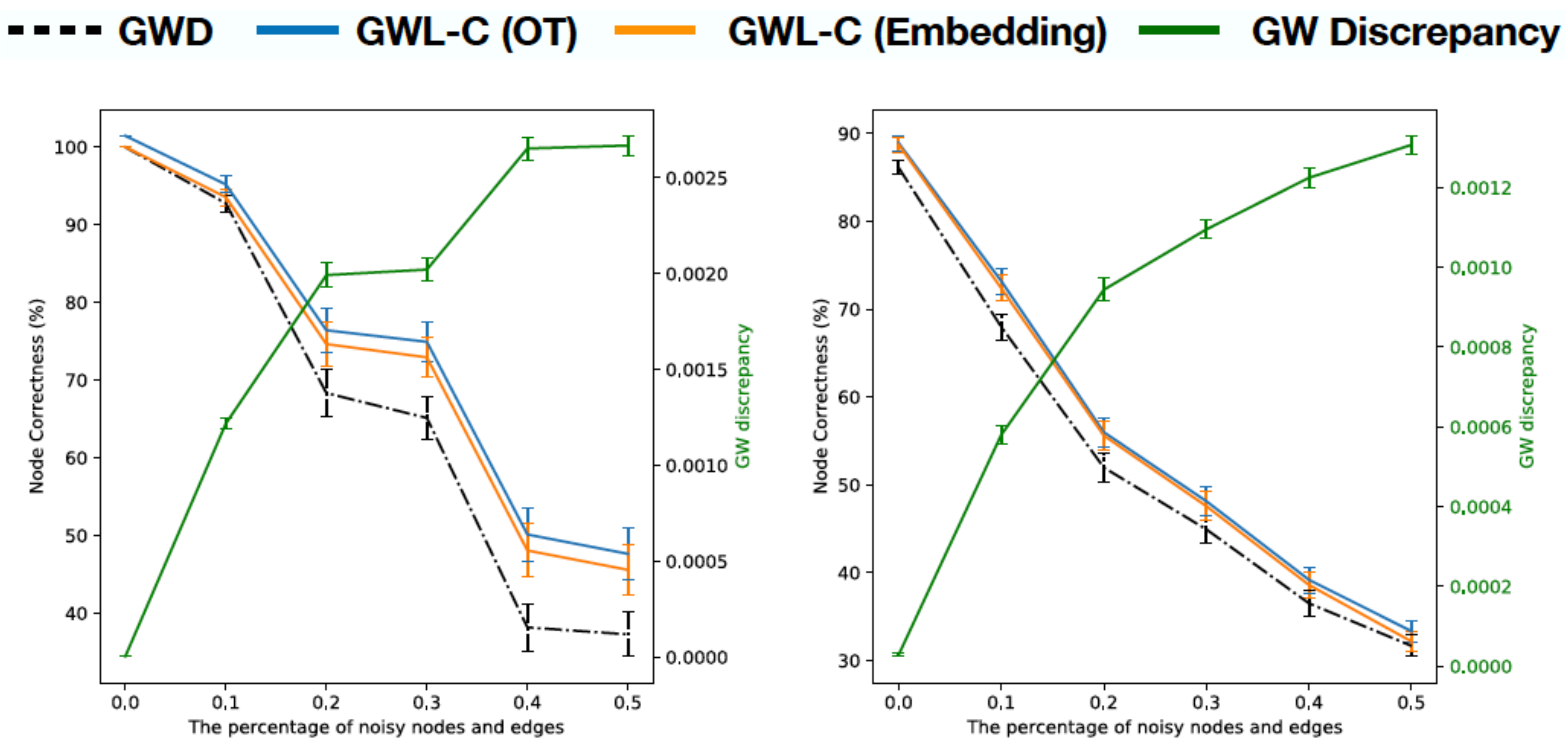
and $H(\boldsymbol{T}) = \sum_{i,j} T_{ij} \log T_{ij}$

Algorithm 2 Proximal Point Method for GW Discrepancy

Input: $\{\boldsymbol{C}_s, \boldsymbol{C}_t\}$, $\{\boldsymbol{\mu}_s, \boldsymbol{\mu}_t\}$, current embeddings $\{\boldsymbol{X}_s^{(m)}, \boldsymbol{X}_t^{(m)}\}$, γ, the number of inner iterations N.
Output: $\widehat{\boldsymbol{T}}^{(m+1)}$.
Initialize $\boldsymbol{T}^{(0)} = \boldsymbol{\mu}_s\boldsymbol{\mu}_t^\top$ and $\boldsymbol{a} = \boldsymbol{\mu}_s$.
for $n = 0 : N-1$ **do**
  Calculate the $\boldsymbol{C}^{(m,n)}$ in (7).
  Set $\boldsymbol{G} = \exp(-\frac{\boldsymbol{C}^{(m,n)}}{\gamma}) \odot \boldsymbol{T}^{(n)}$.
  `\\ Sinkhorn-Knopp algorithm:`
  for $j = 1 : J$ **do**
    $\boldsymbol{b} = \frac{\boldsymbol{\mu}_Y}{\boldsymbol{G}^\top \boldsymbol{a}}$
    $\boldsymbol{a} = \frac{\boldsymbol{\mu}_X}{\boldsymbol{G}\boldsymbol{b}}$
  end for
  $\boldsymbol{T}^{(n+1)} = \mathrm{diag}(\boldsymbol{a})\boldsymbol{G}\mathrm{diag}(\boldsymbol{b})$.
end for
$\widehat{\boldsymbol{T}}^{(m+1)} = \boldsymbol{T}^{(N)}$.

EXPERIMENTS

Explain the x and y axes first.
Then explain the curves.
Lastly, explain the settings and the performance.

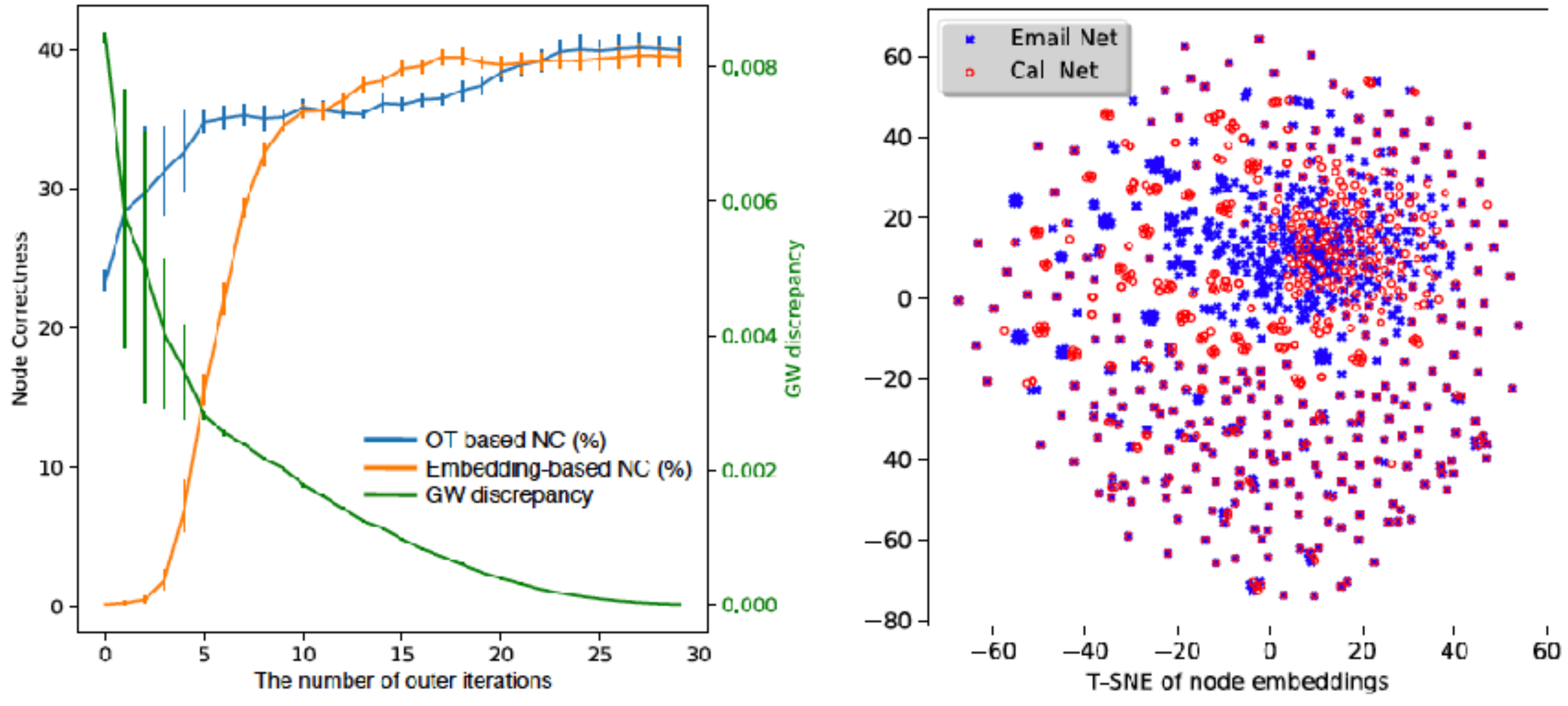


(a) K-NN: $|\mathcal{V}_s| = 50$ (b) BA: $|\mathcal{V}_s| = 50$

Figure 2. The performance of our method on synthetic data.

EXPERIMENTS

need to explain the baselines,
and the differences between
the baselines and the proposed method
how the difference lead to the
the performance improvement.



(a) Stability and convergence (b) Learned embeddings

Figure 3. Visualization of typical experimental results.

Table 1. Communication network matching results.

Method	Call→Email (Sparse) Node Correctness (%)	Call→Email (Dense) Node Correctness (%)
GAA	34.22	0.53
LRSA	38.20	2.93
TAME	37.39	2.67
GRAAL	39.67	0.48
MI-GRAAL	35.53	0.64
MAGNA++	7.88	0.09
HugAlign	36.21	3.86
NETAL	36.87	1.77
GWD	23.16±0.46	1.77±0.22
GWL-R	39.64±0.57	3.80±0.23
GWL-C	**40.45**±0.53	**4.23**±0.27

REFERENCE

- Cuturi, M. and Peyr e, G. Computational optimal transport.2017.
- Peyr e, G., Cuturi, M., and Solomon, J. Gromov-wasserstein averaging of kernel and distance matrices. In ICML, 2016.
- Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In NIPS, pp. 2292–2300, 2013.
- Benamou, J.-D., Carlier, G., Cuturi, M., Nenna, L., and Peyr e, G. Iterative Bregman projections for regularized transportation problems. SIAM Journal on Scientific Computing, 37(2):A1111–A1138, 2015.