Optimal Transport for graph representation

Unsupervised learning, graph prediction and neural OT solvers

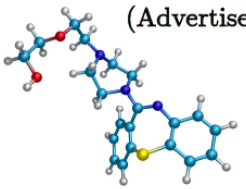
---

Rémi Flamary - CMAP, École Polytechnique, Institut Polytechnique de Paris

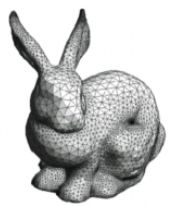
October 21 2025, Harvard CS 2840: Computational Optimal Transport for Machine Learning.

Graphs are everywhere

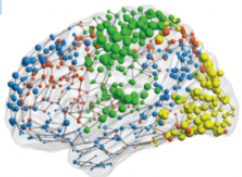
Social networks (Advertisement)



Drug/Material molecules (Chemistry)

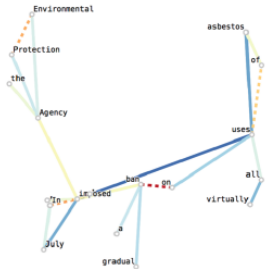


3D Meshes (Computer Graphics)

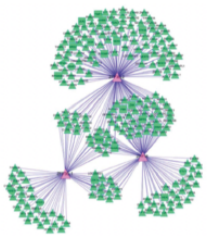


Brain connectivity (Neuroscience)

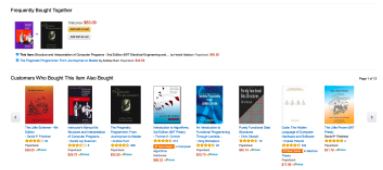
Transportation networks



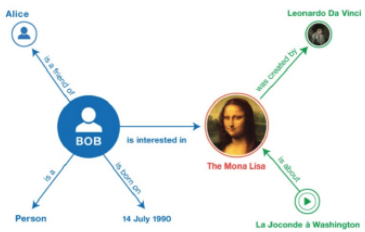
Words relationships (NLP)



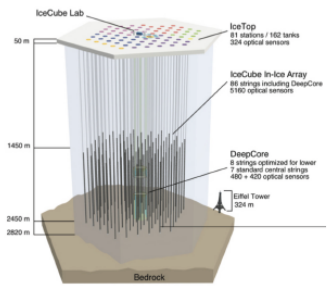
Gene Regulatory Network



Recommender systems (Amazon, Netflix)

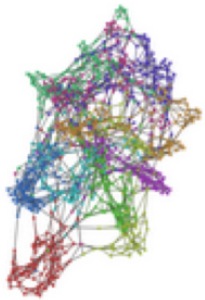


Knowledge graph (Causality)



Neutrino detection (High-energy Physics)

=



Graphs/ Networks

- Classical approach: spectral and Fourier based analysis and processing (GNN)
- What we will talk about: modeling graph as probability distributions (and use OT)

Table of content

Optimal Transport and divergences between graphs

- Gromov-Wasserstein and Fused Gromov-Wasserstein
- Relaxing the marginals constraints
- Graphs seen as distributions for GW

Learning graph representation with optimal transport

- GW barycenters and applications
- Dictionary learning with OT

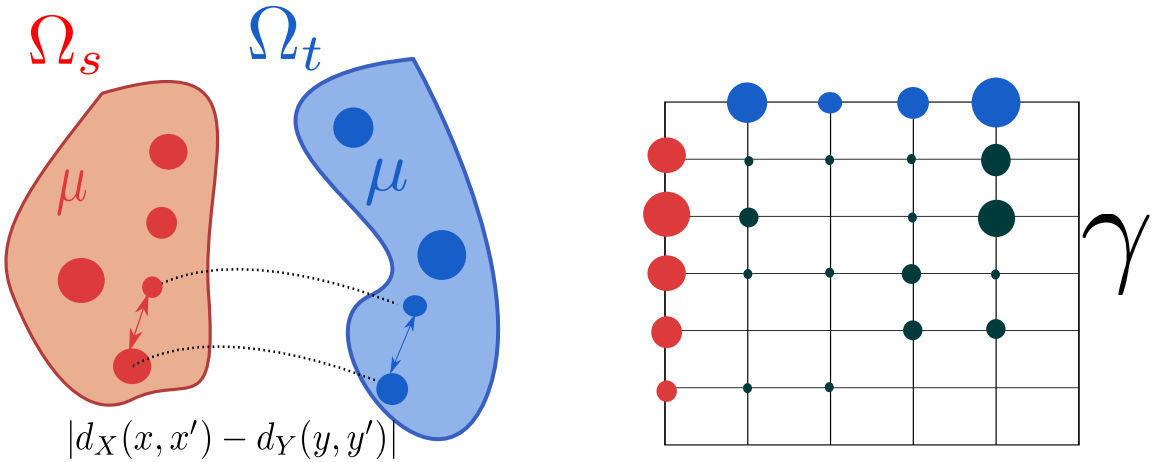
Supervised learning with OT on graphs

- Graph classification with OT
- Structured graph prediction with OT barycenters and Any2Graph

Scaling graph OT solvers with neural networks

- GRAph Level autoEncoder (GRALE)
- Unsupervised learning of OT plan prediction (ULOT)

Optimal Transport and divergences between graphs



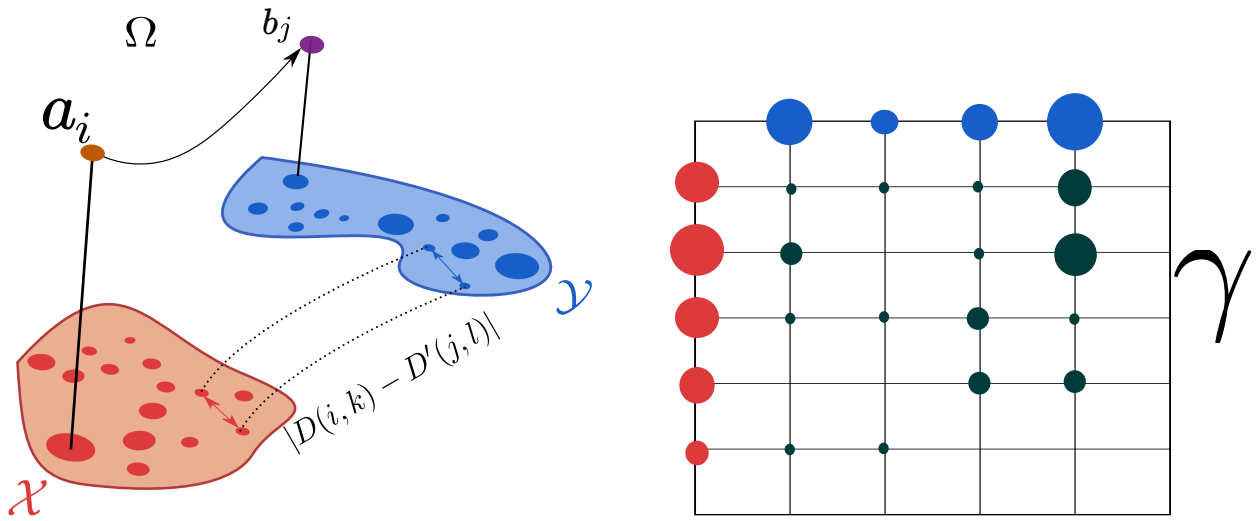
Inspired from Gabriel Peyré

GW for discrete distributions [Memoli, 2011]

$$\mathcal{GW}_p^p(\mu_s, \mu_t) = \min_{T \in \Pi(\mu_s, \mu_t)} \sum_{i,j,k,l} |D_{i,k} - D'_{j,l}|^p T_{i,j}\, T_{k,l}$$

with $\mu_s = \sum_i a_i \delta_{\mathbf{x}_i^s}$ and $\mu_t = \sum_j b_j \delta_{x_j^t}$ and $D_{i,k} = \|\mathbf{x}_i^s - \mathbf{x}_k^s\|$, $D'_{j,l} = \|\mathbf{x}_j^t - \mathbf{x}_l^t\|$

- Distance between metric measured spaces : across different spaces.
- Search for an OT plan that preserve the pairwise relationships between samples.
- Entropy regularized GW proposed in [Peyré et al., 2016].
- Fused GW interpolates between Wass. and GW [Vayer et al., 2018].



FGW for discrete distributions [Vayer et al., 2018]

$$\mathcal{FGW}_p^p(\mu_s, \mu_t) = \min_{T \in \Pi(\mu_s, \mu_t)} \sum_{i,j,k,l} \left((1-\alpha) C_{i,j}^q + \alpha |D_{i,k} - D'_{j,l}|^q\right)^p T_{i,j}\, T_{k,l}$$

with $\mu_s = \sum_i a_i \delta_{\mathbf{x}_i^s}$ and $\mu_t = \sum_j b_j \delta_{x_j^t}$ and $D_{i,k} = \|\mathbf{x}_i^s - \mathbf{x}_k^s\|$, $D'_{j,l} = \|\mathbf{x}_j^t - \mathbf{x}_l^t\|$

- Distance between metric measured spaces : across different spaces.
- Search for an OT plan that preserve the pairwise relationships between samples.
- Entropy regularized GW proposed in [Peyré et al., 2016].
- Fused GW interpolates between Wass. and GW [Vayer et al., 2018].

Unbalanced and semi-relaxed GW

Unbalanced Gromov-Wasserstein [Séjourné et al., 2020]

$$\min_{T \in \Pi(\mu_s, \mu_t)} \sum_{i,j,k,l} |D_{i,k} - D'_{j,l}|^p T_{i,j}\, T_{k,l} + \lambda^u D_\varphi(\mathbf{T}\mathbf{1}_m, \mathbf{a}) + \lambda^u D_\varphi(\mathbf{T}^\top \mathbf{1}_n, \mathbf{b})$$

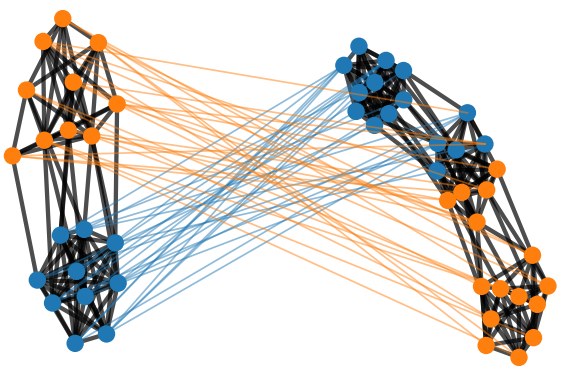
- The marginal constraints are relaxed by penalizing with divergence D_φ.
- Partial GW proposed in [Chapel et al., 2020]
- Unbalanced FGW [Thual et al., 2022] and Low rank [Scetbon et al., 2023].

Semi-relaxed (F)GW [Vincent-Cuaz et al., 2022a]

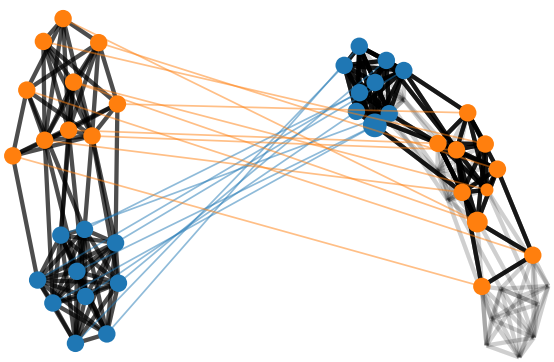
$$\min_{T \geq 0, \mathbf{T}\mathbf{1}_m = \mathbf{a}} \sum_{i,j,k,l} |D_{i,k} - D'_{j,l}|^p T_{i,j}\, T_{k,l}$$

- Second marginal constraint relaxed: optimal weights $\mathbf{b}$ w.r.t. GW.
- Very fast solver (Frank-Wolfe) because constraints are separable

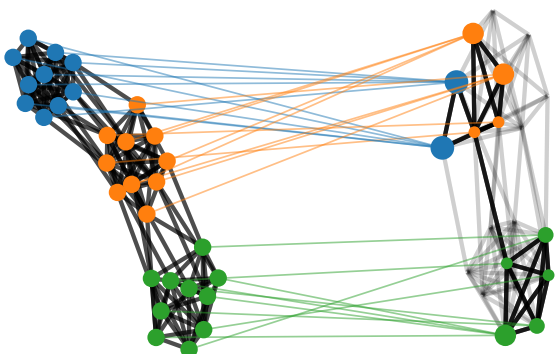
GW($\mathbf{C}, \mathbf{h}, \overline{\mathbf{C}}, \overline{\mathbf{h}}$) = 0.219



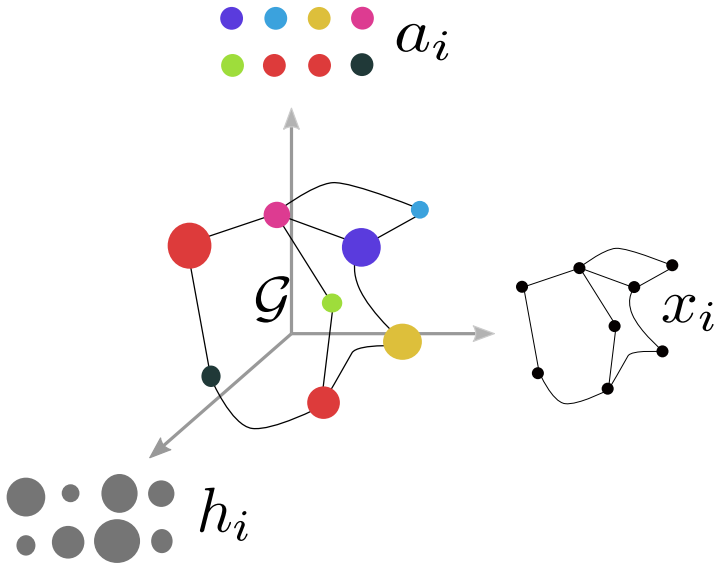
srGW($\mathbf{C}, \mathbf{h}, \overline{\mathbf{C}}$) = 0.05



srGW($\overline{\mathbf{C}}, \overline{\mathbf{h}}, \mathbf{C}$) = 0.113



Gromov-Wasserstein between graphs



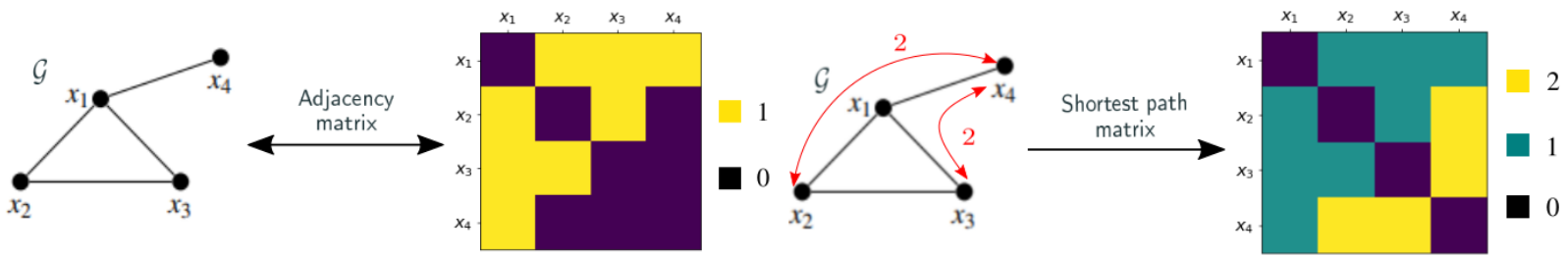
$$\left.\begin{array}{c}\\ \\ \\ \\ \end{array}\right\} \mu = \sum_i h_i \delta_{(x_i, a_i)}$$

$$\left.\begin{array}{c}\\ \\ \\ \\ \end{array}\right\} \mu_A = \sum_i h_i \delta_{a_i}$$

$$\left.\begin{array}{c}\\ \\ \\ \\ \end{array}\right\} \mu_X = \sum_i h_i \delta_{x_i}$$

Graph as a distribution $(\boldsymbol{D}, \mathbf{F}, \boldsymbol{h})$

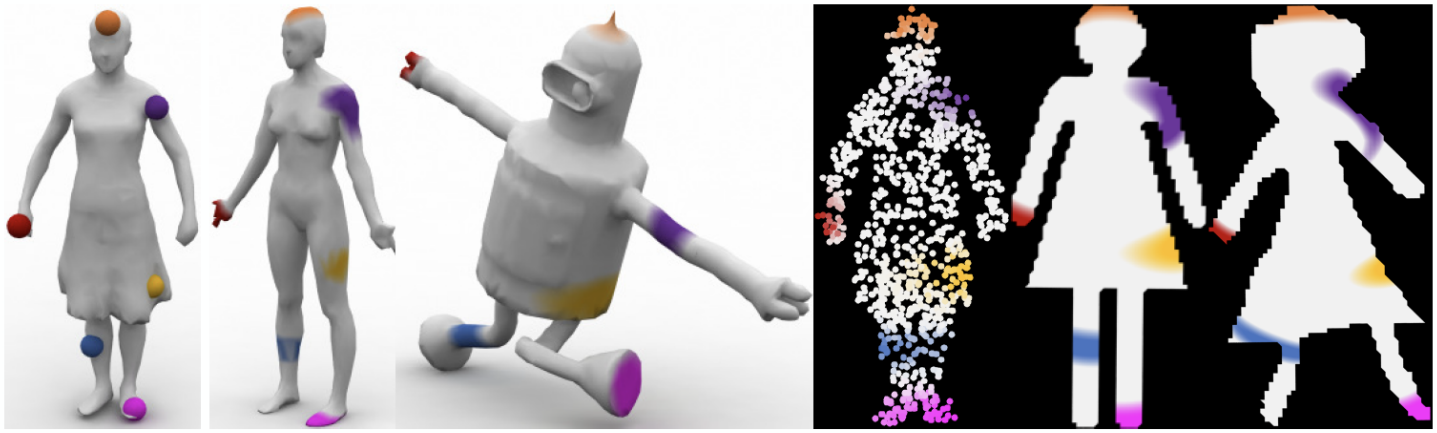
- The positions x_i are implicit and represented as the pairwise matrix $\boldsymbol{D}$.
- Possible choices for $\boldsymbol{D}$: Adjacency matrix, Laplacian, Shortest path, ...



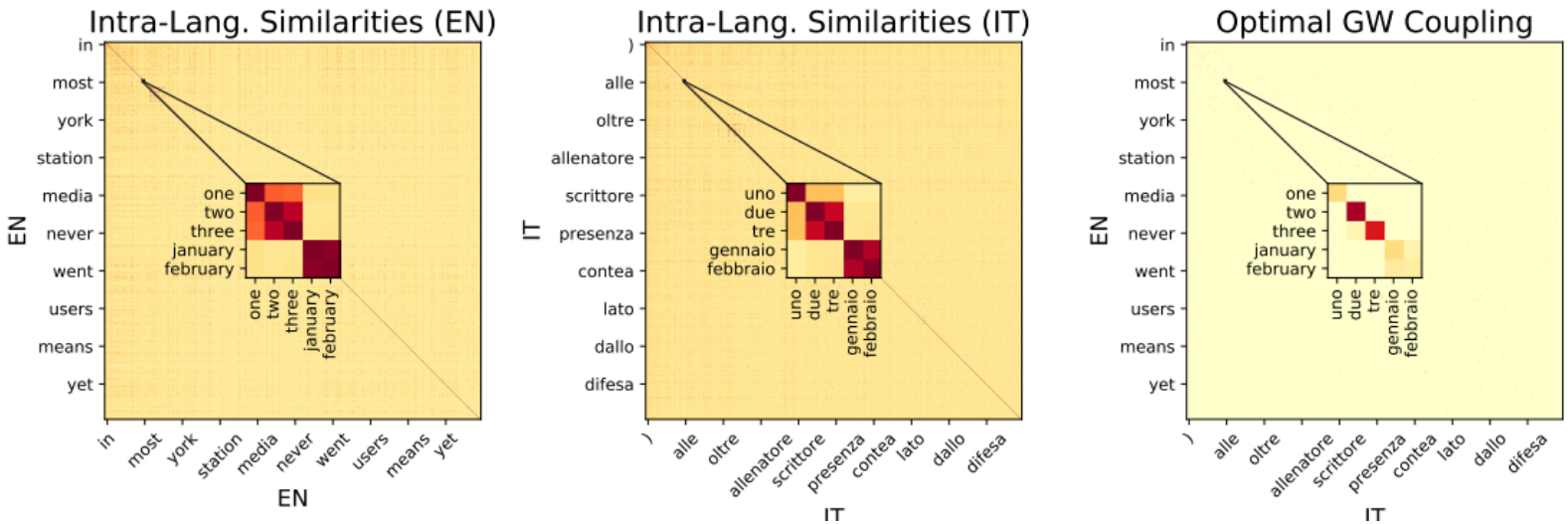
- The node features can be compared between graphs and stored in $\mathbf{F}$.
- h_i are the masses on the nodes of the graphs (uniform by default).

OT plan for graph alignment

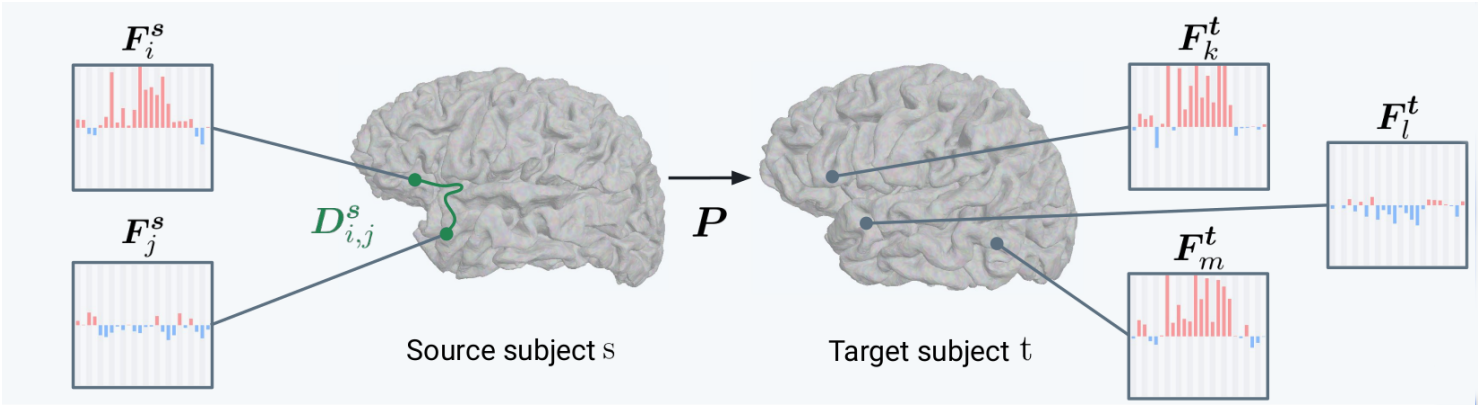
Shape matching between surfaces with GW [Solomon et al., 2016]



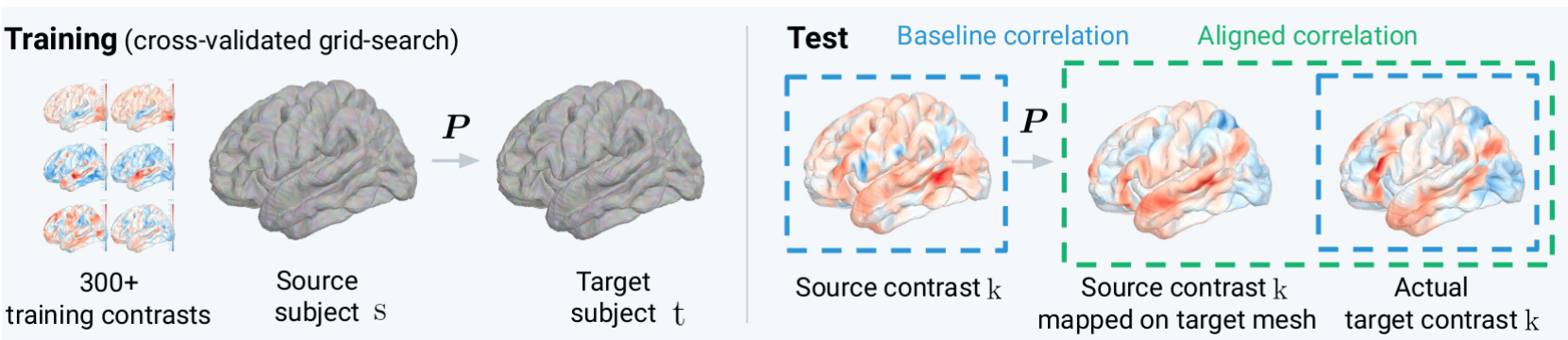
GW alignment of word embedding spaces [Alvarez-Melis and Jaakkola, 2018]

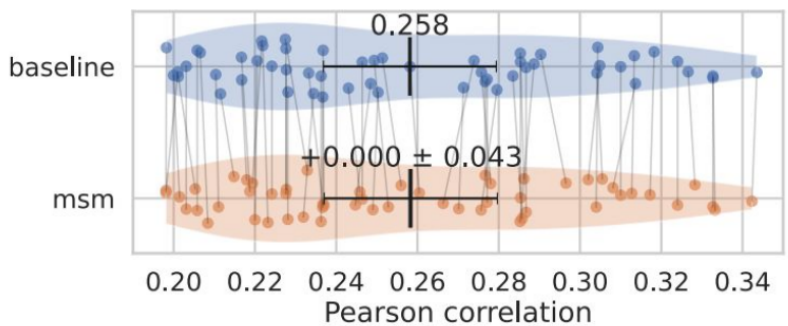


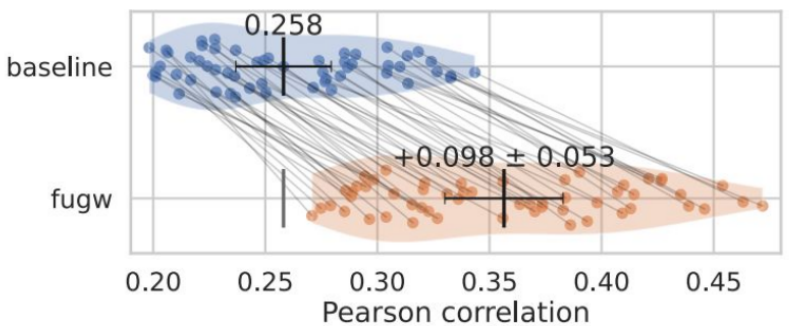
OT plan for brain alignment between individual geometries



Fused Unbalanced Gromov-Wasserstein [Thual et al., 2022]

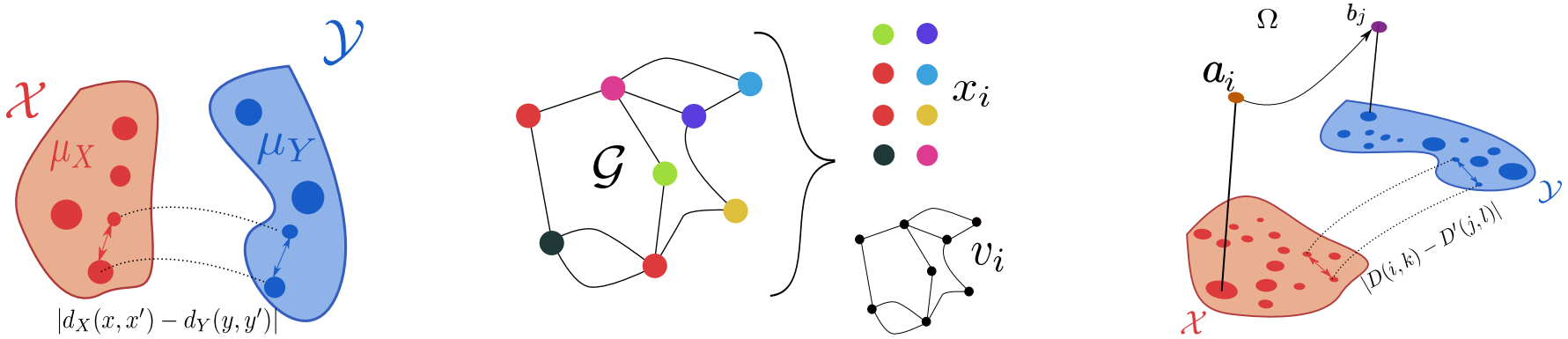






Learning graph representation with optimal transport

GW and FGW : the swiss army knife of OT on graphs



GW and extensions

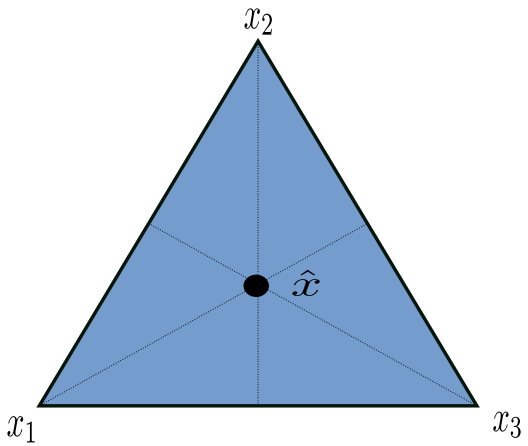
- GW [Memoli, 2011] and FGW [Vayer et al., 2018] are versatile distances for graph and structured data seen as distribution.
- Unbalanced [Séjourné et al., 2020] and semi-relaxed [Vincent-Cuaz et al., 2022a].

GW tools

- OT plan gives interpretable alignment between graphs.
- GW geometry allows barycenter and interpolation between graphs.
- GW provides similarity between graphs (data fitting).

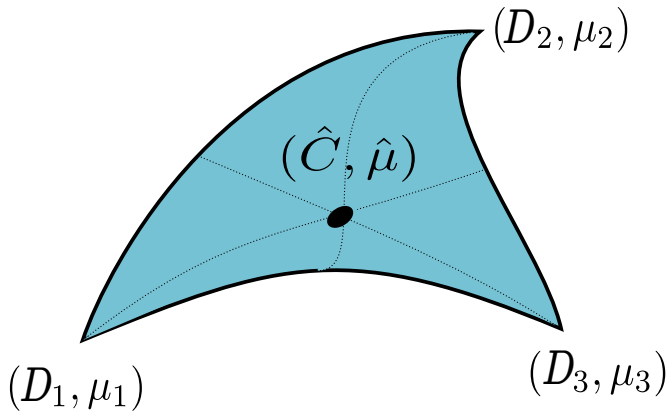
(F)GW barycenter

Euclidean barycenter



$$\min_x \sum_k \lambda_k \|x - x_k\|^2$$

FGW barycenter

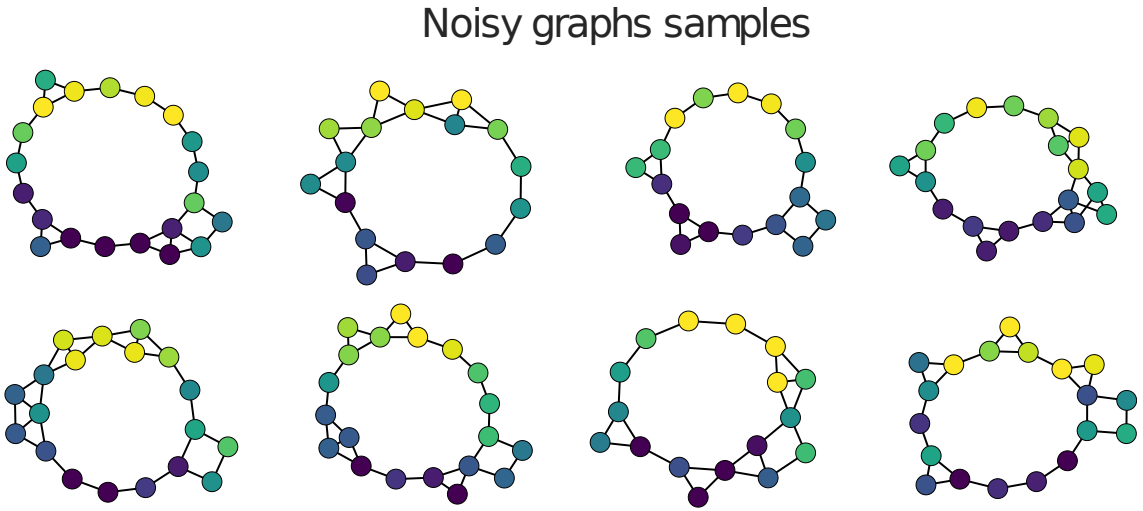


$$\min_{D \in \mathbb{R}^{n \times n}, \mu} \sum_i \lambda_i \mathcal{FGW}(D_i, D, \mu_i, \mu)$$

FGW barycenter

- Estimate FGW barycenter using Fréchet means ([Peyré et al., 2016] for GW).
- Barycenter optimization solved via block coordinate descent (on $\mathbf{T}, \mathbf{D}, \mu$).
- Extention of K-means clustering to FGW [Vayer et al., 2019a].
- Use for data augmentation /mixup in [Ma et al., 2023].

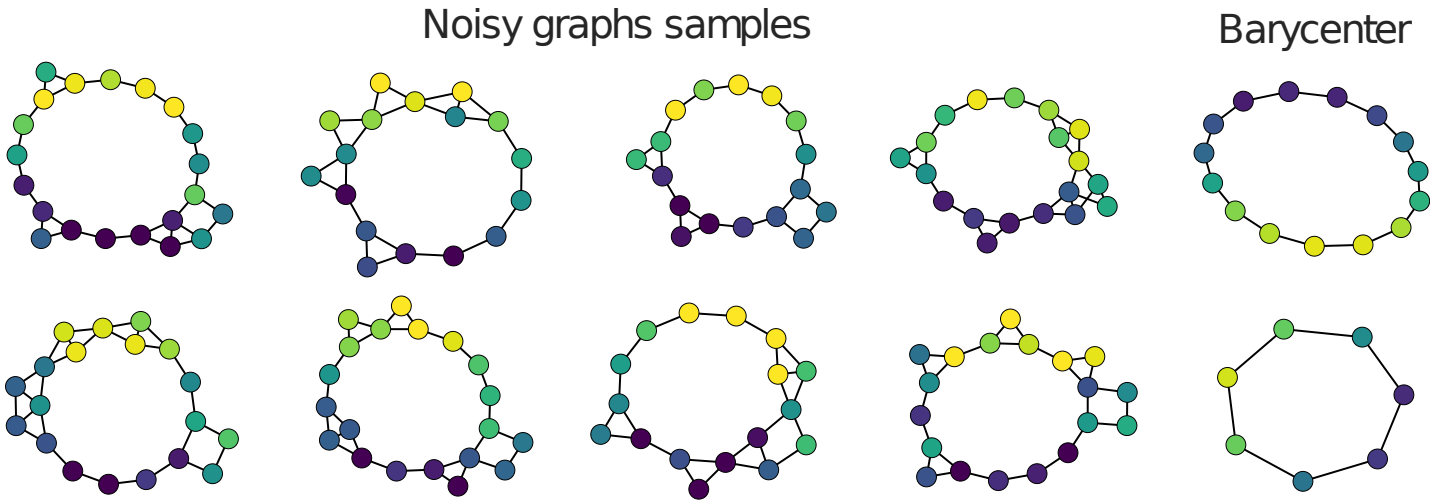
(F)GW barycenter



FGW barycenter

- Estimate FGW barycenter using Fréchet means ([Peyré et al., 2016] for GW).
- Barycenter optimization solved via block coordinate descent (on $\mathbf{T}, \mathbf{D}, \mu$).
- Extention of K-means clustering to FGW [Vayer et al., 2019a].
- Use for data augmentation /mixup in [Ma et al., 2023].

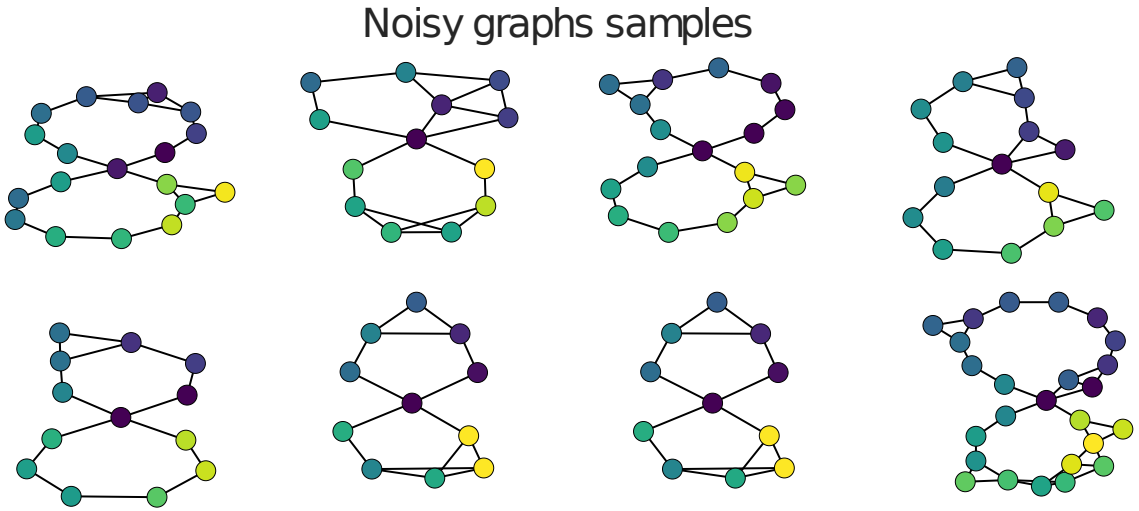
(F)GW barycenter



FGW barycenter

- Estimate FGW barycenter using Fréchet means ([Peyré et al., 2016] for GW).
- Barycenter optimization solved via block coordinate descent (on $\mathbf{T}, \mathbf{D}, \mu$).
- Extention of K-means clustering to FGW [Vayer et al., 2019a].
- Use for data augmentation /mixup in [Ma et al., 2023].

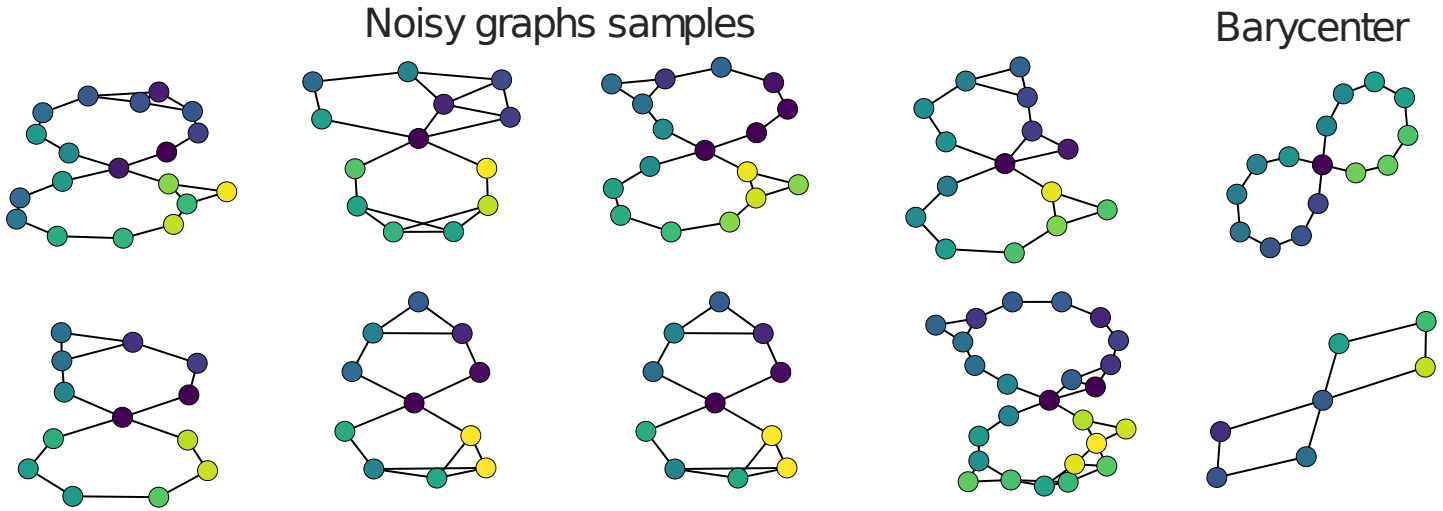
(F)GW barycenter



FGW barycenter

- Estimate FGW barycenter using Fréchet means ([Peyré et al., 2016] for GW).
- Barycenter optimization solved via block coordinate descent (on $\mathbf{T}, \mathbf{D}, \mu$).
- Extention of K-means clustering to FGW [Vayer et al., 2019a].
- Use for data augmentation /mixup in [Ma et al., 2023].

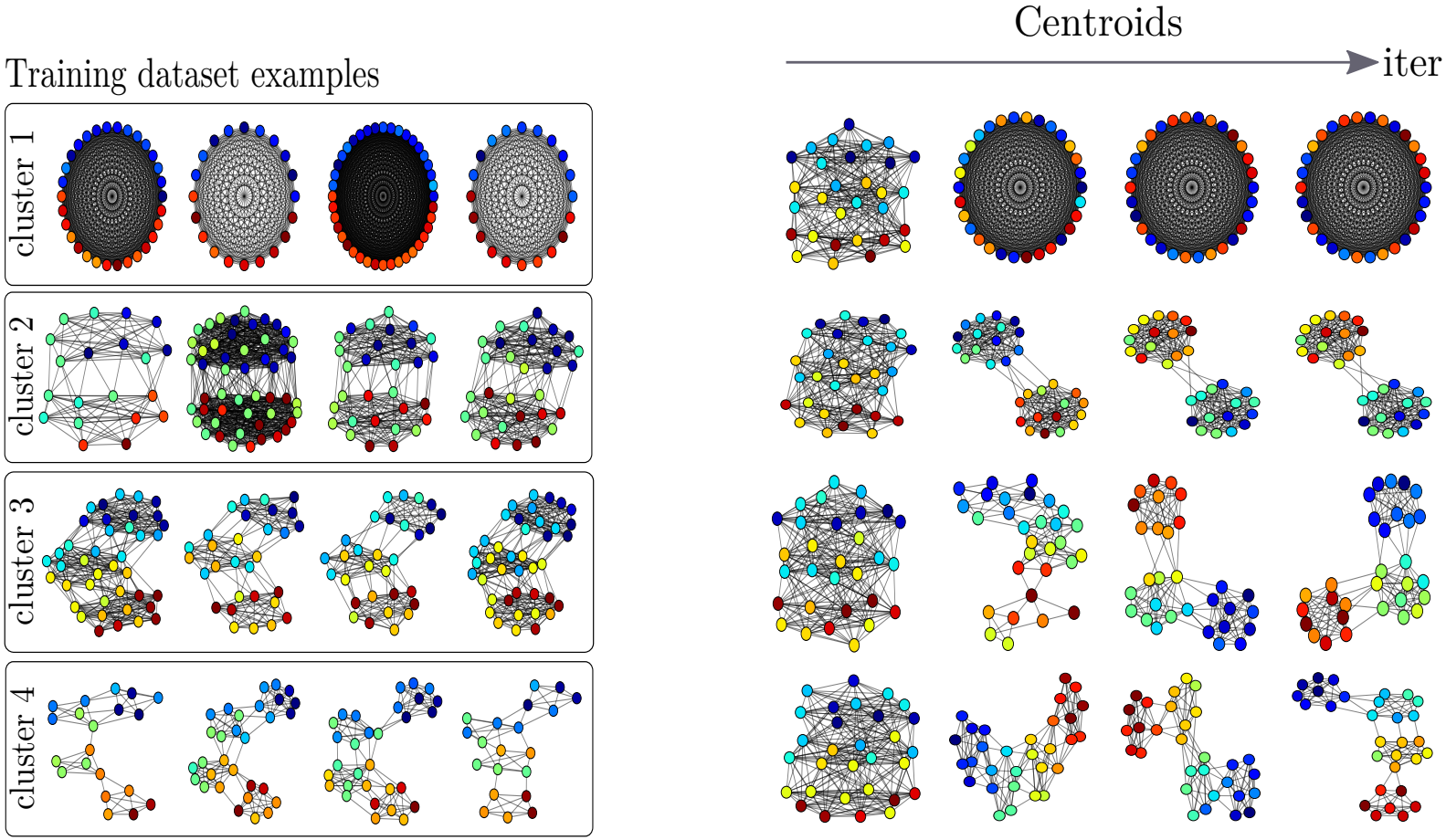
(F)GW barycenter



FGW barycenter

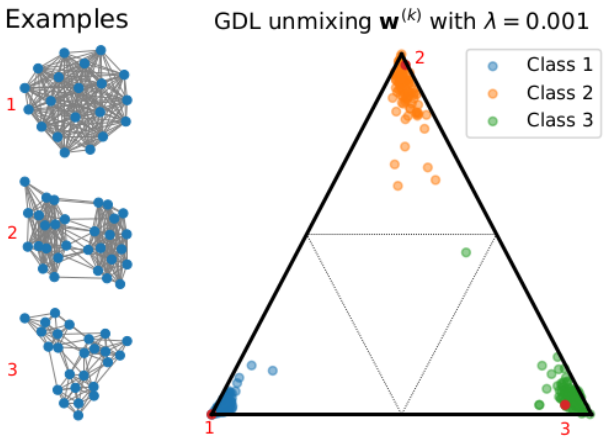
- Estimate FGW barycenter using Fréchet means ([Peyré et al., 2016] for GW).
- Barycenter optimization solved via block coordinate descent (on $\mathbf{T}, \mathbf{D}, \mu$).
- Extention of K-means clustering to FGW [Vayer et al., 2019a].
- Use for data augmentation /mixup in [Ma et al., 2023].

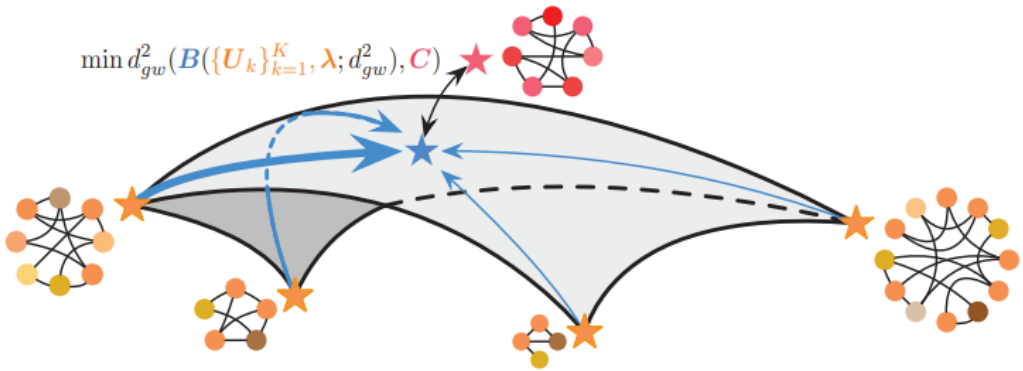
FGW for graphs based clustering



- Clustering of multiple real-valued graphs. Dataset composed of 40 graphs (10 graphs $\times$ 4 types of communities)
- k-means clustering using the FGW barycenter

Graph representation learning: Dictionary Learning





Representation learning for graphs

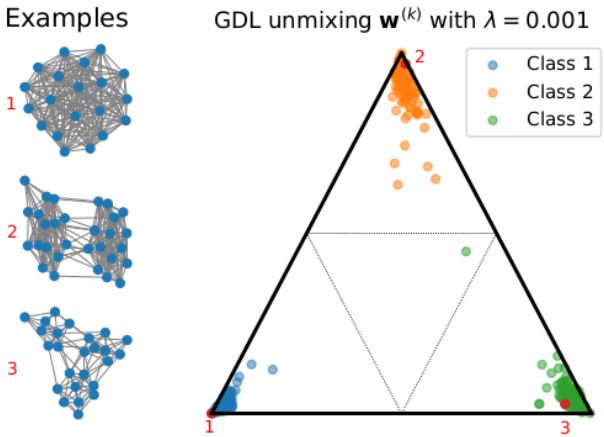
$$\min_{\{\overline{\mathbf{C}_k}\}_k, \{\mathbf{w}_i\}_i} \frac{1}{N} \sum_i GW(\mathbf{C}_i, \widehat{\mathbf{C}}(\mathbf{w}_i))$$

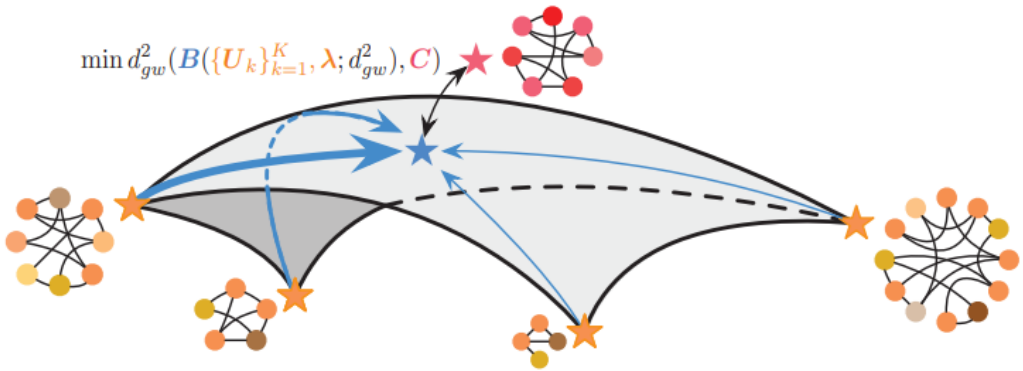
- Learn a dictionary $\{\overline{\mathbf{C}_k}\}_k$ of graph templates to describe a continuous manifold.
- The representation is learned by minimizing the (F)GW distance between the graph reconstruction from the embedding in the dictionary.
- Online Graph Dictionary learning : Linear model [Vincent-Cuaz et al., 2021].

$$\widehat{\mathbf{C}}(\mathbf{w}) = \textstyle\sum_k w_k \overline{\mathbf{C}_k}$$

- GW Factorization : Nonlinear (GW barycenter) model [Xu, 2020].

Graph representation learning: Dictionary Learning





Representation learning for graphs

$$\min_{\{\overline{\mathbf{C}_k}\}_k, \{\mathbf{w}_i\}_i} \frac{1}{N} \sum_i GW(\mathbf{C}_i, \widehat{\mathbf{C}}(\mathbf{w}_i))$$

- Learn a dictionary $\{\overline{\mathbf{C}_k}\}_k$ of graph templates to describe a continuous manifold.
- The representation is learned by minimizing the (F)GW distance between the graph reconstruction from the embedding in the dictionary.
- Online Graph Dictionary learning : Linear model [Vincent-Cuaz et al., 2021].
- GW Factorization : Nonlinear (GW barycenter) model [Xu, 2020].

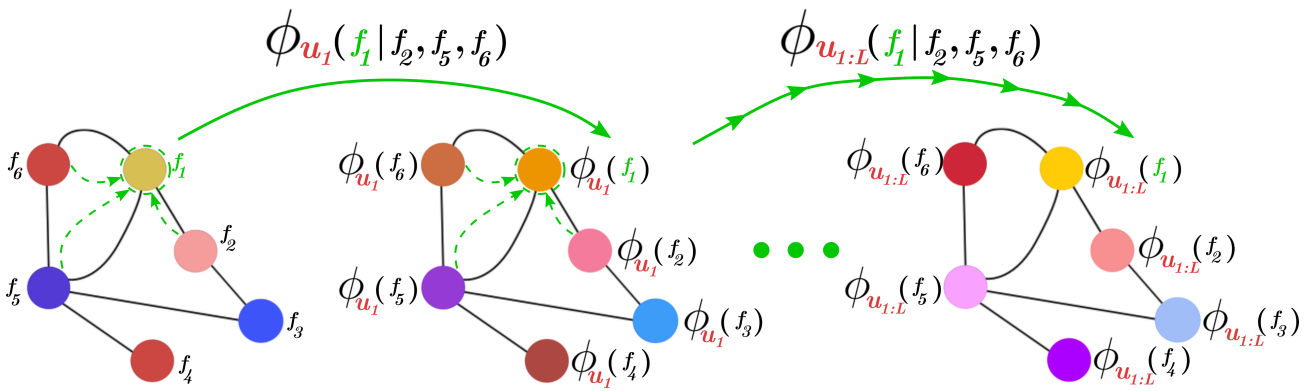
$$\widehat{\mathbf{C}}(\mathbf{w}) = \text{argmin}_{\mathbf{C}} \sum_k w_k GW(\mathbf{C}, \overline{\mathbf{C}_k})$$

Supervised learning with OT on graphs

Graph kernels and FGW

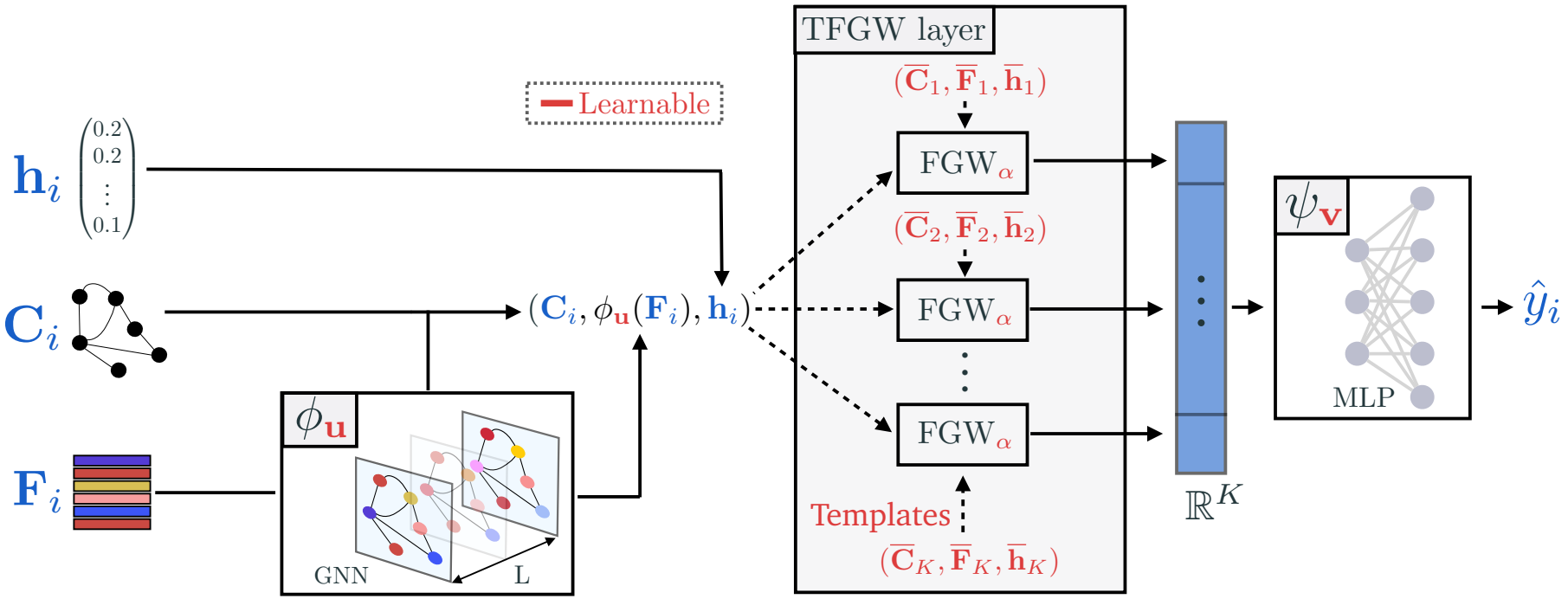
- Graph kernels still SOTA on many datasets : WWL [Togninalli et al., 2019].
- FGW can be used in a non-positive "kernel" [Vayer et al., 2019b].
- Graph dictionary learning methods provide euclidean embeddings for kernels [Vincent-Cuaz et al., 2021, Vincent-Cuaz et al., 2022a].

Graph Neural Networks [Bronstein et al., 2017]



- Each layer of the GNN compute features on graph node using the values from the connected neighbors : message passing principle.
- The final pooling step must remain invariant to permutations (min, max, mean).
- Can we encode graphs as distributions in GNN?

Template based Graph Neural Network with OT Distances



Template based FGW layer (TFGW) [Vincent-Cuaz et al., 2022b]

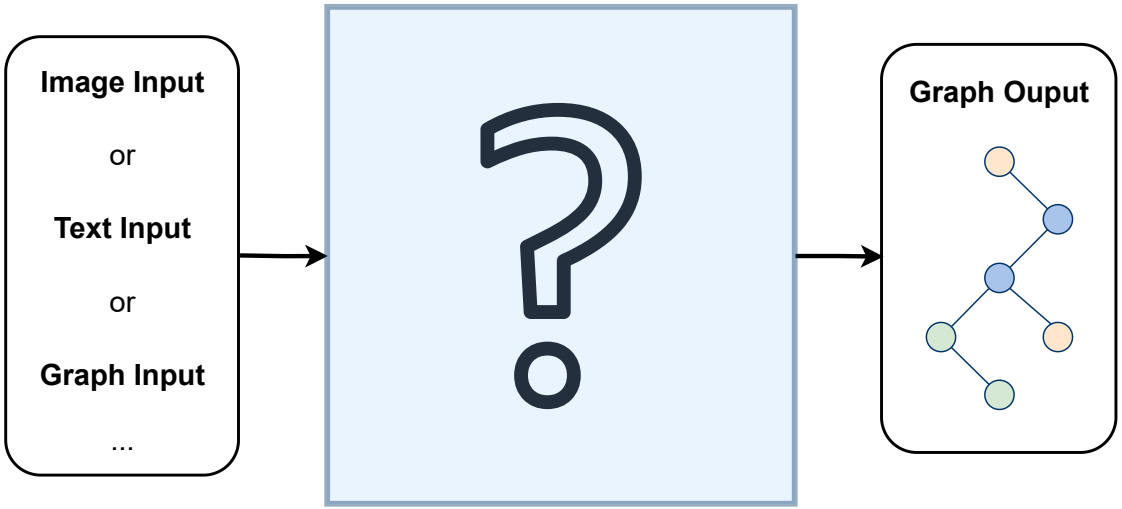
- Principle: represent a graph through its distances to learned templates.
- Novel pooling layer derived from OT distances.
- New end-to-end GNN models for graph-level tasks.
- Learnable parameters are illustrated in red above.

TFGW benchmark

category	model	MUTAG	PTC	ENZYMES	PROTEIN	NCI1	IMDB-B	IMDB-M	COLLAB
Ours	TFGW ADJ (L=2)	**96.4(3.3)**	**72.4(5.7)**	73.8(4.6)	**82.9(2.7)**	**88.1(2.5)**	**78.3(3.7)**	**56.8(3.1)**	**84.3(2.6)**
($\phi_{\mathbf{u}}$ = GIN)	TFGW SP (L=2)	94.8(3.5)	70.8(6.3)	**75.1(5.0)**	82.0(3.0)	86.1(2.7)	74.1(5.4)	54.9(3.9)	80.9(3.1)
OT emb.	OT-GNN (L=2)	91.6(4.6)	68.0(7.5)	66.9(3.8)	76.6(4.0)	82.9(2.1)	67.5(3.5)	52.1(3.0)	80.7(2.9)
	OT-GNN (L=4)	92.1(3.7)	65.4(9.6)	67.3(4.3)	78.0(5.1)	83.6(2.5)	69.1(4.4)	51.9(2.8)	81.1(2.5)
	WEGL	91.0(3.4)	66.0(2.4)	60.0(2.8)	73.7(1.9)	75.5(1.4)	66.4(2.1)	50.3(1.0)	79.6(0.5)
GNN	PATCHYSAN	91.6(4.6)	58.9(3.7)	55.9(4.5)	75.1(3.3)	76.9(2.3)	62.9(3.9)	45.9(2.5)	73.1(2.7)
	GIN	90.1(4.4)	63.1(3.9)	62.2(3.6)	76.2(2.8)	82.2(0.8)	64.3(3.1)	50.9(1.7)	79.3(1.7)
	DropGIN	89.8(6.2)	62.3(6.8)	65.8(2.7)	76.9(4.3)	81.9(2.5)	66.3(4.5)	51.6(3.2)	80.1(2.8)
	PPGN	90.4(5.6)	65.6(6.0)	66.9(4.3)	77.1(4.0)	82.7(1.8)	67.2(4.1)	51.3(2.8)	81.0(2.1)
	DIFFPOOL	86.1(2.0)	45.0(5.2)	61.0(3.1)	71.7(1.4)	80.9(0.7)	61.1(2.0)	45.8(1.4)	80.8(1.6)
Kernels	FGW - ADJ	82.6(7.2)	55.3(8.0)	72.2(4.0)	72.4(4.7)	74.4(2.1)	70.8(3.6)	48.9(3.9)	80.6(1.5)
	FGW - SP	84.4(7.3)	55.5(7.0)	70.5(6.2)	74.3(3.3)	72.8(1.5)	65.0(4.7)	47.8(3.8)	77.8(2.4)
	WL	87.4(5.4)	56.0(3.9)	69.5(3.2)	74.4(2.6)	85.6(1.2)	67.5(4.0)	48.5(4.2)	78.5(1.7)
	WWL	86.3(7.9)	52.6(6.8)	71.4(5.1)	73.1(1.4)	85.7(0.8)	71.6(3.8)	52.6(3.0)	81.4(2.1)
	Gain with TFGW	**+4.3**	**+4.4**	**+2.9**	**+4.9**	**+2.4**	**+6.7**	**+4.2**	**+2.9**

- Comparison with state of the art approach from GNN and graph kernel methods.
- Systematic and significant gain of performance with GIN+TFGW.
- Gain independent of GNN architecture (GIN or GAT).
- 3 year after publication, rankings of TFGW on "papers with code":
  #1 NCI1, #2 COLLAB, IMDB-M, #3 MUTAG, PROTEIN.
- Experiments suggests that TFGW has expressivity beyond Weisfeiler-Lehman Isomorphism tests.

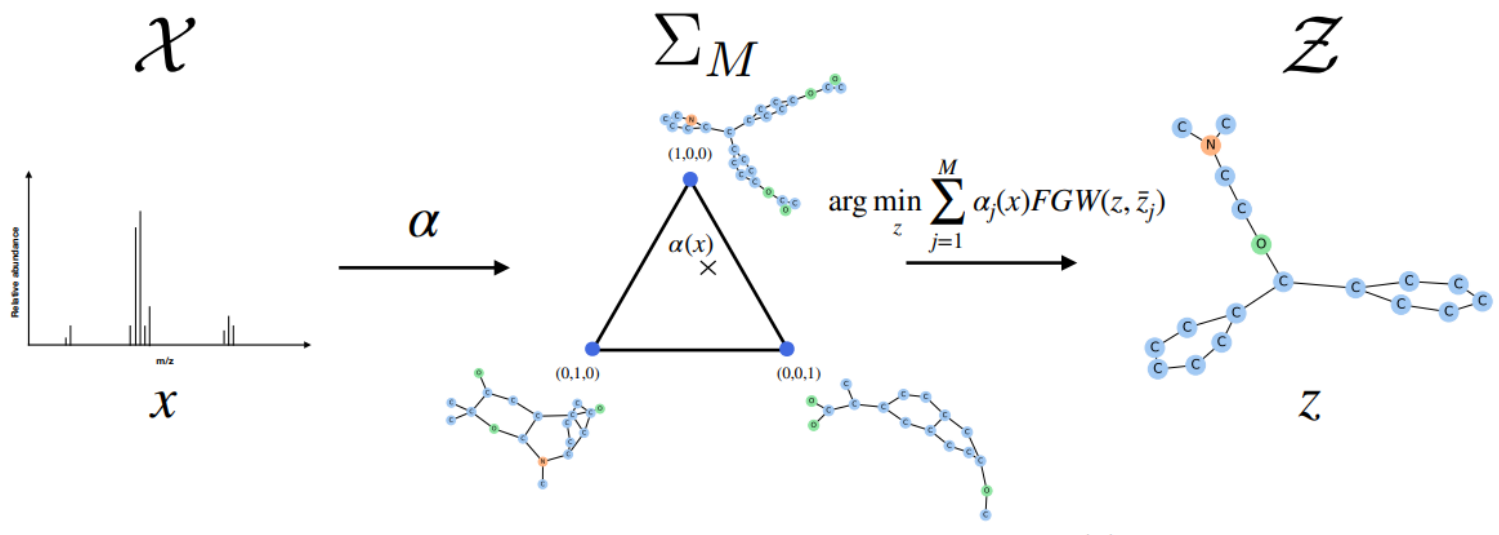
Supervised Graph prediction



Supervised graph prediction (a.k.a graph regression)

- Objective : learn a function f predicting a graph g from an input x.
- Applications of SGP:
  - knowledge graph extraction [Melnyk et al., 2022]
  - Natural language processing [Dozat and Manning, 2017]
  - Molecule identification in chemistry [Brouard et al., 2016]
- Surrogate based methods [Brouard et al., 2016, El Ahmad et al., 2024]:
  - Represent graph as a vector in a high dimensional space (RKHS).
  - Learn a mapping from input to this space.
  - Decode the vector to a graph (e.g. search among finite candidates).
- Linear regression of Adjacency matrix [Calissano et al., 2022].

Structured prediction with conditional FGW barycenters



Structured prediction with GW barycenter [Brogat-Motte et al., 2022]

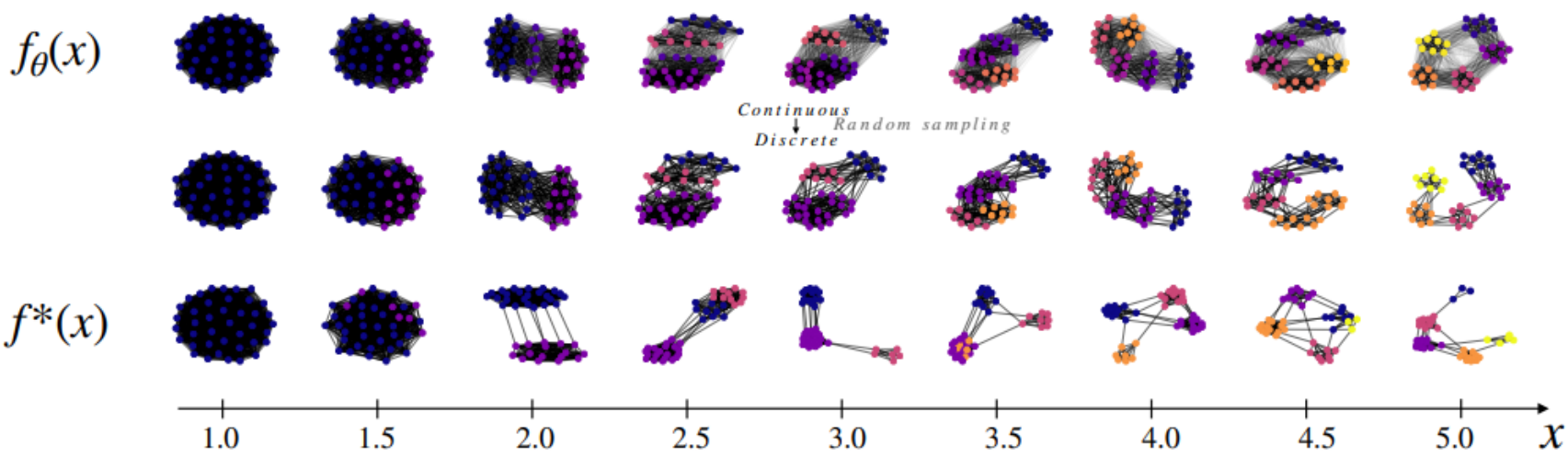
$$f(\mathbf{x}) = \widehat{\mathbf{C}}(\mathbf{w}(\mathbf{x})) = \mathrm{argmin}_{\mathbf{C}} \sum_k w_k(\mathbf{x}) GW(\mathbf{C}, \overline{\mathbf{C}_i})$$

- Prediction of the graph with a GW barycenter with weights conditioned by $\mathbf{x}$.
- Dictionary $\{\overline{\mathbf{C}_k}\}_k$ and conditional weights $\mathbf{w}(x)$ learned simultaneously with

$$\min_{\{\overline{\mathbf{C}_k}\}_k, \mathbf{w}(\cdot)} \quad \frac{1}{N} \sum_i GW(f(\mathbf{x}_i), \mathbf{C}_i)$$

- Both parametric and non parametric estimators [Brogat-Motte et al., 2022].
- Very powerful but slow at training and prediction due to barycenter computation.

Structured prediction with conditional FGW barycenters



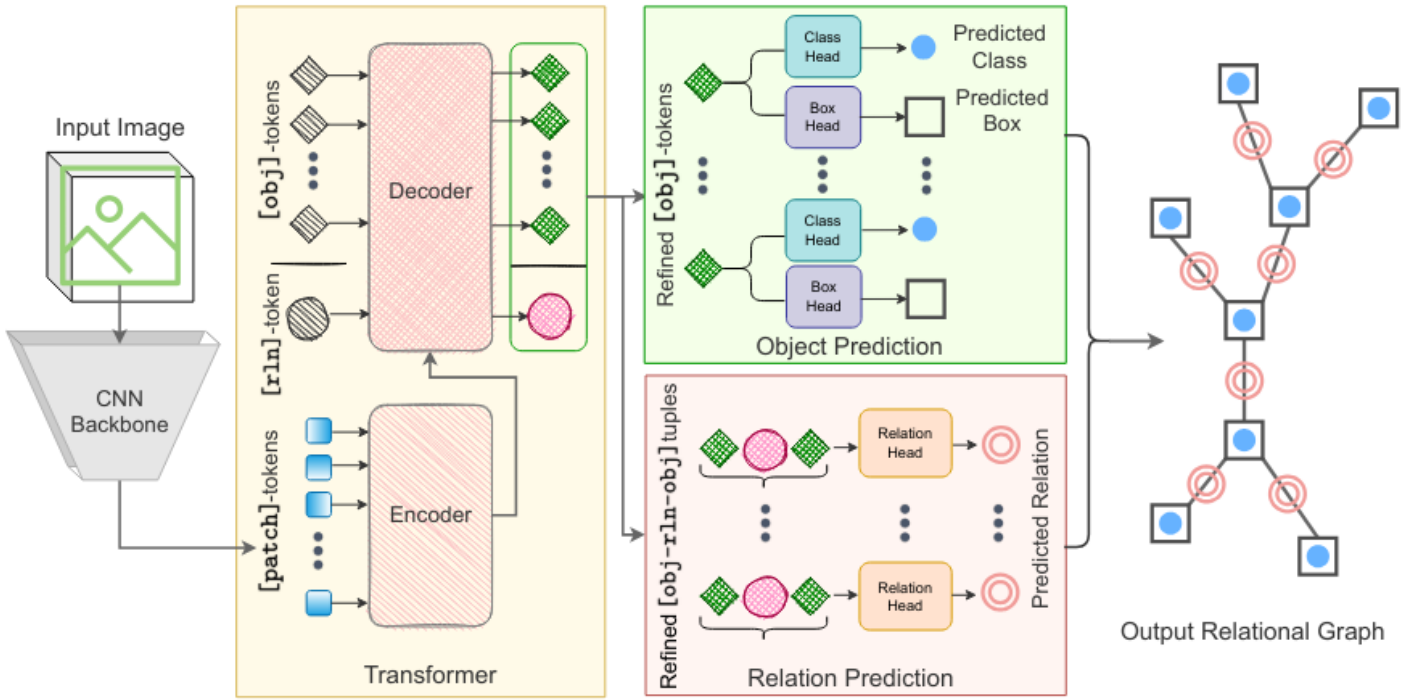
Structured prediction with GW barycenter [Brogat-Motte et al., 2022]

$$f(\mathbf{x}) = \widehat{\mathbf{C}}(\mathbf{w}(\mathbf{x})) = \text{argmin}_{\mathbf{C}} \sum_k w_k(\mathbf{x}) GW(\mathbf{C}, \overline{\mathbf{C}_i})$$

- Prediction of the graph with a GW barycenter with weights conditioned by $\mathbf{x}$.
- Dictionary $\{\overline{\mathbf{C}_k}\}_k$ and conditional weights $\mathbf{w}(x)$ learned simultaneously with

$$\min_{\{\overline{\mathbf{C}_k}\}_k, \mathbf{w}(\cdot)} \quad \frac{1}{N} \sum_i GW(f(\mathbf{x}_i), \mathbf{C}_i)$$

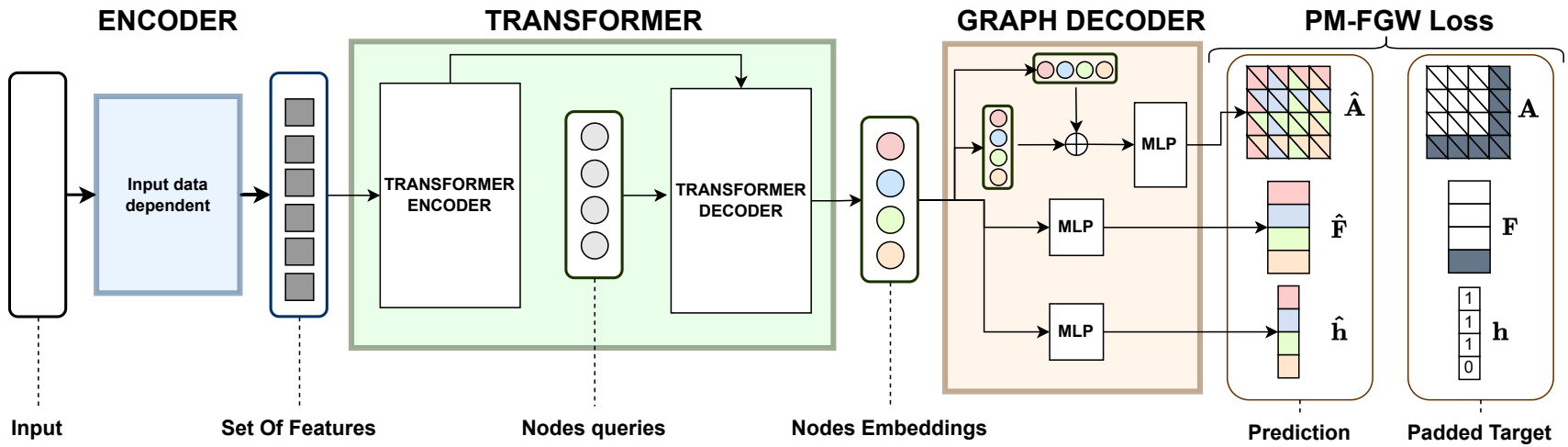
- Both parametric and non parametric estimators [Brogat-Motte et al., 2022].
- Very powerful but slow at training and prediction due to barycenter computation.



Relationformer [Shit et al., 2022]

- Predict a graph of max size M and activation scores for nodes to keep.
- Encoder-Decoder Transformer to predict node embeddings.
- Loss solves linear assignment problem (Hungarian) and uses assignment in quadratic loss between graphs of same size (padding the target).
- Fast prediction (thresholding) of graphs but focused on Image2Graph.

Any2Graph framework



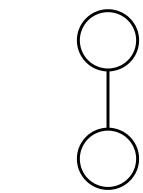
Principle [Krzakala et al., 2024]

- End-to-end supervised graph prediction with a deep learning framework.
- Learning optimization problem:

$$\min_{\theta} \quad \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f_\theta(x_i), \mathcal{P}(g_i)). \tag{1}$$

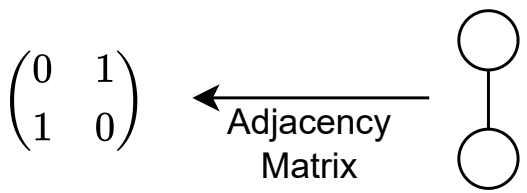
- $\{x_i, g_i\}$ are the input/output training data and $\mathcal{P}$ is a padding operator.
- f_θ is a transformer neural network with fixed max number of nodes M.
- f_θ also predicts is a padding vector $\hat{\boldsymbol{h}}$ (selection of subset of nodes).
- $\mathcal{L}$ is an optimal transport based loss for permutation invariant prediction.

End-to-end SGP pipeline

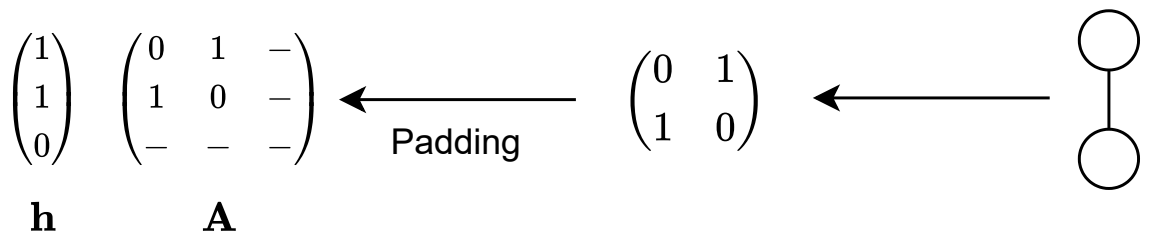


Target Graph

End-to-end SGP pipeline



End-to-end SGP pipeline

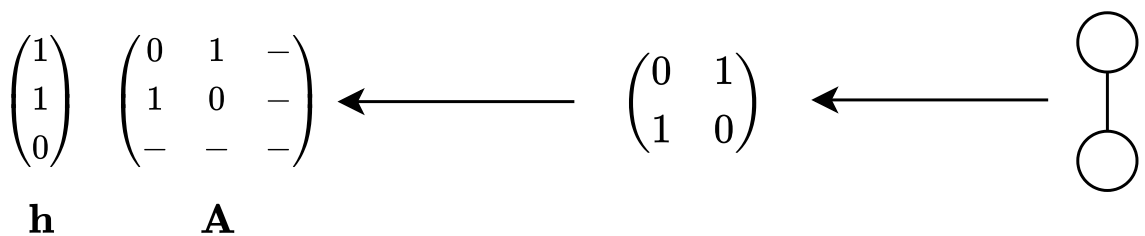


- Pad target graphs to have same size M.

End-to-end SGP pipeline

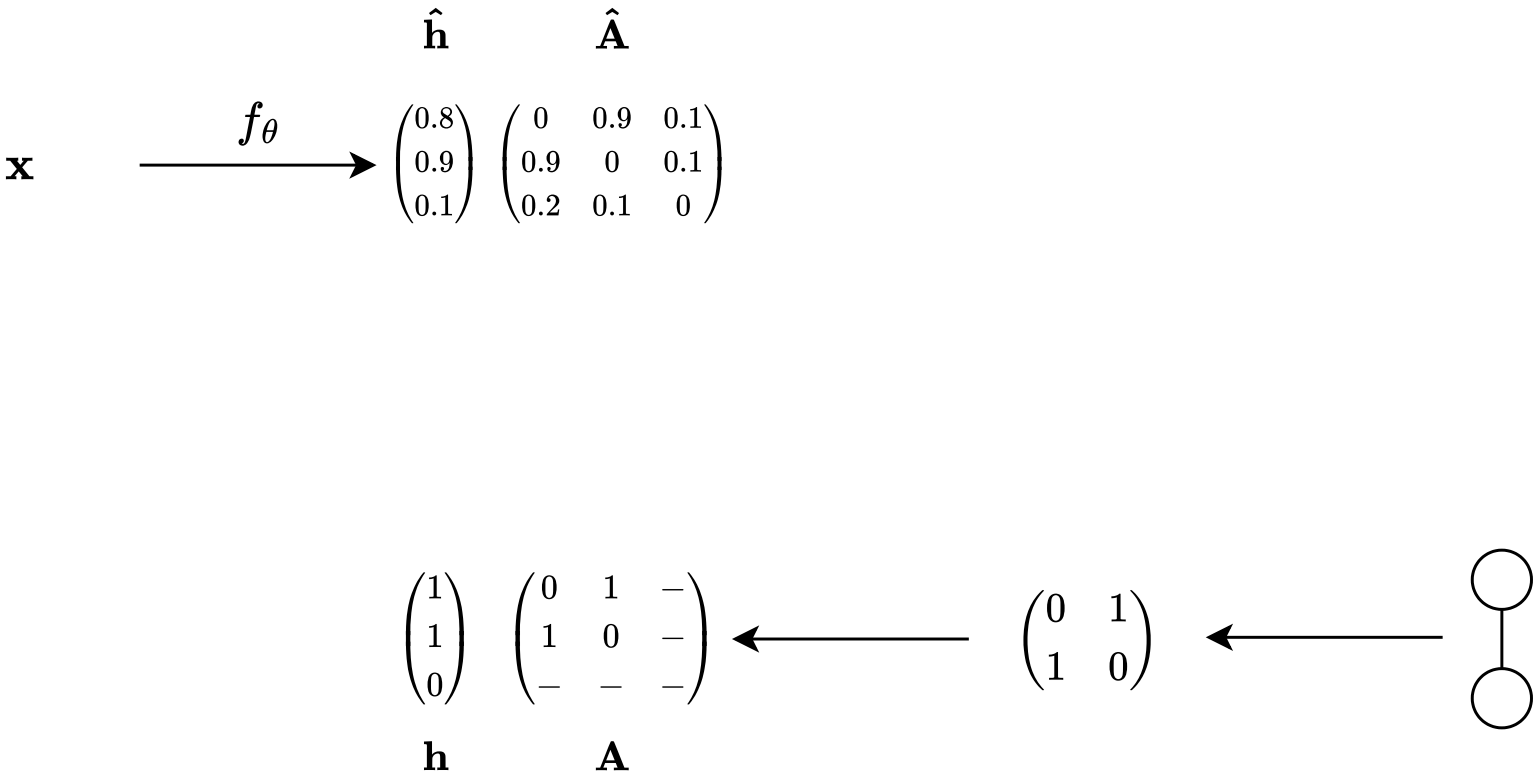
Input

$\mathbf{x}$



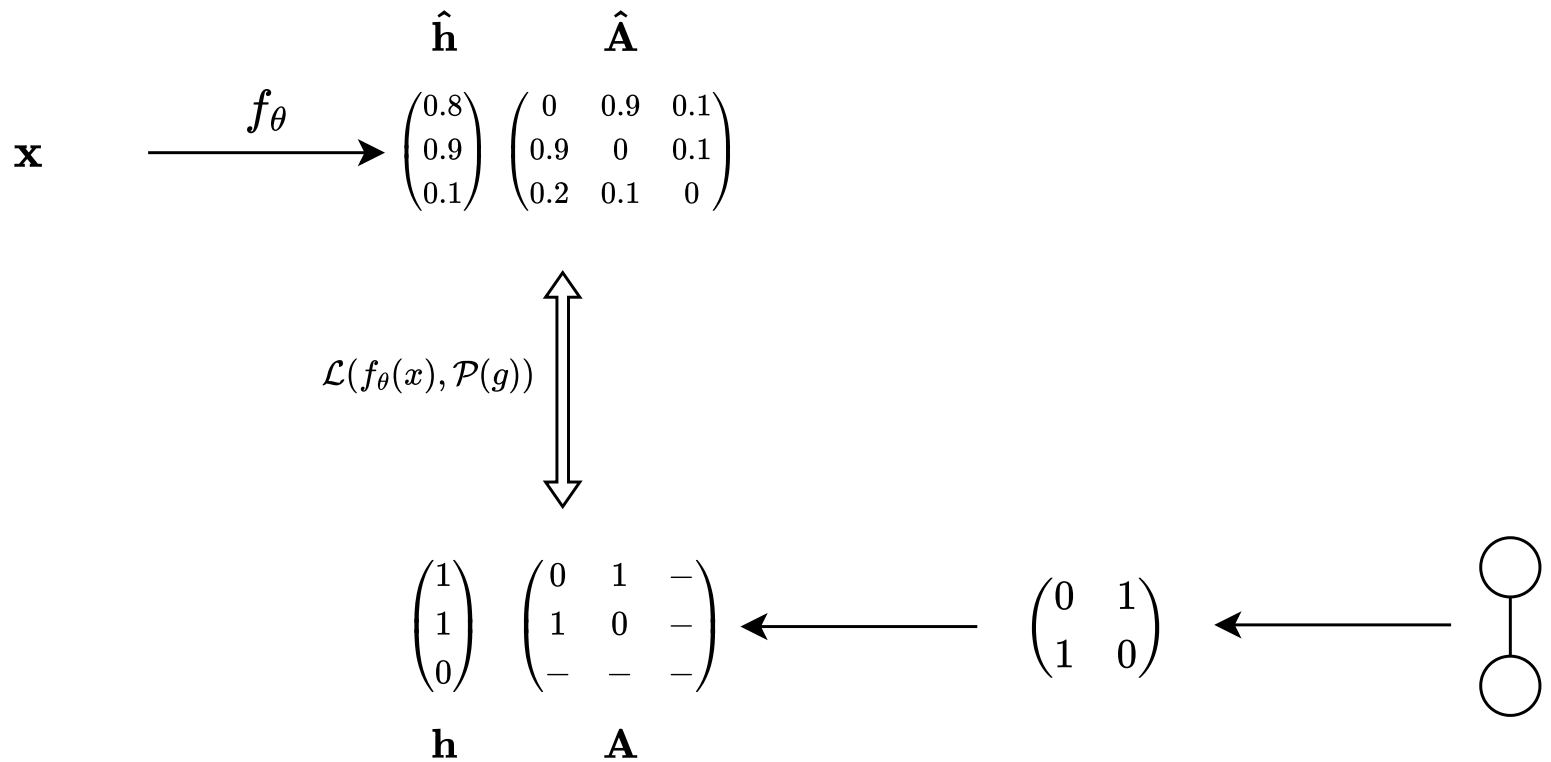
- Pad target graphs to have same size M.

End-to-end SGP pipeline



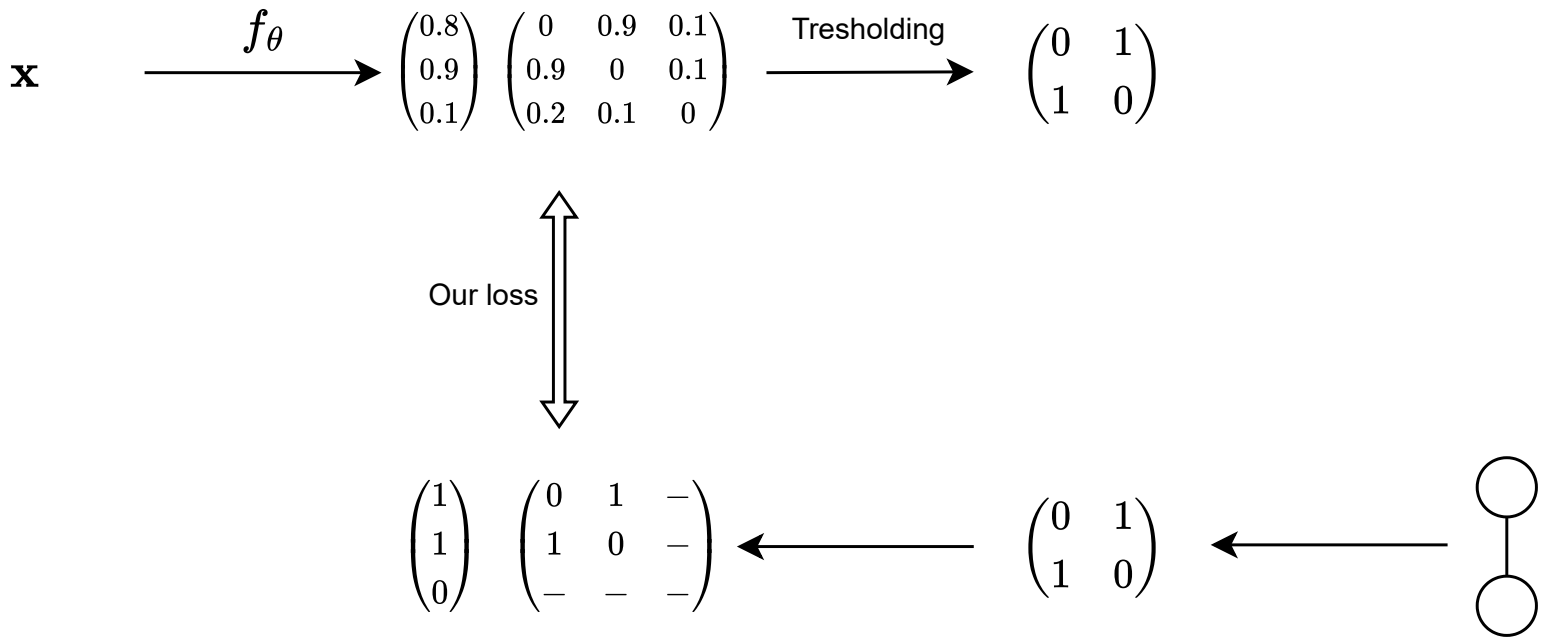
- Pad target graphs to have same size M.
- Predict with f_θ (continuous) size M graph with padding vector $\hat{\boldsymbol{h}}$.

End-to-end SGP pipeline



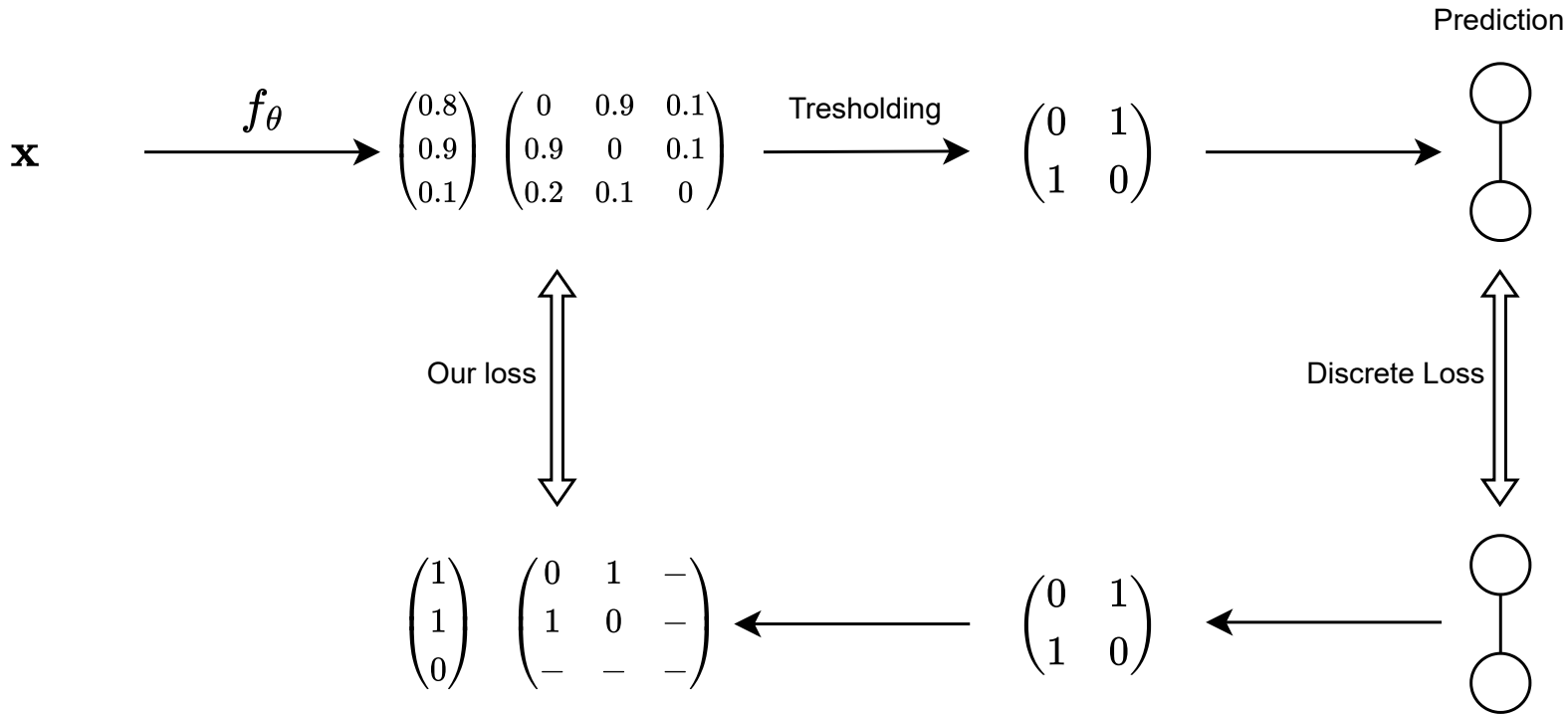
- Pad target graphs to have same size M.
- Predict with f_θ (continuous) size M graph with padding vector $\hat{\boldsymbol{h}}$.

End-to-end SGP pipeline



- Pad target graphs to have same size M.
- Predict with f_θ (continuous) size M graph with padding vector $\hat{\boldsymbol{h}}$.
- Minimize OT loss L between predicted and padded target graphs.

End-to-end SGP pipeline



- Pad target graphs to have same size M.
- Predict with f_θ (continuous) size M graph with padding vector $\hat{\boldsymbol{h}}$.
- Minimize OT loss L between predicted and padded target graphs.
- At test time, thresholding recovers discrete graph.

Definition of PM-FGW

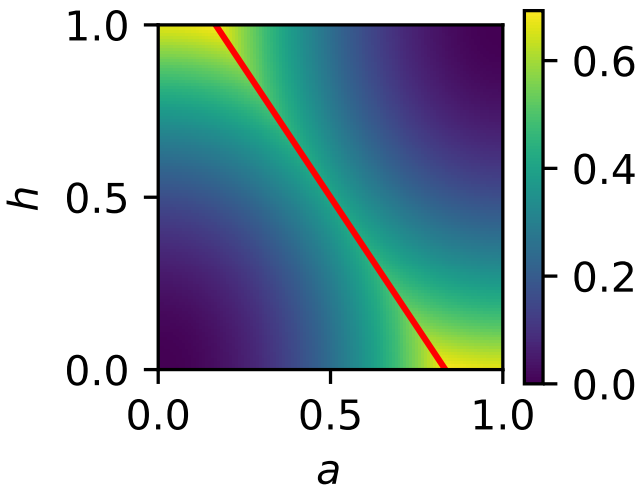
$$\text{PM-FGW}(\hat{y}, y) = \min_{\mathbf{T} \in \Pi_M} \mathcal{L}_{\mathbf{T}}(\hat{y}, y)$$

$$\begin{aligned}
\text{with } \mathcal{L}_{\mathbf{T}}(\hat{y}, y) = & \frac{\alpha_{\mathsf{h}}}{M} \sum_{i,j} T_{i,j} \ell_h(\hat{h}_i, h_j) && \text{Padding loss} \\
& + \frac{\alpha_{\mathsf{f}}}{m} \sum_{i,j} T_{i,j} \ell_f(\hat{\mathbf{f}}_i, \mathbf{f}_j) h_j && \text{Feature loss} \\
& + \frac{\alpha_{\mathsf{A}}}{m^2} \sum_{i,j,k,l} T_{i,j} T_{k,l} \ell_A(\hat{A}_{i,k}, A_{j,l}) h_j h_l. && \text{Structure loss}
\end{aligned}$$

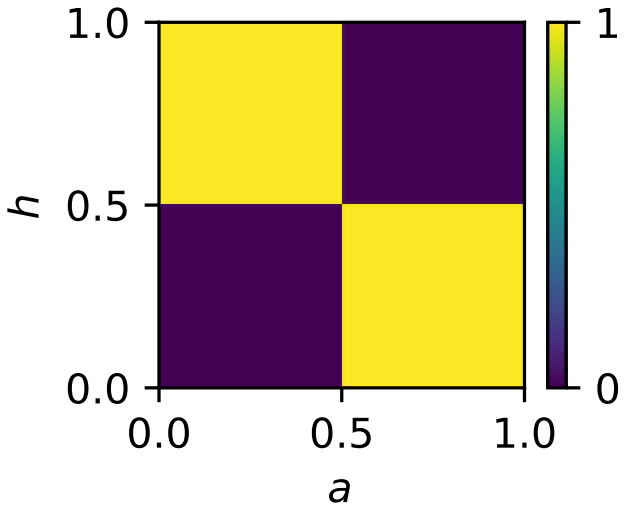
- ℓ_h, ℓ_f and ℓ_A are loss functions for node, feature and adjacency matrix discrepancies (Kullback-Leibler when target discrete, Squared loss when continuous feature).
- α_{h}, α_{f} and α_{A} are hyperparameters on the simplex.
- Loss is highly asymmetric due to the right masking by $\mathbf{h}$.
- Can be solved by Conditional Gradient with $O(M^3 \log M)$ iteration.

Illustration of PM-FGW loss

$\ell(a,h) = \mathcal{L}(\hat{y}_{a,h}, \mathcal{P}(g))$



$\ell(a,h) = \mathrm{ED}(\mathcal{P}^{-1}(\hat{y}_{a,h}), g)$



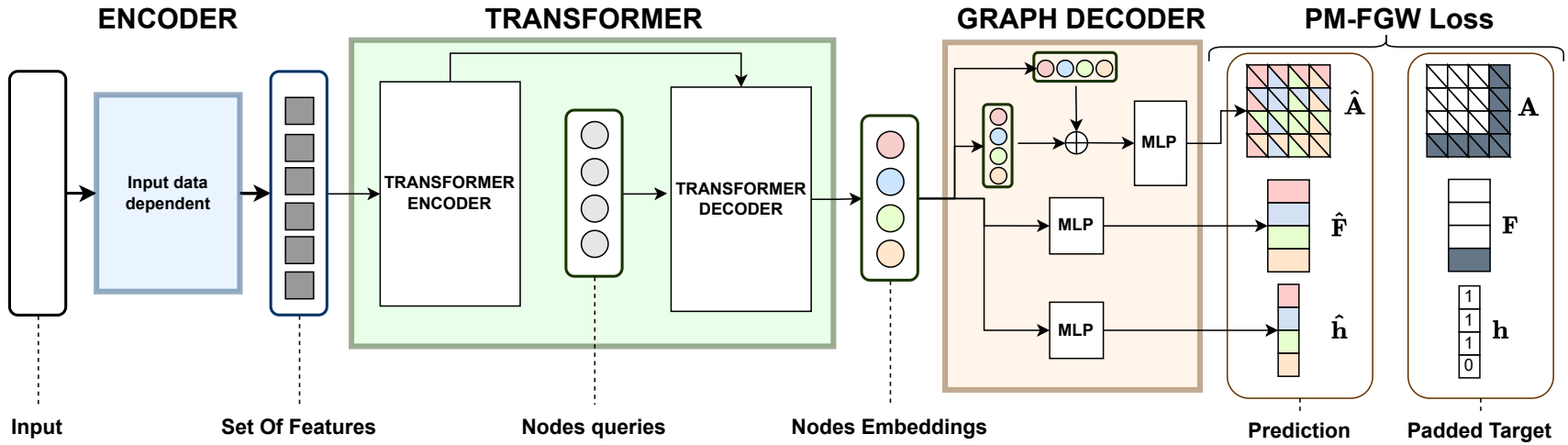
- The target graph is $g = (\mathbf{F}, \mathbf{A})$ with

$$\mathbf{F} = \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{pmatrix}; \mathbf{A} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

- The prediction $\hat{y}_{a,h} = (\hat{\mathbf{h}}, \hat{\mathbf{F}}, \hat{\mathbf{A}})$ is

$$\hat{\mathbf{h}} = \begin{pmatrix} 1 \\ h \\ 1-h \end{pmatrix}; \hat{\mathbf{F}} = \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_2 \end{pmatrix}; \hat{\mathbf{A}} = \begin{pmatrix} 0 & a & 1-a \\ a & 0 & 0 \\ 1-a & 0 & 0 \end{pmatrix}$$

Any2Graph Neural network architecture



- The **encoder** extract a set of features $x \to (\mathbf{V}_1, ..., \mathbf{V}_k) \in \mathbb{R}^{k \times d}$
- The **transformer** translate them into M nodes embedding $(\mathbf{Z}_1, ..., \mathbf{Z}_M) \to \in \mathbb{R}^{M \times d}$
- The **decoder** produce the graph following

$$\hat{h}_i = \sigma(\mathrm{MLP}_m(\mathbf{z}_i)) \qquad \forall i \in \{1, \ldots, M\}$$
$$\hat{F}_i = \mathrm{MLP}_f(\mathbf{z}_i) \qquad \forall i \in \{1, \ldots, M\}$$
$$\hat{A}_{i,j} = \sigma(\mathrm{MLP}_s(\mathbf{z}_i + \mathbf{z}_j)) \qquad \forall i, j \in \{1, \ldots, M\}^2$$

- Similar to Relationformer [Shit et al., 2022] but with symmetric adjacency matrix.

Any2Graph Prediction performances

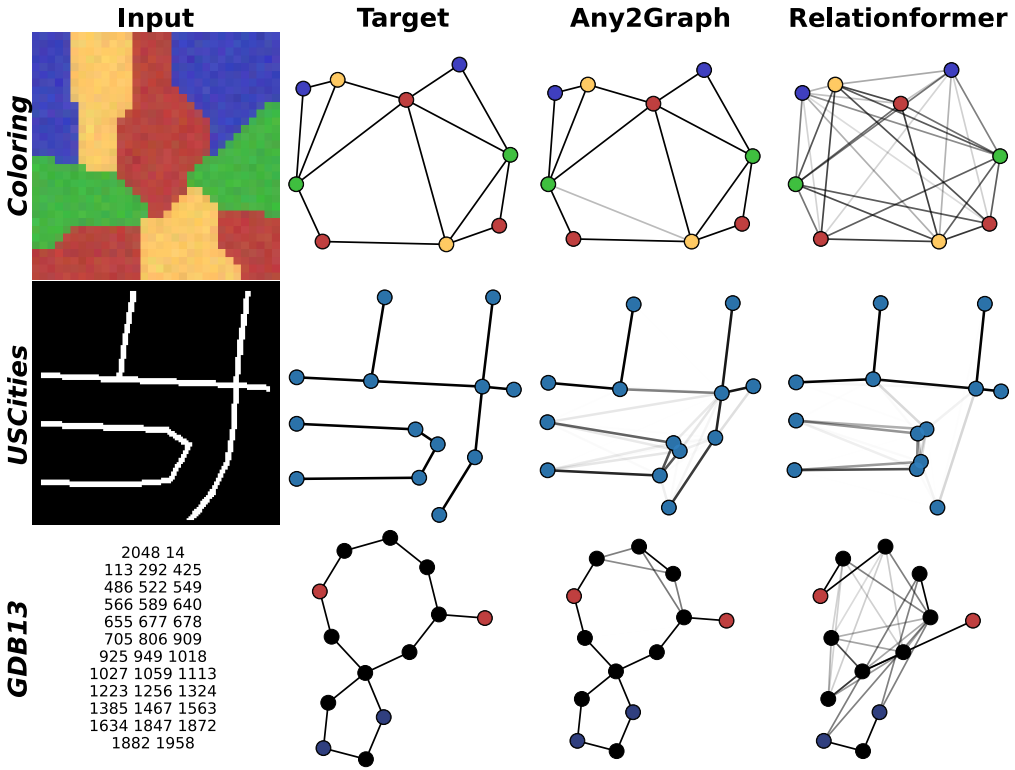


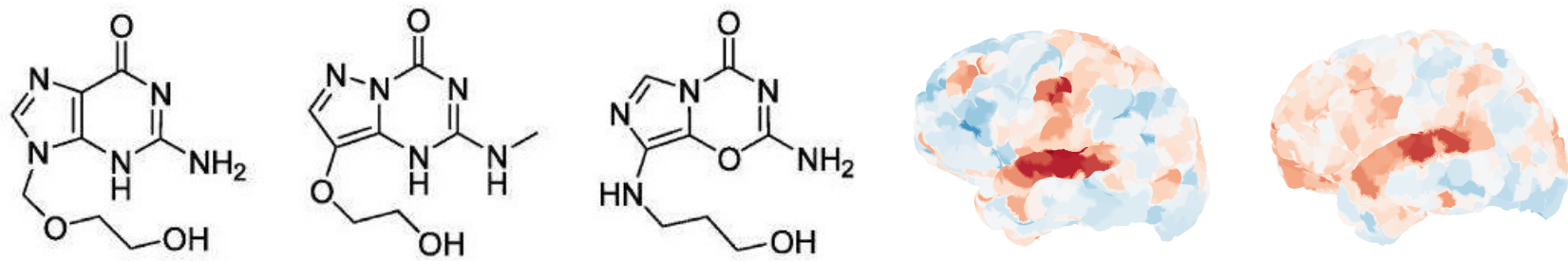
Figure 1: Qualitative comparison of Any2Graph (ours) and Relationformer.

Datasets	Model	Edit Distance ↓
Coloring	FGWBary-NN*	6.73
	Relationformer	5.47
	Any2Graph (Ours)	**0.20**
Toulouse	FGWBary-NN*	8.11
	Relationformer	**0.13**
	Any2Graph (Ours)	**0.13**
USCities	Relationformer	2.09
	Any2Graph (Ours)	**1.86**
QM9	FGWBary-ILE*	2.84
	Relationformer	3.80
	Any2Graph (Ours)	**2.13**
GDB13	Relationformer	8.83
	Any2Graph (Ours)	**3.63**

Table 1: Prediction performances measured with (test) edit distance.

Scaling graph OT solvers with neural networks

Challenges of Graph OT for large scale applications



Challenges

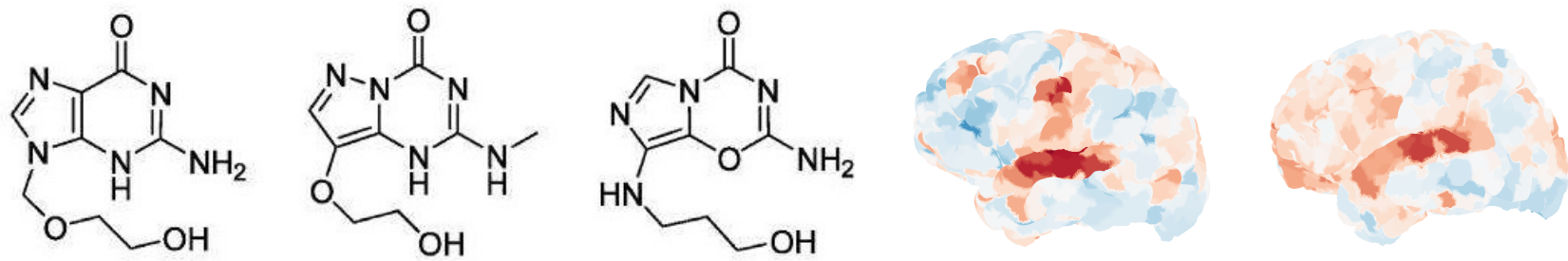
- OT solvers (GW/FGW) iter. scale cubically with the number of nodes.
- Large graphs (thousands of nodes) are too slow for many applications.
- Approximate entropic solvers exists [Peyré et al., 2016, Thual et al., 2022] but still slow and dense OT plans are sub-optimal for graphs.

Scaling OT on graphs with Neural Networks

$$\min_{\mathbf{T}} L_{OT}(\mathbf{T}, G, \hat{G}) \quad \Rightarrow \quad \min_{\theta} L_{OT}(\mathbf{T}_\theta, G, \hat{G})$$

- Learn to optimize with armortized optimization [Amos et al., 2022].
- Predicting the OT plan for large dataset of small graphs [Krzakala et al., 2025].
- Prediction the Unbalanced OT plan between large graphs [Mazelet et al., 2025].

Challenges of Graph OT for large scale applications



Challenges

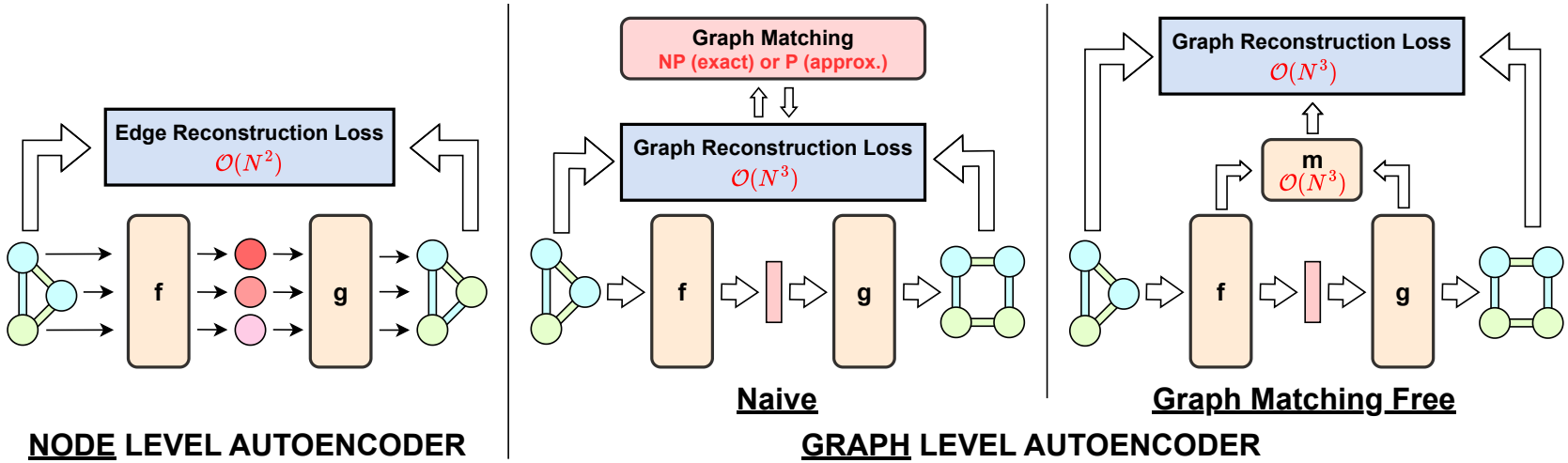
- OT solvers (GW/FGW) iter. scale cubically with the number of nodes.
- Large graphs (thousands of nodes) are too slow for many applications.
- Approximate entropic solvers exists [Peyré et al., 2016, Thual et al., 2022] but still slow and dense OT plans are sub-optimal for graphs.

Scaling OT on graphs with Neural Networks

$$\frac{1}{N}\sum_i \min_{\mathbf{T}} L_{OT}(\mathbf{T}, G^i, \hat{G}^i) \quad \Rightarrow \quad \min_{\theta} \frac{1}{N}\sum_i L_{OT}(\mathbf{T}_\theta(G^i, \hat{G}^i), G^i, \hat{G}^i)$$

- Learn to optimize with armortized optimization [Amos et al., 2022].
- Predicting the OT plan for large dataset of small graphs [Krzakala et al., 2025].
- Prediction the Unbalanced OT plan between large graphs [Mazelet et al., 2025].

GRAph Level autoEncoder (GRALE)



GRALE [Krzakala et al., 2025]

- Train a Graph Level AutoEncoder : Graph2Vec + Vec2Graph.
- Build on Any2Graph architecture for graph decoding [Krzakala et al., 2024].
- Use node embeddings to predict OT plans and optimize PM-FGW loss.
- Train simultaneously the Graphs encoder/decoder and the OT plan predictor.
- Use Evoformer [Jumper et al., 2021] for graph encoding and decoding (new).
- Train on large datasets of small graphs (Coloring, Molecules).

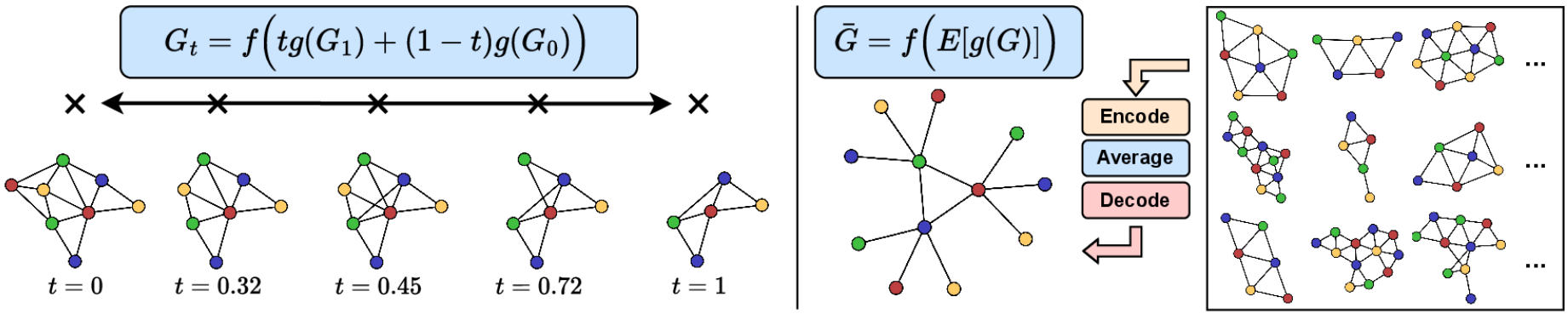
GRALE experiments

Model	COLORING		PUBCHEM 16		PUBCHEM 32	
	Edit. Dist. (↓)	GI Acc. (↑)	Edit. Dist. (↓)	GI Acc. (↑)	Edit. Dist. (↓)	GI Acc. (↑)
GraphVAE	2.13	35.90	3.72	07.8	N.A.	N.A.
PIGVAE*	0.09	85.30	1.69	41.0	2.53	24.91
GRALE	**0.02**	**99.20**	**0.11**	**93.0**	**0.78**	**66.80**

Numerical experiments

- GRALE outperforms state-of-the-art AE competitors on reconstruction and graph isomorphism accuracy.
- GRALE scales to large datasets of small graphs (80M graphs).
- GRALE learns a latent space where interpolation/averaging is possible.
- Embedding allows for semantic operations/editing on graphs.
- Pre-trained GRALE encoder/decoder improves downstream graph tasks (regression, classification, graph prediction).

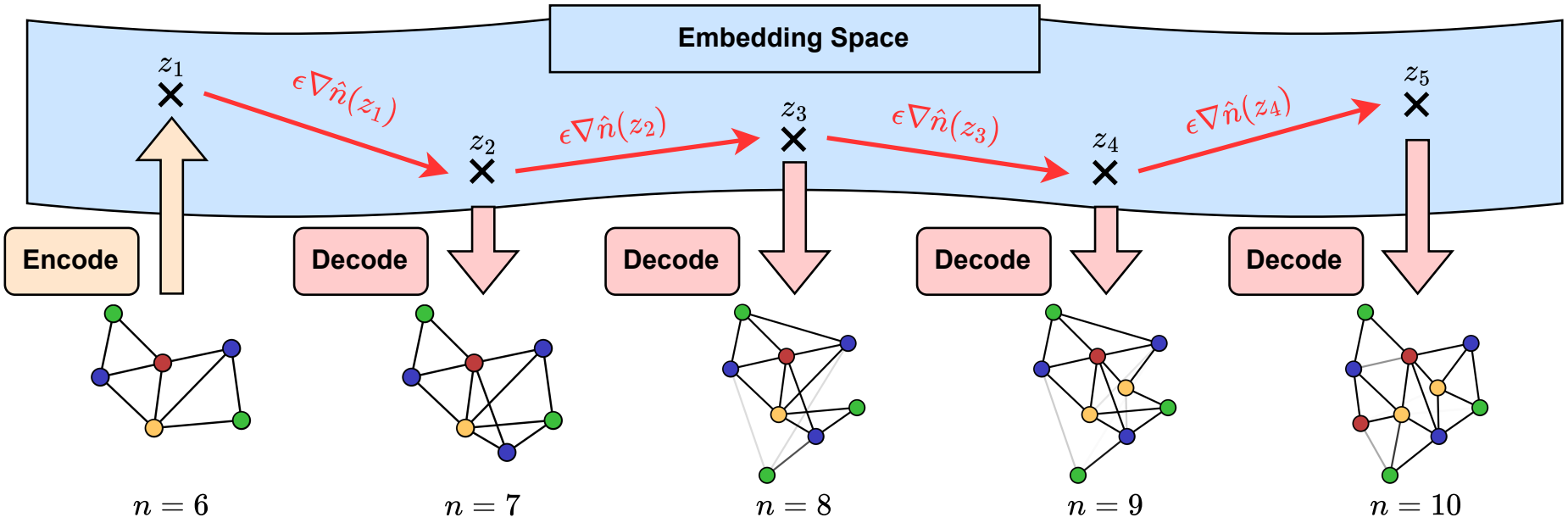
GRALE experiments



Numerical experiments

- GRALE outperforms state-of-the-art AE competitors on reconstruction and graph isomorphism accuracy.
- GRALE scales to large datasets of small graphs (80M graphs).
- GRALE learns a latent space where interpolation/averaging is possible.
- Embedding allows for semantic operations/editing on graphs.
- Pre-trained GRALE encoder/decoder improves downstream graph tasks (regression, classification, graph prediction).

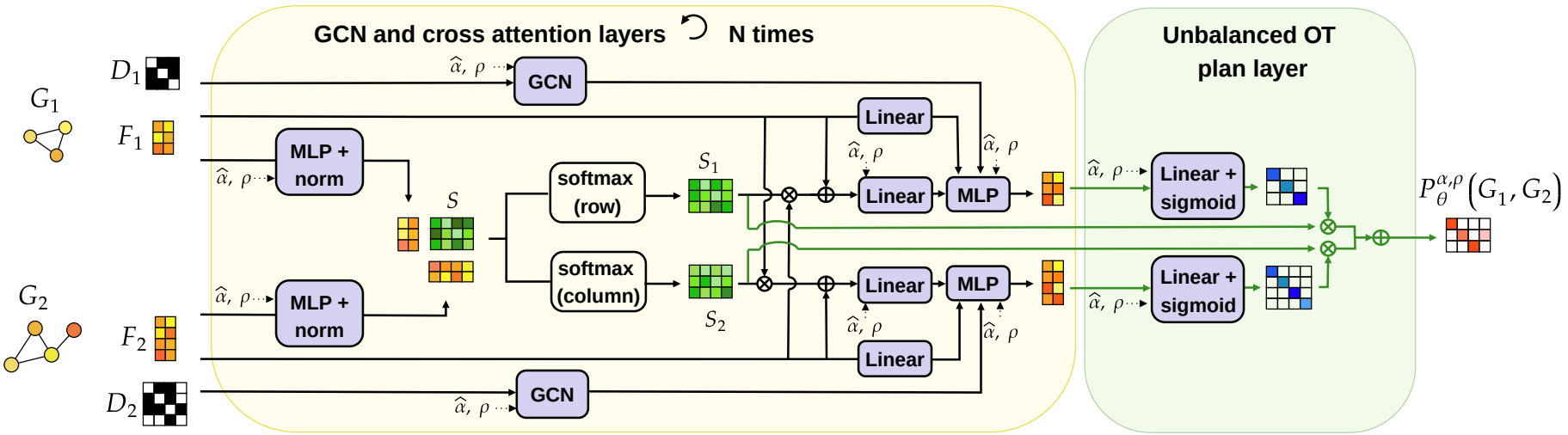
GRALE experiments



Numerical experiments

- GRALE outperforms state-of-the-art AE competitors on reconstruction and graph isomorphism accuracy.
- GRALE scales to large datasets of small graphs (80M graphs).
- GRALE learns a latent space where interpolation/averaging is possible.
- Embedding allows for semantic operations/editing on graphs.
- Pre-trained GRALE encoder/decoder improves downstream graph tasks (regression, classification, graph prediction).

Unsupervised learning of OT plan prediction (ULOT)



ULOT for solving FUGW [Mazelet et al., 2025]

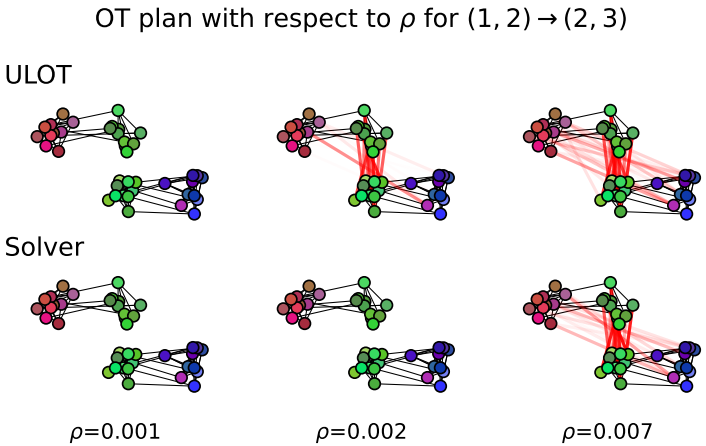
$$\min_{\mathbf{T}\geq 0} \quad \alpha \sum_{i,j,k,l} |D_{i,k} - D'_{j,l}|^2 T_{i,j}\, T_{k,l} + (1-\alpha) \sum_{i,j} C_{i,j} T_{i,j} + \rho(D(\mathbf{T}\mathbf{1}_m, \mathbf{a}) + D(\mathbf{T}^\top \mathbf{1}_n, \mathbf{b}))$$

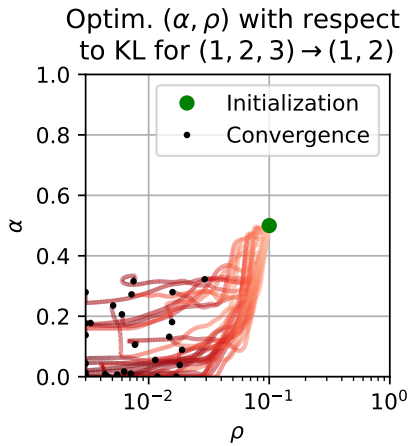
- Learn to predict Unbalanced OT plan $\mathbf{T}_\theta^{\alpha,\rho}(G, G')$ between large graphs.
- Use graph neural networks and Attention layers to parametrize OT plan.
- Optimize the FUGW loss over large dataset of graph pairs and parameters α, ρ:

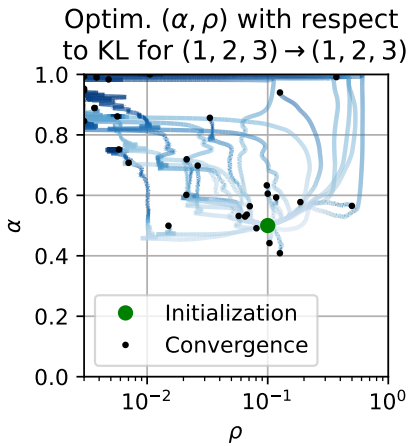
$$\min_\theta \quad xE_{\alpha,\rho,G,G'} \left[L_{FUGW}^{\alpha,\rho}(\mathbf{T}_\theta^{\alpha,\rho}(G, G'), G, G') \right]$$

- Provides after training a differentiable fast approximation of Unbalanced FGW for large graphs (thousands of nodes).

ULOT in practice



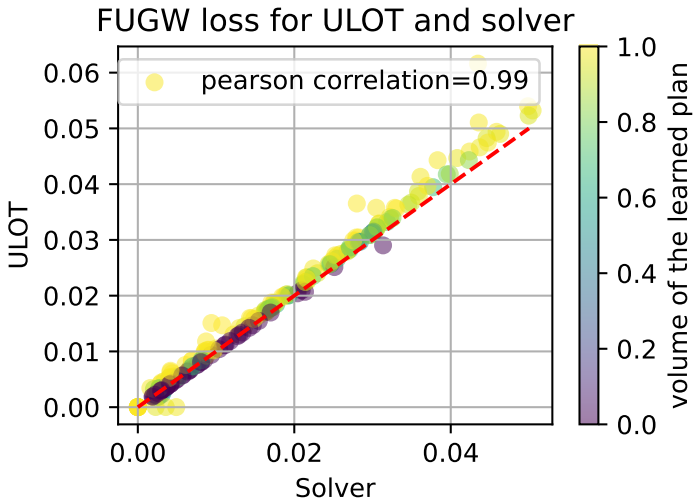


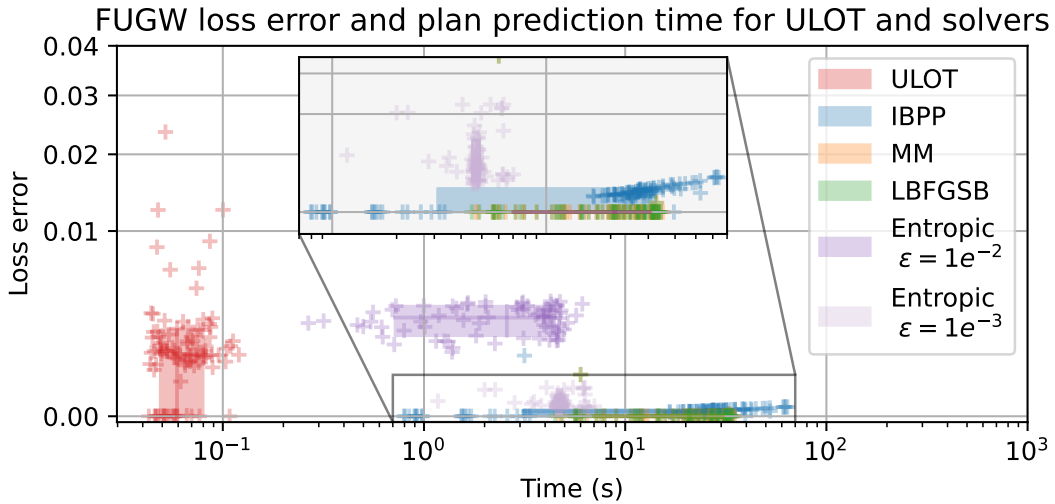


ULOT numerical experiments

- Trained on datases of simulated (SBM) and fMRI brain graphs (1000 nodes).
- Eficient computation of continuous regularization path in ρ, α.
- Differentiable OT layer wrt both input graphs and FUGW parameters ρ, α.
- Correlation of 0.99 with exact FUGW loss on test set (fMRI dataset).
- Much faster than entropic OT approximation (100x) with similar performance.
- Application on fMRI graph registration and prediction tasks.

ULOT in practice

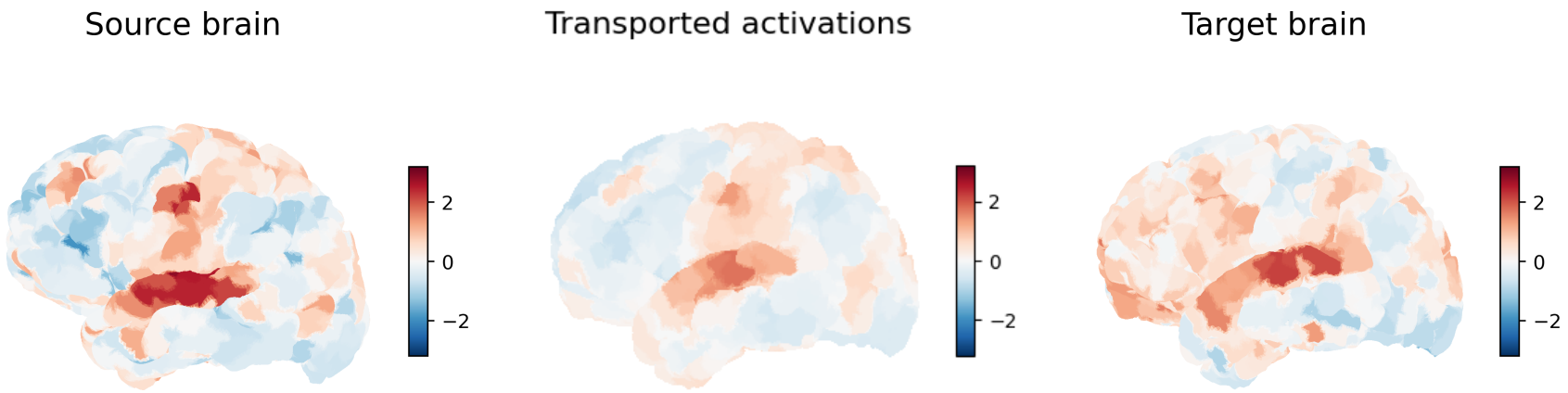




ULOT numerical experiments

- Trained on datases of simulated (SBM) and fMRI brain graphs (1000 nodes).
- Eficient computation of continuous regularization path in ρ, α.
- Differentiable OT layer wrt both input graphs and FUGW parameters ρ, α.
- Correlation of 0.99 with exact FUGW loss on test set (fMRI dataset).
- Much faster than entropic OT approximation (100x) with similar performance.
- Application on fMRI graph registration and prediction tasks.

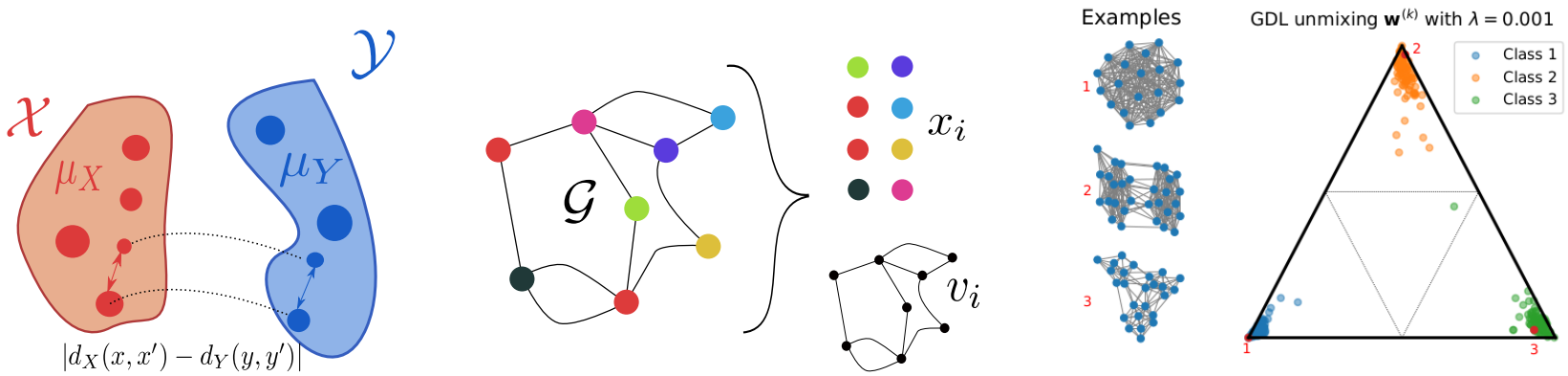
ULOT in practice



ULOT numerical experiments

- Trained on datases of simulated (SBM) and fMRI brain graphs (1000 nodes).
- Eficient computation of continuous regularization path in ρ, α.
- Differentiable OT layer wrt both input graphs and FUGW parameters ρ, α.
- Correlation of 0.99 with exact FUGW loss on test set (fMRI dataset).
- Much faster than entropic OT approximation (100x) with similar performance.
- Application on fMRI graph registration and prediction tasks.

Conclusion



Gromov-Wasserstein family for graph modeling

- Graphs modelled as distributions, $\mathcal{GW}$ can measure their similarity.
- Extensions of GW for labeled graphs and Frechet means can be computed.
- Weights on the nodes are important but rarely available : relax the constraints [Séjourné et al., 2020] or even remove one of them [Vincent-Cuaz et al., 2022a].
- Many applications of FGW from brain imagery [Thual et al., 2022] to Graph Neural Networks [Vincent-Cuaz et al., 2022b].
- OT is a powerful tool for (deep) graph structured prediction models [Brogat-Motte et al., 2022, Krzakala et al., 2024].
- Neural networks can help scale graph OT to large datasets or graphs [Krzakala et al., 2025, Mazelet et al., 2025].

Collaborators about OT on graphs


N. Courty


T. Vayer


L. Chapel


R. Tavenard


P. Krzakala


J. Yang


H. Tran


G. Gasso


M. Corneli


H. Van Assel


C. Vincent-Cuaz


S. Mazelet


A. Thual

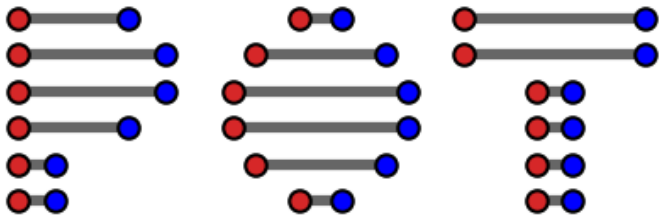

B. Thirion


F. d'Alché-Buc


L. Brogat-Motte


C. Laclau

Thank you



Doc : `https://pythonot.github.io/`

Code : `https://github.com/PythonOT/POT`

- OT LP solver, Sinkhorn (stabilized, GPU)
- Sliced OT, OT on sphere, Gaussian and Gaussian Mixture OT.
- Gromov-Wasserstein, Unbalanced.
- Barycenters, Wasserstein unmixing.
- Differentiable solvers for Numpy/Pytorch/tensorflow/Cupy

Course on OT for ML:

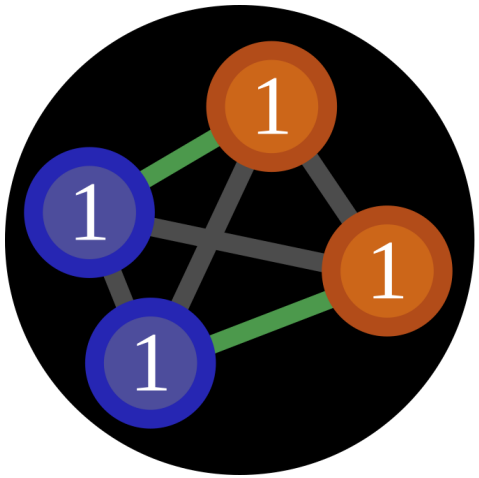
`https://tinyurl.com/otml-course`

Papers available on my website:

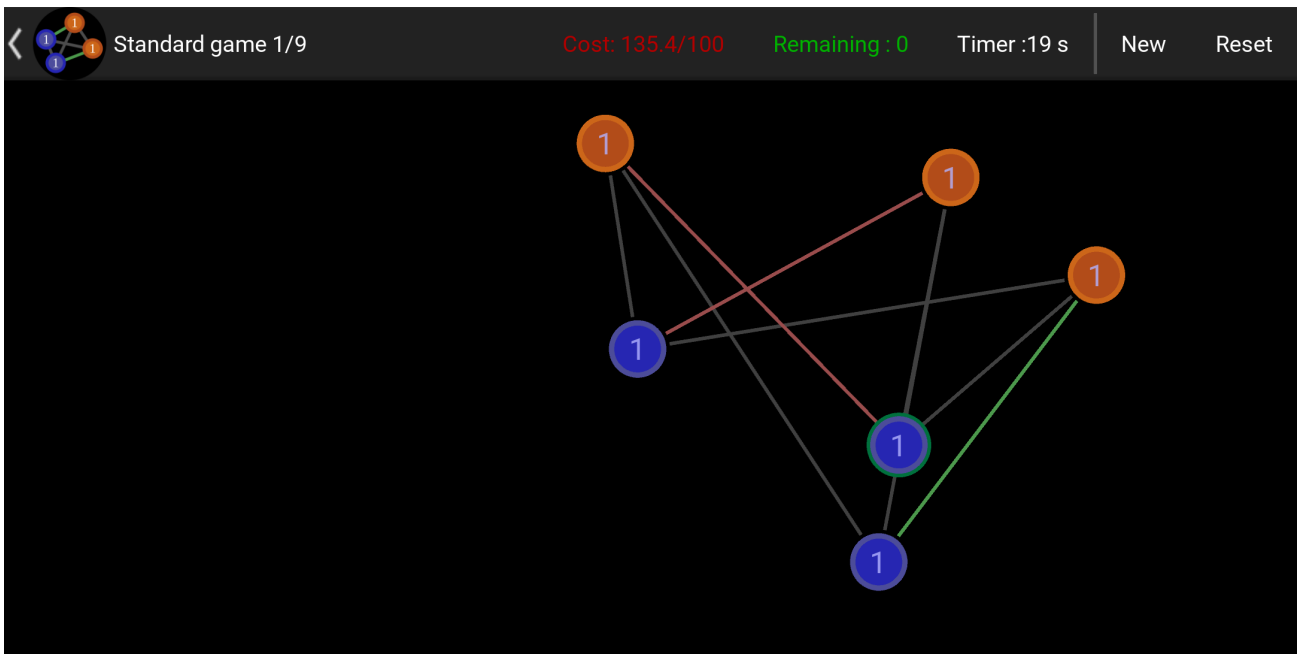
`https://remi.flamary.com/`

Looking for Msc interns or PhD students in Paris area!

OTGame (OT Puzzle game on android)

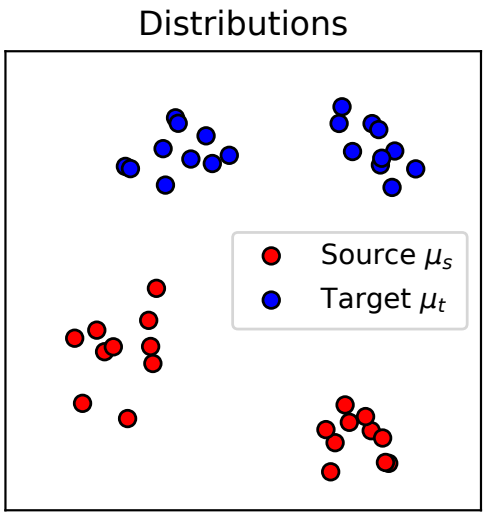


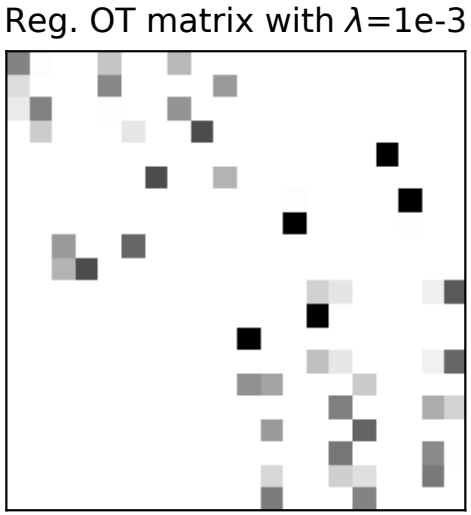
OTGame

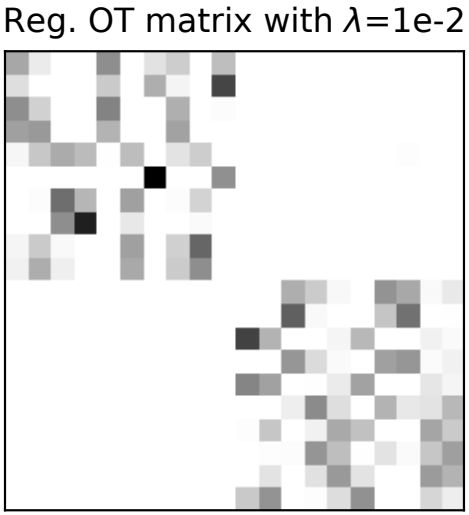


https://play.google.com/store/apps/details?id=com.flamary.otgame

Entropic regularized optimal transport





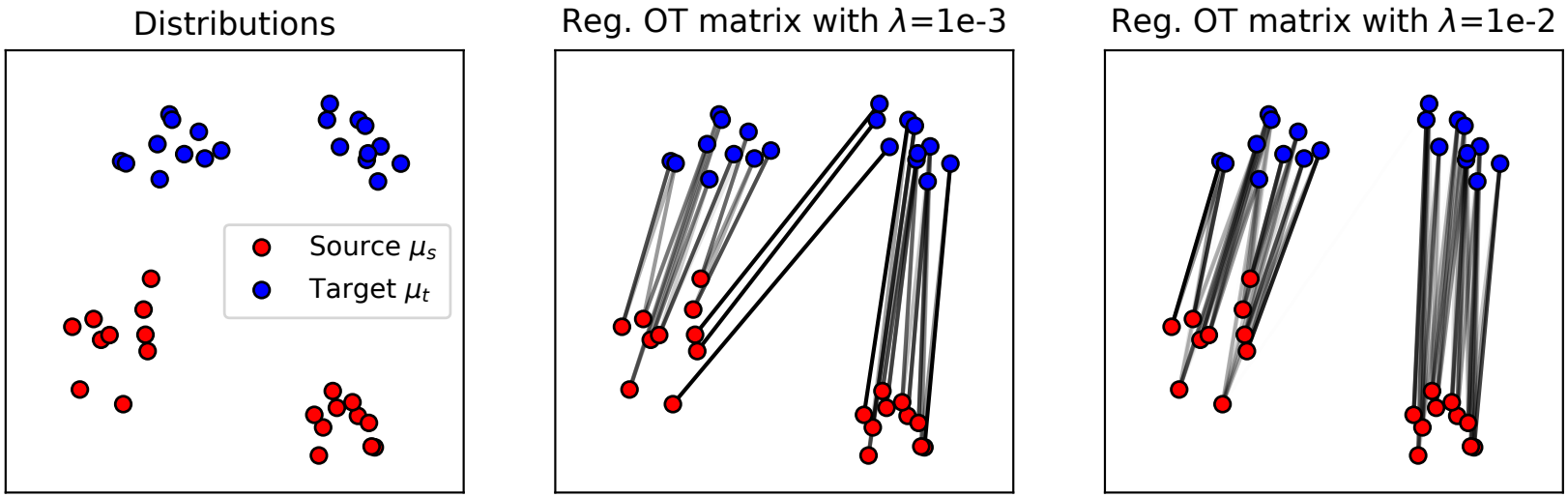


Entropic regularization [Cuturi, 2013]

$$W_\epsilon(\mu_s, \mu_t) = \min_{\mathbf{T} \in \Pi(\mu_s, \mu_t)} \quad \langle \mathbf{T}, \mathbf{C} \rangle_F + \epsilon \sum_{i,j} T_{i,j} \log T_{i,j}$$

- Regularization with the negative entropy $-H(\mathbf{T})$.
- Looses sparsity, but strictly convex optimization problem [Benamou et al., 2015].
- Can be solved with the very efficient Sinkhorn-Knopp matrix scaling algorithm.
- Loss and OT matrix are differentiable and have better statistical properties [Genevay et al., 2018].

Entropic regularized optimal transport



Entropic regularization [Cuturi, 2013]

$$W_\epsilon(\mu_s, \mu_t) = \min_{\mathbf{T} \in \Pi(\mu_s, \mu_t)} \quad \langle \mathbf{T}, \mathbf{C} \rangle_F + \epsilon \sum_{i,j} T_{i,j} \log T_{i,j}$$

- Regularization with the negative entropy $-H(\mathbf{T})$.
- Looses sparsity, but strictly convex optimization problem [Benamou et al., 2015].
- Can be solved with the very efficient Sinkhorn-Knopp matrix scaling algorithm.
- Loss and OT matrix are differentiable and have better statistical properties [Genevay et al., 2018].

Approximating GW in the linear embedding

GW Upper bond [Vincent-Cuaz et al., 2021]

Let two graphs of order N in the linear embedding $\left(\sum_s w_s^{(1)} \overline{\boldsymbol{D}_s}\right)$ and $\left(\sum_s w_s^{(2)} \overline{\boldsymbol{D}_s}\right)$, the $\mathcal{GW}$ divergence can be upper bounded by

$$\mathcal{GW}_2 \left(\sum_{s\in[S]} w_s^{(1)} \overline{\boldsymbol{D}_s}, \sum_{s\in[S]} w_s^{(2)} \overline{\boldsymbol{D}_s} \right) \leq \|\mathbf{w}^{(1)} - \mathbf{w}^{(2)}\|_{\boldsymbol{M}} \tag{2}$$

with $\boldsymbol{M}$ a PSD matrix of components $M_{p,q} = \left\langle \boldsymbol{D_h} \overline{\boldsymbol{D}_p}, \overline{\boldsymbol{D}_q} \boldsymbol{D_h} \right\rangle_F$, $\boldsymbol{D_h} = diag(\boldsymbol{h})$.

Discussion

- The upper bound is the value of GW for a transport $T = diag(\boldsymbol{h})$ assuming that the nodes are already aligned.
- The bound is exact when the weights $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ are close.
- Solving $\mathcal{GW}$ with FW si $O(N^3 \log(N))$ at each iterations.
- Computing the Mahalanobis upper bound is $O(S^2)$: very fast alterative to GW for nearest neighbors retrieval.

Solving the Gromov Wasserstein optimization problem

Optimization problem

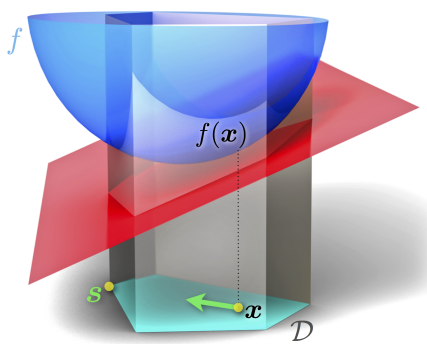
$$\mathcal{GW}_p^p(\mu_s, \mu_t) = \min_{\mathbf{T} \in \Pi(\mu_s, \mu_t)} \sum_{i,j,k,l} |D_{i,k} - D'_{j,l}|^p T_{i,j}\, T_{k,l}$$

with $\mu_s = \sum_i a_i \delta_{\mathbf{x}_i^s}$ and $\mu_t = \sum_j b_j \delta_{x_j^t}$ and $D_{i,k} = \|\mathbf{x}_i^s - \mathbf{x}_k^s\|, D'_{j,l} = \|\mathbf{x}_j^t - \mathbf{x}_l^t\|$

- Quadratic Program (Wasserstein is a linear program).
- Nonconvex, NP-hard, related to Quadratic Assignment Problem (QAP).
- Large problem and non convexity forbid standard QP solvers.

Optimization algorithms

- Local solution with conditional gradient algorithm (Frank-Wolfe) [Frank and Wolfe, 1956].
- Each FW iteration requires solving an OT problems.
- Gromov in 1D has a close form (solved in discrete with a sort) [Vayer et al., 2019c].
- With entropic regularization, one can use mirror descent [Peyré et al., 2016] or fast low rank approximations [Scetbon et al., 2021].



Optimization Problem

$$\mathcal{GW}^p_{p,\epsilon}(\mu_s, \mu_t) = \min_{\mathbf{T}\in\Pi(\mu_s,\mu_t)} \sum_{i,j,k,l} |D_{i,k} - D'_{j,l}|^p T_{i,j}\, T_{k,l} + \epsilon \sum_{i,j} T_{i,j} \log T_{i,j} \qquad (3)$$

with $\mu_s = \sum_i a_i \delta_{\mathbf{x}^s_i}$ and $\mu_t = \sum_j b_j \delta_{x^t_j}$ and $D_{i,k} = \|\mathbf{x}^s_i - \mathbf{x}^s_k\|, D'_{j,l} = \|\mathbf{x}^t_j - \mathbf{x}^t_l\|$

- Smoothing the original GW with a convex and smooth entropic term.

Solving the entropic $\mathcal{GW}$ [Peyré et al., 2016]

- Problem (3) can be solved using a KL mirror descent.
- This is equivalent to solving at each iteration t

$$\mathbf{T}^{(t+1)} = \min_{\mathbf{T}\in\mathcal{P}} \quad \left\langle \mathbf{T}, \mathbf{G}^{(t)} \right\rangle_F + \epsilon \sum_{i,j} T_{i,j} \log T_{i,j}$$

  Where $G^{(t)}_{i,j} = 2\sum_{k,l} |D_{i,k} - D'_{j,l}|^p T^{(t)}_{k,l}$ is the gradient of the GW loss at previous point $\mathbf{T}^{(k)}$.
- Problem above solved using a Sinkhorn-Knopp algorithm of entropic OT.
- Very fast approximation exist for low rank distances [Scetbon et al., 2021].

Solving the unmixing problem

Optimization problem

$$\min_{\mathbf{w}\in\Sigma_S} \quad \mathcal{GW}_2^2\left(\sum_{s\in[S]} w_s\overline{\boldsymbol{D}_s}\ ,\ \boldsymbol{D}\right) - \lambda\|\mathbf{w}\|_2^2$$

- Non-convex Quadratic Program *w.r.t.* $\mathbf{T}$ and $\mathbf{w}$.
- GW for fixed $\mathbf{w}$ already have an existing Frank-Wolfe solver.
- We proposed a Block Coordinate Descent algorithm

BCD Algorithm for sparse GW unmixing [Tseng, 2001]

1: **repeat**
2: Compute OT matrix $\boldsymbol{T}$ of $\mathcal{GW}_2^2(\boldsymbol{D}, \sum_s w_s\overline{\boldsymbol{D}_s})$, with FW [Vayer et al., 2018].
3: Compute the optimal $\mathbf{w}$ given $\boldsymbol{T}$ with Frank-Wolfe algorithm.
4: **until** convergence

- Since the problem is quadratic optimal steps can be obtained for both FW.
- BCD convergence in practice in a few tens of iterations.

GDL on labeled graphs

- For datasets with labeled graphs, on can learn simultaneously a dictionary of the structure $\{\overline{\boldsymbol{D}}_s\}_{s\in[S]}$ and a dictionary on the labels/features $\{\overline{\mathbf{F}}_s\}_{s\in[S]}$.
- Data fitting is Fused Gromov-Wasserstein distance $\mathcal{FGW}$, same stochastic algorithmm.

Dictionary on weights

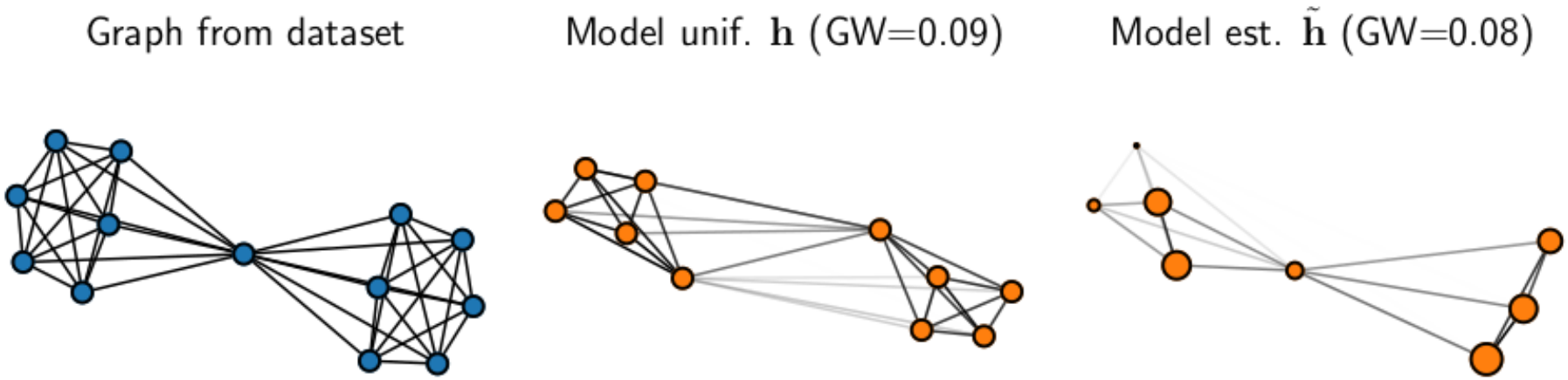
$$\min_{\substack{\{(\mathbf{w}^{(k)},\mathbf{v}^{(k)})\}_k \\ \{(\overline{\boldsymbol{D}}_s,\overline{\boldsymbol{h}_s})\}_s}} \sum_{k=1}^{K} \mathcal{GW}_2^2\left(\boldsymbol{D}^{(k)}, \sum_s w_s^{(k)}\overline{\boldsymbol{D}_s}, \boldsymbol{h}^{(k)}, \sum_s v_s^{(k)}\overline{\boldsymbol{h}_s}\right) - \lambda\|\mathbf{w}^{(k)}\|_2^2 - \mu\|\mathbf{v}^{(k)}\|_2^2$$

- We model the graphs as a linear model on the structure and the node weights

$$(\boldsymbol{D}^{(k)}, \boldsymbol{h}^{(k)}) \longrightarrow \quad \left(\sum_s w_s^{(k)}\boldsymbol{D}_s, \sum_s v_s^{(k)}\overline{\boldsymbol{h}_s}\right)$$

- This allows for sparse weights $\boldsymbol{h}$ so embedded graphs with different order.
- We provide in [Vincent-Cuaz et al., 2021] subgradients of GW *w.r.t.* the mass $\boldsymbol{h}$.

Experiments - Unsupervised representation learning



Comparison of fixed and learned weights dictionaries

- Graph taken from the IMBD dataset.
- Show original graph and representation after projection on the embedding.
- Uniform weight $\boldsymbol{h}$ has a hard time representing a central node.
- Estimated weights $\tilde{\boldsymbol{h}}$ recover a central node.
- In addition some nodes are discarded with 0 weight (graphs can change order).

Experiments - Clustering benchmark

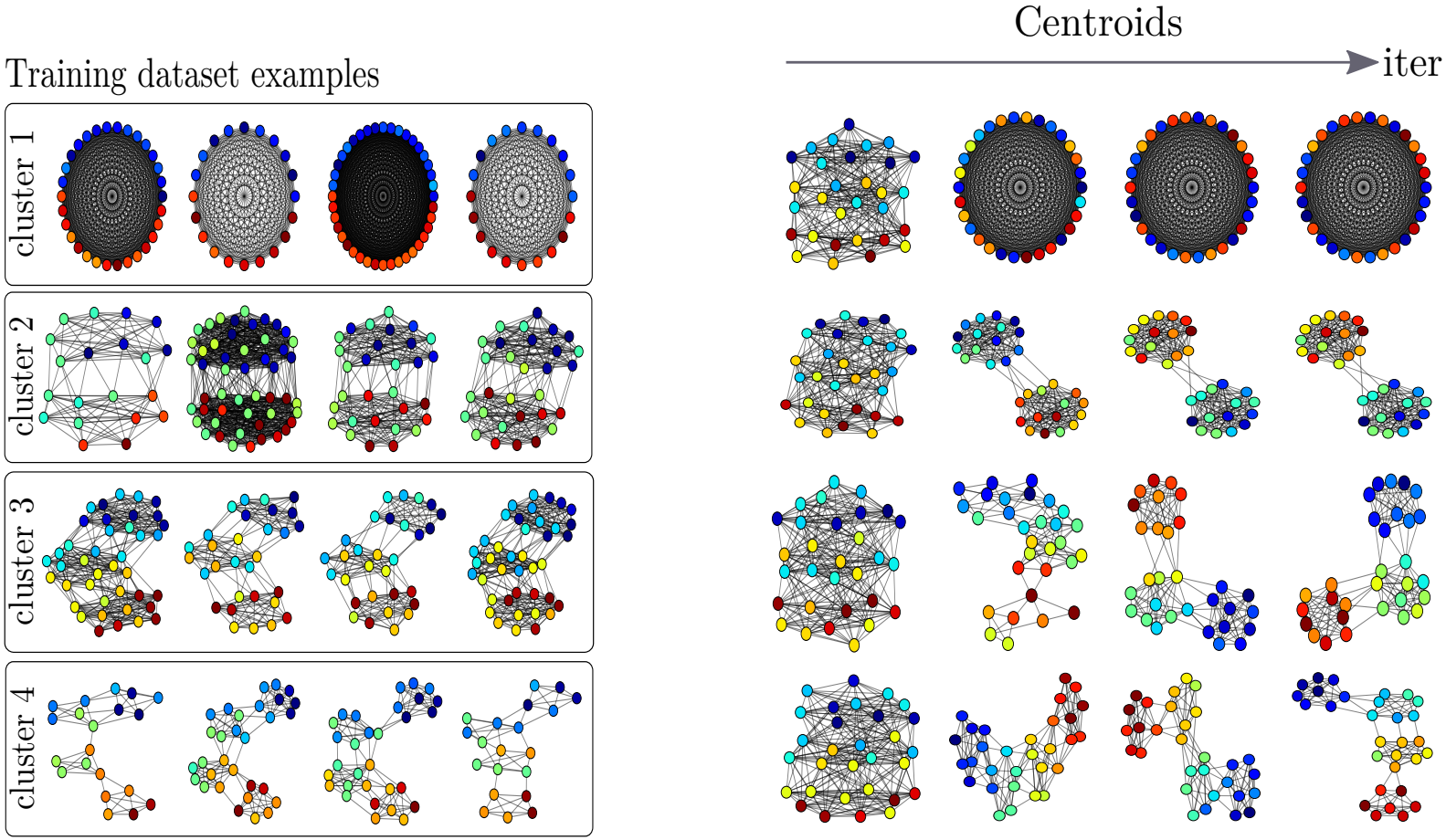
Table 1. Clustering: Rand Index computed for benchmarked approaches on real datasets.

	no attribute		discrete attributes		real attributes			
models	IMDB-B	IMDB-M	MUTAG	PTC-MR	BZR	COX2	ENZYMES	PROTEIN
GDL(ours)	**51.64(0.59)**	55.41(0.20)	**70.89(0.11)**	**51.90(0.54)**	**66.42(1.96)**	**59.48(0.68)**	66.97(0.93)	**60.49(0.71)**
GWF-r	51.24 (0.02)	**55.54(0.03)**	-	-	52.42(2.48)	56.84(0.41)	**72.13(0.19)**	59.96(0.09)
GWF-f	50.47(0.34)	54.01(0.37)	-	-	51.65(2.96)	52.86(0.53)	71.64(0.31)	58.89(0.39)
GW-k	50.32(0.02)	53.65(0.07)	57.56(1.50)	50.44(0.35)	56.72(0.50)	52.48(0.12)	66.33(1.42)	50.08(0.01)
SC	50.11(0.10)	54.40(9.45)	50.82(2.71)	50.45(0.31)	42.73(7.06)	41.32(6.07)	70.74(10.60)	49.92(1.23)

Clustering Experiments on real datasets

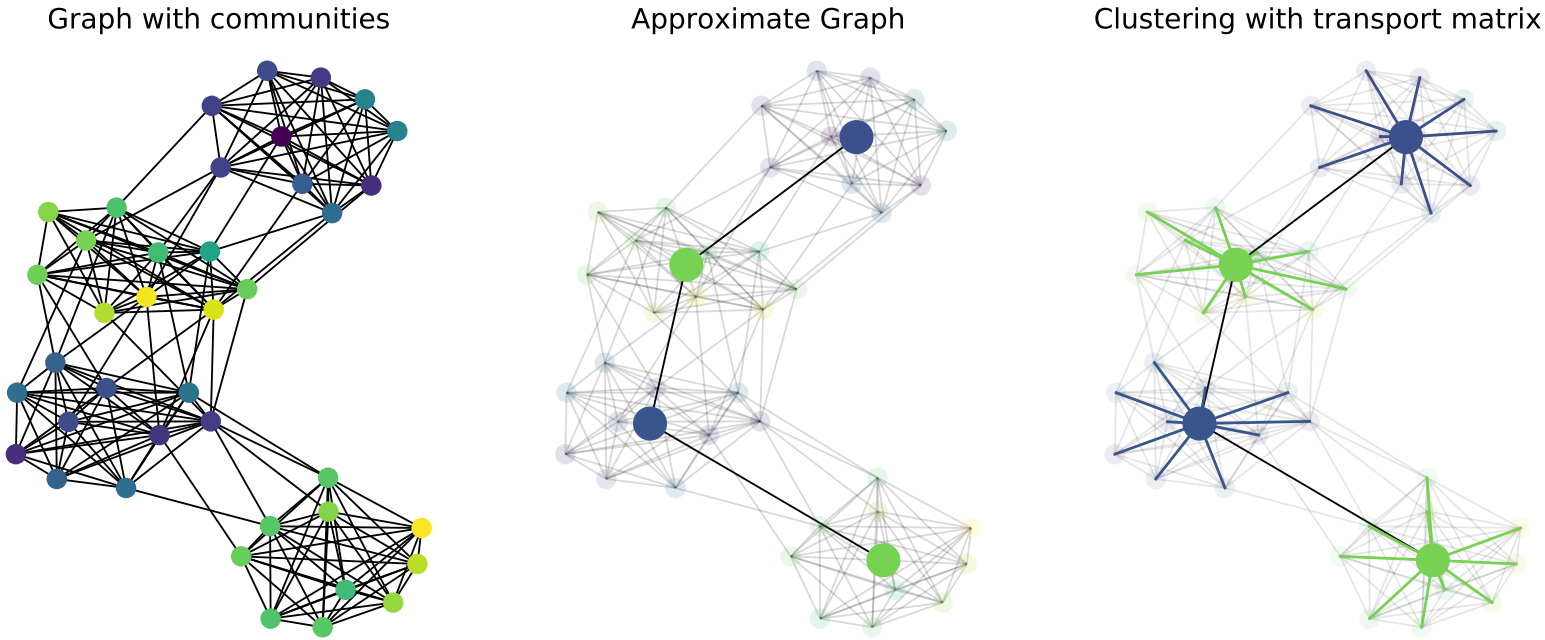
- Different data fitting losses:
  - Graphs without node attributes : Gromov-Wasserstein.
  - Graphs with node attributes (discrete and real): Fused Gromov-Wasserstein.
- We learn a dictionary on the dataset and perform K-means in the embedding using the Mahalanobis distance approximation.
- Compared to GW Factorization (GWF) [Xu, 2020] and spectral clustering.
- Similar performance for supervised classification (using GW in a kernel).

FGW for graphs based clustering



- Clustering of multiple real-valued graphs. Dataset composed of 40 graphs (10 graphs $\times$ 4 types of communities)
- k-means clustering using the FGW barycenter

FGW baryenter for community clustering

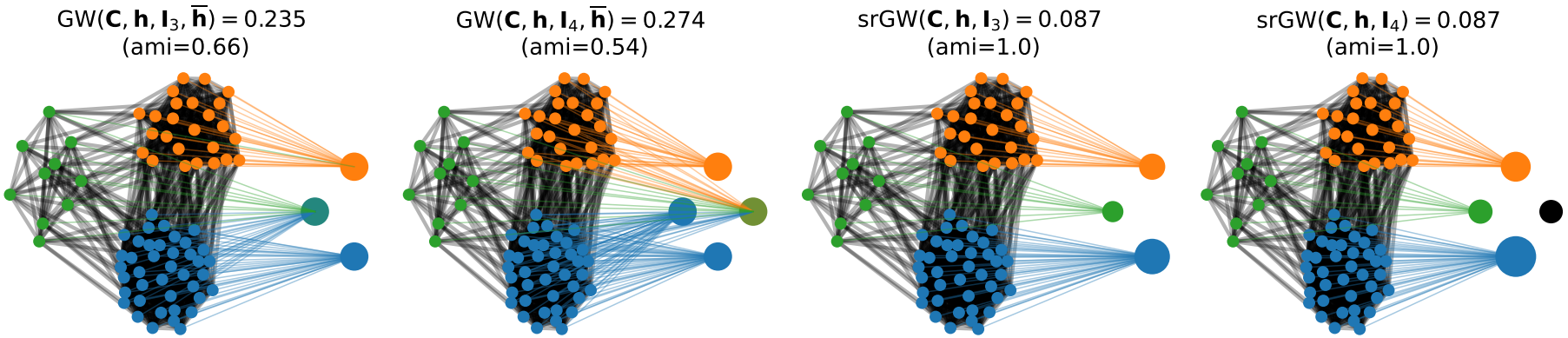


Graph approximation and community clustering [Vayer et al., 2018]

$$\min_{\mathbf{D},\mu} \quad \mathcal{FGW}(\mathbf{D}, \mathbf{D}_0, \mu, \mu_0)$$

- Approximate the graph $(\mathbf{D}_0, \mu_0)$ with a small number of nodes.
- OT matrix give the clustering affectation.
- Semi-relaxed GW estimates cluster proportions [Vincent-Cuaz et al., 2022a].
- Connections with spectral clustering [Chowdhury and Needham, 2021].
- Connections with dimensionality reduction [Van Assel et al., 2025].

FGW baryenter for community clustering

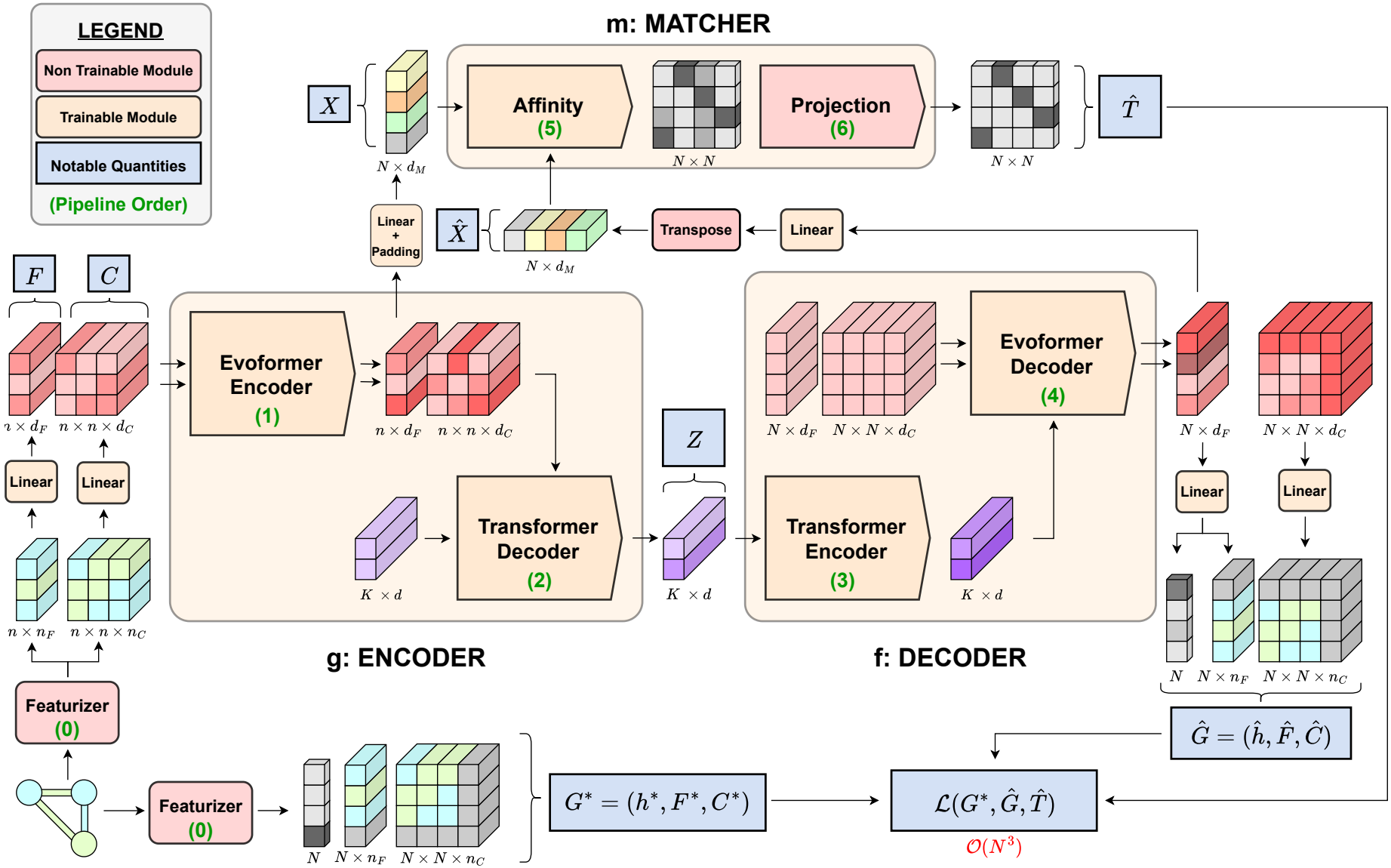


Graph approximation and community clustering [Vayer et al., 2018]

$$\min_{\mathbf{D},\mu} \quad \mathcal{FGW}(\mathbf{D}, \mathbf{D}_0, \mu, \mu_0)$$

- Approximate the graph $(\mathbf{D}_0, \mu_0)$ with a small number of nodes.
- OT matrix give the clustering affectation.
- Semi-relaxed GW estimates cluster proportions [Vincent-Cuaz et al., 2022a].
- Connections with spectral clustering [Chowdhury and Needham, 2021].
- Connections with dimensionality reduction [Van Assel et al., 2025].

GRALE Architecture



References i

Alvarez-Melis, D. and Jaakkola, T. S. (2018).

Gromov-wasserstein alignment of word embedding spaces.

arXiv preprint arXiv:1809.00013.

Amos, B., Cohen, S., Luise, G., and Redko, I. (2022).

Meta optimal transport.

arXiv preprint arXiv:2206.05262.

Benamou, J.-D., Carlier, G., Cuturi, M., Nenna, L., and Peyré, G. (2015).

Iterative Bregman projections for regularized transportation problems.

SISC.

Brogat-Motte, L., Flamary, R., Brouard, C., Rousu, J., and d'Alché Buc, F. (2022).

Learning to predict graphs with fused gromov-wasserstein barycenters.

In *International Conference in Machine Learning (ICML)*.

References ii

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017).

Geometric deep learning: going beyond euclidean data.

IEEE Signal Processing Magazine, 34(4):18–42.

Brouard, C., Shen, H., Dührkop, K., d'Alché-Buc, F., Böcker, S., and Rousu, J. (2016).

Fast metabolite identification with input output kernel regression.

Bioinformatics, 32(12):i28–i36.

Calissano, A., Feragen, A., and Vantini, S. (2022).

Graph-valued regression: Prediction of unlabelled networks in a non-euclidean graph space.

Journal of Multivariate Analysis, 190:104950.

Chapel, L., Alaya, M. Z., and Gasso, G. (2020).

Partial optimal tranport with applications on positive-unlabeled learning.

Advances in Neural Information Processing Systems, 33:2903–2913.

References iii

Chowdhury, S. and Needham, T. (2021).

Generalized spectral clustering via gromov-wasserstein learning.

In *International Conference on Artificial Intelligence and Statistics*, pages 712–720. PMLR.

Cuturi, M. (2013).

Sinkhorn distances: Lightspeed computation of optimal transportation.

In *Neural Information Processing Systems (NIPS)*, pages 2292–2300.

Dozat, T. and Manning, C. D. (2017).

Deep biaffine attention for neural dependency parsing.

In *International Conference on Learning Representations, ICLR*. OpenReview.net.

El Ahmad, T., Brogat-Motte, L., Laforgue, P., and d'Alché Buc, F. (2024).

Sketch in, sketch out: Accelerating both learning and inference for structured prediction with kernels.

In *International Conference on Artificial Intelligence and Statistics*, pages 109–117. PMLR.


Frank, M. and Wolfe, P. (1956).

An algorithm for quadratic programming.

Naval research logistics quarterly, 3(1-2):95–110.


Genevay, A., Chizat, L., Bach, F., Cuturi, M., and Peyré, G. (2018).

Sample complexity of sinkhorn divergences.

arXiv preprint arXiv:1810.02733.


Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Zidek, A., Potapenko, A., et al. (2021).

Highly accurate protein structure prediction with alphafold.

nature, 596(7873):583–589.


Krzakala, P., Melo, G., Laclau, C., d'Alché Buc, F., and Flamary, R. (2025).

The quest for the graph level autoencoder (grale).

In *Neural Information Processing Systems (NeurIPS)*.


Krzakala, P., Yang, J., Flamary, R., d'Alché Buc, F., Laclau, C., and Labeau, M. (2024).

Any2graph: Deep end-to-end supervised graph prediction with an optimal transport loss.


Ma, X., Chu, X., Wang, Y., Lin, Y., Zhao, J., Ma, L., and Zhu, W. (2023).

Fused gromov-wasserstein graph mixup for graph-level classifications.

In *Thirty-seventh Conference on Neural Information Processing Systems*.


Mazelet, S., Flamary, R., and Thirion, B. (2025).

Unsupervised learning for optimal transport plan prediction between unbalanced graphs.

In *Neural Information Processing Systems (NeurIPS)*.


Melnyk, I., Dognin, P., and Das, P. (2022).

Knowledge graph generation from text.

In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1610–1622.

Memoli, F. (2011).

Gromov wasserstein distances and the metric approach to object matching.

Foundations of Computational Mathematics, pages 1–71.

Peyré, G., Cuturi, M., and Solomon, J. (2016).

Gromov-wasserstein averaging of kernel and distance matrices.

In *ICML*, pages 2664–2672.

Scetbon, M., Klein, M., Palla, G., and Cuturi, M. (2023).

Unbalanced low-rank optimal transport solvers.

arXiv preprint arXiv:2305.19727.

Scetbon, M., Peyré, G., and Cuturi, M. (2021).

Linear-time gromov wasserstein distances using low rank couplings and costs.

arXiv preprint arXiv:2106.01128.

Séjourné, T., Vialard, F.-X., and Peyré, G. (2020).

The unbalanced gromov wasserstein distance: Conic formulation and relaxation.

arXiv preprint arXiv:2009.04266.

Shit, S., Koner, R., Wittmann, B., Paetzold, J., Ezhov, I., Li, H., Pan, J., Sharifzadeh, S., Kaissis, G., Tresp, V., et al. (2022).

Relationformer: A unified framework for image-to-graph generation.

In *European Conference on Computer Vision*, pages 422–439. Springer.

Solomon, J., Peyré, G., Kim, V. G., and Sra, S. (2016).

Entropic metric alignment for correspondence problems.

ACM Transactions on Graphics (TOG), 35(4):72.

Thual, A., Tran, H., Zemskova, T., Courty, N., Flamary, R., Dehaene, S., and Thirion, B. (2022).

Aligning individual brains with fused unbalanced gromov-wasserstein.

In *Neural Information Processing Systems (NeurIPS)*.


Togninalli, M., Ghisu, E., Llinares-López, F., Rieck, B., and Borgwardt, K. (2019).

Wasserstein weisfeiler-lehman graph kernels.

Advances in neural information processing systems, 32.


Tseng, P. (2001).

Convergence of a block coordinate descent method for nondifferentiable minimization.

Journal of optimization theory and applications, 109(3):475–494.


Van Assel, H., Vincent-Cuaz, C., Courty, N., Flamary, R., Frossard, P., and Vayer, T. (2025).

Distributional reduction: Unifying dimensionality reduction and clustering with gromov-wasserstein projection.

Transactions of Machine Learning Research (TMLR).

 Vayer, T., Chapel, L., Flamary, R., Tavenard, R., and Courty, N. (2018).
Fused gromov-wasserstein distance for structured objects: theoretical foundations and mathematical properties.

 Vayer, T., Chapel, L., Flamary, R., Tavenard, R., and Courty, N. (2019a).
Optimal transport for structured data with application on graphs.
In *International Conference on Machine Learning (ICML)*.

 Vayer, T., Courty, N., Tavenard, R., and Flamary, R. (2019b).
Optimal transport for structured data with application on graphs.
In *International Conference on Machine Learning*, pages 6275–6284. PMLR.

 Vayer, T., Flamary, R., Tavenard, R., Chapel, L., and Courty, N. (2019c).
Sliced gromov-wasserstein.
In *Neural Information Processing Systems (NeurIPS)*.


Vincent-Cuaz, C., Flamary, R., Corneli, M., Vayer, T., and Courty, N. (2022a).
Semi-relaxed gromov wasserstein divergence with applications on graphs.
In *International Conference on Learning Representations (ICLR)*.


Vincent-Cuaz, C., Flamary, R., Corneli, M., Vayer, T., and Courty, N. (2022b).
Template based graph neural network with optimal transport distances.
In *Neural Information Processing Systems (NeurIPS)*.


Vincent-Cuaz, C., Vayer, T., Flamary, R., Corneli, M., and Courty, N. (2021).
Online graph dictionary learning.
In *International Conference on Machine Learning (ICML)*.


Xu, H. (2020).
Gromov-wasserstein factorization models for graph clustering.
In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6478–6485.