Spatial Similarity as Graph Comparison Problem

Casper van Engelenburg

A bit about me

BSc Applied Physics, Delft

Thesis *DNA Sequence Reading through a Double Nanopore*

Minor Interactive Environments (Industrial Design Engineering)

MSc Systems and Control (Mechanical Engineering)

Thesis *Automated Detection of Malaria in Blood*

Head Engineer | TU Delft Solar Boat Team

Teacher | Deep Learning

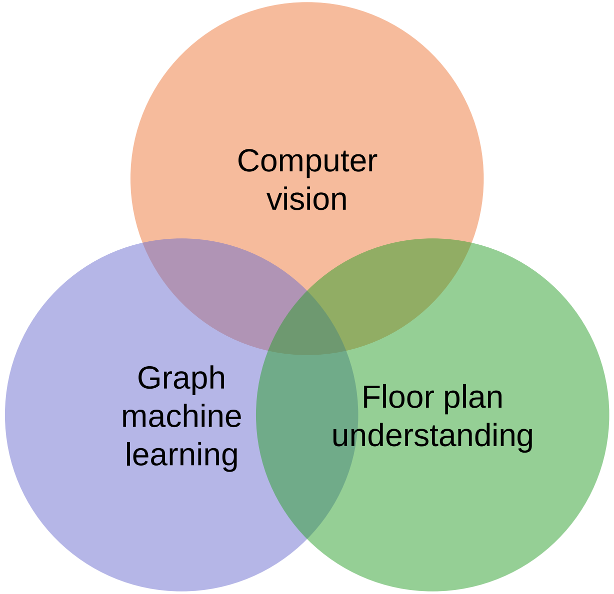
Now PhD Machine Learning in Architecture (5^{th} year, out of 5)

@ AiDAPT lab (Architecture)

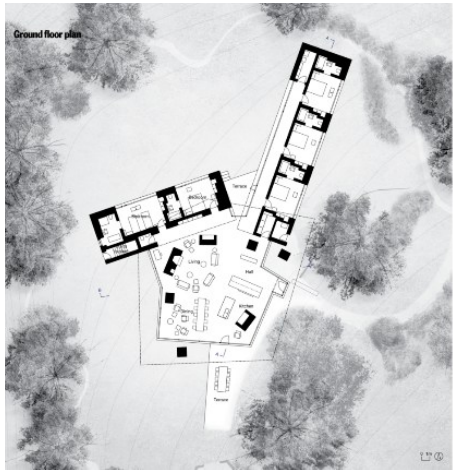
@ Design Data and Society group (Architecture)

@ Computer Vision lab (Computer Science)

Visiting Researcher | Gradient Spaces lab @Stanford

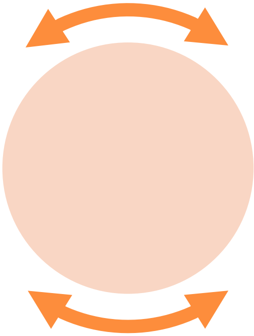


Floor plan similarity: what and how?

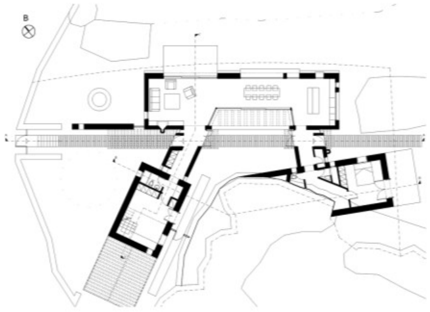


Secular Retreat, England
(Zumthor, 2018).

How spatially similar ?



How to compute ?



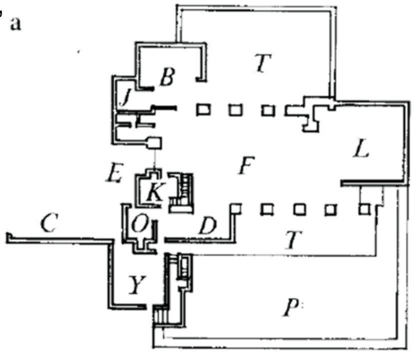
Volax House, Greece
(Aristides Dallas Architects, 2016).

Motivation *cont.*

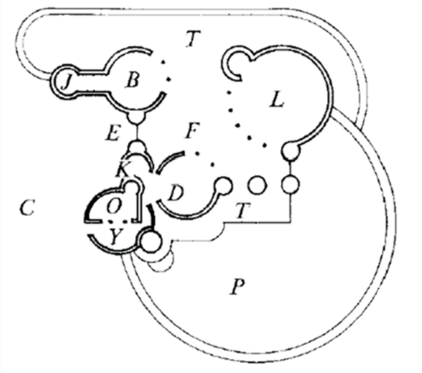
What makes these floor plans similar?

B: bedroom,
(B': only in *Sundt House*),
C: car port,
D: dining room,
E: entrance,
F: family room,
J: bathroom,
K: kitchen,
L: living room,
O: office,
P: pool,
T: terrace,
Y: yard

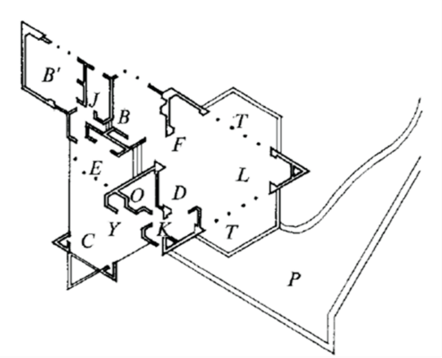
Life



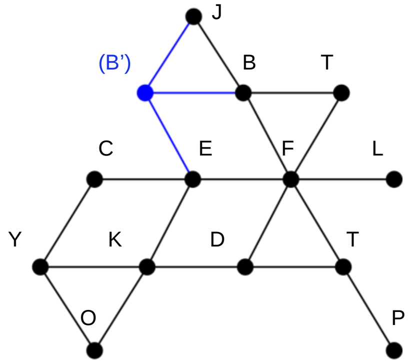
Ralph Jester House



Vigo Sundt House



Same access graph



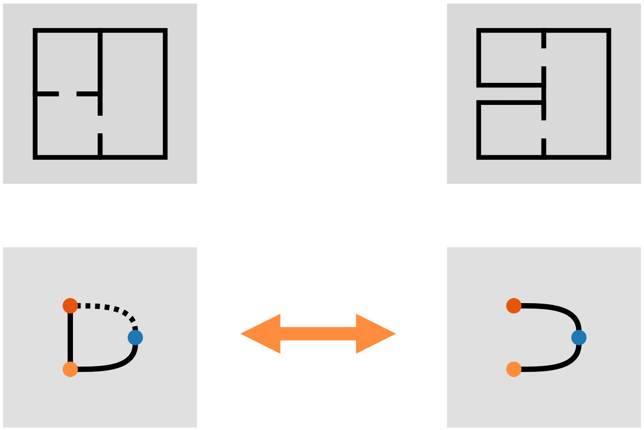
Nodes = rooms
Edges = doors / access

L. March and P. Steadman, **The Geometry of Environment: An Introduction to Spatial Organization in Design**, M.I.T. Press, 1974. [Text and images; except for graph]

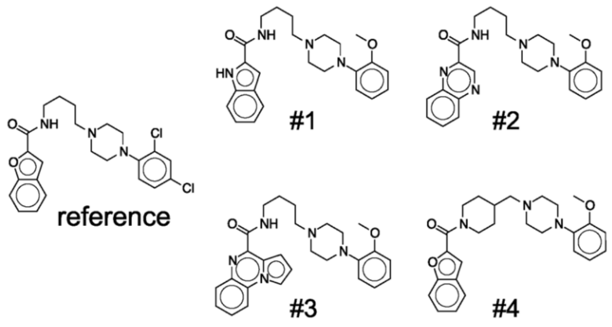
Graph similarity

Floor plans as graphs

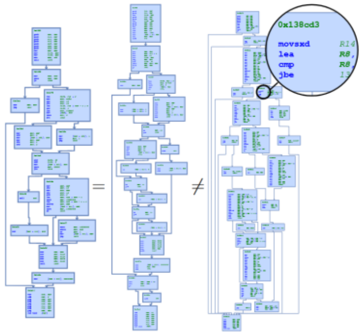
Similarity over graphs



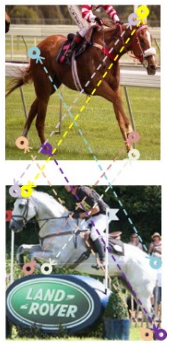
Graph similarity computation (GSC) is a popular and relevant topic in machine learning (ML)



GESim for computing the similarity between molecular structures Cut-out Fig. 2 in [1]

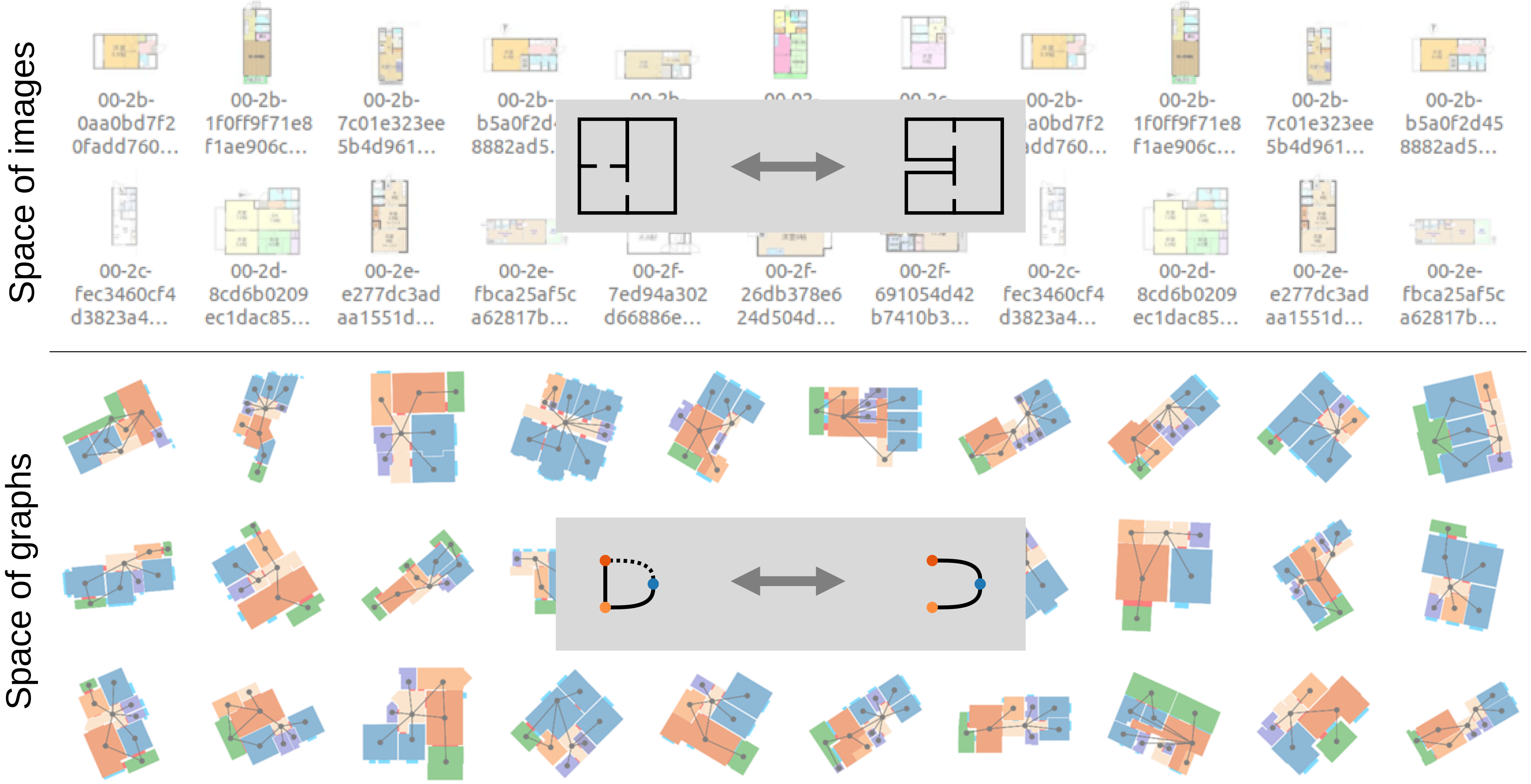


Graph Matching Networks for the identification of vulnerable functions Cutout Fig. 1 in [2]



PCA-GM for matching images Cutout Fig. 1 in [3]

‘Solve’ floor plan similarity in the space of graphs



A function to compare graphs

We seek a function that computes the similarity between two graphs:

$s(\bullet, \bullet)$ $G_1, G_2 \in \mathcal{G}$

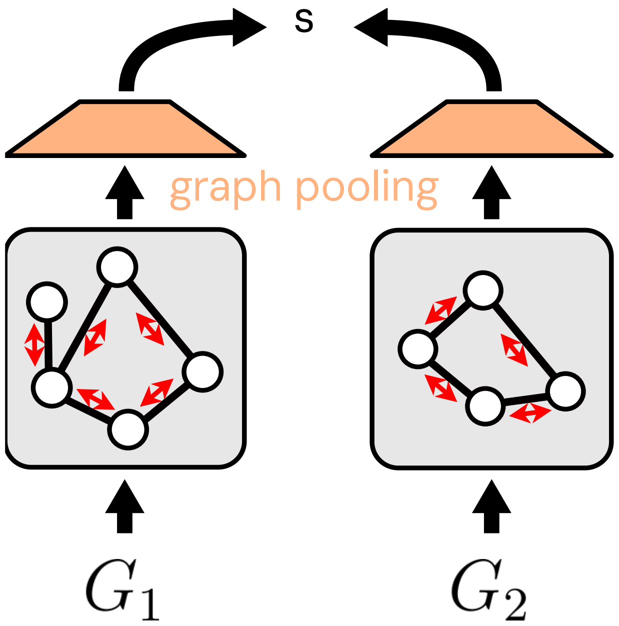
$$s : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}^+,$$

s.t.

1. [**effectiveness**] If ‘true’ similarity is high than *s* is high, and vice versa;
2. [**efficiency**] It can be computed relatively fast

Metric learning to compare graphs

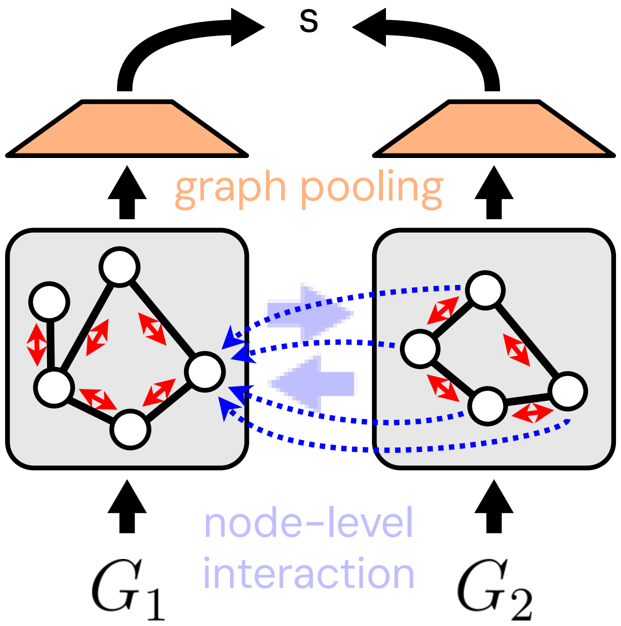
Graph Siamese network



Independent embedding ✅
node-level interactions ❌

$$s(G_1, G_2) = s_{\mathbf{e}}\left(\underbrace{f_\theta(G_1)}_{\mathbf{e}_1}, \underbrace{f_\theta(G_2)}_{\mathbf{e}_2}\right)$$

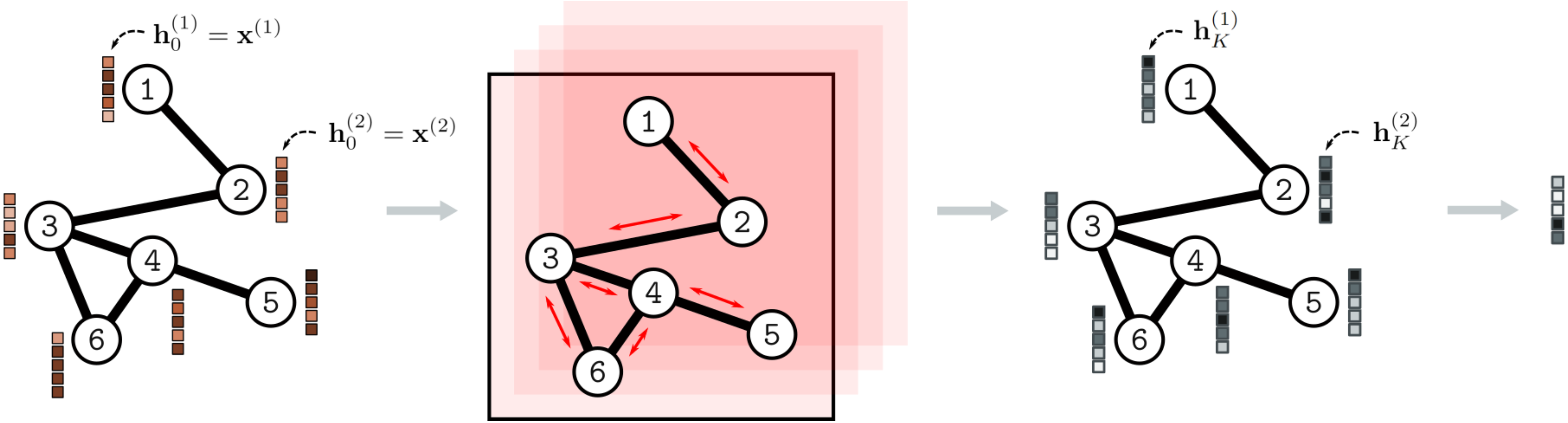
Graph matching network



Independent embedding ❌
node-level interactions ✅

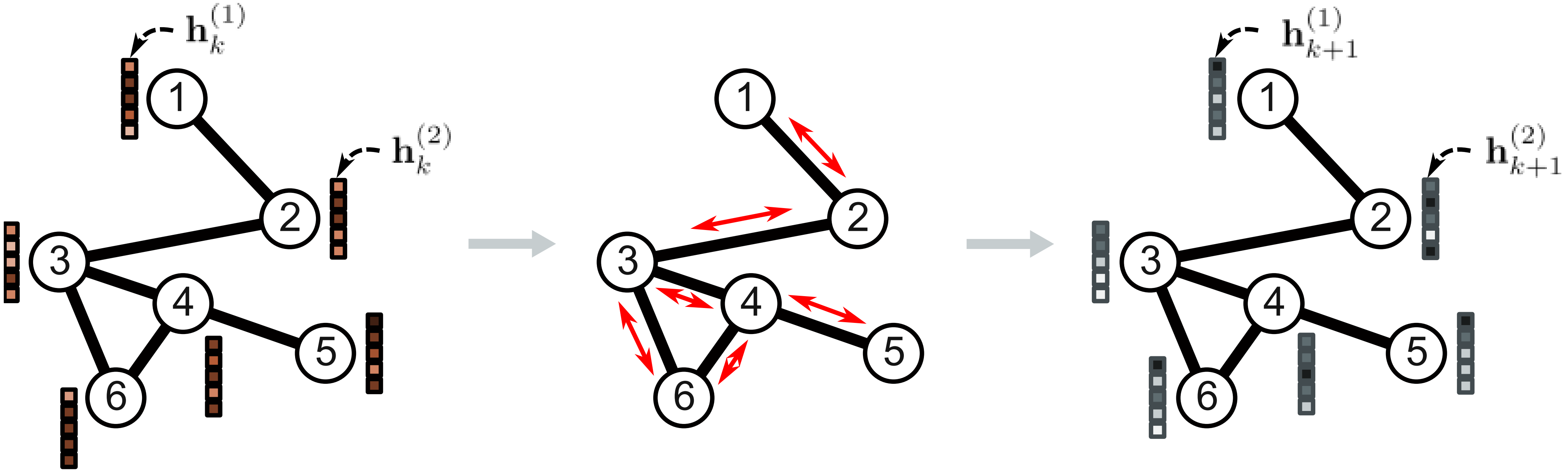
$$s(G_1, G_2) = s_{\mathbf{e}}\left(\underbrace{f_\theta(G_1|G_2)}_{\mathbf{e}_1}, \underbrace{f_\theta(G_2|G_1)}_{\mathbf{e}_2}\right)$$

Graph neural network



$$f_\theta = enc_{\theta_0} \circ \mathrm{gconv}_{\theta_1} \circ \cdots \circ \mathrm{gconv}_{\theta_K} \circ \mathrm{pool}_{\theta_{K+1}}$$

Graph neural network layer



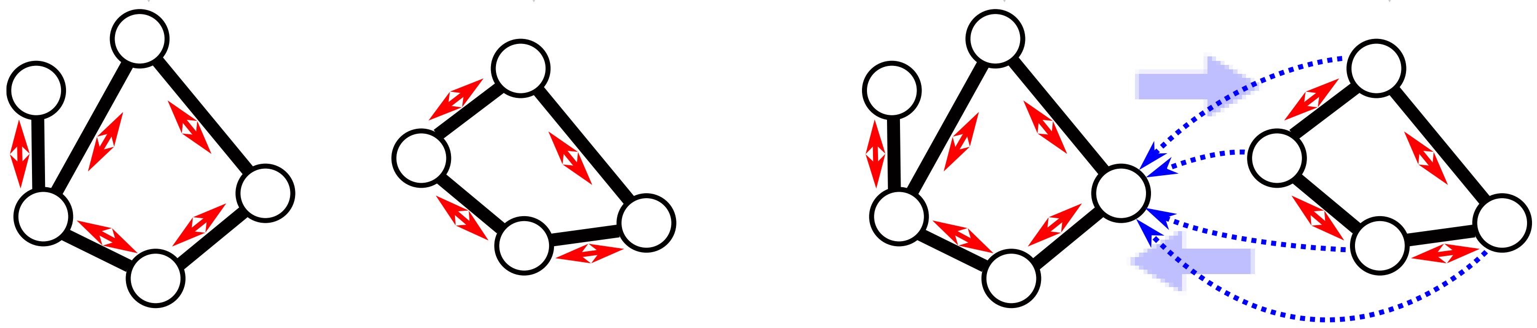
$$\mathbf{m}_k^{(\to n)} = \sum_{m \in \mathrm{ne}(n)} \sigma \circ \left(\mathbf{b}_k + \mathbf{W}_k \mathbf{h}_k^{(m)} \right) \qquad \mathbf{h}_{k+1}^{(n)} = \sigma \circ \left(\boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \begin{bmatrix} \mathbf{h}_k^{(n)} \\ \mathbf{m}_k^{(\to n)} \end{bmatrix} \right)$$

sum over all neighbours

A simple single-layer MLP

Graph matching networks

Graph matching networks (GMN) explicitly model cross-graph node-level interactions



message passing (intra)

$$\mathbf{m}_k^{(\to n)} = \sum_{m \in \mathrm{ne}(n)} \sigma \circ \left(\mathbf{b}_k + \mathbf{W}_k \mathbf{h}_k^{(m)}\right)$$

$$\boldsymbol{\mu}_k^{(\to n)} = \sum_j a_{j \to n} \left(\mathbf{h}_k^{(n)} - \mathbf{g}_k^{(j)}\right)$$

$$a_{j \to n} = \frac{\exp\left(\mathbf{h}_k^{(n)} \cdot \mathbf{g}_k^{(j)}\right)}{\sum_{j' \in \mathcal{E}_2} \exp\left(\mathbf{h}_k^{(n)} \cdot \mathbf{g}_k^{(j')}\right)}$$

cross-graph attention (inter)

node update

$$\mathbf{h}_{k+1}^{(n)} = \sigma \circ \left(\boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \begin{bmatrix} \mathbf{h}_k^{(n)} \\ \mathbf{m}_k^{(\to n)} \end{bmatrix}\right)$$

$$\mathbf{h}_{k+1}^{(n)} = \sigma \circ \left(\boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \begin{bmatrix} \mathbf{h}_k^{(n)} \\ \mathbf{m}_k^{(\to n)} \\ \boldsymbol{\mu}_k^{(\to n)} \end{bmatrix}\right)$$

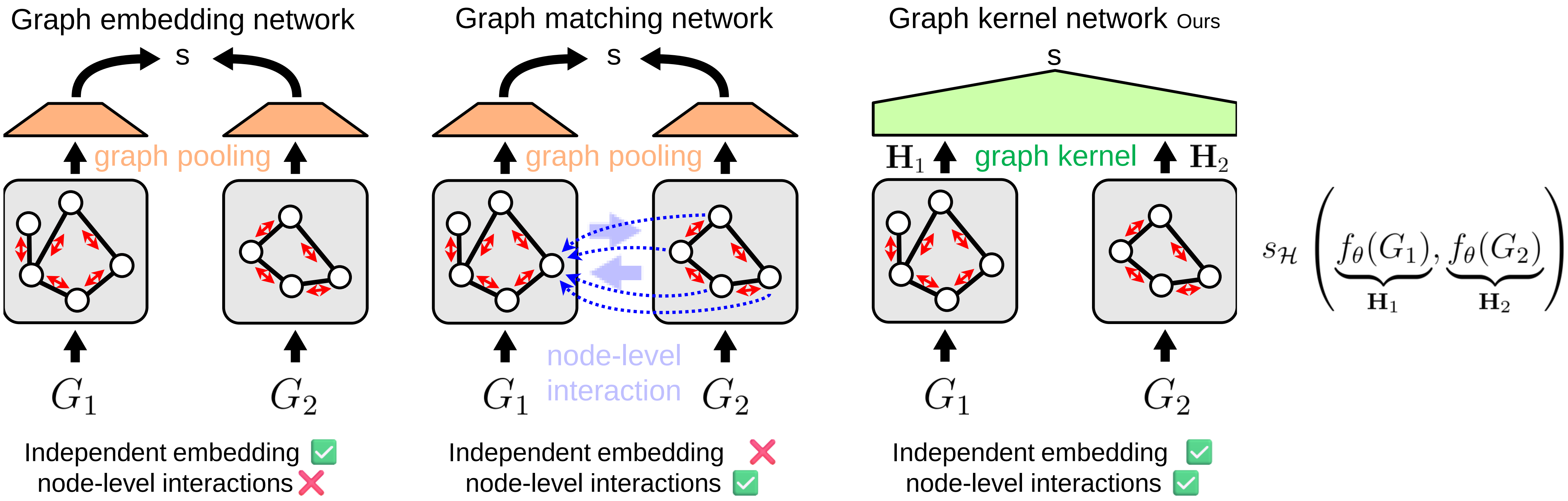
Graph matching networks are slow

Problem: GMNs are <u>slow</u>:

Main: cannot compute node-level embeddings independently

Also: complexity of feedforward itself scales quadratically with the number of nodes

Can we model node-level interactions more efficiently? Yes! → postpone them to similarity metric



C. van Engelenburg, J. van Gemert, S. Khademi, LayoutGKN: Graph Similarity Learning of Floor Plans. In: BMVC, 2025.

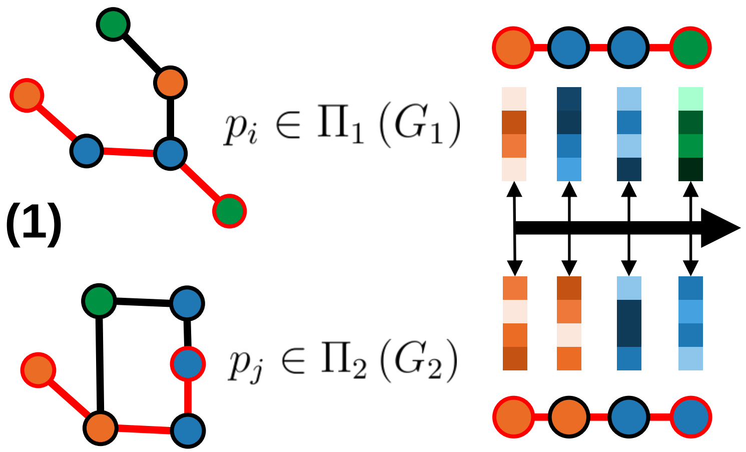
Shortest-path graph kernel as similarity function

How to compare the two sets of learned node embeddings in a topology-aware manner?

→ Use a differentiable graph kernel e.g., shortest path graph kernel:

Similarity is based on the two sets (for each graph) of shortest paths:

- **(1)** Gather shortest paths for each graph
- **(2)** Compute similarity between all (inter-graph) pairs of shortest paths
- **(3)** Sum over all pairwise shortest path similarities
- **(4)** Normalize by self-similarities



$p_i \in \Pi_1(G_1)$

$p_j \in \Pi_2(G_2)$

(2)

$$k_{\text{path}}(p_i, p_j) = \sum_{\substack{u \in p_i \\ v \in p_j}} k_{\text{node}}(\mathbf{h}_u, \mathbf{h}_v)$$

$$k_{\text{node}}(\mathbf{h}_u, \mathbf{h}_v) = \exp\left(-\mu \|\mathbf{h}_u - \mathbf{h}_v\|^2\right)$$

(3)

$$k_G(G_1, G_2) = \sum_{p_i \in \Pi_1} \sum_{p_j \in \Pi_2} k_{\text{path}}(p_i, p_j)$$

(4)

$$s_{\text{shp}}(G_1, G_2) = \frac{k_G(G_1, G_2)}{\sqrt{k_G(G_1, G_1) \cdot k_G(G_2, G_2)}}$$

C. van Engelenburg, J. van Gemert, S. Khademi, LayoutGKN: Graph Similarity Learning of Floor Plans. In: BMVC, 2025.

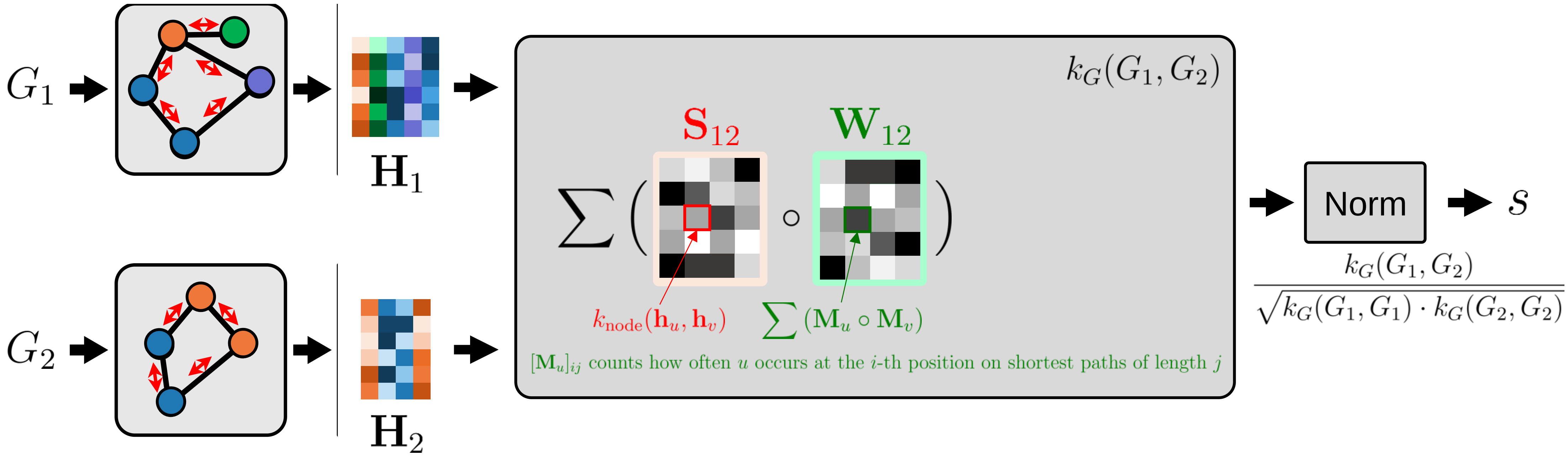
Shortest-path graph kernel in efficient matrix form

The shortest path graph kernel as defined in the previous slide can be written in a simple form:

As a sum over the entries of the pointwise matrix multiplication between

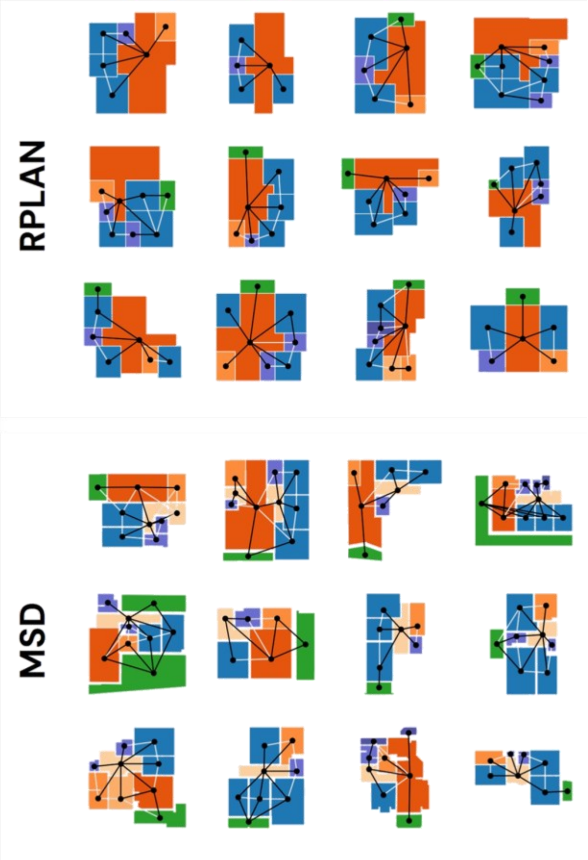
1 similarity matrix **S** between two sets of embeddings **H1** and **H2**

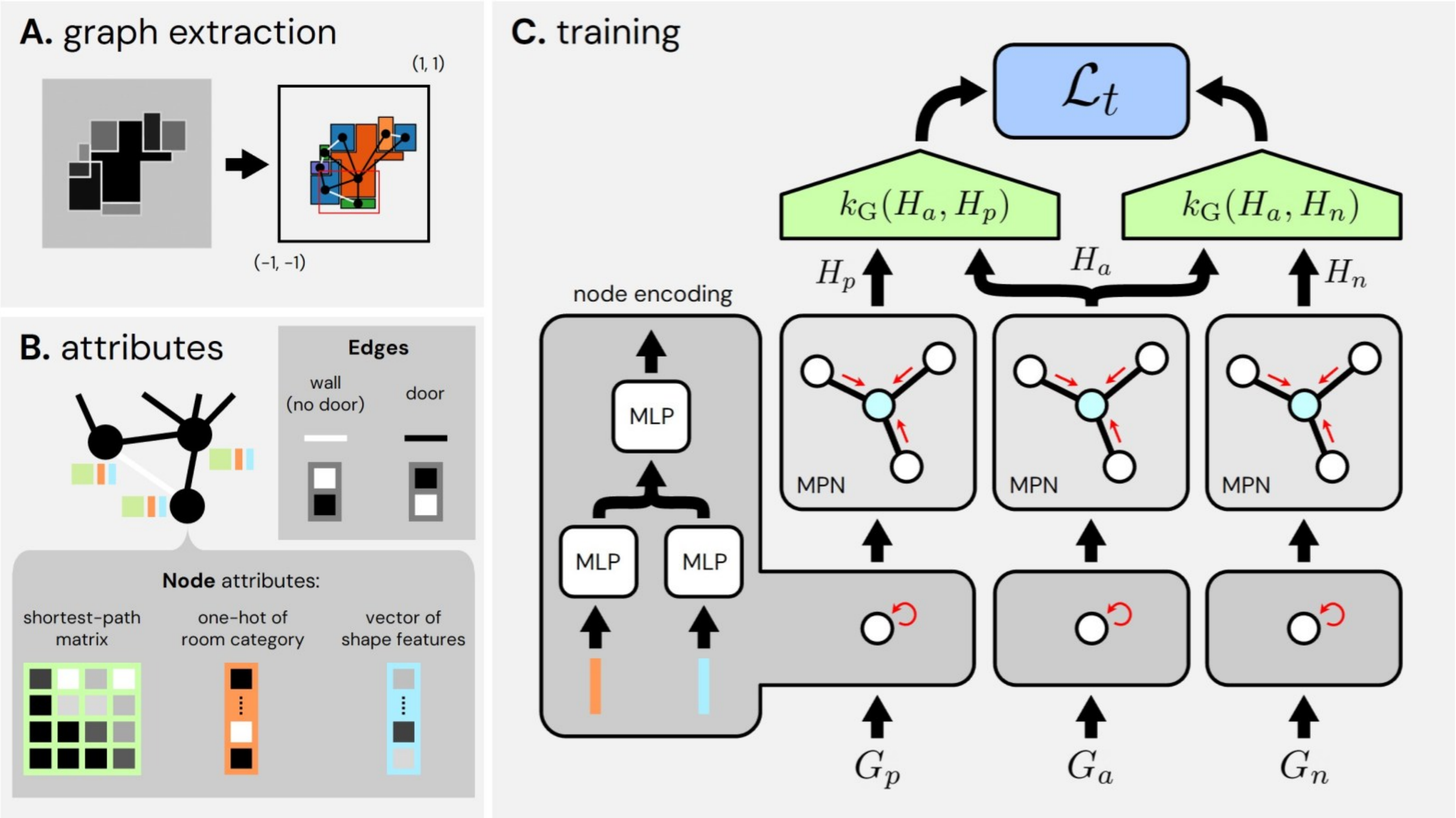
2 weight matrix **W** based on the structural resemblance between nodes in terms of shortest path similarity



C. van Engelenburg, J. van Gemert, S. Khademi, LayoutGKN: Graph Similarity Learning of Floor Plans. In: BMVC, 2025.

LayoutGKN learning to compare floor plan graphs





C. van Engelenburg, J. van Gemert, S. Khademi, LayoutGKN: Graph Similarity Learning of Floor Plans. In: BMVC, 2025.

LayoutGKN main results

Evaluation

- Metrics: triplet accuracy, Precision@k
- RPLAN: test set generalization
- MSD: zero-shot

Results

- Effective and efficient
- Generalizes better (zero-shot)

Table 1: **Performance comparisons on RPLAN and MSD.** We report: the triplet accuracy; precision (P) scores at 5 and 10; and inference time t per 10K pairs. Best in **bold**.

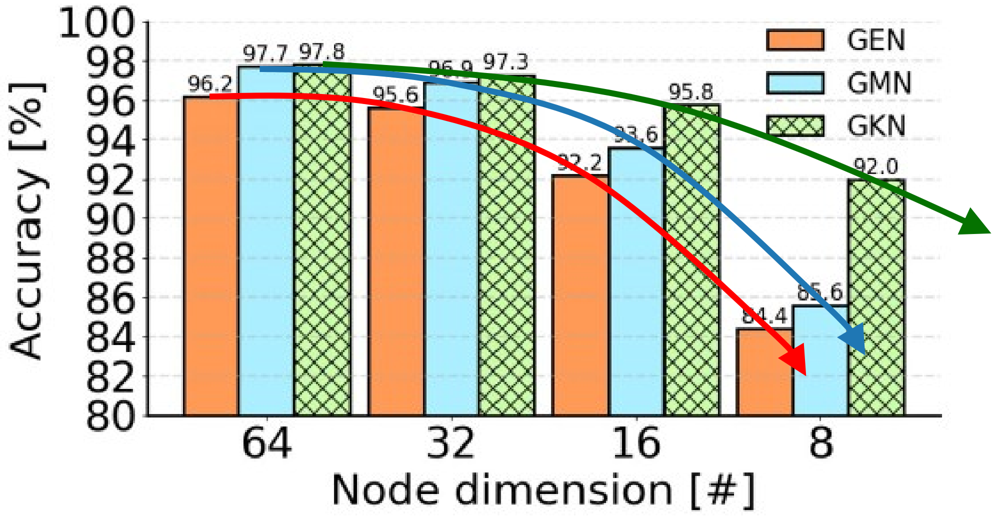
	RPLAN				Generalization to MSD	
	Accuracy (↑)	P@5 (↑)	P@10 (↑)	t (↓)	P@5 (↑)	P@10 (↑)
GK (*base*)	65.63±0.00	0.389±0.000	0.439±0.000	1.2±0.4	na	na
GEN (*base*)	96.24±0.07	0.603±0.007	0.665±0.004	**0.7±0.1**	0.595±0.015	0.605±0.018
GMN [20]	**97.74±0.05**	0.616±0.004	0.675±0.002	35.6±10.5	0.585±0.026	0.596±0.020
GKN (*ours*)	**97.78±0.10**	**0.623±0.004**	**0.683±0.002**	1.8±0.5	**0.674±0.024**	**0.697±0.017**

C. van Engelenburg, J. van Gemert, S. Khademi, LayoutGKN: Graph Similarity Learning of Floor Plans. In: BMVC, 2025.

LayoutGKN performance vs model capacity

Ablation on dimensionality of learned node embeddings

→ Less of a performance drop when the dimensionality decreases (i.e., more scalable!)



That's about it

Want to more about it?

- Paper (arXiv): https://arxiv.org/abs/2509.03737
- GitHub: https://github.com/caspervanengelenburg/LayoutGKN
- Poster: https://bmva-archive.org.uk/bmvc/2025/assets/papers/Paper_184/poster.pdf

Thanks! Qs?

We would be very happy to collaborate on other domains of graph data ☺