

# Cross-Camera Data Association via GNN for Supervised Graph Clustering

Dorđe Nedeljković

Independent Researcher

**Abstract.** Cross-camera data association is one of the cornerstones of the multi-camera computer vision field. Although often integrated into detection and tracking tasks through architecture design and loss definition, it is also recognized as an independent challenge. The ultimate goal is to connect appearances of one item from all cameras, wherever it is visible. Therefore, one possible perspective on this task involves supervised clustering of the affinity graph, where nodes are instances captured by all cameras. They are represented by appropriate visual features and positional attributes. We leverage the advantages of GNN (Graph Neural Network) architecture to examine nodes' relations and generate representative edge embeddings. These embeddings are then classified to determine the existence or non-existence of connections in node pairs. Therefore, the core of this approach is graph connectivity prediction. Experimental validation was conducted on multicamera pedestrian datasets across diverse environments such as the laboratory, basketball court, and terrace. Our proposed method, named SGC-CCA, outperformed the state-of-the-art method named GNN-CCA across all clustering metrics, offering an end-to-end clustering solution without the need for graph post-processing. The code is available at <https://github.com/djordjened92/cca-gnnclust>.

## 1 Introduction

The latest works on multi-camera tasks (multi-target tracking, 3D object detection, pedestrian pose estimation, etc.) [6, 7, 13, 14, 23] usually employ the end-to-end methods, leveraging the very popular transformers architecture. They often provide detections from all camera views to the transformer decoder and rely on the neural network capability to learn instance associations implicitly. However, some methods contain cross-camera instance association as an explicit step in the solution pipeline. This component usually makes a difference, so we find it worth studying as an independent topic.

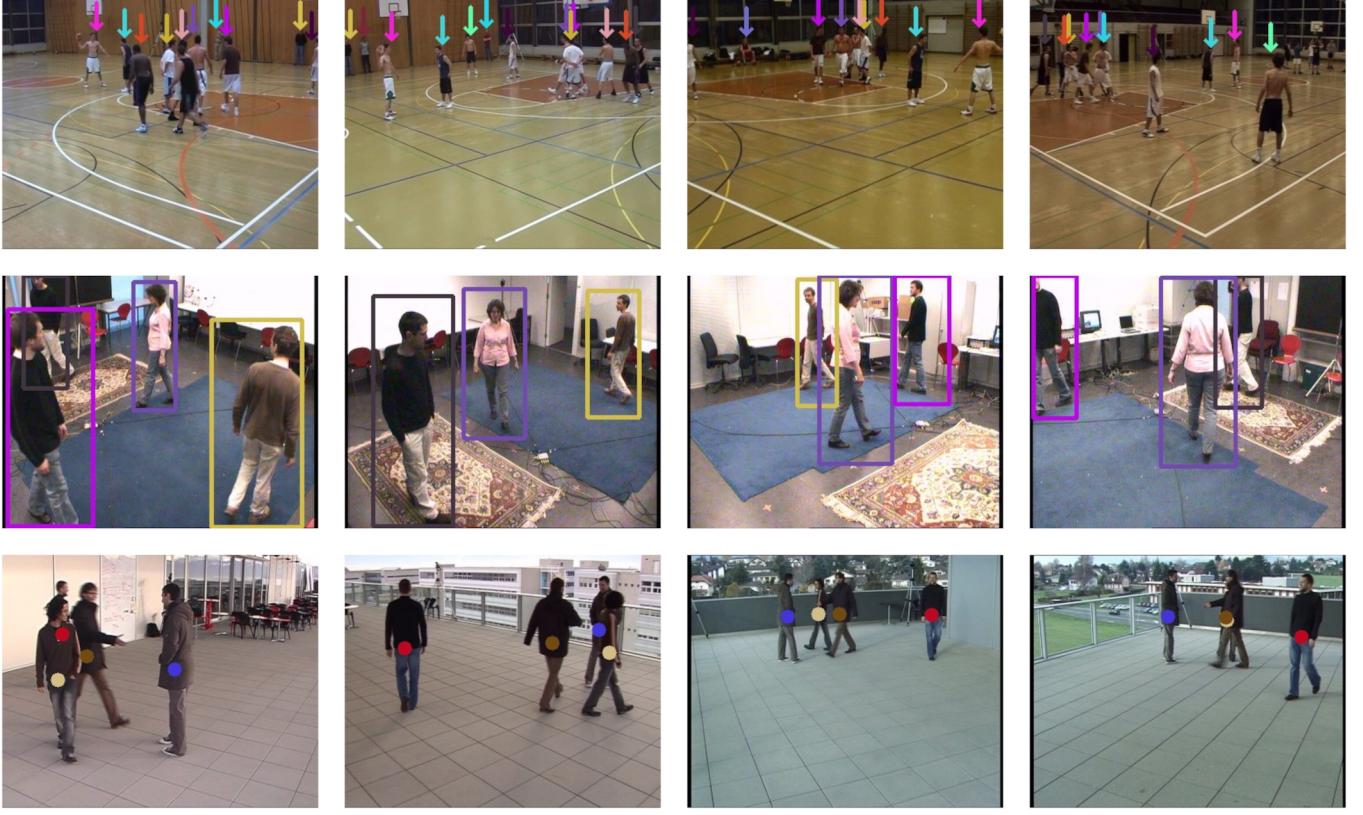
The most common perspective of this problem in literature is decoupling the task into bipartite matching of instances for a particular pair of cameras. The next step is usually the aggregation of camera-pairs results in a global solution. Hungarian algorithm, which is used often in bipartite graph matching exploits association matrix consisting of instance distances. This distance usually reflects the difference in ReIdentification score (obtained from some appearance embedding model) and 3D/2D spatial distance of instances. This approach suffers from high computational demands as the number of instances grows. It also relies on manual thresholding when it comes to association decisions. Moreover, when pairs of views are analyzed independently it can lead to inconsistency when aggregation of the results is performed.

One of the recent works is an attempt to find a global solution for instance association, incorporating all camera views simultaneously. Named GNN-CCA [16], this method exploits GNN Message Passing architecture on top of the complete graph connecting all instances across views. The use case, which we also used in this work, involves associating pedestrians captured by four cameras in three different environments. Images are sampled from video sequences, with pedestrian annotations provided as ground-truth bounding boxes. Some of the best person ReIdentification models provided appearance embeddings for each image crop. Based on the available camera calibration parameters and annotations, the authors managed to provide the ground plane coordinates of each person for each camera view as well. Both appearance and position are used to find a good representation of each node and edge. Following several iterations of message passing, the GNN-CCA [16] method employs binary classification on each edge. This classifier should determine if two connected nodes belong to the same person or not. The final and key step of this method is graph post-processing, consisting of two operations: *edge pruning* and *graph splitting*. These heuristic techniques aim to yield the final set of node clusters (connected components) that represent all visible person identities.

**Contributions** This work provides a novel perspective of an instance matching problem as an end-to-end supervised graph clustering task, unlike the GNN-CCA [16] approach, in which this problem is reduced to the linkage prediction of nodes. Consequently, the following contribution is the avoidance of any tailored graph post-processing heuristics necessary for GNN-CCA [16] approach. Standard graph clustering inference assumes that one large graph is provided as an input that requires partitioning. The IMDB-test, one of the test sets used in the Hi-LANDER [25] paper, consists of 50289 entities/clusters with an average cluster size of 25 (number of cameras in our case). We have shown that the same GNN model can be effectively trained and tested on many smaller, independent input graphs (one graph per scene), with careful adaptations, as presented in Section 2. Furthermore, the proposed SGC-CCA method surpasses GNN-CCA [16] in all clustering metrics.

### 1.1 Why Hi-LANDER?

The motivation for learnable clustering methods via GNN lies in the aim to overcome some limitations of the traditional techniques. K-Means [15] requires the number of clusters as an input and assumes that clusters are convex-shaped. Spectral clustering [20] demands clusters balanced in the number of instances. The shape-agnostic technique is DBSCAN [2], which introduced a density-based ap-



**Figure 1.** Sample scenes from EPFL sequences: Basketball, Laboratory and Terrace.

proach. It brought also a minimal requirement of domain knowledge and good efficiency on the large database at the time. However, it assumes that different clusters have similar densities. The family of linkage-based clustering methods overcomes all mentioned dependencies on the mentioned assumptions. The well-established method of this kind is Rank-Order Clustering [29] and its more efficient successor Approximate Rank-Order Clustering (ARO) [17]. Although these techniques proposed new cluster-level affinity (distance) with demonstrated ability to detect noise instances, instead of traditional  $l_1/l_2$  distance, they are still based on the linkage likelihood estimation by different heuristics.

To mitigate the issues of using arbitrary thresholds, Wang et al. [24] introduce a novel approach employing a learnable clustering framework leveraging the graph convolution network (GCN)[10]. At the heart of their method lies the creation of what they term the Instance Pivot Subgraph (IPS) around each instance acting as a pivot, determined by nearest neighbour criteria. Within each IPS, connectivity between the pivot and its neighbors is assessed using GCN as a learnable component. This assessment is based on node classification, distinguishing between positive (pivot-like) and negative nodes. Only positive nodes within each IPS are connected to the pivot. Following connectivity prediction across all IPSes, the authors employ link merging to derive the final set of clusters.

Another example of the supervised clustering [26] also leverages GCN [10] to tackle the challenge posed by variations of the cluster patterns. The method starts from the kNN affinity graph which encompasses the whole dataset. The first stage is generating Cluster

Proposals which is the heuristic-based approach of subgraphs selection. Those size-varying subgraphs represent multi-scale candidates for the resulting cluster partitioning. The second stage is Cluster Detection where the GCN-D model learns to detect which proposals are kept as good enough and which ones are dropped. The final stage is cluster refinement of the top proposals identified by GCN-D, utilizing another (GCN-S) model. This step is called Cluster Segmentation. Unlike graph-level prediction in the Detection stage, this model outputs a probability for each node to show if it is a valid cluster member or an outlier. The described method relies on the numerous generated subgraphs. The same research group aimed to overcome this redundant computation by employing a fully learnable clustering paradigm in [27]. This method is based on the vertex confidence estimator (GCN-V) - indicates if the vertex belongs to the specific class and edge connectivity estimator (GCN-E) - outputs the probability that specific edge connects two incident vertices. After the affinity graph processing by these two learnable components, it is possible to create a directed path over detected edges from vertices with lower confidence to those with higher confidence. This process deduces several isolated trees, partitioning the affinity graph into clusters.

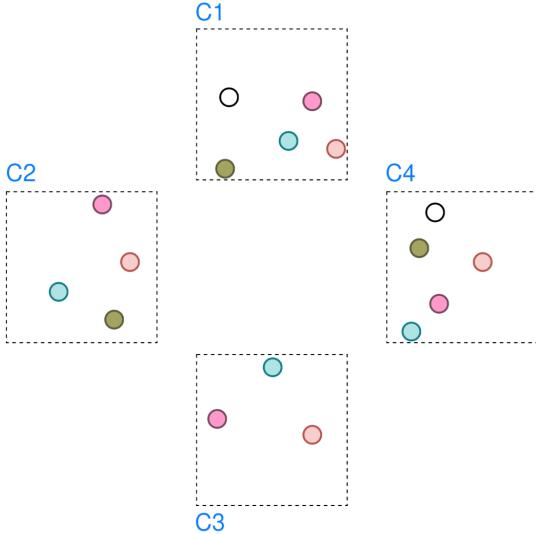
The Hi-LANDER [25] belongs to the same group of supervised GNN-based methods, focusing on linkage prediction as the foundation of clustering. In contrast to earlier single-partitioning techniques [24, 26, 27], Hi-LANDER [25] stands out as the first hierarchical, agglomerative clustering method employing GNN architecture. This multi-level algorithm allows for modeling the "natural granularity" of the data. Unlike previous approaches, Hi-LANDER [25] employs

*full-graph* inference and predicts connectivity based on *edge* features. Unlike [27], Hi-LANDER [25] utilizes a single model for both linkage probability estimation and node density prediction. Node density serves regularization purposes and facilitates additional edge refinement. This method demonstrates superior efficiency alongside significant improvements in metrics. Further details about this approach will be discussed in the following section.

## 2 Methodology

This work applies supervised graph clustering to perform cross-camera instance association. We adapt Hi-LANDER [25] hierarchical GNN architecture to predict graph connectivity and to extract connected components of graph which successfully represent person identities. Although this technique is applicable to the arbitrary number of views and types of objects, we focused on the EPFL<sup>1</sup> pedestrian video sequences with four views. Thus, our results are comparable with GNN-CCA [16] results. Sample scenes of all three setups are shown in Figure 1. The same person across all views is marked with the same color.

The following sections present the implemented clustering and training pipeline from the Hi-LANDER [25]. Changes that are part of the SGC-CCA customization will be **emphasized**.



**Figure 2.** Resulting clusters from Table 1.

### 2.1 Graph Creation

Let  $M$  be the number of cameras in the setup.  $N$  is the total number of pedestrian bounding boxes across all views at the timestamp  $t$ . The set of cameras is denoted as  $C = \{c_i \mid i \in [1, M]\}$  and  $[r_1, r_2, \dots, r_N]$  are bounding box labels from the set  $R = \{r_i \mid i \in [1, O]\}$  of  $O$  pedestrian identities.  $F = \{f_i \mid i \in [1, N]\}$  is the set of visual features extracted on pedestrian crops using a pretrained ReID model.

The input structure for this method is a directed graph  $G = (V, E)$ , where  $V = \{v_i \mid i \in [1, N]\}$  represents the set of nodes denoting all pedestrian bounding boxes. Each node is depicted by embedding  $h_i$  initialized with appropriate, normalized feature  $f_i$ , forming node embeddings set  $H = \{h_i \mid i \in [1, N]\}$ .

<sup>1</sup> the dataset URL

The pedestrian  $v_i$  on the camera  $c_j$  has a ground-truth bounding box in the form  $(x_i, y_i, w_i, h_i)$  - upper-left corner coordinates, width, and height. Estimation of the pedestrian's standing point is the middle point of bounding box lower edge  $(x_i + \frac{w_i}{2}, y_i)$ . Thanks to the available camera's intrinsic and extrinsic parameters, it is possible to create a homography matrix  $H_{c_j}$ . It enables projection from the camera  $c_j$  plane into the common ground plane. Thus, the standing point position in the common ground plane  $(\hat{x}_i, \hat{y}_i)$  is calculated with the:

$$\begin{bmatrix} \hat{x}_i \\ \hat{y}_i \\ 1 \end{bmatrix} = H_{c_j} \begin{bmatrix} x_i + \frac{w_i}{2} \\ y_i \\ 1 \end{bmatrix} \quad (1)$$

**The node distance combines the cosine distance of node embeddings and normalized Euclidean distance of positions**  $(\hat{x}, \hat{y})$ , comparing to the original work where the cosine distance is used only. Therefore the distance between nodes  $v_i$  and  $v_j$  is defined as:

$$m_{ij} = (1 - \langle h_i, h_j \rangle) \frac{\|(\hat{x}_i - \hat{x}_j), (\hat{y}_i - \hat{y}_j)\|_2}{\max_{p,q} \|(\hat{x}_p - \hat{x}_q), (\hat{y}_p - \hat{y}_q)\|_2} \quad (2)$$

where  $\langle \cdot, \cdot \rangle$  denotes the inner product of two vectors, which is equal to cosine similarity for normalized embedding vectors.

For each node of camera  $c_i$ , we find the **one closest neighbor from each other camera view**  $c_j, j \neq i$  (cell  $(l_1, a)$  in Table 1), unlike Hi-LANDER [25] which applies pure kNN over the whole corpus of nodes. This neighbor selection per camera is related to the setup where the pedestrian can appear mostly once in each view. Consequently, each node adds  $M - 1$  edges to the set of graph edges  $E$ . If the node  $v_j$  is the neighbor of the node  $v_i$ , adjacency matrix score  $A(i, j)$  is assigned a cosine similarity of node embeddings  $\langle h_i, h_j \rangle$ .

### 2.2 Graph Clustering

#### 2.2.1 Graph Encoding

The input affinity graph is processed by graph convolution network (GCN)[10]. This message passing paradigm [3] simulates nodes' interaction and information exchange. Using  $h_i$  as the input embedding of the node  $v_i$ , GCN encodes it as a new node embedding  $h'_i$  in the following way:

$$h'_i = \phi(h_i, \sum_{v_j \in N_{v_i}} w_{ji} \psi(h_j)) \quad (3)$$

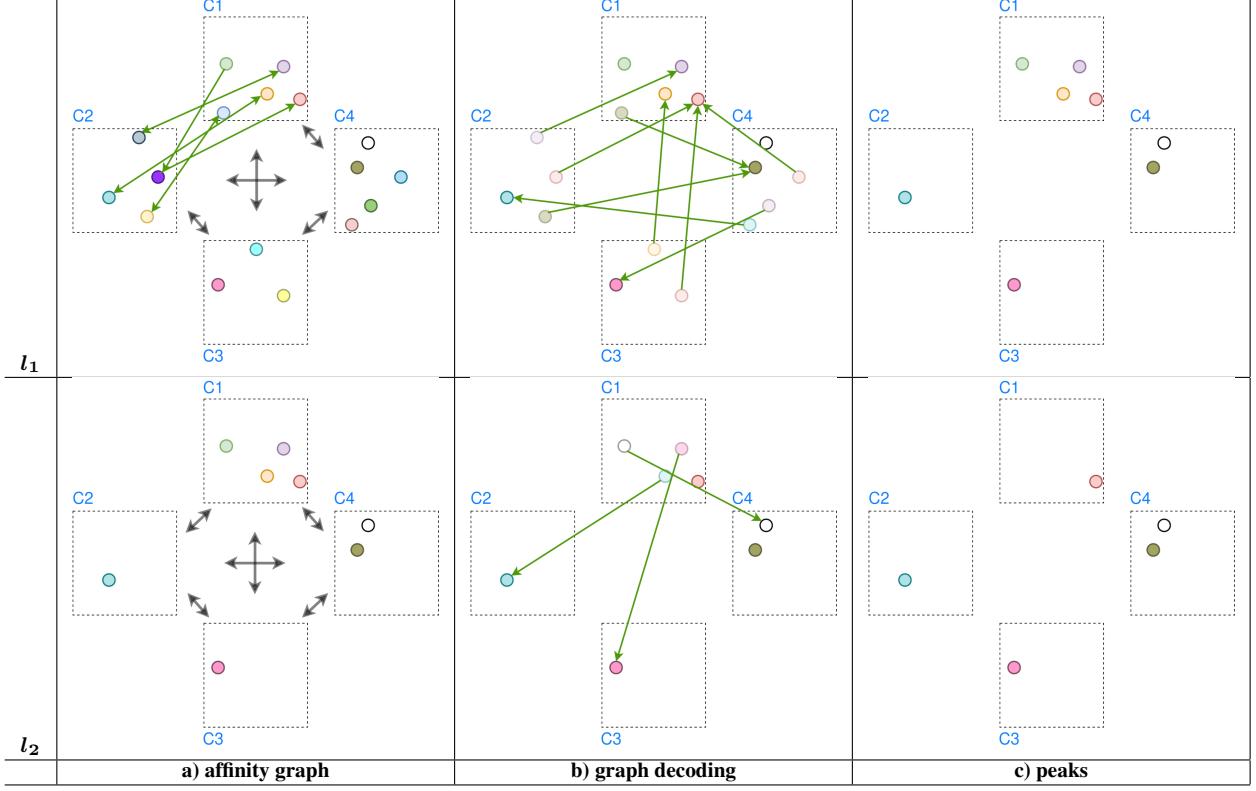
where  $\phi$  and  $\psi$  are MLPs,  $w_{ji}$  is a trainable vector.  $N_{v_i} = \{v_j, (v_j, v_i) \in E\}$  is the neighborhood of node  $v_i$ , defined with the set of incoming edges.

GCN encoder can be applied multiple times on the same graph, so the effect of the number of message passing steps is also explored in this work.

#### 2.2.2 Linkage Prediction and Node Density

After the Graph Encoding step, resulting node features  $H'$  are used to predict the linkage between nodes. The edge  $(v_i, v_j)$  connectivity is predicted by applying MLP classifier  $\theta$  from Equation (4). The input is a vector created from **concatenated node features**  $(h'_i, h'_j)$  and **nodes' ground plane positions**  $(\hat{x}_i, \hat{y}_i), (\hat{x}_j, \hat{y}_j)$ . The original work considers the concatenation of node features only. The output

**Table 1.** Clustering pipeline in two levels (as table rows), based on the peak estimation steps (as table columns)



is a sigmoid activation which estimates the probability that two connected nodes have the same label.

$$\hat{r}_{ij} = P(r_i = r_j) = \sigma(\theta([h'_i, \hat{x}_i, \hat{y}_i, h'_j, \hat{x}_j, \hat{y}_j]^T)) \quad (4)$$

A node density  $d_i$  is the value that depicts the weighted partition of neighbors which have the same label as the node  $v_i$ . Its estimation is defined as:

$$\hat{d}_i = \frac{1}{k} \sum_{j=1}^k \hat{e}_{ij} a_{ij}. \quad (5)$$

where  $a_{i,j} = \langle h_i, h_j \rangle$  is the similarity of nodes' embeddings, and  $\hat{e}_{ij}$  is the edge coefficient defined as:

$$\hat{e}_{ij} = P(r_i = r_j) - P(r_i \neq r_j). \quad (6)$$

### 2.2.3 Graph Decoding

After an estimation of the graph attributes (node density and edge coefficient) using the GNN encoder, it is possible to find connected components of the graph in the next two steps:

1. *Edge filtering:* We initialize a new edge set  $E' = \emptyset$ . The subset of outgoing edges for each node  $v_i$  are created as

$$\varepsilon(i) = \{j \mid (v_i, v_j) \in E \wedge \hat{d}_i \leq \hat{d}_j \wedge \hat{r}_{ij} \geq p_\tau\} \quad (7)$$

where  $\hat{r}_{ij} = P(r_i = r_j)$  and  $p_\tau$  is the edge connection threshold. Each node with non-empty  $\varepsilon_i$  contributes to the set  $E'$  with one edge selected as

$$j = \operatorname{argmax}_{k \in \varepsilon(i)} \hat{e}_{ik} \quad (8)$$

The edge  $(v_i, v_j)$  is added to the  $E'$ . With the condition  $\hat{d}_i \leq \hat{d}_j$  authors introduced an inductive bias to discourage connection to nodes on the border of clusters.

2. *Peak nodes:* The set of edges  $E'$  defines new, refined graph  $G'$  (cell  $(l_1, b)$  in Table 1) on the same set of nodes. The peak nodes are those without outgoing edges. They have a maximum density in the neighborhood. The way  $G'$  is created implies a separation of the graph in the set of connected components  $Q = \{q_i \mid i \in [1, Z]\}$ . Consequently, each connected component has one peak node distinguished by the highest density in the connected component (cell  $(l_1, c)$  in Table 1).

### 2.3 Hierarchical Design

The whole pipeline explained in subsection 2.2 can be repeated on the final set of peak nodes as a new input (row  $l_2$  in Table 1). Multi-level approach demands an aggregation of the features for each connected component  $q_i^{(l)}$  from the level  $l$ , which is replaced with a single node  $v_i^{(l+1)}$  on the level  $l + 1$ . The node embeddings  $h_i^{(l+1)}$  of the next level is defined as a concatenation of the peak node features  $\tilde{h}_{q_i}^l$  and the mean node features  $\bar{h}_{q_i}^l$ :

$$h_i^{(l+1)} = [\tilde{h}_{q_i}^l, \bar{h}_{q_i}^l]. \quad (9)$$

In the SGC-CCA approach, we are passing the ground plane coordinates through the levels as well. Their aggregation for the  $q_i^{(l)}$  is implemented as a mean value:

$$(\hat{x}_i^{(l+1)}, \hat{y}_i^{(l+1)}) = (\frac{1}{Z} \sum_{j=1}^Z \hat{x}_j^{(l)}, \frac{1}{Z} \sum_{j=1}^Z \hat{y}_j^{(l)}). \quad (10)$$

After the aggregation step, repeating the 2.2 provides the ability to merge some of the connected components in order to reach the right level of granularity. The total number of levels  $L$  is a hyperparameter. The algorithm stops when the set  $E'$  is empty in the current level, or the level number  $L$  is reached. The last set of peaks defines the final cluster labels. Each peak label is propagated back to the appropriate set of input nodes. Figure 2 depicts clusters of the input instances, assigning the final peak color from the cell  $(l_2, c)$  in Table 1.

## 2.4 Training

### 2.4.1 Ground-truth Graphs

The previously presented multi-level clustering algorithm assumed an inference time scenario. The training procedure relies on the set of ground-truth graphs  $\{G^{(l)} \mid l \in [1, L]\}$  per each scene. They are created by skipping 2.2.1 *Graph Encoding* and 2.2.2 *Linkage Prediction and Node Density* steps with next substitutions in the 2.2.3 *Graph Decoding*:

1. *Node density*: Instead of the estimation in Equation 5, here we used the ground-truth node density defined in Equation 11

$$d_i = \frac{1}{k} \sum_{j=1}^k (\mathbb{1}(r_i = r_j) - \mathbb{1}(r_i \neq r_j)) a_{ij}, \quad (11)$$

where  $\mathbb{1}$  is an indicator function and  $r_i$  is a pedestrian label of the node  $v_i$ .

2. *Edge filtering*: The subset of outgoing edges for the node  $v_i$  is formed as:

$$\varepsilon(i) = \{j \mid (v_i, v_j) \in E \wedge d_i \leq d_j \wedge a_{ij} \geq p_\tau\}. \quad (12)$$

It can be noticed that ground-truth node density is used and the similarity (adjacency score)  $a_{ij}$  is thresholded instead of predicted probability  $\hat{r}_{ij}$  in Equation 7. Afterwards, the most similar node selection is performed again by adjacency score  $a_{ij}$  unlike the Equation 8, wherein the edge coefficient is used:

$$j = \operatorname{argmax}_{k \in \varepsilon(i)} a_{ik} \quad (13)$$

2.3 *Hierarchical Design* is applied in the way it is explained on top of the modified 2.2.3 *Graph Decoding*.

Once the set of ground-truth graphs is acquired, the 2.2.1 *Graph Encoding* is executed specifically for *Linkage Prediction* and GCN training. The predicted probabilities are then utilized for comparison against labeled connectivity, thereby facilitating loss calculation.

### 2.4.2 Loss

Hi-LANDER [25] utilizes both the linkage and the node density prediction in the composite loss implementation. It consists of connectivity and density loss. **In our study, we determined that optimizing the connectivity loss alone was adequate for our task.** This loss is specified as the binary cross-entropy loss per edge  $l_{ij}$ , based on their linkage prediction  $\hat{r}_{ij}$  (Equation 14 and Equation 15).

$$L = -\frac{1}{|E|} \sum_{(v_i, v_j) \in E} l_{ij} \quad (14)$$

$$l_{ij} = \mathbb{1}(r_i = r_j) \log \hat{r}_{ij} + \mathbb{1}(r_i \neq r_j) \log(1 - \hat{r}_{ij}) \quad (15)$$

The indicator function  $\mathbb{1}(r_i = r_j)$  is the ground-truth label which indicates if two nodes belong to the same cluster (the same pedestrian for this specific case).

Graphs from all levels of all scenes are considered equally and independently. They form a general pool of ground-truth graphs, which are the partitioned in fixed-sized mini-batches for the purpose of loss optimization.

**Table 2.** Dataset combinations used for training (T) and validation (V).

	Laboratory	Terrace	Basketball
S1	T	T	V
S2	T	V	T
S3	V	T	T

## 3 Experiments

### 3.1 Dataset

The EPFL video sequences dataset is highly relevant for the described cross-camera association task. It provides 4 overlapping Field of Views (FOVs), each containing a maximum of 9 pedestrians. As authors of the GNN-CCA [16] explained, the EPFL dataset provides multiple views of the same persons along with camera calibration parameters, facilitating global position calculation. Thus, the EPFL is a very suitable for the approaches such as GNN-CCA [16] and our proposed SGC-CCA. In the main experiment section, we kept the sequences selection and combination strategy employed by the authors of GNN-CCA [16]. This involves utilizing three distinct environments with different sets of pedestrians: Terrace - the outdoor setup, Laboratory - the indoor setup, and Basketball - the sports court setup. Scene samples with marked pedestrians across views can be seen in Figure 1.

The training and validation (inference) datasets are constructed by combining three subsets in various configurations. These combinations are outlined in Table 2 and referred to as S1, S2, and S3.

### 3.2 Implementation Details

#### 3.2.1 Appearance Embeddings

In these methodologies, pedestrian cropped images are encoded using pretrained ReID models. Luna et al. [16] conducted a study on available relevant ReID feature extractors, concluding that model used in the [1] is a good compromise between performance and the input size. This ResNet50 model is pretrained on the Market1501 [28], CUHK03 [11], and DukeMTMC [18]. It has an input size of 128x64 pixels and generates embedding output of size 256.

#### 3.2.2 Architecture Choices

**GNN** The utilized GNN architecture is a broadly used vanilla GCN [10] model. We also explored the GAT [21] alternative, but it didn't show a significant difference. The input node feature size is the visual appearance feature extractor's output size of 256. Our baseline experiment is conducted with 2 message passing steps where the resulting node embedding size is 48.

**Edge Encoder** The edge encoder  $\theta$  in Equation 4 is an MLP consisting of 2 Linear layers with PReLU [4] activation and a final sigmoid regressor.

**Levels** The hierarchical framework is designed using  $L = 3$  levels, with the early-stopping paradigm. It relies on checking if any new edge is added in the current level or if none of the peaks could be additionally connected. If so, the pipeline can be stopped on the current level, and the current set of peaks defines detected clusters.

**Batch** As discussed in the 2.4.1 *Ground-truth Graphs*, ground-truth graphs are created for each level of each scene’s affinity graph. They form a single pool of graphs that are batched for training purposes.

### 3.2.3 Training Procedure

The training is performed using the batch size 48 for 200 epochs. The GCN and MLP models are optimized using the Adam [9] method and one-cycle cosine learning rate scheduler [5] with a base learning rate of 0.07. The regularization of the GCN model is performed by the dropout technique of value 0.1. The edge probability threshold  $p_\tau$  is assigned a value of 0.2.

## 3.3 Evaluation Metrics

Given that the task of data association in this study boils down to clustering person detections across various views, we opt to utilize clustering performance metrics as employed in GNN-CCA [16].

**Adjusted Rand Index (ARI)** This metric [8] is the similarity measure between two clusterings. It is directly related to the number of pairs of instances sharing the same or different labels in both the predicted and true clusterings. The term "Adjusted" indicates that it is adjusted for chance, implying that its value approaches 0 for random labeling and 1 when the clustering is optimal.

**Adjusted Mutual Information (AMI)** AMI [22] reflects the mutual information of two assignments - predicted and ground-truth cluster labels, ignoring adopted label values. Even if we permute label values among predicted or ground-truth clusters, AMI remains the same. That is a measure of agreement between two assignments. Like ARI, this metric is also normalized against chance, to avoid high value for randomly assigned cluster labels.

**V-measure (V-m)** This metric [19] is defined as a harmonic mean between Homogeneity (**H**) and Completeness score (**C**). Homogeneity is the measure of the ground-truth labels diversity among data points inside of the predicted cluster. If each cluster contains only items of one identity, then Homogeneity is satisfied. Completeness score reflects the level of assigning all members of one identity to the same cluster. As aforementioned metrics, these scores are also independent of the absolute values of the labels.

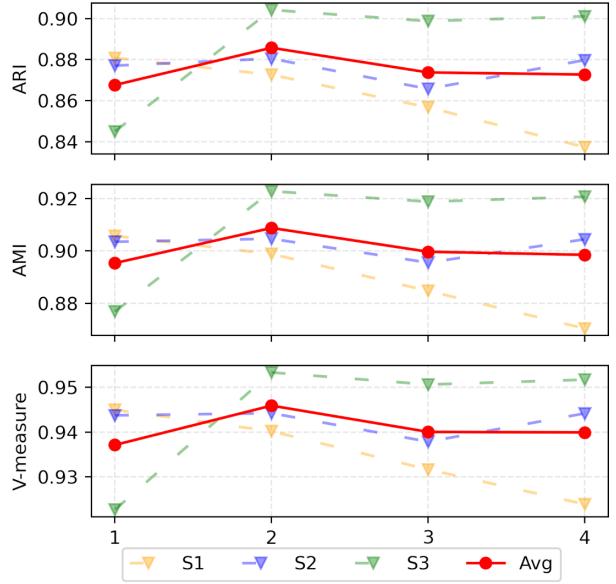
All described metrics have a value of 1 for the perfect labeling (in our reports it is scaled to 100).

## 3.4 Results

Table 3 presents a comparison of the proposed SGC-CCA method with previously explored methods in [16]. We measured ARI, AMI, Homogeneity, Completeness, and V-measure for each dataset configuration S1-3 and aggregated them as the average value. As relevant comparable methods, we consider the GNN-CCA [16] and the combination of geometrical and appearance features approach [12], which appears as the closest opponent to the method proposed in

**Table 3.** Comparison of different approaches, using pretrained ResNet50 model on Market-1051, CUHK03 and DukeMTMC as feature extractor.

	Method	ARI	AMI	H	C	V-m
S1		51.65	56.94	85.55	74.41	79.23
S2	Geom. approach	44.17	50.62	78.79	71.33	73.04
S3	+ Appearance [12]	73.25	75.27	94.40	82.33	87.31
avg.		<u>56.36</u>	<u>60.94</u>	<u>86.25</u>	<u>76.02</u>	<u>79.86</u>
S1		82.99	85.12	94.23	89.97	91.94
S2	GNN-CCA [16]	83.07	86.77	93.59	92.03	92.66
S3	(reported)	88.24	91.07	93.63	95.59	94.42
avg.		<u>84.76</u>	<u>87.65</u>	<u>93.81</u>	<u>92.53</u>	<u>93.00</u>
S1		79.70	82.59	92.48	88.98	90.52
S2	GNN-CCA [16]	82.17	85.61	94.44	91.12	92.55
S3	(reproduced)	83.51	85.16	96.54	87.49	89.40
avg.		<u>81.79</u>	<u>84.45</u>	<u>94.49</u>	<u>89.20</u>	<u>90.82</u>
S1		85.69	88.49	93.60	93.04	93.17
S2	SGC-CCA	88.05	90.46	94.52	94.43	94.42
S3	(ours)	90.42	92.28	95.69	95.14	95.33
avg.		<b>88.05</b>	<b>90.41</b>	<b>94.60</b>	<b>94.20</b>	<b>94.31</b>



**Figure 3.** Metrics for different number of message passing steps.

[16]. Because only the S1 checkpoint of the GNN-CCA [16] is available online, we tried to reproduce the model training for all S1-3 setups. We used the available hyperparameter values in this study. For those that were not provided, we retained the default values from the code repository. The reproduced results are reported in the separate row of the Table 3. However, SGC-CCA shows the best performance on average across all clustering metrics.

We studied the impact of different message passing steps. The saturation of all metrics is achieved for the value of 2, as can be noticed in Figure 3.

### 3.4.1 Generalization

We conducted separate sets of experiments to assess the generalization capabilities of the proposal. Although the main results sec-

**Table 4.** Clustering metrics of SGC-CCA over all training/validation/test dataset setups.

Setup	ARI	AMI	H	C	V-m
BLT	85.33	88.43	93.72	92.66	93.05
BTL	85.84	88.65	93.84	92.19	92.85
LBT	87.11	89.86	94.39	93.82	94.03
LTB	81.78	85.86	92.77	91.00	91.69
TBL	89.86	91.93	95.57	94.80	95.07
TLB	84.86	88.02	93.62	92.44	92.86
avg.	<b>85.80</b>	<b>88.79</b>	<b>93.98</b>	<b>92.82</b>	<b>93.26</b>

tion 3.4 relies on the training and validation setup from the independent domains, we aim to show that high performance is kept even after the model selection during the training, which is guided by the validation dataset. Once the training is finished, we evaluate the model on the third independent set. Given our three distinct environments, the training is performed over all six permutations of *training/validation/test* configurations. Table 4 contains particular results for each setup, along with the final averaged metrics in the last row. The column **Setup** contains encoded names of the dataset combinations, where, for instance, *BTL* denotes Basketball/Laboratory/Terrace sequences as the *training/validation/test* datasets. These results indicate strong generalization capabilities of the proposed method.

## 4 Conclusions

This work shows that the carefully tailored, general-purpose, supervised clustering method Hi-LANDER [25] outperforms the state-of-the-art GNN cross-camera data association method GNN-CCA [16] across all clustering metrics. Furthermore, this approach avoids any graph post-processing such as *affinity graph splitting* or *edge pruning*. Consequently, these heuristics are left to be inferred from the training set, facilitated by the sophisticated hierarchical architecture of [25].

Moreover, the generalization experiments indicate that our proposal maintains the same level of quality even when applied to entirely different environments compared to the training and validation dataset.

## References

- [1] G. Brasó and L. Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6247–6257, 2020.
- [2] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.
- [3] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry, 2017.
- [4] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.
- [5] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li. Bag of tricks for image classification with convolutional neural networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 558–567, 2018.
- [6] J. Huang and G. Huang. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection, 2022.
- [7] J. Huang, G. Huang, Z. Zhu, Y. Ye, and D. Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view, 2022.
- [8] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2:193–218, 1985.
- [9] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [10] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks, 2017.
- [11] W. Li, R. Zhao, T. Xiao, and X. Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 152–159, 2014.
- [12] J. P. Lima, R. Roberto, L. Figueiredo, F. Simoes, and V. Teichrieb. Generalizable multi-camera 3d pedestrian detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1232–1240, 2021.
- [13] Y. Liu, T. Wang, X. Zhang, and J. Sun. Petr: Position embedding transformation for multi-view 3d object detection. In *European Conference on Computer Vision*, pages 531–548. Springer, 2022.
- [14] Y. Liu, J. Yan, F. Jia, S. Li, A. Gao, T. Wang, and X. Zhang. Petrv2: A unified framework for 3d perception from multi-camera images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3262–3272, 2023.
- [15] S. Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.
- [16] E. Luna, J. C. SanMiguel, J. M. Martínez, and P. Carballeira. Graph neural networks for cross-camera data association. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(2):589–601, 2022.
- [17] C. Otto, D. Wang, and A. K. Jain. Clustering millions of faces by identity. *IEEE transactions on pattern analysis and machine intelligence*, 40(2):289–303, 2017.
- [18] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*, pages 17–35. Springer, 2016.
- [19] A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 410–420, 2007.
- [20] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.
- [21] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.
- [22] N. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants. *Properties, Normalization and Correction for Chance*, 18, 2009.
- [23] Y. Wang, V. Guizilini, T. Zhang, Y. Wang, H. Zhao, and J. Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries, 2021.
- [24] Z. Wang, L. Zheng, Y. Li, and S. Wang. Linkage based face clustering via graph convolution network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1117–1125, 2019.
- [25] Y. Xing, T. He, T. Xiao, Y. Wang, Y. Xiong, W. Xia, D. Wipf, Z. Zhang, and S. Soatto. Learning hierarchical graph neural networks for image clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3467–3477, 2021.
- [26] L. Yang, X. Zhan, D. Chen, J. Yan, C. C. Loy, and D. Lin. Learning to cluster faces on an affinity graph. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2298–2306, 2019.
- [27] L. Yang, D. Chen, X. Zhan, R. Zhao, C. C. Loy, and D. Lin. Learning to cluster faces via confidence and connectivity estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13369–13378, 2020.
- [28] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE international conference on computer vision*, pages 1116–1124, 2015.
- [29] C. Zhu, F. Wen, and J. Sun. A rank-order distance based clustering algorithm for face tagging. In *CVPR 2011*, pages 481–488. IEEE, 2011.