ATTENTION-BASED LINK PREDICTION FOR MULTI-CAMERA MULTI-OBJECT TRACKING

Nektarios Totikos

Technical University of Munich
Chair of Human-Machine Communication

ABSTRACT

In this paper, I present the results of my research project on temporal graph neural networks (GNNs) and graph attention (GAT) for multi-camera multiple object tracking (MC-MOT). More specifically, I outline the generation of the evolving graph for the two-cameras setting and how my approach differs from the DyGLIP algorithm that was the main inspiration of this project. My method predicts labels of existing edges between nodes that represent detected objects, whereas DyGLIP only creates edges if two nodes correspond most likely to the same object. This difference might be the reason for the loss of performance by my approach compared to the DyGLIP algorithm.

***Index Terms*—** Graph Attention, Temporal Graphs, Representation Learning, Object Tracking, Computer Vision

1. INTRODUCTION

With the advancement of deep learning algorithms and their applicability to computer vision, MOT has attracted the interest of many researchers in recent years. The main task hereby is to find the global trajectory of different objects in a data sequence [1]. Depending on the application, there are different types of objects to be tracked, for instance, pedestrians or animals for video surveillance and vehicles for autonomous driving. Regular MOT methods fall into the category tracking-by-detection [1]. These approaches usually extract a set of detected objects from a video sequence by defining so-called bounding boxes and corresponding object IDs. The challenge during the tracking process is to determine a global ID for all different bounding boxes of the same target by associating them together. In comparison to the single camera scenario, even more sophisticated techniques are required to handle the association task in a setting with multiple cameras. Common hurdles that come with the use of more than one camera are frequent occlusions and interactions between objects, varying object poses and features, similarities between different objects as well as several matching trajectories [2][3].

In the literature, there are various proposals to treat MOT as a graph problem, whereby graph nodes represent detected objects and edges of the graph describe the associations of nodes [2]. Based on these graphs we can reformulate the data association, for instance, as a greedy data association or a network flow problem that can be solved using min-cost flow algorithms [4]. Nonetheless, the underlying graphs are in general static graphs that fail to capture global interactions between objects [2]. No matter the method, reducing failures in data association is crucial to improve the overall performance of multi-camera MOT algorithms.

Quach et al. [3] tackle this problem by introducing a dynamic graph model with link prediction (DyGLIP). This promising approach motivated the conducted research project and is therefore covered in Section 3 and Section 4 in more detail. In this paper, I showcase my interpretation of dynamic link prediction via graph neural networks for multi-camera MOT inspired by the DyGLIP framework. The focus is on crowded scenes with pedestrians, i.e. 2D data from videos recorded by multiple overlapping cameras.

The structure of the paper is described as follows: Section 2 gives an overview of the relevant theory and introduces important definitions. My work is placed in the existing literature in Section 3. The actual approach to solving the data association is presented in Section 4 and the conducted experiments are specified in Section 5. Finally, Section 6 summarizes the purpose of this research project and sketches potential directions of future work.

2. BACKGROUND

This section introduces the theoretical tools that I used to model the data association as a graph learning problem. Special focus is put on the design of the evolving graph and graph attention.

2.1. Person Re-Identification Features

Suppose you would like to monitor people's behavior in your favorite public city mall using multiple cameras with non-overlapping field of views. Your goal is to identify them correctly and therefore, you assign a unique ID to every detected person. Nevertheless, each camera has a different location,

Visit https://github.com/ntotikos/temporal-gnn-tracking for the code.

timestamp and resolution and captures a different set of people. Due to these variations the person's appearance might change significantly and matching the same persons together gets harder. What kind of information would you use to distinguish between different persons and how would you ensure that you matched the variations of a person correctly to the original person? How about potential imposters, i.e. people with a similar appearance but a different identity? These questions fall into the person re-identification (ReID) step. Recent ReID models use deep convolutional neural networks to learn robust feature representations of the original person that resist the mentioned problems. One popular model, also used in this study, is ResNet-50 [5]. All together, feature vectors connected to a specific person are created by object trackers and ReID models [3].

2.2. Graph Definition

Alone with your smartphone you produce vast amounts of data. Think about your last speech recording, your holiday photos or even the last video you uploaded to your favorite social media platform - these types of data exhibit a grid structure and thus inherit some geometrical properties. The training of convolutional neural networks (CNNs) has achieved great results on this type of data [6], which is why researchers focused on generalizing deep learning algorithms to more complex data that are defined on unstructured meshes, as it is the case in e.g. social networks. To describe these meshes we introduce the term graph. A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is composed of a node set $\mathcal{V}$ and an edge set $\mathcal{E}$, and can be mathematically described by an adjacency matrix $\mathbf{A}$. If an edge exists between two nodes u and v the value for $\mathbf{A}_{(u,v)}$ equals 1, else 0. Graphs can be directed, i.e. we can reach a node u from a node v and cannot go back, and undirected if we can return to u from v. Note that the proposed graph is directed.

For dynamic graphs we emphasize the time dependency with the time t and get a set of graphs $\mathcal{G}^{(t)} = (\mathcal{V}^{(t)}, \mathcal{E}^{(t)})$ [3]. The graph at time step t is called a graph snapshot in this paper. Evolving graphs describe graphs that grow in time, which means that the number of nodes and edges increases at every time step. The node set $\mathcal{V}^{(t)}$ at time t can be split into two disjoint sets, i.e. $\mathcal{V}^{(t)} = \mathcal{V}^{(t-1)} \cup \mathcal{N}^{(t)}$ [3], with $\mathcal{V}^{(t-1)}$ and $\mathcal{N}^{(t)}$ denoting the set of historic and new nodes at time t.

2.2.1. Graph Attention Layer

The attention mechanism has been successfully applied to several domains and got popular due to its breakthrough in natural language processing, as it addresses the problem in relating entities in long sequences [7]. With attention we can focus on the relevant parts of the input for the decision making. Inspired by this method Veličković et al. introduced the graph attention (GAT) layer [8] that enables the weighting of features coming from neighboring nodes.

Let's assume that a node u in our static graph $\mathcal{G}$ has the features $\mathbf{h}_u \in \mathbb{R}^F$ with F being the number of node features. These features are transformed using a weight matrix $\mathbf{W} \in \mathbb{R}^{F' \times F}$, where F' is the output dimension of the GAT layer. The attention mechanism uses a weight vector $\mathbf{a} \in \mathbb{R}^{F'}$ and is applied using the LeakyReLU function and normalized with the softmax function. Mathemacially, the attention coefficients are calculated via

$$a_{uv} = \frac{\exp\left(\text{LeakyReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{h}_u || \mathbf{W}\mathbf{h}_v])\right)}{\sum_{k \in \mathcal{N}_u} \exp\left(\text{LeakyReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{h}_u || \mathbf{W}\mathbf{h}_k])\right)} \quad (1)$$

where v is a node in the one-hop neighborhood $\mathcal{N}_u$ of node u and $||$ indicates the concatenation operation. The output of the GAT layer is finally given as

$$\mathbf{h}_u^{'} = \sigma\left(\sum_{v \in \mathcal{N}_u} a_{uv} \mathbf{W}\mathbf{h}_v\right). \quad (2)$$

Multi-head attention with K heads means to independently compute the coefficients K times and concatenate them [7].

3. RELATED WORK

Data association plays an integral role in multiple object tracking. MOT algorithms can be split into offline and online methods, where the former perform better in general because they make use of future frames in a recorded sequence of frames. However, offline methods are not suited for real-time tracking, as the underlying optimization algorithms introduce a delay due to their algorithmic complexity [9].

In order to tackle the association problem, Quach et al. introduce the DyGLIP algorithm, an online method performing dynamic link prediction on evolving graphs using GAT layers [3]. More specifically, spatial attention is applied to capture meaningful variations between nodes and temporal attention is used to get temporal information across graph snapshots. One node in this setting equals one tracklet of an object. Edges between nodes only exist if nodes represent the same object. More specifically, link prediction is performed to create an edge between two nodes or leave them unconnected. Though DyGLIP is designed to tackle the online association task, it provides results comparable to offline methods [3].

There are also non-probabilistic algorithms used to solve the data association that are part of the greedy search algorithm family [10]. These greedy algorithms are prone to errors when objects interact with each other or occlude others [4]. Another non-probabilistic approach is called Lagrangian relaxation which proposes sub-optimal solutions in order to reduce the computational complexity [11] resulting from relaxed optimization constraints. It can be reformulated as a global network flow problem that can be solved using min-cost flow algorithms, turning it into a probabilistic graphical model [11]. With this formulation the number of association hypotheses are reduced and occlusion can be handled. Data

association might be also solved with a data-driven approach using deep neural networks. One example is the deep affinity network [12], which predicts the optimal assignment using assignment matrices for training.

The main inspiration for the conducted research project is the DyGLIP algorithm. The difference is that no temporal attention is used in my approach and that the graph is constructed another way. The details of my approach are presented in the next section.

4. ARCHITECTURE AND METHODOLOGY

Though the DyGLIP paper was the main inspiration for this research project, I modeled the underlying graph differently than explained in [3] because of the ambiguity in the design. This section therefore explains the generation of the temporal graph dataset and how link prediction is performed, while emphasizing the differences between my approach and DyGLIP.

4.1. Backbone Model

DyGLIP and my algorithm are based on appearance features of detected objects that are extracted from the images. One camera i in the multi-camera setting produces a sequence of frames $V_i = \{I_0, I_1, ..., I_{T-1}\}$ each of which shows several objects. Depending on the number of frames T and the number of detected objects per frame the number of detected objects differs across cameras. Bounding boxes defined by 2D coordinates indicate the position of a specific object in an image. The coordinates can be used to crop out the bounding boxes containing the objects of interest. These cropped boxes are resized to a height $h = 672$ and width $w = 224$, and then fed into the chosen backbone model, namely a ResNet-50 model pre-trained on ImageNet. This model extracts the relevant ReID features for every detected object. The resulting feature vectors are of dimension $k = 2048$, fixed by the ResNet-50 output.

4.2. Temporal Graph Dataset

A temporal graph dataset consists of several snapshots $\{G^{(0)}, G^{(1)}, ..., G^{(T-1)}\}$ of a graph that capture the graph at different time steps. In my implementation, every new detection at time t is modeled as a new node and no node is omitted, i.e. the graph is strictly growing. Every node is equipped with a ReID feature vector of size $k = 2048$, a unique node ID and the ground truth node label. The incorporation of edges is not trivial and therefore explained in the following for the two-camera setting.

At every time step t we have two sets of nodes that originate from both cameras: historic and new nodes. A node is called historic if it already existed at the instance $t-1$ and new if it was detected at time t. Note that for $t = 0$ there are no historic nodes. An edge is created between two nodes if both nodes are likely to represent the same object. We can exclude a connection between new nodes of the same camera because one object is detected in one frame of one camera only once. Every new node is connected to every historic node and every new node of camera 1 is connected to every new node of camera 2. Let's assume that one object was detected by camera 1 and two objects by camera 2. We thus have three new nodes in the graph snapshot at $t = 0$ and no historic nodes, and create edges between nodes of camera 1 and camera 2 nodes. This corresponds to the rectangles A and B in Fig. 1. At $t = 1$ we get one new node for both cameras, respectively, and connect them to all nodes from the previous step. This is exemplified in the segments C and D of Fig. 1. The graph evolves analogously at $t = 2, 3, ..., T-1$ with every new node and edge.

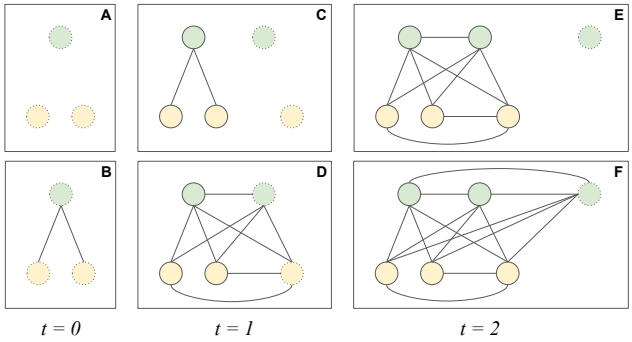


Fig. 1. The graph evolution is shown for three time steps. Determine new and historic nodes (A, C, E) and then create edges accordingly (B, D, F). $t = 0$ corresponds to frame A and B, $t = 1$ to C and D, $t = 2$ to E and F. New nodes are indicated by dotted circles and historic nodes by continuous circles. Within one frame, the upper nodes relate to camera 1 and the lower ones to camera 2.

In contrast to my approach, DyGLIP creates an edge between two nodes only if the predictions for both nodes are the same. More specifically, the term link prediction in the DyGLIP paper [3] refers to the creation of a link between two nodes and in my case labels of existing edges are predicted. The next section explains my interpretation of link prediction and hence the edge labeling.

4.3. Link Prediction

Once the temporal graph dataset is ready, a GNN can be trained to perform link prediction. The task is to predict the edge labels, which are 1 if they are likely to connect two nodes that represent the same object, and 0 else. To this end, the original ReID features are transformed by applying multi-head attention layers and the cosine distance is introduced as a similarity measure between these transformed node features. Using Sigmoid on these distances we get similarity scores that are the basis for the classification.

5. EXPERIMENTS

This section gives an overview of the implementation details, including the used dataset, preprocessing and the temporal graph models for the link prediction. It is rounded off by the results obtained during training and testing.

5.1. WILDTRACK and Preprocessing

The experiments were performed on the WILDTRACK dataset [13] which is a collection of video sequences capturing walking people. It includes 7 synchronized and static cameras with highly overlapping field of views and comprises 400 frames per camera. Each frame has a resolution of 1920x1080 pixels and is equipped with annotations describing the positions of bounding boxes for detected objects. Note that I used two cameras for the sake of simplicity.

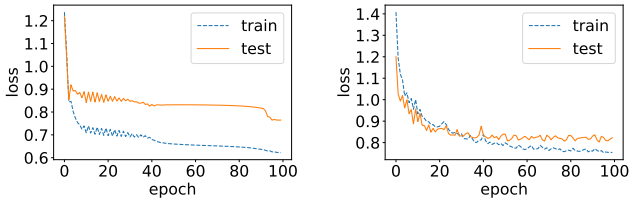
Using the annotations and the pixel coordinates of the bounding boxes, the detected pedestrians were cropped out of the original frame, resized and passed to a ResNet-50 model for ReID feature extraction. Bounding boxes exceeding the boundaries of a frame were omitted.

5.2. Experimental Setup

The goal of the MC-MOT experiments was to predict the labels of edges between nodes at time t, thus assigning a global ID to the detected persons. The evolving graph with corresponding graph snapshots was modeled using PyTorch Geometric Temporal [14]. The temporal graph dataset was split into a train and test set using a ratio of 0.8. The attention coefficients and model parameters were learned by means of the Adam optimizer. For the implementation of the attention layers PyTorch Geometric was used. Other than described in the DyGLIP paper [3], I implemented spatial attention without additional temporal attention. Two spatial attention layers with four heads and output size 128 were applied to the ReID features. For the predictions, the cosine distances of the transformed node features were computed and mapped to a score value using the Sigmoid function. Edges with a high score were labeled with 1, and 0 else. The loss function was chosen to be the binary cross entropy loss.

5.3. Training and Results

I created two different evolving graphs: one covering the single-camera setting and another one that is based on two cameras. In both cases, I consider 10 frames per camera. The number of epochs is set to 100, the learning rate is chosen to be 0.0001 and an additional weight decay of 5e-4 is considered for the optimization task. The gap between train and test loss is smaller in the two-camera case (see Fig 2). The losses decrease in both cases and reach a similar level. In addition, I considered 100 frames per camera but no real improvement was achieved compared to 10 frames per camera.



One camera. Two cameras.

Fig. 2. Train and test losses for single-camera and two-camera setting and 10 frames per camera.

6. CONCLUSION

This paper summarizes how temporal graphs and GAT can be used to reformulate the data association task in MC-MOT and explains how the implemented solution differs from its inspirational source, the DyGLIP algorithm [3]. The main differences between both approaches are the creation of the evolving graph and the use of attention modules. The results do not show any benefit of my approach to performing dynamic link prediction using graph attention and therefore leave large space for improvement. Incorporating more frames and camera views provides more variations in the appearance of detected objects, and hopefully more accurate results. The ReID feature extraction step can be adjusted by actually training the ResNet-50 model on pedestrian data. Further, instead of representing one detection by one node, one node could represent a tracklet, i.e. a sequence of detections of the same object. By averaging over the features of all sequence elements the feature representation of a tracklet might become more robust.

I modeled the graph as a growing graph and edges are established between nodes that might represent the same object. This introduces unnecessary connections in the graph that will be labeled with 0, indicating that both objects differ. There are two improvements that might be relevant: when connecting new and historic nodes, one could create an edge between new nodes and one out of multiple historic nodes that represent the same object. Alternatively, creating edges between nodes only if both nodes are likely to represent the same object as described in [3] might improve the performance.

While the authors of the DyGLIP paper claim that their method leverages moving patterns [3], it occurs to me that their approach is based on appearance features of the objects. Future work therefore includes for instance the incorporation of 3D coordinates of the objects and their moving directions.

7. REFERENCES

[1] Gioele Ciaparrone, Francisco Luque Sánchez, Siham Tabik, Luigi Troiano, Roberto Tagliaferri, and Francisco Herrera, "Deep learning in video multi-object tracking:

A survey," *Neurocomputing*, vol. 381, pp. 61–88, mar 2020.

[2] Jiahe Li, Xu Gao, and Tingting Jiang, "Graph networks for multiple object tracking," in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 708–717.

[3] Kha Gia Quach, Pha Nguyen, Huu Le, Thanh-Dat Truong, Chi Nhan Duong, Minh-Triet Tran, and Khoa Luu, "Dyglip: A dynamic graph model with link prediction for accurate multi-camera multiple object tracking," *CoRR*, vol. abs/2106.06856, 2021.

[4] Asad A. Butt and Robert T. Collins, "Multi-target tracking by lagrangian relaxation to min-cost network flow," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1846–1853.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[6] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun, "Spectral networks and locally connected networks on graphs," 12 2013.

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *ArXiv*, vol. 1409, 09 2014.

[8] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Li, and Yoshua Bengio, "Graph attention networks," 2017.

[9] Kwangjin Yoon, Du Yong Kim, Young-Chul Yoon, and Moongu Jeon, "Data association for multi-object tracking via deep neural networks," *Sensors*, vol. 19, no. 3, 2019.

[10] Robert A. Murphey, Panos M. Pardalos, and Leonidas S. Pitsoulis, "A greedy randomized adaptive search procedure for the multitarget multisensor tracking problem," in *Network Design: Connectivity and Facilities Location*, 1997.

[11] Patrick Emami, Panos M. Pardalos, Lily Elefteriadou, and Sanjay Ranka, "Machine learning methods for solving assignment problems in multi-target tracking," *CoRR*, vol. abs/1802.06897, 2018.

[12] ShiJie Sun, Naveed Akhtar, HuanSheng Song, Ajmal Mian, and Mubarak Shah, "Deep affinity network for multiple object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 104–119, 2021.

[13] Tatjana Chavdarova, Pierre Baqué, Stéphane Bouquet, Andrii Maksai, Cijo Jose, Louis Lettry, Pascal Fua, Luc Van Gool, and François Fleuret, "The WILDTRACK multi-camera person dataset," *CoRR*, vol. abs/1707.09299, 2017.

[14] Benedek Rozemberczki, Paul Scherer, Yixuan He, George Panagopoulos, Maria Sinziana Astefanoaei, Oliver Kiss, Ferenc Béres, Nicolas Collignon, and Rik Sarkar, "Pytorch geometric temporal: Spatiotemporal signal processing with neural machine learning models," *CoRR*, vol. abs/2104.07788, 2021.