Multi-Camera Trajectory Forecasting with Trajectory Tensors

Olly Styles, Tanaya Guha, and Victor Sanchez

Abstract—We introduce the problem of multi-camera trajectory forecasting (MCTF), which involves predicting the trajectory of a moving object across a network of cameras. While multi-camera setups are widespread for applications such as surveillance and traffic monitoring, existing trajectory forecasting methods typically focus on single-camera trajectory forecasting (SCTF), limiting their use for such applications. Furthermore, using a single camera limits the field-of-view available, making long-term trajectory forecasting impossible. We address these shortcomings of SCTF by developing an MCTF framework that simultaneously uses all estimated relative object locations from several viewpoints and predicts the object's future location in all possible viewpoints. Our framework follows a *Which-When-Where* approach that predicts in *which* camera(s) the objects appear and *when* and *where* within the camera views they appear. To this end, we propose the concept of *trajectory tensors*: a new technique to encode trajectories across multiple camera views and the associated uncertainties. We develop several encoder-decoder MCTF models for trajectory tensors and present extensive experiments on our own database (comprising 600 hours of video data from 15 camera views) created particularly for the MCTF task. Results show that our trajectory tensor models outperform coordinate trajectory-based MCTF models and existing SCTF methods adapted for MCTF. Code is available from: https://github.com/olly-styles/Trajectory-Tensors

Index Terms—Trajectory forecasting, multi-camera tracking, person re-identification, multi-camera trajectory forecasting.

1 Introduction

Predicting the future trajectory of objects in videos is a challenging problem with multiple application domains such as intelligent surveillance [1], autonomous driving [2], person re-identification (RE-ID) [3], and traffic monitoring [4]. Existing works in this area focus on single-camera trajectory forecasting (SCTF), i.e., predicting a future trajectory of an object within the same camera view in which the object is observed [2], [5]–[11]. A critical drawback of the single-camera settings is that models cannot anticipate when new objects will enter the scene [12] as they are limited to the data from a single camera. Furthermore, SCTF methods are only suitable for short-term trajectory forecasting, typically 1 to 5 seconds [6]–[11], due to the limited field-of-view. The constraint of a single camera viewpoint must be removed to overcome these issues. To this end, we introduce *multi-camera trajectory forecasting* (MCTF) - a new framework within trajectory prediction. Given the information about an object's location in one or more camera views, we want to predict its future location across a camera network in all possible camera views. Fig. 1 presents an overview of the MCTF framework.

Tracking an object-of-interest across a large camera network requires simultaneously running state-of-the-art algorithms for object detection, tracking, and RE-ID. These algorithms can be computationally demanding, particularly for large camera networks. Processing videos at a lower image resolution or frame-rate may reduce the computational demands, but this often results in missed detections [13]. Alternatively, we may choose to monitor only a subset of the cameras in a network, resulting in missed detections. A successful MCTF model can mitigate this issue by preempting an object's location in a distributed camera network, allowing the system to monitor fewer cameras through an intelligent selection technique. This is distinct from previous works that use trajectory information for person RE-ID [3] or vehicle tracking [14] which are *reactive* to observations *after* an object has been observed in multiple camera views. Our proposed MCTF framework is *proactive* - it predicts the future location of an object *before* it enters the camera view.

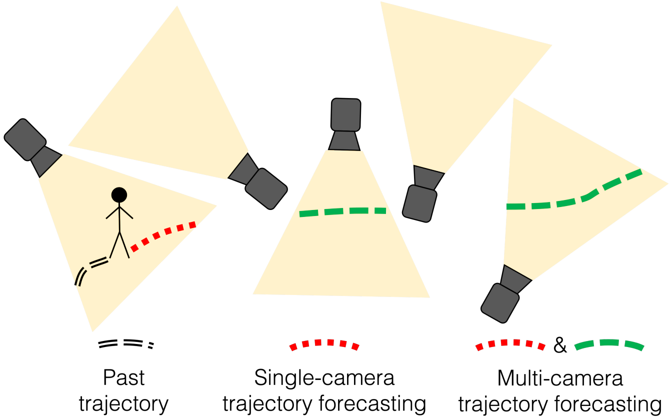

Fig. 1. **Multi-camera trajectory forecasting**. We introduce a novel formulation of the trajectory forecasting task which considers multiple camera views. Companion video: https://youtu.be/IjlNEvKQ634

We present a deep encoder-decoder approach to MCTF that introduces the idea of *trajectory tensors* - a new technique to encode trajectories across multiple camera views and the associated uncertainty in future locations. Trajectory tensors are an attractive alternative to the coordinate-

- *The authors are with the Department of Computer Science, University of Warwick, United Kingdom. Contact email: o.c.styles@warwick.ac.uk*

Manuscript received November 13, 2020; revised May 12, 2021.

based trajectory, which is the de facto representation in existing trajectory forecasting works [6]–[11], with only a few recent exceptions [15]. Coordinate trajectories represent the historical and predicted future locations of an object as a sequence of coordinates. The representation considers coordinates in the image space in a single-camera view or world coordinates if a projection is available. In contrast to the coordinate approach, proposed trajectory tensors divide viewpoints from several cameras into grid cells with values indicating an object's presence or absence. This representation enables us to intuitively model diverse future locations, their associated uncertainty, and object locations in an arbitrary number of viewpoints. Trajectory tensors also offer easy to interpret results that can be visualized easily.

Given an object tracklet in one or more camera view(s), our MCTF framework comprises the following three tasks: (i) In *which* cameras will the object appear next? (ii) *When* will the object appear in those cameras? (iii) *Where* will the object appear in the identified camera views? We use an object tracklet of 2 seconds to predict 12 seconds into the future. Owing to the wide body of complementary literature on pedestrian detection [16] and person RE-ID [17], we focus on pedestrians for our MCTF task. Nevertheless, the task, along with our proposed data representation and model, can be easily generalized to any moving object.

MCTF and the Warwick-NTU Multi-camera Forecasting (WNMF) dataset were first introduced in our earlier work [18]. This work extends our previous work as follows:

- We introduce three problem formulations within the MCTF framework: *which*, *when*, and *where* (Section 3). Each task predicts trajectories with more granularity than the previous.
- We propose trajectory tensors, a new data representation for multi-camera trajectories (Section 4.1). Trajectory tensors overcome many of the shortcomings of coordinate trajectories for MCTF.
- We present a deep encoder-decoder for MCTF (Section 4.2). Our novel approach, based on trajectory tensors, outperforms several relevant baselines (Section 5).

2 Related work

To the best of our knowledge, our work is the first to consider trajectory forecasting in a non-overlapping multi-camera setting. Therefore, we review literature in the most related fields.

Single-camera trajectory forecasting. Trajectory forecasting has seen considerable attention in recent years, generally focusing on forecasting pedestrian trajectories from a birds-eye viewpoint [19], [20]. Many approaches use recurrent models such as long short-term memory (LSTM) networks and focus on extracting features to encode social norms such as avoiding collisions with others, and group movements [6]–[8]. Other works also consider environmental constraints [21]. A comparatively small number of works consider visual features for trajectory forecasting, such as those extracted from human pose [9], [22] or optical flow [10] from non-nadir viewpoints. Models are often optimized using loss functions such as mean squared error (MSE) [9], [10], [23]. As the distribution of future trajectories is multi-modal, using the MSE loss heavily penalizes reasonable but incorrect forecasts, such as turning right rather than left at an intersection. Rather than using MSE, some works use an adversarial loss to train models capable of generating multiple possible future paths using generative adversarial networks (GANs) [7], [8] or variational auto-encoders (VAEs) [2], [24]. However, evaluating multi-output models can be challenging, as typically, only a single ground truth trajectory exists despite multiple plausible futures. Liang et al. overcome this issue by generating a simulated dataset where each trajectory has multiple futures [15]. Note that all of the works mentioned above consider only a single camera viewpoint at a time.

Person RE-ID. Person RE-ID is the task of identifying an individual in a set of gallery images given a probe image, often in a multi-camera setting. Substantial progress has been made in this area over recent years. For example, on the commonly used Market [25] benchmark, rank 1 person RE-ID performance (i.e. correct RE-ID given 1 guess only) has improved from 47.3% in 2015 [25] to 96.8% in 2020 [26]. Most work on person RE-ID has focused on image-level matching, and current state-of-the-art methods [27], [28] exploit this visual cue without other sources of information. However, image-level similarity matching is just a single component of a comprehensive RE-ID system. Persons must first be detected and tracked before matching, which can be computationally expensive to run simultaneously on a large network of cameras. Incorporating trajectory information for RE-ID in a multi-camera setting has seen comparatively little attention, in part due to a lack of publicly available datasets. One notable exception is the recent work of Wang et al. [3]. The authors demonstrate that by retrospectively utilizing trajectory information and visual features, their approach can attain state-of-the-art RE-ID results on the prevalent Duke-MCMT benchmark dataset [29]. The Duke-MCMT dataset is no longer available to the research community [30].

Multi-camera surveillance. A typical automated multi-camera surveillance system consists of detection, tracking, and RE-ID components to monitor objects-of-interest [31]–[33]. Scaling such systems to large camera networks can be challenging due to computational demands. Jain et al. [32] study the impact of scaling multi-camera tracking to large networks and show that filtering the search space to high traffic areas can considerably reduce the search space at only a small cost in recall. In addition to using the traffic level in each area, trajectory information has also been used for both RE-ID [3] and multi-camera tracking [14]. Using trajectories for these tasks can supplement existing appearance-based models by providing a second source of information for matching objects across camera views. However, previous works [3], [14], [31]–[33] reactively use trajectory information, i.e., the object must have already been observed in multiple camera views. Alternatively, Alahi et al. [34] predict the origin and destination of pedestrians using a dense multi-camera setup. The authors use nadir viewpoints and depth sensors to map tracks to the world coordinate space. Our method, in contrast, predicts trajectories in the image space. Our method is therefore applicable to a typical video surveillance setup with non-overlapping views and standard cameras without depth sensors.

3 The MCTF framework

Consider a typical multi-camera surveillance setup where a set of k cameras $\mathcal{C} = \{c_i\}_{i=1}^{k}$ are mounted overhead monitoring objects of interest. Given an object's bounding boxes from the previous n timesteps up to the current timestep t, $B_{(t-n:t)} = \{B_{(t-n)}, \cdots B_{(t-1)}, B_{(t)}\}$, we propose a hierarchy of MCTF problem formulations.

In *which* camera(s) will the object appear in the future? Given $B_{(t-n:t)}$, our task is to identify a subset of $\mathcal{C}$ in which the object may appear at any future timestep, up to a maximum of m timesteps. We cast this as a multi-class multi-label classification problem, where an object can appear in any number of cameras in $\mathcal{C}$. Our goal is to estimate the probability of appearance of the object $P_a(c_i|B_{(t-n:t)})$ for each camera $c_i \in \mathcal{C}$, where a positive class is a camera in which the object re-appears. The output is a vector $[P_a(c_1|B_{(t-n:t)}), \ldots, P_a(c_k|B_{(t-n:t)})]$ of length k.

***When* will the object appear?** The task here is to predict when the object will reappear within the next m timesteps in a given camera. Similar to the *Which* problem, we also formulate this as a multi-class multi-label problem, where we compute the joint probability of appearance of the object $P_a(c_i, t_j|B_{(t-n:t)})$ for each camera $c_i \in \mathcal{C}$ at each timestep t_j with $j = 1, \cdots, m$. The output is a matrix $\left[\begin{smallmatrix} P_a(c_1,t_1|B_{(t-n:t)}) & \cdots \\ \cdots & P_a(c_k,t_m|B_{(t-n:t)}) \end{smallmatrix}\right]$ of dimension $k \times m$.

***Where* will the object appear?** This task aims at spatially localizing the object within a camera view in addition to *which* and *when*. To this end, we divide each camera view into a $w \times h$ grid and predict a probability of appearance score $P_a(c_i, t_j, g_{xy}|B_{(t-n:t)})$ for each grid cell g_{xy}, where $x = 1, \cdots, w$ and $y = 1, \cdots, h$. The output is a tensor $\mathbf{Z}$ of dimension $k \times m \times w \times h$ containing the probability of appearance scores $P_a(c_i, t_j, g_{xy}|B_{(t-n:t)})$ for each camera $c_i \in \mathcal{C}$, timestep t_j with $j = 1, \cdots, m$, and grid cell g_{xy} with $x = 1, \cdots, w$ and $y = 1, \cdots, h$.

It is important to note that this work focuses on modeling target locations in a multi-camera setting where accurate geometry information is not available. We predict the location of a single target, although our new problem formulations open up the possibility of extending this to multiple targets in future work, which has been widely adopted in single-camera settings [6]–[8].

4 Proposed approach

This section introduces our data representation technique, i.e., trajectory tensors, our models, and evaluation strategy for MCTF.

4.1 Trajectory Tensors

Existing works represent trajectories using a coordinate approach [6]–[8]. Coordinate trajectories are $(x, y)_t$ vectors in the image or world space, representing the location of an object at a particular timestep t. Sometimes, the width (w) and height (h) of the object may be also included in the representation, i.e., $(x, y, w, h)_t$ [10], [35].

There are three critical drawbacks to a coordinate trajectory representation:
(i) Standard coordinate-based approaches generally do not define a null trajectory. This representation can be problematic in real-world scenarios where object coordinates can become unavailable, such as when the object is occluded, or the detection algorithm fails. It is particularly problematic in a multi-camera scenario where objects are not visible in all camera views simultaneously.
(ii) Coordinate trajectories can only represent a trajectory as viewed from a single camera. Due to the null trajectory problem, coordinates cannot be easily generalized to multiple cameras, unless all objects are simultaneously visible in all cameras in $\mathcal{C}$, or separate models are created for each camera. Trajectories can instead be mapped to the world coordinate space to overcome this issue. However, this mapping requires accurate measurements of the camera locations and intrinsic parameters which are not always available. This work assumes that such information is not available.
(iii) Finally, coordinate trajectories do not inherently represent uncertainty, which is intrinsic to the trajectory forecasting task. The space of future trajectories is multi-modal, e.g., an object can travel either left or right at a junction. Existing works address this issue by using generative models to simulate multiple futures [7], [15], from which a probability distribution can be created. Our approach, in contrast, intuitively models uncertainty in one shot.

To overcome the shortcomings of coordinate trajectory representation, we introduce the idea of *trajectory tensors* - a novel technique for compact representation of multi-camera trajectories. As shown in Fig. 2, trajectory tensors are constructed in four steps:
(i) We consider a set of cameras $\mathcal{C} = \{c_i\}_{i=1}^{k}$, where an object of interest may appear in any number of the cameras in $\mathcal{C}$.
(ii) Each camera c_i has an associated detection $\mathbf{d_i}$ representing the object bounding box, if present.
(iii) We convert each $\mathbf{d_i}$ into a heatmap $\boldsymbol{H_i}$ which is an alternative representation of the object's location. A heatmap is a matrix of size $w \times h$, where each entry is a binary value indicating the presence or absence of the object in this grid cell. A single object may span any number of grid cells, depending on its size.
(iv) Finally, the heatmaps $\boldsymbol{H_i}$ for each of the k cameras are stacked along the camera dimension, and computed for t timesteps. We also smooth each heatmap using a Gaussian kernel. The result is a trajectory tensor, $\mathbf{Z}$, of shape $k \times t \times w \times h$, where each entry is a value between 0 and 1. $\mathbf{Z}$ represents the trajectory of an object in multiple camera views simultaneously and can be used to represent both past (inputs) and future (predicted) object locations. Encoding object locations as trajectory tensors allows us to elegantly represent objects in multiple cameras. Trajectory tensors enable the representation of null trajectories rather than discarding this data or using placeholder values.

Trajectory tensors share similarities with the grid-cell representation proposed recently by Liang et al. [15]; however, our proposed encoding includes multiple camera views. Besides, objects represented using trajectory tensors may span multiple grid cells, which allows us to account for variability in object scale. Object scale is not considered in

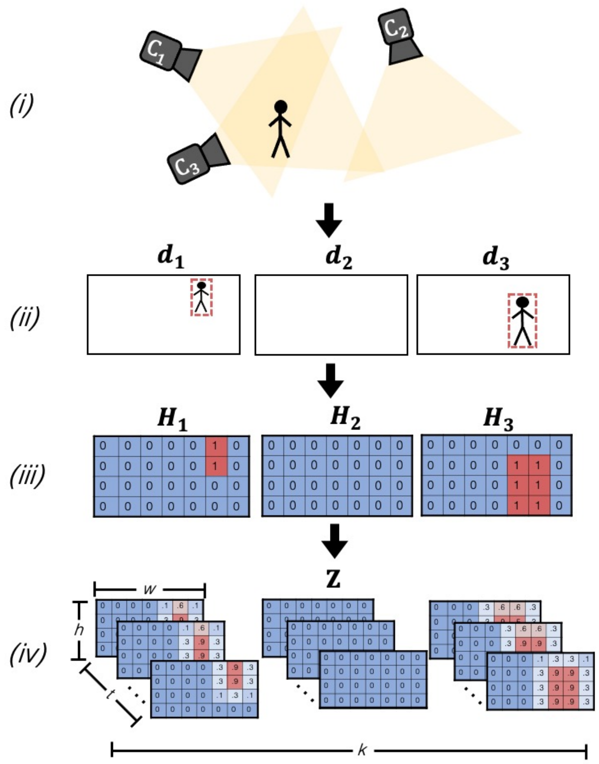

Fig. 2. **Trajectory tensors.** Our proposed trajectory tensors are an intuitive data representation capable of representing object trajectories in multiple camera views, null trajectories, and associated uncertainty.

most SCTF works [6]–[8], [15], but is a critical component in our framework.

4.2 MCTF models

Existing trajectory forecasting methods such as Social-LSTM [6], Social-GAN [7], and SoPhie [8] are designed for SCTF using datasets with birds-eye view cameras [19], [20] that do not include object scale. Although the methods proposed in [9] and [10] forecast object scale in addition to location, they are also designed for SCTF; hence direct comparison for MCTF is not possible. To compare with these existing works, we adapt the methods proposed in [9] and [10] to our MCTF framework. These models, along with new approaches based on trajectory tensors, are summarized in Fig. 3. Each neural network-based model is comprised of a combination of fully-connected, recurrent, and convolutional layers. Inspired by fully-convolutional networks for semantic segmentation [36], we use transposed convolutional layers when predicting trajectory tensors as targets. Due to the large number of neural network models introduced in this section, we introduce each briefly and provide full details of the architectures needed to reproduce our results in the supplementary materials.

Coordinate trajectory approaches

Here we present our coordinate trajectory approaches which are shown in Fig. 3 (a) and (b). Due to the lack of data representations for coordinate trajectories in multiple camera views, our coordinate trajectory approaches each use a separate model for each camera, resulting in k models.

GRU. Recurrent networks have been a prevalent approach for trajectory forecasting in a single camera. We use an adapted version of STED [10], a method for SCTF which uses gated recurrent units (GRUs) with an encoder-decoder architecture. The model proposed in the original work uses two encoders, one for bounding box coordinates, and another which uses a convolutional neural network (CNN) to extract motion information from optical flow. We use only the bounding box encoder for a fair comparison as other methods do not use visual features. For the *which* task, we use a fully-connected classification layer. For the *when* task, we use another GRU as a decoder with 128 hidden units followed by a fully connected classification layer. For the *where* task, the decoder GRU is followed by 2D transposed convolutional (tCNN) layers for spatial upsampling.

LSTM. Our LSTM model is the same as our GRU, with an alternative recurrent unit for both encoder and decoder.

1D-CNN. 1D-CNNs have received some attention as an alternative to the de facto recurrent models for tasks involving time series [37], including trajectory forecasting [9]. We use the encoder architecture proposed by Yagi et al. [9], with modified decoders adapted for the MCTF formulation. For the *which* task, we use a fully-connected classification layer. For the *when* task, we use 1D transposed convolutional layers as a decoder with four layers. For the *where* task, the 1D transposed convolutional layers are followed by 2D transposed convolutional layers for spatial upsampling. Similarly to STED, we use only the trajectory feature extractor for a fair comparison with other methods.

Trajectory tensor approaches

Here we present our trajectory tensor approaches which are shown in Fig. 3 (c), (d), and (e). Our proposed representation enables efficient modeling of object trajectories across multiple camera views. Therefore, each approach consists of a single unified model for all cameras, which is less cumbersome and more scalable than the multi-model approach using coordinate trajectories.

3D-CNN. We use a 3D-CNN with four layers for spatio-temporal feature extraction. 3D convolutions can simultaneously extract spatial and temporal features and have seen some success for tasks such as action recognition [38].

2D-1D-CNN. We use a CNN consisting of three 2D convolutional and pooling layers for spatial feature extraction, followed by three 1D convolutional and pooling layers for temporal feature extraction. Using separate 2D and 1D convolutional layers rather than 3D layers reduces the number of network parameters and is inspired by existing models for video classification [39].

CNN-GRU. We train a convolutional auto-encoder to reduce a trajectory tensor at a single timestep (size $k \times w \times h$) to a feature vector of size 128. This encoding is then used as the input to a GRU which predicts future feature vectors, which are decoded as shown in Fig. 3 (e). The auto-encoder is first pre-trained until convergence and then trained end-to-end with the GRU. This architecture is inspired by [40], which uses a similar strategy of forecasting future convolutional features for predicting future instance segmentation maps.

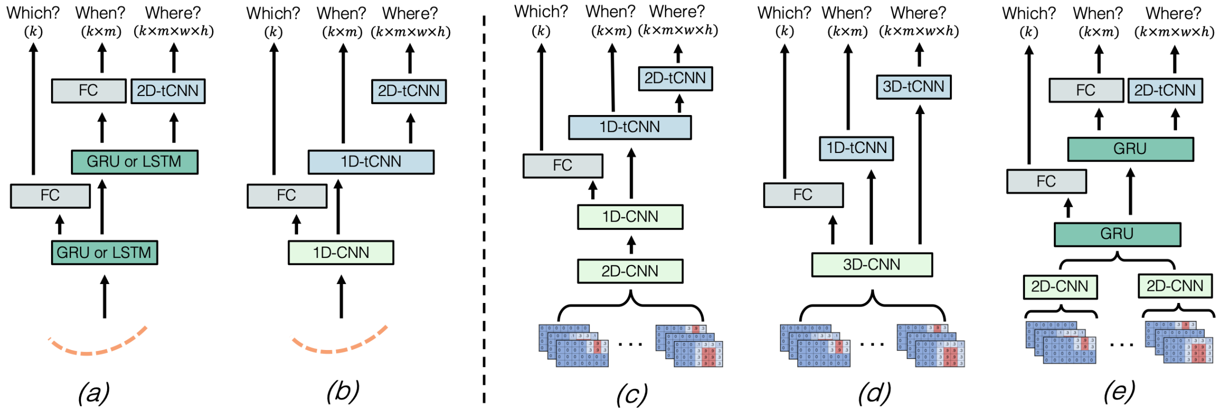

Fig. 3. **MCTF models.** We introduce 2 coordinate-trajectory based (left) and 3 trajectory-tensor based (right) approaches for MCTF. *(a)* A recurrent encoder-decoder adapted from [10]. *(b)* A 1D-CNN adapted from [9]. *(c)* A CNN approach with separated layers for spatial and temporal feature extraction. *(d)* A CNN approach with 3D convolutions for extracting spatial and temporal features simultaneously. *(e)* A hybrid CNN-GRU approach which uses a CNN to extract spatial features which are passed to an GRU for extracting temporal features. Note that for coordinate trajectory models, a separate model is created for each camera. In contrast, trajectory tensor models use a single unified model for all cameras. Each model is trained with either the *which*, *when*, or *where* MCTF formulation.

4.3 Evaluation strategy

Due to the high levels of uncertainty and the multi-modal nature of the MCTF, traditional SCTF metrics such as average and final displacement errors are not well-suited to MCTF without modification. Some works generate multiple trajectories and select the one most similar to the ground truth [7]; however, this evaluation method is optimistic as performance comparable to sophisticated methods can be obtained using a simple constant velocity model that generates trajectories with high variance [41]. We instead compute the average precision (AP) for all problem formulations. We plot precision-recall curves rather than ROC curves. We find ROC to be an overly-optimistic metric due to the considerable class imbalance between the positive (object presence) and negative (no object presence) classes. We define AP_{which}, AP_{when}, and AP_{where} for the three problem definitions respectively.

The AP metrics provide a holistic interpretation of model performance but are not easy to interpret. Therefore, we propose two new metrics for MCTF evaluation, the soft-intersection-over-union ($SIOU$) for the when and where problem formulations. The $SIOU$ for evaluating the *when* formulation is as follows:

$$SIOU_{when} = \frac{1}{|\mathcal{C}^+|} \sum_{c \in \mathcal{C}^+} \frac{\sum_t^{\mathcal{T}^+} \hat{y}_t^c}{\sum_t^{\mathcal{T}} \hat{y}_t^c}, \tag{1}$$

where $\mathcal{C}^+$ and $\mathcal{T}^+$ are sets of true positive cameras and timesteps, respectively. $\hat{y}_t^c \in (0,1)$ is the predicted value for the presence the object in camera c at timestep t. Intuitively, $SIOU_{when}$ is a value between 0 and 1, representing the temporal overlap between the predicted and ground truth timesteps for all cameras where the individual appears.

We compute the $SIOU$ for evaluating the *where* problem as follows:

$$SIOU_{where} = \frac{1}{|\mathcal{C}^+|} \sum_{c \in \mathcal{C}^+} \frac{1}{|\mathcal{T}^+|} \sum_t^{\mathcal{T}^+} \frac{\sum_g^{\mathcal{G}^+} {}^g\hat{y}_t^c}{\sum_g^{\mathcal{G}} {}^g\hat{y}_t^c}, \tag{2}$$

where $\mathcal{G}^+$ is the set of grid cells where the individual is present. $SIOU_{where}$ is similarly a value between 0 and 1 representing the spatial overlap between predicted and ground truth grid cell locations for all true positive cameras and timesteps.

In addition to our new metrics, we also adapt the standard average displacement error (ADE) and final displacement error (FDE) metrics used widely in the SCTF literature [5]–[8]. We refer to these metrics as ADE_{where} and FDE_{where} hereinafter to distinguish them from their SCTF counterparts. We compute a single coordinate value for a heatmap $\boldsymbol{H_i}$ by computing the center of mass, $R_{x,y}$, as follows:

$$R_{x,y} = ((\frac{1}{M} \sum_{m \in \boldsymbol{H_i}} x_m \cdot r_x^m), ((\frac{1}{M} \sum_{m \in \boldsymbol{H_i}} y_m \cdot r_y^m))), \tag{3}$$

where M is the sum of all elements in $\boldsymbol{H_i}$. Note that $\boldsymbol{H_i}$ denotes a heatmap for camera i at a single timestep. r_x^m and r_y^m respectively denote the x and y positions of the m^{th} element in $\boldsymbol{H_i}$. This value corresponds to the centroid of the target bounding box in the image space. As a target can appear in more than one camera view, we compute ADE_{where} and FDE_{where} separately for each camera the target appears and take the mean. ADE_{where} and FDE_{where} are valid for the *where* problem formulation.

5 Performance Evaluation

In this section, we introduce the evaluation dataset, baseline approaches, and assess the performance of each model for the *which*, *when*, and *where* tasks.

5.1 Warwick-NTU multi-camera forecasting dataset

We use the Warwick-NTU Multi-camera Forecasting (WNMF) dataset to train and evaluate each model. WNMF is collected specifically for MCTF using 15 cameras in a building on the Nanyang Technological University campus and contains both overlapping and non-overlapping views recorded over 20 days. The dataset consists of cross-camera trajectories where an individual departs from one camera view and then re-appears in another after no more than 12 seconds. An individual may also be visible in any number of other camera views during this tracking period.

A single individual is labeled for each cross-camera trajectory, and interactions between individuals are limited due to low person density (1.41 individuals per camera per frame on average). Person bounding boxes are generated using pre-trained detection [42] and tracking [43] algorithms and manually verified to be accurate. A human annotator verifies that each track consists of bounding boxes judged to have an IOU of ≥0.5 with the ground truth box. We do not provide identity labels; however, in practice, we observe that many cross-camera trajectories are unique identities due to the long period of data collection. We add two extensions to the dataset compared to our previous release: (i) Multi-viewpoint departures. The original dataset consists of matches across pairs of cameras. We add new annotations for overlapping viewpoints where an individual is visible in multiple cameras simultaneously. (ii) Cleaned annotations. The original dataset contains 2.3K cross-camera matches; however, several erroneous matches have been manually removed, leaving 2.0K matches. The erroneous matches are caused by high visual similarity between different persons during labeling. To verify cross-camera tracks, a human annotator reviews the track pairs proposed by the annotation procedure outlined in [18] by referring to the camera network topology along with the appearance timestamps. Any tracks deemed to be false matches are discarded. More details on the dataset and semi-automated annotation procedure can be found in our previous work [18]. We use a robust 5-fold cross-validation setup in all experiments that follow, where training and testing sets contain footage recorded on different days. The WNMF dataset is available to download from https://github.com/olly-styles/Multi-Camera-Trajectory-Forecasting.

5.2 Baseline approaches

In addition to the models introduced in Section 4.2, we also evaluate several baselines.

Shortest real-world distance. We use the physical distance between cameras in the real world and predict the camera closest to the current camera. This baseline applies to the *which* formulation only.

Training set mean. We consider all training set observations for a particular camera and take the mean of all ground truth labels in the training set.

Most similar trajectory. We find the most similar trajectory in terms of L_2 distance in the same camera from the training set to the observed trajectory and predict the same label.

Hand-crafted features. We extract some features from the bounding boxes and classify them with a fully-connected network. Our 10-dimensional hand-crafted feature vector contains velocity in x and y direction, acceleration in x and y direction, last observed bounding box height and width, and its four coordinates. We compute all features with respect to the 2D coordinate system as captured by the camera.

TABLE 1
***Which* results.** Given observations from one camera, each model predicts which camera(s) the person will re-appear.

	Model	AP_{which} (↑)
Baselines	Shortest real-world distance	45.4
	Training set mean	68.9
	Most similar trajectory	65.3
	Hand-crafted features	76.2
Coordinate trajectories	LSTM	84.1
	GRU	84.0
	1D-CNN	83.3
Trajectory tensors	2D-1D-CNN	87.1
	3D-CNN	**87.5**
	CNN-GRU	86.1

5.3 *Which* camera will they appear?

Experimental setup. We use our proposed network architectures (Fig. 3) and baselines (Section 5.2). For the coordinate trajectory approaches, we train a separate model for each camera. To adapt the SCTF models for MCTF, we leave encoders unchanged and change decoders to fully-connected output layers of size 15, the number of cameras in the WNMF dataset. We use a binary cross-entropy loss function with a sigmoid activation function at the output layer. The activation maps each output to a value between 0 and 1, representing the appearance of the individual in each camera. Coordinate trajectory and trajectory tensor models are trained with the Adam optimizer using learning rates of 1×10^{-3} and 1×10^{-4}, respectively. Coordinate trajectory encoders extract a feature vector of size 128, whereas trajectory tensor approaches extract a feature vector of size 512 as a single encoder is shared across all cameras and therefore requires larger representation capacity. All models are trained using a batch size of 64. We experiment with 3 sizes of input heatmap $\boldsymbol{H}$: 16×9, 32×18, and 48×27. We use a Gaussian smoothing kernel size σ between 0 and 4, chosen using cross-validation. The impact of changing these parameters is investigated in Section 5.6.

Results. The AP_{which} for each model is shown in Table 1 and precision-recall plots are shown in Fig. 4 (left). Coordinate trajectory approaches outperform our 4 baselines, and trajectory tensor approaches perform the best.

5.4 *When* will they appear?

Experimental setup. We use the same encoders for evaluating *when*, with either 1D-transposed convolutional layers or recurrent unit for decoding as shown in Fig. 3. Each model is trained with the binary cross-entropy loss, and the trajectory tensor models are trained with the same hyperparameters as the *which* problem formulation.

Results. The AP_{when} and $SIOU_{when}$ for each model are shown in Table 2, and precision-recall plots are shown in Fig. 4 (center). We observe a similar trend to the results of the *which* task, with the 3D-CNN trajectory tensor model performing best. Some baseline methods perform well in terms of $SIOU_{when}$, but poorly in terms of AP_{when}. This result suggests the most similar trajectory baseline effectively predicts the correct time window an individual will

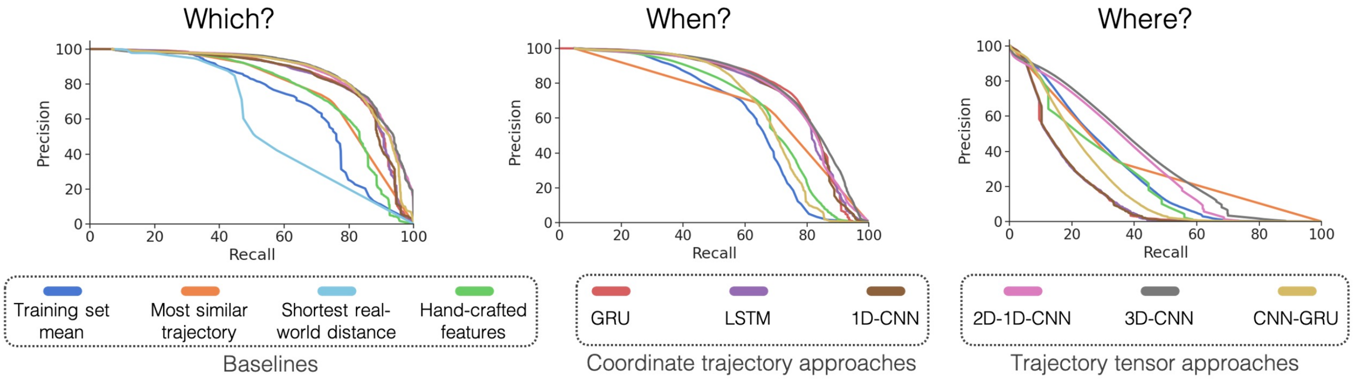

Fig. 4. **Precision-recall plots.** Precision and recall of each model for all three problem formulations. Best viewed in colour.

TABLE 2
***When* results.** Given observations from one camera, each model predicts in which camera(s) the person will re-appear, and in which timesteps they will be present.

	Model	AP_{when} (↑)	$SIOU_{when}$ (↑)
Baselines	Training set mean	61.7	40.0
	Most similar trajectory	46.6	**56.7**
	Hand-crafted features	67.1	48.7
Coordinate trajectories	LSTM	76.8	51.8
	GRU	77.5	53.7
	1D-CNN	77.1	52.8
Trajectory tensors	2D-1D-CNN	77.4	54.4
	3D-CNN	**79.0**	53.9
	CNN-GRU	67.1	41.0

be visible in the target camera but often predicts the wrong camera.

5.5 *Where* will they appear?

Experimental setup. We use the same target heatmap size of 16×9 regardless of the input heatmap size for a fair comparison. We use this small heatmap size to reduce computational complexity, and 16×9 is the smallest possible while maintaining the original aspect ratio of the video frames. The output trajectory tensor is, therefore, of size $[15\times60\times16\times9]$. We do not apply Gaussian smoothing to the ground truth trajectory tensor targets. We use transposed convolutional layers to upsample extracted feature vectors to trajectory tensor outputs, representing an individual's future location. These are 3D transposed convolutions in the 3D-CNN model, and 2D transposed convolutions in the CNN-GRU and 2D-1D CNN models. The trajectory tensor models are trained with the same hyperparameters as in Section 5.3, again using the binary cross-entropy loss.

Results. Results for each model are shown in Table 3 and precision-recall plots are shown in Fig. 4 (right). Note that due to the considerably increased complexity of the *Where* problem formulation compared to the others, the AP is much lower. Results for this problem formulation are more mixed than for the *Which* and *When*, and some methods perform well on one metric but poorly on others. For example, the hand-crafted feature baseline performs the best in terms of ADE_{where} and FDE_{where}, but poorly in terms of AP_{where}. Unlike the AP_{where} metric, the displacement error and $SIOU_{where}$ metrics do not take into account erroneous predictions in camera views that the target does not appear. Therefore, this result suggests that the hand-crafted features baseline accurately forecasts the target trajectory but incorrectly assigns a high likelihood to the target appearing in other camera views.

5.6 Ablation studies

Multi-view trajectory tensors. Our proposed trajectory tensor data representation enables us to model the same individual captured in multiple camera views. To study the impact of multiple camera views, we train our trajectory tensor-based models using only one of the available views, i.e., we set each heatmap at each timestep to the zero matrix for all but one of the camera channels $c \in \mathbf{Z}$. The results in Table 4 show a comparison of our trajectory tensor models with a single-view trajectory (each heatmap is 0 for all but one of the cameras $c \in \mathbf{Z}$) and a multi-view trajectory (where more than one of the camera channels $c \in \mathbf{Z}$ may be non-zero). The results show that the models can exploit the location information available in multiple camera views.

Heatmap size and smoothing. We investigate the impact of heatmap size and standard deviation of the Gaussian filter for smoothing. Larger heatmap sizes afford models more representation power at the cost of a larger number of parameters and a tendency to overfit the training data. On the other hand, smoothing has a regularizing effect by reducing the impact of errors during the detection and tracking stages. We suggest, therefore, that both heatmap size and smoothing σ should be tuned in tandem. Fig. 5 shows the impact of heatmap size and smoothing σ. The results confirm our intuition that larger heatmap sizes generally benefit from more smoothing.

5.7 Discussion and applications

Our results show that learned models using either coordinate trajectories or trajectory tensors as inputs generally outperform other baselines at the three tasks in the MCTF framework. Representing trajectories as tensors allows us to model relative object locations in multiple camera views

TABLE 3
***Where* results.** Given observations from one camera, each model predicts which camera(s) the person will re-appear, in which timesteps they will be present, and where in the camera view they will appear. Best results are highlighted in **bold** typeface, second best are underlined.

	Model	AP_{where} (↑)	$SIOU_{where}$ (↑)	ADE_{where} (↓)	FDE_{where}(↓)
Baselines	Training set mean	28.4	14.4	226.1	260.4
	Most similar trajectory	11.8	**29.8**	312.9	375.4
	Hand-crafted features	25.5	20.2	**216.2**	**255.2**
Coordinate trajectories	LSTM	16.3	8.7	285.9	365.5
	GRU	16.5	8.8	286.0	366.0
	1D-CNN	16.4	8.6	287.2	368.1
Trajectory tensors	2D-1D-CNN	34.5	22.6	225.9	265.8
	3D-CNN	**37.4**	22.9	223.3	279.2
	CNN-GRU	22.4	12.8	280.1	351.7

TABLE 4
Multi-view trajectory tensor results. Comparison of single-view and multi-view trajectories. Observing a trajectory in multiple camera views improves model performance for the *Which* and *Where* tasks.

Model	Multi-view trajectories	AP_{which} (↑)	AP_{when} (↑)	AP_{where}(↑)
2D-1D-CNN	No	81.8	76.6	**37.4**
	Yes	**87.1**	**77.4**	34.5
3D-CNN	No	83.0	77.1	**38.8**
	Yes	**87.5**	**79.0**	37.4
CNN-GRU	No	83.4	66.4	**23.6**
	Yes	**86.1**	**67.1**	22.4

simultaneously, which generally improves MCTF performance when used as inputs and provides an intuitive way to represent multi-modal futures when used as targets. Trajectory tensors are also more elegant than using coordinate trajectories in an MCTF setting, as a single model can be used rather than creating separate models for each camera. Note that our approach and other baselines require that the camera network is the same at training and testing time.

We find that the 3D-CNN architecture consistently performs the best across MCTF tasks. 3D-CNNs have traditionally only seen notable success in settings where vast amounts of data are available for training [38]. However, in our setting, we use resolutions of up to 48×27 rather than the 224×224 or higher commonly used in activity recognition, which considerably reduces the number of parameters and facilitates training on smaller datasets.

The benefits of the MCTF framework are fourfold: (i) Long-term forecasting. This is possible by removing the constraint of a single camera viewpoint. (ii) Intelligent camera monitoring. When tracking a particular object-of-interest across a camera network, a subset of cameras may be monitored intelligently using the predictions from an MCTF model rather than continually monitoring all cameras. (iii) Enhanced tracking. Location predictions may be used in conjunction with a RE-ID model for more robust multi-camera tracking. (iv) Robustness to camera failure. Predicting an individual's location in multiple camera views adds redundancy; i.e., a target may still be identified in the event that one or more cameras on the network are no longer operational.

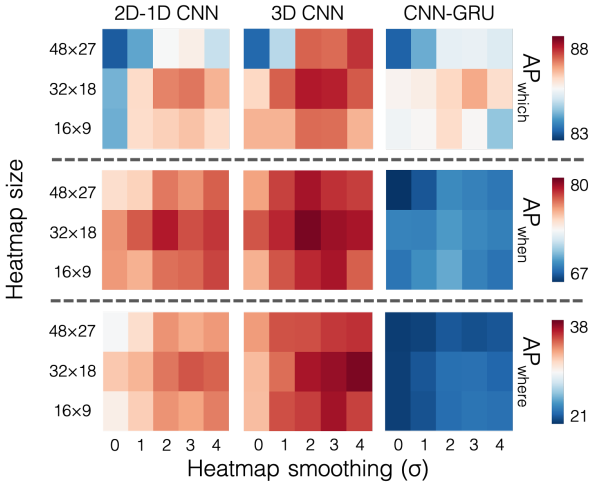

Fig. 5. **Heatmap size and smoothing.** Impact on results of using different heatmap sizes and levels of smoothing (σ). Results shown are computed by 5 fold cross validation.

6 Conclusion

We have developed a complete framework for MCTF formulated in a hierarchy of three spatio-temporal localization tasks: In *which* camera, *when*, and *where* will the object(s) appear? Our work is the first to address the challenges of trajectory forecasting in a multi-camera environment. We introduced the idea of the *trajectory tensor* - a new trajectory representation that facilitates the encoding of multi-camera trajectories and associated uncertainty. Trajectory tensors are an attractive alternative to the traditional coordinate trajectory representation used in previous works. Our proposed MCTF models based on trajectory tensors show promising results on our WNMF database, and we hope to evaluate in more settings when more datasets for MCTF become available. We envision our MCTF model will enhance multi-camera surveillance systems, complementing existing models for person RE-ID and multi-camera tracking.

Acknowledgments

This work is funded by the UK EPSRC (grant number E/L016400/1). Our thanks to NVIDIA for their generous hardware donation. We would also like to thank the ROSE lab at the Nanyang Technological University, Singapore for providing the data used in this research.

References

[1] X. Wang, “Intelligent multi-camera video surveillance: A review,” *Pattern recognition letters*, vol. 34, no. 1, 2013.

[2] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, “Desire: Distant future prediction in dynamic scenes with interacting agents,” in *CVPR*, 2017.

[3] G. Wang, J. Lai, P. Huang, and X. Xie, “Spatial-temporal person re-identification,” in *AAAI*, 2019.

[4] Z. Tang, M. Naphade, M.-Y. Liu, X. Yang, S. Birchfield, S. Wang, R. Kumar, D. Anastasiu, and J.-N. Hwang, “Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification,” in *CVPR*, 2019.

[5] S. Yi, H. Li, and X. Wang, “Pedestrian behavior understanding and prediction with deep neural networks,” in *ECCV*, 2016.

[6] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, “Social lstm: Human trajectory prediction in crowded spaces,” in *CVPR*, 2016.

[7] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, “Social gan: Socially acceptable trajectories with generative adversarial networks,” in *CVPR*, 2018.

[8] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, “Sophie: An attentive gan for predicting paths compliant to social and physical constraints,” in *CVPR*, 2019.

[9] T. Yagi, K. Mangalam, R. Yonetani, and Y. Sato, “Future person localization in first-person videos,” in *CVPR*, 2018.

[10] O. Styles, T. Guha, and V. Sanchez, “Multiple object forecasting: Predicting future object locations in diverse environments,” in *WACV*. IEEE, 2020.

[11] F. Marchetti, F. Becattini, L. Seidenari, and A. D. Bimbo, “Mantra: Memory augmented networks for multiple trajectory prediction,” in *CVPR*, 2020.

[12] H. Minoura, R. Yonetani, M. Nishimura, and Y. Ushiku, “Crowd density forecasting by modeling patch-based dynamics,” *arXiv preprint arXiv:1911.09814*, 2019.

[13] J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” in *CVPR*, 2017.

[14] H.-M. Hsu, T.-W. Huang, G. Wang, J. Cai, Z. Lei, and J.-N. Hwang, “Multi-camera tracking of vehicles based on deep features re-id and trajectory-based camera link models,” in *CVPRW*, 2019.

[15] J. Liang, L. Jiang, K. Murphy, T. Yu, and A. Hauptmann, “The garden of forking paths: Towards multi-future trajectory prediction,” in *CVPR*, 2020.

[16] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele, “Towards reaching human performance in pedestrian detection,” *T-PAMI*, vol. 40, no. 4, 2017.

[17] L. Zheng, Y. Yang, and A. G. Hauptmann, “Person re-identification: Past, present and future,” *arXiv preprint arXiv:1610.02984*, 2016.

[18] O. Styles, T. Guha, V. Sanchez, and A. Kot, “Multi-camera trajectory forecasting: Pedestrian trajectory prediction in a network of cameras,” in *CVPRW*, 2020.

[19] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, “You’ll never walk alone: Modeling social behavior for multi-target tracking,” in *ICCV*. IEEE, 2009.

[20] A. Lerner, Y. Chrysanthou, and D. Lischinski, “Crowds by example,” in *Computer graphics forum*, vol. 26, 2007.

[21] I. Gilitschenski, G. Rosman, A. Gupta, S. Karaman, and D. Rus, “Deep context maps: Agent trajectory prediction using location-specific latent maps,” *IEEE Robotics and Automation Letters*, 2020.

[22] J. Liang, L. Jiang, J. C. Niebles, A. G. Hauptmann, and L. Fei-Fei, “Peeking into the future: Predicting future person activities and locations in videos,” in *CVPR*, 2019.

[23] A. Sadeghian, F. Legros, M. Voisin, R. Vesel, A. Alahi, and S. Savarese, “Car-net: Clairvoyant attentive recurrent network,” in *ECCV*, 2018.

[24] K. Mangalam, H. Girase, S. Agarwal, K.-H. Lee, E. Adeli, J. Malik, and A. Gaidon, “It is not the journey but the destination: Endpoint conditioned trajectory prediction,” in *ECCV*, 2020.

[25] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, “Scalable person re-identification: A benchmark,” in *ICCV*, 2015.

[26] Z. Zhu, X. Jiang, F. Zheng, X. Guo, F. Huang, X. Sun, and W. Zheng, “Viewpoint-aware loss with angular regularization for person re-identification.” in *AAAI*, 2020.

[27] R. Quan, X. Dong, Y. Wu, L. Zhu, and Y. Yang, “Auto-reid: Searching for a part-aware convnet for person re-identification,” *CVPR*, 2019.

[28] H. Luo, Y. Gu, X. Liao, S. Lai, and W. Jiang, “Bag of tricks and a strong baseline for deep person re-identification,” in *CVPRW*, 2019.

[29] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, “Performance measures and a data set for multi-target, multi-camera tracking,” in *ECCV*. Springer, 2016.

[30] J. Satisky, *A Duke study recorded thousands of students’ faces. Now they’re being used all over the world*, 2019 (accessed 20th February, 2020), https://www.dukechronicle.com/article/2019/06/duke-university-facial-recognition-data-set-study-surveillance-video-students-china-uyghur.

[31] N. Martinel, G. L. Foresti, and C. Micheloni, “Person reidentification in a distributed camera network framework,” *IEEE transactions on cybernetics*, vol. 47, no. 11, 2016.

[32] S. Jain, G. Ananthanarayanan, J. Jiang, Y. Shu, and J. Gonzalez, “Scaling video analytics systems to large camera deployments,” in *International Workshop on Mobile Computing Systems and Applications*, 2019.

[33] O. Camps, M. Gou, T. Hebble, S. Karanam, O. Lehmann, Y. Li, R. J. Radke, Z. Wu, and F. Xiong, “From the lab to the real world: Re-identification in an airport camera network,” *IEEE transactions on circuits and systems for video technology*, vol. 27, no. 3, 2016.

[34] A. Alahi, V. Ramanathan, and L. Fei-Fei, “Socially-aware large-scale crowd forecasting,” in *CVPR*, 2014.

[35] Y. Yao, M. Xu, C. Choi, D. J. Crandall, E. M. Atkins, and B. Dariush, “Egocentric vision-based future vehicle localization for intelligent driving assistance systems,” in *ICRA*, 2019.

[36] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *CVPR*, 2015.

[37] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, “Deep learning for time series classification: a review,” *Data Mining and Knowledge Discovery*, 2019.

[38] K. Hara, H. Kataoka, and Y. Satoh, “Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?” in *CVPR*, 2018.

[39] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, “Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification,” in *ECCV*, 2018.

[40] P. Luc, C. Couprie, Y. Lecun, and J. Verbeek, “Predicting future instance segmentation by forecasting convolutional features,” in *ECCV*, 2018.

[41] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll, “What the constant velocity model can teach us about pedestrian motion prediction,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, 2020.

[42] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *ICCV*, 2017.

[43] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in *ICIP*, 2017.

Olly Styles is working towards a PhD degree at the University of Warwick, UK. He received an MEng degree in Computer Science, also from the University of Warwick. His research interests include vision-based forecasting, object tracking, and self-supervised learning. During his PhD, he has collaborated with several other research groups as a visiting scholar including the iProbe Lab at Michigan State University, USA, and the ROSE lab at Nanyang Technological University, Singapore.

Tanaya Guha is an Assistant Professor in the Department of Computer Science at the University of Warwick, UK. She has received her PhD in Electrical and Computer Engineering from the University of British Columbia (UBC), Vancouver, Canada. Her research is focused on modeling and analysis of multimedia data combining machine learning and signal processing with applications in media understanding, healthcare and smart surveillance. She has served in the Program Committee of several conferences including ACM Multimedia, IEEE ICME and Interspeech.

Victor Sanchez received the MSc degree from The University of Alberta, Canada, in 2003, and the PhD degree from The University of British Columbia, Canada, in 2010. From 2011 to 2012, he was with the Video and Image Processing (VIP) Laboratory of The University of California at Berkeley as a Postdoctoral Researcher. In 2012, he was a Visiting Lecturer with the Group on Interactive Coding of Images, Universitat Autonoma de Barcelona. From 2018 to 2019, he was a Visiting Scholar with the School of Electrical and Information Engineering, The University of Sydney, Australia. He is currently an Associate Professor with the Department of Computer Science at The University of Warwick, U.K. His main research interests include signal and information processing and deep/machine learning with applications to multimedia data analysis, computer vision, image and video coding, security, and communications. He has authored several technical articles, book chapters, and a book in these areas. He is a member of the IEEE Inf. Forensics and Security Technical Committee, and IAPR-TC6, a Technical Committee on Computational Forensics under the auspices of the International Association for Pattern Recognition (IAPR). His research has been funded by the Consejo Nacional de Ciencia y Tecnologia, Mexico, the Natural Sciences and Engineering Research Council, Canada, the Canadian Institutes of Health Research, the FP7 and the H2020 Programs of the European Union, the Engineering and Physical Sciences Research Council, U.K., and the Defence and Security Accelerator, U.K.