International Journal of Computer Vision (2020) 128:96–120
https://doi.org/10.1007/s11263-019-01212-1

Tracking Persons-of-Interest via Unsupervised Representation Adaptation

Shun Zhang[1] · Jia-Bin Huang[2] · Jongwoo Lim[3] · Yihong Gong[4] · Jinjun Wang[4] · Narendra Ahuja[5] · Ming-Hsuan Yang[6]

Received: 13 May 2019 / Accepted: 12 August 2019 / Published online: 3 September 2019

Abstract
Multi-face tracking in unconstrained videos is a challenging problem as faces of one person often can appear drastically different in multiple shots due to significant variations in scale, pose, expression, illumination, and make-up. Existing multi-target tracking methods often use low-level features which are not sufficiently discriminative for identifying faces with such large appearance variations. In this paper, we tackle this problem by learning discriminative, video-specific face representations using convolutional neural networks (CNNs). Unlike existing CNN-based approaches which are only trained on large-scale face image datasets offline, we automatically generate a large number of training samples using the contextual constraints for a given video, and further adapt the pre-trained face CNN to the characters in the specific videos using discovered training samples. The embedding feature space is fine-tuned so that the Euclidean distance in the space corresponds to the semantic face similarity. To this end, we devise a symmetric triplet loss function which optimizes the network more effectively than the conventional triplet loss. With the learned discriminative features, we apply an EM clustering algorithm to link tracklets across multiple shots to generate the final trajectories. We extensively evaluate the proposed algorithm on two sets of TV sitcoms and YouTube music videos, analyze the contribution of each component, and demonstrate significant performance improvement over existing techniques.

Keywords Face tracking · Transfer learning · Convolutional neural networks · Triplet loss

1 Introduction

Multi-target tracking (MTT) aims at locating all targets of interest (e.g., faces, pedestrians, players, and cars) and inferring their trajectories in a video over time while maintaining their identities. This problem is at the core of numerous computer vision applications such as video surveillance, robotics, and sports analysis. Multi-face tracking is one important

Communicated by Chen Change Loy.

✉ Ming-Hsuan Yang
mhyang@ucmerced.edu

Shun Zhang
szhang@nwpu.edu.cn

Jia-Bin Huang
jbhuang@vt.edu

Jongwoo Lim
jlim@hanyang.ac.kr

Yihong Gong
ygong@mail.xjtu.edu.cn

Jinjun Wang
jinjun@mail.xjtu.edu.cn

Narendra Ahuja
n-ahuja@illinois.edu

[1] School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710072, People's Republic of China

[2] Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24060, USA

[3] Department of Computer Science, Hanyang University, Seoul 133-791, Korea

[4] Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an 710049, People's Republic of China

[5] Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA

[6] School of Engineering, University of California at Merced, Merced, CA 95344, USA

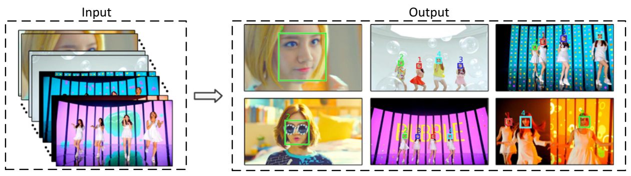


Fig. 1 Multi-face tracking. We tackle the problem of tracking multiple faces of people while maintaining their identities in *unconstrained* videos. Such videos consist of many shots from different cameras. The main challenge is to address large appearance variations of faces from different shots due to changes in pose, view angle, scale, makeup, illumination, camera motion and heavy occlusions

domain of MTT that can be applied to high-level video understanding tasks such as face recognition, content-based retrieval, and interaction analysis.

The problem of multi-face tracking is particularly challenging in unconstrained scenarios where the videos are generated from multiple moving cameras with different views or scenes as shown in Fig. 1. Examples include automatic character tracking in movies, TV sitcoms, or music videos. It has attracted increasing attention in recent years due to the fast-growing popularity of such videos on the Internet. Unlike tracking in the *constrained* counterparts (e.g., a video from a single stationary or moving camera) where the main challenge is to deal with occlusions and intersections, multi-face tracking in *unconstrained* videos needs to address the following issues: (1) A video often consists of many shots and the contents of two neighboring shots may be dramatically different; (2) It entails dealing with re-identifying faces with large appearance variations due to changes in scale, pose, expression, illumination, and makeup in different shots or scenes; and (3) The results of face detection may be unreliable due to low resolution, occlusion, nonrigid deformation, motion blurring and complex backgrounds.

Multi-target tracking has been extensively studied in the literature with the primal focus on humans. Recent approaches often address multi-face tracking by tracking-by-detection techniques. These methods first apply an object detector to locate faces in every frame, and apply data association approaches (Brendel et al. 2011; Collins 2012; Yang and Nevatia 2012a; Zhang et al. 2008; Zhao et al. 2012) that use visual cues (e.g., appearance, position, motion, and size) in an affinity model to link detections or tracklets (track fragments) into trajectories. Such methods are effective when the targets are continuously detected and when the camera is either stationary or slowly moving. However, for unconstrained videos with many shot changes and intermittent appearance of targets, the data association problem becomes more difficult because the assumptions such as appearance and size consistency, and continuous motion no longer hold in neighboring shots. Therefore, the design of discriminative features plays a critical role in identifying faces *across* shots in unconstrained scenarios.

Existing MTT methods (Yang and Nevatia 2012a; Zhang et al. 2008; Zhao et al. 2012) use combinations of low-level features such as color histograms, Haar-like features, or HOG (Dalal and Triggs 2005) to construct an appearance model for each target. However, these hand-crafted features often are not sufficiently discriminative to identify faces with large appearance changes. For example, low-level features extracted from faces of two different persons under the same pose (e.g., frontal poses) are likely more similar than those extracted from faces of the same person under different poses (e.g., frontal and profile poses).

Deep convolutional neural networks (CNNs) have demonstrated significant performance improvements on recognition tasks, e.g., image classification (Krizhevsky et al. 2012). The features extracted from the activation of a pre-trained CNN have been shown to be effective for generic visual recognition tasks (Donahue et al. 2014). In particular, CNN-based features have shown impressive performance on face recognition and verification tasks (Sun et al. 2014b, a; Schroff et al. 2015; Hu et al. 2014). These models are often trained using large-scale face recognition datasets in a fully supervised manner and then serve as feature extractors for unseen face images. However, these models may not achieve good performance in unconstrained videos as the visual domains of the training and testing sets may be significantly different.

In this paper, we address this domain shift by adapting a pre-trained CNN to the *specific* videos. Due to the lack of manual annotations of target identities, we collect a large number of training samples of faces by exploiting contextual constraints of tracklets in the video. With these automatically discovered training samples, we adapt the pre-trained CNN so that the Euclidean distance between the embedded features reflects the semantic distance between face images. Using the learned discriminative features, we apply an Expectation-Maximization (EM) clustering algorithm that links the tracklets across multiple shots into the face trajectories according to the people's identities. We analyze the contribution of each component in the proposed algorithm and demonstrate the effectiveness of the learned features to identify characters in 10 long TV sitcom episodes and singers in 8 challenging music videos. We further apply our adaptive feature learning approach to other objects (e.g., pedestrians) and show competitive performance on pedestrian tracking across cameras.

We make the following contributions in this work:

– We present an end-to-end person tracking system for unconstrained videos with large appearance variations. In contrast to prior work that requires manual clean-up in building the training set (e.g., removing false positives), our system takes raw videos as the input and performs detection, tracking, feature adaptation, and clustering in a fully automatic way.

- We propose a symmetric triplet loss function which simultaneously pulls positive pairs closer and pushes away the negative samples from the positive pairs in the triplets. Our results show that the proposed loss function significantly improves the performance of learned networks.
- We leverage contextual constraints to mine a large number of informative training samples from the given video for learning discriminative features.
- We develop a new dataset with 8 music videos from YouTube containing annotations of 3845 face tracklets and 117,598 face detections. This benchmark dataset is challenging (with frequent shot changes, large appearance variations, and rapid camera motion) and crucial for evaluating multi-face tracking algorithms in unconstrained environments.
- We demonstrate the proposed adaptive feature learning approach can be extended to tracking other objects, and present empirical results on pedestrian tracking across cameras using the DukeMTMC dataset (Ristani et al. 2016).

2 Related Work and Problem Context

2.1 Multi-target Tracking

In recent years, numerous multi-target tracking methods have been proposed by applying a pre-learned object detector to locate instances in every frame, and determine the trajectories by solving a *data association* problem (Brendel et al. 2011; Collins 2012; Pellegrini et al. 2009; Yang and Nevatia 2012a; Zhang et al. 2008; Zhao et al. 2012). A plethora of global optimization methods have been developed for data association based on the Viterbi decoding scheme (Andriluka et al. 2008), Hungarian algorithm (Stauffer 2003; Perera et al. 2006; Kaucic et al. 2005), quadratic Boolean programming (Leibe et al. 2007), maximum weight-independent sets (Brendel et al. 2011), linear programming (Jiang et al. 2007; Berclaz et al. 2011), energy minimization (Andriyenko and Schindler 2011; Andriyenko et al. 2012), and min-cost network flow (Zhang et al. 2008). Some methods tackle the data association problem with a hierarchical association framework. For example, Xing et al. (2009) propose a two-stage association method to combine local and global tracklets association to track multiple targets. Huang et al. (2008) propose a three-level association approach by first linking the detections from consecutive frames into short tracklets at the bottom level and then applying iterative Hungarian algorithm and an EM algorithm at higher levels. Yang and Nevatia (2012b) extend the three-level work in Huang et al. (2008) through learning an online discriminative appearance model.

Data association can also be formulated as a linear assignment problem. Existing algorithms typically integrate appearance and motion cues into an affinity model to infer and link detections (or tracklets) into trajectories (Brendel et al. 2011; Zhang et al. 2008; Andriyenko et al. 2012; Huang et al. 2006; Li et al. 2007). However, these MTT methods do not perform well in unconstrained videos where abrupt changes across different shots occur, and the assumptions of smooth appearance change no longer hold.

To identify targets across shots, discriminative appearance features are required to discern targets in various circumstances. Most existing multi-target tracking methods (Zhang et al. 2008; Ben Shitrit et al. 2011; Huang et al. 2008; Li et al. 2009; Wu et al. 2013a) use color histograms as features and Bhattacharyya distance or correlation coefficient as affinity measures. Several methods (Andriyenko et al. 2012; Andriyenko and Schindler 2011; Roth et al. 2012; Wang et al. 2014; Kuo and Nevatia 2011) use hand-crafted features, e.g., Haar-like (Viola and Jones 2001), SIFT (Fulkerson et al. 2008; Lowe 2004), HOG (Dalal and Triggs 2005) features, or combination (Roth et al. 2012; Wang et al. 2014; Kuo and Nevatia 2011). For robustness, some approaches (Zhang et al. 2015; Kuo et al. 2010; Yang and Nevatia 2012b, a) adaptively select the most discriminative features for a specific video (Breitenstein et al. 2009; Collins et al. 2005; Grabner and Bischof 2006). However, all these hand-crafted feature representations are not tailored for faces, and thus are less effective at handling the large appearance variations in unconstrained scenarios.

2.2 Unsupervised Domain Adaptation

The unsupervised domain adaptation task (Long et al. 2013; Fernando et al. 2015; Ganin and Lempitsky 2014; Tzeng et al. 2014) in our work is to transfer a face classifier from a source domain (e.g., offline face images) to a target domain (e.g., video) where there is no labeled training data for target video.

Several unsupervised domain adaptation methods use feature space alignment to minimize the distance between domains in the feature space by learning a transformation from source to target domains (Fernando et al. 2013, 2015; Saenko et al. 2010) or a joint adaptation layer that embeds features into a new domain-invariant space (Long et al. 2013; Tzeng et al. 2014). Tzeng et al. (2014) use two CNNs for the source and target domains with shared weights and train the network with the classification loss in the source domain and the maximum mean discrepancy (MMD) metric. Ganin et al. (2016) embed domain adaptation into learning representation. The adaptation is achieved by aligning the distributions of features across two domains with standard back-propagation training. Gupta et al. (2016) consider a similar network architecture for cross-modality supervision transfer, and Sun and Saenko (2016) perform end-to-end adaptation in deep neural networks.

There also exist numerous algorithms on unsupervised domain adaptation and transfer with adversarial learning (Goodfellow et al. 2014), where the domain difference is measured by a discriminator network (Liu and Tuzel 2016; Taigman et al. 2016). For example, Taigman et al. (2016) consider cross-domain transfer of images from one style to another without instance-level correspondence between domains using adversarial loss. The coupled GAN (Liu and Tuzel 2016) scheme constructs individual networks for each domain with partially shared higher-layer parameters for generator and discriminator to generate coherent images of two domains. Tzeng et al. (2017) combine discriminative modeling, untied weight sharing and a GAN loss in a unified framework for adversarial domain adaptation. Recently, Shu et al. (2018) propose to refine a classifier by untethering the model from the source training signal and apply approximated natural gradients to further minimize the cluster assumption violation. In contrast, we leverage contextual constraints between tracklets in a target video to mine a large number of informative training samples for learning discriminative features.

2.3 Visual Constraints in Multi-target Tracking

Several approaches (Cinbis et al. 2011; Tapaswi et al. 2014; Wang et al. 2014; Wu et al. 2013a, b) exploit visual constraints from videos for improving tracking performance. These visual constraints are often derived from the spatio-temporal relationships among the extracted tracklets (Cinbis et al. 2011; Wu et al. 2013b). Two types of constraints are commonly used: (1) all samples in the same tracklet represent the same object and (2) a pair of tracklets in the same frame indicates that two different objects are present. Prior work either uses these constraints implicitly for learning a cast-specific metric (Cinbis et al. 2011; Wang et al. 2014); or explicitly for linking cluster or tracklet (Wu et al. 2013a, b).

Numerous cues from contextual constraints have also been used for tracking, e.g., clothing (El Khoury et al. 2010), poselets (Zhang et al. 2015), script (Bauml et al. 2013; Sivic et al. 2009; Everingham et al. 2006), speech (Paul et al. 2014), gender (Zhou et al. 2015), video editing style (Tapaswi et al. 2014), clustering prior (Tang et al. 2015), and dynamic clustering constraints (Zhang et al. 2016). For examples, the methods in Ramanan et al. (2007), Tapaswi et al. (2012) and Anguelov et al. (2007) incorporate clothing appearance for improving person identification accuracy. Joon Oh et al. (2015) require ground-truth head annotations to estimate several contextual regions (e.g. head, upper body and full body). Each region is fed into one or more CNNs to obtain a set of feature vectors. Zhang et al. (2015) match the person predictions coming from poselets to the ground truths to compute part activations. Lin et al. (2010) present a probabilistic context model to jointly tag people across multiple domains of people, events, and locations. The work (Everingham et al. 2006; Anguelov et al. 2007) exploits speaker analysis to improve face labeling.

The contextual constraints in our work differ from the recent literature in the following three aspects: (1) unlike existing methods (Anguelov et al. 2007; El Khoury et al. 2010; Tapaswi et al. 2012; Du and Chellappa 2016) that exploit contextual constraints by augmenting face features with contextual features, our algorithm learns more discriminative face features using the symmetric triplet loss; (2) our work further propagates the relationship of contextual constraints to discover a larger set of positive and negative pairs for training the triplet network effectively; and (3) in contrast to existing approaches (El Khoury et al. 2010; Joon Oh et al. 2015; Zhang et al. 2015) that require ground-truth head annotations to generate contextual regions, our algorithm does not need any additional manual annotations.

2.4 CNN-Based Representation Learning

Recent face recognition and verification methods (Rao et al. 2017, 2019) focus on learning identity-preserving feature representations from deep neural networks. While the models may differ, these CNN-based face representations [e.g., DeepID (Sun et al. 2014b), DeepFace (Taigman et al. 2014), FaceNet (Schroff et al. 2015), VGG-Face (Parkhi et al. 2015)] are learned by training CNNs using large-scale datasets in a fully supervised manner. These CNNs then operate as feature extractors for face recognition, identification, and face clustering. In this work, we also use a CNN to learn identity-preserving features from a face recognition dataset. The main difference lies in that we further adapt the pre-trained representation to a specific video, thereby further improve the specificity of the model and enhance discriminative strength. In addition, we introduce a symmetric triplet-based loss function and demonstrate its effectiveness over the commonly used contrastive loss and triplet loss.

2.5 Adapting Pre-trained CNN Features to Specific Videos

There exist several methods to fine-tune a pre-trained CNN to learn video-specific features on vision tasks, such as single object tracking (Bertinetto et al. 2016) and video object segmentation (Caelles et al. 2017; Yoon et al. 2017). However, most of these methods require manual supervision to generate the training samples. Bertinetto et al. (2016) propose to adapt the parameters of a pre-trained deep model given an exemplar of the object in the first frame. Caelles et al. (2017) segment a particular entity in a video by fine-tuning the pre-trained network given the image/ground-truth segmentation mask pair

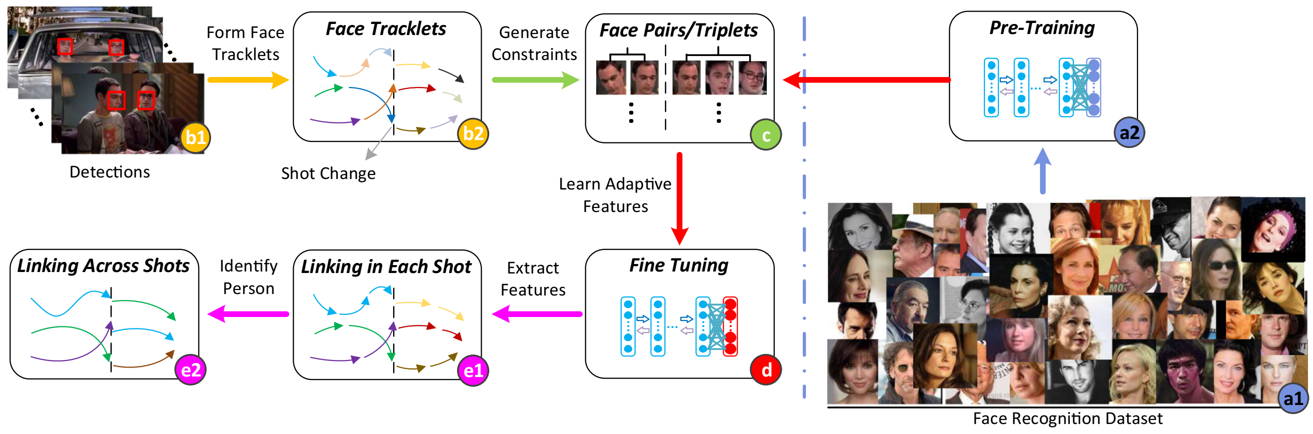


Fig. 2 Algorithm pipeline. Our multi-face tracking algorithm has four main steps: **a** Pre-training a CNN on a large-scale face recognition dataset to learn identity-preserving features; **b** Preprocessing the target video with shot detecting and face detecting; **c** Generating face pairs or face triplets from the tracklets in a specific video with the proposed spatio-temporal constraints and contextual constraints, **d** Adapting the pre-trained CNN to learn video-specific features from the automatically generated training samples, and **e** linking tracklets within each shot and then across shots to form the face trajectories

in the first frame. In contrast, our approach fine-tunes the pre-trained CNN with training samples automatically generated from spatio-temporal and contextual constraints.

2.6 Long-Term Object Tracking

The goal of long-term object tracking (Kalal et al. 2012; Pernici 2012) is to locate a specific target over time even when the target leaves and re-enters the scene. These trackers perform well on various types of targets such as cars and faces. However, online trackers are designed to handle scenes recorded by a stationary or slow-moving camera and thus not effective in tracking faces in unconstrained videos for two reasons. First, these trackers are prone to drift due to online model update with noisy examples. Second, hand-crafted features are not sufficiently discriminative to re-identify faces across shots. We tackle the first issue by processing the video offline, i.e., apply a face detector in every frame and associate all the tracklets in the video. For the second issue, we learn adaptive discriminative representation to account for large appearance variations of faces across shots or scenes.

3 Algorithmic Overview

Our goal is to track multiple faces across multiple shots in an unconstrained video while maintaining identities of the persons of interest. To achieve this, we learn discriminative features that are adapted to the appearance variations in the specific videos. We then determine the identities of the tracklets through clustering them in the learned feature space and produce the final trajectories. We summarize the main steps of the proposed algorithm in Fig. 2.

(a) *Pre-training* The CNN model based on the AlexNet (Krizhevsky et al. 2012) is pre-trained using an external face dataset to learn identity-preserving features (Sect. 4.1).
(b) *Input preprocessing* The input video is divided into non-overlapping shots using an off-the-shelf shot change detector. Within each shot, we apply a face detector and link adjacent detections into short tracklets.
(c) *Discovering the training samples* We discover a large collection of training samples (in pairs or triplets) from the tracklets based on the spatio-temporal and contextual constraints (Sect. 4.2).
(d) *Learning video-specific feature space* We adapt the pre-trained CNN model using the automatically discovered training samples to account for large appearance changes of faces pertaining to a specific video (Sect. 4.3). We present an improved symmetric triplet loss which enhances the discriminative ability of the learned features.
(e) *Linking tracklets* Within each shot, the tracklets are linked first by a conventional multi-face tracking method into short trajectories. Then, we use an EM algorithm to cluster all trajectories in the video based on the leaned features. Finally, we assign the tracklets in each cluster to the same identity (Sect. 5).

4 Learning Discriminative Features

In this section, we present the algorithmic details for learning video-specific features. After describing how the generic face features are obtained from the pre-training step, we introduce the process of discovering training examples and learning discriminative features using the proposed symmetric triplet loss function.

4.1 Supervised Pre-training

We learn identity-preserving features by pre-training a deep neural network on a large-scale face recognition dataset. Based on the AlexNet architecture (Krizhevsky et al. 2012), we replace the output layer with K nodes where each node corresponds to a specific person. We train the network on an external CASIA-WebFace dataset (Yi et al. 2014) (494,414 images of 10,575 subjects) for face recognition in a fully supervised manner. We select 9427 persons, 80% of the images (431,300 images) for training and the remaining 20% (47,140 images) as the validation set. Each face image is normalized to $227 \times 227 \times 3$ pixels. We use stochastic gradient descent with an initial learning rate of 0.01 that decreases by a factor of 10 for every 20,000 iterations using the Caffe (Jia et al. 2014) toolbox.

4.2 Discovering Training Samples

4.2.1 Shot Detection and Tracklets Linking

We first use a shot change detection method to divide each input video into non-overlapping shots.[1] Next, we use a face detector (Mathias et al. 2014) to locate faces in each frame. Given the face detections for each frame, we use a two-threshold strategy (Huang et al. 2008) to generate tracklets within each shot by linking the detected faces in adjacent frames based on similarities in appearances, positions, and scales. Note that the two-threshold strategy for linking detections could be replaced by more sophisticated methods, e.g., tracking using particle filters (Huang et al. 2006; Breitenstein et al. 2009). We discard tracklets shorter than five frames. The extracted face tracklets are formed in a conservative manner with limited temporal spans up to the length of each shot.

4.2.2 Spatio-Temporal Constraints

Existing methods typically exploit spatio-temporal constraints from tracklets to generate training samples from the video. Given a set of tracklets, we can discover a large collection of positive and negative training sample pairs belonging to the same/different persons: (1) all pairs of faces in one tracklet are from one person and (2) two face tracklets that appear in the same frame contain faces of different persons.

Let $\mathbf{T}^i = \{\mathbf{x}_1^i, \ldots, \mathbf{x}_{n_i}^i\}$ denote the ith face tracklet of length n_i. We generate a set of positive pairs $\mathbf{P}^+$ by collecting all within-tracklet face pairs:

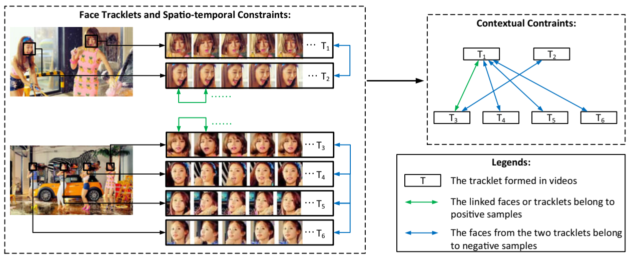

Face Tracklets and Spatio-temporal Constraints:
··· T1
··· T2
··· T3
··· T4
··· T5
··· T6
Contextual Contraints:
T1
T2
T3
T4
T5
T6
Legends:
T
The tracklet formed in videos
The linked faces or tracklets belong to positive samples
The faces from the two tracklets belong to negative samples

Fig. 3 Contextual constraints generation. Here, we label the faces in $\mathbf{T}_1$ and $\mathbf{T}_3$ as the same identity given the sufficiently high similarity between the contextual features of $\mathbf{T}_1$ and $\mathbf{T}_3$. With this additional constraint, we can propagate the constraints transitively and derive that the faces from $\mathbf{T}_1$ and $\mathbf{T}_4$ (or $\mathbf{T}_5$, $\mathbf{T}_6$) are in fact belonging to different identities, and the faces from $\mathbf{T}_3$ and $\mathbf{T}_2$ are from different people

$$\mathbf{P}^+ = \{(\mathbf{x}_k^i, \mathbf{x}_l^i)\}, \text{ s.t. } \forall k, l = 1, \ldots, n_i, \; k \neq l. \tag{1}$$

Similarly, if tracklets $\mathbf{T}^i$ and $\mathbf{T}^j$ overlap in some frames, we can generate a set of negative pairs $\mathbf{N}^-$ by collecting all between-tracklet face pairs:

$$\mathbf{N}^- = \{(\mathbf{x}_k^i, \mathbf{x}_l^j)\}, \text{ s.t. } \forall k = 1, \ldots, n_i, \; \forall l = 1, \ldots, n_j. \tag{2}$$

4.2.3 Contextual Constraints

With the spatio-temporal constraints, we can obtain a large number of face pairs without manual labeling. These training pairs, however, may have some biases. First, the positive (within-tracklet) pairs occur close in time (e.g., only several frames apart in one shot), which means that the positive face pairs often do not have large appearance variations. Second, the negative pairs are all generated from tracklets that co-occur in the *same* shot. Consequently, the learned model may not be effective in distinguishing or linking faces *across* shots (as we do not have training samples for these cases) (Fig. 3).

To address these problems, we mine additional positive and negative face pairs for learning our video-specific features. The idea is to exploit contextual information beyond facial regions for identifying persons across shots. Specifically, we identify the clothing region following Du and Chellappa (2016) and extract features using the AlexNet model. Given the ith face detection $\mathbf{x}_i$ in one frame, we locate the torso region s_i by using a probabilistic mask $I(p \in vs._i|\mathbf{x}_i)$ (See Fig. 4), where p is a pixel in the current frame. We learn this probabilistic mask from the statistics of body part's spatial relationship on the Human in 3D dataset (Bourdev and Malik 2009). We do not include samples for extracting contextual constraints from video frames that do not capture one's body part, e.g., the camera focuses on one's face.

To generate more positive training samples, we try to find the tracklet groups that are highly likely from same per-

[1] http://sourceforge.net/projects/shot-change/.

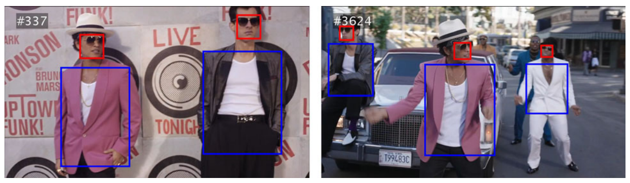


Fig. 4 Face detections and clothing regions. Although the two images belong to different shots, we can determine the face detection results are from the same persons by measuring similarity of face and clothing. Thus, they can be used as positive training samples

sons. By using the concatenated face and clothing features, we apply a Hierarchical Agglomerative Clustering (HAC) algorithm with a stopping criterion to gather the tracklets into clusters. In HAC, the distance between two clusters $\mathbf{T}^a = \{\mathbf{x}_i^a\}$ and $\mathbf{T}^b = \{\mathbf{x}_j^b\}$ is defined as the average element-wise distance, i.e.,

$$D^{ab} = \frac{1}{|\mathbf{T}^a|}\frac{1}{|\mathbf{T}^b|}\sum_i\sum_j \|\mathbf{f}(\mathbf{x}_i^a) - \mathbf{f}(\mathbf{x}_k^b)\|_2^2\,, \tag{3}$$

where $\mathbf{f}(\cdot)$ denotes the feature vector in the embedded feature space. For the tracklets which have overlapped frames, their distances are set as infinity. We also set the stopping threshold with a low value ($\theta = 0.8$ in our experiments) so that the HAC algorithm only finds confident tracklet clusters. These grouped tracklets generally contain faces with similar clothing in the different shots or scenes and thus provide additional positive tracklet pairs. In addition, by leveraging these positive pairs, we can discover more negative pairs by *transitively* propagating the relationship among tracklets. For example, suppose we know that the tracklets A and B are from different persons, and the tracklets A and C are from the same person, automatically implies that the tracklet B and C are from different persons.

Figure 3 illustrates the generation process of contextual constraints. Here, the tracklets $\mathbf{T}_1$ and $\mathbf{T}_2$ co-occur in one shot and the tracklets $\mathbf{T}_3$, $\mathbf{T}_4$, $\mathbf{T}_5$ and $\mathbf{T}_6$ co-occur in another shot. Using only spatio-temporal constraints, we are not able to obtain training samples from different shots. As a result, the tracklets $\mathbf{T}_1$ and $\mathbf{T}_4$ may be incorrectly identified as the same person. However, from the contextual cues, we may be able to identify that the tracklets $\mathbf{T}_1$ and $\mathbf{T}_3$ are the same person. Using this additional positive constraints, we can automatically generate additional negative constraints, e.g., $\mathbf{T}_1$ is a different person from $\mathbf{T}_4$, $\mathbf{T}_5$ and $\mathbf{T}_6$.

4.3 Learning Adaptive Discriminative Features

With the discovered training pairs from applying both contextual and spatio-temporal constraints, we optimize the embedding function $\mathbf{f}(\cdot)$ such that the distance $D(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2))$ in

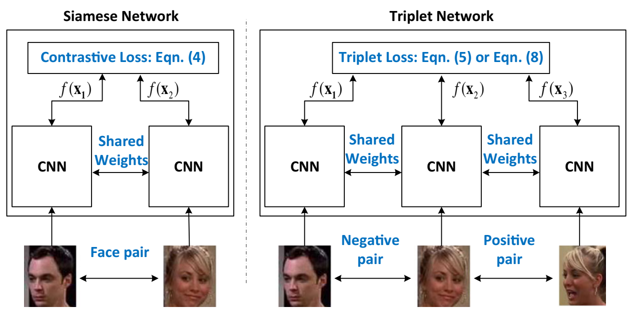


Fig. 5 Siamese versus Triplet network. Illustration of the Siamese network (left) with pairs as inputs and the triplet network (right) with triplets as inputs for learning discriminative features adaptively. The Siamese network consists of two CNNs and uses a contrastive loss. The Triplet network consists of three CNNs and uses a triplet loss. The CNNs in each network share the same architectures and parameters and are initialized with parameters of the CNN pre-trained on the large-scale face recognition dataset

the embedding space reflects the semantic similarity of two face images $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$D(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2)) = \|\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)\|_2^2. \tag{4}$$

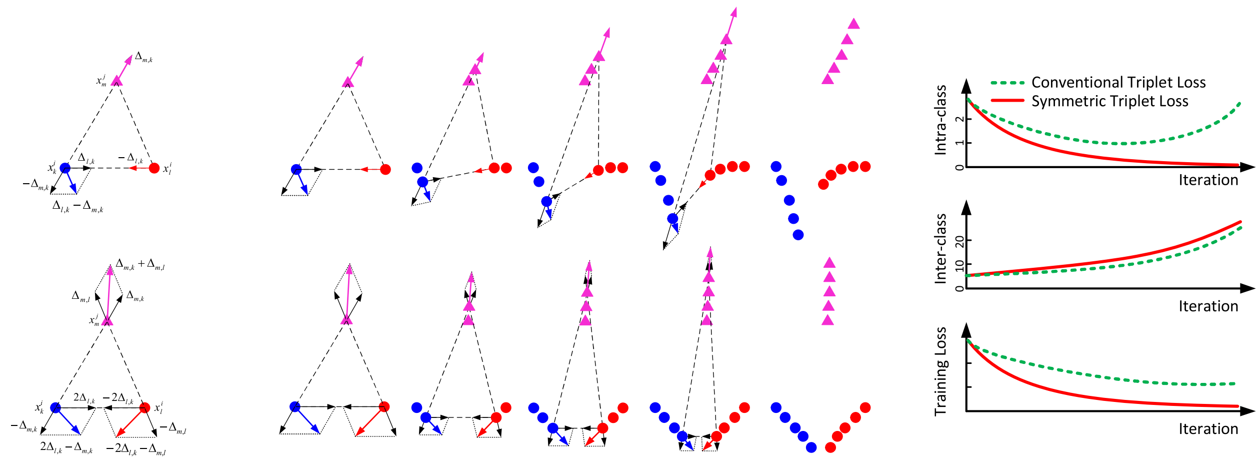
We set the feature dimension of $\mathbf{f}(\cdot)$ as 64 in all of our experiments. We first describe two commonly used loss functions for optimizing the embedding space: (1) contrastive loss and (2) triplet loss, and then present a symmetric triplet loss function for feature learning.

4.3.1 Contrastive Loss

The Siamese network (Chopra et al. 2005; Hadsell et al. 2006) consists of two identical CNNs with the shared architecture and parameters as shown in Fig. 5. Minimizing the contrastive loss function encourages small distance of two images of the same person and large distance otherwise. Denote $(\mathbf{x}_1, \mathbf{x}_2) \in \{\mathbf{P}^+, \mathbf{N}^-\}$ as a pair of training images generated with the spatio-temporal constraints. Similar to Chopra et al. (2005) and Hadsell et al. (2006), the contrastive loss function is:

$$L_p = \begin{cases} \frac{1}{2}D(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2)) & \text{if } (\mathbf{x}_1, \mathbf{x}_2) \in \mathbf{P}^+ \\ \frac{1}{2}\max(0, \tau - D(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2))) & \text{if } (\mathbf{x}_1, \mathbf{x}_2) \in \mathbf{N}^- \end{cases} \tag{5}$$

where τ ($\tau = 1$ in all our experiments) is the margin. Intuitively, if $\mathbf{x}_1$ and $\mathbf{x}_2$ are from the same person, the loss is $\frac{1}{2}D(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2))$ and we aim to decrease $D(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2))$. Otherwise, we increase $D(\mathbf{f}(\mathbf{x}_1), \mathbf{f}(\mathbf{x}_2))$ until it is larger than the margin τ.



(a) Negative partial gradient direction **(b)** Motion trajectory of one triplet **(c)** Training process

Fig. 6 Conventional triplet loss versus SymTriplet loss. **a** Illustration of the negative partial gradient direction to the triplet samples. The top row is the conventional triplet loss, and the bottom is the SymTriplet loss. We highlight the triplet samples $\mathbf{x}_k^i$ (blue), $\mathbf{x}_l^i$ (red), and $\mathbf{x}_m^j$ (magenta). The circles denote faces from the same person whereas the triangle denotes a different person. The gradient directions are color-coded. **b** The corresponding motion trajectories driven by the gradient flow of the conventional triplet loss and the triplet loss; **c** Illustration of the intra-class distances, inter-class distances, and training loss of the two triplet formulations with respect to the number of iterations (Color figure online)

4.3.2 Triplet Loss

The triplet-based network (Schroff et al. 2015) consists of three identical CNNs with the shared architecture and parameters as shown in Fig. 5. One triplet consists of two face images of the same person and one face image from another person. We generate a set of triplets $\mathbf{S}$ from two tracklets $\mathbf{T}^i$ and $\mathbf{T}^j$ belonging to different persons: $\mathbf{S} = \{(\mathbf{x}_k^i, \mathbf{x}_l^i, \mathbf{x}_m^j)\}$, s.t. $\forall k, l = 1, \ldots, n_i,\ k \neq l,\ \forall m = 1, \ldots, n_j$. Here we aim to ensure that the embedded distance of the positive pair $(\mathbf{x}_k^i, \mathbf{x}_l^i)$ is closer than that of the negative pair $(\mathbf{x}_k^i, \mathbf{x}_m^j)$ by a distance margin α ($\alpha = 1$). For one triplet, the triplet loss is of the form:

$$L_t = \frac{1}{2} \max\left(0, D(\mathbf{f}(\mathbf{x}_k^i), \mathbf{f}(\mathbf{x}_l^i)) - D(\mathbf{f}(\mathbf{x}_k^i), \mathbf{f}(\mathbf{x}_m^j)) + \alpha\right). \tag{6}$$

4.3.3 Symmetric Triplet Loss

The conventional triplet loss in (6), takes only two of the three distances into consideration: $D(\mathbf{f}(\mathbf{x}_k^i), \mathbf{f}(\mathbf{x}_l^i))$ and $D(\mathbf{f}(\mathbf{x}_k^i), \mathbf{f}(\mathbf{x}_m^j))$ although there are three distances in each triplet. We illustrate the problem of the conventional triplet loss by analyzing the gradients of the loss function. We denote the difference vector between the triplet ($\mathbf{x}_k^i$, $\mathbf{x}_l^i$ and $\mathbf{x}_m^j$):

$$\Delta_{l,k} = \mathbf{f}(\mathbf{x}_l^i) - \mathbf{f}(\mathbf{x}_k^i),\ \Delta_{m,k} = \mathbf{f}(\mathbf{x}_m^j) - \mathbf{f}(\mathbf{x}_k^i),$$
$$\Delta_{m,l} = \mathbf{f}(\mathbf{x}_m^j) - \mathbf{f}(\mathbf{x}_l^i). \tag{7}$$

For non-zero triplet loss in (6), we can compute the gradients as

$$\frac{\partial L_t}{\partial \mathbf{f}(\mathbf{x}_k^i)} = -(\Delta_{l,k} - \Delta_{m,k}),\ \frac{\partial L_t}{\partial \mathbf{f}(\mathbf{x}_l^i)} = \Delta_{l,k},$$
$$\frac{\partial L_t}{\partial \mathbf{f}(\mathbf{x}_m^j)} = -\Delta_{m,k}. \tag{8}$$

Figure 6a shows the positive and negative gradient directions for each sample. There are two issues with the triplet loss in (6).

(1) The loss function pushes the negative data point $\mathbf{x}_m^j$ away from only one of the positive pair $\mathbf{x}_k^i$ rather than both $\mathbf{x}_k^i$ and $\mathbf{x}_l^i$.
(2) The gradients on the positive pair $(\mathbf{x}_k^i, \mathbf{x}_l^i)$ are not symmetric with respect to the negative data $\mathbf{x}_k^i$.

As shown in Fig. 6b, the above two issues cause that the positive pair $\mathbf{x}_k^i$ and $\mathbf{x}_l^i$ move in *inconsistent* directions. In some cases, the training may lead to increased intra-class distance between positive pair (shown in Fig. 6c).

To address these issues, we propose a symmetric triplet loss function (SymTriplet) by considering all three distances as:

$$L_s = \max\left[0,\ D(\mathbf{f}(\mathbf{x}_k^i), \mathbf{f}(\mathbf{x}_l^i)) - \frac{1}{2}(D(\mathbf{f}(\mathbf{x}_k^i), \mathbf{f}(\mathbf{x}_m^j)) + D(\mathbf{f}(\mathbf{x}_l^i), \mathbf{f}(\mathbf{x}_m^j))) + \alpha\right], \tag{9}$$

where α is the distance margin.

The gradients induced from the proposed SymTriplet loss are

$$\frac{\partial L_s}{\partial \mathbf{f}(\mathbf{x}_k^i)} = -(2\Delta_{l,k} - \Delta_{m,k}), \ \frac{\partial L_s}{\partial \mathbf{f}(\mathbf{x}_l^i)} = 2\Delta_{l,k} + \Delta_{m,l},$$
$$\frac{\partial L_s}{\partial \mathbf{f}(\mathbf{x}_m^j)} = -(\Delta_{m,k} + \Delta_{m,l}). \tag{10}$$

The bottom row in Fig. 6a shows the negative gradient directions. The proposed SymTriplet loss directly optimizes the embedding space such that the positive pair are pulled closer to each other and the negative sample $(\mathbf{x}_m^j)$ is pulled away from the two positive samples $(\mathbf{x}_k^i, \mathbf{x}_l^i)$, as shown in Fig. 6b. This property allows us to improve the discriminative strength of the learned features.

4.4 Training Algorithm

We train the triplet network model with the SymTriplet loss function and stochastic gradient decent method with momentum. We compute the derivatives of (9) as follows:

$$\frac{\partial L_s}{\partial \mathbf{W}} = \begin{cases} \frac{\partial \widetilde{L_s}}{\partial \mathbf{W}} & L_s > 0, \\ 0 & L_s = 0, \end{cases} \tag{11}$$

where

$$\begin{aligned}\frac{\partial \widetilde{L_s}}{\partial \mathbf{W}} &= 2(\mathbf{f}(\mathbf{x}_k^i) - \mathbf{f}(\mathbf{x}_l^i))\frac{\partial \mathbf{f}(\mathbf{x}_k^i) - \partial \mathbf{f}(\mathbf{x}_l^i)}{\partial \mathbf{W}} \\ &- (\mathbf{f}(\mathbf{x}_k^i) - \mathbf{f}(\mathbf{x}_m^j))\frac{\partial \mathbf{f}(\mathbf{x}_k^i) - \partial \mathbf{f}(\mathbf{x}_m^j)}{\partial \mathbf{W}} \\ &- (\mathbf{f}(\mathbf{x}_l^i) - \mathbf{f}(\mathbf{x}_m^j))\frac{\partial \mathbf{f}(\mathbf{x}_l^i) - \partial \mathbf{f}(\mathbf{x}_m^j)}{\partial \mathbf{W}}.\end{aligned} \tag{12}$$

We can compute the gradients from each input triplet examples given the values of $\mathbf{f}(\mathbf{x}_k^i), \mathbf{f}(\mathbf{x}_l^i), \mathbf{f}(\mathbf{x}_m^j)$ and $\frac{\partial \mathbf{f}(\mathbf{x}_k^i)}{\partial \mathbf{W}}, \frac{\partial \mathbf{f}(\mathbf{x}_l^i)}{\partial \mathbf{W}}, \frac{\partial \mathbf{f}(\mathbf{x}_m^j)}{\partial \mathbf{W}}$, which can be obtained using the standard forward and backward propagations separately for each image in the triplet examples. We summarize the main training steps in Algorithm 1.

5 Multi-face Tracking via Tracklet Linking

We use a two-step procedure to link face tracklets generated in Sect. 4.2: (1) linking the face tracklets within each shot into shot-level tracklets, and (2) merging shot-level tracklets across multiple shots into trajectories.

Algorithm 1 Stochastic gradient descent with SymTriplet loss

```
1: Input: Training samples $\{(\mathbf{x}_k^i, \mathbf{x}_l^i, \mathbf{x}_m^j)\}$.
2: Output: Network parameters $\mathbf{W}$,
3: for $t = 1 \to$ Max number of iterations do $\frac{\partial L_s}{\partial \mathbf{W}} = 0$
4:     for all training triplet samples $(\mathbf{x}_k^i, \mathbf{x}_l^i, \mathbf{x}_m^j)$ do
5:         Compute $\mathbf{f}(\mathbf{x}_k^i), \mathbf{f}(\mathbf{x}_l^i)$ and $\mathbf{f}(\mathbf{x}_m^j)$ by forward propagation;
6:         Compute $\frac{\partial \mathbf{f}(\mathbf{x}_k^i)}{\partial \mathbf{W}}, \frac{\partial \mathbf{f}(\mathbf{x}_l^i)}{\partial \mathbf{W}}$ and $\frac{\partial \mathbf{f}(\mathbf{x}_m^j)}{\partial \mathbf{W}}$ by back propagation;
7:         Compute $\frac{\partial L_s}{\partial \mathbf{W}}$ according to (11) and (12).
8:     end for
9:     Update the parameters $\mathbf{W}^t \leftarrow \mathbf{W}^{t-1} - \lambda_t \frac{\partial L_s}{\partial \mathbf{W}}$
10: end for
```

5.1 Linking Tracklets Within Each Shot

We use a typical multi-object tracking framework for linking tracklets within each shot. First, we extract features from each detected face using the learned deep network. We measure the linking probabilities between two tracklets using temporal, kinematic and appearance information. Then, we use the Hungarian algorithm to determine a globally optimal label assignment (Huang et al. 2008; Xing et al. 2009) and link tracklets with the same label are linked into shot-level tracklets. We leave the detailed process of calculating linking probabilities between two tracklets in the supplementary material (https://sites.google.com/site/shunzhang876/facetracking).

5.2 EM Clustering for Linking Tracklets Across Shots

Instead of resorting to simple hierarchical agglomerative clustering algorithms, we apply an EM-based clustering algorithm which can account for non-spherical clusters and outliers. We iteratively update the Gaussian models for the clusters using the member tracklets, and determine the tracklet membership using the updated models—the expectation-maximization procedure. Since the dimensionality of the learned feature space is too large for Gaussian distribution modeling, we first embed the tracklets into a low-dimensional space for clustering. Among the low-dimensional feature embedding methods, we use t-SNE (t-distributed Stochastic Neighbor Embedding) algorithm (Van der Maaten and Hinton 2008). The outlier tracklets such as incorrect detections or the faces of background actors are likely to scatter randomly in the embedded space. As a result, they usually do not belong to the clusters formed by EM procedure or form a separate cluster. Once the EM procedure is converged, we detect the outlier tracklets and filter out by testing their Mahalanobis distance to the nearest cluster.

Unlike linear methods like PCA, t-SNE is a non-linear dimensionality reduction algorithm. It defines two probability distributions on the element similarity in the original and embedded space, and computes the embedded points which minimize the Kullback-Leibler divergence between the two distributions. The details of the t-SNE algorithm can be found in Van der Maaten and Hinton (2008). The detailed process of the EM clustering algorithm is summarized in Algorithm 2. The initial clusters are setup by running K-means, and the data membership is determined by finding the cluster minimizing the Mahalanobis distance

$$c^* = \arg\min_c \left(\sqrt{(\mathbf{x} - \mu_c)^\top \Sigma_c^{-1} (\mathbf{x} - \mu_c)} \right). \quad (13)$$

Then each cluster mean μ_c and covariance Σ_c is updated using the member data, and this procedure is iterated until convergence. During the iteration, if a cluster becomes empty, it is discarded and a new cluster is initialized at a random position. To effectively capture the outliers, we maintain one outlier cluster with a large variance in the EM procedure. Upon convergence, the tracklets far from all clusters are classified as outliers and discarded. The tracklets in the same clusters are given with the same identities and linked together to form the final trajectories.

Algorithm 2 Tracklet linking using EM clustering

```
1: procedure EM_CLUSTERING($K$, $tracklets$)    ▷ K is number of inlier clusters
2:   $C$ = K-means(t-SNE($tracklets$), $K+1$)
3:   The outlier cluster : $c_{out} = \arg\max_c(\Sigma_c)$
4:   The inlier clusters : $C_{in} = C \backslash \{c_{out}\}$
5:   while not converged do
6:     Set the outlier threshold $\theta_o = \text{avg}(D_{\Sigma_o}(\mathbf{x}_o))$, $\mathbf{x}_o \in c_{out}$
7:     for each data $\mathbf{x}$, do
8:       if $\min_{c \in C_{in}} D_{\Sigma_c}(\mathbf{x}) > \theta_o$ then
9:         Classify $\mathbf{x}$ into the outlier cluster
10:      else
11:        Classify $\mathbf{x}$ into $\arg\min_{c \in C_{in}} D_{\Sigma_c}(\mathbf{x})$
12:      end if
13:    end for
14:    if any cluster is empty then
15:      Randomly re-initialize the cluster.
16:    end if
17:  end while
18: end procedure
```

6 Experimental Results

In this section, we first describe the implementation details, datasets, and evaluation metrics. Next, we present the evaluation results of the proposed algorithm against the state-of-the-art methods. More experimental results and videos are available in the supplementary material (https://sites.google.com/site/shunzhang876/facetracking). The code of the proposed algorithm for tracking persons-of-interest (TPI) and annotated datasets are available on our project website.[2]

6.1 Implementation Details

CNN fine-tuning We adapt the pre-trained CNN with the proposed SymTriplet loss. For feature embedding, we replace the classification layer in the pre-trained network with 64 output nodes. We use stochastic gradient descent with the momentum term set to 0.9. For the network training, we set a fixed learning rate to 0.00001 for finetuning and a weight decay of 0.0001. We use a mini-batch size 128 and train the network for 2000 epochs.

6.2 Datasets

We evaluate the proposed algorithm on three types of videos containing multiple persons:

1. Videos in a laboratory setting: FRONTAL (Wu et al. 2013a)
2. TV sitcoms: The Big Bang Theory (BBT) (Wu et al. 2013a, b) and Buffy the Vampire Slayer (BUFFY) (Sivic et al. 2009; Everingham et al. 2006; Du and Chellappa 2016) datasets
3. Music video dataset from YouTube

FRONTAL *video* FRONTAL is a short video in a constrained scene acquired indoors with a fixed camera. Four persons facing the camera move around and occlude each other.
BBT dataset We select the first 7 episodes from Season 1 of the Big Bang Theory TV Sitcom (referred to as BBT01- 07). Each video is about 23 min long with the main cast of 5–13 people and is recorded mostly indoors. The main difficulty lies in identifying faces of the same person from frequent changes of camera views and scenes, where there are large appearance variations in viewing angle, pose, scale, and illumination.
BUFFY dataset The BUFFY dataset has been widely evaluated in the context of automatic face labeling (Sivic et al. 2009; Everingham et al. 2006; Du and Chellappa 2016). The dataset contains three episodes (episode 2, 5 and 6) from Season 5 of the TV series Buffy the Vampire Slayer (referred to as BUFFY02, BUFFY05, and BUFFY06). Each video is about 40 minutes long with the main cast of 13–19 people. The illumination condition in this video dataset is more challenging than that in the BBT dataset as it contains many scenes with dim light.
Music video dataset We contribute a new dataset consisting of 8 music videos from YouTube. We provide full annotations of 3845 face tracklets and 117,598 face detections. Compared

[2] https://sites.google.com/site/shunzhang876/facetracking.

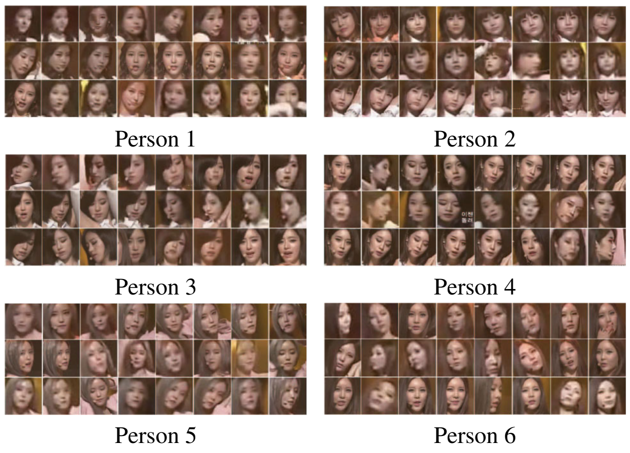


Fig. 7 Sampling ground-truth thumbnail faces of 6 people on the T-ARA sequence to illustrate the challenge of similar looking in our dataset

to existing face tracking datasets, the new dataset presents a new set of challenges (e.g., frequent shot/scene changes, large appearance variations, and rapid camera motion) that are crucial for developing multi-face tracking algorithms in unconstrained environments. It is challenging to track multiple faces in these videos due to large variations caused by frequent shot/scene changes, large appearance variations, and rapid camera motion. Three sequences (T- ARA, WESTLIFE and PUSSYCAT DOLLS) are recorded from live music performance with multiple cameras in different views. The other sequences (BRUNO MARS, APINK, HELLO BUBBLE, DARLING and GIRLS ALOUD) are MTV videos. Faces in these videos often undergo large appearance variations due to changes in pose, scale, makeup, illumination, camera motion, and occlusions.
Challenges of the presented music video dataset:

(i) The music video dataset contains faces with similar appearances. In the T- ARA, HELLOBUBBLE, DARLING, APINK and GIRLSALOUD sequences, numerous faces resemble to each other. Figure 7 shows cropped face images from the T- ARA sequence, in which all six persons have similar looks. It is difficult to distinguish person 3 and 4, and person 5 and 6 have very similar looks. As all 6 persons have similar looks, it is challenging for face recognition methods to perform well, especially under the unsupervised setting. More similar cropped images can be found in Figures 1–4 of the supplementary material (https://sites.google.com/site/shunzhang876/facetracking). Although existing ace datasets contain a large number of images in the wild, the number of faces with similar looks is small, and most of them can be easily distinguished.
(ii) Although the music video dataset contains only 52 subjects, it includes a number of face images captured from

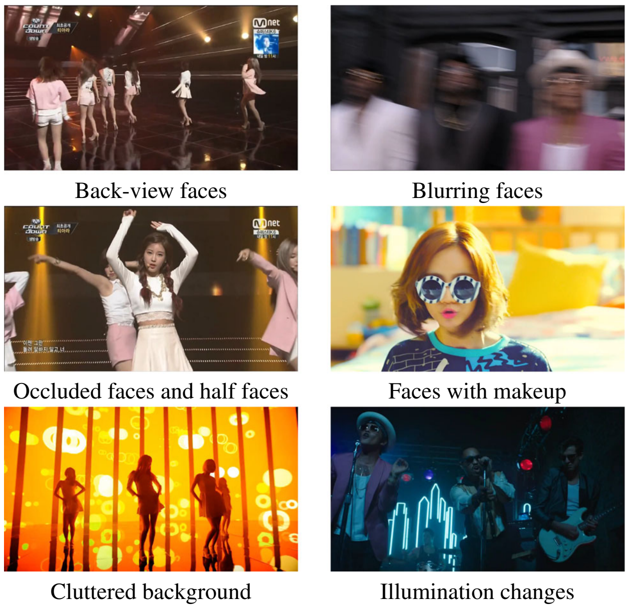


Fig. 8 The music video dataset presents a set of challenges, such as different cluttered background, large appearance changes, large viewing angle changes, motion blurring and so on

a wide range of viewpoints with different poses. In addition, these images are captured in different camera conditions (lighting, zoom, and poses). Thus, the face images contain large appearance variations in cluttered backgrounds. As shown in Fig. 8, this dataset is challenging as it contains heavy occlusion, motion blur, and low contrast.

6.3 Evaluation Metrics

We evaluate the proposed method in two main aspects. First, to evaluate the effectiveness of the learned video-specific features, we use a bottom-up hierarchical agglomerative clustering algorithm to merge pairs of tracklets until all tracklets have been merged into the pre-defined number of clusters (i.e., the actual number of people in the video). We measure the quality of clustering using the weighted purity:

$$W = \frac{1}{M} \sum_{c} m_c \cdot p_c, \tag{14}$$

where each cluster c contains m_c elements and its purity p_c is measured as the fraction of the largest number of faces from the same person to m_c, and M denotes the total number of faces in the video.

Second, we evaluate the method with the metrics commonly used in multi-target tracking (Zhang et al. 2015), including Recall, Precision, F1, FAF, IDS, Frag, MOTA, and MOTP. We list the definitions of these metrics in the supplementary material(https://sites.google.com/site/shunzhang876/facetracking). The up and down arrows indicate whether higher or lower scores are better for each metric.

6.4 Evaluation on Features

We evaluate the proposed adaptive features against several alternatives summarized in Table 1.

6.4.1 Adaptive Features Versus off-the-shelf Features

We evaluate the proposed features (TPI-Siamese, TPI-Triplet, TPI-SymTriplet and TPI-SymTriplet-Contx) adapted to a specific video against the off-the-shelf features (HOG, AlexNet, Pre-trained, VGG-Face and VGG-Face2) in Table 2 and 3. We show that the identity-preserving features (Pre-trained, VGG-Face and VGG-Face2) trained on face datasets offline achieve better performance over generic feature representation (e.g., AlexNet and HOG). Our video-specific features trained with Siamese and triplet networks achieve favorable performance than other alternatives, highlighting the importance of learning video-specific features. For example, in the Daring sequence, the proposed method with TPI-SymTriplet-Contx feature achieves the weighted purity of 0.76, significantly outperforming the off-the-shelf features, i.e., VGG-Face: 0.20, VGG-Face2: 0.32, AlexNet: 0.18 and HOG: 0.19. Overall, the results with the proposed features are more than twice as accurate as that using off-the-shelf features in music videos. For the BBT dataset, the proposed feature adaptation consistently outperforms that with off-the-shelf features.

6.4.2 Measuring the Effectiveness of Features via Clustering

Here, we validate the effectiveness the proposed features compared to the baselines. Figure 9 shows the results in terms of clustering purity versus the number of clusters on 7 BBT sequences, 3 BUFFY episodes, and 5 music videos. The pink dashed line means that all faces are correctly grouped with weighted purity $W_C = 1$. For more effective features, the weighted purity measures approach 1 at a faster rate. For each feature type, we show the weighted purity at the ideal number cluster (i.e., number of people in a video) in the legend.

Figure 10 shows 2D visualization of extracted features from T- ara using the t-SNE algorithm (Van der Maaten and Hinton 2008). The visualization illustrates the difficulty in handling large appearance variations in unconstrained videos. For HOG features, there exist no clear cluster structures, and faces of the same person are scattered around. Although AlexNet and pre-trained features increase inter-person distances, the clusters of the same person do not appear in close proximity. In contrast, the proposed adaptive features form tighter clusters for the same person and greater separation between different persons.

6.4.3 Nonlinear and Linear Metric Learning

Unlike several existing approaches (Cinbis et al. 2011; Wu et al. 2013a, b; Tapaswi et al. 2014; Xiao et al. 2014) that rely on hand-crafted features and linear metric learning, we use a deep nonlinear metric learning by finetuning all layers to learn discriminative face representations. To demonstrate the contribution of the nonlinear metric learning, we compare our adaptive features with VGG-Face-ULDML which learns Mahalanobis distance on the VGG-Face features in Table 2. We show that the proposed method achieves higher clustering purity than VGG-Face-ULDML on all videos. For example, on the T- ara sequence, the clustering purity by TPI-SymTriplet-Contx and VGG-Face-ULDML is 0.84, and 0.26, respectively.

6.4.4 SymTriplet and Conventional Siamese/Triplet Loss

We evaluate the effectiveness of the proposed SymTriplet loss (TPI-SymTriplet) with comparisons to the contrastive loss (TPI-Siamese) and the triplet loss (TPI-Triplet) on all videos in Table 2. The proposed method with the SymTriplet loss performs well against the other methods since positive sample pairs are pulled closer and negative samples are pushed away from the positive pairs. For example, on the Bbt05 sequence, TPI-SymTriplet (0.85) achieves higher clustering purity than TPI-Triplet (0.68) and TPI-Siamese (0.70);

To gain more insight of the proposed SymTriplet loss function, we visualize the top layer (i.e. the embedded layer) features obtained from theTPI-FullModel. The conventional triplet loss from the TPI-FullModel-Triplet is selected as the baseline. Figure 11 shows the feature visualizations for the two different losses respectively. It is clearly seen that, the proposed SymTriplet loss can help making the learned features with better within-cluster compactness and between-cluster separability as compared to the conventional triplet loss.

6.4.5 Contextual and Spatio-Temporal Constraints

We demonstrate the effectiveness of contextual constraints. Using the presented SymTriplet loss, we compare the features learned from using only spatio-temporal constraints (TPI-SymTriplet), and *both* contextual and spatio-temporal constraints (TPI-SymTriplet-Contx). Table 2 shows that TPI-SymTriplet-Contx achieves better performance when compared with TPI-SymTriplet on all videos. We attribute the performance improvement to the additional positive and negative face pairs discovered through contextual cues and the transitive constraint propagation.

Table 1 Summary of the evaluated features

Method name	Feature dimension	Architecture	Training loss	Clustering algorithm	Description
HOG (Dalal and Triggs 2005)	4356	–	–	HAC	A conventional hand-crafted feature
AlexNet (Krizhevsky et al. 2012)	4096	AlexNet	Softmax loss	HAC	A generic feature representation
Pre-trained	4096	AlexNet	Softmax loss	HAC	Face representation trained on the WebFace dataset
VGG-Face (Parkhi et al. 2015)	4096	16-layer VGG	Softmax loss	HAC	A publicly available face descriptor
VGG-Face-ULDML	4096	16-layer VGG	ULDML (Cinbis et al. 2011)	HAC	A Mahalanobis mapping from the VGG-Face features
VGG-Face2 (Cao et al. 2018)	2048	50-layer ResNet	Softmax loss	HAC	A publicly available face descriptor, fine-tuned on VGG-Face2 training set based on a pre-trained model on Ms-Celeb-1M dataset
TPI-Triplet	64	AlexNet	Triplet loss	HAC	Trained with traditional spatio-temporal constraints based on Pre-trained model
TPI-Siamese	64	AlexNet	Contrastive loss	HAC	Trained with traditional spatio-temporal constraints based on Pre-trained model
TPI-SymTriplet	64	AlexNet	SymTriplet loss	HAC	Trained with traditional spatio-temporal constraints based on Pre-trained model
TPI-SymTriplet-Contx	64	AlexNet	SymTriplet loss	HAC	Trained with the contextual constraints based on Pre-trained model
TPI-SymTriplet-BBT02	64	AlexNet	SymTriplet loss	HAC	Trained on the BBT02 video with spatio-temporal constraints based on Pre-trained model
TPI-FullModel	64	AlexNet	SymTriplet loss	EM clustering	Trained with the contextual constraints based on Pre-trained model

Table 2 Clustering results on 7 BBT videos and 3 BUFFY videos. The weighted purity of each video is measured on the ideal number of clusters

Methods	BBT dataset							BUFFY dataset		
	BBT01	BBT02	BBT03	BBT04	BBT05	BBT06	BBT07	BUFFY02	BUFFY05	BUFFY06
HOG (Dalal and Triggs 2005)	0.37	0.31	0.37	0.36	0.29	0.26	0.30	0.21	0.38	0.25
AlexNet (Krizhevsky et al. 2012)	0.47	0.31	0.45	0.36	0.29	0.26	0.39	0.33	0.37	0.26
Pre-trained	0.86	0.71	0.73	0.59	0.51	0.50	0.73	0.26	0.42	0.32
VGG-Face (Parkhi et al. 2015)	0.91	0.85	0.85	0.54	0.65	0.46	0.79	0.22	0.51	0.41
VGG-Face-ULDML	0.92	0.87	0.86	0.60	0.68	0.23	0.85	0.34	0.54	0.43
VGG-Face2	0.92	0.88	0.90	0.64	0.72	0.45	0.82	0.28	0.54	0.46
TPI-Siamese	*0.94*	*0.95*	0.87	0.74	0.70	0.70	0.89	0.44	0.67	0.61
TPI-Triplet	*0.94*	*0.95*	0.92	0.74	0.68	0.70	0.89	0.45	0.66	0.70
TPI-SymTriplet	*0.94*	*0.95*	0.92	0.78	0.85	0.75	0.91	0.46	0.68	0.73
TPI-SymTriplet-Contx	**0.95**	*0.95*	*0.93*	*0.84*	*0.86*	**0.83**	*0.92*	*0.58*	*0.70*	*0.75*
TPI-SymTriplet-BBT02	0.90	*0.95*	0.87	0.74	0.79	0.67	0.88	–	–	–
TPI-FullModel	**0.95**	**0.98**	**0.95**	**0.92**	**0.93**	*0.81*	**0.96**	**0.83**	**0.83**	**0.84**

Bold values indicate the best and italic values indicate the second-best performance

Table 3 Clustering results on 8 music videos. The weighted purity of each video is measured on the actual number of clusters

Methods	Music dataset							
	T- ARA	PUSSYCAT DOLLS	BRUNO MARS	HELLO BUBBLE	DARLING	APINK	WESTLIFE	GIRLS ALOUD
HOG (Dalal and Triggs 2005)	0.22	0.28	0.36	0.33	0.20	0.20	0.27	0.29
AlexNet (Krizhevsky et al. 2012)	0.24	0.32	0.35	0.31	0.19	0.21	0.37	0.30
Pre-trained	0.31	0.31	0.49	0.34	0.25	0.28	0.32	0.33
VGG-Face (Parkhi et al. 2015)	0.23	0.46	0.44	0.29	0.21	0.24	0.27	0.31
VGG-Face-ULDML	0.26	0.44	0.47	0.34	0.28	0.26	0.41	0.32
VGG-Face2	0.31	0.51	0.48	0.39	0.32	0.35	0.42	0.35
TPI-Siamese	0.69	0.77	0.88	0.54	0.46	0.48	0.54	0.67
TPI-Triplet	0.68	0.77	0.83	0.60	0.49	0.60	0.52	0.67
TPI-SymTriplet	0.69	0.78	*0.90*	0.64	0.50	0.57	0.56	0.69
TPI-SymTriplet-Contx	*0.84*	*0.83*	**0.91**	*0.69*	*0.72*	*0.74*	*0.66*	*0.75*
TPI-FullModel	**0.95**	**0.93**	**0.91**	**0.81**	**0.73**	**0.86**	**0.91**	**0.97**

Bold values indicate the best and italic values indicate the second-best performance

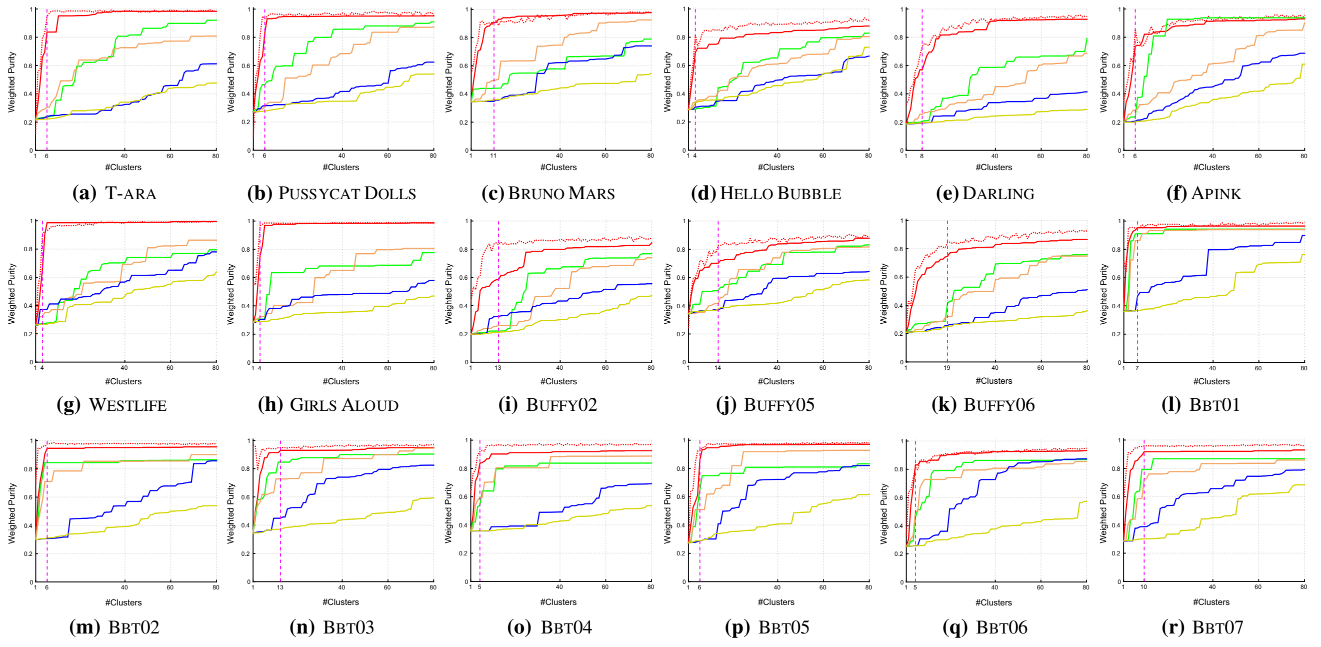
6.4.6 EM Clustering and HAC Algorithm

We evaluate the effectiveness of the presented EM clustering algorithm. We compare TPI-FullModel (using EM clustering) with TPI-SymTriplet-Contx (using HAC) on all test videos with the clustering weighted purity measured on the actual number of clusters in Tables 2 and 3. The results show that using EM clustering algorithm achieves the best clustering performance on almost all videos due to the ability to remove outliers from clusters. In Fig. 9 we also compare TPI-FullModel with TPI-SymTriplet-Contx in terms of clustering purity versus the number of clusters. The figure illustrates that the EM clustering algorithm achieves higher curve than the HAC algorithm on most of videos, e.g., DARLING, T-ARA, HELLO BUBBLE, APINK, BUFFY02 and BBT04. Using the EM clustering algorithm improves the performance by a large margin, particularly for the Music Video dataset, in which people in the video look very similar.

6.4.7 Comparisons with Other Face Clustering Algorithms

We compare our method with five recent state-of-the-art face clustering algorithms (Wu et al. 2013a, b; Cinbis et al. 2011; Xiao et al. 2014; Zhang et al. 2016) on the FRONTAL, BBT01, BUFFY02, and NOTTING HILL videos. Table 4 shows the clustering accuracy over faces and tracklets [using the same datasets and metrics as Wu et al. (2013a, b)].[3] In con-

[3] The code and data of some methods, e.g., Tapaswi et al. (2014) are not available.



(a) T-ARA **(b)** PUSSYCAT DOLLS **(c)** BRUNO MARS **(d)** HELLO BUBBLE **(e)** DARLING **(f)** APINK

(g) WESTLIFE **(h)** GIRLS ALOUD **(i)** BUFFY02 **(j)** BUFFY05 **(k)** BUFFY06 **(l)** BBT01

(m) BBT02 **(n)** BBT03 **(o)** BBT04 **(p)** BBT05 **(q)** BBT06 **(r)** BBT07

Fig. 9 Clustering performance. Clustering purity versus number of clusters with different features on YouTube music video, Big Bang Theory and BUFFY datasets. The ideal line indicates that all faces are correctly grouped into ideal clusters, and its corresponding weighted purity is equal to 1. For the more effective feature, its purity approximates to 1 faster with the increase in the number of clusters. Red dotted line: TPI-FullModel; red solid line: TPI-SymTriplet-Contx; green solid line: VGG-Face; orange solid line: Pre-trained; blue solid line: AlexNet; yellow solid line: HOG; Pink dashed line: Ideal Clusters (Color figure online)

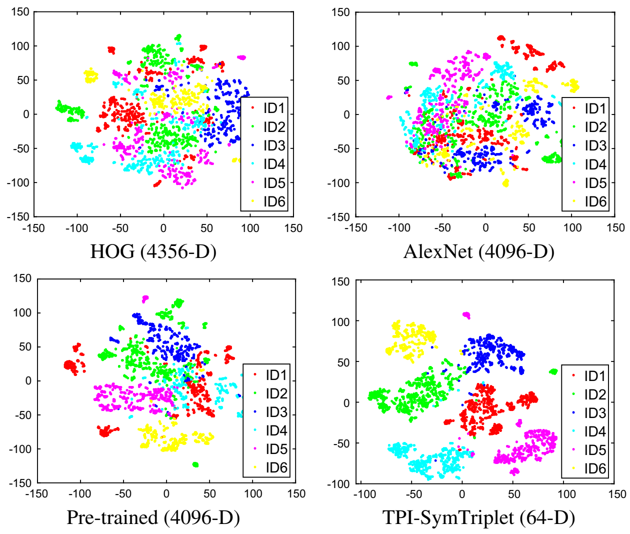
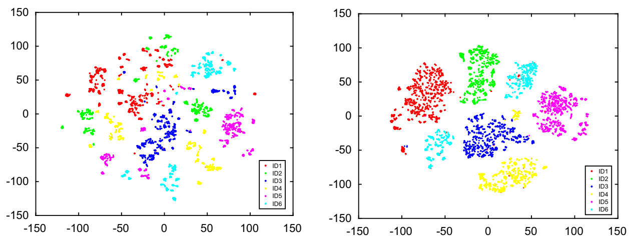


Fig. 10 2D tSNE visualization. 2D tSNE visualization of all face features from the proposed fine-tuned CNN for adapting video-specific variations, compared with HOG, AlexNet, and pre-trained features. T-ARA has 6 main casts. The faces of different people are color coded (Color figure online)



(a) The conventional triplet loss **(b)** The SymTriplet loss

Fig. 11 Feature visualization. Feature visualization by t-SNE, using **a** the conventional triplet loss; **b** the presented SymTriplet loss. APINK has 6 main casts. The faces of different people are color coded (Color figure online)

trast to the methods in Wu et al. (2013a, b), Cinbis et al. (2011) and Xiao et al. (2014) which learn linear transformations over the extracted features, our work learns nonlinear metrics by adapting all layers of the CNN and performs favorably on the FRONTAL, BBT01, and BUFFY02 sequences. Both Zhang et al. (2016) and our method discover more informative face pairs to adapt the pre-trained models to learn discriminative face representations and achieve similar clustering performance on the BUFFY02 and NOTTING HILL videos.

6.4.8 Comparison with other Unsupervised Domain Adaption Methods

We present additional comparisons to several unsupervised domain adaptation algorithms, including Subspace Alignment (SA) (Fernando et al. 2013), Correlation Alignment

Table 4 Clustering accuracy on the FRONTAL, BBT01, BUFFY02 and NOTTING HILL videos

Method	FRONTAL		BBT01		BUFFY02	NOTTING HILL
	Faces	Tracklets	Faces	Tracklets	Faces	Faces
HOG (Dalal and Triggs 2005)	0.411	0.402	0.495	0.472	0.304	0.451
AlexNet (Krizhevsky et al. 2012)	0.591	0.435	0.716	0.698	0.426	0.634
Pre-trained	0.777	0.381	*0.747*	*0.775*	0.516	0.791
Cinbis-ICCV-11 (Cinbis et al. 2011)	0.844	0.861	0.581	0.565	0.416	0.732
Wu-CVPR-13 (Wu et al. 2013b)	0.950	0.907	0.626	0.596	0.503	0.844
Wu-ICCV-13 (Wu et al. 2013a)	0.950	0.907	0.665	0.668	–	–
Xiao-ECCV-14 (Xiao et al. 2014)	*0.962*	*0.938*	0.694	0.721	0.628	0.963
Zhang-ECCV-16 (Zhang et al. 2016)	–		–	–	*0.921*	**0.990**
TPI-FullModel	**0.998**	**0.998**	**0.946**	**0.982**	**0.926**	*0.980*

We compare our results with three baseline features and five other state-of-the-art face clustering methods (Wu et al. 2013a, b; Cinbis et al. 2011; Xiao et al. 2014; Zhang et al. 2016) based on the same face tracks input and metrics as in Wu et al. (2013a, b)
Bold values indicate the best and italic values indicate the second-best performance

Table 5 Clustering results on the music dataset. The weighted purity of each video is measured on the actual number of clusters

Methods	T- ARA	PUSSYCAT DOLLS	BRUNO MARS	HELLO BUBBLE	DARLING	APINK	WESTLIFE	GIRLS ALOUD
TPI-FullModel	**0.95**	**0.93**	**0.91**	**0.81**	**0.73**	**0.86**	**0.91**	**0.97**
SA (Fernando et al. 2013)	0.37	0.46	0.52	0.43	0.43	0.38	0.44	0.40
CORAL (Sun et al. 2016)	0.41	0.48	0.53	0.47	0.38	0.47	0.45	0.45
Deep CORAL (Sun and Saenko 2016)	0.52	0.58	0.62	0.54	0.49	0.52	0.56	0.51
DIRT-T (Shu et al. 2018)	0.56	0.61	0.64	0.54	0.47	0.55	0.58	0.53

We compare our clustering results with several state-of-the-art unsupervised domain adaption methods (Fernando et al. 2013; Sun et al. 2016; Sun and Saenko 2016; Shu et al. 2018)
Bold values indicate the best performance

(CORAL) (Sun et al. 2016), Deep CORAL (Sun and Saenko 2016) and DIRT-T (Shu et al. 2018), on the music dataset for clustering performance analysis. Table 5 shows that the existing unsupervised domain adaption techniques (e.g., SA and CORAL) do not perform well on the music dataset due to large domain differences. In contrast, our method allows learning rich feature hierarchies via multiple layers of nonlinear transformations with end-to-end training of deep networks based on discovered training samples and contextual constraints in videos. Table 5 shows that our method performs favorably against the SA and CORAL methods in music videos.

6.4.9 Comparisons with Different Number of Feature Dimensions

We investigate the effect of the dimensionality of the embedded features. Figure 12 shows the clustering purity versus the number of clusters in comparison with a different number of feature dimensions on the sequence BRUNO MARS. In general, the clustering accuracy is not very sensitive to the selection of feature dimension. However, we do observe that using large feature dimension (e.g., 512 and 1024) does not perform well compared to smaller ones. We attribute this to the insufficient training samples. The evaluation on feature dimension also validates the selection of using 64-dimensional features for accuracy and efficiency.

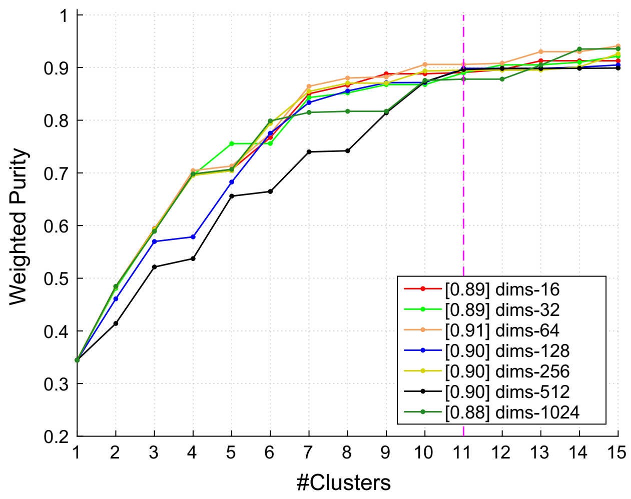


Fig. 12 Effect of feature dimensionality. The legend shows the weighted purity at the ideal number of clusters for each feature on BRUNO MARS sequence (Color figure online)

6.5 Multi-face Tracking

6.5.1 Comparisons with the State-of-the-Art Multi-target Trackers

We compare the proposed algorithm with several state-of-the-art MTT trackers including the modified version of TLD (Kalal et al. 2012), ADMM (Ayazoglu et al. 2012), IHTLS (Dicle et al. 2013), and methods by Wu et al. (2013a, b).[4] The TLD (Kalal et al. 2012) scheme is a long-term single-target tracker which can re-detect targets of interest when targets leave and re-enter a scene. We implement two extensions of TLD for multi-face tracking. The first one is the mTLD scheme where in each sequence, we run multiple TLD trackers for all targets and each TLD tracker is initialized with the ground-truth bounding box in the first frame. For the second extension of TLD, we integrate the mTLD into our framework (referred to as TPI-mTLD). We use the mTLD to generate shot-level trajectories within each shot instead of using the two-threshold and Hungarian algorithms. At the beginning of each shot, we initialize TLD trackers with untracked detections and link the detections in the following frames according to the overlap scores with TLD outputs.

Table 6 shows quantitative results of the proposed algorithm, the mTLD (Kalal et al. 2012), ADMM (Ayazoglu et al. 2012), and IHTLS (Dicle et al. 2013) on the BBT, BUFFY and music video datasets. We also show the tracking results with the pre-trained features without adapting to a specific video. Note that the results shown in Table 6 are based on the overall evaluation. We leave the results from each individual sequence in the supplementary material (https://sites.google.com/site/shunzhang876/facetracking).

The mTLD method does not perform well on both datasets in terms of recall, precision, F1, and MOTA metrics. The ADMM (Ayazoglu et al. 2012) and IHTLS (Dicle et al. 2013) schemes often generate numerous identity switches and fragments as both methods do not re-identify persons well when abrupt camera motions or shot changes occur. The tracker with the pre-trained features is not effective to re-identify faces in different shots and achieve low MOTA. The TPI-mTLD scheme has more IDS and Frag than the TPI-SymTriplet method. The shot-level trajectories determined by the mTLD method are short and noisy since TLD trackers sometimes drift or do not perform well when large appearance changes occur. In contrast, TPI-FullModel performs well in terms of precision, F1, and MOTA metrics, with significantly fewer identity switches and fragments.

[4] The method in Du and Chellappa (2016) manually corrected some annotation errors and added several missing face tracks. Both the code and data are not publicly available.

6.5.2 Qualitative Results

Figure 13 shows sample tracking results of our algorithm with TPI-FullModel features on all eight music videos. Figure 14 shows the results on three BUFFY videos and three selected BBT sequences. The numbers and the colors indicate the inferred identities of the targets. The proposed algorithm is able to track multiple faces well despite large appearance variations in unconstrained videos. In Fig. 13, for example, there are significant changes in scale and appearance (due to makeup and hairstyle) in the Hello Bubble sequence (first row). In the fourth row, the six singers have similar looks and thus make multi-face tracking particularly challenging within and across shots. Nonetheless, our approach can distinguish the faces and track them reliably with few id switches. The results in other rows illustrate that our method is able to generate correct identities and trajectories when the same person appears in different shots or different scenes.

6.5.3 Time Analysis

We provide a complete time analysis of each component of the proposed method in Table 7. The Buffy02 episode is about 42 min. Our total processing time is about 40 minutes. The step of adaptive feature learning takes about 50% of the time. There is a trade-off between performance and time. Adapting the network with more iterations and training samples yields improved results, but at the cost of additional time for processing the video. As a result, our technique is less suitable for online streaming video processing where the computational time is a critical concern.

6.6 Ablation Study

We conduct an ablation study to demonstrate the effectiveness of individual modules in our approach by removing modules.

(1) TPI-FullModel-noPT: without pre-training model.
(2) TPI-FullModel-noFT: without using the contextual constraints to fine-tune the pre-trained model.
(3) TPI-FullModel-noSL: without the SymTriplet loss (Replaced with the conventional triplet loss).
(4) TPI-FullModel-noT: without linking tracklets within each shot.
(5) TPI-FullModel-noC: without clustering the shot-level tracklets across different shots.

All experiments are conducted on the BBT dataset using the multi-target tracking metrics.

Table 6 Quantitative comparison with other state-of-the-art multi-target tracking methods on the BBT and music video datasets. Among these trackers, mTLD is an online tracking method while others are running offline

Method	Recall↑ (%)	Precision↑ (%)	F1↑ (%)	FAF↓	IDS↓	Frag↓	MOTA↑ (%)	MOTP↑ (%)
BBT dataset								
mTLD (Kalal et al. 2012)	1.1	8.1	1.9	**0.18**	**8**	**83**	− 11.2	73.2
ADMM (Ayazoglu et al. 2012)	*78.3*	56.8	65.8	0.49	2709	4623	39.5	72.7
IHTLS (Dicle et al. 2013)	77.7	63.4	69.8	0.49	2648	4496	39.2	72.7
Pre-trained	45.0	76.8	56.8	*0.19*	908	*2435*	30.0	**77.9**
TPI-mTLD	63.7	78.8	70.5	0.24	1224	3487	44.6	*77.6*
TPI-Siamese	74.5	*81.4*	77.8	0.24	884	4051	56.1	77.4
TPI-Triplet	76.2	80.2	78.1	0.27	944	4223	55.8	77.3
TPI-SymTriplet	76.6	81.0	78.7	0.26	846	4261	57.2	77.2
TPI-SymTriplet-Contx	76.8	**81.7**	*79.2*	0.23	817	4073	*59.6*	*77.6*
TPI-FullModel	**80.5**	79.4	**79.9**	0.22	*810*	3806	**60.1**	77.3
BUFFY dataset								
mTLD (Kalal et al. 2012)	4.6	21.5	7.6	0.32	**192**	**453**	− 8.2	69.1
ADMM (Ayazoglu et al. 2012)	**78.3**	64.9	70.9	0.37	1420	2445	31.6	*70.1*
IHTLS (Dicle et al. 2013)	*78.0*	68.9	73.2	0.30	1558	*2424*	38.1	**70.2**
Pre-trained	52.3	72.1	60.6	**0.12**	405	2672	38.3	68.8
tpi-mTLD	65.4	73.5	69.2	0.27	413	2503	45.3	**70.2**
TPI-Siamese	66.1	74.3	70.0	0.19	389	2470	45.6	**70.2**
tpi-Triplet	67.3	74.6	70.8	0.20	388	2462	47.4	**70.2**
TPI-SymTriplet	68.1	*74.7*	71.2	0.19	363	2460	47.6	**70.2**
TPI-SymTriplet-Contx	70.9	**77.5**	*74.0*	*0.18*	*293*	2446	*49.4*	**70.2**
TPI-FullModel	71.2	**77.5**	**74.2**	*0.18*	363	2471	**49.6**	**70.2**
Music video dataset								
mTLD (Kalal et al. 2012)	9.7	36.1	15.3	0.39	**280**	**621**	− 7.7	68.4
ADMM (Ayazoglu et al. 2012)	**75.5**	61.8	68.0	0.50	2382	2959	51.7	63.7
IHTLS (Dicle et al. 2013)	**75.5**	68.0	71.6	0.41	2013	2880	56.2	63.7
Pre-trained	60.1	88.8	71.7	**0.17**	931	*2140*	51.5	*79.5*
TPI-mTLD	69.1	88.1	77.4	0.21	1914	2786	57.7	**80.1**
TPI-Siamese	71.5	89.4	79.5	*0.19*	986	2512	62.3	64.0
TPI-Triplet	71.8	88.8	79.4	0.20	902	2546	61.8	64.2
TPI-SymTriplet	71.8	89.7	79.8	*0.19*	699	2563	62.8	64.3
TPI-SymTriplet-Contx	73.2	**90.5**	*80.9*	*0.19*	*625*	2417	*64.1*	64.2
TPI-FullModel	*73.4*	*90.4*	**81.0**	*0.19*	789	2388	**64.3**	64.2

Bold values indicate the best and italic values indicate the second-best performance

6.6.1 Effect of Pre-training

Our feature adaption approach builds upon a pre-trained face CNN model (pre-trained using an external face dataset to learn identity-preserving features). To evaluate the effect of model pre-training, we evaluate two models: TPI-FullModel with and without pre-training while keeping all other factors are the same. In Table 8 we compare the TPI-FullModel with TPI-FullModel-noPT methods. The model without pre-training (TPI-FullModel-noPT) has significantly lower performance in terms of Recall, F1 and MOTA. The results show that pre-training identity-preserving features are important for the success of our feature adaption method.

6.6.2 Effect of Fine-Tuning and SymTriplet Loss

We generate the TPI-FullModel with and without fine-tuning and compare tracking results between the SymTriplet loss and the conventional triplet loss. Table 8 shows that both TPI-FullModel-noFT has lower performance in terms of Recall, F1, IDS, Frag and MOTA metrics, highlighting the importance of learning video-specific features. We evaluate the effectiveness of the proposed SymTriplet loss with compar-

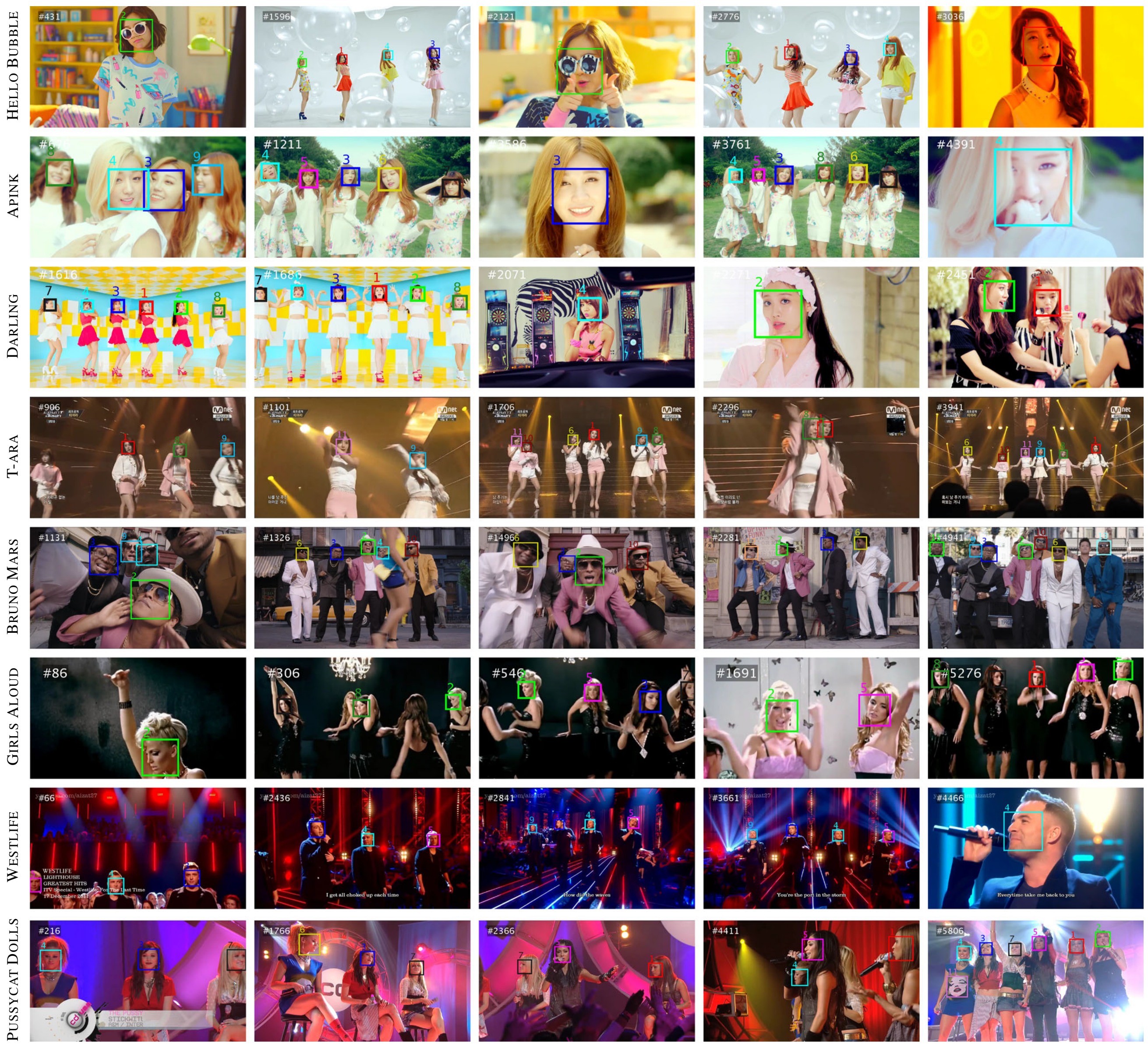


Fig. 13 Tracking results on YouTube Music video dataset. Shown from the top to bottom are Hello Bubble, Apink, Darling, T- ara, Bruno Mars, Girls Aloud, Westlife and Pussycat Dolls. The faces of the different people are color coded (Color figure online)

isons to the conventional triplet loss (TPI-FullModel-noSL). The TPI-FullModel with the SymTriplet loss performs well against the TPI-FullModel-noSL method since in the presented SymTriplet loss positive sample pairs are pulled closer and negative samples are pushed away from the positive pairs.

6.6.3 Effect of Tracking and Clustering

Table 8 shows that the performance of both TPI-FullModel-noT and TPI-FullModel-noC methods decreases in terms of Recall, F1, IDS, Frag and MOTA metrics. Without linking tracklets within each shot, the TPI-FullModel-noT method cannot recover several missed faces, and thus yields lower performance on the Recall. Although some separated tracklets in each shot can be grouped together by the clustering algorithm, numerous tracklets may be grouped incorrectly due to the lack of consideration of spatio-temporal coherence within each shot. This explains the increase of IDS and Frag. Without clustering tracklets across different shots, the TPI-FullModel-noC method assigns each short tracklet in each shot with different identities, which results in significantly more identity switches and lower MOTA.

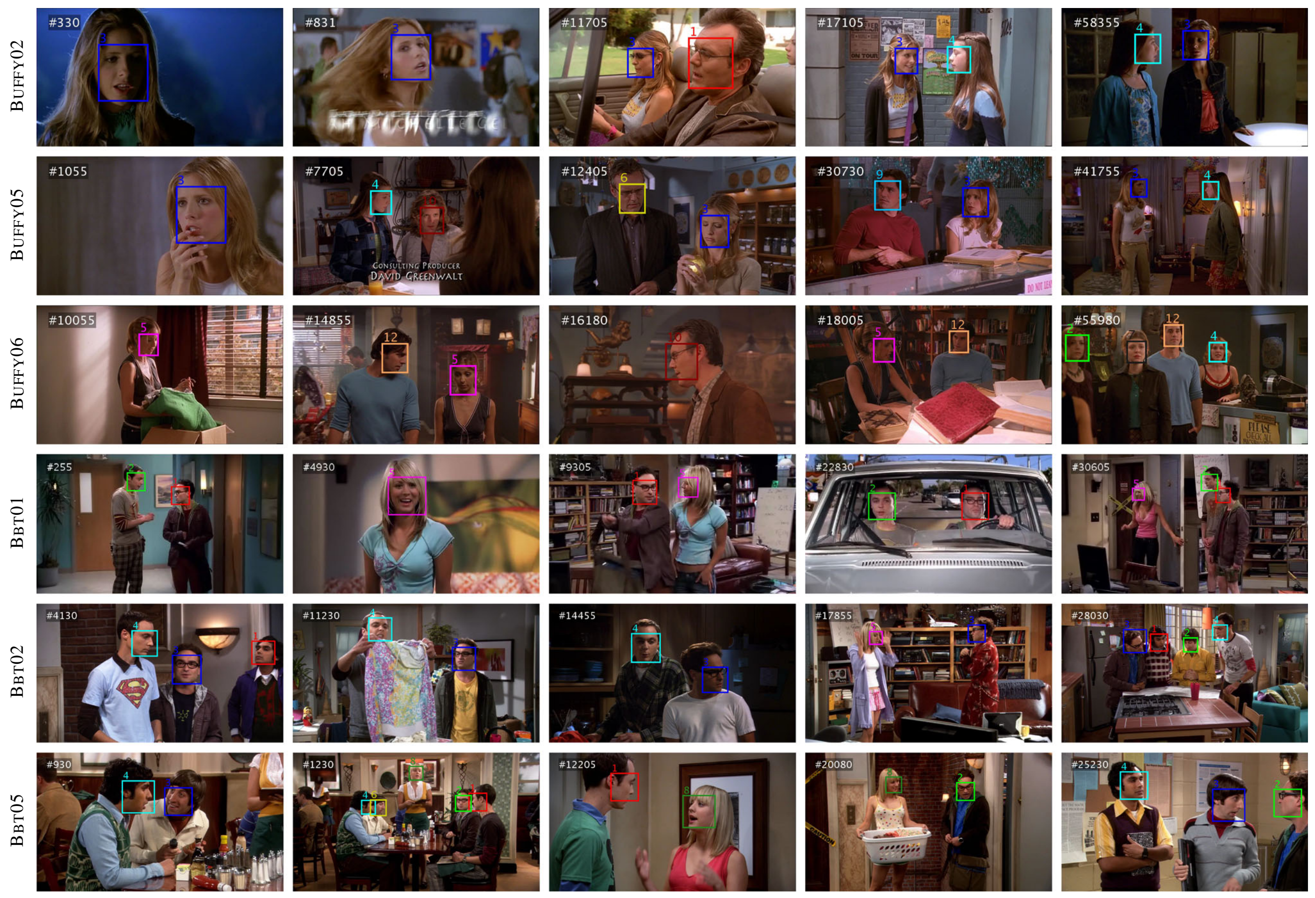


Fig. 14 Tracking results on BUFFY and BBT dataset. The faces of the different people are color coded (Color figure online)

Table 7 Time analysis of each component on BUFFY02 episode

Component	Time
Pre-train face CNN	∼7 h
BUFFY02 episode	∼42 min
Detect faces	∼900 s
Shot detects	∼20 s
Generate constraints	∼200 s
Adaptive feature learning	∼20 min
Linking tracklets within shots	∼100 s
Clustering tracklets across shots	∼15 s

6.6.4 Effect of Backbone Network

We evaluate our algorithm using a ResNet-50 model as our base feature extractor. We choose the ResNet-50 model publicly available on the VGG-Face2 project (Cao et al. 2018) as our pre-trained face model (pre-trained on the VGG-Face2 training dataset and the Ms-Celeb-1M dataset). Based on this pre-trained ResNet model, we then replace the classification output layer with a 64-node feature embedding layer (same as the setting in adapting a pre-trained AlexNet model) to construct the triplet network for learning adaptive discriminative features. We train the ResNet model using the automatically discovered samples in the target videos. We refer to the adapted model as TPI-FullModel-ResNet. Table 9 presents the clustering and tracking results on the music video dataset. With a stronger backbone network, the TPI-FullModel-ResNet has moderate performance improvement in the term of the weighted purities on the music video dataset. In particular, on the videos HELLOBUBBLE, DARLING and APINK, the model with ResNet achieves higher clustering performance than that with AlexNet. For example, in the DARING sequence, the TPI-FullModel-ResNet achieves the weighted purity of 0.84, significantly outperforming the TPI-FullModel with AlexNet: 0.73. For evaluation on tracking, the ResNet based model also outperforms the AlexNet based model in term of IDs (drops 187), Frag (drops 82) and MOTA (improves 0.8%). We attribute the performance improvement to the larger training datasets and a network with higher capacity.

Table 8 Ablation study on the BBT dataset

Method	Recall↑ (%)	Precision↑ (%)	F1↑ (%)	FAF↓	IDS↓	Frag↓	MOTA↑(%)	MOTP↑ (%)
BBT dataset								
TPI-FullModel-noPT	58.4	80.7	67.8	*0.20*	976	*2451*	42.2	77.2
TPI-FullModel-noFT	56.0	80.2	66.0	**0.19**	925	**2438**	40.4	**77.9**
TPI-FullModel-noSL	*76.6*	81.0	*78.7*	0.26	*846*	4261	*57.2*	77.2
TPI-FullModel-noT	65.3	**81.3**	72.4	0.24	952	5109	43.1	*77.4*
TPI-FullModel-noC	68.7	*81.2*	74.4	0.25	1496	4082	46.2	77.2
TPI-FullModel	**80.5**	79.4	**79.9**	0.22	**810**	3806	**60.1**	77.3

Bold values indicate the best and italic values indicate the second-best performance

Table 9 Qualitative results with the replacement of the AlexNet model by the ResNet model. For the clustering results, the weighted purity of each video is measured on the actual number of clusters

Methods	Clustering results on Music dataset							
	T- ARA	PUSSYCAT DOLLS	BRUNO MARS	HELLO BUBBLE	DARLING	APINK	WESTLIFE	GIRLS ALOUD
TPI-FullModel	0.95	0.93	0.91	0.81	0.73	0.86	0.91	**0.97**
TPI-FullModel-ResNet	**0.98**	**0.95**	**0.96**	**0.84**	**0.84**	**0.90**	**0.93**	**0.97**

Methods	Multi-face tracking results on Music dataset							
	Recall↑	Precision↑	F1↑	FAF↓	IDS↓	Frag↓	MOTA↑	MOTP↑
TPI-FullModel	73.4%	90.4%	81.0%	**0.19**	789	2388	64.3%	**64.2%**
TPI-FullModel-ResNet	**73.8%**	**90.6%**	**81.3%**	**0.19**	**602**	**2306**	**65.1%**	**64.2%**

Bold values indicate the best performance

6.6.5 Effect of Shot Detector, Face Detector, and Tracker

We evaluate the effect of different shot detectors, face detectors, and trackers used in our approach.

(1) TPI-FullModel-shot: applying the computed tomography method (Varghese and Nair 2016) to generate shot boundary detections in videos.
(2) TPI-FullModel-Yolo: applying the state-of-the-art object detector, YOLOv3 (Redmon and Farhadi 2018), to generate face detections for each frame.
(3) TPI-FullModel-DSort: applying the Deep Sort Tracking method in Wojke et al. (2017) to generate shot-level face tracks within each shot.

All experiments are conducted on the music dataset using the multi-target tracking metrics.

Table 10 shows the quantitative tracking results with different shot detectors, face detectors and trackers on the music dataset. The TPI-FullModel-shot method performs equally well as the TPI-FullModel scheme as both shot detection algorithms (either Varghese and Nair (2016) or the method on the website[5] used in our approach) detect almost all of the shot boundaries detections correctly on the music videos. The TPI-FullModel-Yolo method outperforms the TPI-FullModel scheme in term of Recall, F1, IDS, Frag and MOTA, since it uses the state-of-the-art object detector. The TPI-FullModel-Yolo method can generate more accurate face detections which help link more and longer tracklets within each shot. The TPI-FullModel-DSort scheme also outperforms the TPI-FullModel approach in term of IDS, Frag and MOTA, since it uses a deep CNN method to link detections into shot-level tracklets.

[5] http://sourceforge.net/projects/shot-change/.

6.7 Pedestrian Tracking Across Cameras

In this section, we show that the proposed method for learning adaptive discriminative features from tracklets is also applicable to other objects, e.g., pedestrians or cars in surveillance videos. We validate our approach on the task of pedestrian tracking from multiple non-overlapping cameras.

The problem of multiple target tracking across cameras is challenging as we need to re-identify people from different images acquired at different viewing angles and imaging conditions. In unconstrained scenes, the appearances of people also exhibit significant differences across cameras. The motion cues of people are unreliable due to the non-overlapping views without knowing camera configurations apriori. The re-identification problem becomes even

Table 10 Qualitative tracking results with different shot detectors, face detectors and trackers on the music dataset

Methods	Recall↑ (%)	Precision↑ (%)	F1↑ (%)	FAF↓	IDS↓	Frag↓	MOTA↑ (%)	MOTP↑ (%)
TPI-FullModel	73.4	90.4	81.0	0.19	789	2388	64.3	64.2
TPI-FullModel-shot	73.4	90.4	81.0	0.19	786	2382	64.3	64.2
TPI-FullModel-Yolo	77.1	90.1	83.1	0.20	714	2308	65.2	64.2
TPI-FullModel-DSort	74.1	90.2	81.4	0.19	762	2353	64.5	64.2

Table 11 Tracking results on the DukeMTMC datset

Method	IDS↓	Frag↓	MOTA↑ (%)	MOTP↑ (%)	IDP↑ (%)	IDR↑ (%)	IDF1↑ (%)
Camera 2							
Ristani et al. (2016)	866	1929	49.2	61.7	69.1	63.8	66.3
TPI-FullModel	835	2011	51.0	60.9	69.4	64.6	66.9
Camera 5							
Ristani et al. (2016)	162	292	73.1	70.5	84.9	68.0	75.5
TPI-FullModel	154	307	75.7	68.5	85.9	69.2	76.7

more challenging when a large number of people needs to be tracked across views.

Similar to pre-training a CNN using face recognition dataset for learning identity-preserving features, we first train a CNN for people re-identification using the Market1501 dataset (Zheng et al. 2015) containing 32,668 images of 1501 identities. We evaluate our method on the DukeMTMC (Ristani et al. 2016) dataset which contains surveillance footage from 8 cameras with approximately 85 min of videos for each one.

We conduct the experiment using images from camera 2 and 5 because they are disjoint and have the most number of people. The performance evaluation on the complete dataset can be found in our project webpage.[6] We use the pedestrian detections provided by the DukeMTMCT dataset as our inputs and apply the two-threshold strategy to generate tracklets on the videos from both cameras. Next, we collect training samples based on the tracklets using spatio-temporal and contextual constraints. We exploit the contextual constraints by locating the torso region to extract local clothing features for measuring similarity. Similar to the experiments on multi-face tracking, we fine-tune the pre-trained CNN with the discovered training samples using the SymTriplet loss function.

After extracting the learned features for each detection, we first link the tracklets within one camera into camera-level trajectories. We then group these camera-level trajectories into tracking results across the two cameras. Following Ristani et al. (2016), we measure the tracking performance using identification precision (IDP), identification recall (IDR), and the corresponding F1 score IDF1, as well as other metrics. The identification precision (recall) is the fraction of computed (ground-truth) detections that are correctly identified.

[6] https://sites.google.com/site/shunzhang876/facetracking.

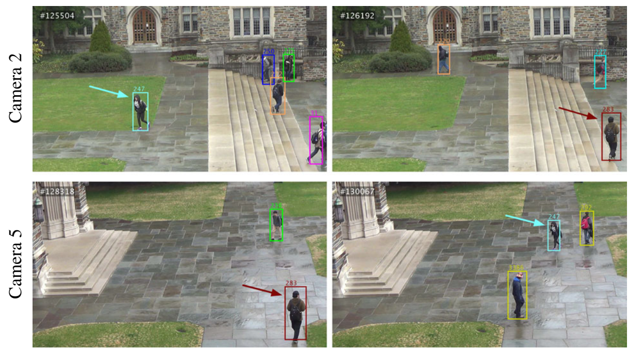


Fig. 15 Sample pedestrian tracking results. Shown from the top to bottom are Camera 2 and Camera 5 of the DukeMTMC dataset. The different people are color coded (Color figure online)

The IDF1 metric is the ratio of correctly identified detections over the average number of ground-truth and computed detections. Both ID precision and ID recall indicate tracking trade-offs, while the IDF1 score allows ranking all methods on a single scale that balances identification precision and recall through the harmonic mean. Table 11 shows the tracking results on both cameras in the DukeMTMC dataset. Overall, the proposed method performs favorably against the other methods in Ristani et al. (2016) in term of IDS, MOTA, IDP, and IDF1. We show sample visual results of the DukeMTMC dataset in Fig. 15. Person 247 and person 283 both appear in Camera 2 and Camera 5, and are correctly matched across cameras with our method.

6.8 Discussion

While the proposed algorithm performs favorably against the state-of-the-art face tracking and clustering methods in

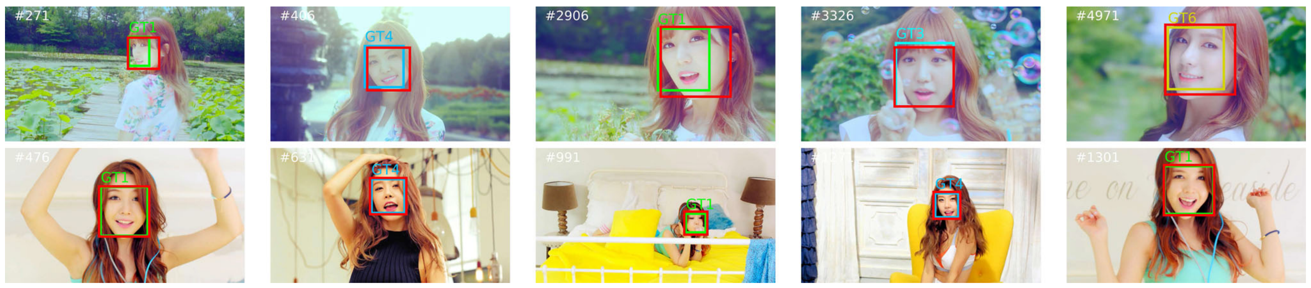


Fig. 16 Failure cases. Our method incorrectly identifies different persons as the same one across shots on the Apink and Darling sequences. Numbers and colors of rectangles indicate the *ground-truth* identities of persons. The red rectangles show the predicted locations and are tracked as one person by our method. On the Apink sequence on the top row, Persons 1, 3, 4 and 6 are incorrectly assigned with the same identity. On the Darling sequence on the bottom row, our method incorrectly identifies Persons 1 and 4 as the same one across shots (Color figure online)

handling challenging video sequences, there are four main limitations.

First, as our algorithm takes face detections as inputs, the tracking performance depends on whether faces can be reliably detected. For example, in the fourth row of Fig. 13, the leftmost person was not detected in frame 906 and next few images due to occlusion. In addition, falsely detected faces could be incorrectly linked as a trajectory, e.g., the Marilyn Monroe image on the T-shirt in frame 5806 in the eighth row of Fig. 13.

Second, the proposed algorithm may not perform well on sequences where many shots contain only one single person. We show in Fig. 16 two failure cases in the Darling and Apink sequences. In such cases, the proposed method does not generate negative face pairs for training the Siamese/triplet network for distinguishing similar faces. As such, different persons are incorrectly identified as the same one. One remedy is to exploit other weak supervision signals (e.g., scripts, voice, contextual information) to generate visual constraints for different scenarios.

Third, the CNN fine-tuning process is time-consuming. It takes around 1 hour on an NVIDIA GT980Ti GPU for 10,000 back-propagation iterations. There are three approaches that may alleviate this issue. First, we may use faster training algorithms (Lin et al. 2015). Second, similar to the method in Bertinetto et al. (2016), we may train a deep CNN model offline instead of online fine-tuning to improve run-time performance. Third, for TV Sitcom episodes we can use one or a few videos for feature adaptation and apply the learned features to all other episodes. Note that we only need to adapt features once as the main characters are the same. In Table 2, we train TPI-SymTriplet features on BBT02 (referred to as TPI-SymTriplet-BBT02) and evaluate on other episodes. Although the weight purity of TPI-SymTriplet-BBT02 is slightly inferior to that of TPI-SymTriplet, it still outperforms the pre-trained and VGG-Face features.

7 Conclusions

In this paper, we tackle the multi-face tracking problem in unconstrained videos by learning video-specific features. We first pre-train a CNN on a large-scale face recognition dataset to learn identity-preserving face representation. We then adapt the pre-trained CNN using training samples extracted through the spatio-temporal and contextual constraints. To learn discriminative features for handling large appearance variations of faces presented in a specific video, we propose the SymTriplet loss function. Using the learned features for modeling face tracklets, we apply an EM clustering algorithm to link face tracklets across multiple shots. In addition to multi-face tracking, we demonstrate that the proposed algorithm can also be applied to other domains such as pedestrian tracking across multiple cameras. Experimental results show that the proposed algorithm outperforms the state-of-the-art methods in terms of clustering accuracy and tracking performance. As the performance of our approach depends on the automatically discovered visual constraints in the video, we believe that exploiting multi-modal information (e.g., sound/script alignment) is a promising direction for further improving the performance.

Acknowledgements The work is supported by National Basic Research Program of China (973 Program, 2015CB351705), National Key Research and Development Program of China (2017YFA0700805), NSFC (61703344), Office of Naval Research (N0014-16-1-2314), Ministry of Science and ICT of Korea (NRF-2017R1A2B4011928 and Next-Generation Information Computing Development program NRF-2017M3C4A7069369), NSF CRII (1755785), NSF CAREER (1149783) and gifts from Adobe, Panasonic, NEC, and NVIDIA.

References

Andriluka, M., Roth, S., & Schiele, B. (2008). People-tracking-by-detection and people-detection-by-tracking. In *CVPR*.

Andriyenko, A., & Schindler, K. (2011). Multi-target tracking by continuous energy minimization. In *CVPR*.

Andriyenko, A., Schindler, K., & Roth, S. (2012). Discrete-continuous optimization for multi-target tracking. In *CVPR*.

Anguelov, D., Lee, K. C., Gokturk, S. B., & Sumengen, B. (2007). Contextual identity recognition in personal photo albums. In *CVPR*.

Ayazoglu, M., Sznaier, M., & Camps, O. I. (2012) Fast algorithms for structured robust principal component analysis. In *CVPR*.

Bauml, M., Tapaswi, M., & Stiefelhagen, R. (2013). Semi-supervised learning with constraints for person identification in multimedia data. In *CVPR*.

Ben Shitrit, H., Berclaz, J., Fleuret, F., & Fua, P. (2011). Tracking multiple people under global appearance constraints. In *ICCV*.

Berclaz, J., Fleuret, F., Turetken, E., & Fua, P. (2011). Multiple object tracking using k-shortest paths optimization. *PAMI*, *33*(9), 1806–1819.

Bertinetto, L., Henriques, J. F., Valmadre, J., Torr, P., & Vedaldi, A. (2016) Learning feed-forward one-shot learners. In *NIPS* (pp. 523–531).

Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., & Torr, P. H. (2016). Fully-convolutional siamese networks for object tracking. In *European conference on computer vision* (pp. 850–865). Springer.

Bourdev, L., & Malik, J. (2009). Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV* (pp. 1365–1372).

Breitenstein, M. D., Reichlin, F., Leibe, B., Koller-Meier, E., & Van Gool, L. (2009). Robust tracking-by-detection using a detector confidence particle filter. In *ICCV*.

Brendel, W., Amer, M., & Todorovic, S. (2011). Multiobject tracking as maximum weight independent set. In *CVPR*.

Caelles, S., Maninis, K. K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., & Van Gool, L. (2017). One-shot video object segmentation. In *CVPR*. IEEE.

Cao, Q., Shen, L., Xie, W., Parkhi, O. M., & Zisserman, A. (2018). Vggface2: A dataset for recognising faces across pose and age. In *FG* (pp. 67–74). IEEE.

Chopra, S., Hadsell, R., & LeCun, Y. (2005) Learning a similarity metric discriminatively, with application to face verification. In *CVPR*.

Cinbis, R. G., Verbeek, J., & Schmid, C. (2011). Unsupervised metric learning for face identification in tv video. In *ICCV*.

Collins, R. T. (2012). Multitarget data association with higher-order motion models. In *CVPR*.

Collins, R. T., Liu, Y., & Leordeanu, M. (2005). Online selection of discriminative tracking features. *PAMI*, *27*(10), 1631–1643.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *CVPR*.

Dicle, C., Camps, O. I., Sznaier, M. (2013). The way they move: Tracking multiple targets with similar appearance. In *ICCV*.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., et al. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*.

Du, M., & Chellappa, R. (2016). Face association for videos using conditional random fields and max-margin markov networks. *PAMI*, *38*(9), 1762–1773.

El Khoury, E., Senac, C., & Joly, P. (2010). Face-and-clothing based people clustering in video content. In *ICMR*.

Everingham, M., Sivic, J., & Zisserman, A. (2006). "Hello! My name is... Buffy"—Automatic naming of characters in tv video. In *BMVC*.

Fernando, B., Habrard, A., Sebban, M., & Tuytelaars, T. (2013). Unsupervised visual domain adaptation using subspace alignment. In *ICCV* (pp. 2960–2967).

Fernando, B., Tommasi, T., & Tuytelaars, T. (2015). Joint cross-domain classification and subspace learning for unsupervised adaptation. *Pattern Recognition Letters*, *65*, 60–66.

Fulkerson, B., Vedaldi, A., & Soatto, S. (2008). Localizing objects with smart dictionaries. In *ECCV*.

Ganin, Y., & Lempitsky, V. (2014) Unsupervised domain adaptation by backpropagation. arXiv.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., et al. (2016). Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, *17*(1), 2096–2030.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). Generative adversarial nets. In *NIPS* (pp. 2672–2680).

Grabner, H., & Bischof, H. (2006). On-line boosting and vision. In *CVPR*.

Gupta, S., Hoffman, J., & Malik, J. (2016). Cross modal distillation for supervision transfer. In *CVPR* (pp. 2827–2836).

Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *CVPR*.

Hu, J., Lu, J., & Tan, Y. P. (2014). Discriminative deep metric learning for face verification in the wild. In *CVPR*.

Huang, C., Li, Y., Ai, H., et al. (2006). Robust head tracking with particles based on multiple cues. In *ECCVW*.

Huang, C., Wu, B., & Nevatia, R. (2008). Robust object tracking by hierarchical association of detection responses. In *ECCV*.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R. et al. (2014). Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*.

Jiang, H., Fels, S., & Little, J. J. (2007). A linear programming approach for multiple object tracking. In *CVPR*.

Joon Oh, S., Benenson, R., Fritz, M., & Schiele, B. (2015). Person recognition in personal photo collections. In *ICCV* (pp. 3862–3870).

Kalal, Z., Mikolajczyk, K., & Matas, J. (2012). *Tracking-learning-detection. PAMI*, *34*(7), 1409–1422.

Kaucic, R., Perera, A. A., Brooksby, G., Kaufhold, J., & Hoogs, A. (2005). A unified framework for tracking through occlusions and across sensor gaps. In *CVPR*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *NIPS*.

Kuo, C. H., Huang, C., & Nevatia, R. (2010). Multi-target tracking by on-line learned discriminative appearance models. In *CVPR*.

Kuo, C. H., & Nevatia, R. (2011). How does person identity recognition help multi-person tracking? In *CVPR*.

Leibe, B., Schindler, K., & Van Gool, L. (2007). Coupled detection and trajectory estimation for multi-object tracking. In *ICCV*.

Li, Y., Ai, H., Yamashita, T., Lao, S., & Kawade, M. (2007). Tracking in low frame rate video: A cascade particle filter with discriminative observers of different lifespans. In *CVPR*.

Li, Y., Huang, C., & Nevatia, R. (2009). Learning to associate: Hybrid-boosted multi-target tracker for crowded scene. In *CVPR*.

Lin, D., Kapoor, A., Hua, G., & Baker, S. (2010). Joint people, event, and location recognition in personal photo collections using cross-domain context. In *ECCV*.

Lin, Z., Courbariaux, M., Memisevic, R., & Bengio, Y. (2015). Neural networks with few multiplications. arXiv.

Liu, M. Y., & Tuzel, O. (2016). Coupled generative adversarial networks. In *NIPS* (pp. 469–477).

Long, M., Wang, J., Ding, G., Sun, J., & Yu, P. S. (2013). Transfer feature learning with joint distribution adaptation. In *ICCV* (pp. 2200–2207).

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *IJCV*, *60*(2), 91–110.

Mathias, M., Benenson, R., Pedersoli, M., & Van Gool, L. (2014). Face detection without bells and whistles. In *ECCV*.

Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition. In *BMVC*.

Paul, G., Elie, K., Sylvain, M., Jean-Marc, O., & Paul, D. (2014). A conditional random field approach for audio-visual people diarization. In *ICASSP*.

Pellegrini, S., Ess, A., Schindler, K., & Van Gool, L. (2009). You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*.

Perera, A. A., Srinivas, C., Hoogs, A., Brooksby, G., & Hu, W. (2006). Multi-object tracking through simultaneous long occlusions and split-merge conditions. In *CVPR*.

Pernici, F. (2012). Facehugger: The alien tracker applied to faces. In *ECCV*.

Ramanan, D., Baker, S., & Kakade, S. (2007). Leveraging archival video for building face datasets. In *ICCV*.

Rao, Y., Lin, J., Lu, J., & Zhou, J. (2017). Learning discriminative aggregation network for video-based face recognition. In *Proceedings of the IEEE international conference on computer vision* (pp. 3781–3790).

Rao, Y., Lu, J., & Zhou, J. (2019). Learning discriminative aggregation network for video-based face recognition and person re-identification. *International Journal of Computer Vision*, *127*(6–7), 701–718.

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv.

Ristani, E., Solera, F., Zou, R., Cucchiara, R., Tomasi, C. (2016). Performance measures and a data set for multi-target, multi-camera tracking. In *ECCVW* (pp. 17–35). Springer.

Roth, M., Bauml, M., Nevatia, R., & Stiefelhagen, R. (2012). Robust multi-pose face tracking by multi-stage tracklet association. In *ICPR*.

Saenko, K., Kulis, B., Fritz, M., & Darrell, T. (2010). Adapting visual category models to new domains. In *ECCV* (pp. 213–226). Springer.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. In *CVPR*.

Shu, R., Bui, H. H., Narui, H., & Ermon, S. (2018). A dirt-t approach to unsupervised domain adaptation. In *ICLR*.

Sivic, J., Everingham, M., & Zisserman, A. (2009). "Who are you?"—Learning person specific classifiers from video. In *CVPR*.

Stauffer, C. (2003). Estimating tracking sources and sinks. In *CVPR*.

Sun, B., Feng, J., & Saenko, K. (2016). Return of frustratingly easy domain adaptation. In *AAAI* (Vol. 6, p. 8).

Sun, B., & Saenko, K. (2016). Deep coral: Correlation alignment for deep domain adaptation. In *ECCV* (pp. 443–450). Springer.

Sun, Y., Chen, Y., Wang, X., & Tang, X. (2014a). Deep learning face representation by joint identification-verification. In *NIPS*.

Sun, Y., Wang, X., & Tang, X. (2014b). Deep learning face representation from predicting 10,000 classes. In *CVPR*.

Taigman, Y., Polyak, A., & Wolf, L. (2016). Unsupervised cross-domain image generation. arXiv preprint arXiv:1611.02200.

Taigman, Y., Yang, M., Ranzato, M., & Wolf, L. (2014). DeepFace: Closing the gap to human-level performance in face verification. In *CVPR*.

Tang, Z., Zhang, Y., Li, Z., & Lu, H. (2015). Face clustering in videos with proportion prior. In *IJCAI*.

Tapaswi, M., Bauml, M., & Stiefelhagen, R. (2012). "Knock! Knock! Who is it?" probabilistic person identification in tv-series. In *CVPR*.

Tapaswi, M., Parkhi, O. M., Rahtu, E., Sommerlade, E., Stiefelhagen, R., & Zisserman, A. (2014). Total cluster: A person agnostic clustering method for broadcast videos. In *ICVGIP*.

Tzeng, E., Hoffman, J., Saenko, K., & Darrell, T. (2017). Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7167–7176).

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., & Darrell, T. (2014). Deep domain confusion: Maximizing for domain invariance. arXiv.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-sne. *JMLR*, *9*(2579–2605), 85.

Varghese, J., & Nair, K. (2016). Detecting video shot boundaries by modified tomography. In *VisionNet* (pp. 131–135). ACM.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *CVPR*.

Wang, B., Wang, G., Chan, K. L., & Wang, L. (2014). Tracklet association with online target-specific metric learning. In *CVPR*.

Wojke, N., Bewley, A., & Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)* (pp. 3645–3649). IEEE.

Wu, B., Lyu, S., Hu, B. G., & Ji, Q. (2013a). Simultaneous clustering and tracklet linking for multi-face tracking in videos. In *ICCV*.

Wu, B., Zhang, Y., Hu, B. G., & Ji, Q. (2013b). Constrained clustering and its application to face clustering in videos. In *CVPR*.

Xiao, S., Tan, M., & Xu, D. (2014). Weighted block-sparse low rank representation for face clustering in videos. In *ECCV*.

Xing, J., Ai, H., & Lao, S. (2009). Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. In *CVPR*.

Yang, B., Nevatia, R. (2012a). Multi-target tracking by online learning of non-linear motion patterns and robust appearance models. In *CVPR*.

Yang, B., & Nevatia, R. (2012b). Online learned discriminative part-based appearance models for multi-human tracking. In *ECCV*.

Yi, D., Lei, Z., Liao, S., & Li, S. Z. (2014). Learning face representation from scratch. arXiv.

Yoon, J. S., Rameau, F., Kim, J., Lee, S., Shin, S., & Kweon, I. S. (2017). Pixel-level matching for video object segmentation using convolutional neural networks. In *ICCV* (pp. 2186–2195). IEEE.

Zhang, L., Li, Y., & Nevatia, R. (2008). Global data association for multi-object tracking using network flows. In *CVPR*.

Zhang, N., Paluri, M., Taigman, Y., Fergus, R., & Bourdev, L. (2015). Beyond frontal faces: Improving person recognition using multiple cues. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4804–4813).

Zhang, S., Wang, J., Wang, Z., Gong, Y., & Liu, Y. (2015). Multi-target tracking by learning local-to-global trajectory models. *PR*, *48*(2), 580–590.

Zhang, Z., Luo, P., Loy, C. C., & Tang, X. (2016). Joint face representation adaptation and clustering in videos. In *ECCV*.

Zhao, X., Gong, D., & Medioni, G. (2012). Tracking using motion patterns for very crowded scenes. In *ECCV*.

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., & Tian, Q. (2015). Scalable person re-identification: A benchmark. In *ICCV*.

Zhou, C., Zhang, C., Fu, H., Wang, R., & Cao, X. (2015). Multi-cue augmented face clustering. In *ACM MM*.