Video Face Clustering with Self-Supervised Representation Learning

Vivek Sharma, Makarand Tapaswi, M. Saquib Sarfraz, and Rainer Stiefelhagen

Abstract—Characters are a key component of understanding the story conveyed in TV series and movies. With the rise of advanced deep face models, identifying face images may seem like a solved problem. However, as face detectors get better, clustering and identification need to be revisited to address increasing diversity in facial appearance. In this paper, we propose unsupervised methods for feature refinement with application to video face clustering. Our emphasis is on distilling the essential information, *identity*, from the representations obtained using deep pre-trained face networks. We propose a self-supervised Siamese network that can be trained without the need for video/track based supervision, that can also be applied to image collections. We evaluate our methods on three video face clustering datasets. Thorough experiments including generalization studies show that our methods outperform current state-of-the-art methods on all datasets. This paper is extension of [1]. The datasets and code are available at `https://github.com/vivoutlaw/SSIAM`.

Index Terms—Video Understanding, Video Face Clustering, Self-Supervised Learning, Representation Learning, Siamese Networks, Variational Autoencoders.

1 Introduction

Long videos such as TV series episodes or movies are often pre-processed via shot and scene change detection to make the video more accessible. In recent years, person clustering and identification are gaining importance as several emerging research areas [2], [3], [4], [5] can benefit from it. For example, in video question-answering [3], most questions center around the characters asking who they are, what they do, and even why they act in certain ways. The related task of video captioning [2] often uses a character agnostic way (replacing names by *someone*) making the captions very artificial and uninformative (*e.g. someone* opens the door). However, recent work [6] suggests that more meaningful captions can be achieved from an improved understanding of characters. In general, the ability to predict which character appears where and when facilitates a deeper understanding of videos that is grounded in the storyline.

Motivated by this goal, person clustering [7], [8], [9], [10], [11] and identification [12], [13], [14], [15], [16] in videos has seen over a decade of research. In particular, fully automatic person identification is achieved in a weakly supervised manner either by aligning subtitles and transcripts [12], [13], [14], or using web images for actors and characters [16], [17]. On the other hand, clustering [7], [10], [18], [19], [20], [21] has mainly relied on *must-link* and *cannot-link* information obtained by tracking faces in a shot and analyzing their co-occurrence.

As face detectors improve (*e.g.* [22]), clustering and identification need to be revisited as more faces that exhibit extreme viewpoints, illumination, and resolution become available and need to be grouped or identified. Deep Convolutional Neural Networks (CNNs) have also yielded large performance gains for face representations [23], [24], [25], [26]. These networks are typically trained using hundreds-of-thousands to millions of face images gathered from the web, and show super-human performance on face verification tasks on images (LFW [27]) and videos (YouTube-Faces [28]). Nevertheless, it is important to note that faces in videos such as TV series/movies exhibit more variety in comparison to *e.g.* LFW, where the images are obtained from Yahoo News by cropping mostly frontal faces. While these deep models generalize well, they are difficult to train from scratch (require lots of training data), and are typically transferred to other datasets via *net surgery*: fine-tuning [10], [20], [29], or use of additional embeddings on the features from the last layer [11], [30], [31], or both.

Video face clustering also has potential applications in understanding other user-generated videos (*e.g.* content on YouTube) – mainly towards automatic summarization and content-based retrieval. For a method to work with such videos, it is especially important that the method be completely unsupervised (or self-supervised), as any required manual annotation will not scale with the exponential growth in the amount of video uploaded daily.

Representations. Clustering inherently builds on the notion of representations. We acknowledge the critical role of good features, and in this paper, we address the problem of effectively learning representations to improve video face

- *Vivek Sharma is with (1) The Department of Informatics - Institute for Anthropomatics and Robotics (IAR), Computer Vision for Human-Computer Interaction Lab (CV:HCI), Karlsruhe Institute of Technology, Karlsruhe, Germany; and (2) Massachusetts Institute of Technology, Cambridge, USA.*
E-mail: vvsharma@mit.edu
Web: https://web.media.mit.edu/∼vvsharma/
- *M. Saquib Sarfraz, and Rainer Stiefelhagen are with the Department of Informatics - Institute for Anthropomatics and Robotics (IAR), Computer Vision for Human-Computer Interaction Lab (CV:HCI), Karlsruhe Institute of Technology, Karlsruhe, Germany.*
- *Makarand Tapaswi is currently with Inria, Paris, France. A majority of this work was done when Makarand was at University of Toronto, and the Vector Institute, Toronto, Canada.*

clustering. A good feature representation should exhibit small intra-person distances (*positive* pair of faces from the same person should be close) and large inter-person-distance (*negative* pair of faces from different people should be far). Recent works show that CNN representations can be improved via positive and negative pairs that are discovered through a Markov Random Field (MRF) [10]; or a revised triplet-loss [20]. In contrast, we propose simple methods that do not require complex optimization functions or supervision to improve the feature representation. We emphasize that while video-level constraints are not new, they need to be used properly to extract the most out of them. This is especially in light of CNN face representations that are very similar even across different identities. For example, Figure 6 shows a large overlap between the cosine similarity score distributions of positive (same identity) and negative (different identities) face pairs using base features. More importantly, the absolute values of similarity scores between different identities are surprisingly high and all above 0.93.

Contributions. Given a set of face images or tracks from several characters, our goal is to group them such that face images in a cluster belong to the same character. We propose and evaluate several simple ideas: discriminative and generative (see Section 3), that aim to further improve deep network representations. Note that all methods proposed in this paper are either fully unsupervised, or use supervision that is obtained automatically, hence can be thought as unsupervised.

We propose two variants of *discriminative* approaches, and highlight the key differences below. In Track-supervised Siamese Network (TSiam), we include additional negative training pairs for singleton tracks – tracks that are not temporally co-occurring with any others in contrast to previous methods *e.g*. [7]. In our second approach, Self-supervised Siamese Network (SSiam), we obtain hard positive and negative pairs by sorting distances (*i.e*. ranking) on a subset of frames. Thus, SSiam can mine positive and negative pairs without the need for tracking, additionally enabling application of our method to image collections.

We compare our proposed methods against alternatives from *generative* modeling (auto-encoders) as strong baselines. In particular, Variational AutoEncoders (VAEs) [32] can effectively model the distribution of face representations and achieve good generalization performance when working with the same set of characters. We perform extensive empirical studies and demonstrate the effectiveness and generalization of all methods. Our methods are powerful, yet simple, and obtain performance comparable or higher than state-of-the-art when evaluated on three challenging video face clustering datasets.

This paper extends our previous work [1] in several key aspects. (1) We discuss the use of generative models, in particular, Variational Autoencoders, as a strong baseline that can learn the latent identity information by modeling the underlying distribution of face representations. (2) We include an in depth empirical analysis and comparison of TSiam, SSiam, and VAE with comparison of generalization performance across videos with same or different characters. (3) Finally, we include qualitative results to shed light on what the models may have learned as key identity information.

The remainder of this paper is structured as follows: Section 2 provides an overview of related work. In Section 3, we propose TSiam, SSiam, and present a strong generative model as a baseline (VAE) to further refine deep features. Extensive experiments, an ablation study, comparison to the state-of-the-art, and qualitative results are presented in Section 4. We summarize key messages in a discussion (Section 5) and finally conclude in Section 6.

2 Related Work

Over the last decade, several advances have been made in video face clustering through discriminative models that aim to improve representations. In this section, we will review related work in this area, but also discuss some work on generative modeling (specifically VAEs) that may be used to improve face representations.

Generative face models. Along with MNIST handwritten digits [33], faces are a common test bed for many generative models as they are a specific domain of images that can be modeled relatively well. A couple examples include Robust Boltzmann machines [34], and recent advances in Generative Adversarial Networks (GANs) that are able to generate stunning high resolution faces [35].

Variational Autoencoders (VAEs) have also seen growing use in face analysis, especially in generating new face images [36], [37], [38], [39]. In particular, [37] produces face images with desired attributes, while [40] combines VAEs and GANs towards the same goal. [38] replaces the pixel-level reconstruction loss by comparing similarity between deep representations. Recently, VAEs have been used to predict facial action coding [41] and model user reactions to movies [42].

There are some examples of VAEs adopted for clustering, however, video face datasets are not considered. Stacked Autoencoders are used to simultaneously learn the representation and clustering [43], and Gaussian Mixture Models are combined with VAEs for clustering [44], [45]. Perhaps closest to our work, VAEs are used in conjunction with the triplet loss [46] in a supervised way to learn good representations (but not evaluated on faces). We propose to use VAEs as a strong baseline and a different approach of learning feature representations as compared to standard discriminative approaches. To the best of our knowledge, we are the first to use VAEs in an unsupervised way to model and improve deep face representations, resulting in improved clustering performance.

Video face clustering. Clustering faces in videos commonly uses pairwise constraints obtained by analyzing tracks and some form of representation/metric learning. Different approaches can generally be categorized by the source of constraints.

One of the most commonly adopted source is the temporal information provided by face tracks. Face image pairs belonging to the same track are labeled positive (same character), while face images from co-occurring tracks help create negatives (different characters). This strategy has been exploited by learning a metric to obtain cast-specific distances [7] (ULDML); iteratively clustering and associating short sequences based on hidden Markov Random Field (HMRF) [18], [19]; or performing clustering in a sub-space obtained by a weighted block-sparse low-rank representation (WBSLRR) [47]. In addition to pairwise constraints, video editing cues are used in an unsupervised way to merge tracks [48]. Here, track and cluster representations are learned on-the-fly with dense-SIFT Fisher vectors [49]. Recently, the problem of face detection and clustering is considered jointly [9], and a link-based clustering (Erdös-Rényi) based on rank-1 counts verification is adopted. The linking is done by comparing a given frame with a reference frame and learning a threshold to merge/not-merge frames.

Face track clustering/identification methods have also used additional cues such as clothing appearance [50], speech [51], voice models [16], context [52], gender [53], name mentions (first, second, and third person references) in subtitles [54], multispectral information [55], [56], weak labels using transcripts/subtitles [12], [14], and joint action and actor labeling [57] using transcripts.

With the popularity of CNNs, there is a growing focus on improving face representations using video-level constraints. An improved form of triplet loss is used to fine-tune the network and push the positive and negative samples apart in addition to requiring anchor and positive to be close, and anchor and negative far [20]. Zhang *et al.* [10] learn better representations by dynamic clustering constraints that are discovered iteratively during clustering that is performed via a Markov Random Field (MRF). Roethlingshoefer*et al.* [58] use graph neural network to learn representation of face-tracks. In contrast to related work, we propose a simple, yet effective approach (SSiam) to learn good representations by sorting distances on a subset of frames and not requiring video/track level constraints to generate positive/negative training pairs.

Another point of comparison lies in Zhang *et al.* [10], [20] and Datta *et al.* [21] only using video-level constraints to generate a set of similar and dissimilar face pairs. Thus, the model does not see negative pairs for singleton (non co-occurring) tracks. In contrast, our method TSiam incorporates negative pairs for the singleton tracks by exploiting track-level distances.

Recently, advances in clustering approaches themselves have contributed to better performance. Sarfraz *et al.* [59] propose a new clustering algorithm (FINCH) based on first neighbor relations. However, FINCH is not trainable in contrast to our method, and would only benefit further from improved feature representations. In [60], the authors use inverse reinforcement learning on a ground-truth dataset to find a reward function for deciding whether to merge a given pair of facial features. Contrary to these methods, we expect neither the existence of ground-truth data, nor a measure of face quality. Parallel to this work, Tapaswi *et al.* [11] propose to learn an embedding space that creates a fixed-radius ball for each character thus allowing to estimate the number of clusters. However, their work requires supervised labels during training, while our models learn the embedding in a self-supervised setting.

Finally, there are related works that "harvest" training data from unlabeled sources which is in the similar spirit of SSiam and TSiam. Fernando *et al.* [61] and Mishra *et al.* [62] shuffle the video frames and treat them as positive or negative training data for reordering video frames; Wang *et al.* [63] collect positive and negative training data by tracking bounding boxes (*i.e.* motion information) in order to learn effective visual representations. In contrast, we propose new techniques to generate labels and utilize them efficiently to improve video face clustering.

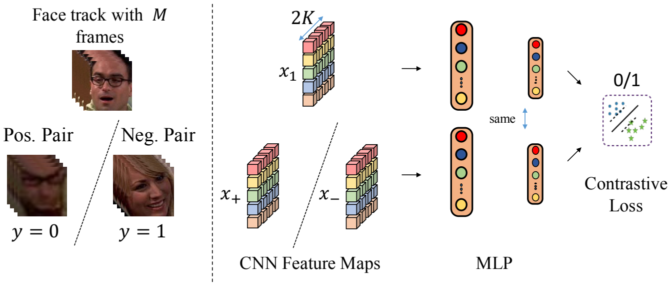


Fig. 1. **Track-supervised Siamese network (TSiam)**. Illustration of the Siamese architecture used in our track-supervised Siamese networks. Note that the MLP is shared across both feature maps. $2K$ corresponds to batch size.

3 Refining Face Representations for Clustering

Our goal is to improve face representations using simple methods that build upon the success of deep CNNs. More precisely, we propose models to refine the face descriptors automatically, without the need for manually curated labels. Note that we do not fine-tune the base CNN, and only learn a few linear layers above it. Our approach has three key benefits: (i) it is easily applicable to new videos (does not require labels); (ii) it does not need large amounts of training data (few hundred tracks are enough); and (iii) specialized networks can be trained to overfit on each episode or film.

We start this section by first introducing the notation used throughout the remainder of the paper. We then propose the discriminative models: (1) Track-supervised Siamese Network (TSiam), and (2) Self-supervised Siamese Network (SSiam) (Section 3.1). Finally, we present how Variational Autoencoders (VAE) can be used to improve representation learning and act as a strong generative model baseline (Section 3.2).

Preliminaries. Consider a video with N face tracks $\{T^1, \ldots, T^N\}$ belonging to C characters. Each track corresponds to one of the characters, and consists of $T^i = \{f_1, \ldots, f_{M^i}\}$ face images. Our goal is to group tracks into sets $\{G_1, \ldots, G_{|C|}\}$ such that each track is assigned to only one group, and ideally, each group contains all tracks from the same character. We use a deep CNN (VGG2 [26]) and extract a descriptor for each face image $\mathbf{x}_k^i \in \mathbb{R}^D$, $k = 1, \ldots, M^i$ from the penultimate layer (before classification) of the network. We refer to these as *base features*, and demonstrate that they already achieve a high performance. As a form of data augmentation, we use 10 crops obtained from an expanded bounding box surrounding the face image during training. Evaluation is based on one center crop.

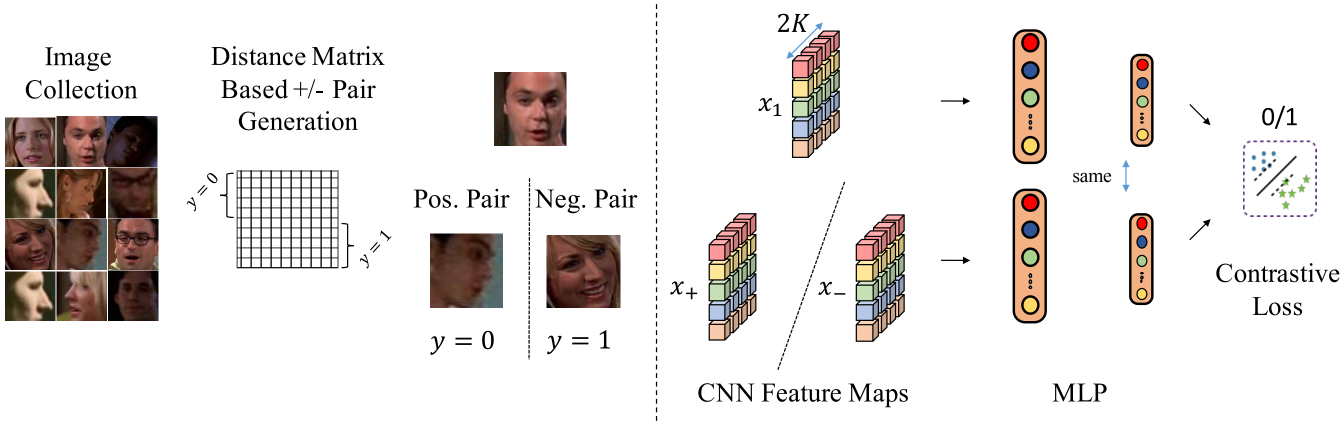


Fig. 2. **Self-supervised Siamese network (SSiam)**. Illustration of the Siamese architecture used in our self-supervised Siamese networks. SSiam selects hard pairs: farthest positives and closest negatives using a ranked list based on Euclidean distance for learning similarity and dissimilarity respectively. Note that the MLP is the same across both feature maps. $2K$ corresponds to batch.

Track-level representations are obtained by aggregating the face image descriptors

$$\mathbf{t}^i = \frac{1}{M^i}\sum_k \mathbf{x}_k^i \,. \tag{1}$$

We additionally normalize track representations to be unit-norm, $\hat{\mathbf{t}}^i = \mathbf{t}^i/\|\mathbf{t}^i\|_2$ before using them for clustering.

Hierarchical Agglomerative Clustering (HAC) has been the clustering method adopted by several previous works [10], [20], [48]. For a fair comparison, we also use HAC to obtain a fixed number of clusters equal to the number of characters (known a priori). We use the minimum variance ward linkage [64] for all methods. See Figure 4 for an illustration.

3.1 Discriminative models

Discriminative clustering models typically associate a binary label y with a pair of features. We designate $y = 0$ when a pair of features $(\mathbf{x}_1, \mathbf{x}_2)$ belong to the same character (identity), and $y = 1$ otherwise [65].

We use a shallow MLP to reduce the dimensionality and improve generalization of the features (see Figure 1, 2). Here, each face image is encoded as $Q_\phi(\mathbf{x}_k^i)$, where ϕ corresponds to the trainable parameters of the MLP. We find $Q_\phi(\cdot)$ to perform best when using a linear layer (for details see Section 4.2). To perform clustering, we compute track-level aggregated features by average pooling across the embedded frame-level representations [66], [67]

$$\mathbf{t}^i = \frac{1}{M^i}\sum_k Q_\phi(\mathbf{x}_k^i)\,, \tag{2}$$

followed by ℓ_2-normalization.

We train our model parameters by minimizing the contrastive loss [65] at the frame-level:

$$\begin{aligned}&\mathcal{L}\left(W, y, Q_\phi(\mathbf{x}_1), Q_\phi(\mathbf{x}_2)\right) = \\ &\quad \frac{1}{2}\left((1-y)\cdot(d_W)^2 + y\cdot(\max(0, m-d_W))^2\right)\,,\end{aligned} \tag{3}$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ are a pair of face representations with $y = 0$ when coming from the same character, and $y = 1$ otherwise. $W : \mathbb{R}^{D\times d}$ is a linear layer that embeds $Q_\phi(\mathbf{x})$ such that $d \ll D$ (in our case, $d = 2$). d_W is the Euclidean distance $d_W = \|W\cdot Q_\phi(\mathbf{x}_1) - W\cdot Q_\phi(\mathbf{x}_2)\|^2$, and m is the margin, empirically chosen to be 1.

In the following, we present two strategies to automatically obtain supervision for pairs of frames: Figure 1 illustrates the Track-level supervision, and Figure 2 shows the Self-supervision for Siamese network training.

3.1.1 Track-supervised Siamese network (TSiam).

Video face clustering often employs face tracking to link face detections made in a series of consecutive frames. The tracking acts as a form of high precision clustering (grouping detections within a shot) and is popularly used to automatically generate positive and negative pairs of face images [7], [19], [21], [48]. In each frame, we assume that characters appear on screen only once. Thus, all face images within a track can be used as positive pairs, while face images from co-occurring tracks are used as negative pairs. For each frame in the track, we sample two frames within the same track to form positive pairs, and sample four frames from a co-occurring track (if it exists) to form negative pairs.

Depending on the filming style of the series/movie, characters may appear alone or together on screen. As we will see through experiments on diverse datasets, some videos have 35% tracks with co-occurring tracks, while this can be as large as 70% for other videos. For isolated tracks, we sort all other tracks in the same video based on track-level distances (computed on base features) and randomly sample frames from the farthest $F = 25$ tracks. Note that all previous works ignore negative pairs for singleton (not co-occurring) tracks. We will highlight their impact in our experiments.

3.1.2 Self-supervised Siamese network (SSiam)

Supervision from tracking may not always be available or may also be unreliable. An example is face clustering within image collections (*e.g.* on social media platforms). To enable the use of metric learning without any supervision we propose an effective approach that can generate the required pairs automatically during training. SSiam is inspired by pseudo-relevance feedback (pseudo-RF) [68], [69] that is commonly used in information retrieval.

We hypothesize that the first and last samples of a ranked list based on Euclidean distance are strong candidates for learning similarity and dissimilarity respectively. We exploit this in a meaningful way and generate promising similar and dissimilar pairs from a representative subset of the data.

Formally, consider a subset $\mathcal{S} = \{\mathbf{x}_1, \ldots, \mathbf{x}_B\}$ of face image representations from the dataset (sampled randomly, not from the same track). We treat each frame $\mathbf{x}_b, b = 1, \ldots, B$ as a query and compute Euclidean distance against every other frame in the set. We sort rows of the resulting matrix in an ascending order (smallest to largest distance) to obtain an ordered index matrix $\mathcal{O}(\mathcal{S}) = [s_1^o; \ldots; s_B^o]$. Each row s_b^o contains an ordered index of the closest to farthest faces corresponding to $\mathbf{x}_b$. Note that the first column of such a matrix is the index b itself at distance 0. The second column corresponds to nearest neighbors for each frame and can be used to form the set of positive pairs $\mathcal{S}_+$. Similarly, the last column corresponds to farthest neighbors and forms the set of negative pairs $\mathcal{S}_-$. Each element of the above sets stores: query index b, nearest/farthest neighbor r, and the Euclidean distance d.

During training, we first form pairs dynamically by picking a random subset of B frames at each iteration. We compute the distances, sort them, and obtain positive and negative pairs sets $\mathcal{S}_+, \mathcal{S}_-$, each with B elements as described above. Among them, we choose K pairs from the positive set that have the largest distances and K pairs from the negative set with the smallest distances. This allows us to select semi-hard positive pairs and semi-hard negative pairs from each representative set of B elements. Finally, these $2K$ pairs are used in a contrastive setting (Eq. 3) to train network parameters.

To encourage variety in the sample set $\mathcal{S}$ and reduce the chance of false positives/negatives in the chosen $2K$ pairs, B is chosen to be much larger than K ($B = 1000, K = 64$). Experiments on several datasets and generalization studies show the benefit and effectiveness of this approach in collecting positive and negative pairs to train the network.

Note that, SSiam can be thought of as an improved version of pseudo-RF with batch processing. Rather than selecting farthest negatives and closest positives for each independent query, we emphasize that SSiam selects $2K$ hard pairs: farthest positives and closest negatives by looking at the batch of queries B jointly. This selection of sorted pairs from the positive $\mathcal{S}_+$ and negative $\mathcal{S}_-$ sets is quite important as will be shown later.

3.2 Generative models

We now present generative models as an alternative strong baseline that can also achieve similar improvements to feature representations. Similar to SSiam, we do not require track-level supervision, in fact, auto-encoders consider single images (and not pairs) at a time.

3.2.1 *Variational Autoencoder (VAE)*

Deep face CNNs are trained to identify and distinguish between people, and are supposed to be invariant to effects of pose, illumination, *etc*. However, in reality, pose, background, and other image-specific characteristics leak into the model, reducing performance. Our goal is to learn a latent variable model that separates identity from other spurious artifacts given a deep representation. We assume that face descriptors are generated by a random process that first involves sampling a continuous latent variable z representing identity of the characters. This is followed by a conditional model $p(\mathbf{x}|z;\theta)$ with some parameters θ, modeled as a neural network (specifically, an MLP)

$$p(\mathbf{x}) = \int p(\mathbf{x}|z;\theta)p(z)dz\,. \tag{4}$$

We propose to adopt a Variational Autoencoder (VAE) [32] consisting of an encoder MLP $Q(z|\mathbf{x};\phi)$ with parameters ϕ and the decoder $P(\mathbf{x}|z;\theta)$. The encoder provides an approximate posterior over the latent variable, and the model parameters are trained to maximize the variational lower bound

$$\mathcal{L}_v = \mathbb{E}_{q(z|\mathbf{x};\phi)}[\log p(\mathbf{x}|z;\theta)] - D_{KL}(q(z|\mathbf{x};\phi)\|p(z))\,. \tag{5}$$

Note that $\log p(\mathbf{x}) \geq \mathcal{L}_v$ and thus maximizing $\mathcal{L}_v$ corresponds to maximizing the log-likelihood of the samples. D_{KL} is the Kullback-Leibler Divergence between the approximate posterior $q(z|\mathbf{x};\phi)$ and the latent variable prior $p(z)$, and acts like a regularization on the distribution of latent variables. The first term corresponds to the log-likelihood of observing $\mathbf{x}$ given z and is a form of reconstruction error. Note that it requires sampling from $q(z|\mathbf{x};\phi)$ which is achieved using the reparameterization trick [32]. In practice, for each input $\mathbf{x}$, the encoder MLP Q_ϕ predicts the latent variable mean μ and a diagonal variance Σ that model a Gaussian prior. We refer the interested reader to [70] for a gentle introduction.

Our encoder is a two-layer MLP Q_ϕ and generates (μ_k^i, Σ_k^i) for each face image representation $\mathbf{x}_k^i$. The decoder is also a two-layer MLP P_θ that takes as input a latent variable sample

$$z_k^i = \mu_k^i + \epsilon {\Sigma_k^i}^{0.5}, \quad \epsilon \sim \mathcal{N}(0, I)\,, \tag{6}$$

and produces a reconstruction $\hat{\mathbf{x}}_k^i = P_\theta(z_k^i)$. Both the reconstructed representation $\hat{\mathbf{x}}_k^i$ and the latent variable predicted mean μ_k^i can be used for clustering. We form two final track representations based on the reconstructed features $\mathbf{t}_{rec}^i = \frac{1}{M^i}\sum_k \hat{\mathbf{x}}_k^i$ and based on the latent means $\mathbf{t}_\mu^i = \frac{1}{M^i}\sum_k \mu_k^i$. Figure 3 illustrates the model.

Note that, we propose VAEs as an alternative method to discriminative approaches, and a strong baseline. We show in our experiments that discriminative methods TSiam and SSiam, often perform equally or better than VAEs.

4 Evaluation

We present our evaluation on three challenging datasets. We first describe the clustering metric, followed by a thorough analysis of the proposed methods, ending with a comparison to state-of-the-art.

4.1 Experimental Setup

Datasets. We conduct experiments on three challenging video face identification/clustering datasets: (i) *Buffy the Vampire Slayer* (BF) [10], [14] (season 5, episodes 1 to 6): a

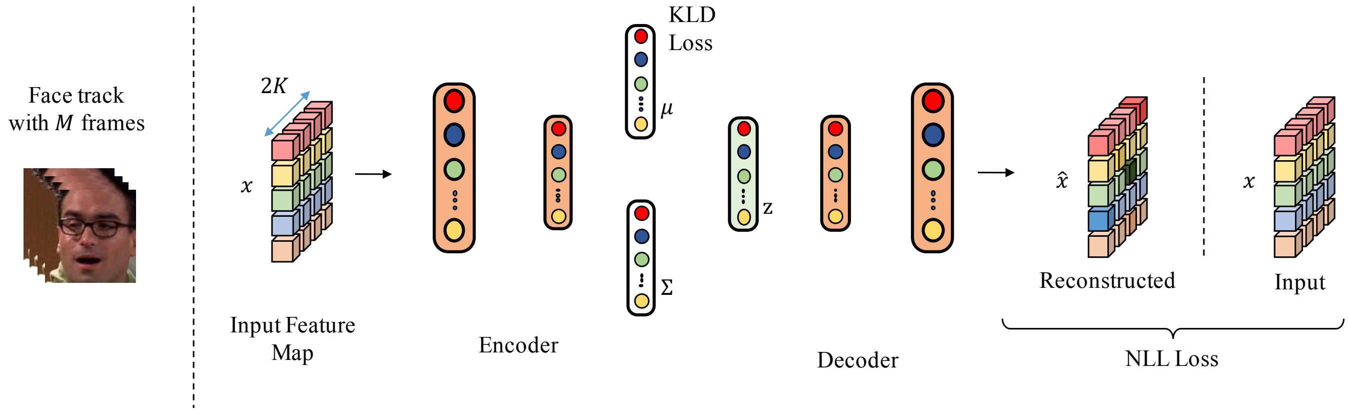


Fig. 3. Illustration of a Variational Autoencoder used as a strong baseline generative model. In contrast to the Siamese networks, the VAE sees single frames (not pairs) and is trained by two losses: KL-Divergence and the Reconstruction NLL. $2K$ corresponds to batch.

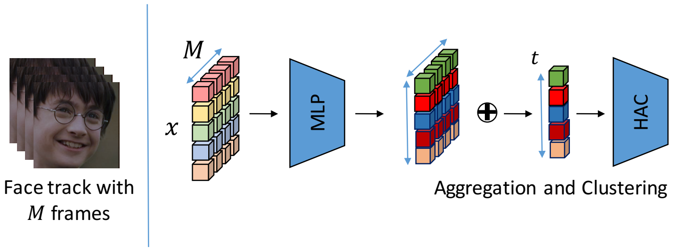


Fig. 4. **Illustration of the test time evaluation scheme**. Given our pre-trained MLPs, TSiam or SSiam, we extract the frame-level features for the track, followed by mean pooling to obtain a track-level representation. All such track representations from the video are grouped using HAC to obtain a known number of clusters.

TABLE 1
Dataset statistics for the most commonly used episode of BBT, BF and the first movie from the ACCIO series.

		This work		Previous work
Datasets	#Cast	#TR (#FR)	LC/SC (%)	#TR (#FR)
BBT0101	5	644 (41220)	37.2 / 4.1	182 (11525)
BF0502	6	568 (39263)	36.2 / 5.0	229 (17337)
ACCIO	36	3243 (166885)	30.93/0.05	3243 (166885)

drama series with several shots in the dark at night; (ii) *Big Bang Theory* (BBT) [14], [18], [20], [50] (season 1, episodes 1 to 6): a sitcom with small cast list shot mainly indoors, and (iii) *ACCIO* [71]: *Accio-1* first installment of *"Harry Potter"* movie series with a large number of dark scenes and several tracks with non-frontal faces.

Most previous works [7], [10], [19], [20] on video-face clustering assume the number of main characters/clusters is known. We follow the same protocols that are widely employed in the previous works [10], [20], [66] and train on a single episode of BBT (episode 1), BF (episode 2), and the first movie from the ACCIO series. We also use the same number of characters as previous methods [10], [20], however, it is important to note that we do not discard tracks/faces that are small or have large pose variation. When not mentioned otherwise, we use an updated version of face tracks released by [14] that incorporate several detectors to encompass all pan angles and in-plane rotations up to 45 degrees. Tracks are created via an online tracking-by-detection scheme with a particle filter.

We present a summary of the dataset used in this work in Table 1, and also indicate the number of tracks (#TR) and frames (#FR) used in other works, showing that our data is indeed more challenging. Additionally, it is important to note that different characters have wide variations in number of tracks, indicated by the cluster skew between largest class (LC) to smallest class (SC). Figure 5 shows a few examples of difficult faces included in our dataset.

Evaluation metric. We use Clustering Accuracy (ACC) [10] also called Weighted Clustering Purity (WCP) [48] as the metric to evaluate the quality of clustering. As we compare methods that generate equal numbers of clusters (number of main cast), ACC is a fair metric for comparison.

$$\text{ACC} = \frac{1}{N} \sum_{c=1}^{|C|} n_c \cdot p_c \,, \tag{7}$$

where N is the total number of tracks in the video, n_c is the number of samples in the cluster c, and cluster purity p_c is measured as the fraction of the largest number of samples from the same label to n_c. $|C|$ corresponds to the number of main cast members, and in our case also the number of clusters.

In addition to ACC, for ACCIO, we report BCubed Precision (P), Recall (R) and F-measure (F) used in previous work.

4.2 Implementation Details

Figure 4 illustrates the network architecture during test time.

CNN. We adopt the VGG-2 face CNN [26], a ResNet50 model, pre-trained on MS-Celeb-1M [72] and fine-tuned on 3.31M face images of 9131 subjects (VGG2 data). Input RGB face images are resized to 224 × 224, and pushed through the CNN. We extract `pool5_7x7_s1` features, resulting in $\mathbf{x}_k^i \in \mathbb{R}^{2048}$.

Siamese network MLP. The network comprises of fully-connected layers ($\mathbb{R}^{2048} \rightarrow \mathbb{R}^{256} \rightarrow \mathbb{R}^{2}$). Note that the second linear layer is part of the contrastive loss (corresponds to W in Eq. 3), and we use the feature representations at $\mathbb{R}^{256}$ for clustering.

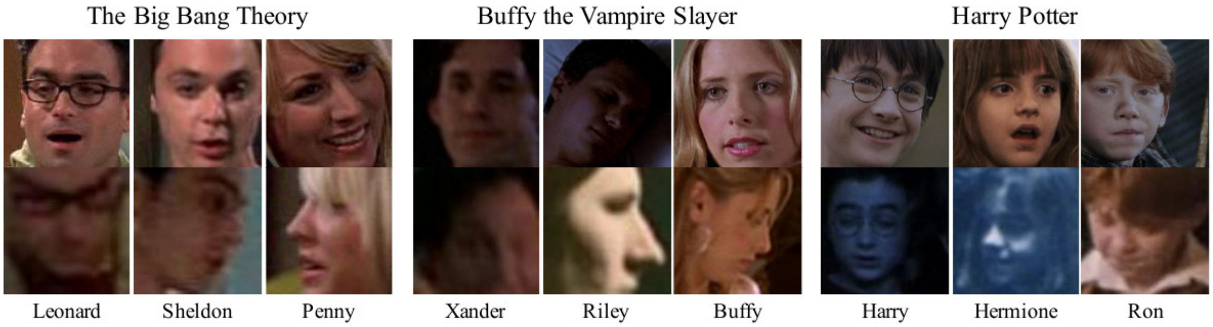


Fig. 5. Example images for a few characters from our dataset. We show one easy view and one difficult view. The extreme variation in illumination, pose, resolution, and attributes (spectacles) make the datasets challenging.

We train our Siamese network with track-level supervision (TSiam) with about 102k positive and 204k negative frame pairs (for BBT-0101) by mining 2 positive and 4 negative pairs for each frame. For the Self-supervised Siamese network (SSiam), we generate batches of size $B = 1000$, and select $K = 64$ positive and negative pairs each. Higher batch sizes $B = 2000, 3000$, did not provide significant improvements.

The MLP is trained using the contrastive loss, and parameters are updated using Stochastic Gradient Descent (SGD) with a fixed learning rate of 10^{-3}. Since the labels are obtained automatically for each video, overfitting is not a concern. We train our model until convergence (loss does not reduce significantly any further).

VAE. Our VAE uses a two-layer MLP encoder ($\mathbb{R}^{2048} \rightarrow \mathbb{R}^{1024} \rightarrow \mathbb{R}^{256\times2}$, μ and diagonal co-variance Σ); and a two-layer MLP decoder ($\mathbb{R}^{256} \rightarrow \mathbb{R}^{1024} \rightarrow \mathbb{R}^{2048}$). The VAE is trained using SGD, with a learning rate of 10^{-3} until convergence.

4.3 Clustering Performance Ablation Studies

TABLE 2
Clustering accuracy on the base face representations.

Dataset	Track Level VGG1	Track Level VGG2	Frame Level VGG1	Frame Level VGG2
BBT-0101	0.916	**0.932**	0.938	**0.940**
BF-0502	0.831	**0.836**	0.901	**0.912**

4.3.1 Base features

We begin our analysis by comparing track- and frame-level performance of two commonly used CNNs to obtain face representations: VGG1 [25] and VGG2 [26]. Track-level results use mean-pool of frames. Results are reported in Table 2. Note that the differences between VGG1 and VGG2 are typically within 1% of each other indicating that the results in the subsequent experiments are not just due to having better CNNs trained with more data. We refer to VGG2 features as Base for the remainder of this paper.

4.3.2 Role of effective mining of +/- pairs

We emphasize that especially in light of CNN face representations, the features are very similar even across different identities, and thus positive and negative pairs need to be created carefully to gain the most improvements. Figure 6 proves this point as (i) we see a large overlap between the cosine similarity distributions of positive (same identity) and negative (different identities) track pairs on the base features; and (ii) note the scale on the x-axis, even negative pairs have cosine similarity scores higher than 0.9.

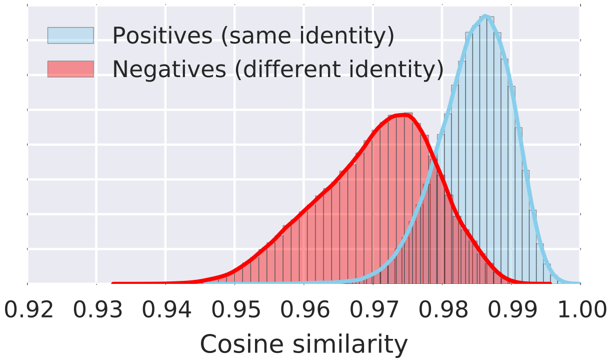


Fig. 6. Histograms of pairwise cosine similarity between tracks of same identity (positive, blue) and different identity (negative, red) for BBT-0101. Best seen in color.

4.3.3 TSiam, impact of singleton tracks

Previous work with video-level constraints [10], [20] and [21], ignore singleton (not co-occurring) tracks. In TSiam, we include negative pairs for singletons based on track distances. Table 3 shows that 50-70% tracks are singleton and ignoring them lowers accuracy by 3-4%. This confirms our hypothesis that incorporating negative pairs of singletons helps improve performance. We believe that characters with minor roles benefit most as otherwise, they are likely to have few training pairs.

TABLE 3
Ignoring singleton tracks (and possibly characters) leads to significant performance drop. Accuracy on track-level clustering.

Dataset	TSiam w/o Single [21]	TSiam Ours	# Tracks Total	# Tracks Single	# Tracks Co-oc
BBT-0101	0.936	**0.964**	644	331	313
BF-0502	0.849	**0.893**	568	395	173

4.3.4 SSiam and Pseudo-Relevance Feedback

In Pseudo-RF [68], [69], all samples are treated independent of each other, there is no batch of data B from which $2K$ pairs are chosen. A pair of samples closest in distance are chosen as positive, and farthest as negative. However, this usually corresponds to samples that already satisfy the loss margin, thus leading to small (possibly even 0) gradient updates. Table 4 shows that SSiam that involves sorting a batch of queries is much more effective than pseudo-RF as it has the potential to select harder positives and negatives. We see a consistent gain in performance, 3% for BBT-0101 and over 9% for BF-0502.

TABLE 4
Comparison between *SSiam* and *pseudo-RF*.

Method	BBT-0101	BF-0502
Pseudo-RF	0.930	0.814
SSiam	**0.962**	**0.909**

4.4 Studying Generalization

Please note that generalization experiments are presented here to explore the underlying properties of our discriminative and generative models. If achieving high performance is the only goal, we assert that our models can be trained and evaluated on each video rapidly and fully automatically.

4.4.1 Performance on training videos

We report clustering performance on training videos in Table 5. Note that all our models are trained in an unsupervised manner, or with automatically generated labels. We additionally report results for an AutoEncoder (AE) with the same network architecture as the VAE, but without the variational space and sampling. The AE is trained using NLL loss for reconstruction.

We observe that VAE and SSiam show large performance boost over the base VGG2 features on BBT and BF. In particular, VAE shows large improvement on videos with few characters (BBT). With training and evaluation on the same video, the generative models demonstrate comparable performance to discriminative models.

TABLE 5
Clustering accuracy computed at track-level on the training episodes, with a comparison to all evaluated models.

Train/Test	Base	TSiam	SSiam	AE	VAE
BBT-0101	0.932	0.964	0.962	0.967	**0.984**
BF-0502	0.836	0.893	**0.909**	0.842	0.889

4.4.2 Generalization within series

In this experiment, we evaluate the generalization capability of our models. We train on one episode each, BBT-0101 and BF-0502, and evaluate on all other episodes of the same TV series. Table 6 reports averaged clustering accuracy over the remaining 5 episodes for each series. Both SSiam or TSiam perform similar (slightly lower/higher) to the base features, possibly due to overfitting. VAE performs better here in comparison to the discriminative models. This indicates that VAEs are able to model the distribution of face representations by extracting the latent character identity that is common across the episodes.

TABLE 6
Clustering accuracy computed at track-level across episodes within the same TV series. Numbers are averaged across 5 test episodes.

Train	Test	Base	TSiam	SSiam	AE	VAE
BBT-0101	BBT-01[02-06]	0.935	0.930	0.914	0.917	**0.945**
BF-0502	BF-05[01,03-06]	0.892	0.889	0.904	0.899	**0.908**

4.4.3 Generalization across series

We further analyze our models by evaluating generalization across series. Based on Table 7, we bring the readers attention towards three key observations:

1) TSiam and SSiam retain their discriminative power and can transfer to other series more gently. As they learn to score similarity between pairs of faces, the underlying distribution of identities does not matter much. For example, the drop when training TSiam on BBT-0101 and evaluating on BF is 0.890 (train on BF-0502) to 0.875.
2) As filming styles differ, underlying distributions of the face identities can be quite different. VAEs are unable to cope with this shift, and show drop in performance. Training on BBT-0101 and evaluating on BF reduces performance from 0.905 (train on BF-0502) to 0.831.
3) We clearly see that the similarity between videos can affect generative models. For example, BBT is quite similar with mostly bright scenes during the day. BF on the other hand has almost half the scenes at night causing large variations.

TABLE 7
Clustering accuracy when evaluating across video series. Each row indicates that the model was trained on one episode of BBT / BF, but evaluated on all 6 episodes of the two series.

	Train	Test series	
	Episode	BBT-01[01-06]	BF-05[01-06]
TSiam	BBT-0101	0.936	0.875
	BF-0502	0.915	0.890
SSiam	BBT-0101	0.922	0.862
	BF-0502	0.883	0.905
VAE	BBT-0101	0.952	0.831
	BF-0502	0.830	0.905

4.4.4 Generalization to unseen characters

In the ideal setting, we would like to cluster all characters appearing in an episode including (main, other named, unknown, and background). However, this is a very difficult setting, and in fact, disambiguating background characters is even hard for humans and there are no datasets that include such labels. For BBT and BF, we do however have all named characters labeled. Firstly, expanding the clustering experiment to include them drastically changes the class balance. For example, BF-0502 has 6 main and 12 secondary characters with class balance shifting from 36.2/5.0 to 40.8/0.1 (lowest to highest cluster membership in percentage).

We present clustering accuracy for this setting in Table 8. All proposed methods show a drop in performance when extending to unseen characters. Note that the models have been trained on only the main characters data and tested on all (including unseen) characters. However, the drop is small when adding just 1 new character (BBT-0101) vs. introduction of 6 in BF-0502.

SSiam's performance generalizes gracefully, probably since it is trained with a diverse set of pairs (dynamically generated during training) and can generalize to unseen characters.

TABLE 8
Clustering accuracy when extending to all named characters within the episode. BBT-0101 has 5 main and 6 named characters. BF-0502 has 6 main and 12 named characters.

	BBT-0101			BF-0502		
	TSiam	SSiam	VAE	TSiam	SSiam	VAE
Main cast	0.964	0.962	0.984	0.893	0.909	0.889
All named cast	0.958	0.922	0.978	0.829	0.870	0.807

4.4.5 Number of clusters when purity = 1

Table 9 shows the number of clusters we can achieve while maintaining purity to be 1. In a similar spirit to [48], this metric indicates when the first mistake in agglomerative merging occurs – smaller the number, the better it is. SSiam works best on the harder BF dataset, while VAE can reduce the clusters most on BBT.

TABLE 9
Similar to [48], we evaluate the number of clusters we can reach when maintaining clustering accuracy/purity at 1. Lower is better.

Video	#Tracks	Base	TSiam	SSiam	VAE	Ideal
BBT-0101	644	365	369	389	**245**	5
BF-0502	568	460	490	**253**	312	6

4.4.6 Generalization to joint training

For this evaluation, we train a model combining BBT-0101, BF-0502, NH [1]. We report results in Table 10. The drop in performance is expected, however, note that unsupervised overfitting to each episode is not necessarily bad. Interestingly, VAE retains performance on BF-0502, we suspect this may be due to more visual variation in BF. Our discriminative methods do perform well when trained and evaluated on larger datasets (see Table 12), while VAE suffers due to large number of characters and a high skew in cluster ratios.

We show generalization studies to better understand our methods. VAEs seem to transfer well to within domain (same characters), while discriminative TSiam and SSiam transfer well across TV series. However, training on each episode should yield best performance for all methods.

TABLE 10
Impact of training on combined dataset of BBT-0101, BF-0502, and NH.

	Train	TSiam	SSiam	VAE
BBT-0101	BBT-0101	**0.964**	**0.962**	**0.984**
	BBT+BF+NH	0.930	0.930	0.938
BF-0502	BF-0502	**0.893**	**0.909**	0.889
	BBT+BF+NH	0.852	0.887	**0.890**

4.5 Comparison with the state-of-the-art

BBT and BF. We compare our proposed methods (TSiam, and SSiam) with the state-of-the-art approaches in Table 11. We report clustering accuracy (%) on two videos: BBT-0101 and BF-0502. Historically, previous works have reported performance at a frame-level. We follow this for TSiam and SSiam.

Note that our evaluation uses 2-4 times larger number of frames than previous works [10], [20] making direct comparison hard. Specifically in BBT-0101 we have 41,220 frames while [20] uses 11,525 frames. Similarly, we use 39,263 frames for BF-0502 (vs. 17,337 [10]). Even though we cluster more frames and tracks (with more visual diversity), our approaches are comparable to or even better than the current results.

TSiam, SSiam and VAE are all better than the improved triplet method [20] on BBT-0101. SSiam obtains 99.04% accuracy which is 3.04% higher, and VAE obtains 2.4% better performance (absolute gains). On BF-0502, TSiam performs the best with 92.46% which is 0.33% better than the JFAC [10].

ACCIO. We evaluate our methods on ACCIO dataset with 36 named characters, 3243 tracks, and 166885 faces. The largest to smallest cluster ratios are very skewed: 30.65% and 0.06%. In fact, half the characters correspond to less than 10% of all tracks. Table 12 presents the results when performing clustering to yield 36 clusters (equivalent to the number of characters). In addition, as in [10], Table 13 (num. clusters = 40) shows that our discriminative methods are not affected much by this skew, and in fact improve performance by a significant margin over the state-of-the-art.

Computational complexity. Our models essentially consist of a few linear layers and are very fast to compute at inference time. In fact, training the SSiam for about 15 epochs on BBT-0101 requires less than 25 minutes (on a GTX 1080 GPU using the matconvnet framework [74]).

[1] *Notting Hill* (NH) [10], [19]: a romantic comedy movie. NH has 5 main casts with 240 tracks, and 16872 frames. The LC/SC (%) is 43.1/7.4. Tracks for NH are provided by [19].

TABLE 11
Comparison to state-of-the-art. Metric is clustering accuracy (%) evaluated at frame level. Please note that many previous works use fewer tracks (# of frames) (also indicated in Table 1) making the task relatively easier. We use an updated version of face tracks provided by [14].

Method	BBT-0101	BF-0502	Data Source BBT	BF
ULDML (ICCV '11) [7]	57.00	41.62	–	[7]
HMRF (CVPR '13) [19]	59.61	50.30	[73]	[12]
HMRF2 (ICCV '13) [18]	66.77	–	[73]	–
WBSLRR (ECCV '14) [47]	72.00	62.76	–	[12]
VDF (CVPR '17) [66]	89.62	87.46	[14]	[14]
Imp-Triplet (PacRim '16) [20]	96.00	–	[73]	–
JFAC (ECCV '16) [10]	–	92.13	–	[12]
Ours (with HAC)				
TSiam	**98.58**	**92.46**	[14]*	[14]*
SSiam	**99.04**	90.87		
VAE	**98.40**	85.30		

TABLE 12
Performance comparison of TSiam and SSiam with JFAC [10] on ACCIO.

Methods	#cluster=36 P	R	F
JFAC (ECCV '16) [10]	0.690	0.350	0.460
Ours (with HAC)			
TSiam	**0.749**	**0.382**	**0.506**
SSiam	**0.766**	**0.386**	**0.514**
VAE	**0.710**	0.325	0.446

4.6 Qualitative Results

Here we show an exploration of VAE and SSiam (feature maps) with intuitive visualization in the image space. Figure 7 and Figure 8 show these visualizations for a couple tracks from BBT and BF respectively.

We visualize the feature space for **VAE** as follows:

1) Given a face track T with input faces f_t, we randomly sample 8 frames for visualization $t = 1 \dots 8$. The VGG2 features $\mathbf{x}_t$ for f_t are encoded by the VAE-encoder (Q_ϕ) to obtain μ_t, and we then reconstruct μ_t through the VAE-decoder (P_θ) to get $\hat{\mathbf{x}}_t$.
2) Then, for each $\hat{\mathbf{x}}_t$, we find the closest neighbor within $\mathbf{x}_t$ and corresponding face image $\hat{f}_t$.
3) We plot the images $\{f_0, f_t, \dots, f_T\}$ (top row) and $\{\hat{f}_0, \hat{f}_t, \dots, \hat{f}_T\}$ as the nearest neighbor faces for VAE (middle row).

We visualize **SSiam** in a similar way:

1) Given a face track T with input faces f_t, we randomly sample 8 frames $t = 1 \dots 8$. The VGG2 features $\mathbf{x}_t$ for f_t are fed through the MLP to obtain final SSiam representation $Q_\phi(\mathbf{x}_t)$.

TABLE 13
Performance comparison of different methods on the ACCIO dataset.

Methods	# clusters=40 P	R	F
DIFFRAC-DeepID2$^+$ (ICCV '11) [10]	0.557	0.213	0.301
WBSLRR-DeepID2$^+$ (ECCV '14) [10]	0.502	0.206	0.292
HMRF-DeepID2$^+$ (CVPR '13) [10]	0.599	0.230	0.332
JFAC (ECCV '16) [10]	0.711	0.352	0.471
Ours (with HAC)			
TSiam	**0.763**	**0.362**	**0.491**
SSiam	**0.777**	**0.371**	**0.502**
VAE	**0.718**	0.305	0.428

2) Different to VAE, in SSiam for each $Q_\phi(\mathbf{x}_t)$, we find the closest neighbor within the feature space and corresponding face image $\hat{f}_t$.
3) We plot the original images $\{f_0, f_t, \dots, f_T\}$ in the top row, and nearest neighbors in SSiam space $\{\hat{f}_0, \hat{f}_t, \dots, \hat{f}_T\}$ on the bottom row.

Earlier, we have shown that both discriminative (SSiam) and generative (VAE) models can achieve good performance. Here, in Figure 7 and Figure 8, we attempt to understand the underlying feature space that VAEs are able to learn the identity information, and thus show its invariance to the effect of pose and illumination. Interestingly, we see that as SSiam is trained with a discriminative loss the f and $\hat{f}$ are almost identical. On the other hand, VAEs attempt to learn an *average* face while trying to capture identity information.

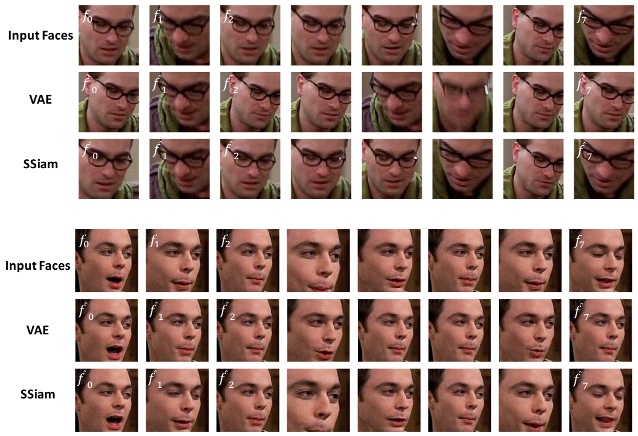


Fig. 7. Visualisation results of BBT-0101 in the image space for VAE and SSiam, where Input Faces are the original faces from the given track.

5 Discussion

Comparison of TSiam/SSiam to VAE. In Table 14, we report the frame-level clustering performance of both discriminative (*i.e.* TSiam and SSiam) and generative (VAE) methods. We observe that TSiam and SSiam generally perform better than base features and are more consistent. In addition, SSiam is generally better than TSiam. VAEs exhibit an unclear fluctuating behavior, and in fact, have

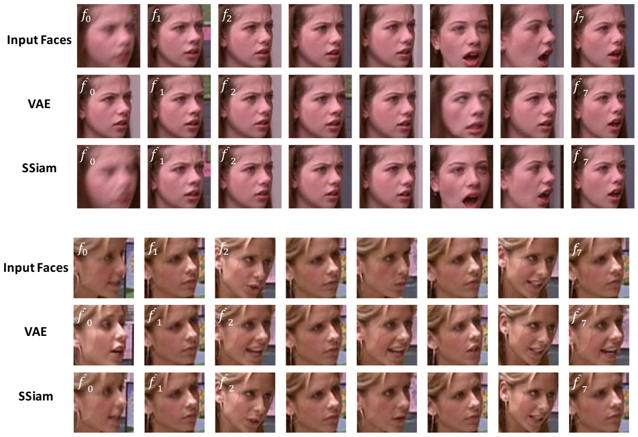


Fig. 8. Visualisation results of BF-0502 in the image space for VAE and SSiam, where Input Faces are the original faces from the given track.

TABLE 14
Accuracy (%) performance comparison of TSiam, SSiam, and VAE over all three dataset BBT-0101, BF-0502 and ACCIO

	#Cast	Base	TSiam	SSiam	VAE
BBT-0101	5	94.00	98.58	**99.04**	98.40
BF-0502	6	91.20	**92.46**	90.87	85.30
ACCIO	36	79.90	81.30	**82.00**	76.44

worse performance than base features on harder datasets (BF, ACCIO).

Feature representations are a crucial component of face clustering in videos. If the representation is robust, we can expect that the face tracks of each identity will be merged together in a unique cluster. Therefore, we recommend use of self-supervised discriminative methods: TSiam and SSiam, over unsupervised generative methods method: VAE.

In Figure 9 and Figure 10, we show the distribution of pairwise cosine similarities between tracks of same identity and different identity for the base features, TSiam and SSiam. We can observe that TSiam makes positive pairs have a very strong peak due to track-level supervision. At the same time SSiam (BF-0502) is hard to interpret using the histogram of similarity between tracks - this maybe due to absence of any supervision.

Do constraints help to merge tracks in face clustering? Conventional techniques for face clustering use handcrafted features that are not very effective in the presence of illumination, and viewpoint variations. In this setting, *must-link* and *must-not-link* pairwise constraints are useful. However, when the feature representation is trained in a discriminative manner, one can obtain a similar or even better clustering performance without using these constraints. We hypothesize that any modeling performed on a powerful representation is complimentary to using such constraints, and hence leads to a better face grouping. We have shown, under such a setting, the face representation can be readily used with simple offline features and learning an efficient method to model additional constraints is meaningful.

An important consideration in clustering is to automatically infer the number of clusters along with the clustering. As numbers saturate, we hope the community moves towards this challenging problem [11]. We are working towards methods that can learn and infer an optimal number of clusters while effectively learning representations. Our work is a hint towards achieving this goal without relying on explicit external constraints as the feature representations are discriminative enough to learn the data grouping with relaxed thresholds.

6 Conclusion

We proposed simple, self-supervised approaches for face clustering in videos, by distilling the identity factor from deep face representations. We showed that discriminative models can leverage dynamic generation of positive/negative constraints based on ordered face distances and do not have to only rely on track-level information that is typically used. We also presented Variational Autoencoder as a strong generative model that can learn the underlying distribution of face representations, and model identity as the latent variable. Our proposed models are unsupervised (or use automatically generated labels) and can be trained and evaluated efficiently as they involve only a few matrix multiplications.

We conducted experiments on three challenging video datasets, comparing their differences in usage in past works. We observed that VAEs are able to generalize well when they have seen the set of characters (*e.g.* across episodes of a series), while discriminative models performed better when generalizing to new series. Overall, our models are fast to train and evaluate and outperform the state-of-the-art while operating on datasets that contain more tracks with higher diversity in appearance.

Acknowledgments

This work is supported by the DFG (German Research Foundation) funded PLUMCOT project.

References

[1] V. Sharma, M. Tapaswi, M. S. Sarfraz, and R. Stiefelhagen, "Self-supervised learning of face representations for video face clustering," in *International Conference on Automatic Face and Gesture Recognition (FG)*, 2019.

[2] A. Rohrbach, A. Torabi, M. Rohrbach, N. Tandon, C. Pal, H. Larochelle, A. Courville, and B. Schiele, "Movie Description," *International Journal of Computer Vision (IJCV)*, 2017.

[3] M. Tapaswi, Y. Zhu, R. Stiefelhagen, R. Urtasun, and S. Fidler, "MovieQA: Understanding Stories in Movies through Question-Answering," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

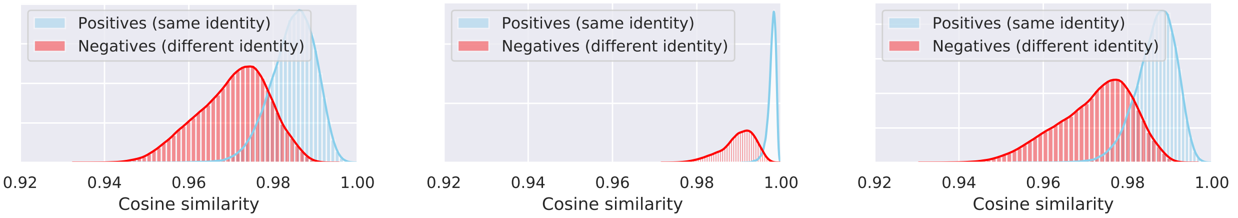
[4] P. Vicol, M. Tapaswi, L. Castrejon, and S. Fidler, "MovieGraphs: Towards Understanding Human-Centric Situations from Videos," *arXiv:1712.06761*, 2017.

[5] H. Zhou, M. Tapaswi, and S. Fidler, "Now You Shake Me: Towards Automatic 4D Cinema," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[6] A. Rohrbach, M. Rohrbach, S. Tang, S. J. Oh, and B. Schiele, "Generating Descriptions with Grounded and Co-Referenced People," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[7] R. G. Cinbis, J. Verbeek, and C. Schmid, "Unsupervised Metric Learning for Face Identification in TV Video," in *International Conference on Computer Vision (ICCV)*, 2011.

[8] M. Guillaumin, J. Verbeek, and C. Schmid, "Is that you? Metric Learning Approaches for Face Identification," in *International Conference on Computer Vision (ICCV)*, 2009.

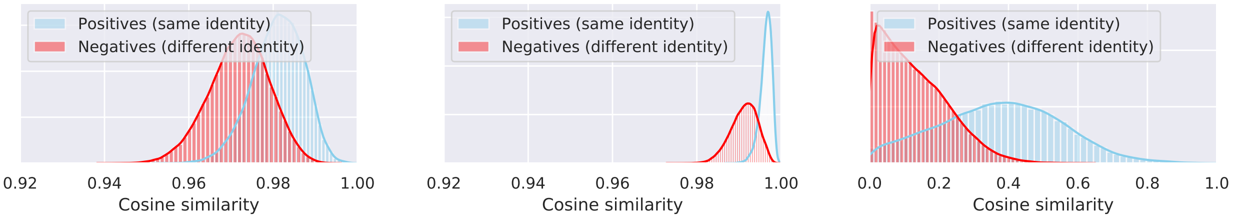


(a) BBT-0101 - Base

(b) BBT-0101 - TSiam

(c) BBT-0101 - SSiam

Fig. 9. Histograms of pairwise cosine similarity between tracks of same identity (pos) and different identity (neg) for BBT-0101.



(a) BF-0502 - Base

(b) BF-0502 - TSiam

(c) BF-0502 - SSiam

Fig. 10. Histograms of pairwise cosine similarity between tracks of same identity (pos) and different identity (neg) for BF-0502.

[9] S. Jin, H. Su, C. Stauffer, and E. Learned-Miller, “End-to-end Face Detection and Cast Grouping in Movies using Erds-Rnyi Clustering,” in *International Conference on Computer Vision (ICCV)*, 2017.

[10] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, “Joint Face Representation Adaptation and Clustering in Videos,” in *European Conference on Computer Vision (ECCV)*, 2016.

[11] M. Tapaswi, M. T. Law, and S. Fidler, “Video face clustering with unknown number of clusters,” in *International Conference on Computer Vision (ICCV)*, 2019.

[12] M. Everingham, J. Sivic, and A. Zisserman, ““Hello! My name is ... Buffy” Automatic Naming of Characters in TV Video,” in *British Machine Vision Conference (BMVC)*, 2006.

[13] J. Sivic, M. Everingham, and A. Zisserman, ““Who are you?” – Learning person specific classifiers from video,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[14] M. Bäuml, M. Tapaswi, and R. Stiefelhagen, “Semi-supervised Learning with Constraints for Person Identification in Multimedia Data,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[15] V. Ramanathan, A. Joulin, P. Liang, and L. Fei-Fei, “Linking people in videos with “their” names using coreference resolution,” in *European Conference on Computer Vision (ECCV)*, 2014.

[16] A. Nagrani and A. Zisserman, “From Benedict Cumberbatch to Sherlock Holmes: Character Identification in TV series without a Script,” in *British Machine Vision Conference (BMVC)*, 2017.

[17] R. Aljundi, P. Chakravarty, and T. Tuytelaars, “Who’s that Actor? Automatic Labelling of Actors in TV series starting from IMDB Images,” in *Asian Conference on Computer Vision (ACCV)*, 2016.

[18] B. Wu, S. Lyu, B.-G. Hu, and Q. Ji, “Simultaneous Clustering and Tracklet Linking for Multi-face Tracking in Videos,” in *International Conference on Computer Vision (ICCV)*, 2013.

[19] B. Wu, Y. Zhang, B.-G. Hu, and Q. Ji, “Constrained Clustering and its Application to Face Clustering in Videos,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[20] S. Zhang, Y. Gong, and J. Wang, “Deep Metric Learning with Improved Triplet Loss for Face Clustering in Videos,” in *Pacific Rim Conference on Multimedia*, 2016.

[21] S. Datta, G. Sharma, and C. V. Jawahar, “Unsupervised learning of face representations,” in *International Conference on Automatic Face and Gesture Recognition (FG)*, 2018.

[22] P. Hu and D. Ramanan, “Finding Tiny Faces,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[23] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “DeepFace: Closing the Gap to Human-Level Performance in Face Verification,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[24] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A Unified Embedding for Face Recognition and Clustering,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[25] O. M. Parkhi, A. Vedaldi, and A. Zisserman, “Deep Face Recognition,” in *British Machine Vision Conference (BMVC)*, 2015.

[26] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, “VGGFace2: A Dataset for Recognising Faces across Pose and Age,” in *International Conference on Automatic Face and Gesture Recognition (FG)*, 2018.

[27] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, “Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments,” in *ECCV Workshop on Faces in Real-life Images*, 2008.

[28] L. Wolf, T. Hassner, and I. Maoz, “Face Recognition in Unconstrained Videos with Matched Background Similarity,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[29] V. Sharma, A. Diba, D. Neven, M. S. Brown, L. Van Gool, and R. Stiefelhagen, “Classification driven dynamic image enhancement,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[30] M. S. Sarfraz, A. Schumann, A. Eberle, and R. Stiefelhagen, “A pose-sensitive embedding for person re-identification with expanded cross neighborhood re-ranking,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[31] A. Diba, V. Sharma, and L. Van Gool, “Deep temporal linear encoding networks,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[32] D. Kingma and M. Welling, “Auto-encoding Variational Bayes,” in *International Conference on Learning Representations (ICLR)*, 2014.

[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based Learning Applied to Document Recognition,” *Proceedings of IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[34] Y. Tang, R. Salakhutdinov, and G. Hinton, “Robust Boltzmann Machines for Recognition and Denoising,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[35] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive Growing of GANs for Improved Quality, Stability, and Variation,” in *International Conference on Learning Representations (ICLR)*, 2018.

[36] “Morphing faces.” [Online]. Available: `https://vdumoulin.github.io/morphing_faces/`

[37] X. Yan, J. Yang, K. Sohn, and H. Lee, “Attribute2Image: Conditional Image Generation from Visual Attributes,” in *European Conference on Computer Vision (ECCV)*, 2016.

[38] X. Hou, L. Shen, K. Sun, and G. Qiu, "Deep Feature Consistent Variational Autoencoder," in *Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[39] N. Siddharth, B. Paige, J.-W. van de Meent, A. Desmaison, N. D. Goodman, P. Kohli, F. Wood, and P. Torr, "Learning Disentangled Representations with Semi-Supervised Deep Generative Models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[40] A. B. Lindbo Larsen, S. K. Snderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," *arXiv:1512.09300*, 2015.

[41] D. L. Tran, R. Walecki, O. Rudovic, S. Eleftheriadis, B. Schuller, and M. Pantic, "DeepCoder: Semi-parametric Variational Autoencoders for Automatic Facial Action Coding," in *International Conference on Computer Vision (ICCV)*, 2017.

[42] Z. Deng, R. Navarathna, P. Carr, S. Mandt, Y. Yue, I. Matthews, and G. Mori, "Factorized Variational Autoencoders for Modeling Audience Reactions to Movies," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[43] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised Deep Embedding for Clustering Analysis," in *International Conference on Machine Learning (ICML)*, 2016.

[44] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering," in *IJCAI*, 2017.

[45] N. Dilokthanakul, P. A. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, "Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders," *arXiv:1611.02648*, 2016.

[46] H. Ishfaq, A. Hoogi, and D. Rubin, "TVAE: Deep Metric Learning Approach for Variational Autoencoder," in *ICLR Workshop*, 2018.

[47] S. Xiao, M. Tan, and D. Xu, "Weighted Block-sparse Low Rank Representation for Face Clustering in Videos," in *European Conference on Computer Vision (ECCV)*, 2014.

[48] M. Tapaswi, O. M. Parkhi, E. Rahtu, E. Sommerlade, R. Stiefelhagen, and A. Zisserman, "Total Cluster: A Person Agnostic Clustering Method for Broadcast Videos," in *ICVGIP*, 2014.

[49] O. M. Parkhi, K. Simonyan, A. Vedaldi, and A. Zisserman, "A Compact and Discriminative Face Track Descriptor," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[50] M. Tapaswi, M. Bäuml, and R. Stiefelhagen, ""Knock! Knock! Who is it?" Probabilistic Person Identification in TV-Series," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[51] G. Paul, K. Elie, M. Sylvain, O. Jean-Marc, and D. Paul, "A conditional random field approach for audio-visual people diarization," in *International Conference on Audio, Speech, and Signal Processing*, 2014.

[52] L. Zhang, D. V. Kalashnikov, and S. Mehrotra, "A unified framework for context assisted face clustering," in *ACM International Conference on Multimedia (ACM MM)*, 2013.

[53] C. Zhou, C. Zhang, H. Fu, R. Wang, and X. Cao, "Multi-cue augmented face clustering," in *ACM International Conference on Multimedia (ACM MM)*, 2015.

[54] Z. A.-H. Monica-Laura Haurilet, Makarand Tapaswi and R. Stiefelhagen, "Naming TV Characters by Watching and Analyzing Dialogs," in *Winter Conference on Applications of Computer Vision (WACV)*, 2016.

[55] V. Sharma and L. Van Gool, "Image-level classification in hyperspectral images using feature descriptors, with application to face recognition," *arXiv:1605.03428*, 2016.

[56] V. Sharma, A. Diba, T. Tuytelaars, and L. Van Gool, "Hyperspectral cnn for image classification & band selection, with application to face recognition," *Technical report KUL/ESAT/PSI/1604, KU Leuven, ESAT, Leuven, Belgium*, 2016.

[57] A. Miech, J.-B. Alayrac, P. Bojanowski, I. Laptev, and J. Sivic, "Learning from video and text via large-scale discriminative clustering," in *International Conference on Computer Vision (ICCV)*, 2017.

[58] V. Roethlingshoefer, V. Sharma, and R. Stiefelhagen, "Self-supervised face-grouping on graph," in *ACM International Conference on Multimedia (ACM MM)*, 2019.

[59] M. S. Sarfraz, V. Sharma, and R. Stiefelhagen, "Efficient parameter-free clustering using first neighbor relations," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[60] Y. He, K. Cao, C. Li, and C. C. Loy, "Merge or not? learning to group faces via imitation learning," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[61] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[62] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: unsupervised learning using temporal order verification," in *European Conference on Computer Vision (ECCV)*, 2016.

[63] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *International Conference on Computer Vision (ICCV)*, 2015.

[64] J. H. Ward Jr., "Hierarchical Grouping to Optimize an Objective Function," *JASA*, 1963.

[65] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[66] V. Sharma, M. S. Sarfraz, and R. Stiefelhagen, "A simple and effective technique for face clustering in tv series," in *CVPR: Brave New Motion Representations Workshop*, 2017.

[67] A. Diba, M. Fayyaz, V. Sharma, A. Hossein Karami, M. Mahdi Arzani, R. Yousefzadeh, and L. Van Gool, "Temporal 3d convnets using temporal transition layer," in *CVPR Workshop*, 2018.

[68] R. Yan, A. G. Hauptmann, and R. Jin, "Negative pseudo-relevance feedback in content-based video retrieval," in *ACM International Conference on Multimedia (ACM MM)*, 2003.

[69] R. Yan, A. Hauptmann, and R. Jin, "Multimedia search with pseudo-relevance feedback," in *Image and Video Retrieval*, 2003, pp. 238–247.

[70] C. Doersch, "Tutorial on Variational Autoencoders," *arXiv:1606.05908*, 2016.

[71] E. Ghaleb, M. Tapaswi, Z. Al-Halah, H. K. Ekenel, and R. Stiefelhagen, "Accio: A Dataset for Face Track Retrieval in Movies Across Age," in *ICMR*, 2015.

[72] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, "MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition," in *European Conference on Computer Vision (ECCV)*, 2016.

[73] M. Roth, M. Bäuml, R. Nevatia, and R. Stiefelhagen, "Robust Multi-pose Face Tracking by Multi-stage Tracklet Association," in *International Conference on Pattern Recognition (ICPR)*, 2012.

[74] A. Vedaldi and K. Lenc, "Matconvnet: Convolutional neural networks for matlab," in *ACM International Conference on Multimedia (ACM MM)*, 2015.

Vivek Sharma is a PhD candidate at Karlsruhe Institute of Technology (KIT), Germany. He is also a research affiliate at Massachusetts Institute of Technology and Harvard Medical School, Harvard University. His main research interests lies in supervised and unsupervised representation learning, video understanding, and multi/hyper-spectral imaging. WWW: https://vivoutlaw.github.io

Makarand Tapaswi is currently a postdoctoral fellow at Inria, Paris, France. He completed his PhD at Karlsruhe Institute of Technology (KIT) in 2016 and was a PostDoctoral Fellow at the University of Toronto for much of this work. His main research interests lie at the intersection of video and language. In particular, he is interested in teaching machines about human behavior through the analysis of stories presented in movies and TV series.

M. Saquib Sarfraz Dr. Sarfraz has obtained his PhD in Computer Vision at Technical University Berlin, Germany in 2009. Currently he shares his time both at Karlsruhe Institute of Technology (KIT) and at Daimler as senior scientist. He is member of several related funded projects, where he is working on perception of people for HCI interfaces. He has published in peer-reviewed journals, conferences and as invited book chapters. He has received five best paper awards at international vision conferences in 2008, 2010, 2015 and 2019. He is serving as a reviewer for several related conferences and journals. His research interests include statistical machine learning for face/person recognition, unsupervised learning and multi-modal biometrics.

Rainer Stiefelhagen Rainer Stiefelhagen received his Diplom (Dipl.-Inform) and Doctoral degree (Dr.-Ing.) from the Universitt Karlsruhe (TH) in 1996 and 2002, respectively. He is currently a full professor for "Information technology systems for visually impaired students" at the Karlsruhe Institute of Technology (KIT), where he directs the Computer Vision for Human-Computer Interaction Lab at the Institute for Anthropomatics and Robotics as well as KIT's Study Center for Visually Impaired Students. His research interests include computer vision methods for visual perception of humans and their activities, in order to facilitate perceptive multimodal interfaces, humanoid robots, smart environments, multimedia analysis and assistive technology for persons with visual impairments.