A Compact and Discriminative Face Track Descriptor

Omkar M. Parkhi Karen Simonyan Andrea Vedaldi Andrew Zisserman
Visual Geometry Group, Department of Engineering Science, University of Oxford, UK
{omkar,karen,vedaldi,az}@robots.ox.ac.uk

Abstract

Our goal is to learn a compact, discriminative vector representation of a face track, suitable for the face recognition tasks of verification and classification. To this end, we propose a novel face track descriptor, based on the Fisher Vector representation, and demonstrate that it has a number of favourable properties. First, the descriptor is suitable for tracks of both frontal and profile faces, and is insensitive to their pose. Second, the descriptor is compact due to discriminative dimensionality reduction, and it can be further compressed using binarization. Third, the descriptor can be computed quickly (using hard quantization) and its compact size and fast computation render it very suitable for large scale visual repositories. Finally, the descriptor demonstrates good generalization when trained on one dataset and tested on another, reflecting its tolerance to the dataset bias. In the experiments we show that the descriptor exceeds the state of the art on both face verification task (YouTube Faces without outside training data, and INRIA-Buffy benchmarks), and face classification task (using the Oxford-Buffy dataset).

1. Introduction

Video repositories of the latest generation such as Netflix, the BBC iPlayer and YouTube have made available data of unprecedented size and scope, driving the demand for technology that can process vast amounts of videos intelligently. The aim is to automatically and efficiently discover, search, and organize semantic information in them. Since these videos are largely about *people*, a significant amount of effort has been dedicated to understanding people in videos by estimating their pose, actions, activities, interactions, relation to the environment, *etc*. This paper, in particular, addresses the challenging problem of *recognizing people* in videos, defined as either classifying or verifying their identity.

The evident advantage of recognizing people in videos, compared to recognition from a single still image, is that a video potentially contains multiple images of the same face. Despite this advantage, face recognition in large unconstrained video collections poses some serious technical challenges. The first is to achieve *invariant recognition* as the appearance of faces in videos varies hugely due to changes in viewpoint, illumination, resolution, noise and many other nuisance factors. In particular, one of our goals is to be robust to substantial face rotations, from frontal to profile. The second challenge is to achieve *efficient recognition* by constructing indexes of vast video libraries that can fit in central memory and be searched in seconds. The third is to achieve a *discriminant representation* that can be used to distinguish one person from another.

We consider *face tracks* [31] as the fundamental unit of face recognition in videos and develop a *Video Fisher Vector Faces* (VF^2) descriptor, a video track representation that is, as required, discriminative, invariant, and yet very compact, capable of efficiently operating at the scale of a vast video collections such as YouTube. This face track descriptor is based on the Fisher Vector (FV) encoding [25], and in particular our Fisher Vector Faces representation [28], a powerful descriptor developed for a single-image face verification. VF^2 inherits and extends its advantages as discussed below.

First, VF^2 is easy and fast to compute. It does not rely on landmark detection for image sampling, but uses conceptually simple dense SIFT features encoded with Fisher vectors. Unlike descriptor stacking, FV encoding scales well to pooling over the whole face track, and dataset augmentation can also be incorporated in the pooling stage. Furthermore, VF^2 utilizes a hard-assignment FV variant that is particularly well-suited for videos due to its efficiency.

Second, using discriminative dimensionality reduction, we propose both low-dimensional real-valued and binary-valued variants. The resulting face representation is extremely compact, resulting in fast face classification and verification and compact indexing. The binary variant has a very low memory footprint and is extremely fast to match by using hardware-accelerated Hamming distance computation, making it well suited for big data problems or low-power devices such as mobile phones. While binary rep-

resentations have attracted significant attention in the local descriptor field [5, 19, 26, 30] and large-scale image retrieval [14, 15, 35], to the best of our knowledge we are the first to propose binary descriptors for face recognition.

Third, we show that our representation is versatile in that it achieves state-of-the-art performance in both face verification (Sect. 4) and face classification (Sect. 5) on several challenging benchmark datasets. Furthermore, we demonstrate that our model generalizes well by learning the representation on one dataset and then applying it with very competitive performance on other tasks and data.

After developing our face track representation in detail (Sect. 3), the paper includes a thorough experimental validation of the quality, compactness, and generality of the face descriptors on several benchmark datasets (Sect. 4, 5) as well as an analysis indicating *why* it is so effective as a face representation (Sect. 6).

2. Related work

This paper focuses on the problem of video face recognition "in the wild". For the most part face tracks have been limited to frontal faces only, but a small number of papers have also included profile faces [13, 32, 34, 36]. Irrespective of pose, there are two approaches that have been followed in the previous search for representations. The first approach is based on representing faces in a track individually and comparing two face tracks as sets of images. The second, is to represent the track as a whole, for example by a single vector or as a 3D model or other manifold, with no explicit reference to the individual face detections. A good survey of these methods can be found in [4].

The first approach has been the most popular, and allows representations developed for still images (their overview can be found in [28]) to be applied directly to face tracks. A standard method is that of [12] where features are computed around facial landmarks and concatenated to form a descriptor vector for each frame of the track. Tracks are subsequently classified using the min-min distance on the set of descriptors (either as a nearest neighbor classifier or as a kernel). Others have varied the landmark features, *e.g.* multi-scale SIFT or HOG [1, 10, 31, 32], as well as the type of the classifier, *e.g.* random forests, MKL, or SVM-minus scoring [1, 32, 37], and also the weight given to each frame [33].

The second approach has used both generative and statistical representations. One method is to build 3D models from the data [18, 39] and use these to explain new data, for example by view synthesis [23]. Often, the face track is represented as a linear or non-linear manifold [2, 16]. Others have learnt local features across the track and represented them using a BoW model [31], or a GMM [20], or by sparse coding [11]. For speed and low storage requirements it is also common to use the descriptor vector of a single representative (mean) face [24, 34]. An alternative representation is to simply use the affine hull of the face sets [6, 38]. Our approach is similar in nature to the second family of representations.

3. Face representation

This section describes in detail VF^2, our face track representation. This is obtained in three steps: (a) fast Fisher vector faces computation (Sect. 3.1), (b) discriminative dimensionality reduction (Sect. 3.2), and (c) descriptor binarization (Sect. 3.3).

3.1. The VF^2 face track descriptor

Our starting point is the Fisher Vector Face (FVF) representation, proposed in [28] for the case of still images. FVF is based on the Fisher Vector (FV) method of [25] which has recently enjoyed widespread success, compared to previous feature encoding methods [7]. While this is our basis for descriptor computation, we make a number of important changes. First, information is pooled not from a single image but from the entire face track using spatio-temporal pooling. Second, the computation of the Fisher Vector as described in [25] is simplified and accelerated with nearly no loss of performance. Finally, we show that dataset augmentation can be incorporated in the Fisher vector encoding to further improve its performance.

The Fisher vector encoding. The FV representation $\phi(\ell)$ of an image ℓ is formed by computing a set of local features $\mathbf{x}_1, \ldots, \mathbf{x}_p, \ldots, \mathbf{x}_N \in \mathbb{R}^D$ (*e.g.* densely extracted SIFT features) and accumulating the 1st and 2nd order statistics of their distribution in an image [25] $\phi(\ell) = \left[\Phi_1^{(1)}, \Phi_1^{(2)}, \ldots, \Phi_K^{(1)}, \Phi_K^{(2)}\right]$ where each dimension $\Phi_{ik}^{(1)}$ and $\Phi_{ik}^{(2)}$, $i = 1, \ldots, D$ of the vectors $\Phi_k^{(1)}$ and $\Phi_k^{(2)}$ is

$$\Phi_{ik}^{(1)} = \frac{1}{N\sqrt{\pi_k}} \sum_{p=1}^{N} \alpha_k(\mathbf{x}_p) \left(\frac{x_{ip} - \mu_{ik}}{\sigma_{ik}}\right), \quad (1)$$

$$\Phi_{ik}^{(2)} = \frac{1}{N\sqrt{2\pi_k}} \sum_{p=1}^{N} \alpha_k(\mathbf{x}_p) \left[\left(\frac{x_{ip} - \mu_{ik}}{\sigma_{ik}}\right)^2 - 1\right]. \quad (2)$$

Here the feature soft-assignment $\alpha_k(\mathbf{x}_p) \propto \pi_k \exp\left[-\frac{1}{2}(\mathbf{x}_p - \mu_k)\Sigma_k^{-1}(\mathbf{x}_p - \mu_k)\right]$ is the posterior probability $p(k|\mathbf{x}_p)$ of the k-th component of a Gaussian Mixture Model (GMM) with component prior, mean, and covariance parameters $(\pi_k, \mu_k, \Sigma_k : k = 1, \ldots, K)$ modeling the descriptor distribution. The covariance matrices $\Sigma_k = \mathrm{diag}(\sigma_{1k}^2, \ldots, \sigma_{Dk}^2)$ are assumed to be diagonal, so the local features $\mathbf{x}_k$ are decorrelated and optionally dimensionality-reduced using PCA before FV computation.

The *Improved Fisher Vector* (IFV) [25] further post-process $\Phi(\ell)$ by applying the signed square root function

Figure 1. **Unsupervised GMM implicitly performs robust face part tracking.** On the left frame of the track in the *top row*, we manually selected 4 points on eyelids, nose and mouth. These points correspond to 4 SIFT patches, which are assigned to particular Gaussians from the GMM (colour-coded with red, green, blue, and yellow). We visualise the pixels assigned to these Gaussians on 9 frames of the track. As can be seen, the same face parts get consistently assigned to the same Gaussians, which indicates the robustness of the representation with respect to the face position variation. In the *bottom* row, we visualise the face pixels assigned to exactly the same Gaussians as in the top row, but for a different person. Again, they correspond to the same face parts, which means that the assignment is consistent across different identities. The comparison of Fisher vectors thus can be seen as the implicit and robust comparison of face parts.

$\text{sign}(z)\sqrt{|z|}$ and then renormalizing the result in l^2 norm. As shown in [25], this step is essential to achieve good performance with linear SVM classifiers. In the rest of the paper, when FV are discussed it is intended that this improved variant is used.

Efficient computation of hard-assignment Fisher vectors. In the traditional FV formulation (1) the soft assignment coefficients $\alpha_k(\mathbf{x}_p)$ assign local features $\mathbf{x}_p$ to all K components of the GMM. When the number of Gaussians is large, averaging the statistics over N features and K Gaussian components becomes a computational bottleneck. This problem is exacerbated in dealing with multiple frames forming a face track. To accelerate this computation we employ an FV encoding variant, called hard-FV [29], that replaces the soft assignment of features to GMM components with the hard assignment to a single Gaussian, which corresponds to the largest likelihood. Typically, the hard assignment delivers a speed up in computation by a factor of six, at the cost of a small (1%) drop in performance.

FV encoding of face tracks. Similar to [28], FV is applied to faces by extracting RootSIFT [3, 21] descriptors densely and at multiple resolutions from each face image. This is a key difference from other approaches that extract descriptors around predefined landmarks. [28] shows that this solution yields state-of-the-art accuracy; and furthermore, in the case of videos, not having to run facial feature detectors on each frame is a significant computational advantage. To retain spatial information, the dense RootSIFT features are augmented with their normalized x, y location in the face window before clustering with the GMM [27].

Another aspect to discuss is how to merge information across multiple frames. We experimented with two alternatives: (i) *image pooling* – computing Fisher Vectors for each of the face track frames individually, followed by averaging the Fisher Vectors across frames; and (ii) *video pooling* – computing a single Fisher Vector over the whole face track by pooling together SIFT features from all the faces in a track. As shown in Sect. 4, video pooling outperforms image pooling. It should be noted that the two methods are not equivalent, since in the image pooling method each frame FV is individually normalized prior to averaging, while in video pooling the features from the whole track are combined together and normalized only once. As illustrated in Fig. 1 and further elaborated in Sect. 6, the Gaussian components act as dense pseudo-part detectors, allowing the descriptor to be robust to pose variations.

Jittered pooling. Data jittering, also known as virtual sampling, is a common technique to improve the invariance of learned descriptors or classifiers. The idea is to enrich the training set with transformed variants of the data to simulate distortions that are expected to occur at test time (*e.g.* small face rotations). Rather than increasing the size of the training set, however, here we use the simple idea of *extending video pooling to include jittered version of the data*. This is computationally quite cheap, results in descriptors of the same dimensionality, and indeed achieves the best results in our tests (Sect. 4). In the current implementation we considered as jitters only the horizontal flips of individual images in a track, but it is trivial to extend this to further variations such as rotations and crops. Note that this efficient augmentation is applied at test time as well.

Implementation details. We employ dense multi-scale RootSIFT [3] features, computed over 24×24 patches every 2 pixels at 5 levels of the scale-space pyramid with $\sqrt{2}$ scaling factor. The RootSIFT features are then PCA-projected to 64 dimensions, and augmented with their normalized spatial coordinates, which leads to 66-D local features, containing both visual and spatial information. The local features are then encoded using the Fisher vector, with the GMM codebook size set to 512. In order to reduce memory consumption while computing FVs, we incrementally encode SIFTs from each frame of the track into a FV but do not perform the normalization. Normalization is only performed at the end when all frames are encoded. This is

possible due to the additive construction of FVs (Eq. 1) and saves us from having to store SIFTs for all frames in a track into memory for pooling.

3.2. Discriminative dimensionality reduction

The FV features computed in Sect. 3.1 are very high-dimensional. To reduce the dimensionality of the representation and increase its discriminative power we use the same metric learning technique we introduced in [28].

Full metric learning. Metric learning is formulated as learning a linear projection $W \in \mathbb{R}^{p\times d}$, $p \ll d$, which maps the high-dimensional Fisher vectors $\phi \in \mathbb{R}^d$ to low-dimensional vectors $W\phi \in \mathbb{R}^p$, such that the squared Euclidean distance $d^2_W(\phi_i, \phi_j) = \|W\phi_i - W\phi_j\|_2^2$ is a good discriminant. This is formulated as a non-convex objective functional and optimized as described in [28] for the case of still face images.

Diagonal pseudo-metric learning. While the complexity of W can be controlled by reducing the number of its rows, which corresponds to learning a lower-dimensional projection of the data, small datasets may be insufficient to properly learn all the parameters. In this case we resort to learning a diagonal matrix $W^\top W = \operatorname{diag} \mathbf{w}$, where the vector $\mathbf{w}$ is learned with a conventional linear SVM formulation from constraints [28].

Joint metric-similarity learning. [8, 9] suggest simultaneously learning a low-rank Mahalanobis distance $(\phi_i - \phi_j)^T W^T W(\phi_i - \phi_j)$ *and* a low-rank kernel (inner product) $\phi_i^T V^T V \phi_j$ and use the difference between the distance and the inner product between features ϕ_i for verification. We incorporate this formulation in the objective function as in [28] and report results with this variant as well.

3.3. Binary compression

While the low-rank metric learning method of Sect. 3.2 is already capable of achieving a very good compression factor, for large scale applications the goal is to further decrease the number of bits required to encode each face track. This section describes a method to map a low-dimensional real valued descriptor $\psi \in \mathbb{R}^m$ to a binary code $\beta \in \{0,1\}^q$ with the bit length q (where $q \geq m$). While there are several alternative hashing methods that could be used for this purpose, [17] suggests a very competitive and efficient method based on computing a Parseval tight frame, followed by thresholding. Recently, this binarization method has been successfully applied by [30] in the local feature descriptor domain.

In more detail, a *frame* is a matrix $U \in \mathbb{R}^{q\times m}$ whose row-span is the space of vectors $\psi \in \mathbb{R}^m = \operatorname{span} U^\top$ to be encoded. A *tight frame* has the additional property that $U^\top U = I$. Multiplying the vector ψ by the tight frame produces an over-complete representation $U\psi \in \mathbb{R}^q$; while the latter has an increased dimensionality, it "spreads the information" among many redundant dimensions. The result is that thresholding this vector by the sign function (defined as $\operatorname{sign}(a) = 1$ iff $a > 0$ and 0 otherwise) yields a binary vector

$$\beta = \operatorname{sign}(U\psi) \tag{3}$$

that is an accurate representation (in terms of the Euclidean distances) of the vector ψ. In practice, as suggested by [17], U is computed by keeping the first m columns of an orthogonal matrix obtained from a QR-decomposition of a random $q \times q$ matrix. For thresholding the $sign$ function to produce a meaningful binary string, the vector ψ needs to be zero centered. This is achieved by subtracting a mean vector computed over a large number of feature vectors.

Note that, while the dimensionality q of the binary code is not smaller than the dimensionality of the data m, the representation is significantly more compact provided that $q/m \ll 32$; since encoding each binary dimension requires a single bit, whereas encoding a floating point number usually requires 32 or 64 bits. In this manner, changing q allows us to generate the binary descriptors with any desired bitrate. Additionally, the Euclidean distance between binary vector reduces to the Hamming distance which can be computed very quickly using the XOR and POPCNT (population count) instructions in recent CPUs.

4. Experiments on face verification

This section begins a thorough evaluation of the face track descriptor developed in Sect. 3 on a number of challenging face recognition benchmarks. It is shown that: the representation achieves state-of-the-art performance on certain face verification and face recognition tasks; binarization preserves the discriminative power of the representation while achieving a very high compression ratio; and that the learnt representation can be transferred across datasets with negligible loss of performance, demonstrating good generalization. This section is dedicated to the face verification problem and Sect. 5 investigates face classification.

The face verification task is the following: given a pair of face tracks, determine if they portray the same person or not. Here we address this task by computing VF^2 descriptors, reducing their dimension, and comparing their Euclidean distance (or Hamming distance for binary descriptors).

Test time flip. In the implementation we use horizontal flipping at test time for augmentation: each face detection is reflected horizontally and face track descriptors computed for the original and flipped track. The distance between the test tracks is then computed by averaging the four distances between them. However, this method is redundant in the case of the *jitter pooled* variant of the descriptor and hence is not used in that case.

4.1. Face verification datasets

YouTube Faces (YTF) [36]. The YouTube Faces dataset is a popular benchmark for face verification in video data captured in uncontrolled conditions. It contains 3,425 videos of 1,595 celebrities collected from YouTube, with an average of 2 videos per identity, and the average clip length of 181 frames. The dataset includes pre-extracted face tracks, the alignment of faces to a canonical frame (normalization), as well as a specification of 6,000 track pairs, divided in 10 splits of 600 pairs. Each of the 10 splits contains 300 positive pairs (same identity) and 300 negative pairs. Two evaluation settings are defined. In the *restricted setting*, 9 splits (5,400 pairs) are used for training and the remaining split – for testing. The final performance measure is obtained by averaging the results over the 10 folds. In the *unrestricted setting*, one is allowed to combine the training tracks in an arbitrary number of training pairs. Face verification results are reported using three metrics: Area under the Receiver Operating Characteristic curve (AUC), *Receiver Operating Characteristic Equal Error Rate* (ROC-EER, or simply EER) and verification accuracy [36].

INRIA-Buffy dataset [10]. This dataset consists of 639 face tracks, automatically extracted from episodes 9, 21, and 45 of the television series "Buffy the Vampire Slayer". Tracks are assigned one of nine labels: eight for the main characters in the series and the ninth denoting "others". The training set consists of 312 tracks, and the test set of 327 tracks. Note that there are no track-level face identity labels available in the training set. Therefore we will use this dataset mainly to evaluate transferring representations learned on YTF.

4.2. Face verification evaluation

YouTube Faces verification. Our face track representation is computed on the aligned face images, provided by the YTF organizers. From these images, we extract a 150×150 patch around the center of the face, and use it for feature computation. The resulting VF^2 is compressed to a low-dimensional vector using discriminative dimensionality reduction (Sect. 3.2), followed by optional binarization (Sect. 3.3).

In Table 1, we evaluate different settings of our discriminatively projected VF^2 in terms of the EER performance measure. The experiments are performed in the unrestricted setting. First, we compare FV pooling techniques (Sect. 3.1): image pooling (row 1) and video pooling (row 2). As can be seen, the latter performs better, which indicates that it is beneficial to aggregate the visual statistics across the whole face track prior to normalization. Soft quantized Fisher vectors improve the results by 1.2% (row 3) but increases the processing time up to 6 times. Jittered pooling (row 4) yields a better improvement at a smaller cost by pooling more features together. Next, we asses how the error rate changes depending on the projected FV dimension (rows 5–7) and conclude that, on this benchmark, there is no improvement beyond dimension 128 due to model overfitting. In rows 8 and 9 we report the verification performance of the 128-D projected FV descriptor (row 2), binarized to 1024 and 2048 bits respectively using the method of Sect. 3.3. As a result of binarization, not only is the face descriptor memory footprint decreased from 512 bytes (128×4-byte single-precision values) to 128 bytes (1024 bits), but also the EER decreases from 16.2% (row 2) to 15.0% (row 8) due to the binary representation being more robust. A similar improvement is observed for the jittered pooling variant: 14.2% (row 4) to 13.4% (row 10).

Finally, rows 11–13 show the result of the joint similarity-distance learning formulation [8, 28] (Sect. 3.2),

Setting	VF^2 variant	Proj. Dim.	EER
1	image pool.	128	17.3
2	video pool.	128	16.2
3	video pool. + soft quantization	128	15.0
4	video pool. + jittered pool.	128	14.2
5	video pool.	256	16.9
6	video pool.	512	17.0
7	video pool.	1024	17.0
8	video pool. + binar. 1024 bit	128	15.0
9	video pool. + binar. 2048 bit	128	15.0
10	video pool. + binar. 1024 bit + jitt. pool.	128	13.4
11	video pool. + joint sim.	128×2	14.4
12	video pool. + joint sim. + flip	128×2	13.0
13	video pool. + joint sim. + jitt. pool.	128×2	**12.3**

Table 1. **The performance of different descriptor settings on YouTube face verification (unrestricted setting).** Reported is the equal error rate measure (smaller values correspond to better verification).

	Method	Accuracy	AUC	EER
1	MGBS & SVM- [37]	78.9 ± 1.9	86.9	21.2
2	APEM FUSION [20]	79.1 ± 1.5	86.6	21.4
3	STFRD & PMML [11]	79.5 ± 2.5	88.6	19.9
4	VSOF & OSS (Adaboost) [22]	79.7 ± 1.8	89.4	20.0
5	Our VF^2 (restricted)	83.5 ± 2.3	92.0	16.1
6	Our VF^2 (restricted & flip)	84.7 ± 1.4	93.0	14.9
7	Our VF^2 (unrestricted & flip)	83.5 ± 2.1	94.0	13.0
8	Our VF^2 (unrestricted & jitt. pool.)	83.8 ± 1.6	95.0	**12.3**

Table 2. **Comparison with the state of the art on YouTube verification (restricted and unrestricted).** Our Fisher vector face track descriptors significantly outperform the existing methods, setting the new state of the art. All rows except the last two rows show results on the restricted training setting.

Setting	Method	EER
1	Cinbis *et al.* [10]	42.50
2	Our VF^2 (GMM trained on YTF)	35.27
3	Our VF^2 (GMM trained on Buffy)	30.48
4	Our VF^2 (GMM trained on Buffy) & flip	**30.11**

Table 3. **Comparison with the state of the art on INRIA-Buffy verification (using the unsupervised Euclidean distance).** Our method outperforms the state of the art by 12%.

Setting	VF^2 variant	Proj. Dim.	EER
1	Cinbis *et al.* [10] (trained on LFW)	N.A.	36.20
2	Cinbis *et al.* [10] (trained on Buffy)	N.A.	30.00
3	video pool.	128	30.24
4	video pool. + flip	128	28.16
5	video pool. + joint sim. + flip	128×2	25.77
6	video pool. + binar. 1024 bit + flip	128	22.21
7	video pool. + binar. 2048 bit + flip	128	**21.90**

Table 4. **Comparison with the state of the art on INRIA-Buffy verification (using the learnt distance).** proj-n – projection dimensionality. bin-n – dimensionality after binarization.

reducing the error rate to 14.4% compared to learning only a distance metric (rows 2 and 11) and to 12.3% for the jittered pooled descriptor (rows 4 and 13). Although using test time flips improves results (row 12), dataset augmentation using jittered pooling achieves the best results of 12.3% (row 13).

The comparison with the current state-of-the-art methods is given in Table 2. In the restricted setting, we could not train a full projection matrix W due to the small number of training pairs available in this case, leading to overfitting. Instead, we used diagonal metric learning (Sect. 3.2). This combination outperforms the state of the art by $\sim 4\%$ and achieves an EER of 16.1% (row 5). This is improved further to 14.9% with test-time flips (row 6).

In the large-scale unrestricted setting (Table 2, rows 7-8), we can fully leverage the available amount of annotation and learn a full projection matrix W. To the best of our knowledge, we are the first to report the performance in the unrestricted setting. Compared to the restricted case, the EER in the unrestricted scenario is smaller by 2.6% (12.3% vs 14.9%).

INRIA-Buffy verification. Similar to the YouTube face verification task, this task involves determining whether a given pair of tracks portrays the same person or not. We compare VF^2 with the state-of-the-art method [10] in two settings: unsupervised and supervised.

In the unsupervised setting, the Euclidean distance between the uncompressed VF^2 descriptors computed on the 80×80 face patches provided with the dataset is used for comparison (Table 3). We compare training the GMM used to compute the VF^2 representation on YouTube Faces and INRIA-Buffy. Even when the GMM is trained on a different dataset (YTF), we significantly (7.2%) outperform the descriptor of [10] on INRIA-Buffy (row 2). The results are further improved by computing the GMM on INRIA-Buffy directly, leading to a substantial 12% improvement over the state of the art (rows 3 and 4).

Having shown the superiority of VF^2 in the unsupervised case, we now couple it with the discriminatively trained metric, which simultaneously leads to dimensionality reduction. In order to learn the metric, we require positive and negative face tracks. Unfortunately, forming positive track pairs is not possible in INRIA-Buffy due to a limitation of the dataset (one only knows that frames within a track correspond to the same person)[1]. Hence we learn the metric on YTF and evaluate the resulting descriptor on INRIA-Buffy, verifying in this manner the ability of the method to transfer from one dataset to another. Note that this requires training the GMM on YTF as well. As shown in Table 4, despite learning on a completely different dataset, we achieve an EER of 30.24% (row 3), 6% better than [10] (row 1). Using joint metric learning and test time flips improves the EER to 25.77% (row 5). Binarization improves the EER further to 21.90%, 8% better than the state of the art (row 2).

5. Experiments on face classification

In the previous section we have demonstrated the excellent properties of the proposed face track representation on the face verification task. This section shows that the representation is equally applicable to the face classification task, where it also achieves the state-of-the-art accuracy. The task here is to label each track with the identity of the person. We treat this problem as one of multi-way classification and learn a linear classifier for each person.

5.1. Face classification dataset

Oxford-Buffy dataset [32] is a popular benchmark for the face classification methods. It consists of 7 episodes from the season 5 of "Buffy The Vampire Slayer". The face tracks were formed by tracking frontal and profile face detections of the speakers within a shot. The dataset contains two types of annotation. First, for each track, the Ground-Truth (GT) label of the speaker is provided; the second type of annotation is noisy, as it contains the speaker label prediction, mined from subtitles and transcripts. The task is to label all faces in a video, training a model using noisy labels and testing using GT labels. We follow the evaluation protocol of [32] and measure the performance in terms of precision and "recall" of speaker label prediction. "Recall" in this context is the percentage of tracks labeled at a particular classifier confidence level, *i.e.* the ratio of tracks with the classification score higher than a threshold.

[1]The method of [10] does learn a metric from INRIA-Buffy, but uses individual track frames rather than tracks as a whole as we require.

To ensure a fair comparison with [32], we use the same video data, face tracks, and training/test splits, kindly provided by the authors.

5.2. Face classification evaluation

The results of the classification experiments (together with the state-of-the-art results of [32]) are presented in Table 5. We use linear one-vs-rest SVMs trained on all the variants of our VF^2 descriptor using the methods described in Sect. 3.

Namely, in rows 3–4 we use uncompressed VF^2 vectors, computed using the GMM trained on Oxford-Buffy and YTF respectively. As can be seen, the drop in performance incurred by transferring the GMM from another dataset is small (1%). With the GMM trained on Buffy, we achieve better performance than Sivic *et al.* [32], who used multiple-kernel RBF-SVM over HOG features.

Rows 4–7 show the results of training the SVM classifiers on low-dimensional VF^2 projections (the projection matrix was trained on YTF). Discriminative compression to 1024 dimensions simultaneously improves the performance by 2%, whereas compression to 256 dimension has the same performance as the uncompressed descriptors. Using jittered pooling version of the descriptor (row 7) improves results further by 3% to get an average mAP of 0.86. This is 6% better than the previous state of the art (row 2).

A similar trend is observed after applying binarization, which dramatically reduces the memory footprint (to *e.g.* 256 bytes per VF^2 vector in row 9) with a small drop in performance ($\sim 1\%$).

The fact that we obtained very good classification results (rows 4–8) by utilizing features trained on a different dataset (YTF) emphasizes that VF^2 generalizes well to new, unseen datasets for the classification tasks as well. At the same time, it should be noted that using outside data to learn VF^2 is not essential: we achieve the same (or better) level of performance when training solely on Buffy (row 3).

The results in Table 5 correspond to the evaluation protocol where training is performed on noisy speaker labels. For completeness, we also report the classification performance when training on the ground-truth labels. In that case, we achieve 92% accuracy with the same setting as row 3 of Table 5 and 93% with same setting as row 7, better than the 91% reported by [32].

6. Discussion

Our VF^2 representation is remarkably simple as it does not account *explicitly* for invariance properties of faces in a track, and yet it achieves state-of-the-art results in a number of datasets. Fig. 1 shows that, in fact, this representation is achieving invariance *implicitly* by visualizing the pixels in a face image, whose descriptors are hard-assigned to selected GMM components. The key observation is that GMM components fire in a *transformation co-variant* manner, such that different components can be seen as capturing different parts of the face (*e.g.* eyes, mouth, and nose). Furthermore, the same components fire on analogous parts in different faces, establishing semantically meaningful correspondences. By stacking x, y coordinates to the SIFT descriptor, moreover, these act as "spring terms" encouraging given components to fire around the mean part location, with a regularization effect. Note that no explicit selection, learning or detection of facial landmarks is required.

7. Conclusion

We have proposed VF^2, a new face track descriptor which has many desirable properties: it is applicable to frontal, three-quarter and profile faces; it does not require the pose to be specified or facial feature points to be detected; it is compact and can be binarized with almost no loss in performance; and it is fast to compute. Furthermore, and importantly, it exceeds the state-of-the-art substantially in several settings on three standard benchmark datasets in face verification and classification in videos. We justified the robustness of the method by visualizations that show how the GMM in the VF^2 computation implicitly identifies corresponding facial regions up to challenging visual transformations due to variation in pose, lighting, and deformations from expressions.

Acknowledgements. This work was supported by ERC grant VisRec no. 228180 and EU Project AXES ICT–269980.

References

[1] N. E. Apostoloff and A. Zisserman. Who are you? – real-time person identification. In *Proc. BMVC.*, 2007. 2

[2] O. Arandjelović, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell. Face recognition with image sets using manifold density divergence. In *Proc. CVPR*, 2005. 2

[3] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *Proc. CVPR*, 2012. 3

[4] J. R. Barr, K. W. Bowyer, P. J. Flynn, and S. Biswas. Face recognition from video: A review. *Int. J. Pattern Recognition and Artificial Intelligence*, 26(5), 2012. 2

[5] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF: Binary robust independent elementary features. In *Proc. ECCV*, 2010. 2

[6] H. Cevikalp and B. Triggs. Face recognition based on image sets. In *Proc. CVPR*, 2010. 2

[7] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *Proc. BMVC.*, 2011. 2

[8] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesian face revisited: A joint formulation. In *Proc. ECCV*, pages 566–579, 2012. 4, 5

[9] D. Chen, X. Cao, F. Wen, and J. Sun. Blessing of dimension-

Setting	GMM & proj-n train. set	Proj-n	Bin-n	1	2	3	4	5	6	7	Avg
1	Sivic *et al.* [32]			0.89	0.83	0.68	0.82	0.85	0.69	0.78	0.79
2	Sivic *et al.* [32] (our implementation)			0.91	0.80	0.63	0.86	0.87	0.67	0.83	0.80
3	Buffy	none	none	0.89	0.80	0.67	0.88	0.89	0.72	0.84	0.81
4	YouTube Faces	none	none	0.87	0.77	0.61	0.86	0.89	0.71	0.87	0.80
5	YouTube Faces	256	none	0.91	0.75	0.72	0.83	0.91	0.66	0.87	0.81
6	YouTube Faces	1024	none	0.93	0.80	0.70	0.86	0.92	0.75	0.88	0.83
7	YouTube Faces (Jitt. Pool.)	1024	none	0.94	0.83	0.78	0.89	0.92	0.74	0.90	**0.86**
8	YouTube Faces	256	1024 bits	0.90	0.76	0.72	0.86	0.91	0.69	0.85	0.81
9	YouTube Faces	1024	2048 bits	0.90	0.77	0.71	0.86	0.90	0.73	0.86	0.82

Table 5. **The comparison of different settings of our framework and the state of the art on Oxford-Buffy classification.** Training is performed on the noisy speaker labels. We use a linear SVM on our VF^2, [32] is based on MKL RBF-SVM on HOG features.

ality: High dimensional feature and its efficient compression for face verification. In *Proc. CVPR*, 2013. 4

[10] R. G. Cinbis, J. J. Verbeek, and C. Schmid. Unsupervised metric learning for face identification in TV video. In *Proc. ICCV*, pages 1559–1566, 2011. 2, 5, 6

[11] Z. Cui, W. Li, D. Xu, S. Shan, and X. Chen. Fusing robust face region descriptors via multiple metric learning for face recognition in the wild. In *Proc. CVPR*, 2013. 2, 5

[12] M. Everingham, J. Sivic, and A. Zisserman. Taking the bite out of automatic naming of characters in TV video. *Image and Vision Computing*, 27(5), 2009. 2

[13] M. Everingham and A. Zisserman. Identifying individuals in video by combining generative and discriminative head models. In *Proc. ICCV*, 2005. 2

[14] A. Ginois, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Int. Conf. on Very Large Data Bases*, 1999. 2

[15] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *Proc. CVPR*, 2011. 2

[16] A. Hadid and M. Pietikäinen. Manifold learning for video-to-video face recognition. In *Biometric ID Management and Multimodal Communication*, 2009. 2

[17] H. Jégou, T. Furon, and J. J. Fuchs. Anti-sparse coding for approximate nearest neighbor search. In *International Conference on Acoustics, Speech, and Signal Processing*, 2012. 4

[18] V. Krüger, R. Gross, and S. Baker. Appearance-based 3-D face recognition from video. In *Proc. German Pattern Recognition Symp.*, 2002. 2

[19] S. Leutenegger, M. Chli, and R. Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *Proc. ICCV*, 2011. 2

[20] H. Li, G. Hua, J. Brandt, and J. Yang. Probabilistic elastic matching for pose variant face verification. In *Proc. CVPR*, 2013. 2, 5

[21] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 3

[22] H. Mendez-Vazquez, Y. Martinez-Diaz, and Z. Chai. Volume structured ordinal features with background similarity measure for video face recognition. In *International Conference on Biometrics (ICB)*, 2013. 5

[23] U. Park, H. Chen, and A. Jain. 3d model-assisted face recognition in video. In *Proc. Canadian Conf. Computer and Robot Vision*, 2005. 2

[24] O. M. Parkhi, A. Vedaldi, and A. Zisserman. On-the-fly specific person retrieval. In *International Workshop on Image Analysis for Multimedia Interactive Services*. IEEE, 2012. 2

[25] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *Proc. ECCV*, 2010. 1, 2, 3

[26] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: an efficient alternative to SIFT or SURF. In *Proc. ICCV*, 2011. 2

[27] J. Sánchez, F. Perronnin, and T. de Campos. Modeling the spatial layout of images beyond spatial pyramids. *Pattern Recognition Letters*, 33(16), 2012. 3

[28] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher Vector Faces in the Wild. In *Proc. BMVC.*, 2013. 1, 2, 3, 4, 5

[29] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep Fisher networks for large-scale image classification. In *NIPS*, 2013. 3

[30] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning local feature descriptors using convex optimisation. *IEEE PAMI*, 2014. 2, 4

[31] J. Sivic, M. Everingham, and A. Zisserman. Person spotting: Video shot retrieval for face sets. In *Proc. CIVR*, 2005. 1, 2

[32] J. Sivic, M. Everingham, and A. Zisserman. "Who are you?" – learning person specific classifiers from video. In *Proc. CVPR*, 2009. 2, 6, 7, 8

[33] J. Stallkamp, H. K. Ekenel, and R. Stiefelhagen. Video-based face recognition on real-world data. In *Proc. ICCV*, 2007. 2

[34] M. Tapaswi, M. Baeuml, and R. Stiefelhagen. "knock! knock! who is it?" probabilistic person identification in tv series. In *Proc. CVPR*, 2012. 2

[35] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Proc. NIPS*, 2007. 2

[36] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *Proc. CVPR*, 2011. 2, 5

[37] L. Wolf and N. Levy. The svm-minus similarity score for video face recognition. In *Proc. CVPR*, 2013. 2, 5

[38] M. Yang, P. Zhu, L. Van Gool, and L. Zhang. Face recognition based on regularized nearest points between image sets. In *Proc. Int. Conf. Autom. Face and Gesture Recog.*, 2013. 2

[39] S. Zhou, V. Krueger, and R. Chellappa. Face recognition from video: A CONDENSATION approach. In *Proc. Int. Conf. Autom. Face and Gesture Recog.*, 2002. 2