To Frontalize or Not To Frontalize: Do We Really Need Elaborate Pre-Processing to Improve Face Recognition Performance?

Sandipan Banerjee*[1], Joel Brogan*[1], Janez Križaj[2], Aparna Bharati[1], Brandon RichardWebster[1], Vitomir Štruc[2], Patrick Flynn[1] and Walter Scheirer[1]
[1] Dept. of Computer Science & Engineering, University of Notre Dame, USA
[2] Faculty of Electrical Engineering, University of Ljubljana, Slovenia
{sbanerj1, jbrogan4, abharati, brichar1, flynn, wscheire}@nd.edu
{janez.krizaj, vitomir.struc}@fe.uni-lj.si

***Abstract*— Automatic face recognition performance has improved remarkably in the last decade. Much of this success can be attributed to the development of deep learning techniques like convolutional neural networks (CNNs). But the training process for CNNs requires a large amount of clean and well-labelled training data. If a CNN is intended to work with non-frontal face images, should this training data be diverse in terms of facial poses, or should face images be frontalized as a pre-processing step? We address this question in this paper. We evaluate a set of popular facial landmarking and pose frontalization algorithms to understand their effect on facial recognition performance. We also introduce a new automatic frontalization scheme that operates over a single image without the need for a subject-specific 3D model, and perform a comparative analysis between the new scheme and other methods in the literature. A CNN trained on face images frontalized using different pre-processing methods is used to extract features from the Point and Shoot Challenge (PaSC) video dataset. The verification and identification performance of the CNN serves to quantify the effectiveness of each landmarking and frontalization scheme. We find that frontalization, although an intuitive pre-processing strategy, does not significantly improve face recognition performance when compared with a simple 2D face alignment.**

I. INTRODUCTION

Facial recognition has been an open problem in biometrics and computer vision research for decades. Recently, the advent of deep learning [1] methods such as convolutional neural networks (CNNs) has allowed face recognition performance on hard datasets to improve significantly. For instance, Google FaceNet [2], a CNN based method, has achieved over 99% verification accuracy on the LFW dataset [4], which was once considered a very challenging dataset for face recognition algorithms [5]. Because CNNs possess the ability to automatically learn complex representations of face data, they systematically outperform methods based on hand-crafted features. Since these representations are learned from the data itself, it is often assumed that we must provide CNNs clean, pre-processed data for training. In this work, we posed the following question: can CNNs automatically learn robust representations invariant of facial pose, or does training with frontalized face images yield better results? To reach an answer, we conducted an extensive comparative analysis of different facial pre-processing techniques.

* denotes equal contribution

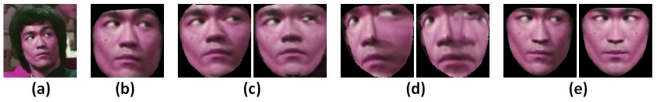

Fig. 1: Examples of frontalization on a sample image (a) from the CASIA-WebFace dataset [7]: (b) 2D aligned – no frontalization, (c) Zhu and Ramanan [14] & Hassner et al. [13], (d) Kazemi and Sullivan [15] & Hassner et al., (e) our method. The frontalization for the left and right images are asymmetric and symmetric respectively for (c), (d), and (e). Note how different the frontalization results look for each approach. Does this difference impact face recognition performance? We seek to answer this question in this work.

Commonly used public face datasets [6], [4], [7], [8] contain thousands to millions of unconstrained, frequently non-frontal face images. While researchers have utilized face image frontalization as a pre-processing step before training, a thorough evaluation of the effects of frontalization on face recognition is lacking. We have tried to fill this gap by performing a wide range of facial recognition peformance comparisons using a set of different frontalization techniques.

For these experiments, we selected CASIA Web Face [7] as our training dataset. Two frontalization techniques were chosen for our training and testing evaluation: the well-established method proposed by Hassner et al. (H) [13], and our own newly proposed method. To evaluate the effect of facial landmarking on the frontalization process, we used three effective landmarking techniques: the methods of Zhu and Ramanan (ZR) [14], Kazemi and Sullivan (KS) [15], and a novel technique — a Cascade Mixture of Regressors (CMR).

Remarkably different face frontalization results obtained using various combinations of these methods on a sample image from CASIA-WebFace can be seen in Fig. 1. We used the popular VGG-FACE CNN architecture proposed in [8] as our base architecture for training multiple networks using different pre-processing strategies. For the testing phase, we selected the challenging PaSC video dataset [16]. We extracted face representations from individual video frames in PaSC using a network trained with a particular pre-processing strategy. These features were used for verification and recognition purposes by applying a cosine similarity score-based face matching procedure. Since the focus of our study was to evaluate the effect of the pre-processing

methods on CNN-based face recognition, we chose not to use any elaborate detection algorithms like those used by most of the PaSC 2016 Challenge participants [16].

As a set of baselines for network training, we used 1) no pre-processing at all, 2) a simple 2D alignment that corrects for in-plane rotation using eye centers, and 3) a snapshot of the VGG-FACE model [8] pre-trained on the 2D aligned VGG-FACE dataset. This was used to evaluate how much the additional training on CASIA-WebFace improved the face representation capability of the CNN model. The effect of each data augmentation is manifested in the performance of each subsequent CNN model.

In summary, the contribution of this paper is two fold:

1) We evaluate a set of popular facial landmarking and frontalization methods and quantify their effect on performance in video-based face recognition tasks using a CNN.
2) We propose a new facial landmarking and frontalization technique for comparison.

II. RELATED WORK

Previous work relevant to this subject can be categorized into three broad groups as listed below:

Facial landmarking: Facial landmarking is an essential part of frontalization, as the landmarks define the facial coordinates. Over the past decade, an array of landmarking techniques have been developed that rely on handcrafted features [15]. Recently, deep learning has been used for training and regression [17]. Current algorithms provide landmark sets of size between 7 and 194 points. Of late, landmarkers have begun to conform to a 68-point standard to improve comparative analysis between algorithms and across different landmarking challenges and datasets [14], [15], [18].

Face frontalization: Once the facial landmarks are detected on a non-frontal face, frontalization can be performed using one of the two main approaches. The first approach utilizes unique 3D models for each face in the gallery, either inferred statistically [19], collected at acquisition time [20], or generic [13]. Once the image is mapped to a 3D model, matching can be performed by either reposing the gallery image to match the pose of the query image or the query image can be frontalized [21]. These methods have been utilized in breakthrough recognition algorithms [22]. The second approach uses statistical models to infer a frontal view of the face by minimizing off-pose faces to their lowest rank reconstruction [23]. Additionally, methods have been explored for inferring frontal faces using deep learning [24].

Face recognition: Face recognition has been explored by researchers for decades. In its infancy, researchers used handcrafted features and descriptors for representing faces [25]. More recently, state-of-the-art Deep CNN methods for face recognition have achieved near-perfect recognition scores on the once-challenging LFW dataset [4]. While some of these methods concentrate on creating novel network architectures [8], [26], others focus on feeding a large pool of data to the network training stage [22], [2]. Researchers have now shifted their attention to the more challenging problem of face recognition from videos. The Youtube Faces (YTF) dataset [27], IJB-A [12] and PaSC [16] exemplify both unconstained and controlled video settings. Researchers have developed multi pose based CNN models [10], [9] or used face image frontalization as an data augmentation step [11], for recognizing faces from these challenging video datasets.

III. DESCRIPTION OF CHOSEN LANDMARKING AND FRONTALIZATION METHODS

Here we present brief descriptions of the face landmarking and frontalization techniques used in this paper.

A. Landmarking

Zhu and Ramanan (ZR) [14]: This method allows for simultaneous face detection, landmarking, and pose detection, accommodating up to 175 degrees of facial yaw. It uses a mixture of trees approach, similar to that of phylogenetic inference. The algorithm proposed in [28] is used to optimize the tree structure with maximum likelihood calculations based on training priors. Due to the algorithm performing localization and landmarking concurrently, it is comparatively slow.

Kazemi and Sullivan (KS) [15]: This method uses a cascade of multiple regressors to estimate landmark points on the face using only a small, sparse subset of pixel intensities from the image. This unique subsampling renders it extremely fast, while maintaining a high level of accuracy. This landmarker is popular due to its ease of use and availability — it is implemented in the widely used Dlib library [29].

Cascade Mixture of Regressors (CMR): We introduce this method as a fast performing alternative for facial landmarking that works well on lower-resolution images in datasets such as CASIA-Webface and PaSC. Similar to the Supervised Descent Method [30], this method initializes its set of landmarks in a defined initial formation around the detected face. Then a regression function in the form of a Gaussian mixture of regressors is applied to local features extracted from the initial landmark locations. The landmark positions are fine-tuned iteratively using a cascade of mixture regressors similar to [3].

B. Frontalization

Hassner et al. (H) [13]: This method allows 2D face images to be frontalized without any prior 3D knowledge. We chose to analyze this method due to its prominence in the facial biometrics community, and because an open source implementation of the algorithm exists. Using a set of reference 3D facial landmark points determined by a template 3D face model, the 2D facial landmarks detected in an input image are projected into the 3D space. A 3D camera homography is then estimated between them. Back-projection is then used to map pixel intensities from the face onto the canonical, frontal template. Optional soft symmetry is then applied as a post-processing method by replacing areas of the face that are self-occluded. Figure 2 shows

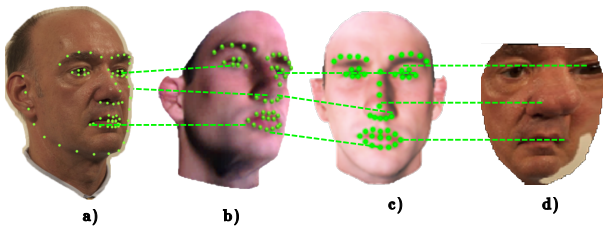

Fig. 2: Frontalization correspondences for the method of Hassner et al. [13]. a) Input face in non-frontal pose. b) Face matched to template pose via landmarks. c) Back-projection of template from input pose to frontal. d) Back-projection of pixel values to frontalized template, including artifacts from self-occlusion

the general mapping procedure for projecting non-frontal faces onto a 3D model. This method projects pixels onto the template in a global, non-piecewise manner. Due to this, local artifacts from self-occlusions can get streched and smeared across a frontalized face. This can cause loss of high-frequency features used for matching.

IV. A NEW METHOD FOR SINGLE-IMAGE FRONTALIZATION

In this section, we present our proposed frontalization procedure, which is capable of synthesizing a frontalized face image from a single input image with arbitrary facial orientation without requiring a subject-specific 3D model.

A. Face Detection, Landmarking and Model Fitting

Our frontalization procedure starts (see Fig. 3 (a)) by detecting the facial region in the input image I_0 using the Viola-Jones face detector [31]. Using the CMR method, we detect 68 facial landmark points, i.e., $\boldsymbol{p}_0 = [x_1, y_1, \ldots, x_{68}, y_{68}]^\top \in \mathbb{R}^{2\cdot 68\times 1}$. The landmarks can be used to determine the pose and orientation of the processed face. We crop the facial area, I_c, from the input image based on the detected landmarks and use it as the basis for frontalization.

To transform the face in the input image to a frontal pose, we require a depth estimate for each of the pixels in the cropped facial area. To this end, we use a generic 3D face model and fit it to the cropped image I_c. Our 3D model is a frontal depth image I_r from the FRGC dataset [25] manually annotated with the same 68 landmarks as detected by the CMR procedure. We fit the 3D model to the cropped image through a piece-wise warping procedure guided by the Delaunay triangulation of the annotated landmarks. Since the annotated landmarks reside in a 3D space, i.e., $\boldsymbol{p}_r = [x_1, y_1, z_1 \ldots, x_{68}, y_{68}, z_{68}]^\top \in \mathbb{R}^{3\cdot 68\times 1}$, we use only the 2D coordinates in the XY-plane to compute the triangulation. The fitting procedure then aligns the generic 3D model with the shape of the cropped image and provides the depth information needed for the 3D transformation of the input face to a frontal pose (see Fig. 3 (b)). The depth information generated by the warping procedure represents only a rough estimate of the true values, but, as we show later, is sufficient to produce visually convincing frontalization results.

B. 3D Transformation and Texture Mapping

After the fitting process, we use the landmarks $\boldsymbol{p}_a \in \mathbb{R}^{3\cdot 68\times 1}$ corresponding to the aligned 3D model I_a and the landmarks $\boldsymbol{p}_r \in \mathbb{R}^{3\cdot 68\times 1}$ of the generic 3D face model to estimate a 3D transformation, $\boldsymbol{T} \in \mathbb{R}^{4\times 4}$, that maps the fitted model I_a back to frontal pose (Fig. 3 (c)). We use Horn's quaternion based method [32] to calculate the necessary scaling, rotation and translation to align the 3D points in $\boldsymbol{p}_a$ and $\boldsymbol{p}_r$ and construct the transformation matrix $\boldsymbol{T}$. Any given point of the aligned 3D model $\boldsymbol{P} = [X, Y, Z, 1]^\top$ can then be mapped to a new point in 3D space based on the following expression:

$$\boldsymbol{P'} = \boldsymbol{T}\boldsymbol{P}, \tag{1}$$

where $\boldsymbol{P'} = [X', Y', Z', 1]^\top$ represents a point of the frontalized 3D model I_f (see Fig. 3 (d)).

The cropped image I_c and the aligned model I_a are defined over the same XY-grid. The known 2D-to-3D point correspondences can, therefore, be exploited to map the texture from the arbitrarily posed image I_c to its frontalized form I_t. Values missing from I_t after the mapping are filled in by interpolation. The results of the presented procedure are shown in Fig. 3 (d). Here, images in the the upper row illustrate the transformation of the 3D models in accordance with $\boldsymbol{T}$, while the lower row depicts the corresponding texture mapping. The mapped texture image I_t represents an initial frontal view of the input face, but is distorted in some areas. We correct for these distortions with the postprocessing steps described in the next section.

C. Image Correction and Postprocessing

Similar to the method of [13], our approach utilizes a generic 3D face model to generate frontalized face images, but we adapt our model in accordance with the shape of the input face to ensure a better fit. Triangulation is performed on the input face landmark coordinates. Each triangle is then mapped back to the generic 3D face model, and an affine transform is calculated per-triangle. Because the piecewise alignment is performed with a warping procedure, minor distortions are introduced into the shape of the aligned 3D model, which lead to artifacts in the mapped texture image I_t. Additional artifacts are also introduced by the interpolation procedure needed to compensate for the obscured or occluded areas in the input images caused by in-plane rotations and self-occlusions.

We correct for the outlined issues by analyzing the frontalized 3D model I_f. Since Eq. (1) defines a mapping from I_a to I_f, the frontlized 3D model I_f is not necessarily defined over a rectangular grid, but in general represents a point cloud with areas of different point density. We idenitfy obscured pixels in I_a based on point densities. If the density for a given pixel falls below a particular threshold, we mirror the corresponding pixel from the other side of the face to form a more symmetric face.

The effect of the presented image correction procedure is illustrated in Fig. 3 (e). The image, marked as I_m, contains white patches that were identified as being occluded in I_a,

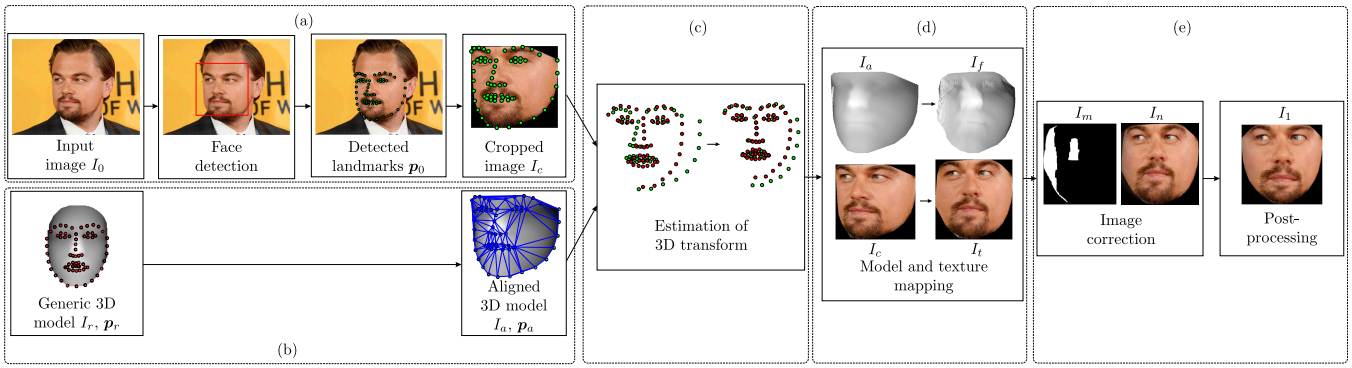

Fig. 3: Overview of the proposed frontalization procedure. The procedure first detects the facial area and a number of facial landmarks in the input image (a). It then aligns a generic 3D model with the input face (b) and calculates a 3D transform that maps the aligned 3D model back to frontal pose (c). Based on the known 2D-to-3D point correspondences, a synthetic frontal view of the input face is generated (d) and post-processed to generate the final results of the frontalization (e).

while I_n represents the corrected image with pixels mirrored from one side of the face to the other (examine the difference in the appearance of the nostrils between I_t and I_n). In the last step of our frontalization procedure we map the image I_n to a predefined mean shape. This mapping generates the final frontalized output I_1 of our procedure and is shown in the last image of Fig. 3 (e).

V. FACE RECOGNITION PIPELINE

In this section, we provide details about our face recognition pipeline. This includes the training and testing data, pre-processing methods, network architecture, and the scoring protocol. We also discuss the rationale behind choosing each component and method. A schematic representation of our pipeline can be found in Fig. 4.

A. Training Data: CASIA-WebFace

The CASIA-WebFace [7] dataset contains 494,414 face images of 10,575 subjects, with 46 face images per subject on average. The dataset contains face images of varying gender, age, ethnicity and poses. It was originally released for training CNNs and reporting performance on LFW [4]. In comparison, the MegaFace [6] and VGG-FACE [8] datasets both contain over a million face images. However, the average number of images per subject is too low (1.5) for MegaFace [11], and VGG-FACE contains many labeling errors. Therefore, we decided not use either MegaFace or VGG-FACE for training. Instead, we chose a subset of CASIA-WebFace, containing 303,481 face images of 7,577 subjects, as our training dataset. This subset was used instead of the whole dataset to account for time and resource constraints. Sample images can be seen in Fig. 4.

B. Pre-processing Methods

The pre-processing schemes for the training data in our experiments were comprised of different combinations of landmarkers and frontalizers described in Secs. III and IV - 1) Zhu and Ramanan (ZR) and Hassner et al. (H), 2) Kazemi and Sullivan (KS) and Hassner et al. (H), and 3) CMR and our frontalization method. Output of the frontalization techniques on sample images can be seen in Fig. 5.

In addition, we compared these methods to three baseline approaches: 1) Training VGG-FACE with only 2D aligned CASIA-WebFace images, rotated using eye-centers, *i.e.* no frontalization (Figure 1.b). 2) Training VGG-FACE with original CASIA-WebFace images, *i.e.* no pre-processing. 3) Instead of training our own version of the VGG-FACE model, we used a pre-trained snapshot of the VGG-FACE model as a feature extractor *i.e.*, no additional training. This snapshot was trained with 2D aligned face images from the VGG-FACE dataset. The last baseline method was used to evaluate how much the additional training on CASIA-WebFace improved the face representation capability of the VGG-FACE CNN model.

C. CNN architecture: VGG-FACE

The CNN architecture in this work is similar to the one described by Parkhi et al. [8], a variant of the 16 layer model proposed by Simonyan and Zisserman [34]. The architecture is comprised of linear convolution layers with non-linear operators including ReLU and max pooling filters in between the convolutional layers. These are followed by three fully connected (FC) layers with a filter size that is the same as the input image (224×224). The first two FC layers (denoted as *fc6* and *fc7*) are 4,096 dimensional, while the size of the final FC layer (*fc8*) depends on the number of classes in the training data.

The reasons for choosing this particular 16 layer architecture are three-fold: 1) it generates verification results comparable to Google FaceNet [2] on LFW [4] while requiring a fraction of its training data. 2) The model does reasonably well in identifying faces from YTF [27] videos. 3) A snapshot of this model, pre-trained on 2 million face images, is present in the Caffe [33] model zoo[1]. We used this pre-trained model to fine-tune connection weights in our training experiments for faster convergence [35].

[1] https://github.com/BVLC/caffe/wiki/Model-Zoo

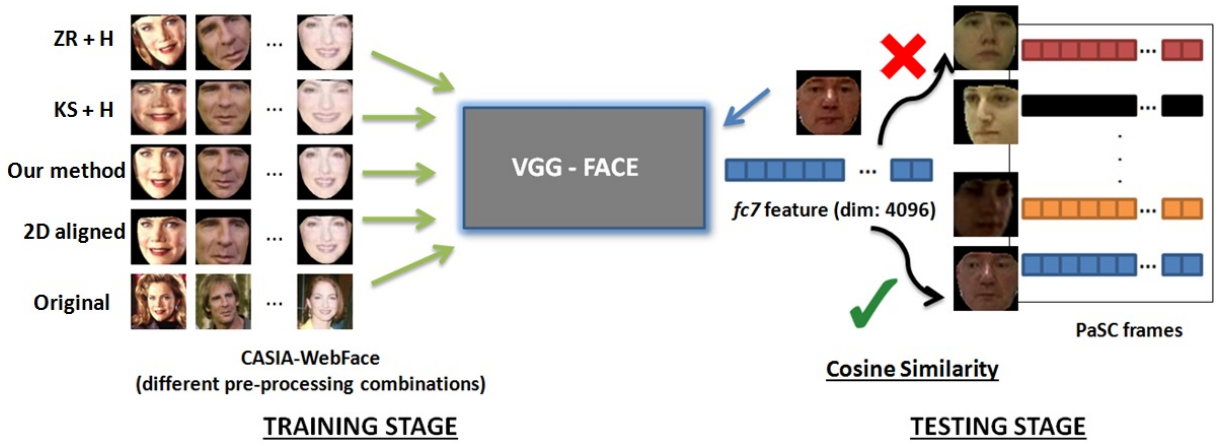

Fig. 4: The face recognition pipeline: the training phase is on the left and the testing phase is on the right of the image.

D. Testing Dataset: PaSC

The PaSC dataset [16] is a collection of videos acquired at the University of Notre Dame over seven weeks in the Spring semester of 2011. The human subjects in each video clip performed different pre-determined action each week. The actions were captured using two cameras (handeld and tripod mounted) simultaneously. The handheld videos were acquired with five different cameras, while tripod mounted videos were shot with the same camera. The dataset contains 1,401 videos from handheld cameras and 1,401 videos from the tripod mounted camera. A small training set of 280 videos is also available with the dataset.

While both YTF [27] and IJB-A [12] are well-established video datasets, they are collections of video data from the Internet. On the other hand, PaSC solely consists of video sequences physically collected specifically for the task of video face recognition. This type of controlled acquisition is beneficial for our video-to-video matching-based evaluation.

E. Feature Extraction and Scoring

We used networks trained on data pre-processed with each of the combinations mentioned above as feature extractors for PaSC video frames. The face region from each frame was extracted beforehand using the bounding box provided with the dataset. Bad detections were filtered out by taking the default track coordinates of the faces and removing detections that were outside a 2.5 sigma distance range from the assumed face track. After this pruning process, a total of 1,348 and 1,378 videos were left for tripod-mounted and handheld videos respectively. We used our simple 2D-alignment using eye centers to align the PaSC Dataset. Different preprocessing methods lead to significantly different PaSC testing set yields. To keep a uniform testing set for all experiments, the 2D-aligned PaSC data was used for testing on all CNNs trained with different preprocessing methods

A 4,096 dimensional feature vecture was then extraced at the *fc7* layer for every face image using each CNN model. Once all feature vectors for all frames were collected, we computed the accumulated feature-wise means at each dimension to generate a single representative vector for that video. This accumulated vector can be represented as [f_1, f_2, f_3, ..., f_{4096}], such that

$$f_k = \frac{1}{N}\sum_{i=1}^{N}(v_k)_i \tag{2}$$

where $(v_k)_i$ is the k-th feature in frame i of the video and N is the total number of frames in that video.

Once we had the representative vector for all the videos, we used cosine similarity to compute match scores between different accumulated feature vectors from two different videos, as shown below:

$$S(u,v) = \frac{u \cdot v}{\|u\|_2 \, \|v\|_2} \tag{3}$$

where *S(u,v)* is the similarity score between the two vectors u = [f_1, f_2, f_3, ..., f_{4096}] and v = [f_1, f_2, f_3, ..., f_{4096}]. If u and v are perfectly alike (0 angle), then *S(u,v)* is 1. Ideally, two videos (vectors) of the same subject should generate a cosine similarity score of approximately 1, *i.e.*, a true match. Cosine similarity is a common scoring approach for video-to-video matching, and we chose to use it here because

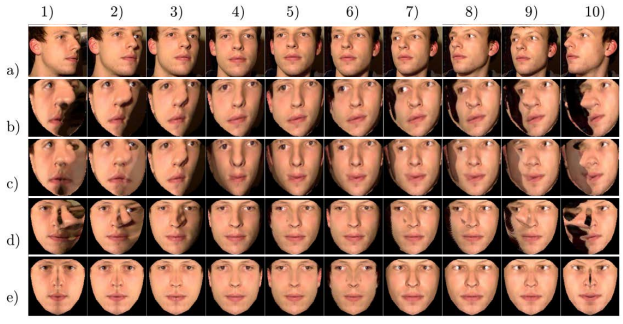

Fig. 5: Difference in frontalization techniques. Left-to-right: a set of faces with different degrees of yaw. Top-to-bottom: a) original input face image. b) H (asymmetric). c) H (symmetric). d) Our method (asymmetric). e) Our method (hard symmetric).

TABLE I: Yield of each pre-processing method

Pre-processing method	**CASIA-WebFace images** (# of subjects)	**Yield** (%)
ZR & H (asymmetric)	223,894 (6762)	73.77
ZR & H (symmetric)	254,377 (7,576)	83.81
KS & H (asymmetric)	294,020 (7,576)	96.88
KS & H (symmetric)	293,920 (7,576)	96.84
CMR & our method (asymmetric)	240,164 (7,369)	79.13
CMR & our method (hard-symmetric)	240,164 (7,365)	79.13
2D alignment (not frontalized)	260,882 (7,576)	85.96

of its prevelance in the 2016 PaSC challenge [16]. These similarity scores were used for calculating the verification and identification accuracy rates of each CNN.

VI. EXPERIMENTS AND RESULTS

Here we present details about our experiments and the subsequent results.

A. Methodology

To analyze the effect that facial frontalization has on recognition performance, we performed two experiments: 1) training VGG-FACE with face images pre-processed with different strategies, then analyzing its performance on only the 2D aligned PaSC dataset, *i.e.*, frontalization at training time, and 2) frontalizing the PaSC dataset as well *i.e.*, frontalization both at training and testing time.

For each frontalization method, we kept two pre-processed versions of the same face: one without any symmetry (asymmetric), such as the lefthand side of Figure 1 (c), and the other with symmetry, where one vertical half is used for both sides of the face, as in the righthand side of Figure 1 (c). For the symmetric versions, the half to replicate was chosen automatically based on the quality of the facial landmark points. However, for our own pre-processing method, we decided to replicate only the left half of the face for the symmetric version (hard-symmetry), as shown in the right part of Figure 1 (e). This was done to evaluate the effect of the automatic quality based replication process, used for the other two frontalization strategies.

Many samples in CASIA-WebFace proved difficult to calculate landmark detections due to scale, extreme pose, or occlusion. These samples were disgarded from the training set; subsequently, most pre-processing methods yielded an image count well below the total number of images originally in the dataset. The yield of frontalized faces varied for each combination as can be seen in Table I.

We trained a VGG-FACE model separately for each set of training data pre-processed with a given method. For each

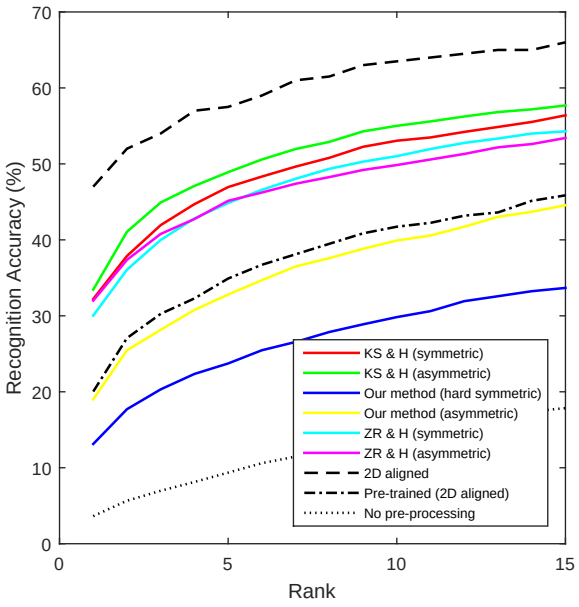

Fig. 6: Identification performance on 2D aligned handheld PaSC videos. The pre-trained VGG-FACE model (red) was trained *a priori* with 2D aligned face images from the VGG-FACE dataset. The network trained on 2D aligned data outperformed all other methods in this experiment.

method, we randomly split 80% of the CASIA-WebFace data into training and the rest into validation sets. A single NVIDIA Titan X GPU was used to run all of our training experiments using the Caffe [33] deep learning package. Network weights were initialized from a snapshot of VGG-FACE pre-trained on 2 million face images [35].

We used Stochastic Gradient Descent [38] for CNN training. The hyperparameters for this method were selected using the HyperOpt software package [36]. Once one set of hyperparameters was calculated, the same set was repeated across the different experiments to maintain consistency. The base learning rate was set to 0.01, which was multiplied by a factor of 0.1 (gamma) following a stepwise learning policy, with a step size set to 50,000 training iterations. The training batch size was set to 64, where the training image resolution was 224×224 for the VGG-FACE model. We set a hard bound of 50 epochs, after which training was terminated. Convergence was reached within the 50 epochs for all the different training regimes. The snapshot at 50 epochs was then used for feature extraction in the testing phase.

B. Results

For the first set of experiments, we used 2D aligned face images from extracted from PaSC videos. We did two tests on the tripod-mounted[2] and handheld images separately. Each experiment consisted of two parts: 1) We computed the verification accuracy using a ROC curve, as done traditionally on PaSC. 2) We computed the rank-based recongition accuracy, *i.e.*, identification using a CMC curve. We felt both

[2]Check last page for results on tripod mounted videos.

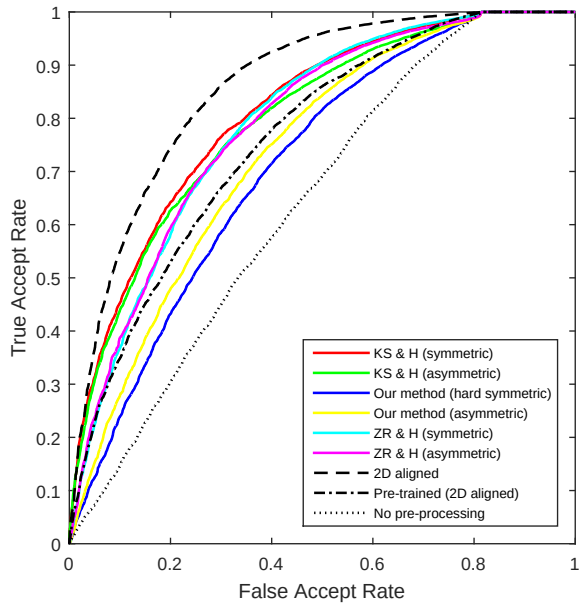

Fig. 7: Verification performance on 2D aligned handheld PaSC videos. The 2D aligned curve gave the best performance, while training with the non pre-processed images actually hampered the CNN's face representation capability (dotted curve).

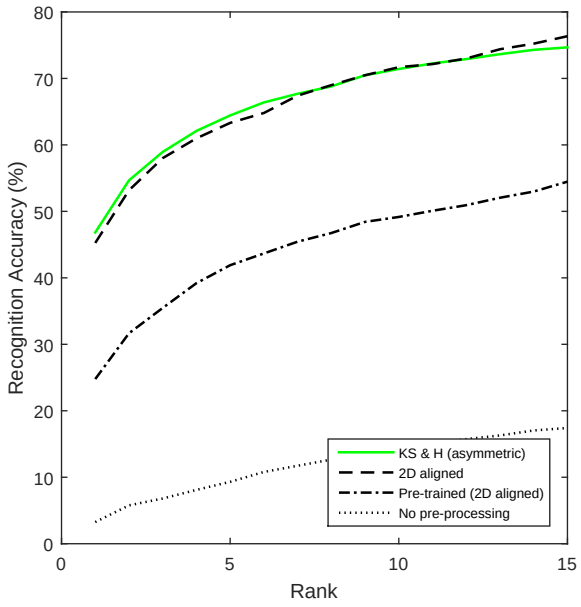

Fig. 8: Identification performance on frontalized (method: KS and H, asymmetric) handheld PaSC videos. The CNNs trained on frontalized and 2D aligned face images both gave similar recognition accuracies.

verification and identification performance measures were pertinent in analyzing the behavior of each frontalization scheme. For the handheld data, the identification and verification performance of the different CNN models can be seen in Figures 6 and 7 respectively.

The VGG-FACE model trained on 2D aligned face images outperformed every other pre-processing and baseline method. The asymmetric version of the Kazemi and Sullivan (KS) and Hassner et al. (H) combination performed the best among the different frontalization techniques. Interestingly, our frontalization method performed worst among the pre-processing methods, although face images frontalized using it look natural to the human eye. We suspect the warping mechanism in our frontalization method to be responsible for this. However, note that it is much better than doing no preprocessing whatsoever — a dominant mode of operation for CNN-based face recognition.

To evaluate the effect of frontalization at the time of testing, we frontalized faces extracted from the PaSC frames using the asymmetric version of the KS and H combination. This was the best frontalization combination from the first set of experiments. However, the yield was poor for frontalization - only 1074 handheld and 1166 tripod mounted of the 1401 PaSC videos had at least one frontalized frame of sufficient quality. VGG-FACE models trained on this combination and the three baselines were used for feature extraction. The identification and verification performance of these models can be found in Figures 8 and 9 respectively. As can be seen, the CNN trained with 2D aligned images does nearly well as that trained with frontalized images.

VII. DISCUSSION

Several key observations can be drawn from our experiments and used inform future face recognition experiments:

1) Frontalization comes at the cost of loss of data, *i.e.*, low yield. This loss is further amplified when the face images to be frontalized are from drawn from video clips instead of still images.

2) A simple 2D alignment procedure is a computationally cheaper pre-processing option for face images used to train CNNs, compared to face image frontalization. Features extracted using a CNN trained with 2D aligned face images yield better recognition accuracy than a CNN trained with frontalized face images when the face images used for testing are not frontalized.

3) Frontalizing both the training and testing face images, although computationally much more expensive, does not significantly improve the recognition performance when compared to a simple 2D alignment process.

4) Asymmetrically frontalized face images tend to boost a CNN's face representation capability slightly when compared with symmetrically frontalized faces. Our experiments also showed that setting a hard symmetry function, *i.e.*, always replicating one particular half of the face, generates visually unnatural results. And training a CNN with these unnaturally frontalized faces might even hamper its face representation capability.

5) Results generated from one frontalization method might look superior to the human eye than the results generated from another frontalization method. However, that does not guarantee that a CNN trained with face images frontalized using the first method will represent faces better than a CNN

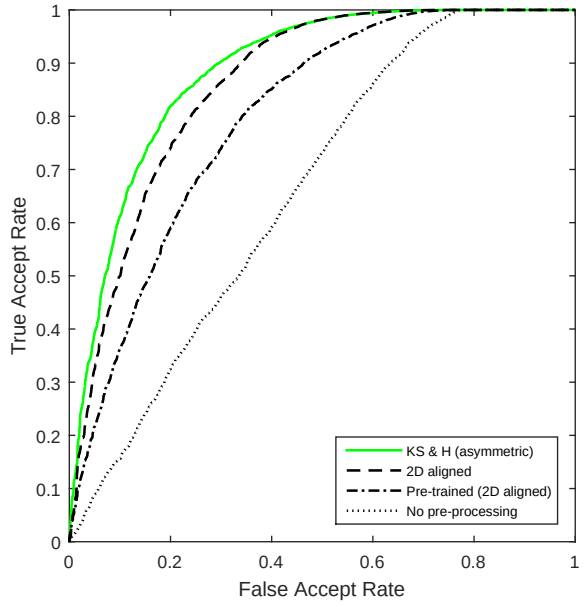

Fig. 9: Verification performance on frontalized (method: KS and H, asymmetric) handheld PaSC videos. The CNNs trained on 2D aligned and frontalized face images, both generated identical curves.

trained with face images frontalized using the second. In fact, its performance might dip in some cases. For example, the face images frontalized using the Kazemi and Sullivan and Hassner et al. method looked far less realistic than those frontalized with CMR and our method but the VGG-FACE model trained on face images frontalized with the first method generated far better results than the second method. This echoes one of the critical findings from the NIST FRVT 2006 evaluation [37]: "Quality is not in the eye of the beholder; it is in the recognition performance figures!"

Acknowledgements: Hardware support was generously provided by the NVIDIA Corporation.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning", in *Nature*, 521(7553), pp. 436 - 444, 2015.
[2] F. Schroff, D. Kalenichenko and J. Philbin, "Facenet: A unified embedding for face recognition and clustering", in Proc. *IEEE CVPR*, 2015.
[3] O. Tuzel, S. Tambe and T. K. Marks, "Robust Face Alignment Using a Mixture of Invariant Experts", in Proc. *ECCV*, 2016.
[4] 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", Technical Report 07-49, UMass, Amherst, 2007.
[5] E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li and G. Hua, "Labeled Faces in the Wild: A Survey", in *Advances in Face Detection and Facial Image Analysis (Springer)*, pp. 189 - 248, 2016.
[6] I. Kemelmacher-Shlizerman, S. Seitz, D. Miller and E. Brossard, "The megaface benchmark: 1 million faces for recognition at scale", in Proc. *IEEE CVPR*, 2016.
[7] D. Yi, Z. Lei, S. Liao and S. Z. Li, "Learning face representation from scratch", *arXiv preprint arXiv:1411.7923*, 2014.
[8] O. M. Parkhi, A. Vedaldi and A. Zisserman, "Deep face recognition", in Proc. *BMVC*, 2015.
[9] W. AbdAlmageed, Y. Wu, S. Rawls, S. Harel, T. Hassner, I. Masi, J. Choi, J. Lekust, J. Kim, P. Natarajan, R. Nevatia and G. Medioni,"Face Recognition Using Deep Multi-Pose Representations", in Proc. *IEEE WACV*, 2016.
[10] I. Masi, S. Rawls, G. Medioni and P. Natarajan, "Pose-Aware Face Recognition in the Wild", in Proc. *IEEE CVPR*, 2016.
[11] I. Masi, A. T. Trãn, Anh Tuan, T. Hassner, J. Leksut and G. Medioni,"Do We Really Need to Collect Millions of Faces for Effective Face Recognition?", in Proc. *ECCV*, 2016.
[12] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, M. Burge and A. K. Jain, "Pushing the frontiers of unconstrained face detection and recognition: IARPA Janus Benchmark A", in Proc. *IEEE CVPR*, 2015.
[13] T. Hassner, S. Harel, E. Paz and E. Enbar, "Effective face frontalization in unconstrained images", in Proc. *IEEE CVPR*, 2015.
[14] X. Zhu and D. Ramanan, "Face detection, pose estimation and landmark localization in the wild", in Proc. *IEEE CVPR*, 2012.
[15] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees", in Proc. *IEEE CVPR*, 2014.
[16] W. Scheirer et al.,"Report on the BTAS 2016 Video Person Recognition Evaluation", in Proc. *IEEE BTAS*, 2016.
[17] Y. Wu and T. Hassner, "Facial Landmark Detection with Tweaked Convolutional Neural Networks", *arXiv preprint arXiv:1511.04031*, 2016.
[18] J. Shen, S. Zafeiriou, G. Chrysos, J. Kossaifi, G. Tzimiropoulos and M. Pantic, "The first facial landmark tracking in-the-wild challenge: Benchmark and results", in Proc. *IEEE ICCV Workshops*, 2015.
[19] T. Hassner, "Viewing real-world faces in 3D", in Proc. *IEEE CVPR*, 2013.
[20] V. Blanz and T. Vetter, "Face recognition based on fitting a 3D morphable model", in *IEEE TPAMI*, 25(9), pp. 1063 - 1074, 2003.
[21] X. Zhang and Y. Gao, "Face recognition across pose: A review", in *Pattern Recognition*, 42(11), pp. 2876 - 2896, 2009.
[22] Y. Taigman, M. Yang, M. Ranzato and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification", in Proc. *IEEE CVPR*, 2014.
[23] C. Sagonas, Y. Panagakis, S. Zafeiriou and M. Pantic, "Robust Statistical Face Frontalization", in Proc. *IEEE ICCV*, 2015.
[24] J. Yim, H. Jung, B. Yoo, C. Choi. D. Park and J. Kim, "Rotating your face using multi-task deep neural network", in Proc. *IEEE CVPR*, 2015.
[25] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min and W. Worek, "Overview of the face recognition grand challenge", in Proc. *IEEE CVPR*, 2005.
[26] Y. Sun, D. Liang, X. Wang and X. Tang, "Deepid3: Face recognition with very deep neural networks", in *arXiv preprint arXiv:1502.00873*, 2015.
[27] L. Wolf, T. Hassner and I. Maoz, "Face recognition in unconstrained videos with matched background similarity", in Proc. *IEEE CVPR*, 2011.
[28] S. Kirshner, P. Smyth, A. W. Robertson, "Conditional Chow-Liu tree structures for modeling discrete-valued vector time series", in Proc. *UAI*, 2004.
[29] D. E. King, "Dlib-ml: A machine learning toolkit", in *JMLR*, vol. 10, pp. 1755 - 1758, 2009.
[30] X. Xiong and F. De la Torre, "Supervised descent method and its applications to face alignment", in Proc. *IEEE CVPR*, 2013.
[31] P. Viola and M. J. Jones, "Robust Real-Time Face Detection", in *IJCV*, 57(2), pp. 137 - 154, 2004.
[32] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions", in *JOSA A*, 4(4), pp. 629 - 642, 1987.
[33] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama and T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding", in Proc. *ACM MM*, 2014.
[34] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", in *ICLR*, 2015.
[35] J. Yosinski, J. Clune, Y. Bengio and H. Lipson, "How transferable are features in deep neural networks?", in Proc. *NIPS*, 2014.
[36] J. Bergstra, D. Yamins and D. D. Cox, "Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms", in Proc. *SciPy*, 2013.
[37] J. R. Beveridge, G. H. Givens, P. J. Phillips, B. A. Draper and Y. M. Lui, "Focus on quality, predicting FRVT 2006 performance", in Proc. *IEEE FG*, 2008.
[38] L. Bottou, "Large-scale machine learning with stochastic gradient descent", in Proc. *COMPSTAT*, 2010.

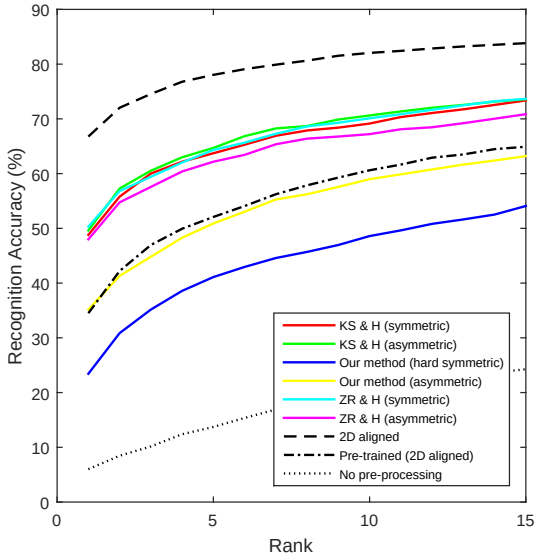

Fig. 10: Identification performance on 2D aligned tripod mounted PaSC videos.

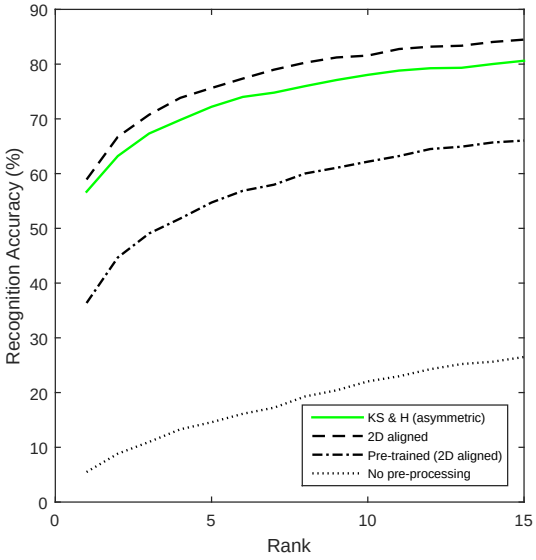

Fig. 12: Identification performance on frontalized (method: KS and H, asymmetric) tripod mounted PaSC videos.

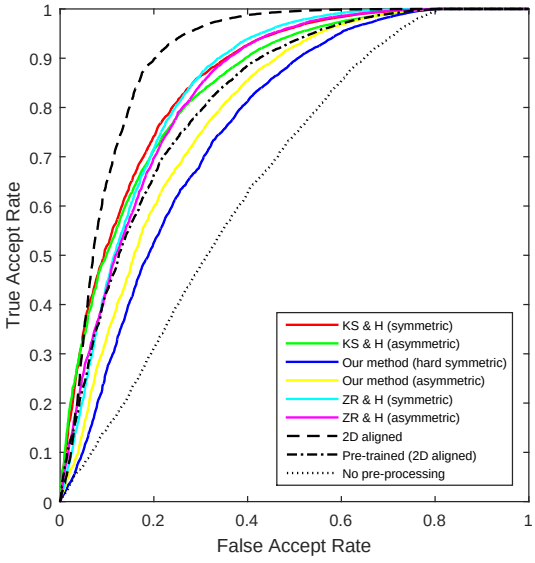

Fig. 11: Verification performance on 2D aligned tripod mounted PaSC videos.

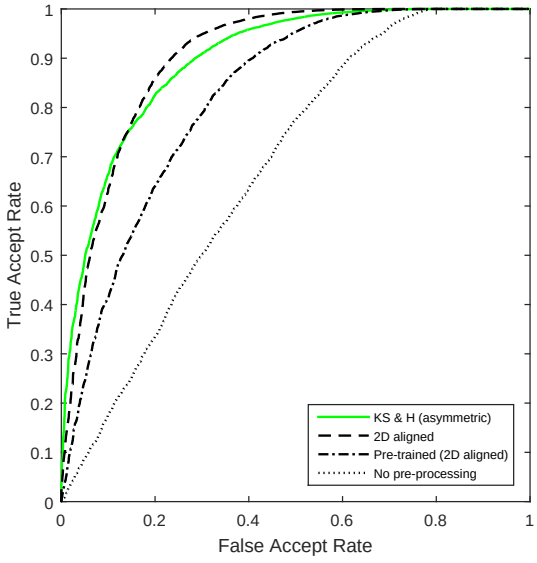

Fig. 13: Verification performance on frontalized (method: KS and H, asymmetric) tripod mounted PaSC videos.