Enhancing Convolutional Neural Networks for Face Recognition with Occlusion Maps and Batch Triplet Loss

Daniel Sáez Trigueros, Li Meng
School of Engineering and Technology
University of Hertfordshire
{d.saez-trigueros, l.1.meng}@herts.ac.uk

Margaret Hartnett
IDscan Biometrics

Abstract

Despite the recent success of convolutional neural networks for computer vision applications, unconstrained face recognition remains a challenge. In this work, we make two contributions to the field. Firstly, we consider the problem of face recognition with partial occlusions and show how current approaches might suffer significant performance degradation when dealing with this kind of face images. We propose a simple method to find out which parts of the human face are more important to achieve a high recognition rate, and use that information during training to force a convolutional neural network to learn discriminative features from all the face regions more equally, including those that typical approaches tend to pay less attention to. We test the accuracy of the proposed method when dealing with real-life occlusions using the AR face database. Secondly, we propose a novel loss function called batch triplet loss that improves the performance of the triplet loss by adding an extra term to the loss function to cause minimisation of the standard deviation of both positive and negative scores. We show consistent improvement in the Labeled Faces in the Wild (LFW) benchmark by combining both proposed adjustments to the convolutional neural network training.

1. Introduction

Deep learning models, in particular convolutional neural networks (CNNs), have revolutionised many computer vision applications, including face recognition. As recent benchmarks [1, 2, 3] show, most of the top performing face recognition algorithms are based on CNNs. Even though these models need to be trained with hundreds of thousands of faces to achieve state-of-the-art accuracy, several large-scale face datasets [4, 5, 3] have recently been made publicly available to facilitate this.

Most of the recent research in the field has focused on unconstrained face recognition. CNN models have shown excellent performance on this task, as they are able to extract features that are robust to variations present in the training data (if enough samples containing these variations are provided). Nonetheless, in this work, we show how partial facial occlusions remain a problem for unconstrained face recognition. This is because most databases used for training do not present enough occluded faces for a CNN to learn how to deal with them. Common sources of occlusion include sunglasses, hats, scarves, hair, or any object between the face and the camera. This is of particular relevance to applications where the subjects are not expected to be co-operative (e.g. in security applications). One way of overcoming this problem is to train CNN models with datasets that contain more occluded faces. However, this task can be challenging because the main source of face images is usually the web, where labelled faces with occlusions are less abundant.

Bearing this in mind, we propose a novel data augmentation approach for generating occluded face images in a strategic manner. We use a technique similar to the occlusion sensitivity experiment proposed in [6] to identify the face regions where a CNN extracts the most discriminative features from. In our proposed method, the identified face regions are covered during training to force a CNN to extract discriminative features from the non-occluded face regions with the goal of reducing the model's reliance on the identified face regions. Our CNN models trained using this approach have demonstrated noticeable performance improvement on face images presenting real-life facial occlusions in the AR face dataset [7]

CNN models for face recognition can be trained using different approaches. One of them consists of treating the problem as a classification one, wherein each identity in the training set corresponds to a class. Af-

ter training, the model can be used to recognise faces that are not present in the training set by discarding the classification layer and using the features of the previous layer as the face representation. In the realm of deep learning, these features are commonly referred to as bottleneck features. Following this first training stage, the model can be further trained using other techniques to optimise the bottleneck features for the target application [4, 8]. Another common approach to learning face representation is to directly learn bottleneck features by optimising a distance metric between pairs of faces [9, 10] or triplets of faces [11].

Positive results have been demonstrated when combining these two techniques, either by (i) jointly training with a classification loss and a distance metric loss [12]; or (ii) by first training with a classification loss and then fine-tuning the CNN model with a distance metric loss [5, 13, 14]. In this work, we adopt the latter approach and use the triplet loss to optimise bottleneck features. The goal of the triplet loss is to separate positive scores (obtained when comparing pairs of faces belonging to the same subject) from negative scores (obtained when comparing pairs of faces belonging to different subjects) by a minimum margin. We argue that training with this loss function can lead to undesired results. Thus, we propose a new loss function to alleviate this issue by also minimising the standard deviation of both positive and negative scores. Using the Labeled Faces in the Wild (LFW) benchmark, we show that the CNN models trained with our proposed loss function consistently outperform those trained with the triplet loss function.

The remainder of this paper is organised as follows. Section 2 provides a review of the related work, with a focus on deep learning approaches and face recognition with occlusion. Section 3 details our CNN architecture, training procedure and (i) our method of improving recognition of partially occluded faces; and (ii) our novel loss function. Section 4 describes our experimental results, and our conclusions are presented in Section 5.

2. Related Work

One of the first successful applications of convolutional neural networks was handwritten character recognition [15]. Soon after, the first face recognition algorithm that included a CNN was proposed in [16]. However, unlike [15], their algorithm [16] was not entirely based on neural networks. Years later, [9] proposed an end-to-end Siamese architecture trained with a contrastive loss function to directly minimise the distance between pairs of faces from the same subject while increasing the distance between pairs of faces from different subjects. These CNN-based face recognition models did not achieve groundbreaking results, mainly due to the low capacity of the networks used and the relatively small datasets available for training at the time. It was not until these models were scaled up and trained with large amounts of data [17] that CNNs became the state-of-the-art approach for face recognition.

In particular, Facebook's DeepFace [18], one of the first CNN-based approaches for face recognition that used a high capacity model, achieved an accuracy of 97.35% on the LFW benchmark, reducing the error of the previous state-of-the-art by 27%. DeepFace used an effective 3D alignment algorithm to frontalise faces before feeding them to a CNN with several convolutional, max-pooling and locally connected layers. The CNN was trained with a dataset containing 4.4 million faces from 4,030 subjects. Concurrently, the DeepID system [8] achieved similar results by concatenating the bottleneck features of 60 CNNs trained on different face crops and optimising the concatenated feature vector using the Joint Bayesian method proposed in [19]. More work by the same authors [12] achieved further performance improvements by simultaneously training with a contrastive loss (similar to the one used in [9]) and a classification loss. The authors claimed that the contrastive loss reduced intra-personal variations and the classification loss increased inter-personal variations. The described system achieved an accuracy of 99.15% on the LFW benchmark using a relatively small training set containing 202,599 face images of 10,177 identities.

As shown in [20], training data is one of the most important factors for increasing the accuracy of CNN-based approaches. In particular, it was shown that a CNN model becomes more accurate as the number of different identities in the training set increases, provided that several samples per identity are available. A good example is Google's FaceNet [11], which used between 100 million and 200 million face images of about 8 million different people for training. A triplet loss function with a novel online triplet sampling strategy was used for training FaceNet, which achieved an accuracy of 99.63% on the LFW benchmark. The triplet loss has been subsequently used to fine-tune CNNs pre-trained with a classification loss with good results [5, 21]. Indeed, the triplet loss has become one of the most popular training objectives for face verification [11, 5, 21, 13, 14], and has been used in other image similarity tasks such as ranking images [22, 23, 24] and learning local image descriptors [25, 26]. Other popular tricks to improve the performance of CNN-based face recognition include Joint Bayesian [4, 9, 19, 27] and building ensemble models trained on different face

crops [9, 19, 21, 27].

Recognition of faces with occlusions has been typically handled using two different types of methods, namely, (i) methods that extract local features from the non-occluded regions or (ii) methods that attempt to reconstruct occluded regions. In the first type of methods, occluded regions are detected first and discarded from the set of local regions used to represent a face [28, 29, 30]. The second type of methods is based on dictionary or collaborative methods. These methods attempt to represent an occluded test image using images or features from the training set together with a residual term that accounts for the occluded region [31, 32, 33, 34]. However, these approaches do not generalise well, as they assume that the training set contains images of the same class as the test images. Only recently, a deep learning approach based on LSTM-Autoencoders [35] has been proposed with the purpose of reconstructing occluded faces that are not contained in the training set. We are not aware of any other studies that deal with face recognition with occlusion using neural networks.

3. Proposed Methods

We use the CNN architecture proposed in [4], which has demonstrated the ability to achieve high accuracy on the LFW benchmark while maintaining low computational complexity. This CNN architecture is similar to that used in [36] but comprises only ten convolutional layers and one fully-connected layer. The input to this CNN is a greyscale image of size 100×100 pixels aligned using a simple 2D affine transformation. More details about this CNN architecture can be found in [4].

As a first training stage, our method adopts the approach of training a classifier wherein the CNN produces a vector of scores $\boldsymbol{s}$ for each class j, which is passed to a softmax function to calculate the probability p of the correct class y:

$$p = \frac{e^{s_y}}{\sum_j e^{s_j}} \tag{1}$$

The total loss of the CNN is defined as the average cross-entropy loss for each training sample i:

$$L = -\sum_{i}^{N} \log p_i \tag{2}$$

where N is the number of samples in a batch of training samples.

In order to use the trained CNN classification model to compare face images that are not present in the training set, the classification layer (i.e. the layer producing the scores $\boldsymbol{s}$) is discarded and the features from the previous layer are used as bottleneck features. These bottleneck features can directly be used as the feature vector representing a face or can be further optimised as described in Section 3.2. We have adopted cosine similarity to compare pairs of feature vectors to get a similarity score that indicates the likelihood of two face images belonging to the same identity.

We trained such a CNN classification model using the CASIA-WebFace database [4]. This database contains 494,414 face images of 10,575 different celebrities gathered from the Internet. We randomly selected 10% of the images as validation images and used the rest as training images. We consider this CNN model as the baseline for performance comparison in our work and refer to it henceforth as model A.

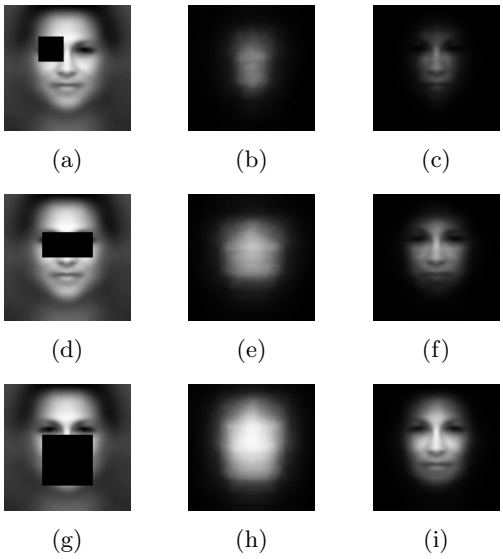
3.1. Occlusions Maps

As shown in [6], it is possible to use visualisation techniques to gain insight into the behaviour of CNN models after they have been trained. To solve the facial occlusion challenge, we are interested in identifying which face regions a CNN model relies on the most, as we want to avoid this reliance. Using a classification model, one way of visualising these regions is by observing how a correct class score fluctuates when different face regions are occluded. A similar type of occlusion sensitivity experiment has been conducted in [6] in the context of object recognition. In our case, by occluding a face image for which a CNN model predicts the correct class, we can generate a *binary occlusion map* $\boldsymbol{O_I}$ to indicate whether placing an occluder at a particular spatial location in the image $\boldsymbol{I}$ would cause the model to predict an incorrect class. More formally, a binary occlusion map $\boldsymbol{O_I}$ is defined as follows:

$$\boldsymbol{O_I}(i,j) = \begin{cases} 0, & \text{if } y'_{i,j} = y \\ 1, & \text{otherwise} \end{cases} \tag{3}$$

where $y'_{i,j}$ is the predicted class when the centre of an occluder is placed at the location (i,j) of the image $\boldsymbol{I}$ and y is the correct class for the image $\boldsymbol{I}$.

Since we are using face images that are aligned, we can construct a generic *occlusion map* $\boldsymbol{O}$ by simply averaging the binary occlusion maps of a set of face images. Each value of an occlusion map $\boldsymbol{O}(i,j)$ corresponds to the classification error incurred by a model when an occluder is placed at the location (i,j) in all the images used to generate $\boldsymbol{O}$. For convenience, we refer to face regions that present high classification error as *high effect regions* (as these are the regions in which the model relies on the most). By contrast, we refer to



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 1: (a), (d), (g) Mean image occluded at a random location with an occluder of 20×20, 20×40, and 40×40 respectively. (b), (e), (h) Occlusion maps $\boldsymbol{O}_{20\times20}$, $\boldsymbol{O}_{20\times40}$, and $\boldsymbol{O}_{40\times40}$ generated using model A and the corresponding occluders. The pixel intensity of the occlusion maps represents the classification error rate when placing the occluder at each location. (c), (f), (i) Masked mean image using the occlusion maps $\boldsymbol{O}_{20\times20}$, $\boldsymbol{O}_{20\times40}$, and $\boldsymbol{O}_{40\times40}$ respectively.

face regions that present low classification error as *low effect regions*. These high and low effect regions correspond to the bright and dark areas in the occlusion maps shown in Figure 1 respectively.

Considering the 100×100 face images used as input to our model, we experiment with occluders of three different sizes. In particular, we use (i) a square occluder of 20×20 pixels that can cover small regions such as one eye, the nose or the mouth as shown in Figure 1a; (ii) a rectangular occluder of 20×40 pixels that can cover wider regions such as both eyes simultaneously as shown in Figure 1d; and (iii) a larger square occluder of 40×40 pixels that can cover several face regions simultaneously as shown in Figure 1g. We denote the occlusion maps generated with the 20×20, 20×40 and 40×40 occluders by $\boldsymbol{O}_{20\times20}$, $\boldsymbol{O}_{20\times40}$ and $\boldsymbol{O}_{40\times40}$ respectively. Figures 1b, 1e and 1h show an example of these occlusion maps generated by model A using 1,000 images from our validation set.

According to Figures 1c, 1f and 1i, the central part of the face is one of the highest effect regions. This might be due to the presence of non-frontal face images in the training set. In these kind of face images, the outer parts of the face might not be visible, therefore, the model is not able to learn features from them that are as discriminative as the features extracted from the central part of the face, which is usually not occluded. We can reverse this behaviour by training with more face images that present occlusions located in high effect regions (central part of the face), as this will force the model to learn more discriminative features from low effect regions (outer parts of the face).

(a) (b) (c)

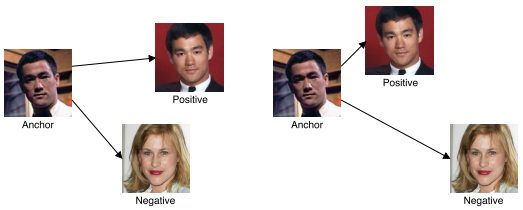
Figure 2: Example occluders used during training with different intensities, noise types and noise levels. (a) Salt-and-pepper noise. (b) Speckle noise. (c) Gaussian noise.

One way of achieving this is by augmenting the training set with face images that present occlusions located at random locations. To do this, during training we can generate occluded training images by overlaying the original training images with a randomly located occluder. However, since we want to favour occlusions in high effect regions, we propose to augment the training set with face images that present occlusions located in high effect regions more frequently than in low effect regions. For this reason, the location of the occluder is sampled from a probability distribution $\boldsymbol{P}$ generated by applying the softmax function with the temperature parameter T to an occlusion map $\boldsymbol{O}$:

$$P(i,j) = \frac{e^{\frac{O(i,j)}{T}}}{\sum_{n,m} e^{\frac{O(n,m)}{T}}} \tag{4}$$

With high temperatures, all locations have the same probability. With low temperatures, locations in high effect regions are assigned a higher probability. In our experiments we used a temperature T of 0.25 with $\boldsymbol{O}_{20\times20}$, 0.4 with $\boldsymbol{O}_{20\times40}$ and 0.6 with $\boldsymbol{O}_{40\times40}$. The size of the occluder used to generate $\boldsymbol{P}$ is the same as the size of the occluder used to generate the occluded training images.

As shown in Figure 2, we use occluders of random intensities (or random colours if we were dealing with colour images) that present different types of random noise (salt-and-pepper, speckle and Gaussian noise). This is important because if the face is always covered by the same type of occluder, the CNN would only learn features that are robust against that particular type of occlusion. For example, if a black patch is



(a) (b)

Figure 3: Example triplet from the CASIA-WebFace dataset. (a) Before triplet training. (b) After triplet training.

always used to occlude faces during training, the CNN model would perform well when the face is occluded by a black patch, but not when it is occluded by a patch of a different intensity.

This training procedure produces two desired outcomes, namely, (i) the training set is augmented with variations not present in the original data, and (ii) the occluder has a regulariser effect, helping the CNN to learn features from all face regions equally. Both of these increase the generalisation capability of the model and prevent overfitting. In Section 4 we provide experimental results, with both occluded and non-occluded face images, to validate these claims.

3.2. Batch Triplet Loss

In order to make the bottleneck features generalise better to classes not present in the training set, we fine-tune model A using a triplet loss function. This training objective is also used in other similar works [5, 21, 13, 14]. However, in this work, we fine-tune the bottleneck features directly instead of learning a linear projection from them. It could be argued that the CNN model could be trained from scratch using a triplet loss function, as proposed in [11]. But, according to our experiments, training with softmax cross-entropy loss offers faster convergence than training with a triplet loss when a reasonable amount of samples per class are available and the number of classes is not very large.

To form a triplet we need an anchor image, a positive image and a negative image. The anchor and the positive images belong to the same class and the negative image belongs to a different class. Denoting the output vector of the CNN model as $\boldsymbol{z}$ (in our setting this would be the bottleneck features), we can represent the output features for a particular triplet i as $(\boldsymbol{z}_i^a, \boldsymbol{z}_i^p, \boldsymbol{z}_i^n)$, denoting the output features for the anchor, positive and negative images respectively. The goal of a triplet loss function is to make the distance between $\boldsymbol{z}_i^a$ and $\boldsymbol{z}_i^n$ (i.e. images from different classes) larger than the distance between $\boldsymbol{z}_i^a$ and $\boldsymbol{z}_i^p$ (i.e. images from the same class) by at least a minimum margin α. Figure 3 shows a visual representation of a triplet before and after training. In this work, we consider the following as the *standard triplet loss function*:

$$L = \sum_i^N \max\left(0, \|\boldsymbol{z}_i^a - \boldsymbol{z}_i^p\|_2^2 - \|\boldsymbol{z}_i^a - \boldsymbol{z}_i^n\|_2^2 + \alpha\right) \quad (5)$$

Alternative versions of the standard triplet loss function can be defined with distance metrics other than the squared euclidean distance. For example, the dot product is used as the similarity measure in [13]. More generally, we can write:

$$L = \sum_i^N \max\left(0, d(\boldsymbol{z}_i^a, \boldsymbol{z}_i^p) - d(\boldsymbol{z}_i^a, \boldsymbol{z}_i^n) + \alpha\right) \quad (6)$$

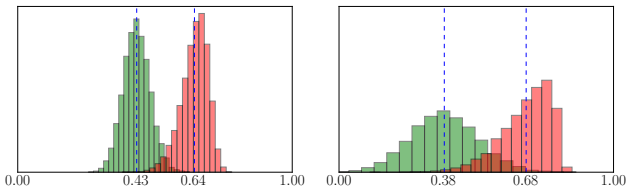
where $d(\boldsymbol{x}, \boldsymbol{y})$ is any function that gives a score indicating distance between two feature vectors. As seen in Equation 6, only triplets that violate the margin condition $d(\boldsymbol{z}_i^a, \boldsymbol{z}_i^p) + \alpha > d(\boldsymbol{z}_i^a, \boldsymbol{z}_i^n)$ produce a loss greater than zero and therefore contribute to the model's convergence. To increase the training efficiency, we adopt the online triplet sampling strategy proposed in [11] to select such triplets and only use them during training. Taking this into consideration, we can rewrite Equation 6 as:

$$L = \mu_{ap} - \mu_{an} + \alpha \quad (7)$$

where μ_{ap} and μ_{an} are the mean values of the distribution of positive and negative scores respectively.

From Equation 7 we can see that the loss becomes zero whenever μ_{an} is equal to μ_{ap} plus the margin α. In other words, the triplet loss function tries to separate the mean values of the distribution of positive scores μ_{ap} from the mean value of the distribution of negative scores μ_{an} by a minimum margin α.

A problem with the standard triplet loss function is that, in general, separating the mean values of the two score distributions does not ensure that the model performs well in a verification task. In Figure 4 we show how a CNN model that has been fine-tuned with the standard triplet loss function is able to further separate the mean values of the two score distributions but does not produce a better accuracy. This is because there might be more overlapping between the two distributions, causing more false positives and/or false negatives. A solution to this problem is to also minimise the standard deviation of each score distribution. Our loss function is inspired by the concept of *decidability*, proposed in [37] as a way of measuring the achievable accuracy of a verification system regardless of the selected threshold or operating point. A possible



(a) (b)

Figure 4: (a) Distribution of positive and negative scores after training a CNN classification model. (b) Distribution of positive and negative scores after fine-tuning the same CNN model with the standard triplet loss. Observe how even though the triplet training has been able to further separate the mean values of the two distributions, there is more overlapping between them, causing more false positives and/or false negatives

measure of decidability is defined as follows [37]:

$$d = \frac{|\mu_{ap} - \mu_{an}|}{\sqrt{\frac{1}{2}\left(\sigma_{ap}^2 + \sigma_{an}^2\right)}} \tag{8}$$

where σ_{ap}^2 and σ_{an}^2 are the variances of the distributions of positive and negative scores respectively.

Equation 8 implies that a higher decidability d is achieved by increasing the difference between the mean values of the two score distributions while decreasing both of their variances. Although it would be possible to use the inverse of Equation 8 as our training objective, in practice, using the margin parameter α leads to a better separation between the two score score distributions. For this reason, we construct our loss function by adding a new term to Equation 7 that accounts for the variance in the two score distributions:

$$L = (1 - \beta)\left(\mu_{ap} - \mu_{an} + \alpha\right) + \beta\left(\sigma_{ap}^2 + \sigma_{an}^2\right) \tag{9}$$

where β is a parameter that balances the contribution of the two terms. In particular, at $\beta = 1$, the term that accounts for the difference between the mean values of each score distribution vanishes and only the term that accounts for the variances of the score distributions has an effect. The opposite happens when $\beta = 0$.

An advantage of using Equation 9 as the training objective is that even if a triplet does not violate the margin condition, the loss will usually be greater than zero since the term that accounts for the variances of the score distributions is non-zero. Even though this means that adopting an online triplet sampling strategy is not strictly needed, in our experiments we noticed faster convergence when using it. Concurrent to our work, a similar loss function has been proposed in [26] to learn local image descriptors. However, [26] does not make use of online triplet sampling.

Note that the loss function in Equation 9 cannot be expressed as the average loss for each training image since the variances need to be computed with more than one sample. Ideally, we need to train using large enough batches of images so that the variance estimation is more accurate. For this reason, we refer to this form of triplet loss as *batch triplet loss*. In Section 4.3, we show the improved accuracy when using our loss function compared to the standard triplet loss function.

4. Experiments

In this section, we provide experimental results for our two contributions. In Sections 4.1 and 4.2 we test different CNN models trained with occluded training images as described in Section 3.1. In Section 4.1, we evaluate the performance on face images that present artificial occlusions generated from the validation set of the CASIA-WebFace database [4]. In Section 4.2 we evaluate the performance on face images that present real-life occlusions from the the AR face database [7]. In Section 4.3 we show our experimental results on the LFW [1] benchmark using the CNN models evaluated in Section 4.2 and their fine-tuned versions using the standard triplet loss function and the proposed batch triplet loss function.

4.1. Performance on Faces presenting Artificial Occlusions

In Section 3.1 we described a training procedure for increasing the CNN model classification accuracy on occluded faces by using a probability distribution $\boldsymbol{P}$ to augment the training set with occluded training images. In this section we will compare the performance of two different training schemes. The first training scheme comprises fine-tuning our baseline model A, described in Section 3, with occluded training images generated by sampling the occluder locations from a probability distribution $\boldsymbol{P}$ as described in Section 3.1. By contrast, the second training scheme comprises fine-tuning model A with occluded training images generated by sampling the occluder locations from a standard normal distribution. The goal of training with these two training schemes is to assess the benefits of training CNN models with images overlayed by strategically located occluders as opposed to randomly located occluders.

We train several CNN models in this manner, one for each of the occluder sizes shown in Figure 1. We add the size of the occluder used to generate the occluded training images to the name of each fine-tuned model. Additionally, if the model was trained following the second training scheme, an R is added to the

model name. For example, model A fine-tuned with occluded training images overlayed by an occluder of 20×20 pixels becomes $A_{20\times20}$ if the locations of the occluder are sampled from $\boldsymbol{P}$ (first training scheme) and $A_{20\times20R}$ if the locations of the occluder are sampled from a standard normal distribution (second training scheme).

To compare the accuracy of these fine-tuned CNN models we generate occlusion maps $\boldsymbol{O}$ from them (one for each of the occluder sizes). Since an occlusion map indicates the classification error incurred by a model at each spatial location, we can easily calculate the mean classification accuracy as $1-\sum_{i,j} O(i,j)$. Table 1 shows the mean classification accuracy and standard deviation for each occlusion map generated by each fine-tuned model. Observe that not only all the models fine-tuned with occluded training images achieve a higher classification accuracy than model A but their standard deviations are considerably smaller. This indicates that the performance of the fine-tuned models is much less affected by the location of the occluder, i.e. the models are able to extract discriminant features from all the face regions more equally, regardless of the location of occluder. Moreover, the results in Table 1 show the better performance of the models trained with occluded training images overlayed by strategically located occluders compared to those trained with occluded training images overlayed by randomly located occluders.

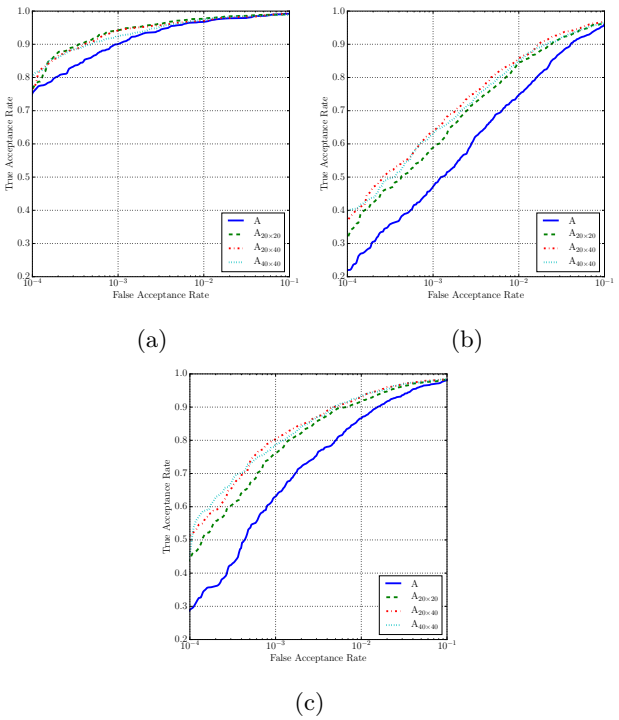
Note that all the occlusion maps referred to in Table 1 were generated using 1,000 images from the randomly selected validation set of the CASIA-WebFace database. However, to avoid any bias in the results, we selected a different set of 1,000 images to generate the probablity distribution $\boldsymbol{P}$ used when training each model and to generate the occlusion maps used to compute the results shown in Table 1. Also note that, in practice, to generate each occlusion map $\boldsymbol{O}$, we use slightly different sets of 1,000 images, as we always want to ensure that for each trained model all the evaluated images are correctly classified if none of them are occluded.

4.2. Performance on Faces presenting Real-life Occlusions

Since the goal is to improve the accuracy when dealing with real-life occlusions, we have further evaluated the performance of our CNN models on the AR face database [7]. The AR face database contains 4,000 face images of 126 different subjects with different facial expressions, illumination conditions and occlusions. Out of these, we only use faces with different illumination conditions and occlusions. The different illumination

(a) (b) (c)

Figure 5: Example images from the AR database. In each subfigure, the highlighted image on the top left is the reference image (target image) used to compare against the other three images (query images). (a) Non-occluded. (b) Wearing sunglasses. (c) Wearing scarf.



(a) (b)

(c)

Figure 6: AR database ROC curves (a) Non-occluded. (b) Wearing sunglasses. (c) Wearing scarf.

conditions correspond to face images with a light on the left side, right side or both. The occluded face images consist of people wearing either sunglasses or a scarf. We carry out three different evaluations. In each evaluation, we compare non-occluded faces against (i) other non-occluded faces, (ii) faces occluded by a pair of sunglasses, and (iii) faces occluded by a scarf. Figure 5 shows examples of each type of image used in the three evaluations.

As shown by the resulting ROC curves in Figures 6a

Model	$\boldsymbol{O}_{20\times20}$	$\boldsymbol{O}_{20\times40}$	$\boldsymbol{O}_{40\times40}$
A	$92.9\% \pm 10.99$	$86.18\% \pm 18.51$	$76.19\% \pm 27.89$
$A_{20\times20}$	$\mathbf{97.69\% \pm 2.62}$	$\mathbf{95.1\% \pm 5.64}$	$\mathbf{88.9\% \pm 14.13}$
$A_{20\times20R}$	$97.12\% \pm 3.55$	$93.98\% \pm 7.39$	$86.93\% \pm 16.98$
$A_{20\times40}$	$\mathbf{97.75\% \pm 2.42}$	$\mathbf{95.85\% \pm 4.03}$	$\mathbf{90.64\% \pm 10.88}$
$A_{20\times40R}$	$97.62\% \pm 2.9$	$95.45\% \pm 5.12$	$89.54\% \pm 13.29$
$A_{40\times40}$	$\mathbf{98.37\% \pm 1.7}$	$\mathbf{96.8\% \pm 3.16}$	$\mathbf{93.47\% \pm 6.94}$
$A_{40\times40R}$	$98.31\% \pm 2.29$	$96.52\% \pm 4.14$	$92.61\% \pm 9.13$

Table 1: Mean classification accuracy and standard deviation of each occlusion map $\boldsymbol{O}$ generated by different CNN models.

to 6c, the performance of the models trained with occluded training images consistently outperform the baseline model A, particularly at low False Acceptance Rates. Note that the performance does not seem to be greatly affected by the occluder size. The ROC curve for the evaluation of non-occluded faces (Figure 6a) shows that model $A_{40\times40}$ performs slightly worse than models $A_{20\times20}$ and $A_{20\times40}$, perhaps because the large occluder used during training makes the model rely on fewer features. As a consequence, the model performs worse when presented with non-occluded faces in which all the face regions are visible and contain useful features. In contrast, model $A_{20\times20}$ performs worse than the other two when presented with faces occluded by a pair of sunglasses (Figure 6b) or a scarf (Figure 6c). This might be because the occluder used during training is too small to simulate these types of occlusions.

Observe that, even though model $A_{40\times40}$ achieved the best classification accuracy when evaluated on face images that present artificial occlusions (Section 4.1), the results on the AR face database differ because the evaluation involves comparing pairs of occluded and non-occluded face images instead of only classifying occluded face images. For this reason, the models need to perform well not only when presented with occluded face images but also with non-occluded face images. It seems that using a medium-sized occluder like the one used to train model $A_{20\times40}$ offers the best performance when taking into account the three different evaluations, as it avoids the problems encountered with small occluders (not being able to simulate large occlusions like sunglasses and scarves) and large occluders (worse performance when presented with non-occluded faces). Note that we did not repeat these experiments with the models trained using the second training scheme described in Section 4.1, as their performance was already shown to be inferior.

4.3. Performance on the LFW benchmark

We now adopt another approach to training by fine-tuning model A using the standard triplet loss and the batch triplet loss described in Section 3.2. We do this by discarding the classification layer, normalising the features of the previous layer (bottleneck features) using the L_2-norm and training the whole CNN with one of the two loss functions. We refer to the CNN model fine-tuned with the standard triplet loss function as model B, and the CNN model fine-tuned with the batch triplet loss function as model C. The parameter α is set to 0.5 when using any of these training objectives and the parameter β is set to 0.7 when training with the batch triplet loss function.

Additionally, we also trained these CNN models with occluded training images. Similarly to the notation followed in Section 4.1, we append the size of the occluder used during training to the model name. In this case, we trained these models by fine-tuning a model that has already been trained with occluded training images instead of fine-tuning model A. For example, to train model $B_{20\times40}$ we fine-tuned model $A_{20\times40}$ (and not model A) with occluded training images overlayed by an occluder of 20×40 placed at locations sampled from a probability distribution $\boldsymbol{P}$.

Our models are evaluated on the LFW dataset following the *unrestricted, labeled outside data* protocol [38] (i.e. the protocol that allows training with data that is not part of the LFW dataset). The LFW protocol divides the test set in ten splits. The classification accuracy on each test split is calculated by counting the amount of matching and mismatching pairs given a certain threshold (in our case, pairs that give a similarity score above the threshold are counted as matching pairs and pairs that give a similarity score below the threshold are counted as mismatching pairs). For each test split, we selected the threshold that gives the highest amount of correct classifications in the other nine splits. The final reported value is the mean classification accuracy and the standard deviation calculated from the

(a) (b)

Figure 7: Example pairs from the LFW benchmark. (a) Matching pairs. (b) Mismatching pairs.

Model	Accuracy
A	97.33% ± 0.71
B	97.73% ± 0.76
C	**98.12% ± 0.65**
$A_{20\times20}$	97.4% ± 0.71
$B_{20\times20}$	97.85% ± 0.69
$C_{20\times20}$	**98.35% ± 0.73**
$A_{20\times40}$	97.68% ± 0.83
$B_{20\times40}$	97.79% ± 0.82
$C_{20\times40}$	**98.42% ± 0.68**
$A_{40\times40}$	97.18% ± 0.63
$B_{40\times40}$	97.5% ± 0.57
$C_{40\times40}$	**98.16% ± 0.64**

Table 2: Mean classification accuracy and standard deviation of different CNN models evaluated following the LFW unrestricted, labeled outside data protocol.

ten test splits. Note that most of the face images in the LFW dataset are not occluded, therefore, we do not expect to see a performance improvement as large as that seen in our experiments with occluded faces in Section 4.2. Figure 7 shows examples of matching and mismatching pairs of face images from the LFW benchmark.

As shown in Table 2, all the CNN models fine-tuned with the batch triplet loss outperform the CNN models trained with the standard triplet loss, validating the usefulness of our approach. Moreover, consistent with the results shown in Section 4.2, the CNN models trained with the 20×40 occluder are the best performers.

5. Conclusions

We have investigated which parts of the human face have the highest impact on face recognition accuracy. The proposed occlusion maps are a good way of visualising these regions and, at the same time, provide useful information about a classification model's performance on faces that present artificial occlusions. According to our experimental results, even a state-of-the-art CNN-based face recognition model fails to maintain its high performance when these face regions are occluded (e.g. by a pair of sunglasses or a scarf). We have demonstrated how these occlusion maps can be used during the training procedure to augment the training set with face images that present artificial occlusions. These artificial occlusions are strategically positioned in locations where the performance of a CNN model trained in a conventional way is most sensitive. Training with these augmented training sets, we produce CNN models that are more robust to face occlusions. As shown in our experimental results, our proposed method has shown consistent performance improvement on face images that present artificial or real-life occlusions and on face images that do not present any occlusions.

Additionally, we have revisited the problem of learning features for a verification task using distance metric objectives. We have extended the widely used triplet loss function by adding a new term that minimises the standard deviation of the distributions of positive and negative scores. In our experiments on the LFW benchmark, the proposed batch triplet loss has consistently achieved better results than the standard triplet loss. Finally, experimental results have confirmed that the best CNN models result from a combination of our two proposed approaches, regardless of whether the face images are occluded or not.

Acknowledgments

This work resulted from a collaborative research project between University of Hertfordshire and IDscan Biometrics (a GBG company) as part of a Knowledge Transfer Partnership (KTP) programme.

References

[1] 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," tech. rep., Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[2] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard, "The megaface benchmark: 1 million faces for recognition at scale," in *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4873–4882, 2016.

[3] 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*, pp. 87–102, Springer, 2016.

[4] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014.

[5] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *British Machine Vision Conference*, vol. 1, p. 6, 2015.

[6] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision-ECCV 2014*, pp. 818–833, Springer, 2014.

[7] A. M. Martinez, "The ar face database," *CVC Technical Report*, vol. 24, 1998.

[8] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 1891–1898, IEEE, 2014.

[9] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 539–546, IEEE, 2005.

[10] H. Fan, Z. Cao, Y. Jiang, Q. Yin, and C. Doudou, "Learning deep face representation," *arXiv preprint arXiv:1403.2802*, 2014.

[11] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 815–823, 2015.

[12] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in Neural Information Processing Systems*, pp. 1988–1996, 2014.

[13] S. Sankaranarayanan, A. Alavi, C. Castillo, and R. Chellappa, "Triplet probabilistic embedding for face verification and clustering," *arXiv preprint arXiv:1604.05417*, 2016.

[14] S. Sankaranarayanan, A. Alavi, and R. Chellappa, "Triplet similarity embedding for face verification," *arXiv:1602.03418 [cs]*, 2016.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[16] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *Neural Networks, IEEE Transactions on*, vol. 8, no. 1, pp. 98–113, 1997.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[18] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 1701–1708, IEEE, 2014.

[19] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, "Bayesian face revisited: A joint formulation," *Computer Vision–ECCV 2012*, pp. 566–579, 2012.

[20] E. Zhou, Z. Cao, and Q. Yin, "Naive-deep face recognition: Touching the limit of lfw benchmark or not?," *arXiv preprint arXiv:1501.04690*, 2015.

[21] J. Liu, Y. Deng, T. Bai, Z. Wei, and C. Huang, "Targeting ultimate accuracy: Face recognition via deep embedding," *arXiv:1506.07310 [cs]*, 2015.

[22] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1386–1393, 2014.

[23] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*, pp. 84–92, Springer, 2015.

[24] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," in *European Conference on Computer Vision*, pp. 241–257, Springer, 2016.

[25] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3d pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3109–3118, 2015.

[26] B. Kumar, G. Carneiro, I. Reid, *et al.*, "Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5385–5394, 2016.

[27] C. Ding and D. Tao, "Robust face recognition via multimodal deep face representation," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 2049–2058, 2015.

[28] R. Min, A. Hadid, and J. L. Dugelay, "Improving the recognition of faces occluded by facial accessories," in *2011 IEEE International Conference on Automatic Face Gesture Recognition and Workshops (FG 2011)*, pp. 442–447, 2011.

[29] M. Sharma, S. Prakash, and P. Gupta, "An efficient partial occluded face recognition system," *Neurocomputing*, vol. 116, pp. 231–241, 2013.

[30] S. Park, H. Lee, J.-H. Yoo, G. Kim, and S. Kim, "Partially occluded facial image retrieval based on a similarity measurement," *Mathematical Problems in Engineering*, vol. 2015, p. e217568, 2015.

[31] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[32] Z. Zhou, A. Wagner, H. Mobahi, J. Wright, and Y. Ma, "Face recognition with contiguous occlusion using markov random fields.," in *ICCV*, pp. 1050–1057, 2009.

[33] M. Yang, L. Zhang, S. C. K. Shiu, and D. Zhang, "Gabor feature based robust representation and classification for face recognition with gabor occlusion dictionary," *Pattern Recognition*, vol. 46, no. 7, pp. 1865–1878, 2013.

[34] R. He, W. S. Zheng, and B. G. Hu, "Maximum correntropy criterion for robust face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1561–1576, 2011.

[35] F. Zhao, J. Feng, J. Zhao, W. Yang, and S. Yan, "Robust lstm-autoencoders for face de-occlusion in the wild," *arXiv preprint arXiv:1612.08534*, 2016.

[36] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[37] J. Daugman, *Biometric decision landscapes.* No. 482, University of Cambridge, Computer Laboratory, 2000.

[38] G. B. Huang and E. Learned-Miller, "Labeled faces in the wild: Updates and new reporting procedures," *Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep*, pp. 14–003, 2014.