Fingerprint Liveness Detection Using Convolutional Neural Networks

Rodrigo Frassetto Nogueira, Roberto de Alencar Lotufo, and Rubens Campos Machado

***Abstract*— With the growing use of biometric authentication systems in the recent years, spoof fingerprint detection has become increasingly important. In this paper, we use convolutional neural networks (CNNs) for fingerprint liveness detection. Our system is evaluated on the data sets used in the liveness detection competition of the years 2009, 2011, and 2013, which comprises almost 50 000 real and fake fingerprints images. We compare four different models: two CNNs pretrained on natural images and fine-tuned with the fingerprint images, CNN with random weights, and a classical local binary pattern approach. We show that pretrained CNNs can yield the state-of-the-art results with no need for architecture or hyperparameter selection. Data set augmentation is used to increase the classifiers performance, not only for deep architectures but also for shallow ones. We also report good accuracy on very small training sets (400 samples) using these large pretrained networks. Our best model achieves an overall rate of 97.1% of correctly classified samples—a relative improvement of 16% in test error when compared with the best previously published results. This model won the first prize in the fingerprint liveness detection competition 2015 with an overall accuracy of 95.5%.**

***Index Terms*— Fingerprint recognition, machine learning, supervised learning, neural networks.**

I. Introduction

THE BASIC aim of biometrics is to automatically discriminate subjects in a reliable manner for a target application based on one or more signals derived from physical or behavioral traits, such as fingerprint, face, iris, voice, palm, or handwritten signature. Biometric technology presents several advantages over classical security methods based on either some information (PIN, Password, etc.) or physical devices (key, card, etc.) [2]. However, providing to the sensor a fake physical biometric can be an easy way to overtake the systems security. Fingerprints, in particular, can be easily spoofed from common materials, such as gelatin, silicone, and wood glue [2]. Therefore, a safe fingerprint system must correctly distinguish a spoof from an authentic finger (Figure 1). Different fingerprint liveness detection algorithms have been proposed [3]–[5], and they can be broadly divided into two approaches: hardware and software. In the hardware approach, a specific device is added to the sensor in order to detect particular properties of a living trait such as blood pressure [6],

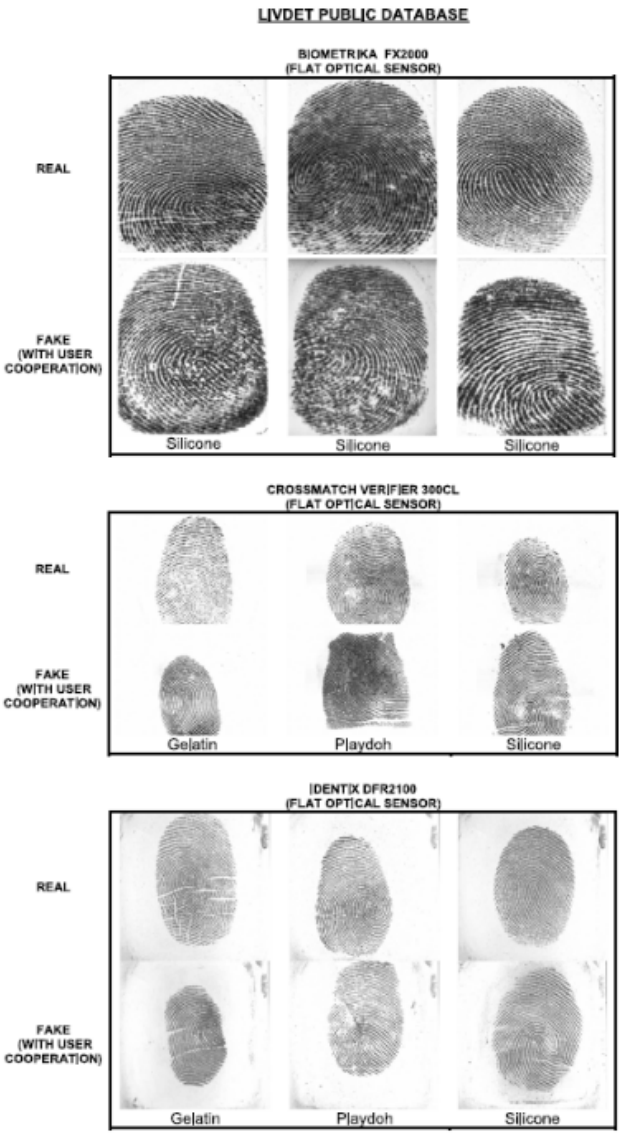

Fig. 1. Typical examples of real and fake fingerprint images that can be obtained from the LivDet2009 database used in the experiments. Figure extracted from [9].

Manuscript received July 2, 2015; revised October 20, 2015 and December 20, 2015; accepted January 4, 2016. Date of publication January 22, 2016; date of current version March 23, 2016. The work of R. de Alencar Lotufo was supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico under 311228/2014-3. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Sebastien Marcel.

R. F. Nogueira is with the Department of Computer Science, New York University, New York, NY 11209 USA (e-mail: rodrigonogueira4@gmail.com).

R. de Alencar Lotufo is with the Department of Electrical and Computer Engineering, University of Campinas, Campinas 13083-852, Brazil (e-mail: lotufo@unicamp.br).

R. Campos Machado is with the Center for Information Technology Renato Archer, Campinas 13069-901, Brazil (e-mail: rubens.campos.machado@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2016.2520880

skin distortion [7], or odor [8]. In the software approach, which is used in this study, fake traits are detected once the sample has been acquired with a standard sensor.

The features used to distinguish between real and fake fingers are extracted from the image of the fingerprint. There are techniques such as those in [2] and [9], in which the features used in the classifier are based on specific fingerprint measurements, such as ridge strength, continuity, and clarity. In contrast, some works use general feature extractors such as Weber Local Descriptor (WLD) [10], which is a texture descriptor composed of differential excitation and orientation components. A new local descriptor that uses local amplitude contrast (spatial domain) and phase (frequency domain) to form a bi-dimensional contrast-phase histogram was proposed in [11]. In [12] two general feature extractors are compared: Convolutional Neural Networks (CNN) with random (i.e., not learned) weights (also explored in [13]), and Local Binary Patterns (LBP), whose multi-scale variant reported in [14] achieves good results in fingerprint liveness detection benchmarks. In contrast to more sophisticated techniques that use texture descriptors as features vectors, such as Local Phase Quantization (LPQ) [15], LBP with wavelets [16], and BSIF [17], their LBP implementation uses the original and uniform LBP coding schemes. Moreover, a variety of optional preprocessing techniques such as contrast normalization, frequency filtering, and region of interest (ROI) extraction were attempted without success. Augmented datasets [18], [19] are successfully used to increase the classifiers robustness against small variations by creating additional samples from image translations and horizontal reflections. In this study we extend the work presented in [12] by using a similar model from the well known AlexNet [19], pre-trained on the ILSVRC-2012 dataset [20], which contains over 1.2 million images and 1000 classes, and then fine-tuned on fingerprint images. We show that although the pre-trained model was designed to detect objects in natural images, fine-tuning it to the task of fingerprint liveness detection yields better results than if trained the model using randomly initialized weights. Furthermore, we train our system using a larger pre-trained model [21], VGG, the second place in the ILSVRC-2014 [20], to increase the accuracy of the classifier by another 2% in absolute values.

Thus, the contributions of this study are three-fold:

- Deep networks designed and trained for the task of object recognition can be used to achieve state-of-the-art accuracy in fingerprint liveness detection. No specific hand-engineered technique for the task of fingerprint liveness detection was used. Thus, we provide another success case of transfer learning for deep learning techniques.
- Pre-trained Deep networks require less labeled data to achieve good accuracy in a new task.
- Dataset augmentation helps to increase accuracy not only for deep architectures but also for shallow techniques such as LBP.

II. Methodology

Transfer Learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. In this study, we showed that it is possible to achieve state-of-the-art fingerprint liveness detection by using models that were originally designed and trained to detect objects in natural images (such as animals, car, people). The same idea is explored in [22], for which the authors achieved state of the art performance in CIFAR-10, Flicker Style Wikipaintings benchmarks using a pre-trained convolutional network. One important difference from their experiments to ours is that all the datasets they used contain similar images to the ImageNET dataset (Figure 2), such as objects and scenes. In our study, fingerprint images were used, which differ significantly from those of other domains.

Fig. 2. Some images from the ImageNET dataset used to pre-train the networks. Despite their difference to the fingerprint images, pre-training with natural images do help in the task of fingerprint liveness detection.

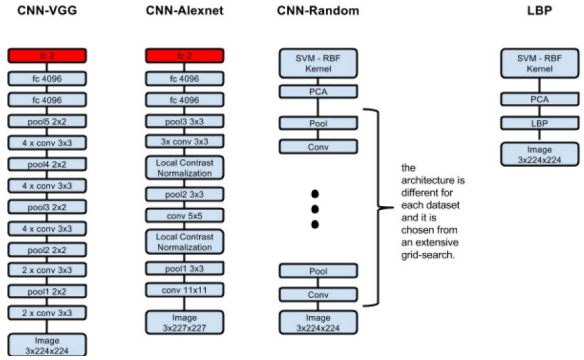

Fig. 3. Illustration of the models used in this study. The boxes in red are the only layers that are different from the original VGG-19 and Alexnet models.

A. Models

Table I describes the models in this study. All of them use dataset augmentation. Additionally, we show the architecture of the models in Figure 3. For CNN-VGG and CNN-Alexnet, the architecture is the same as described in [19] and [20], respectively, except that we replaced the last 1000-unit softmax layer by a 2-unit softmax layer (shown in red in the figure), so the network can output the 2 classes (if the image is real or fake) instead of the original 1000 classes that the networks were designed for. For the CNN-Random the architecture is different for each dataset and it was chosen via an extensive grid-search as described in [12].

B. Convolutional Networks

Convolutional Networks [23] have demonstrated state-of-the-art performance in a variety of image recognition benchmarks, such as MNIST [24], CIFAR-10 [24], CIFAR-100 [24],

TABLE I
SUMMARY OF THE MODELS USED IN THIS STUDY

Model Name	Pipeline	Description
CNN-VGG	16 Convolutional Layers + 3 Fully Connected Layers	Pre-trained model from [20] and finetuned using liveness detection datasets.
CNN-Alexnet	8 Convolutional Layers + 3 Fully Connected Layers	Pre-trained model from [18] and finetuned using liveness detection datasets.
CNN-Random	CNN-Random + PCA + SVM	Features are extracted using Convolutional Networks. The feature vector is reduced using PCA and then fed into a SVM classifier using (Gaussian) RBF kernel.
LBP	LBP + PCA + SVM	Features are extracted using LBP. The feature vector is reduced using PCA and then fed into a SVM classifier with (Gaussian) RBF kernel.

SVHN [24], and ImageNet [25]. A classical convolutional network is composed of alternating layers of convolution and local pooling (i.e., subsampling). The aim of a convolutional layer is to extract patterns found within local regions of the inputted images that are common throughout the dataset by convolving a template over the inputted image pixels and outputting this as a feature map c, for each filter in the layer. A non-linear function $f(c)$ is then applied element-wise to each feature map $c : a = f(c)$. A range of functions can be used for $f(c)$, with $max(0; c)$ a common choice. The resulting activations f(c) are then passed to the pooling layer. This aggregates the information within a set of small local regions, R, producing a pooled feature map s (normally of smaller size) as the output. Denoting the aggregation function as $pool()$, for each feature map c we have: $s_j = pool(f(c_i))\forall i \in R_j$, where R_j is the pooling region j in feature map c and i is the index of each element within it. Among the various types of pooling, max-pooling is commonly used, which selects the maximum value of the region R_j.

The motivation behind pooling is that the activations in the pooled map s are less sensitive to the precise locations of structures within the image than the original feature map c. In a multi-layer model, the convolutional layers, which take the pooled maps as input, can thus extract features that are increasingly invariant to local transformations of the input image [26], [27]. This is important for classification tasks, since these transformations obfuscate the object identity. Achieving invariance to changes in position or lighting conditions, robustness to clutter, and compactness of representation, are all common goals of pooling.

Figure 4 illustrates the feed-forward pass of a single layer convolutional network. The input sample is convoluted with three random filters of size 5×5 (enlarged to make visualization easier), generating 3 convoluted images, which are then subject to non-linear function $max(x, 0)$, followed by a max-pooling operation, and subsampled by a factor of 2.

In this study we compared three different models of convolutional networks.

The first one, CNN-Random, uses only random filter weights draw from a Gaussian distribution. Although the filter weights can be learned, filters with random weights can perform well and they have the advantage that they do not need to be learned [28]–[30]. The architecture of the model is the same as that used in [12]. It uses a convolutional network

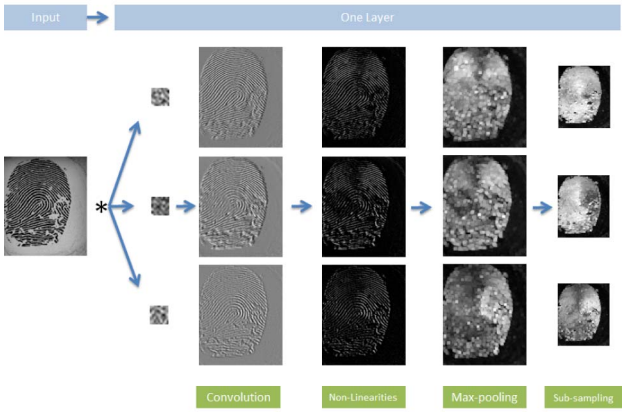

Fig. 4. Illustration of a sequence of operations performed by a single layer convolutional network in a sample image.

TABLE II
RANGE OF HYPER-PARAMETERS SEARCHED FOR THE CNN-RANDOM AND LBP PIPELINES

Pipeline Element	Hyper-parameter	Range
CNN-Random	# Layers	1, 2, 3, 4, 5
CNN-Random	# Filters (in each layer)	32, 64, ..., 2048
CNN-Random	Filter Size Convolution	5x5, 7x7, ..., 15x15
CNN-Random	Filter Size Pooling	3x3, 5x5, 7x7, 9x9
CNN-Random	Stride (reduction factor)	2, 3, ..., 7
LBP	Coding	Standard or Uniform
LBP	# Images Divisions	1x1 (no division), 3x3, 5x5, 7x7
PCA	# Components	30, 100, 300, 500, 800, 1000, 1300
SVM	Regularization Parameter C	0.1, 1, ..., 10^5
SVM	Kernel coefficient γ	10^-7, 10^-6, ..., 10^-1

with random weights as the feature extractor, the dimensions are further reduced using PCA and a SVM classifier with RBF kernels used as the classifier. An extensive search for hyper-parameter fine-tune was performed automatically on more than 2000 combinations of hyper-parameters, listed in table II. The best hyper-parameters were chosen per sensor and per dataset (ex. Biometrika 2009, Bimetrika 2011, etc) through a 5×2 cross validation method [31] which used the training dataset of each sensor in each LivDet dataset (2009, 2011, 2013).

The second model, CNN-Alexnet, is very similar to AlexNet [19], pre-trained on the ILSVRC-2012 dataset.

This model won both classification and localization tasks in the ILSVRC-2012 competition. Their trained model has been used to improve accuracy in a variety of other benchmarks such as CIFAR-10, CIFAR-100. The pre-trained network provides a good starting point for learning the network weights for other tasks, such as fingerprint liveness detection.

The third model, CNN-VGG, is very similar to the one used in [21], a 19 layer CNN which achieved the second place in the detection task of the ImageNet 2014 challenge.

For CNN-ALEXNET and CNN-VGG models, the last 1000-unit soft-max layer (originally designed to predict 1000 classes) was replaced by a 2-unit softmax layer, which assigns a score for true and fake classes. The pre-trained model was further trained with the fingerprint datasets.

The algorithm used to train CNN-Alexnet and CNN-VGG is the Stochastic Gradient Descent (SGD) with a minibatch of size 5, using momentum [32], [33] 0.9 and a fixed learning rate of 10^{-6}.

C. Local Binary Patterns

Local Binary Patterns (LBP) are a local texture descriptor that have performed well in various computer vision applications, including texture classification and segmentation, image retrieval, surface inspection, and face detection [34]. It is a widely used method for fingerprint liveness detection [14] and it is used in this work as a baseline method.

In its original version, the LBP operator assigns a label to every pixel of an image by thresholding each of the 8 neighbors of the 3×3-neighborhood with the center pixel value and considering the result as a unique 8-bit code representing the 256 possible neighborhood combinations. As the comparison with the neighborhood is performed with the central pixel, the LBP is an illumination invariant descriptor. The operator can be extended to use neighborhoods of different sizes [35].

Another extension to the original operator is the definition of so-called uniform patterns, which can be used to reduce the length of the feature vector and implement a simple rotation-invariant descriptor [35]. An LBP is called uniform if the binary pattern contains at most two bitwise transitions from 0 to 1 or vice versa when the bit pattern is considered circular. The number of different labels of LBP reduces from 256 to just 10 in the uniform pattern.

The normalized histogram of the LBPs (with 256 and 10 bins for non-uniform and uniform operators, respectively) is used as a feature vector. The assumption underlying the computation of a histogram is that the distribution of patterns matters, but the exact spatial location does not. Thus, the advantage of extracting the histogram is the spatial invariance property. To investigate if location matters to our problem, we also implemented the method presented in [36], for face recognition, where the LBP filtered images are equally divided in rectangles and their histograms are concatenated to form a final feature vector.

In this study, the histogram of the LBP image was further reduced using PCA, and a SVM with RBF kernel is used as the classifier. Similarly to the CNN-Random models, the hyper-parameters, such as the number of PCA components and SVM regularization parameter, where found using an extensive brute force search on more than 2000 combinations, listed in table II.

D. Increasing the Classifiers Generalization Through Dataset Augmentation

Dataset Augmentation is a technique that involves artificially creating slightly modified samples from the original ones. By using them during training, it is expected that the classifier will become more robust against small variations that may be present in the data, forcing it to learn larger (and possibly more important) structures. It has been successfully used in computer vision benchmarks such as in [19], [37], and [38]. It is particularly suitable to out-of-core algorithms (algorithms that do not need all the data to be loaded in memory during training) such as CNNs trained with Stochastic Gradient Descent. Our dataset augmentation implementation is similar to the one presented in [19]: from each image of the dataset five smaller images with 80% of each dimension of the original images are extracted: four patches from each corner and one at the center. For each patch, horizontal reflections are created. As a result, we obtain a dataset that is 10 times larger than the original one: 5 times are due to translations and 2 times are due to reflections. At test time, the classifier makes a prediction by averaging the individual predictions on the ten patches.

III. Experiments

A. Datasets

The datasets provided by the Liveness Detection Competition (LivDet) in the years of 2009 [39], 2011 [40], and 2013 [41] are used in this study.

LivDet 2009 comprises almost 18,000 images of real and fake fingerprints acquired from three different sensors (Biometrika FX2000, Crossmatch Verifier 300 LC, and Identix DFR 2100). Fake fingerprints were obtained from three different materials: Gelatin, Play Doh, and Silicone. Approximately one third of the images of the dataset are used for training and the remaining for testing.

LivDet 2011 comprises 16,000 images acquired from four different sensors (Biometrika FX2000, Digital 4000B, Italdata ET10, and Sagem MSO300), each having 2000 images of fake and real fingerprints. Half of the dataset is used for training and the other half for testing. Fake fingerprints were obtained from four different materials: Gelatin, Wood Glue, Eco Flex, and Silgum.

LivDet 2013 comprises 16,000 images acquired from four different sensors (Biometrika FX2000, Crossmatch L SCAN GUARDIAN, Italdata ET10, and Swipe), each having approximately 2,000 images of fake and real fingerprints. Half of the dataset is used for training and the other half for testing. Fake fingerprints were obtained from five different materials: Gelatin, Latex, Eco Flex, Wood Glue, and Modasil.

In all datasets, the real/fake fingerprint ratio is 1/1 and they are equally distributed between training and testing sets. The sizes of the images vary from sensor to sensor, ranging from 240×320 to 700×800 pixels, but they were all resized according to the input size of the pre-trained models, which

TABLE III

AVERAGE CLASSIFICATION ERROR ON TESTING DATASETS

Dataset	State-of-the-Art	CNN-VGG	CNN-Alexnet	CNN-Random	LBP
Crossmatch 2013	7.9 [13]	3.4	4.7	3.2	49.4
Swipe 2013	2.8 [43]	3.7	4.3	7.6	3.3
Italdata 2013	0.8 [41]	0.4	0.5	2.4	2.3
Biometrika 2013	1.1 [17]	1.8	1.9	0.8	1.7
Italdata 2011	11.2 [43]	8.0	9.1	9.2	12.3
Biometrika 2011	4.9 [11]	5.2	5.6	8.2	8.8
Digital 2011	2.0 [43]	3.2	4.6	3.6	4.1
Sagem 2011	3.2 [11]	1.7	3.1	4.6	7.5
Biometrika 2009	1.0 [11]	4.1	5.6	9.2	10.4
Crossmatch 2009	3.3 [43]	0.6	1.1	1.7	3.6
Identix 2009	0.5 [43]	0.2	0.4	0.8	2.6
Average	3.5	**2.9**	3.7	4.7	9.6

is 224×224 for the CNN-Alexnet model and 227×227 pixels for the CNN-VGG model.

B. Performance Metrics

The classification results were evaluated by the Average Classification Error (ACE), which is the standard metric for evaluation in LivDet competitions. It is defined as

$$ACE = \frac{SFPR + SFNR}{2} \quad (1)$$

where SFPR (Spoof False Positive Rate) is the percentage of misclassified live fingerprints and SFNR (Spoof False Negative Rate) is the percentage of misclassified fake fingerprints.

C. Implementation Details

CNN-VGG and CNN-Random were trained using the Caffe package [42], which provides very fast CPU and GPU implementations and a user-friendly interface in Python. For the CNN-Random and LBP models, we wrote an improved cross-validation/grid-search algorithm for choosing the best combination of hyper-parameters, in which each element of the pipeline is computed only when its training data is changed (the term element refers to operations such as preprocessing, feature extraction, dimensionality reduction or classification). This modification speeded-up the validation phase by approximately 10 times, although the gain can greatly vary as it depends on the element types and number of hyper-parameters chosen. An important aspect of this work is that the algorithms were run on cloud service computers, where the user can rent virtual computers and pay only for the hours that the machines are running. To train the algorithms, we used the GPU instances that allowed us to run dataset augmented experiments in a few hours; using traditional CPUs the training would take weeks.

IV. RESULTS

The average error for each testing dataset is shown on Table III. Along with the models used in this study, we also show the error rate of the state-of-the-art method for each dataset, of which most of them were found in the compilation made by [43].

Particularly interesting results are for the Crossmatch 2013 dataset. As commented by [43], most techniques have problems in this dataset. For example, the LBP presents error rates close to zero at validation time and around 50% at test time. It can be noticed from LivDet 2013 competition results that this dataset is particularly difficult to generalize, since nine of the eleven participants presented error rates greater than 45%. Contrary to these results, CNN models perform very well in this dataset, with error rates between 3.2%-4.7%.

TABLE IV

AUGMENTATION VS NO AUGMENTATION: AVERAGE ERROR ON ALL DATASETS

Model	No Augmentation	With Augmentation
CNN-VGG	4.2	2.9
CNN-Alexnet	5.0	3.7
CNN-Random	9.4	4.7
LBP	21.2	9.6

It is important to highlight that CNN-Random did require an exhaustive hyper-parameter finetune (number of layers, filter size, number of filters, etc.) in order to get a model with good accuracy. On the other hand, the architectures of CNN-Alexnet and CNN-VGG, which were already carefully selected for the ImageNet object detection task, are general enough to be reused for the fingerprint liveness detection task and yield excellent accuracy. Another interesting aspect is that the CNN-VGG performed better than the CNN-Alexnet in both object detection from ILSVRC-2012 and fingerprint liveness detection tasks. This suggests that further improvements in models for object recognition can be applied to increase accuracy in fingerprint liveness detection.

The higher performance of our CNN-VGG solution was confirmed as this model won the first place in the Fingerprint Liveness Detection 2015 Competition (LivDet) 2015 [1], with an overall accuracy of 95.51%, while the second place achieved an overall accuracy of 93.23%.

A. Effect of Dataset Augmentation

Table IV compares the effect of dataset augmentation in our proposed models. Despite its longer training and running times, the technique helps to improve accuracy: the error was reduced by a factor of 2 in some cases. More importantly, the technique is not only effective on deep architectures, as commonly known, but also in shallow architectures, such as LBP.

B. Cross-Dataset Evaluation

We would like to verify how a classifier would perform when unseen samples acquired from spoofy materials and individuals during training are presented at test time. Additionally, we want to test the hypothesis that the images share common characteristics for distinguishing fake fingerprints from real ones, that is, the important features for classification are independent from the acquisition device. For that, Cross-dataset experiments were performed, which involve training a classifier using one dataset and testing on another. For instance, a cross-dataset experiment would involve training a classifier using Biometrika-2011 dataset and testing it using Italdata-2013. In summary, these experiments should reflect how well the classifier is able to learn relevant characteristics

TABLE V
AVERAGE CLASSIFICATION ERROR ON CROSS-DATASET EXPERIMENTS

Train Dataset	Test Dataset	CNN-VGG	CNN-Alexnet	CNN-Random	LBP
Biometrika 2011	Biometrika 2013	**15.5**	15.9	20.4	16.5
Biometrika 2013	Biometrika 2011	**46.8**	47.0	48.0	47.9
Italdata 2011	Italdata 2013	14.6	15.8	21.0	**10.6**
Italdata 2013	Italdata 2011	**46.0**	49.1	46.8	50.0
Biometrika 2011	Italdata 2011	**37.2**	39.8	49.2	47.1
Italdata 2011	Biometrika 2011	**31.0**	33.9	46.5	49.4
Biometrika 2013	Italdata 2013	**8.8**	9.5	47.9	43.7
Italdata 2013	Biometrika 2013	**2.3**	3.9	48.9	48.4

TABLE VI
AVERAGE CLASSIFICATION ERROR ON CROSS-MATERIAL EXPERIMENTS

Dataset	Materials - Train	Materials - Test	CNN-VGG	CNN-Alexnet	CNN-Random	LBP
Biometrika 2011	EcoFlex, Gelatine, Latex	Silgum, Wood Glue	**10.1**	12.2	13.5	17.7
Biometrika 2013	Modalsil, Wood Glue	EcoFlex, Gelatine, Latex	**4.9**	5.8	10.0	8.5
Italdata 2011	EcoFlex, Gelatine, Latex	Silgum, Wood Glue, Other	22.1	25.8	26.0	**30.9**
Italdata 2013	Modalsil, Wood Glue	EcoFlex, Gelatine, Latex	**6.3**	8.0	10.8	10.7

that distinguish real from fake fingerprints when samples acquired from different environments and sensors are presented.

We chose to use only Biometrika and Italdata sensors from datasets of years of 2011 and 2013 of the LivDet competition, since executing all possible dataset combinations would be almost impractical to run under the current computer architecture. All the models evaluated use dataset augmentation.

Table V shows the testing error. CNN-Alexnet and CNN-VGG clearly outperform CNN-Random and LBP in most cases. However, the testing error is still high ($>20\%$) in 4 out of 8 of the experiments, indicating that the models fail to generalize when the type of sensor used for testing is different from the one used in training. Similarly, Jia *et al.* [14] reported that their multi-resolution LBP technique had poor results in cross-device experiments, with errors of around 40-50%.

C. Cross-Material Evaluation

Additionally to the influence of training and testing with different sensors (section IV-B), we investigated the performance of the classifiers when they are tested with spoofing materials never seen during training. The results are shown in Table VI. The error rates are lower than Cross-dataset experiments, which suggests that most of the generalization error can be attributed to different sensors and not to different materials.

D. Training All Datasets at Once

In this experiment we report the error rates when training and testing a single classifier using all datasets (2009, 2011, 2013), except for Swipe-2013 whose images are very different from the rest. The testing error rates, shown in Table VII, are compared with the results obtained when individual classifiers are trained per dataset, which are reported in Table III. The results show that training a single classifier with all datasets yields comparable error rates when individual classifiers are trained per dataset, which suggests that the effort to design and deploy a liveness detection system can be considerably reduced if all datasets are trained together, as the hyper-parameter fine tuning needs to be performed for only one model.

TABLE VII
AVERAGE CLASSIFICATION ERROR WHEN A SINGLE CLASSIFIER IS TRAINED USING ALL DATASETS VS ONE CLASSIFIER PER DATASET

Model	One Classifier trained with All Training Datasets	One Classifier per Dataset
CNN-VGG	3.4	2.9
CNN-Alexnet	4.1	3.7
CNN-Random	6.0	4.7
LBP	10.0	9.6

TABLE VIII
AVERAGE CLASSIFICATION ERROR FOR TESTING DATASET COMPARING THE EFFICACY OF PRE-TRAINED MODELS WITH THE ONES SOLELY TRAINED ON THE LIVDET DATASETS. THE TRAINING ERROR IS SHOWED IN PARENTHESIS

Model	Training on LivDet datasets Only	Training on ImageNet then LivDet datasets
CNN-VGG	49.4 (0.0)	2.9 (1.5)
CNN-Alexnet	48.1 (0.0)	3.7 (1.2)

E. Pre-Training Effect

In this experiment the effect of using pre-trained networks is investigated. Table VIII compares the accuracy for the CNN-VGG and CNN-Alexnet models trained using only fingerprint images and when they are first pre-trained with the ImageNet dataset and then finetuned with fingerprint images. It can be seem that pre-training is necessary for those large networks as training them using only the fingerprint images results in overfitting.

F. Number of Training Samples vs Error

Deep learning techniques require large number of labeled training data in order to achieve a good performance when the

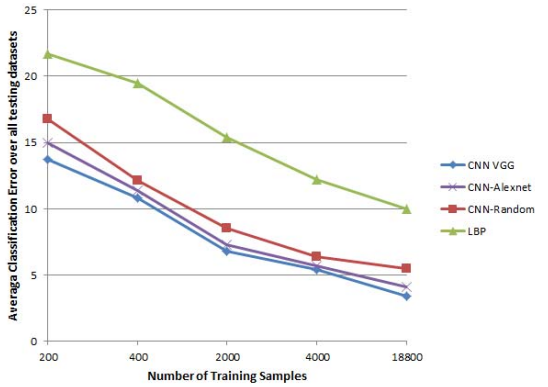

Fig. 5. Number of training samples vs Avg. Classification Error on all testing datasets.

TABLE IX
AVERAGE TRAINING AND TESTING TIMES

Technique	Training all Datasets	Testing per Image (1-core CPU)
CNN-VGG	20-40 hours (GPU)	650ms
CNN-Alexnet	5-10 hours (GPU)	230ms
CNN-Random	5-10 hours (32-core CPU)	110ms
LBP	5-10 hours (32-core CPU)	50ms

models are initialized with random weights, since there are a lot of parameters that must be learned, thus requiring many samples. However, when the weights were already learned from another task, the number of required samples can be surprisingly low in order to achieve good accuracy.

Figure 5 shows the number of training samples versus the average classification error in the test set for all datasets. Using only 400 training samples, CNN-VGG has almost the same performance as LBP using all the 18,800 training images. This suggests that less samples are needed when pre-trained models are used.

G. Processing Times

In real applications, a good fingerprint liveness detection system must be able to classify images in a short amount of time. Table IX shows the average testing/classification times for a single image (no augmentation) on a single core machine (1.8 GHz, 64-bit, with 4GB memory).

We also show the times for training all datasets together. The pre-trained CNN models (CNN-Alexnet and CNN-VGG) take around 5-40 hours to converge using a Nvidia GTX Titan GPU. The CNN-Random and LBP models take around 5-10 hours to converge on a 32-Cores machine (the larger portion of these times are required for dimensionality reduction using PCA).

V. CONCLUSIONS

Convolutional Neural Networks were used to detect false vs real fingerprints. Pre-trained CNNs can yield state-of-the-art results on benchmark datasets without requiring architecture or hyperparameter selection. We also showed that these models have good accuracy on very small training sets (∼400 samples). Additionally, no task-specific hand-engineered technique was used as in classical computer vision approaches.

Despite the differences between images acquired from different sensors, we show that training a single classifier using all datasets helps to improve accuracy and robustness. This suggests that the effort required to design a liveness detection system (such as hyper-parameters fine tuning) can be significantly reduced if different datasets (and acquiring devices) are combined during the training of a single classifier. Additionally, the pre-trained networks showed stronger generalization capabilities in cross-dataset experiments than CNN with random weights and the classic LBP pipeline.

Dataset augmentation plays an important role in increasing accuracy and it is also simple to implement. We suggest that the method should always be considered for the training and prediction phases if time is not a major concern. Given the promising results provided by the technique, more types of image transformations should be included, such as color manipulation and multiple scales described in [44] and [45].

ACKNOWLEDGMENT

The authors would like to thank Nvidia for the donation of the GPUs used in this study.

REFERENCES

[1] V. Mura, L. Ghiani, G. L. Marcialis, F. Roli, D. A. Yambay, and S. A. Schuckers, "Livdet 2015 fingerprint liveness detection competition 2015," in *Proc. IEEE 7th Int. Conf. Biometrics Theory, Appl. Syst.*, Sep. 2015, pp. 1–6.

[2] J. Galbally, F. Alonso-Fernandez, J. Fierrez, and J. Ortega-Garcia, "A high performance fingerprint liveness detection method based on quality related features," *Future Generat. Comput. Syst.*, vol. 28, no. 1, pp. 311–321, 2012.

[3] Y. Chen, A. Jain, and S. Dass, "Fingerprint deformation for spoof detection," in *Proc. Biometric Symp.*, 2005, p. 21.

[4] B. Tan and S. Schuckers, "Comparison of ridge- and intensity-based perspiration liveness detection methods in fingerprint scanners," *Proc. SPIE*, vol. 6202, p. 62020A, Apr. 2006.

[5] P. Coli, G. L. Marcialis, and F. Roli, "Fingerprint silicon replicas: Static and dynamic features for vitality detection using an optical capture device," *Int. J. Image Graph.*, vol. 8, no. 4, pp. 495–512, 2008.

[6] P. D. Lapsley, J. A. Lee, D. F. Pare, Jr., and N. Hoffman, "Anti-fraud biometric scanner that accurately detects blood flow," U.S. Patent 5 737 439, Apr. 7, 1998.

[7] A. Antonelli, R. Cappelli, D. Maio, and D. Maltoni, "Fake finger detection by skin distortion analysis," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 3, pp. 360–373, Sep. 2006.

[8] D. Baldisserra, A. Franco, D. Maio, and D. Maltoni, "Fake fingerprint detection by odor analysis," in *Advances in Biometrics*. Heidelberg, Germany: Springer, 2005, pp. 265–272.

[9] A. K. Jain, Y. Chen, and M. Demirkus, "Pores and ridges: High-resolution fingerprint matching using level 3 features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 15–27, Jan. 2007.

[10] D. Gragnaniello, G. Poggi, C. Sansone, and L. Verdoliva, "Fingerprint liveness detection based on weber local image descriptor," in *Proc. IEEE Workshop Biometric Meas. Syst. Secur. Med. Appl. (BIOMS)*, Sep. 2013, pp. 46–50.

[11] D. Gragnaniello, G. Poggi, C. Sansone, and L. Verdoliva, "Local contrast phase descriptor for fingerprint liveness detection," *Pattern Recognit.*, vol. 48, no. 4, pp. 1050–1058, 2015.

[12] R. F. Nogueira, R. de Alencar Lotufo, and R. C. Machado, "Evaluating software-based fingerprint liveness detection using convolutional networks and local binary patterns," in *Proc. IEEE Workshop Biometric Meas. Syst. Secur. Med. Appl. (BIOMS)*, Oct. 2014, pp. 22–29.

[13] D. Menotti *et al.*, "Deep representations for iris, face, and fingerprint spoofing detection," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 4, pp. 864–879, Apr. 2015.

[14] X. Jia *et al.*, "Multi-scale local binary pattern with filters for spoof fingerprint detection," *Inf. Sci.*, vol. 268, pp. 91–102, Jun. 2014.
[15] L. Ghiani, G. L. Marcialis, and F. Roli, "Fingerprint liveness detection by local phase quantization," in *Proc. 21st Int. Conf. Pattern Recognit. (ICPR)*, 2012, pp. 537–540.
[16] S. B. Nikam and S. Agarwal, "Local binary pattern and wavelet-based spoof fingerprint detection," *Int. J. Biometrics*, vol. 1, no. 2, pp. 141–159, 2008.
[17] L. Ghiani, A. Hadid, G. L. Marcialis, and F. Roli, "Fingerprint liveness detection using binarized statistical image features," in *Proc. IEEE 6th Int. Conf. Biometrics, Theory, Appl. Syst. (BTAS)*, Sep./Oct. 2013, pp. 1–6.
[18] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Proc. 7th Int. Conf. Document Anal. Recognit.*, 2003, pp. 958–963.
[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[20] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
[21] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: http://arxiv.org/abs/1409.1556
[22] S. Karayev *et al.* (2013). "Recognizing image style." [Online]. Available: http://arxiv.org/abs/1311.3715
[23] Y. LeCun, "Generalization and network design strategies," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, Tech. Rep. CRG-TR-89-4, 1989, pp. 55–143.
[24] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus, "Regularization of neural networks using DropConnect," in *Proc. 30th Int. Conf. Mach. Learn. (ICML)*, 2013, pp. 1058–1066.
[25] S. Ioffe and C. Szegedy. (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift." [Online]. Available: http://arxiv.org/abs/1502.03167
[26] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 111–118.
[27] M. D. Zeiler and R. Fergus. (2013). "Stochastic pooling for regularization of deep convolutional neural networks." [Online]. Available: http://arxiv.org/abs/1301.3557
[28] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May/Jun. 2010, pp. 253–256.
[29] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 2146–2153.
[30] A. Saxe, P. W. Koh, Z. Chen, M. Bhand, B. Suresh, and A. Y. Ng, "On random weights and unsupervised feature learning," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 1089–1096.
[31] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, 1998.
[32] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Comput. Math. Math. Phys.*, vol. 4, no. 5, pp. 1–17, 1964.
[33] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. 30th Int. Conf. Mach. Learn. (ICML)*, 2013, pp. 1139–1147.
[34] A. Hadid, M. Pietikäinen, and T. Ahonen, "A discriminative feature space for detecting and recognizing faces," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2. Jun./Jul. 2004, pp. II-797–II-804.
[35] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.
[36] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *Computer Vision*. Heidelberg, Germany: Springer, 2004, pp. 469–481.
[37] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber. (2011). "High-performance neural networks for visual object classification." [Online]. Available: http://arxiv.org/abs/1102.0183
[38] D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 3642–3649.
[39] G. L. Marcialis *et al.*, "First international fingerprint liveness detection competition—LivDet 2009," in *Image Analysis and Processing*. Heidelberg, Germany: Springer, 2009, pp. 12–23.
[40] D. Yambay, L. Ghiani, P. Denti, G. L. Marcialis, F. Roli, and S. Schuckers, "LivDet 2011—Fingerprint liveness detection competition 2011," in *Proc. 5th IAPR Int. Conf. Biometrics (ICB)*, 2012, pp. 208–215.
[41] L. Ghiani *et al.*, "LivDet 2013 fingerprint liveness detection competition 2013," in *Proc. Int. Conf. Biometrics (ICB)*, 2013, pp. 1–6.
[42] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM Int. Conf. Multimedia*, 2014, pp. 675–678.
[43] D. Gragnaniello, G. Poggi, C. Sansone, and L. Verdoliva, "An investigation of local descriptors for biometric spoofing detection," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 4, pp. 849–863, Apr. 2015.
[44] C. Szegedy *et al.* (2014). "Going deeper with convolutions." [Online]. Available: http://arxiv.org/abs/1409.4842
[45] A. G. Howard. (2013). "Some improvements on deep convolutional neural network based image classification." [Online]. Available: http://arxiv.org/abs/1312.5402

Rodrigo Frassetto Nogueira received the B.S. degree in electrical engineering and the master's degree in computer engineering from the University of Campinas, Brazil, in 2009 and 2014, respectively. He is currently pursuing the Ph.D. degree with the School of Engineering, New York University, USA. His principal research interests are in the areas of machine learning, computer vision, and natural language processing.

Roberto de Alencar Lotufo received the B.S. degree in electrical engineering from the Instituto Tecnologico de Aeronautica, Brazil, in 1978, and the Ph.D. degree in electrical engineering from the University of Bristol, U.K., in 1990. He has been with the School of Electrical and Computer Engineering, University of Campinas (Unicamp), Brazil, since 1981, where he is currently a Full Professor. His principal research interests are in the areas of image processing and analysis, pattern recognition, and machine learning. He has authored over 150 refereed international journal and full conference papers. He was awarded the Innovation Personality in 2008 and the Zeferino Vaz Academic Recognition in 2011 from Unicamp.

Rubens Campos Machado received the B.Sc. degree in electonic and telecommunications engineering from the Catholical University of Minas Gerais, Brazil, in 1978, and the M.Sc. degree in electrical engineering from the University of Campinas, Brazil, in 2002.

He was the Head of the Realtime Processing Division with the Center of Information Technology (CTI) Renato Archer, Campinas, Brazil, from 1984 to 1987, the Process Control Department from 1987 to 1996, and the Applied Control Methodologies Division from 1996 to 2000. He has been with CTI since 1983. He develops advanced automation projects for several Brazilian industries. He is a Senior Researcher with the Robotics and Computer Vision Division, CTI. His main interest areas are computer vision and machine learning/deep learning.