LIPSNN: A Light Intrusion-Proving Siamese Neural Network Model for Facial Verification

Asier Alcaide[1], Miguel A. Patricio[2], Antonio Berlanga[2], Angel Arroyo[3], Juan J. Cuadrado-Gallego[4,5] *

[1] Ultra Tendency International GmbH, 39326 Colbitz (Germany)
[2] Applied Artificial Intelligence Group, University Carlos III de Madrid, Colmenarejo, Madrid (Spain)
[3] Department of Information Systems. Technical University of Madrid, Madrid (Spain)
[4] Department of Computer Science, University of Alcalá, Madrid (Spain)
[5] Department of Computer Science and Software Engineering, Concordia University, Montreal (Canada)

Received 25 March 2021 | Accepted 20 September 2021 | Published 9 November 2021

unir
LA UNIVERSIDAD
EN INTERNET

Abstract

Facial verification has experienced a breakthrough in recent years, not only due to the improvement in accuracy of the verification systems but also because of their increased use. One of the main reasons for this has been the appearance and use of new models of Deep Learning to address this problem. This extension in the use of facial verification has had a high impact due to the importance of its applications, especially on security, but the extension of its use could be significantly higher if the problem of the required complex calculations needed by the Deep Learning models, that usually need to be executed on machines with specialised hardware, were solved. That would allow the use of facial verification to be extended, making it possible to run this software on computers with low computing resources, such as Smartphones or tablets. To solve this problem, this paper presents the proposal of a new neural model, called *Light Intrusion-Proving Siamese Neural Network*, LIPSNN. This new light model, which is based on Siamese Neural Networks, is fully presented from the description of its two block architecture, going through its development, including its training with the well- known dataset Labeled Faces in the Wild, LFW; to its benchmarking with other traditional and deep learning models for facial verification in order to compare its performance for its use in low computing resources systems for facial recognition. For this comparison the attribute parameters, storage, accuracy and precision have been used, and from the results obtained it can be concluded that the LIPSNN can be an alternative to the existing models to solve the facet problem of running facial verification in low computing resource devices.

Keywords

Deep Learning, Facial Verification, Neural Networks, Siamese Neural Networks.

DOI: 10.9781/ijimai.2021.11.003

I. Introduction

Facial biometrics is a specific biometric mechanism that can allow a person's identity to be determined by analyzing his or her face, which is possible due to the fact that the face is a physical complex characteristic that makes it possible to distinguish and identify people with great accuracy.

Other biometric systems are capable of doing the same and are widely used is fingerprint biometrics. However, although apparently they are similar, they belong to different kinds of biometric systems because of different reasons: from an interaction perspective there are those which need physical contact or collaboration from the user, as in fingerprint biometrics, and those which don't need this, as in facial biometrics. From a security perspective, a fingerprint can be stolen when the person concerned is asleep or unconscios, while facial recognition often requires the eyes to be open and a natural facial expression to be maintained, which results in facial biometrics being considered a robust and accurate bio metric mechanism.

At this point a distinction must be made between the two main facial biometrics uses: *facial verification*, which is the domain of application of the model developed in this research; and facial recognition, as these do not have the same meaning. Facial recognition is based on the comparison of an image of a person against a known database such as, for example, a database of criminals held by the police, and there is not, generally, any output from the facial recognition or direct personal benefit to the individual (1-N). SMNLR model used a multi-class Support Vector Machine classifier, obtaining different results that are used to predict the accurate label from noisy labelled facial images [1]. On the other hand, facial verification is based on the comparison of two face images, and the output of the inference between them allows the result of the comparison to be determined, such as, for example, giving access to a service or space (1-1), and is an action that one is aware that one is doing and in fact is usually an action one chooses to take in order to gain access to some personal benefit. For this reason, accuracy and speed are the key attributes for the facial verification models.

* Corresponding author.

E-mail addresses: asier.martinez@ultratendency.com (A. Alcaide), mpatrici@inf.uc3m.es (M. A. Patricio), aberlan@ia.uc3m.es (A. Berlanga), aarroyo@etsisi.upm.es (A. Arroyo), jjcg@uah.es (J. J. Cuadrado-Gallego).

The performance of these verification models is usually evaluated by applying a confusion matrix. A confusion matrix is a two-dimensional matrix that represents all the evaluation results of a classifier with respect to some test data. The first dimension of the table represents the true class of an input, and the other dimension represents the value assigned by the classifier. In facial verification approaches, the confusion matrix forms 4 elements:

- True Positive, TP. Portion of results that the classifier predicted positive when the truth is indeed positive. Images that are the same person and indeed the model classifies them as two images of the same identity.
- True Negative, TN. Portion of results with a negative detection given that the actual instance is also negative. Images that are not the same person and indeed the model classifies them as different people.
- False Positive, FP. Portion of results with a positive detection given that the actual instance is negative. Images that are not the same person but the model has detected them as the same person.
- False Negative, FN. Portion of results with a negative detection when the actual instance is positive. Images that are the same person, but the model has detected them as different ones.

The wrong predictions in a face verification system are then the FP and FN. FN are important to consider, as facial verification models improve their usability when the user is not constantly and repeatedly trying to access any privilege depending on the specified application. However, FP and FN should not be considered with same importance. A high FP rate would affect the system in a very negative way, as the model will allow access or privileges to people that should not be granted to them.

To solve the problem of facial verification using neural networks, different methods have been published in the literature. Nowadays those methods can be distinguished between deep learning methods, the newest ones, and traditional methods, and the determining feature for this distinction lies in the recognition process followed by the model:

- Traditional methods are carried out in several phases: First a pre-processing phase is needed, followed by a phase of local feature extraction and feature transformation. It is possible that some of these steps can be improved separately, however none of these improvements have resulted in significant growth in accuracy. Furthermore, most of these methods are not capable of extracting stable characteristics that are invariant to real situations [2].
- Deep learning methods use a set of layers that learn different representations at multiple levels. The features obtained from these models are robust to variations in lighting, pose and expression.

The model presented in this paper can be classified as a Deep Learning Method, and for this reason, these will be dealt with in the introduction. One of the first deep learning architectures is the work with the DeepFace network. This architecture is composed of new layers of convolutional neural networks. In recent years, facial verification models have appeared that are built on deep convolutional neural networks (CNN): Facenet [3] maps images of faces to a compact Euclidean space using CNNs and then analyses similarities between faces; in [4], authors introduce a new loss function in the learning process of CNNs, that they call centre loss, which combined with the softmax function allows for greater discriminating power in face recognition systems; in [5], the authors propose normalisation operations on the layers of a CNN, as well as the loss functions necessary for training the normalised features; finally, in [6] they propose a new learning process based on angular softmax loss function in order to learn more discriminative features of their CNN, called SphereFace.

Regarding the results obtained by existing models, DeepFace [7] achieved a verification accuracy of 95.92% with the Labeled Faces in the Wild (LFW) dataset [8] [9]. Since that moment, more complex models of deep learning architectures have been published, such as the models mentioned in the previous paragraph; reaching the latest models published that outperform the previous ones in accuracy. The most recent ones are the ArcFace model [10], Circleloss-ResNet34 [11] and Prodpoly-ResNet [12] models, where a 99.53%, 99.73% and 99.83% in verification accuracy over the LFW dataset are achieved respectively. Typically, these complex models require specialized hardware to run, such as the ”Nvidia Titan” style GPU. These models cannot be run on devices with limited computing capabilities. In [13] the most recent works in this field can be found, including the current challenges of facial biometrics (different poses, changes in lighting or expressions, among others).

To design such powerful neural networks, specialized hardware is needed to reduce the training/inference time. Different proposals have emerged that allow the complexity of these networks to be reduced, such as the use of Binary Networks [14], [15], Network Pruning [16]–[18] or Mimic Networks [19], [20], among others. With these methods the time of training and inference is improved, supposing a small loss of precision. However, in the field of face verification, it is expected that the algorithms to be used are robust and, above all, do not allow the appearance of false positives. Nevertheless, the computation power needed is still too great to be executed in devices with low computation resources. Taking this into account, the aim of this article is the presentation of a new face verification system called Light Intrusion-Proving Siamese Neural Network, LIPSNN, that allows:

1. Creation of a facial verification system capable of being executed in devices with computational limitations.
2. A highly effective facial verification system to be obtained, against possible supplanting of authentication. That is to say, that it minimizes the occurrence of false positives.

The proposed model will use an architecture based on Siamese Convolutional Neural Networks. These architectures are based on the fusion of two parallel networks on which a cost function is applied, whose main task is to classify the characteristics formed from the networks. In consequence, a Siamese network consists of replicating part of the architecture of a neural network, and then merging them into one or more common layers, which allows results to be obtained through the entire previous process of both replications. This allows us to compare two inputs, in our case two images of people, extract the ”characteristics” of each of the inputs and perform any type of classifying method, which defines an output easily interpreted as a result, in this case an affirmative or negative depending on whether the person is correctly verified. The first Siamese models for facial verification emerged at the beginning of this century [21], where they used dimension reduction methods to later compare the characteristics between pairs of images. Later on, new models of Deep Learning appeared, using new ways to extract characteristics from users, as a multi-task learning of False Rejects, FR, and age estimation approach [22]; or a heterogeneous face recognition model published in [23] that consists of a visible and near-infrared pairs of images as input made thanks to a siamese network. However, the structure has always been the same and has not changed in its basic form: a symmetrical and independent part, together with a comparison between the two.

The rest of the paper is organized as follows. Section II presents the proposed Light Intrusion-Proving Siamese Neural Network architecture. Section III Describes the four steps performed to develop the model. Section IV provides the results of the multiple evaluations developed to compare the LIPSNN model performance with many of the traditional and deep learning models published. Section V presents the conclusions and future work.

II. LIPSNN Architecture

The architecture of Light Intrusion-Proving Siamese Neural Network is based on traditional Siamese networks but it introduces new characteristics that improve their performance for face verification fundamentally in two ways: Increasing the detection false positives and Reducing the latency. In addition, this must be achieved with limited computational resources. To do this, an architecture with two consecutive blocks, shown in Fig. 1 is defined, each one of them has the following characteristics and operation:

- Block I: Is a replication of two pre-trained deep learning models with exactly the same architecture, weights and biases. This block extracts the image features of two inputs. Each input is made up of the image of a facial identity, and each image is pre-processed and computed across a Convolutional Neural Network, obtaining as output one vector for each image, called *Bottleneck*.
- Block II: Is a small, light binary classifier, based on a Fully-Connected neural network. Instead of a basic point distance between the two outputs of Block I that traditional Siamese Neural Networks implement, this second block of LIPSNN model takes the two Bottlenecks obtained from the Block I, and *compares* them with the new neural network, to finally obtain the result predicting whether the pairs of images are or not the same person. Also a new penalization technique, which will be explained in detail later, has been implemented during the training phase, with the objective of incrementing the total loss of the batches with false positive cases, as these cases are considered with more priority than those of false negatives.

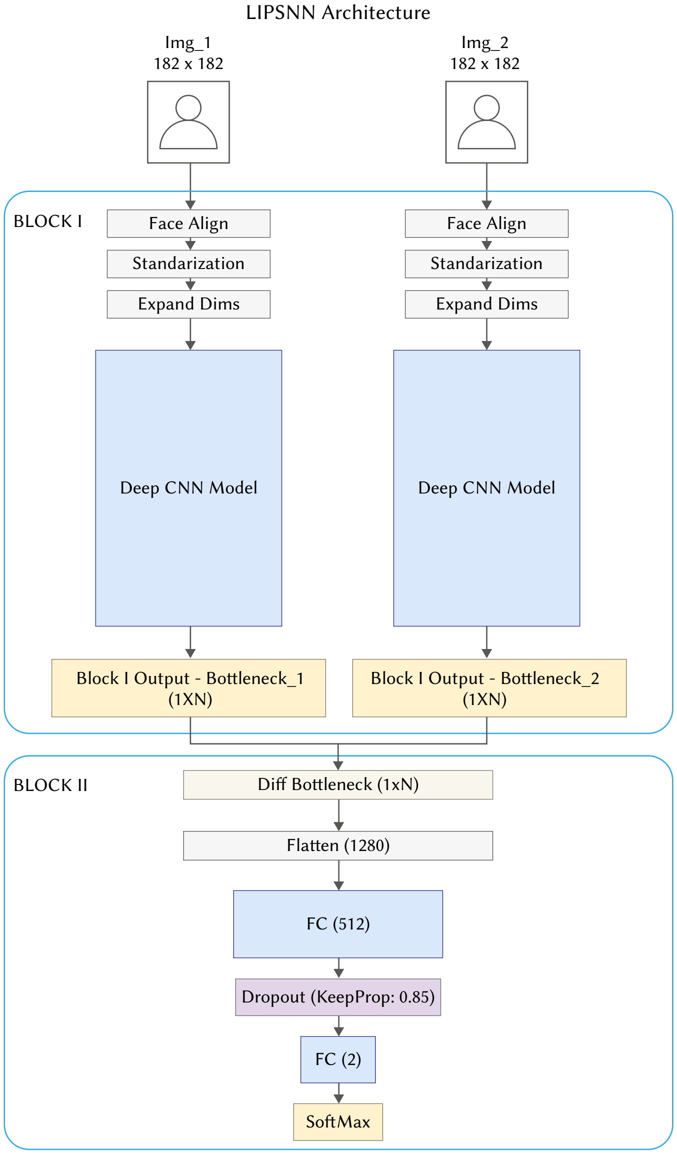


Fig. 1. LIPSNN architecture. Siamese Neural Network basis.

In the following section the architecture of each block is explained in detail.

A. Block I: Feature Extraction

As is well known, feature extraction is a crucial step in Deep Learning training steps and predictions. To deal with the solution of this problem, this block extracts all possible information from its inputs in order of greater or lesser importance depending on the selected convolutional architecture chosen for the model. Block I has the following characteristics:

- Both convolutional networks are completely identical, having the same internal structure, weights and biases.
- Both nets have already been trained and optimized by big organisations for face recognition purposes.
- The two nets have far more parameters and require more computational resources than Block II, made by only a binary classifier.
- The last layers have been removed from both blocks of each architecture made by fully-connected ones, in order to extract the bottleneck of each.
- Both bottlenecks are made of raw feature data extracted by the last convolutional output of the model architectures that has been chosen.

B. Block II: Binary Classifier

The second block is sequentially after the processing of the first Block and, as is shown in Fig. 2, consists of a supervised binary classifier that has as its input the absolute difference between each value of the bottleneck arrays, having the same size as them, this procedure allows the total number of inputs in this block model to be simplified, thus reducing the total number of parameters. As output, the model predicts whether the difference bottleneck obtained from the Block I corresponds to the same person or not. This output is normalized, giving a similarity value that is used to obtain the final prediction.

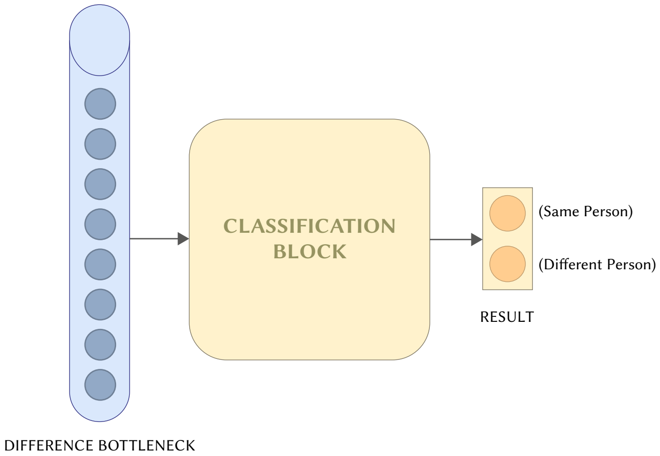


Fig. 2. Block II High-level Architecture.

The structure of the Neural Network consists of two Fully-Connected, FC, layers with 512 and 2 neurons respectively. It has been previously considered to output a single probability instead of two different values for both positive and negative results. However, this architecture has been used in order to collect every specific similarity exclusively when cases are positives, and evaluate them based on the probability threshold. For every positive value that doesn't reach the threshold set, it is automatically discarded as positive and changed into a negative case.

Moreover, as is shown in Fig. 3 a flatten layer has been added before the first FC layer in order to prepare the input dimensions; a dropout regularization technique in order to reduce the over-fitting; and a soft-max layer at the end of the network to normalize the outputs.A Sigmoid activation function has also been used to determine the output of the network.

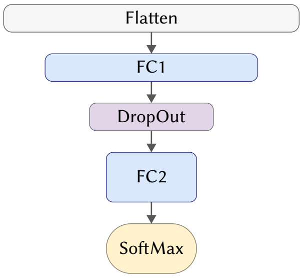


Fig. 3. Block II Low-level Architecture.

III. LISPNN Development

Having established the Light Intrusion-Proving Siamese Neural Network architecture in the previous section, in this section the process used to define will be is presented. This process consists of the following four phases:

A. Neural Network Model Selection

As described before, Siamese Neural Networks need a model that is used for the feature extraction before the comparison step, and in LISPNN this has been implemented with four models of two types specialized in facial purposes, with the objective of outperforming the current results in facial verification and exploiting the strengths of each of them. The four models used for the Block I, classified by type, are the following:

- InceptionResNet. This the first type of model used. From this type two versions have been used:
    1. InceptionResNetV1. This is a combination of two deep learning models with different characteristics: InceptionV3 [24] and ResNet [25].
    2. InceptionResNetV2 [26]. This is a second version and improvement of a combination of two previous architectures: Inception V4 [26] and the residual network techniques of ResNet's [25]. This net gives significant results thanks to the residual techniques, accelerating the training of Inception networks significantly with lower resources compared to others
- MobileNet. This is the second group of models used. They are designed to run in lighter environments, using fewer parameters. Two versions have been used as well:
    1. MobileNetV2. [27] This is an architecture specialised in light and resource-limited devices, an improvement of its previous version MobileNetV1. This net implements the residual connections technique, based on ResNet architectures.
    2. MobileNetV3 [28]. This has overtaken its previous versions in accuracy and latency, latency being one of its biggest improvements. Thus, the Siamese Network implemented in this model is one of the most promising architectures of the moment, considering the low number of parameters.

These pre-trained models can extract a huge amount of information from each facial image used as input compared to traditional Convolutional Neural Networks. In other words, they can extract as much information as traditional ones in a faster way.

B. Pre-Processing

For the training and inference steps or operations, the model needs to receive as inputs the images ready for it. To do so, the images received as inputs are treated using three techniques:

1. Normalization. Align and cropping techniques have been employed, using libraries, such as the Multi-Task Cascaded Convolutional Neural Networks (MTCNN) [29], employing the model to put great effort in noisy data and focus on the main problem of facial verification. During the construction of the training and evaluation data-set, all images have been prepared with this normalization process.
2. Data Augmentation. The data has been *augmented* or, in other words, new multiple data has been created based on the original ones during the training process, creating a more complete dataset. Moreover, random rotating and exposition, sizing, flipping and cropping techniques have been implemented on each training image.
3. Data-set filtering. For proper data preparation, the training data-set has been created with the selection, from the Labeled Faces in the Wild, LFW, data-set (this data set will be described in the following section *Training*), of the classes of people that have more than 15 images each. This technique increases the ease of learning for the model and, to avoid unbalance between classes, an upper limit of images per person has been established.

C. Training

This section presents the features of the LISPNN model training. It begins by describing the data set used, followed by a description of the details of the training of each block and finishes by describing the training of the model as a whole.

1. Training Data Set

The model has been trained with the well-known Labeled Faces in the Wild (LFW) dataset [8] [9], mentioned in the previous section. LFW contains images with faces of famous people obtained through the Internet. The dataset contains 13233 images of 5749 different people, where 1680 people have more than one image. Every person has a varying number of images in the database but, 1680 persons have at least two distinct images.

2. Training Block I

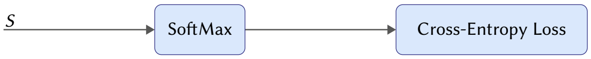
During the training phase, Block I has remained constant, as the architectures of Inception-Resnet and MobileNet are previously trained by external users. Inception-ResNet-V1 has been obtained from a contribution by David Sandberg [30] based on FaceNet [3], a face recognition system developed by researchers at Google with many competitive results. The model was trained by the VGGFace2 [31] training dataset available here. Inception-ResNet-V2, MobileNet-V2 and MobileNet-V3 architectures, weights and bias are obtained from the official GitHub Tensorflow repository [32], trained on the ILSVRC-2012-CLS image classification dataset [33].

These pre-trained models have been used to build the Block I architectures, removing the last classification layers and keeping only the ones that are used to make the feature extraction of the future input of images; and then taking the weights and biases of each of them, excepting those last layers.

3. Training Block II

Block II, however, has been trained in order to optimize its weights and biases so that it can get the best performance related to the minimization of false positive cases. To do so, a new technique has been developed during the training step.

The new technique is implemented in the loss strategy used, in order to reach better back-propagation results. In this case, a Softmax Cross-Entropy Loss strategy (also called Categorical Cross-Entropy loss) has been included, shown in Fig. 4 which consists of a Softmax activation plus a Cross-Entropy loss. By using this technique, the model output will be the probability over the C classes for each image. As the total classes of the full architecture are two (same person, and different person), the model will directly get as output the probability of both images being the same person in one class, and vice versa.



$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \quad CE = -\sum_i^C t_i \log\left(f(s)_i\right)$$

Fig. 4. Softmax Cross-Entropy loss function and equation. [34].

Where, for a given class s_i, C is the number of classes; s_j are the scores inferred by the net for each class in C, t_i and s_i are the groundtruth and the CNN score for each class i in C.

The technique implemented in this paper consists of a **loss penalization in case of False Positives**. The model will modify and increment its loss proportionally to the number of False Positives found in every training input batch. The Equation 1 represents the loss function approach proposed.

$$Loss_f = \begin{cases} softmaxCrossEntropyLoss(logits, labels) * \alpha \\ if\ FP > 0 \\ softmaxCrossEntropyLoss(logits, labels) * 1 \\ if\ FP = 0 \end{cases} \tag{1}$$

Alpha determines the number of False Positives found per batch of predictions during the training phase; and *softmaxCrossEntrophyLoss* function calculates the loss between the predicted array results (logits) and the real array labels. This process allows the appearance of False Positives in the LIPSNN architectureto be penalized, this being this one of the main objectives of our proposal. With regards to mini-batches with false positives, the correct predictions are not only the ones without false positives, but also the ones with false negatives, suffering a considered loss that the model will process. However, in our approach, any prediction obtained from the ones that have false positives is considered as a common cross-entropy loss output, and each false positive found will linearly increment the total loss of the corresponding batch. Thus a proportionally higher loss is obtained if the total number of false positive cases increments. A batch with no false positives will generate usual loss according to every prediction. A variation of this α parameter permits a simple calibration of false positive/negative proportional rates to be generated. By increasing this value, it will proportionally increase the loss of the batches where false positives are detected, being stricter in the intrusion cases compared with false negative ones, and vice versa.

4. Training LIPSNN Model as a Whole

The training consists of a multiple hyper-parameter optimization for each of the possible combinations. These (hyper-)parameters are the following:

- Architecture of Block I: These are the pre-trained models used for the feature extraction of each image. Four models have been added:
  1. Inception-ResNet-V1
  2. Inception-ResNet-V2
  3. MobileNet-V2
  4. MobileNet-V3
- Seed: integer used for the weights and biases initialization state of a pseudo random number generator. Used for randomization control. Seeds set used:

  $$seed \in \{13, 25, 29, 31, 42, 51, 67, 80, 90\}$$
- Batch size: number of images per each training and evaluation iteration. Used:

  $$batch\_size \in \{8, 16, 32\}$$
- Max steps: maximum number of iteration in each training and evaluation step. Parameters set from 250 to 2000 steps.

  $$max\_steps \in \{250, 500, 1000, 2000\}$$
- Dropout: regularization technique for reducing overfitting in neural networks by preventing complex co-adaptations on training data. In other words, it consists of *dropping out* random neurons from the net. It has been kept in a 0.85 dropOut-Keep-Probability, which means a 15% of dropout.
- Learning Rate: determines the step size at each iteration while moving toward a minimum of a loss function. It represents the learning speed of a model. 0.01, 0.001 and 0.0001 Learning Rates have been used.

  $$learning\_rate \in \{0.01, 0.001, 0.0001\}$$

By combining these hyper-parameters, a total of 576 models have been trained in a laptop “Xiaomi Mi Laptop Pro 15,6 inch Intel Core i7-10510U NVIDIA GeForce MX250 16GB DDR4 RAM”. The software libraries and frameworks: Python 3.6.8, Tensorflow 1.14.0, Numpy 1.16.4, and OpenCV 3.4.2 .

D. Model Architectures: Comparison

In this section, experimental results for the evaluation of the four architectures implemented in LIPSNN model are given. Two different branches have been considered:

- Efficacy. Efficacy is related to the real performance, depending on the model precision, accuracy, etc.
- Efficiency. Efficiency is related to the model performance depending on the latency times, which mostly has the same behaviour as the total number of parameters per architecture.

For each model represented, their effectiveness and efficiency are presented and compared. After that, a unique architecture is chosen for the model results.

For the collection of the results, the evaluation dataset used is the official LFW test dataset [8], which consists of a thousand face pairs similar to the entire LFW dataset; five hundred image pairs with positive results (both images are the same person) and another five hundred with negative results (face images are not the same person). Each image pair also contains the result of the comparison, whether is positive or negative (1 and 0 respectively). It can be considered that, for a better robustness level of the results, a test dataset based on ten thousand image pairs with same number of positive and negative results has been used during the evaluation. Those image pairs obtain a better approximation of the effectiveness of the model.

TABLE I. Hyper-Parameter Selection of Each Architecture and Their $F_{0.5}$ Values

Architecture	BatchSize	MaxSteps	DropoutKeepProb	LearningRate	Seed	*Max$F_{0.5}$Score*
InceptionResNetV1	8	500	0.85	0.001	29	0.974555
InceptionResNetV2	16	250	0.85	0.001	13	0.959028
MobilenetV2	8	250	0.85	0.001	13	0.959071
MobileNetV3	16	250	0.85	0.001	80	0.958183

1. Efficacy Evaluation

For the evaluation of the efficacy part, the $F_{0.5}$ score has been used as the most relevant metric. It is based on the F_β metric. Chinchor[53] defines this metric which will be used further on:

$$F_\beta = \frac{(\beta^2 + 1) \cdot PR}{\beta^2 \cdot P + R} \quad (0 \leq \beta \leq +\infty) \tag{2}$$

Where P refers to the precision and R refers to the Recall obtained on each evaluation, and β is the parameter used to give a proportional weight importance between the Precision and Recall. Both P and R (True Positive Rate) are inversely related to the proportion of False Positives and False Negatives respectively.

For this study, a case that the False Positive errors are crucial to minimize, the β chosen has a value of 0.5. Thus, a bigger fluctuation in the value of F score will be obtained when the proportion of False Positives are modified when this is compared with a same fluctuation of False Negatives. Thanks to this metric number, the performance of the models that minimize the False Positive Rate can be better justified without ignoring the False Negatives Rate, a problem that appears when a hyper-parameter combination that minimizes the False Positives Rate is only considered, generating inappropriate results of the false negatives that are really high.

From the $F_{0.5}$ values obtained on each architecture, those that maximize this metric have been used. The Table I shows the hyper-parameter combination that maximizes each architec ture evaluation, Inception-ResNet-V1 obtaining the highest value.

Moreover, by getting the best hyper-parameter selections of each of the four architectures, an evaluation of the FPR and TPR has been made by making a complete variation of the similarity threshold and a representation of the results in the Receiver Operating Characteristic (ROC) curve in Fig. 5.

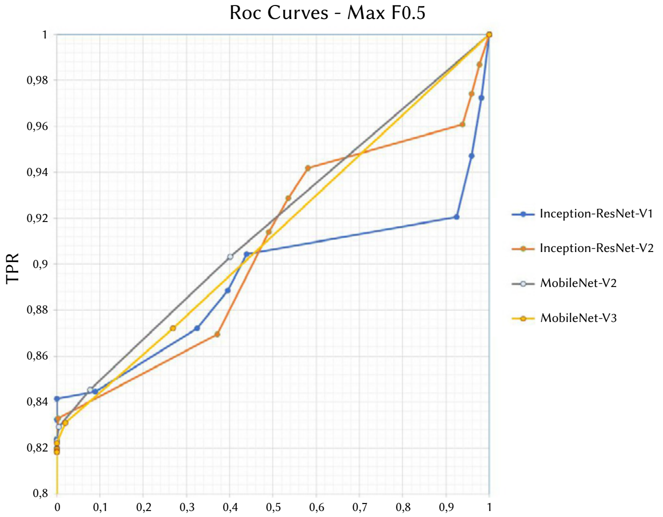


Fig. 5. ROC Curves of each architecture that maximize F0.5 values.

All architectures used for the LIPSNN model have similar results in this curve. As mentioned before, this happens as the training process has been carried out only in the second part, Block II, which consists of a basic neural network binary classifier.

After looking into the ROC curve, in Table II the way the Area Under the Curve (AUC) behaves can be seen.

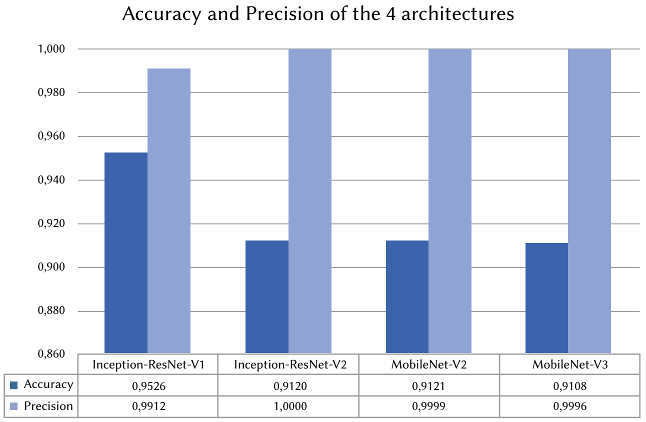
TABLE II. AUC Values of Each Architecture That Maximize $F_{0.5}$ Values

Architecture	AUC (Max $F_{0.5}$) (M)
InceptionResNetV1	0.903
InceptionResNetV2	0.921
MobilenetV2	0.957
MobileNetV3	**0.965**

Surprisingly, the Mobile-Net architectures reach a total of 0.965 and 0.95 values, outperforming the Inception-ResNet ones, with a total of 0.92 and 0.90. This anomaly happens due to the generalization that is made up from the AUC compared to the individual selection of hyper-parameters obtained in the table I. In other words, the models in the table are made up exclusively of one hyper-parameter setting per architecture, obtaining the best performing results possible, whilst the ROC and AUC may contain models of each architecture that decreases the total results per architecture, as they are based in multiple results depending on the similarity threshold.

Once the models are represented and analysed through the $F_{0.5}$ metric, it has been considered an accuracy and precision analysis used for the final conclusions of the architecture performance evaluation in terms of efficacy.

In Fig. 6, it is clearly seen that the results of both metrics are based on the hyper-parameter selection that maximizes $F_{0.5}$.



	Inception-ResNet-V1	Inception-ResNet-V2	MobileNet-V2	MobileNet-V3
■ Accuracy	0,9526	0,9120	0,9121	0,9108
■ Precision	0,9912	1,0000	0,9999	0,9996

Fig. 6. Accuracy and precision of the four architectures, using parameters that maximize $F_{0.5}$ values according to Table I.

The three Inception-ResNetV2, MobileNet-V2 and MobileNet V3 architectures have obtained similar results from both attributes, having almost 100% precision and around 91% accuracy each. On the other hand, the Inception-Resnet-V1 has obtained better accuracy by reducing the precision of the model. This contrast needs to be evaluated, as the main goal and purpose of this work consists in proposing a model that can ensure biometric authentication security by reducing access of undesirable intruders (False Positives), but at the same time it should remain an effective access for people that are allowed to access the hypothetical system.

2. Efficiency Evaluation

$F_{0.5}$, Box Plots, ROC curve and AUC, and Accuracy and Precision attributes are those used for an efficacy performance evaluation. In addition, efficiency evaluation is half of the importance when evaluating this model focused on mobile devices, where the latency and computation times are crucial for a proper evaluation.

Consequently, two tables are represented below in order to visualize the size of each architecture and their latency. Table III represents the total number of parameters that form each architecture, and Table IV shows the inference times they gave during the evaluation step, made in an isolated environment that allows the correct comparison between them.

TABLE III. Total Parameters Per Architecture Implemented for LIPSNN

Architecture	Parameters (M)
InceptionResNetV1	43
InceptionResNetV2	55.8
MobilenetV2	**6**
MobileNetV3	**5.4**

In Table III each of the sizes of the four architectures can be compared with the previous tables [27], [28], [35]. The Inception-ResNet-V1 value is represented as an approximation range due to the lack of information found in other publications. Szegedy, in his publication [26], affirms that this first version of Inception- Resnet has a lower number of parameters than its successor. In any case, it can be seen that both MobileNet architectures have values approximately ten times lower than Inception-ResNet ones.

TABLE IV. LIPSNN Inference Times of Each of the Four Architectures Used in Ms

Architecture	BlockI Inf	BlockII AvgInf	BlockII StdInf	Total TotalInf
InceptionResNetV1	121.2	3.5	0.3	124.7
InceptionResNetV2	271	1.3	0.2	272.2
MobilenetV2	52.5	3.4	0.5	**56**
MobileNetV3	67.1	1.2	0.1	**68.3**

In Table IV, the most relevant aspects to consider are the huge difference between the latency of Inception-ResNets and Mobile- Net ones. The first ones, with 124.7 and 272.2 milliseconds of processing time for each prediction. The second ones, surprisingly, reach a latency of 56 and 68.3 milliseconds. It is also interesting to see that most of the times are related directly to the times of Block I, as this block is the one with the pre-trained architectures, which is bigger than the binary classification model included in Block II. Moreover, it is important to mention that the total inference time in Block I is considered as a sequential computation of the two inferences of each image to be compared included as inputs. This means that the model considered as the worst case of inference can perfectly be implemented in a parallel computing hardware in order to make the inferences at the same time.

After all the analysis of the different architecture performance evaluations of efficacy and efficiency, considering the total accuracy and precision, the ROC curve and AUC, and the total number of parameters per architecture as well as the total number of hyper-parameters, the two networks that perform best in this analysis are the Mobile-Net architectures. Moreover, based on multiple publications [27], [28], [36], [37], the Mobile-Net-V3 generates much better performance than MobileNet-V2 in similar fields such as facial recognition and segmentation [28]. Thus, for the public model performance comparison shown in the following section, the Mobile-Net-V3 is the chosen architecture to be used as for the LIPSNN model evaluation.

IV. LIPSNN Performance Benchmarking

In this section, the results of the various evaluations that have been carried out in order to reach a comparison between the Light Intrusion-Proving Siamese Neural Network model developed and some of the best known models published currently for dealing with this problem are presented. The results obtained from the LIPSNN model to compare with the rest of publications are based on the ROC curve, a metric frequently used in this area. It can represent the False Positive rate, as well as indirectly the False Negative one.

A. Performance Benchmarking Attributes

Taking the purpose of this study into account, four attributes are used for the benchmarking of this research:

- Number of Parameters of the model. This attribute measures the model size with the number of parameters that the model needs. A higher number of parameters would decrease the efficiency of the model, as it would proportionally increase its inference time. Thus, a lower number of parameters of the model allows inferences to be executed faster than a model with high number of them. The metric used to measure this attribute is millions of parameters used, (Millions).
- Storage Space. This attribute measures the model size in information units. As the number of parameters, a higher storage space would increase the total evaluation time and hence the efficiency of the model. That means, the model will be slower in terms of each facial verification. The metric used for this attribute is Megabytes (MB).
- Accuracy. This attribute describes how the model performs across all classes: positives or negatives. It is calculated as the ratio between the number of correct predictions to the total number of predictions, and it is ranged between 0 and 1, or in percentage. A number close to 1 means a model with many correct predictions, either positive or negative. Thus, as mentioned in previous sections, the efficacy of the model would be incremented with an accuracy close to 1 and hence the verification error cases will be reduced. The metric used for this attribute is Percentage (%).
- Precision. This attribute describes the ratio between the number of Positive samples correctly classified to the total number of samples classified as Positive. Unlike the accuracy metric, precision describes how the model can perform with the False Positive cases. It is ranged between 0 and 1, or in percentage. A precision close to 1 proportionally means a low False Positive ratio. The metric used for this attribute is Percentage (%).

As the main objectives of this research is to enhance the facial verification performance with a model with the lowest computational resources used and, considering all previous attribute definitions mentioned above, what would be needed for this case is a model with the highest precision result possible, with the lowest number of parameters and storage space. With regard to the accuracy, a lower result of this attribute would not be crucial for the main objective, as what is really needed is to reduce the number of parameters and the storage space whilst precision remains at least at the same level, avoiding the cases of False Positives. This last attribute is crucial in this facial verification approach in order to minimize these cases.

B. Performance Benchmarking Data Set

For the evaluation and collection of the results, as in the comparison of the architectures used for the model, the LIPSNN model has been evaluated with the official LFW test dataset [38], made up of 1000 face pairs equally divided in 500 same and 500 different face matches. The LIPSNN model has been trained with 5760 face pairs based on the Labeled Faces in the Wild (LFW) dataset, equally distributed between negative and positive matches. Any of these training images are not overlapped with the testing ones, in order to avoid any over-fitting and biases between them.

In the evaluation dataset, each image pair contains the information about each of both images to introduce into the model, and the result of the comparison or label: '0' when the images are different people or '1' when they are the same person. This dataset is used in other models, whose results are included in the next sections.

C. Traditional Models Benchmarking

The early facial verification models do not implement any deep learning techniques as is done nowadays. The models use multiple facial verification techniques such as the large multiple distance metrics used by the LM3L model [39]; the Linear Discriminant Analysis (LDA) coupled with "Within Class Covariance Normalization" (WCCN) in the DDML model [40]; or the intra personal focused feature extraction of the Similarity Metric Learning (SML) method [41]. A table of the traditional models obtained is shown in Table V. But although their approximations to solve the problem are quite different, and it is not possible to find in the literature results that include all the attributes defined for this research, because only their accuracy has been published, it is interesting to analyze their results with the same evaluation dataset that this work has used. Those results are in Table V.

TABLE V. Accuracy Benchmarking Between Traditional Models

Model	Accuracy (%)
PCCA	83.80
PAF	87.77
CSML + SVM	88.00
SFRD + PMML	89.35
LM3L	89.57
Sub-SML	89.73
DDML	90.68
VMRS	91.10

From the table it can be seen that although those models achieved acceptable accuracy results, because the models mainly make correct predictions, they still have results that can be outperformed with the new models. Moreover, there is a lack of information in these their results using this approach, as the number of parameters, storage space and precision would enhance the conclusions about them related to this study.

D. Deep Learning Models Benchmarking

Using the four attributes defined, number of parameters, storage space, accuracy and precision, the performance of twelve deep learning models for facial verification, including LIPSNN, have been compared. For most of them, eight models, it has been possible to obtain three of the four attributes used; number of parameters, storage space and accuracy from the published literature [42] [12] [11] [43] [3] [44] [45]; for other, CenterLoss [4] only two attributes have been published, parameters and storage space, with a result of 19.6M for the first and 99.28MB for the second; and for the other three, LBPNet LBPNet [46], High-dim LBP [47], and DeepID2 [48], it has only been possible to obtain their accuracy from the literature, with a result of 94.04%, 95.17% and 95.43% respectively.

The results of the bechmarking of the LIPSNN model, along with the rest of deep learning models for those which have at least three of the four attributes defined, can be found in Table VI. The same results with the addition of the model which only has results for parameters and storage space, that is, CenterLoss, are also presented in Fig. 7. To make the comparison between models easier and the impact of each attribute, and also their combination clearer in the performance of the model, Fig. 7 has been developed using a normalized value, between 0 and 1, of the attributes. This normalization has been calculated for parameters and storage space by dividing the rest of each value minus the maximum value observed for the attribute by the maximum value observed for the attribute minus the minimum value observed for it. To obtain the right scale for this study, in which, the minimum values for those attributes are the best ones, the amount obtained in the previous calculation has been subtracted from one. As attribute accuracy is directly measured in an scale from 0 to 100, its normalization to a range between 0 and 1, is obtained dividing each observed value by 100.

TABLE VI. Attributes Benchmarking Between Deep Learning Models

Model	Parameters (Millions)	Storage Space (MB)	Accuracy (%)
CircleLoss-ResNet34 [11]	60.5M	83MB	99.73
DeepFace [43]	120M	488MB	95.92
FaceNet [3]	140M	186MB	98.87
Light CNN A [44]	3.96M	26MB	97.97
Light CNN B [44]	5.56M	32.8MB	98.80
LIPSNN	5.50M	65.9MB	91.08
Prodpoly-ResNet [12]	14.70M	181.8MB	99.83
VGG [45]	27.75M	533MB	97.27

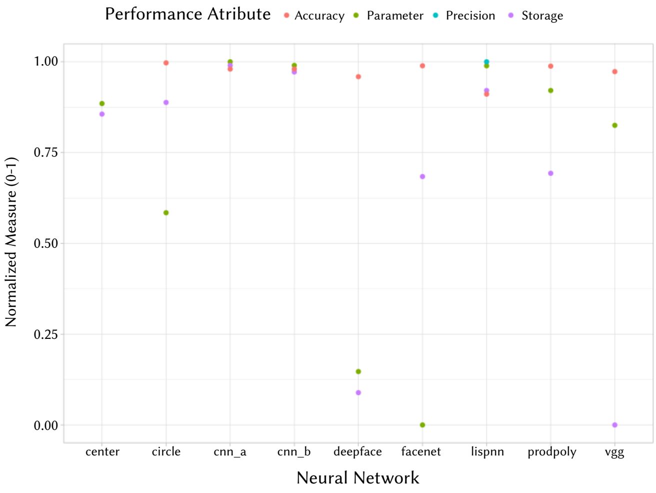


Fig. 7. Attributes Benchmarking between Deep Learning models.

The main purpose of this research was to reduce as far as possible the attribute parameters and storage needed by the deep learning neural network developed in order to be able to operate it in devices with low computation resources, achieving this without a signification loss of accuracy and precision, especially the latter, in the results in its application to facial verification. From Table VI and Fig. 7 it can be seen that both things can be obtained, because the normalized value, between 0 and 1 for parameters is 0.989 and for storage 0.921, both of them being very close to 1, which means that the computational resources needed by LIPSNN are very low; and that when they are compared with the other models are better than most of them, and in some cases significantly better, and are comparable with the best models.

At the same time that the desired results for parameters and storage have largely been obtained, the purpose of maintaining acceptable results on the second two attributes, accuracy and precision has also been achieved. A normalized accuracy of 0.911 is very close to 1. When it is compared with the rest of the models it is slightly lower but it is acceptable taking into account two important related considerations: the first is the good results obtained for parameters and storage, which are better than most of the published models; and the second, even more importantly, is the excellent results obtained for precision, which in its normalized value is 0.9996. It is not known if it has greater or lesser precision than the current models because this value has not been published for them, but with the obtained measure the other models can only present the same or lower value.

At this point is important to point out that the LIPSNN model has focused on a maximization of precision more than in the maximisation of accuracy. This is because the main goal in this work is to reduce the total number of intrusions in the model by reducing the False Positives ratio, and that is achived by increasing precision. Although accuracy must have as high value as possible, it has less importance precision in a facial verification when it is used alone. This is because, as it gives us together the number of the total false positives and negatives of a facial verification study, it doesn't give enough information to make it possible to distinguish between them in the results. For that reason, high measures of accuracy only means a good performance of the model when this is combined with high levels of precision and not only by itself.

As we have seen, LIPSNN is a lightweight network (it can be used in devices with scarce computational resources) with significantly good accuracy. However, the most important advantage of LIPSNN over other architectures is that the False Positive ratio is practically zero, thanks to the loss function used (see Fig. 7). This capability makes it especially useful for the verification of persons in critical services or facilities.

From the presented benchmaking study it can be concluded that the LIPSNN models have a parameters and storage results, with very low measures from both of them, that allow to it to be included within the group of the lightest models and that the LIPSNN model is a viable alternative model to them. If the sizing levels are combined with the precision results obtained, one of the most important conclusions is that, whilst the LIPSNN model doesn't achieve as good performance results in accuracy as if it does in precision, these are, nonetheless, good enough. It has an architecture and development different form the other previously published models, that is adaptable and modular, as is the case, for example, in block I, whose architecture could be replaced by lighter and more accurate and precise future deep learning models, which would allow its future improvement to continue and open up the development of a new alternative deep learning facial verification model, different from the existing previous ones.

V. Conclusions and Future Work

This study presented a solution focused on the design and development of a new deep neural network model for facial verification focused on the computational reduction resources needed for it to operate. The aim was for it to able to be executed in portable devices like mobile phones or tablets without loosing precision in the results. To achieve this the two sided problem of reducing parameters and storage without losing accuracy and precision has been overcome thorough the development of a new architecture and development of a Siamese Neural Network called Light Intrusion-Proving Siamese Neural Networks, LIPSNN.

For the development of the architecture, two blocks have been designed. For the first block, four pre-trained architectures specialized in facial verification were been analyzed in order to extract the face characteristics. For the second block, a binary neural network with a new loss function was developed in order to optimize the false positive cases. After the definition of the architecture, a process with four phases: model selection, pre-processing, training and architecture comparison, to define the neural network was carried out. All these processes were performed with the well-known LFW dataset and the following architecture models: Inception-ResNet-V1, Inception-ResNet-V2, MobileNetV2, MobileNetV3.

Once the LIPSNN design and development was finished, a performance benchmarking study with some of the best known models published nowadays for dealing with this problem was carried out. For this, four attributes were used: Number of Parameters of the model, Storage Space, Accuracy and Precision. The first two measure the size needed to execute the model, the second two, its validity for face recognition. From this comparison study it was concluded that the LIPSNN model requires a low number of parameters and storage needs that allow it to be classified in the set of lightweight models, while at the same time presenting very high levels of accuracy and precision, especially the latter. As the model presents quite different architectures and design to the previous light models published in the literature, it constitutes a new alternative to those ones.

Once a lightweight deep learning model based on a new architecture and design has been obtained, future research will focus on a continuous improvement in the reduction of the parameters and storage measures obtained and increasing accuracy without losing precision. To achieve this fundamentally it will focus on: the incorporation of new pre-trained architecture models in block I, as the LIPSNN model allows the modification of the architecture of the block where the facial features extraction is carried out; training using new modern and sophisticated datasets; and the design and development of new architectures of the binary classifier of Block II.

Acknowledgment

This study was funded by the private research project of Company BQ, the public research projects of the Spanish Ministry of Economy and Competitiveness (MINECO), references TEC2017-88048-C2-2-R, RTC-2016-5595-2, RTC-2016-5191-8 and RTC-2016-5059-8, and the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with UC3M in the line of Excellence of University Professors (EPUC3M17), and in the context of the V PRICIT (Regional Programme of Research and Technological Innovation).

References

[1] A. Suruliandi, A. Kasthuris, S. P. Raja, "Deep feature representation and similarity matrix based noise label refinement method for efficient face annotation," *International Journal of Interactive Multimedia and Artificial Intelligence*, In Press, 2021, doi: 10.9781/ijimai.2021.05.001.

[2] Imatest, "Automated image quality analysis," 2020. [Online]. Available: http://www.imatest.com/products/test-charts-sfrplus/.

[3] F. Schroff, D. Kalenichenko, J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June-2015, 2015.

[4] Y. Wen, K. Zhang, Z. Li, Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9911 LNCS, 2016.

[5] F. Wang, X. Xiang, J. Cheng, A. L. Yuille, "NormFace: L2 hypersphere embedding for face verification," in *MM 2017 - Proceedings of the 2017 ACM Multimedia Conference*, 2017.

[6] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, L. Song, "SphereFace: Deep hypersphere embedding for face recognition," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-January, 2017.

[7] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014.

[8] G. B. Huang, M. Ramesh, T. Berg, E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, October 2007.

[9] G. B. Huang, E. Learned-miller, "Labeled faces in the wild: Updates and new reporting procedures," *University of Massachusetts Amherst Technical Report*, 2014.

[10] J. Deng, J. Guo, N. Xue, S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

[11] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, Y. Wei, "Circle loss: A unified perspective of pair similarity optimization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6398–6407.

[12] G. G. Chrysos, S. Moschoglou, G. Bouritsas, J. Deng, Y. Panagakis, S. P. Zafeiriou, "Deep polynomial neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[13] U. Jayaraman, P. Gupta, S. Gupta, G. Arora, K. Tiwari, "Recent development in face recognition," *Neurocomputing*, 2020, doi: 10.1016/j.neucom.2019.08.110.

[14] M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi, "XNOR- Net : ImageNet Classification Using Binary," *Eccv2016*, 2016.

[15] T. Simons, D. J. Lee, "A review of binarized neural networks," 2019. doi: 10.3390/electronics8060661.

[16] Z. Wang, F. Li, G. Shi, X. Xie, F. Wang, "Network pruning using sparse learning and genetic algorithm," *Neurocomputing*, vol. 404, 2020, doi: 10.1016/j.neucom.2020.03.082.

[17] F. Tung, G. Mori, "Deep Neural Network Compression by In-Parallel Pruning-Quantization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 3, 2020, doi: 10.1109/TPAMI.2018.2886192.

[18] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, C. Zhang, "Learning Efficient Convolutional Networks through Network Slimming," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, 2017.

[19] Q. Li, S. Jin, J. Yan, "Mimicking Very Efficient Network for Object Detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7341–7349.

[20] Y. Wei, X. Pan, H. Qin, W. Ouyang, J. Yan, "Quantization Mimic: Towards Very Tiny CNN for Object Detection," in *Computer Vision – ECCV 2018*, Cham, 2018, pp. 274–290, Springer International Publishing.

[21] S. Chopra, R. Hadsell, Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, vol. I, 2005.

[22] X. Wang, Y. Zhou, D. Kong, J. Currey, D. Li, J. Zhou, "Unleash the Black Magic in Age: A Multi-Task Deep Neural Network Approach for Cross-Age Face Verification," in *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*, 2017, pp. 596–603.

[23] C. Reale, N. M. Nasrabadi, H. Kwon, R. Chellappa, "Seeing the Forest from the Trees: A Holistic Approach to Near-Infrared Heterogeneous Face Recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2016.

[24] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, "Rethinking the inception architecture for computer vision. arxiv 2015," *arXiv preprint arXiv:1512.00567*, vol. 1512, 2015.

[25] K. Z. He, K. Zhang, "X., ren, s. & sun j. deep residual learning for image recognition," *Preprint at https://arxiv. org/abs/1512.03385*, 2015.

[26] C. Szegedy, S. Ioffe, V. Vanhoucke, A. A. Alemi, "Inception- v4, inception-ResNet and the impact of residual connections on learning," in *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, 2017.

[27] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.

[28] A. Howard, M. Sandler, B. Chen, W. Wang, L. C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, Q. Le, H. Adam, "Searching for mobileNetV3," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019- October, 2019.

[29] Z. Mian, W. Hong, "Face verification using gabor wavelets and AdaBoost," in *Proceedings - International Conference on Pattern Recognition*, vol. 1, 2006.

[30] D. Sandberg, "Face Recognition using Tensorflow." https://github.com/davidsandberg/facenet, 2018.

[31] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*, 2018, pp. 67–74, IEEE.

[32] N. Silberman, S. Guadarrama, "TensorFlow-Slim image classification model library." https://github.com/ tensorflow/models/tree/master/research/slim, 2016.

[33] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, F. Li, "Large scale visual recognition challenge 2012 (ilsvrc-2012)," 2012.

[34] R. Gomez, "Understanding categorical cross-entropy loss, binary cross-entropy loss, softmax loss, logistic loss, focal loss and all those confusing names," *URL: https://gombru.github.io/2018/05/23/cross_entropy_loss/ (visited on 29/03/2019)*, 2018.

[35] S. Bianco, R. Cadene, L. Celona, P. Napoletano, "Benchmark analysis of representative deep neural network architectures," *IEEE Access*, vol. 6, 2018, doi: 10.1109/ACCESS.2018.2877890.

[36] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017.

[37] J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, "Squeeze-and-Excitation Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, 2020, doi: 10.1109/TPAMI.2019.2913372.

[38] G. B. Huang, M. Ramesh, T. Berg, E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, October 2007.

[39] H. Junlin, J. Lu, Y. Junsong, T. Yap-Peng, "Large Margin Multi-metric Learning for Face and Kinship Verification in the Wild," in *Computer Vision – ACCV 2014*, Cham, 2015, pp. 252–267, Springer International Publishing.

[40] O. Barkan, J. Weill, L. Wolf, H. Aronowitz, "Fast high dimensional vector multiplication face recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013.

[41] Q. Cao, Y. Ying, P. Li, "Similarity metric learning for face recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013.

[42] "12th Chinese Conference on Biometric Recognition, CCBR 2017," 2017.

[43] O. M. Parkhi, A. Vedaldi, A. Zisserman, "Deep face recognition," in *Proceedings of the British Machine Vision Conference (BMVC)*, September 2015, pp. 41.1–41.12, BMVA Press.

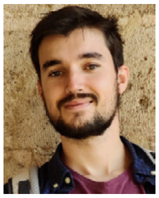
[44] X. Wu, R. He, Z. Sun, T. Tan, "A light CNN for deep face representation with noisy labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, 2018, doi: 10.1109/TIFS.2018.2833032.

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

[46] M. Xi, L. Chen, D. Polajnar, W. Tong, "Local binary pattern network: A deep learning approach for face recognition," in *Proceedings -International Conference on Image Processing, ICIP*, vol. 2016-August, 2016.

[47] D. Chen, X. Cao, F. Wen, J. Sun, "Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013.

[48] Y. Sun, Y. Chen, X. Wang, X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger Eds., Curran Associates, Inc., 2014, pp. 1988–1996.



Asier Alcaide

Asier Alcaide has a degree in Computer Engineering and Business Administration from the University Carlos III of Madrid, Spain (2020). He has participated with UC3M and other enterprises in the research and development of artificial intelligence and machine learning technologies, and he was involved in machine learning competitions in the Kaggle community. He is currently focused on big data solutions combined with AI, and new disruptive biometric sensor devices.

Miguel A. Patricio

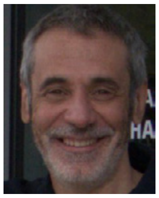
Received his BSc in Computer Science in 1991, his MSc in Computer Science in 1995 and his Ph.D. in Artificial Intelligence in 2002, all from the Universidad Politécnica de Madrid. He is currently an Associate Professor at the Escuela Politécnica Superior of the Universidad Carlos III de Madrid. He is the coauthor of over 100 books, book chapters, journal papers, technical reports, etc., published by organizations including Elsevier, IEEE, ACM, AAAI, Springer Verlag, Kluwer, etc., and most of these present practical and theoretical achievements of data analysis, computer vision and distributed systems.

Antonio Berlanga

Antonio Berlanga holds a Ph.D in Computer Science from Universidad Carlos III de Madrid (Spain) in 2000 and a B.S. degree in Physics from Universidad Autónoma de Madrid (Spain), in 1995. He is Associate Professor at the Universidad Carlos III de Madrid since 2000. His main research lines are foundation of multiobjective evolutionary computation and applications of artificial intelligence and decision support techniques in business and society.

Angel Arroyo

Received his BSc in Computer Science in 1999 from Universidad Carlos III de Madrid and his Ph.D. in Artificial Intelligence in 2011 from Universidad de Alcalá de Henares de Madrid. He is currently a Professor at the Escuela Politécnica Superior de Ingenieros en Sistemas Informáticos in Universidad Politécnica de Madrid. He has a long track record in research projects in the field of Artificial Intelligence, mainly in Computer Vision, Evolutionary Computing and Virtual Environments. In recent years, his main line of research focuses on the study of 3D persistent spatio- temporal environments, in which he has developed cognitive models and software tools for the construction of autonomous agents.



Angel Arroyo

Received a BSc in Physics in 1994 and a MRes in Physics in 1996, both from the University Complutense de Madrid; he also received a Ph.D. in Computer Science Engineering in 2001, from the University Carlos III de Madrid. He is currently an Associate Professor at the Computer Science Department of the University of Alcalá, in Madrid, Spain, and Affiliate Associate Professor at the Department of Computer Science and Software Engineering of the Gina Cody School of Engineering and Computer Science of the Concordia University, in Montreal, Canada. He is author or coauthor of over 100 scientific publications in Computer Science, principally in the fields of Data Science and Software Engineering.