Noname manuscript No.
(will be inserted by the editor)

Fusing CNNs and statistical indicators to improve image classification

Javier Huertas-Tato · Alejandro Martín · Julian Fierrez · David Camacho

Received: date / Accepted: date

Abstract Convolutional Neural Networks have dominated the field of computer vision for the last ten years, exhibiting extremely powerful feature extraction capabilities and outstanding classification performance. The main strategy to prolong this trend in the state-of-the-art literature relies on further upscaling networks in size. However, costs increase rapidly while performance improvements may be marginal. Our main hypothesis is that adding additional sources of information can help to increase performance and that this approach is more cost-effective than building bigger networks, which involve higher training time, larger parametrization space and higher computational resources requirements. In this paper, an ensemble method for accurate image classification is proposed, fusing automatically detected features through a Convolutional Neural Network and a set of manually defined statistical indicators. Through a combination of the predictions of a CNN and a secondary classifier trained on statistical features, a better classification performance can be achieved cheaply. We test five different CNN architectures and multiple learning algorithms in a diverse number of datasets to validate our proposal. According to the results, the inclusion of additional indicators and an ensemble classification approach help to increase the performance in all datasets. Both code and datasets are publicly available via GitHub at: `https://github.com/jahuerta92/cnn-prob-ensemble`.

Keywords Convolutional Neural Networks · Feature Extraction · Ensemble Learning · Data Fusion · Statistical Indicators

Javier Huertas-Tato
Department of Computer Engineering Systems, Universidad Politécnica de Madrid
Madrid, Spain
E-mail: javier.huertas.tato@upm.es

Alejandro Martín
Department of Computer Engineering Systems, Universidad Politécnica de Madrid
Madrid, Spain
E-mail: alejandro.martin@upm.es

Julian Fierrez
Biometric Recognition Group-ATVS, EPS, Universidad Autónoma de Madrid
Madrid, Spain
E-mail: julian.fierrez@uam.es

David Camacho
Department of Computer Engineering Systems, Universidad Politécnica de Madrid
Madrid, Spain
E-mail: david.camacho@upm.es

1 Introduction

Convolutional Neural Networks [43] have been positioned as an important referent in varied and well-known tasks. Taking the animal virtual cortex as a source of inspiration, these deep architectures extract patterns from data with great ease, greatly improving upon previous machine learning based approaches. These architectures have been widely used in academic and commercial contexts, exhibiting great success in multiple domains such as speech recognition or computer vision, being very powerful in many of then [44]. Given their power and wide range of applications, there is a growing trend towards developing stronger architectures, which frequently consist in making wider, deeper or denser networks. This tendency is repeatedly shown in the most recent approaches. Despite not being a convolutional network, a concerning example of this trend is GPT-3 [6], a deep language model so large (175 billion

parameters) that it requires access to a large supercomputer to be trained.

Some of the most used convolutional architectures such as VGG-16 and VGG-19 [73] employ around 140 million parameters and 25 layers of depth which, depending on the domain, require weeks of processing. Other models adopt deeper architectures with a lower number of parameters. This is the case of Inception V3 [79], which employs 25 million parameters and 150 layers. AmoebaNet [69] is another recent example of growing parameter space size, a model that requires a large increase in network size (from 90 million to 470 million parameters) to achieve a significant improvement of 1.1% accuracy in the ImageNet problem.

Although building deeper and heavier architectures allows to constantly overcome previous state-of-the-art scores, we are wasting the opportunity of leveraging additional well-known mechanisms to improve current results efficiently, and of increasing explanatory power. While CNNs are powerful methods with excellent feature extraction capabilities, their capabilities for explanation are limited when compared to, for example, decision trees, k-Nearest Neighbours, and so on. On the other hand, these explainable techniques are far less powerful than CNNs in terms of recognition. Decision trees, SVMs, among others, rely on manually extracted features which offer rich information but usually incomplete and which lack an spatial understanding of the image itself. These models are very limited in complex domains where shape, locality, and other characteristics are key to inference. Therefore, a balance must be found between performance and explainability.

The key to understanding this trade-off is that CNNs and feature methods employ different types of reasoning during the inference process. CNNs seek high performance by automatising the feature extraction process, while methods focused on manually extracted features are explainable. While different, these methods can be complementary, building a fruitful combination that can help to increase performance while maintaining a high degree of explainability by using feature-based methods. In this research, we present a curated list of statistical indicators that, once combined with the output of a CNN model through an ensemble approach, provide useful additional features that serve towards improving image classification performance. The main contributions of this research can be summarized as follows:

- A curated list of statistical indicators that allow to extract useful differentiating features from images.
- An analysis of five well-known CNN architectures and their performance in 10 different datasets.
- An ensemble approach to combine CNN architectures with classifiers trained on statistical indicators to improve performance in image classification tasks.
- A thorough experimentation, making use of 10 different image classification datasets to validate the ensemble.
- An ablation study to assess the performance of each individual statistical indicator in all combinations of datasets and CNN architectures tested.

Finally, this manuscript is organised as follows: Section 2 presents a summary of the state-of-the-art literature, with special emphasis on statistical features for image classification; Section 3 defines our proposal for combining statistical features and CNN architectures; Section 4 presents the different datasets used in the experiments, the image preprocessing steps applied, the CNN architectures tested, the hyperparameters used to train these architectures and the ML methods, details related to the execution environment, and a link to access the repository with all the code. Section 5 shows the experimental results and Section 6 outlines the main conclusions and potential lines of future work rising from this work.

2 Related work

Upscaling architectures is one of the most frequently used approaches to increase the performance of neural networks. However, as parameters increase, so does the computational and time cost of the training process. This fact highlights the importance of considering alternative methods for improving the throughput of these methods. Some authors have recently started raising concerns about this trend, from the environmental problems of the high performance computing required to train larger architectures [71] to the accessibility of Deep Learning research [77]. Although lightweight architectures are trainable by making use of standard hardware, most of the state-of-the-art models require multiple expensive GPUs resources to be trained [75]. Due to this, there is a growing interest in efficiently improving CNN architectures with minimal additions.

A potential course of action to be explored is network compression [7], which aims to reduce the size of deep neural networks while trying to maintain their effectiveness intact. Several techniques have been proposed in this direction [9,49]. For instance, Knowledge Distillation [31], which trains a student model with the predictions of an already existing larger (teacher) model. Another typical approach consists in pruning redundant or low-information weights [29,46,50] that are able

to compress state-of-the-art CNNs to achieve at least 2x speedup with minimal losses (around 1% accuracy). Compacted filters [12] have also been proposed in order to optimize the size of the network by factorizing filters. For example, 3x3 convolution can be replaced by 1x1 convolution with SqueezeNet [37], achieving 50x fewer parameters than AlexNet. Other approaches focus on searching for the most appropriate architecture [48].

Computational efficiency is also an important research line in this scenario. Different architectures have been recently developed in this line optimising convolutional architectures. For example, EfficientNet [80] focuses on speeding up the neural network instead of scaling the network with more parameters, offering speedups ranging from 5 to 10 compared to competitive state-of-the-art models such as AmoebaNet, NASNet [87] or GPipe [34], and maintaining similar effectiveness while reducing the number of parameters by a factor of 4 to 7. Other methods focus on making lightweight powerful networks for mobile devices, assuming that larger networks cannot be handled as in other platforms. Two examples of this trend are ShuffleNet [86] and MobileNet [32], which focus on developing novel operators and architecture optimisation to create lighter models.

Information fusion [20,21] also represents an important avenue for the improvement of CNNs, which is commonly built in two distinct ways: ensemble of neural networks or appending an information fusion algorithm after the CNN representation features are extracted. An example of this would be a recent approach [70] which fuses the representation features of AlexNet [42] and VGG16 to increase performance over benchmark datasets. Alternatively, it is possible to split the image into several patches and process them separately with convolutions to later merge the representation features [3]. Another interesting application [14] has been proposed in a domain where various shallow networks coexist with a PCA analysis of inputs to build a combined feature vector for biomedical information. As mentioned before, ensembles can also be built by appending an information fusion stage after the features are extracted as proposed by [24]. This notion is again used by [55] in their research, where another SVM is appended after the convolutional features to make decisions. In [45] an ensemble of a set of deep learning models is designed to classify chest X-rays using small datasets for training, which is later combined with an explainable artificial intelligence (XAI) technique, based on combining the individual heatmaps obtained from each model in the ensemble. This approach, based on fusion and XAI methods, is used to improve both the overall performance classification of CNN and the interpretability of the CNNs, highlighting those areas of the image which are more relevant to generate the classification. Nevertheless, the use of fusion methods to improve performance has been demonstrated in plenty of domains [47].

Taking a step back, before CNNs were used, features were usually extracted by experts manually. During this process, the expert takes some input data (image pixels in computer vision) and transforms them into a manageable set of features [56,76]. Once the features are extracted, they can be processed through other machine learning algorithms. There are many challenges in this area [39] but, when they are properly addressed, it is possible to achieve high performance on high dimensional data. There is plenty of literature published around this issue. Originally, textural features were coined by Haralick et al. [27], who proposed a grey level co-occurrence matrix (GLCM) that contains the relative frequency of pixel pairs in an image. Further textural analysis has also been proposed, such as Local Binary Patterns (LBPs) [57], computing comparisons on neighbouring pixels across a circle and extracting the frequency histogram of these comparisons.

These techniques are paired with machine learning algorithms. Support Vector Machines (SVM) [13] are used in several relevant works, for example, in texture classification [40], on face biometrics [25], on-road car detection [84] or handwriting recognition [18], among others. K-Nearest Neighbors is also used to classify features, for example, in medical abnormality detection [67], plant leaf recognition [52] or MRI image classification [66]. Another frequent classification algorithm for image features are Random Forests (RF) [5], with relevant works at a variety of tasks such as radar image classification [16], leukhemia detection [51] or brain scan classification [26].

Other authors concentrate on the information that can be extracted with different levels of granularity. Ding et al. propose the use of a two-level classification [15]. In this case, making use of the attention mechanism and a pyramidal strategy to refine the classification with fine-grained information. In contrast, our target is to extract general patterns using two different strategies to extract complementary information. Chang et al. [8] also concentrate on fine-grained image classification using a mutual channel loss function that includes discriminality and diversity components. Our approach also leverages the information contained across channels, but using different statistical indicators such as average or deviation that can help to extract important patterns with maximum efficiency. Another fusion approach for image classification proposed in the literature consists in making use of the mid-level and high-level features extracted using a Bilinear CNN [63].

In this work, a technique for the efficient improvement of image classification tasks is proposed by fusing manual feature extraction and CNNs. In the next sections, we show that the predictions offered by models that rely on manually extracted features and those provided by CNNs entail complementary descriptive features, and therefore, that their combination will successfully improve both techniques. As manual features require lightweight machine learning algorithms to predict, the end performance of the ensemble will improve with a negligible increase in cost compared to the CNN computational requirements. To further improve CNNs performance, an ensemble is proposed consisting on the combination of manual feature extraction and CNNs by combining their respective end classification labels into a single label prediction [19].

3 Improving CNN classification with statistical indicators

With the rise of research focused on convolutional architectures, previous approaches using manually extracted features have been mostly abandoned. However, although less powerful, statistical indicators from images proved to be useful in combination with classical machine learning classifiers, reaching reasonable classification performance rates in certain domains. Besides, in contrast to the features extracted after different filters are applied in a convolutional architecture, these statistical indicators provide understandable information with minimal computational resources.

In this research, we leverage these statistical indicators to improve the performance of Convolutional Neural architectures. The core idea behind this method is to build an ensemble approach, combining a CNN model with a classical machine learning algorithm trained on the statistical information of the images, aiming to achieve improved CNN performance by adding light-weight models. In short, the method proposed combines the output probabilities of a CNN with softmax activation with the output probabilities of a classic machine learning algorithm trained on a series of statistical indicators representing different characteristics of the input images. This is meant to blend expert knowledge, providing a set of general-purpose statistical indicators and automatic feature detection obtained with a powerful CNN architecture. The end goal is to significantly improve the performance of CNN architectures with minimal additional investment.

3.1 Architecture overview

An overview of the solution is presented in Figure 1. Four modules are found in this system: the CNN, the statistical features extraction process, the statistical features-based classifier, and the fusion algorithm. The latter consists of a classical machine learning algorithm used to achieve the best combination between the output probabilities delivered by the CNN and the statistical features-based classifier.

In detail, to train the architecture proposed, the following steps are executed in accordance to Figure 1:

1. From the images, a pool of general-purpose features are extracted, transforming each image into a vector of representative features.
2. The image features extracted in step 1 are used to train the features-based model. Using a robust method is encouraged as some of the general-purpose features could be irrelevant for some datasets, however, most methods are able to improve classification performance.
3. In parallel to steps 1 and 2 and using the same images as before, a CNN is trained. Hyper-parameters and the optimal topology have to be adjusted as usual, in order to achieve high performance on a validation dataset.
4. The output probabilities obtained from step 2 and step 3 are concatenated for each processed image. All images from the training set are transformed into concatenated vectors of probabilities. If a machine learning algorithm does not support probabilities, the one-hot representation of the class is enough to represent the prediction.
5. The concatenated probabilities are used to train a fusion learning algorithm. It is worth noting that if optimal results are to be achieved, several models have to be tested in steps 3 and 5, but improving results upon the original CNN does not require such experimentation. Additionally, we also test a simple average between both vectors.
6. A vector is obtained with the final classification.

The following sections describe the methods tested to fulfill the CNN, Feature and Fusion algorithm modules of the architecture. Our motivation is to show that, in most cases, a CNN complemented with any of the proposed features and the fusion algorithm achieve better performance on classification tasks than the CNN alone.

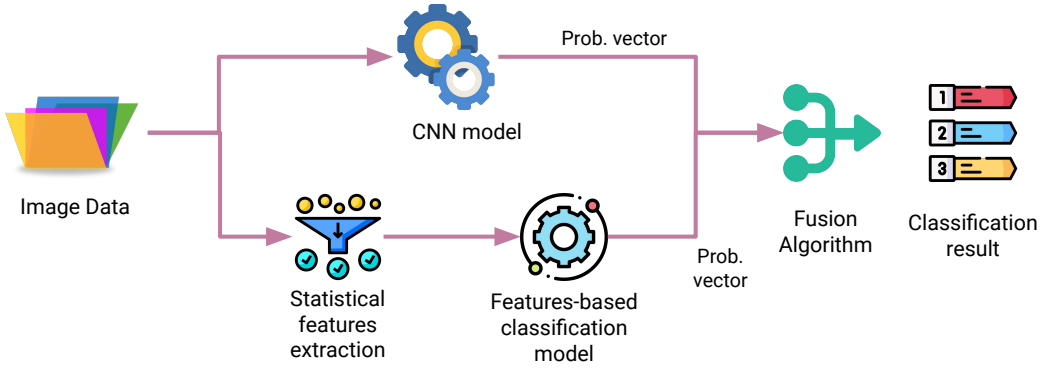


Fig. 1: Visual representation of the fusion approach proposed. The images are used to train a CNN architecture. Simultaneously, different statistical indicators are extracted from every image, and a classification model is trained on these features. Finally, an ensemble is built, training a new classification algorithm that combines the outputs of the CNN and the features-based classifier.

3.2 Image classification with CNNs

As shown in Figure 1, one of the classification methods that composes the architecture proposed is a CNN, which takes the raw image and automatically extracts features through different convolutional layers and infers knowledge in the last layers of the network. The fusion method proposed is independent of the CNN architecture, so it can be used with any convolutional network. Once the CNN is trained on a training set, the output probabilities of the model are combined with the output of a classifier trained on the statistical information through a machine learning classifier (or the average between probability vectors). As we will describe in Section 4, we tested the approach in five different state-of-the-art CNN architectures, including DenseNet-201, Inception-ResNet-V2, Inception-V3, VGG19 and Xception V1.

3.3 Extraction of statistical indicators

Manual extraction of features from images to perform computer vision tasks has remained a common approach in the state-of-the-art literature, although not as prolific as CNNs automatic feature extraction. On applied domains of application, simple statistical features are still relevant [30,64,81,82]. In this research, we propose a general-purpose statistical-based feature set for domain-independent computer vision tasks. Due to the specificity level of each of these features, their importance can vary across domains, which means that some of them may entail irrelevant knowledge in certain domains. Thus, a robust learning algorithm is used to process these statistical features. This process allows to take advantage from the information offered by these features while removing superfluous information that could hinder the learning capabilities of the approach. A set of commonly used machine learning algorithms was tested, including Random Forest (RF) [5], Support Vector Machines (SVM) [13], Linear Discriminant Analysis (LDA), among others.

Our features can be divided into two categories found in the literature about statistical features for computer vision tasks: *spectral features* and *textural features* [27, 38,62,85]. Spectral features represent colour as an statistic, for example, via average, deviation or differentials. On the other hand, textural features [27] represent an image in terms of edges and other abstract traits. This is achieved via a Grey Level Covariance Matrix (GLCM), a technique that detects the frequency of contiguous appearances of pixel values. Image textures are computed from the GLCM.

Both types of features are extracted from the raw image, including colour-based information using statistical summarization of each channel (average, deviation and so on) and from Grey Level Covariance Matrix (GLCM) for each colour. The GLCM detects the frequency of contiguous appearances of pixel values, allowing the detection of textures within the image which can reveal important patterns.

More specifically, the set of features extracted includes the average, standard deviation, skewness, average colour difference, histogram (with 5 bins), average colour ratio, textural average, variance, homogeneity, contrast, dissimilarity, entropy, second movement, and correlation. Each of these features is calculated for each colour channel, summing up to 60 features. These are all described in Table 1, where the n-th image on a particular channel c is named I_n^c with pixels $I^c(i,j)$ on position i and j with size NxM. The corresponding probability found in the GLCM for two contiguous grey

values a and b is $p^c(a, b)$ for Z grey levels measured. We define $p^c_x(a)$ as the a-th value of the sum of rows of the p^c matrix.

Once extracted, five different well-known classical machine learning classifiers are trained. The aim is to infer knowledge from these statistical indicators that can help to improve the performance of a CNN model. The ML algorithms, trained with the Scikit-learn [60], are the following:

- **K-Neighbours Classifier:** This method, in contrast to the previous ones, stores certain training instances which are later used to predict the output of a new instance. For that purpose, the label is calculated according to the neighbours of the new instance.
- **Linear Discriminant Analysis:** LDA is also a linear classification method which generates decision boundaries using class conditional densities and Bayes' rule.
- **Logistic Regression:** A linear classification model where the outputs are calculated according to a logistic curve and using the Broyden–Fletcher–Goldfarb–Shanno algorithm [23] as optimiser.
- **Random Forest:** A combination of decision tree classifiers trained on sub-samples of the data that averages the output of each tree to improve the accuracy.
- **Support Vector Machine:** This algorithm finds the hyperplane which better separates each class region from the other and maximising the separation of every data point. We have trained SVMs with two different kernels: sigmoidal and with a radial-basis function.

3.4 Fusion of CNN and statistical-based classification output probabilities

The result of training a CNN architecture in a given domain is a model where the last fully connected layer, for a certain input instance, delivers a vector defining a set of probabilities for each class. Normally, the maximum argument in this vector indicates the final label. However, classes could also reach similar probabilities, causing a decision without enough certainty. We argue that the inclusion of the statistical indicators previously described can help to achieve more accurate decisions in these cases, leading to an improvement of the final classification performance.

The final classification in our proposal is the result of the combination of two probability vectors: (a) the classification probabilities according to the CNN model, and (b) the output probabilities of a ML algorithm trained with the statistical indicators extracted. If the ML algorithm finally selected only provides a label instead of class probabilities, a one hot-encoding representation is used. After the modules for image and feature classification have been selected, a final module is required to combine the outputs of both sources. For that purpose, the probabilities of an image are calculated for both classifiers, the output vectors are concatenated and, finally, act as training examples for a fusion algorithm. For this final method, the same set of classifiers taking part of the statistical features-based classifier (described in Section 3.3) is used. The fusion algorithm outputs the end label from the original image.

4 Experimental setup

This section describes in detail the data, CNN architectures tested, the image preprocessing procedure and parameters used during the experimentation.

4.1 Datasets

The concern of this research is to present a powerful method for improving CNNs independent of the applied domain. Therefore, a wide set of 10 heterogeneous datasets has been selected to validate our method. These are summarised as follows:

a) *Caltech birds2011* [83]: This dataset contains images from mostly North American bird species in different poses and perspectives, usually at the centre of the image. This set contains 11.788 instances distributed in 200 classes.
b) *Cars 196* [41]: This dataset contains car images without context, where the vehicle is the focus with varying perspectives. There are 16.185 instances evenly distributed on 196 car types.
c) *Cassava* [53]: A collection of 9430 images of the cassava plant, including healthy and four types of disease, with high unbalance between classes.
d) *Cats vs dogs* [17]: This dataset is a binary classification problem of cat and dog images with context, in some instances sharing space with a human out of focus or with obstructions. This dataset contains 23.262 evenly distributed images among both classes.
e) *Citrus leaves* [68]: This dataset has a low number of images from healthy and unhealthy citrus fruit leaves. This dataset has 759 instances distributed in 4 classes.

Average	$\mu^c = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M} I^c(i,j)$
Standard Deviation	$\sigma^c = \sqrt{\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(I^c(i,j)-\mu^c)^2}$
Skewness	$\gamma^c = \frac{\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(I^c(i,j)-\mu^c)^3}{(\sigma^c)^3}$
Difference	$d^{c1,c2} = \mu^{c1} - \mu^{c2}$
Histogram	$h^c(B) = \sum_{i,j} 1$ for i,j subject to $b_{min} < I^c(i,j) < b_{max}$, where $B = \{b_{min}, b_{max}\}$ is a given histogram bin, and b_{min} and b_{max} are the minimum and maximum pixel values for a bin. (Only values within that bin are counted in the sum.)
Ratio	$r^{c1,c2} = \frac{\mu^{c1}}{\mu^{c2}}$
Textural Average	$f_1^c = \sum_{k=2}^{2Z} k\{\sum_{a=1}^{Z}\sum_{b=1}^{Z} p^c(a,b)\}; a+b=k$
Variance	$f_2^c = \sum_{a=1}^{Z}\sum_{b=1}^{Z}(a-p_\mu^c)p^c(a,b)$ where p_μ^c is the average of p^c
Homogeneity	$f_3^c = \sum_{a=1}^{Z}\sum_{b=1}^{Z}\frac{1}{1+(a-b)^2}p^c(a,b)$
Contrast	$f_4^c = \sum_{a=1}^{Z}\sum_{b=1}^{Z} p^c(a,b)(a-b)^2$
Dissimilarity	$f_5^c = \sum_{a=1}^{Z}\sum_{b=1}^{Z} p^c(a,b)\lvert a-b\rvert$
Entropy	$f_6^c = -\sum_{a=1}^{Z}\sum_{b=1}^{Z} p^c(a,b)\log(p^c(a,b))$
Second Moment	$f_7^c = \sum_{a=1}^{Z}\sum_{b=1}^{Z} p^c(a,b)^2$
Correlation	$f_8^c = \frac{\sum_{a=1}^{Z}\sum_{b=1}^{Z}(ij)p^c(a,b)-p_{x,\mu}^c p_{y,\mu}^c}{p_{x,\sigma}^c p_{y,\sigma}^c}$ where $p_{x,\mu}^c, p_{y,\mu}^c, p_{x,\sigma}^c, p_{y,\sigma}^c$ are the averages and standard deviations of p_x^c, p_y^c.

Table 1: List of features and formulas used for the proposed hand-crafted feature extraction.

f) *Deep weeds* [58]: This dataset contains weed images from the grasslands of Queensland, Australia, with the aim of telling apart weeds from grass and identifying the type of weed. It contains 17,509 instances evenly distributed across 9 classes. In the original article, Inception v3 and ResNet-50 are used with outstanding results.

g) *Malaria* [65]: This dataset is a repository of 27.558 segmented cell images, with an equal number of instances of parasitized cells and uninfected cells, with the aim of improving the diagnostic accuracy

	Version	No. labels	Train size	Valid. size	Test size	Ref.
Caltech birds 2011	0.1.1	200	5,395	599	5,794	[83]
Cars 196	2.0.0	196	7,330	814	8,041	[41]
Cassava	0.1.0	5	5,656	1,889	1,885	[53]
Cats vs dogs	4.0.0	2	16,283	2,327	4,652	[17]
Citrus leaves	0.1.2	4	416	59	119	[68]
Deep weeds	3.0.0	9	12,256	1,751	3,502	[58]
Malaria	1.0.0	2	19,291	2,755	5,512	[65]
Oxford flowers 102	2.1.1	102	1,020	1,020	6,149	[54]
Plant leaves	0.1.0	22	3,151	451	900	[11]
Plant village	1.0.2	38	38,012	5,430	10,861	[36]

Table 2: Datasets used in the experimentation, showing the version number, number of labels and the data splits size. See Figure 2 for example images.

of malaria cases. The original article achieves strong results with a ResNet-50.

h) *Oxford flowers102* [54]: This dataset has flower images of different species, focused on the flower with limited context. It contains 8.189 instances distributed across 102 classes, with 40 to 258 images per label.

i) *Plant leaves* [11]: This dataset contains plant images without the context of different leaf species and health conditions. This requires detection of both: plant type and plant health. There are 4.502 images of 22 classes total, evenly distributed.

j) *Plant village* [36]: This dataset contains a large number of leaf images, focused and without context. This dataset contains 54.303 instances of leaves, distributed evenly in 38 classes.

Table 2 shows a summary of all these datasets, including version, number of labels, and size of every split. The default partitions were kept. In those cases where only one (training) or two (training and test) splits were defined, a new partitioning was made to allocate 70% instances for training, 10% for validation and 20% for testing. Figure 2 shows example images for all datasets.

4.2 Image preprocessing

In order to modify every image as little as possible, only two preprocessing steps were followed before passing through the CNN architecture. First, since the datasets provide pictures with different sizes, every image was resized to a 224x224 format. Then, every pixel value was normalised from [0,255] to [0, 1].

4.3 CNN architectures

Five different architectures were tested in this research:

- *VGG19* [73]: The VGG16 and VGG19 architectures are composed by a feed-forward set of units, containing 2 to 4 convolutional layers, an activation and a pooling layer. Out of all architectures here, it is the most straightforward, with no additional forward connections or auxiliary outputs. This network has a very large number of parameters to adjust.
- *DenseNet* [33]: DenseNet, instead of relying on adding more units to its design, it strengthens the number of connections between layers. The main idea behind this design is to forward connect every unit. Each unit has again several convolutional layers, a batch normalisation layer (for regularisation), activation, and pooling.
- InceptionV3 [79]: The following architectures stem from the same family of architectures. This architecture factorises convolutions into simpler operations, for example, a 5x5 convolution into two 3x3 convolutions. These factorisations are separated in different pipelines inside a unit and several units are concatenated to achieve the end result.
- Xception [79]: Xception proposes a depth-wise separable convolution which is an extension of the factorisation proposed in InceptionV3. This convolutional operation involves two steps: first, an spatial convolution, transforming the channel, followed by a pointwise convolution, a 1x1 filter over the channels. Xception also incorporates feedforward residual connections, similar to a ResNet [28], connecting the last layer to the next one via a single convolution and addition operation.
- InceptionResNetV2 [78]: This architecture combines the ideas from InceptionV3 and ResNet. By stacking several Inception units and connecting them with residual feedforward addition operations, the capabilities of the network are enhanced.

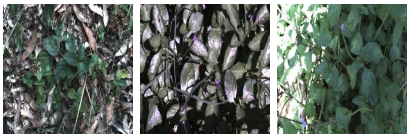
(a) Caltech birds2011

(b) Cars196

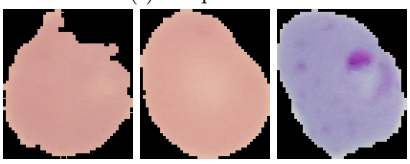
(c) Cassava

(d) Cats vs dogs

(e) Citrus leaves



(f) Deep weeds



(g) Malaria

(h) Oxford flowers102

(i) Plant leaves

(j) Plant village

Fig. 2: Samples images of the nine datasets used in the experiments.

All these architectures were initialised with transferred weights from the ImageNet domain, a percent of which are frozen and connected to a dense layer with 50% dropout. The optimisation algorithm chosen is Adam, while any other remaining hyper-parameter has been tuned empirically. The end result of the network is an output vector of probabilities from the last dense layer (with softmax activation). These probabilities have values between 0 and 1, summing 1 between them, and the argmax is selected as the true label for classification purposes.

4.4 CNN models execution

Every CNN model listed in Section 4.3 was executed five times with the Keras library for Python [10] with the TensorFlow [1] backend for each dataset, starting from pretrained Keras models. All CNN relevant hyper-parameters are summarized in Table 3. Then, all CNNs were fitted using the categorical cross-entropy as loss function, the Adam optimiser with gradient norm scaling if the vector exceeds 1.0, a learning rate of 0.0001, and fixing a maximum of 1,000 epochs. An early stopping criteria was also set to stop the training when the validation accuracy did not improve in the last 25 epochs more than a threshold $\theta = 0.0001$. Additionally, the training of every model was performed using a batch size of 64 examples. All datasets were streamlined by the TensorFlow dataset tools.

4.5 ML models hyperparameters

For the execution of the classification algorithms listed in Section 3.3, the scikit-learn [60] was used, using default parameters for all of them. In the case of Random Forest, 500 internal trees were employed. Due to the randomness of the Random Forest classifier when assigning samples to estimators and the selection of features during the definition of new branches in the trees, it was calculated the average of 10 different executions.

4.6 Execution environment & Github repository

All executions have been run on a machine with a 48GB Nvidia Quadro RTX 8000, an Intel(R) Xeon(R) Bronze 3206R CPU @ 1.90GHz and 256GB RAM.

The code developed for this article is publicly available and can be found at: `https://github.com/jahuerta92/cnn-prob-ensemble`.

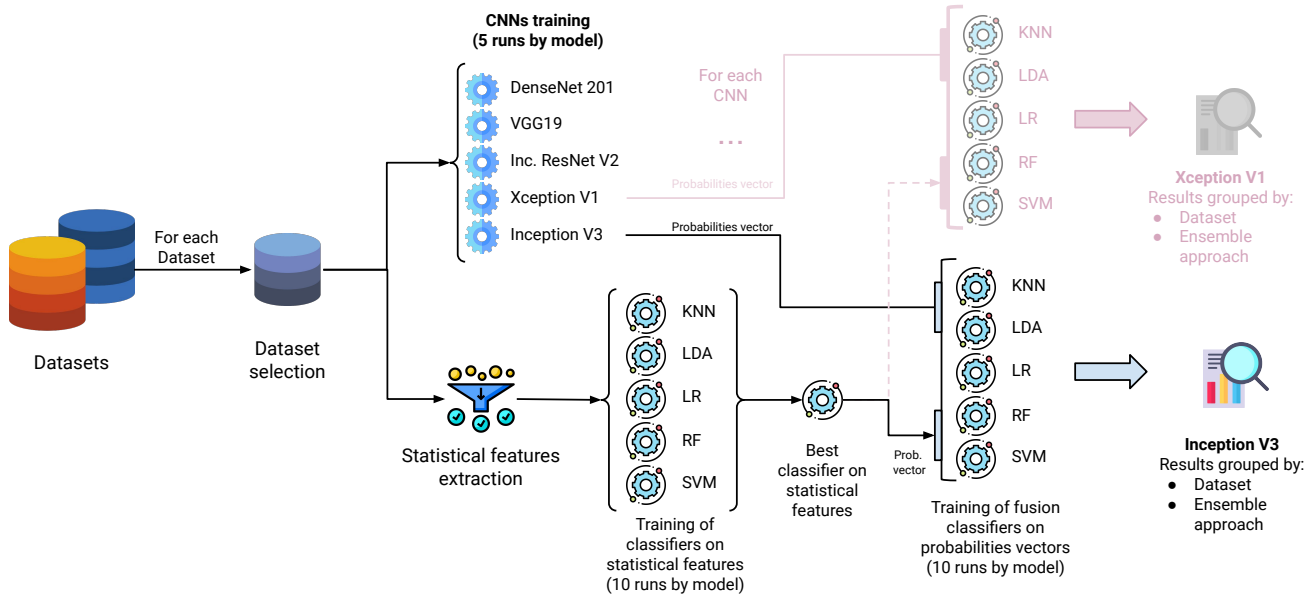


Fig. 3: This flowchart shows how the experiments are conducted. For each dataset considered, 5 different CNN architectures are trained and the set of statistical features is extracted. With this last set, different classical machine learning classifiers are trained, selecting the one which performed better. Then, for each CNN model trained, it is evaluated how it can be combined with statistical features based classification algorithm, using again different machine learning classification algorithms. Finally, the results show how, for each dataset and each specific CNN architecture, an ensemble can be built in order to improve the results of the CNN base architecture in isolation.

First layers frozen (%)	
VGG19	0%
InceptionResNetV2	25%
InceptionV3	10%
DenseNet201	25%
XceptionV1	25%
Adam optimizer	
α *Learning Rate*	1e-4
β_1 *Momentum 1*	0.9
β_2 *Momentum 2*	0.999
ϵ *Stability constant*	1e-7
Gradient normalization	1
Feed-forward classifier	
Global average pooling	
Neurons	2048
Dropout	50%
Activation	ReLU
Training methodology	
Maximum epochs	1000
Early stopping patience	25
θ *Early stopping threshold*	1e-4
Batch size	64
Image size (NxN)	N=224

Table 3: Summary of hyper-parameters relevant to CNN networks transfer learning, optimization, classification and training.

5 Experimentation

This section describes all experiments performed and the results obtained. First, the performance of the base five CNN architectures is evaluated, showing how each of them performs in each dataset. Then, the use of statistical features is assessed in all datasets using different machine learning classifiers to later evaluate the performance of the proposed fusion approach in comparison with the performance of the CNN architectures. Then, an ablation study helps to analyse the individual contribution of each statistical feature considered. Finally, a brief analysis of the time performance of the approach is provided.

In order to better describe how all the experiments were conducted, Figure 3 shows an overview of the different steps followed. The fusion approach was evaluated independently in 10 different datasets for 5 different CNN architectures. In case of the classification model based on statistical information, different machine learning methods with default hyperparameters were tested, selecting the one which performed better. Finally, for each dataset and architecture, different classifiers were again tested, with the goal of building the best possible combination between the probability vector obtained from the CNN and the vector obtained

CNN model	Measure	Dataset									
		Caltech birds 2011	**Cars196**	**Cassava**	**Cats vs dogs**	**Citrus leaves**	**Deep weeds**	**Malaria**	**Oxford flowers 102**	**Plant leaves**	**Plant village**
VGG19	**Max.**	0.0%	0.01%	78.8%	98.24%	96.15%	92.15%	96.59%	32.47%	94.49%	99.24%
	Mean	0.0%	0.01%	76.55%	97.93%	**92.81%**	90.73%	82.0%	16.03%	93.44%	99.18%
	Std.	0.0%	0.0%	1.96%	0.35%	2.96%	1.15%	31.99%	15.03%	1.03%	0.12%
DenseNet 201	**Max.**	73.08%	82.92%	82.52%	99.34%	94.28%	93.97%	97.33%	85.33%	96.83%	99.66%
	Mean	**72.55%**	**82.28%**	**81.71%**	**99.17%**	91.96%	**93.7%**	**97.16%**	**85.04%**	**96.14%**	**99.58%**
	Std.	0.55%	0.46%	0.92%	0.17%	1.67%	0.28%	0.15%	0.19%	0.49%	0.08%
Inception V3	**Max.**	61.78%	72.54%	76.68%	99.0%	89.05%	91.26%	97.23%	69.67%	95.24%	99.44%
	Mean	61.49%	68.39%	75.05%	98.78%	86.7%	90.06%	96.95%	68.6%	94.55%	99.32%
	Std.	0.36%	4.38%	1.24%	0.16%	2.6%	1.15%	0.2%	0.73%	0.58%	0.07%
Xception V1	**Max.**	62.78%	71.32%	76.3%	98.97%	91.65%	87.43%	97.08%	78.49%	95.17%	99.55%
	Mean	62.08%	67.97%	75.95%	98.82%	89.25%	87.13%	96.87%	77.82%	94.63%	99.41%
	Std.	0.59%	2.31%	0.43%	0.14%	1.78%	0.43%	0.22%	0.54%	0.38%	0.08%
Inception ResNet V2	**Max.**	68.27%	77.21%	78.4%	99.2%	92.49%	91.61%	97.28%	79.21%	96.37%	99.58%
	Mean	67.2%	75.26%	77.43%	99.04%	89.74%	90.6%	97.11%	77.94%	95.81%	99.44%
	Std.	0.79%	1.42%	0.7%	0.1%	1.76%	0.72%	0.22%	0.94%	0.42%	0.13%

Table 4: Averaged results of the five CNN architectures evaluated in 10 different datasets. All values indicate the macro average weighted precision. The maximum row refers to the best execution according to the precision obtained in the validation set.

from the best classifier trained on the statistical information.

5.1 CNN models performance

The first step is to train and evaluate several times each CNN architecture defined in Section 4.3. This serves as a baseline for the performance of these models to later quantify the improvement achieved by the addition of statistical manually extracted features. The results, in terms of macro average weighted precision, are shown in Table 4. For each dataset and each CNN architecture, the table shows the average and standard deviation of macro average weighted precision in the test set for each execution and the best result according to the validation set. DenseNet presents the best results among the different datasets used, reaching the best position in 9 of 10 datasets. One of the main characteristics of this model lies in that it has an important number of parameters (20,242,984) and a topological depth of 201 elements (for the DenseNet 201 version considered in this research). Despite not being the most complex architecture, DenseNet shows an excellent performance among datasets. Inception ResNet V2, although it is a more complex architecture, with 55,873,736 parameters a topology composed of 572 elements shows slightly worse results.

Although VGG-19 also produces the best result in one domain (Citrus leaves) and excellent results in other four domains (Cats vs dogs, Deep weeds, Plant leaves, and Plant Village), encountered severe difficulties in training an accurate model in 3 domains: Cars 196, Caltech birds2011, and Oxford flowers102. This effect coincides with the fact that these datasets have a large number of labels, more than 100 in the three cases, leading to conclude that VGG19 does not allow to retrieve enough discriminating information to distinguish between such a large number of labels. It is worth mentioning that VGG tends to erroneously classify all samples to the same label during the training phase. Additionally, this low precision could also be caused due to the weights of the pre-trained model or the necessity of modifying certain hyperparameters. To avoid any type of biased decision, we run all models with default parameters.

In almost all datasets there is room for improvement. However, in the case of Plant village and Cats vs dogs, the results surpassed 99% precision, so it is not expected to increase this value with the inclusion of statistical information. However, we decided to include these datasets in order to verify that the use of additional features and a second classification step does not lead to a counterproductive approach.

5.2 Image classification based on statistical indicators

The second step of the experimentation followed involves evaluating the capacity of the statistical features manually extracted to reveal important patterns that can help to differentiate between labels. This new representation, a vector of 54 values extracted from the image (as shown in Table 1), was tested using differ-

Classifier	Dataset									
	Caltech birds 2011	**Cars196**	**Cassava**	**Cats vs dogs**	**Citrus leaves**	**Deep weeds**	**Malaria**	**Oxford flowers 102**	**Plant leaves**	**Plant village**
Logistic Regression	2.38%	1.89%	56.16%	62.34%	88.74%	54.05%	84.95%	22.09%	66.05%	63.0%
LD Analysis	**3.44%**	**3.6%**	64.32%	**63.51%**	**89.41%**	61.86%	90.17%	**33.19%**	**81.98%**	76.27%
KNeighbors	2.58%	2.56%	55.94%	58.87%	86.7%	65.35%	78.42%	24.71%	72.5%	77.79%
SVM-rbf	2.74%	1.63%	**66.51%**	62.98%	87.57%	64.44%	89.1%	22.41%	70.27%	74.57%
SVM-sigmoid	0.07%	0.01%	35.79%	58.81%	42.32%	29.61%	49.62%	13.99%	6.83%	20.09%
Random Forest	3.1%	3.59%	60.86%	63.43%	85.62%	**68.72%**	**95.84%**	23.09%	76.62%	**79.86%**

Table 5: Results of different classifiers trained with the statistical features extracted from every image. All values indicate the macro average weighted precision.

ent machine learning classifiers, obtaining the results shown in Table 5. As in the case of the evaluation of the base CNN architecture, and given the large imbalance between classes, the macro average weighted precision was used.

As shown in the table, LDA (Linear Discriminant Analysis) and Random Forests are the stronger models to classify images according to statistical features. It is remarkable how the statistical features extracted retain a different degree of discriminating information depending on the domain. While on datasets such as Citrus leaves or Malaria, the result is very close to the one obtained with a CNN network, in case of Cars196 and Caltech birds2011, the classifiers reach low precision rates.

In Citrus leaves and Malaria, the solely use of these statistical indicators allows to reach close or more than 89% precision. This is a very remarkable result. The CNN architectures trained in these domains improved this figure only by 2% or 3%, but making use of extraordinary bigger computation and time resources. Moreover, the use of statistical indicators provide a useful instrument in specific domains, as we can know exactly what each feature means, instead of the complex and convoluted features extracted within the CNN architecture. The use of statistical features manually extracted together with a rule-base classifier, for instance, Random Forests, provides an explainable classification path [4], rather than a black box [59], as it is the case of a Convolutional Neural Network. In the next subsection, we evaluate if the combination of classification probabilities obtained from a CNN architecture and the help of a classification model based on statistical indicators leads to a stronger classification approach.

5.3 Fusion of CNN and statistical indicators-based classification

As the description of the results obtained with the base CNNs in Section 5.1 demonstrates, there are important differences in the performance for all architectures and domains tested, evidencing that there is room for improvement. The goal of this section is to demonstrate that the performance of CNNs for image classification can be enhanced by fusing its output with a classification model trained with statistical indicators manually extracted.

For this purpose, we evaluate each CNN model together with the classification algorithm which showed the best performance in each domain as shown in previous Section 5.2. The output vectors of these two sources are combined using a new supervised classifier. The results are displayed in Table 6. In this case, since both vectors represent the same type of information (the probability of each input image to be classified as one of the possible labels), we also consider the average between both probability vectors.

In all cases, a combination of CNN probabilities and statistical information exceeds the performance of the base CNN. A simple average between the output vector of the CNN and a classifier trained on statistical features is, despite its apparent simplicity, a competitive method that even shows the best results for most of the CNN architectures in three domains: Cassava, Citrus leaves and Malaria. This probability score fusion has been providing excellent results in many classifier combination problems [20] for long. In the rest of datasets, the combination between both vectors has to be conducted using a supervised classifier. Although each domain produces varied results and there is not a clear winner approach, values obtained with Random Forest are stable across datasets and CNN architectures.

The fusion approach produces, however, considerable differences between domains and CNN architectures. For instance, in the case of Citrus leaves, only the average helps to improve the results of the base CNN, while all the classifiers tested fail at improving the CNN performance, showing lower results. The same effect occurs in the Cars 196 dataset, but in this case Random Forest is the only approach capable of improv-

Dataset	CNN architecture	Base CNN (avg. weighted precision)	Ensemble (avg. weighted precision)						
			Avg.	KNN	LDA	LR	RF	SVM rbf	SVG sig
Caltech birds 2011	**VGG19**	0.0±0.0%	3.45%	3.17%	**4.15%**	3.09%	3.69±0.14%	3.38%	3.65%
	DenseNet 201	72.55±0.55%	70.09%	72.65%	74.31%	72.85%	**78.49±0.33%**	72.58%	72.89%
	Inception V3	61.49±0.36%	59.66%	61.49%	64.57%	61.72%	**68.42±0.62%**	61.94%	61.69%
	Xception V1	62.08±0.59%	60.78%	62.44%	63.99%	62.5%	**64.83±0.63%**	62.61%	62.47%
	Inc. ResNet V2	67.2±0.79%	65.95%	66.87%	67.21%	67.05%	**69.72±0.42%**	66.94%	66.93%
Cars 196	**VGG19**	0.01±0.0%	3.6%	3.39%	3.73%	2.9%	4.35±0.19%	4.52%	**4.54%**
	DenseNet 201	82.28±0.46%	81.02%	81.94%	81.72%	81.81%	**84.66±0.15%**	82.2%	81.96%
	Inception V3	68.39±4.38%	71.7%	72.65%	72.97%	72.53%	**75.94±0.24%**	73.0%	72.78%
	Xception V1	67.97±2.31%	70.49%	71.44%	71.35%	71.3%	**73.48±0.24%**	71.52%	71.34%
	Inc. ResNet V2	75.26±1.42%	74.46%	75.39%	73.9%	75.18%	**77.52±0.23%**	76.44%	75.3%
Cassava	**VGG19**	76.55±1.96%	83.93%	**84.12%**	83.96%	84.09%	84.0±0.09%	83.96%	84.03%
	DenseNet 201	81.71±0.92%	86.62%	86.85%	86.92%	86.99%	86.97±0.09%	86.76%	**87.06%**
	Inception V3	75.05±1.24%	**81.58%**	81.28%	81.38%	81.29%	81.41±0.08%	81.33%	81.56%
	Xception V1	75.95±0.43%	**81.18%**	80.66%	80.63%	80.61%	80.3±0.12%	80.43%	80.42%
	Inc. ResNet V2	77.43±0.7%	**82.83%**	82.44%	82.63%	82.56%	82.42±0.13%	82.46%	82.49%
Cats vs dogs	**VGG19**	97.93±0.35%	98.26%	98.28%	98.24%	98.24%	98.14±0.03%	**98.3%**	98.26%
	DenseNet 201	99.17±0.17%	99.29%	99.33%	99.33%	99.33%	**99.36±0.0%**	**99.36%**	99.33%
	Inception V3	98.78±0.16%	**98.84%**	98.8%	98.8%	98.8%	98.82±0.0%	98.82%	98.8%
	Xception V1	98.82±0.14%	98.88%	**98.97%**	**98.97%**	**98.97%**	**98.97±0.0%**	**98.97%**	**98.97%**
	Inc. ResNet V2	99.04±0.1%	98.97%	99.08%	98.97%	98.97%	**99.09±0.01%**	98.99%	98.95%
Citrus leaves	**VGG19**	92.81±2.96%	**95.34%**	92.61%	91.68%	92.61%	92.54±0.45%	91.68%	93.38%
	DenseNet 201	91.96±1.67%	**94.42%**	91.0%	91.0%	91.82%	91.0±0.0%	91.0%	91.0%
	Inception V3	86.7±2.6%	**92.73%**	88.66%	86.69%	89.33%	87.35±1.08%	88.18%	87.67%
	Xception V1	89.25±1.78%	**94.23%**	89.12%	87.09%	90.1%	87.98±0.5%	87.98%	88.35%
	Inc. ResNet V2	89.74±1.76%	**92.47%**	89.48%	89.48%	90.18%	89.48±0.59%	89.42%	90.17%
Deep weeds	**VGG19**	90.73±1.15%	91.41%	**91.43%**	91.13%	91.39%	84.21±1.31%	91.18%	91.07%
	DenseNet 201	93.7±0.28%	94.54%	**94.67%**	94.64%	94.59%	88.88±2.72%	94.61%	94.66%
	Inception V3	90.06±1.15%	90.96%	90.92%	**91.05%**	91.02%	85.38±1.19%	90.95%	90.9%
	Xception V1	87.13±0.43%	88.78%	88.73%	88.17%	88.83%	83.8±0.72%	**88.9%**	88.73%
	Inc. ResNet V2	90.6±0.72%	92.81%	91.55%	**92.89%**	92.57%	84.02±0.66%	91.14%	90.84%
Malaria	**VGG19**	82.0±31.99%	**96.65%**	96.14%	96.63%	96.52%	95.86±0.01%	96.25%	96.55%
	DenseNet 201	97.16±0.15%	97.39%	97.33%	97.33%	**97.41%**	97.32±0.0%	97.35%	97.39%
	Inception V3	96.95±0.2%	**97.1%**	96.11%	97.01%	96.79%	95.89±0.02%	96.28%	96.43%
	Xception V1	96.87±0.22%	**97.04%**	96.39%	96.96%	96.77%	95.86±0.02%	96.48%	96.32%
	Inc. ResNet V2	97.11±0.22%	**97.32%**	97.21%	97.23%	97.29%	96.22±0.43%	97.31%	97.21%
Oxford flowers 102	**VGG19**	16.03±15.03%	**43.44%**	42.97%	38.48%	40.95%	39.94±0.39%	40.78%	39.64%
	DenseNet 201	85.04±0.19%	67.51%	79.88%	90.59%	83.29%	**92.37±2.48%**	84.87%	86.41%
	Inception V3	68.6±0.73%	62.21%	71.2%	83.91%	73.21%	**86.14±1.83%**	74.68%	74.62%
	Xception V1	77.82±0.54%	68.18%	77.66%	88.87%	80.17%	**91.88±0.48%**	81.3%	81.67%
	Inc. ResNet V2	77.94±0.94%	74.54%	80.76%	86.71%	81.91%	**88.83±0.26%**	83.21%	82.9%
Plant leaves	**VGG19**	93.44±1.03%	94.01%	**95.15%**	94.82%	94.93%	94.67±0.09%	94.89%	94.73%
	DenseNet 201	96.14±0.49%	95.28%	96.39%	96.16%	**96.59%**	**96.59±0.05%**	96.26%	96.18%
	Inception V3	94.55±0.58%	94.56%	95.19%	94.58%	**95.22%**	94.61±0.06%	94.95%	94.92%
	Xception V1	94.63±0.38%	94.76%	**95.49%**	94.77%	95.46%	94.9±0.09%	95.37%	95.47%
	Inc. ResNet V2	95.81±0.42%	95.24%	95.85%	95.39%	**96.02%**	95.6±0.14%	95.71%	95.6%
Plant village	**VGG19**	99.18±0.12%	**99.46%**	99.43%	99.44%	99.43%	97.05±0.32%	99.43%	99.43%
	DenseNet 201	99.58±0.08%	**99.79%**	99.76%	99.75%	99.77%	97.28±0.2%	99.76%	99.76%
	Inception V3	99.32±0.07%	99.42%	99.39%	99.35%	**99.43%**	96.89±0.15%	99.42%	99.4%
	Xception V1	99.41±0.08%	**99.64%**	99.62%	99.6%	99.63%	97.12±0.29%	99.62%	99.62%
	Inc. ResNet V2	99.44±0.13%	99.57%	**99.59%**	99.4%	99.56%	97.1±0.23%	99.58%	99.56%

Table 6: Results obtained from the combination of the probabilities of each CNN model and the probabilities of the best classification algorithm trained with statistical features. Both vectors are combined using again classical supervised classification algorithms and the average between probability vectors. The classification approach with the best result for each convolutional architecture and dataset is highlighted in bold. All values indicate the macro average weighted precision in the test partition.

ing the performance for all CNNs except VGG19 (which does not infer knowledge in this dataset). In contrast, in domains such as Deep weeds or Cats vs dogs, several classifiers allow to exceed the base performance. All this indicates that, to to achieve the best possible ensemble, it is required to evaluate different methods. But, if the maximum efficiency is pursued, the average or Random Forest classifier will provide high precision rates.

On a more general level, although Random Forest shows high performance in a high number of combinations, Support Vector Machine leads to higher values in Deep weeds, Malaria and Plant village. This coincides with the three datasets with a larger number of samples. Thus, due to the high computational demand required to train SVMs, we suggest to use these models only in cases where a high number of examples exists, while Random Forest will perform better and will require less computational resources in other situations.

In case of the CNN architecture, all except VGG are improved similarly in all datasets. In the case of VGG, as it is shown in Section 5.1, it is not able to distinguish between labels in Caltech birds 2011 and Cars 196. In these cases, the increment of precision in the ensemble approach is due to the knowledge provided by the statistical features based classifier.

In comparison to the base CNN performance, in those datasets where the CNNs shows lower results, such as Caltech birds 2011, Cars 196, Cassava or Oxford flowers 102, the proposed approach clearly provides an improvement. For instance, in Caltech birds 2011, results improve by 6% when using DenseNet and 7% with Inception V3, while in Inception ResNet V2 and Xception V1 results increase by 2%, in all cases using the Random Forest classifier. In Oxford flowers, the difference is even bigger. The results obtained in this dataset with Inception ResNet V2 increase 11% and, in the case of Inception V3, the improvement reaches 18%.

All these results have shown that the inclusion of statistical features can help image classification approaches based on deep architectures. By extracting different estimators directly calculated from the pixels of every image, it is possible to extract useful additional information that leads to varied improvements depending on the domain. It should be mentioned that the large improvement observed in certain experiments, as it is the case of Oxford flowers102, where the precision is increased up to 18%, can be affected by the treatment of the output probabilities of the CNN architecture with a standard classifier, arising patterns that can improve the precision in specific domains. The computation of the statistical data and the training of classical classifiers provide a useful approach with a minimal computation load in comparison to the training of a CNN.

5.4 Ablation study

In the previous section, it has been demonstrated that the addition of statistical information leads to an improvement in the classification performance in all datasets and architectures. We now assess the individual performance of each statistical indicator considered, in order to quantify its contribution in the final classification and to check that all of them contribute in the same direction with no deterioration of the performance. For that purpose, this section describes the results after an ablation study addressed for each architecture and dataset.

Figs. 5, 8, 6, 7 and 4 show the results of the ablation study for each CNN architecture. The goal of these figures is to describe the effect of building the fusion approach without including one statistical feature at a time. Each cell shows the difference between the ensemble approach where one specific statistical feature is left out and the ensemble with all statistical features. All results were calculated after 10 different executions for both experiments. Negative values indicate a reduction of the macro average weighted precision as compared to the ensemble with all features.

In all cases, the algorithm for the ensemble approach is selected according to the results shown in Table 6. For the sake of clarity, statistical features have been grouped by type of feature for all channels, meaning that each statistical indicator shown in the figure refers to an ensemble approach without that feature for the three channels (i.e. dissimilarity relates to three features depending on the green, red and blue channels).

The results show high similarities across CNN architectures in the five figures. For instance, in the results for DenseNet 201 (Figure 5) for the first two datasets, Caltech birds 2011 and Cars 196, it can be seen small variations between features. While the elimination of specific features slightly increase the performance, in other cases it decreases. However, the differences in these cells (values lower than 0,4%) can be attributed to the use of the Random Forest classifier, the model which showed better results for all architectures except VGG19 in these two domains.

The differences shown in the ablation study fall under the standard deviation, which are of 0.33%, 0.42%, 0.62% and 0.63% for DenseNet 201, Inception ResNet V2, Inception V3 and Xception V1 respectively (see Table 6). In the case of the VGG19 in these two datasets, there are no differences with the result obtained including all features, a fact that can be appreciated for all datasets when using VGG19. This evidences that the different statistical features extracted present strong complementarities. Nevertheless, due to the low com-

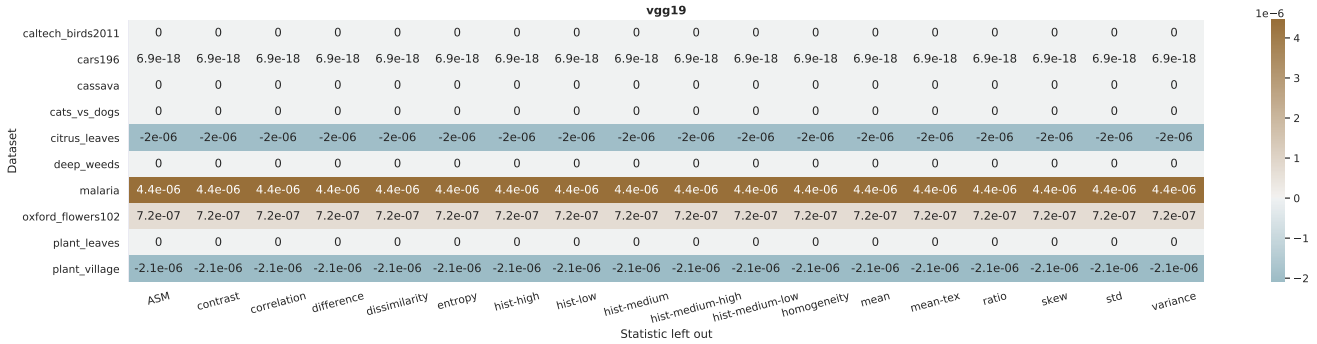


Fig. 4: Results of the ablation study for the VGG19 architecture.

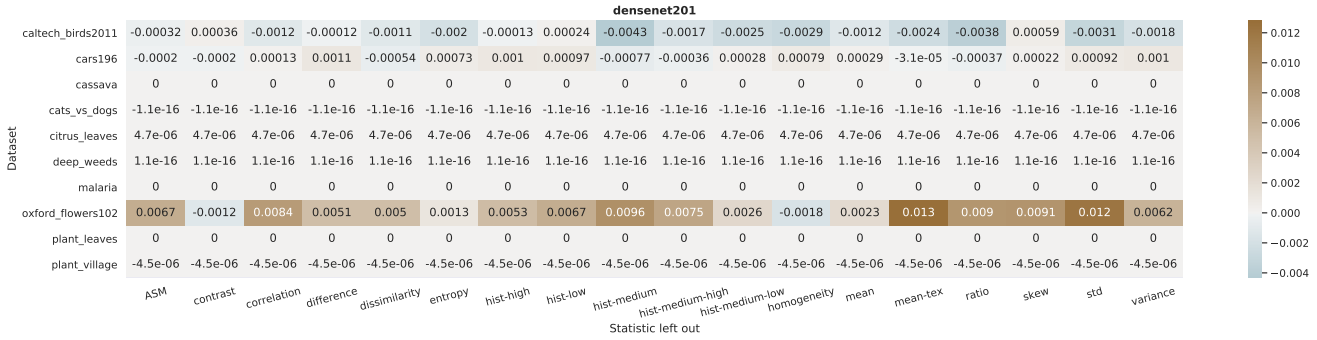


Fig. 5: Results of the ablation study for the DenseNet 201 architecture.

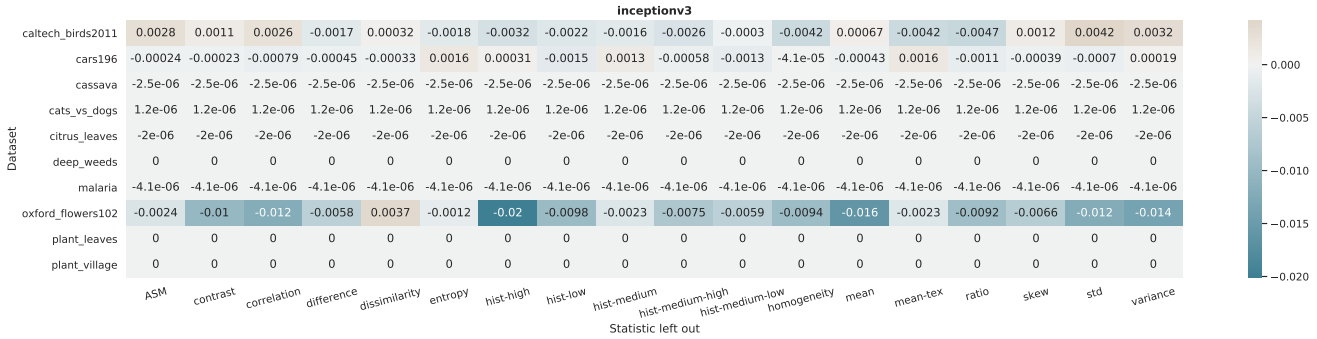


Fig. 6: Results of the ablation study for the InceptionV3 architecture.

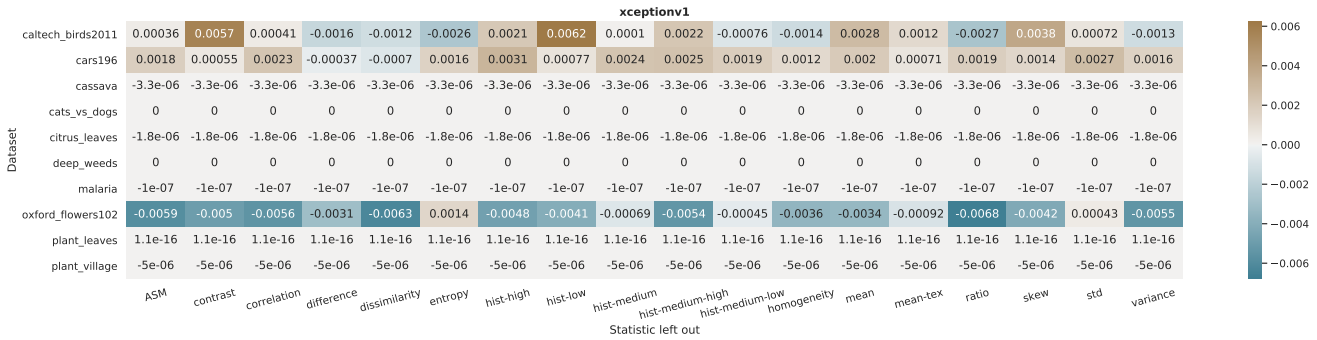


Fig. 7: Results of the ablation study for the Xception V1 architecture.

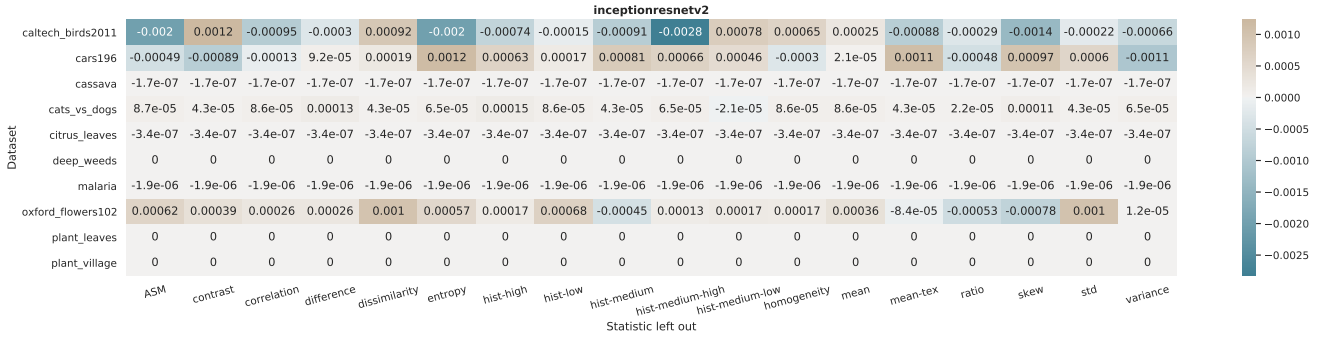


Fig. 8: Results of the ablation study for the Inception ResNet V2 architecture.

putational resources required to extract each of them, we do not encourage to use a subset of them as they can improve the performance in each specific scenario. For instance, in Oxford flowers 102 with Inception V3 (see Figure 6), the elimination of hist-high causes a 2% decrease in the performance, while contrast and correlation leads to a 1% decrease, evidencing the importance of these two features.

5.5 Time performance analysis

An analysis based on the time performance of the proposed approach is discussed in this section. Table 7, provided in Appendix A, shows all training times in minutes for the different parts of the method proposed, for all datasets, CNN architectures, and ensemble algorithms.

As expected, the CNN is the most time-consuming element of the approach, and it is highly dependent on the number of training examples but also on the general complexity of the domain. The feature extraction process, in contrast, is very efficient and, in most cases, it lasts less than one minute. In one specific case, for the Plant leaves dataset, due to the characteristics of this dataset, it takes 15 minutes. However, it is still a quarter of the training time of the CNN. For the statistical features-based classification, the time required is also minimal in comparison to the CNN. This is caused by the low size of the feature space, consisting of 54 different characteristics. In this case, the number of training examples will increase the training time accordingly.

The final ensemble approach is also very efficient. However, in this case, the use of Support Vector Machines causes large training times in several datasets (Caltech birds 2011, Cars 196 or Plant village). This is caused due to the number of labels but also due to the number of training examples. If maximum efficiency is pursued, we recommend to avoid this type of classifier, as those based on decision trees or Linear Discriminant Analysis will provide high precision rates more efficiently. In any case, the minimum training time is obtained using the average between both probability vectors, which, as previously showed, achieves high values in several datasets and architectures.

6 Conclusions and future works

In this article, a novel approach to improve CNNs has been proposed. This approach consists in combining traditional manually extracted features with the automatic feature extraction of Convolutional Neural Networks. This ensemble model contains a CNN module that labels examples with a probability vector, a statistical features based classifier that labels examples with either a probability vector or a one-hot encoded vector, and, finally, both vectors are combined and reclassified through a fusion approach to get the final labelling of the image. This method has been tested with several CNN algorithms on heterogeneous datasets to demonstrate the general capabilities of the method.

The experimentation shows promising results, as the proposed additions are never detrimental independently of the domain or CNN network applied. Although the improvement is subject to the performance of the CNN, which implies that in those domains where the CNN reaches high accuracy rates there will be little space for improvement, in all domains tested, the approach is able to improve the results. Besides, training the feature classifier and fusion classifier modules is much cheaper than training upscaled versions of the studied networks. As we show in the Appendix, the time required to train both classifiers is minimal in comparison to the training of the CNN. It is also worth noting that some feature classifiers (and fusion classifiers) have better explainability than the CNN [4,59], although these explainable models are not always the best option.

This proposal could be improved in several ways in future work. For example, manually extracted features could be further improved by using new approaches to feature extraction, including more advanced texture detection techniques, or hand-crafted feature extraction techniques specifically developed for the domain at hand.

Other venues for future research may include using boosting techniques in the fusion classifier such as extreme gradient boosting, among others. The fusion classifier can also be improved by first designing it using large-scale development data representing general probability behaviors in wide domains, and then fine-tuning it to specific problems or application areas in a kind of adapted fusion scheme [21,22].

An interesting research line could appear from the study and comparison of machine learning extracted features and manually extracted features as indicated by previous research [35]. Our results show that the predictions of CNNs and other algorithms are fundamentally different, and that their combination is greatly beneficial to performance. These differences could be the key to understand and explain why CNNs fail to classify some examples into specific domains. This example-based behavior that makes CNNs fail while other simpler classifiers can work much better can be exploited with context-based fusion classifiers [74], e.g., switching between them depending on the input images and context [2], or combining them with example-dependent adaptive fusion schemes [21]. Finally, we also plan to adapt the proposed ensemble approach to reduce undesired biases [61,72] in pre-trained networks.

Acknowledgements This work has been partially supported by the following grants and funding agencies: Spanish Ministry of Science and Innovation under TIN2017-85727-C4-3-P (DeepBio), FightDIS (PID2020-117263GB-100) and RTI2018-101248-B-I00 (BIBECA) grants, by Comunidad Autónoma de Madrid under S2018/TCS-4566 grant (CYNAMON), by BBVA Foundation grants for scientific research teams SARS-CoV-2 and COVID-19 under the grant: "*CIVIC: Intelligent characterisation of the veracity of the information related to COVID-19*" and by CHIST-ERA 2017 BDSI PACMEL (PCI2019-103623). Finally, the work has been supported by the Comunidad Autónoma de Madrid under: "Convenio Plurianual with the Universidad Politécnica de Madrid in the actuation line of *Programa de Excelencia para el Profesorado Universitario*".

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015). URL `http://tensorflow.org/`. Software available from tensorflow.org
2. Alonso-Fernandez, F., Fierrez, J., Ramos, D., Gonzalez-Rodriguez, J.: Quality-based conditional processing in multi-biometrics: application to sensor interoperability. IEEE Trans. on Systems, Man and Cybernetics Part A **40**(6), 1168–1179 (2010)
3. Amin-Naji, M., Aghagolzadeh, A., Ezoji, M.: Ensemble of CNN for multi-focus image fusion. Information Fusion **51**, 201–214 (2019). DOI 10.1016/j.inffus.2019.02.003
4. Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F.: Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information Fusion **58**, 82 – 115 (2020)
5. Breiman, L.: Random Forests. Machine Learning **45**(1), 5–32 (2001)
6. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language Models are Few-Shot Learners. arXiv (2020). URL `https://arxiv.org/abs/2005.14165v4`
7. Bucilua, C., Caruana, R., Niculescu-Mizil, A.: Model compression. Association for Computing Machinery, New York, NY, USA (2006). DOI 10.1145/1150402.1150464
8. Chang, D., Ding, Y., Xie, J., Bhunia, A.K., Li, X., Ma, Z., Wu, M., Guo, J., Song, Y.Z.: The devil is in the channels: Mutual-channel loss for fine-grained image classification. IEEE Transactions on Image Processing **29**, 4683–4695 (2020)
9. Cheng, Y., Wang, D., Zhou, P., Zhang, T.: A Survey of Model Compression and Acceleration for Deep Neural Networks. arXiv (2017). URL `https://arxiv.org/abs/1710.09282v9`
10. Chollet, F., et al.: Keras. `https://keras.io` (2015)
11. Chouhan, S.S., Kaul, A., Singh, U.P., Jain, S.: A Database of Leaf Images: Practice towards Plant Conservation with Plant Pathology. mendeley **1** (2019). DOI 10.17632/hb74ynkjcn.1
12. Cohen, T.S., Welling, M.: Group Equivariant Convolutional Networks. arXiv (2016). URL `https://arxiv.org/abs/1602.07576v3`
13. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3), 273–297 (1995)
14. Deng, P., Chen, H., Huang, M., Ruan, X., Xu, L.: An ensemble CNN method for biomedical entity normalization. ACL Anthology pp. 143–149 (2019). DOI 10.18653/v1/D19-5721
15. Ding, Y., Ma, Z., Wen, S., Xie, J., Chang, D., Si, Z., Wu, M., Ling, H.: Ap-cnn: weakly supervised attention pyramid convolutional neural network for fine-grained visual classification. IEEE Transactions on Image Processing **30**, 2826–2836 (2021)
16. Du, P., Samat, A., Waske, B., Liu, S., Li, Z.: Random Forest and Rotation Forest for fully polarized SAR image classification using polarimetric and spatial features. ISPRS J. Photogramm. Remote Sens. **105**, 38–53 (2015). DOI 10.1016/j.isprsjprs.2015.03.002

17. Elson, J., Douceur, J.J., Howell, J., Saul, J.: Asirra: A captcha that exploits interest-aligned manual image categorization. In: Proceedings of 14th ACM Conference on Computer and Communications Security (CCS). Association for Computing Machinery, Inc. (2007)
18. Faundez-Zanuy, M., Fierrez, J., Ferrer, M.A., Diaz, M., Tolosana, R., Plamondon, R.: Handwriting biometrics: Applications and future trends in e-security and e-health. Cognitive Computation (2020)
19. Fierrez, J.: Adapted fusion schemes for multimodal biometric authentication. Ph.D. thesis, Universidad Politecnica de Madrid (2006)
20. Fierrez, J., Morales, A., Vera-Rodriguez, R., Camacho, D.: Multiple classifiers in biometrics. part 1: Fundamentals and review. Information Fusion **44**, 57–64 (2018)
21. Fierrez, J., Morales, A., Vera-Rodriguez, R., Camacho, D.: Multiple classifiers in biometrics. part 2: Trends and challenges. Information Fusion **44**, 103–112 (2018)
22. Fierrez-Aguilar, J., Garcia-Romero, D., Ortega-Garcia, J., Gonzalez-Rodriguez, J.: Adapted user-dependent multimodal biometric authentication exploiting general information. Pattern Recognition Letters **26**(16), 2628–2639 (2005)
23. Fletcher, R.: Practical methods of optimization. John Wiley & Sons (2013)
24. Fu Jie Huang, LeCun, Y.: Large-scale learning with svm and convolutional for generic object categorization. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 1, pp. 284–291 (2006). DOI 10.1109/CVPR.2006.164
25. Gonzalez-Sosa, E., Fierrez, J., Vera-Rodriguez, R., Alonso-Fernandez, F.: Facial soft biometrics for recognition in the wild: Recent works, annotation and cots evaluation. IEEE Trans. on Information Forensics and Security **13**(8), 2001–2014 (2018)
26. Gray, K.R., Aljabar, P., Heckemann, R.A., Hammers, A., Rueckert, D.: Random forest-based similarity measures for multi-modal classification of Alzheimer's disease. Neuroimage **65**, 167–175 (2013). DOI 10.1016/j.neuroimage.2012.09.065
27. Haralick, R.M., Dinstein, I., Shanmugam, K.: Textural Features for Image Classification. IEEE Transactions on Systems, Man and Cybernetics **SMC-3**(6), 610–621 (1973)
28. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. arXiv (2015). URL `https://arxiv.org/abs/1512.03385v1`
29. He, Y., Zhang, X., Sun, J.: Channel Pruning for Accelerating Very Deep Neural Networks. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 1389–1397 (2017). [Online; accessed 20. Oct. 2020]
30. Heinle, A., Macke, A., Srivastav, A.: Automatic cloud classification of whole sky images. Atmospheric Measurement Techniques **3**(3), 557–567 (2010)
31. Hinton, G., Vinyals, O., Dean, J.: Distilling the Knowledge in a Neural Network. arXiv (2015). URL `https://arxiv.org/abs/1503.02531v1`
32. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv (2017). URL `https://arxiv.org/abs/1704.04861`
33. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 2261–2269 (2017). DOI 10.1109/CVPR.2017.243
34. Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, M.X., Chen, D., Lee, H., Ngiam, J., Le, Q.V., Wu, Y., Chen, Z.: GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism. arXiv (2018). URL `https://arxiv.org/abs/1811.06965`
35. Huertas-Tato, J., Martín, A., Camacho, D.: Cloud Type Identification Using Data Fusion and Ensemble Learning, *Lecture Notes in Computer Science*, vol. 12490, p. 137–147. Springer International Publishing (2020)
36. Hughes, D.P., Salath'e , M.: An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing. CoRR **abs/1511.08060** (2015). URL `http://arxiv.org/abs/1511.08060`
37. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. arXiv (2016). URL `https://arxiv.org/abs/1602.07360v4`
38. Jensen, J.R.: Spectral and textural features to classify elusive land cover at the urban fringe. The Professional Geographer **31**(4), 400–409 (1979)
39. Khalid, S., Khalil, T., Nasreen, S.: A survey of feature selection and feature extraction techniques in machine learning. 2014 Science and Information Conference pp. 372–378 (2014). DOI 10.1109/SAI.2014.6918213
40. Kim, K.I., Jung, K., Park, S.H., Kim, H.J.: Support vector machines for texture classification. IEEE Trans. Pattern Anal. Mach. Intell. **24**(11), 1542–1550 (2002). DOI 10.1109/TPAMI.2002.1046177
41. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13). Sydney, Australia (2013)
42. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Communications of the ACM **60**(6), 84–90 (2017)
43. LeCun, Y., Bengio, Y., et al.: Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks **3361**(10), 1995 (1995)
44. Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., Alsaadi, F.E.: A survey of deep neural network architectures and their applications. Neurocomputing **234**, 11–26 (2017)
45. Liz, H., Sánchez-Montañés, M., Tagarro, A., Domínguez-Rodríguez, S., Dagan, R., Camacho, D.: Ensembles of convolutional neural network models for pediatric pneumonia diagnosis. Future Generation Computer Systems **122**, 220–233 (2021)
46. Luo, J.H., Wu, J., Lin, W.: Thinet: A filter level pruning method for deep neural network compression. In: Proceedings of the IEEE international conference on computer vision, pp. 5058–5066 (2017)
47. Martín, A., Lara-Cabrera, R., Camacho, D.: Android malware detection through hybrid features fusion and ensemble classifiers: The andropytool framework and the omnidroid dataset. Information Fusion **52**, 128–142 (2019)
48. Martín, A., Lara-Cabrera, R., Fuentes-Hurtado, F., Naranjo, V., Camacho, D.: Evodeep: a new evolutionary approach for automatic deep neural networks parametrisation. Journal of Parallel and Distributed Computing **117**, 180–191 (2018)
49. Martin, A., Lara-Cabrera, R., Vargas, V.M., Gutiérrez, P.A., Hervás-Martínez, C., Camacho, D.: Statistically-driven coral reef metaheuristic for automatic hyperparameter setting and architecture design of convolutional neural networks. In: 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. IEEE (2020)

50. Martín, A., Vargas, V.M., Gutiérrez, P.A., Camacho, D., Hervás-Martínez, C.: Optimising convolutional neural networks using a hybrid statistically-driven coral reef optimisation algorithm. Applied Soft Computing **90**, 106144 (2020)
51. Mishra, S., Majhi, B., Sa, P.K., Sharma, L.: Gray level co-occurrence matrix and random forest based acute lymphoblastic leukemia detection. Biomed. Signal Process. Control **33**, 272–280 (2017). DOI 10.1016/j.bspc.2016.11.021
52. Munisami, T., Ramsurn, M., Kishnah, S., Pudaruth, S.: Plant Leaf Recognition Using Shape Features and Colour Histogram with K-nearest Neighbour Classifiers. Procedia Comput. Sci. **58**, 740–747 (2015). DOI 10.1016/j.procs.2015.08.095
53. Mwebaze, E., Gebru, T., Frome, A., Nsumba, S., Tusubira, J.: icassava 2019fine-grained visual categorization challenge (2019)
54. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (2008)
55. Niu, X.X., Suen, C.Y.: A novel hybrid CNN–SVM classifier for recognizing handwritten digits. Pattern Recognit. **45**(4), 1318–1325 (2012). DOI 10.1016/j.patcog.2011.09.021
56. Nixon, M., Aguado, A.S.: Feature Extraction & Image Processing for Computer Vision, Third Edition. Academic Press, Inc.6277 Sea Harbor Drive Orlando, FLUnited States (2012). DOI 10.5555/2385460
57. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 971–987 (2002). DOI 10.1109/TPAMI.2002.1017623
58. Olsen, A., Konovalov, D.A., Philippa, B., Ridd, P., Wood, J.C., Johns, J., Banks, W., Girgenti, B., Kenny, O., Whinney, J., Calvert, B., Rahimi Azghadi, M., White, R.D.: DeepWeeds: A Multiclass Weed Species Image Dataset for Deep Learning. Scientific Reports **9**(2058) (2019). DOI 10.1038/s41598-018-38343-3
59. Ortega, A., Fierrez, J., Morales, A., Wang, Z., Ribeiro, T.: Symbolic ai for xai: Evaluating lfit inductive programming for fair and explainable automatic recruitment. In: IEEE/CVF Winter Conf. on Applications of Computer Vision Workshops (WACVw) (2021)
60. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
61. Pena, A., Serna, I., Morales, A., Fierrez, J.: Bias in multimodal ai: Testbed for fair automatic recruitment. In: IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops (CVPRw) (2020). Also at ICML 2020 Workshop on Human-in-the-Loop Learning
62. Peña, J.M., Gutiérrez, P.A., Hervás-Martínez, C., Six, J., Plant, R.E., López-Granados, F.: Object-based image classification of summer crops with machine learning methods. Remote Sensing **6**(6), 5019–5041 (2014)
63. Peng, Y., Liao, M., Song, Y., Liu, Z., He, H., Deng, H., Wang, Y.: Fb-cnn: Feature fusion-based bilinear cnn for classification of fruit fly image. IEEE Access **8**, 3987–3995 (2019)
64. Pu, H., Sun, D.W., Ma, J., Cheng, J.H.: Classification of fresh and frozen-thawed pork muscles using visible and near infrared hyperspectral imaging and textural analysis. Meat Science **99**, 81–88 (2015)
65. Rajaraman, S., Antani, S.K., Poostchi, M., Silamut, K., Hossain, M.A., Maude, R.J., Jaeger, S., Thoma, G.R.: Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images. PeerJ **6**, e4568 (2018)
66. Rajini, N.H., Bhavani, R.: Classification of mri brain images using k-nearest neighbor and artificial neural network. In: 2011 International Conference on Recent Trends in Information Technology (ICRTIT), pp. 563–568 (2011). DOI 10.1109/ICRTIT.2011.5972341
67. Ramteke, R., Monali, K.Y.: Automatic medical image classification and abnormality detection using k-nearest neighbour. International Journal of Advanced Computer Research **2**(4), 190 (2012)
68. Rauf, H.T., Saleem, B.A., Lali, M.I.U., Khan, M.A., Sharif, M., Bukhari, S.A.C.: A citrus fruits and leaves dataset for detection and classification of citrus diseases through machine learning. Data in brief **26**, 104340 (2019)
69. Real, E., Aggarwal, A., Huang, Y., Le, Q.V.: Regularized Evolution for Image Classifier Architecture Search. arXiv (2018). URL `https://arxiv.org/abs/1802.01548`
70. Roy, S.K., Dubey, S.R., Chanda, B., Chaudhuri, B.B., Ghosh, D.K.: TexFusionNet: An Ensemble of Deep CNN Feature for Texture Classification. In: Proceedings of 3rd International Conference on Computer Vision and Image Processing, pp. 271–283. Springer, Singapore (2019). DOI 10.1007/978-981-32-9291-8_22
71. Schwartz, R., Dodge, J., Smith, N.A., Etzioni, O.: Green AI. arXiv (2019). URL `https://arxiv.org/abs/1907.10597v3`
72. Serna, I., Pena, A., Morales, A., Fierrez, J.: Insidebias: Measuring bias in deep networks and application to face gender biometrics. In: IAPR Intl. Conf. on Pattern Recognition (ICPR) (2021)
73. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv (2014). URL `https://arxiv.org/abs/1409.1556v6`
74. Snidaro, L., García, J., Llinas, J.: Context-based information fusion: A survey and discussion. Information Fusion **25**, 16 – 31 (2015)
75. Steinkraus, D., Buck, I., Simard, P.Y.: Using gpus for machine learning algorithms. In: Eighth International Conference on Document Analysis and Recognition (ICDAR'05), pp. 1115–1120 Vol. 2 (2005). DOI 10.1109/ICDAR.2005.251
76. Storcheus, D., Rostamizadeh, A., Kumar, S.: A survey of modern questions and challenges in feature extraction. In: Feature Extraction: Modern Questions and Challenges, pp. 1–18 (2015)
77. Strubell, E., Ganesh, A., McCallum, A.: Energy and Policy Considerations for Modern Deep Learning Research. Proceedings of the AAAI Conference on Artificial Intelligence **34**(09), 13693–13696 (2020)
78. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv (2016). URL `https://arxiv.org/abs/1602.07261v2`
79. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2818–2826 (2016). DOI 10.1109/CVPR.2016.308. URL `https://arxiv.org/abs/1512.00567`
80. Tan, M., Le, Q.V.: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv (2019). URL `https://arxiv.org/abs/1905.11946`

81. Tuominen, S., Pekkarinen, A.: Performance of different spectral and textural aerial photograph features in multi-source forest inventory. Remote sensing of Environment **94**(2), 256–268 (2005)
82. Venkataraman, D., Mangayarkarasi, N.: Computer vision based feature extraction of leaves for identification of medicinal values of plants. In: 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), p. 1–5. IEEE (2016)
83. Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., Perona, P.: Caltech-UCSD Birds 200. Tech. Rep. CNS-TR-2010-001, California Institute of Technology (2010)
84. Zehang Sun, Bebis, G., Miller, R.: On-road vehicle detection using gabor filters and support vector machines. In: 2002 14th International Conference on Digital Signal Processing Proceedings. DSP 2002 (Cat. No.02TH8628), vol. 2, pp. 1019–1022 vol.2 (2002). DOI 10.1109/ICDSP.2002.1028263
85. Zhang, H., Li, Q., Liu, J., Shang, J., Du, X., McNairn, H., Champagne, C., Dong, T., Liu, M.: Image classification using rapideye data: Integration of spectral and textual features in a random forest classifier. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **10**(12), 5334–5349 (2017)
86. Zhang, X., Zhou, X., Lin, M., Sun, J.: ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices (2018). URL `https://arxiv.org/abs/1707.01083`
87. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning Transferable Architectures for Scalable Image Recognition. arXiv (2017). URL `https://arxiv.org/abs/1707.07012`

A Time performance of the proposed fusion approach

Dataset	CNN	Base CNN training time (m)	Feature extraction time (m)	Statistical features based classifier training time (m)						Ensemble training time (m)						
				KNN	LDA	LR	RF	SVM rbf	SVM sig	Avg.	KNN	LDA	LR	RF	SVM rbf	SVM sig
Caltech birds 2011	DenseNet201	66,00	0,77	2,45	2,51	2,49	2,47	2,48	2,46	2,02E-04	5,40	0,96	3,25	18,00	106,68	148,98
	Inc. ResNet V2	52,60														
	Inc. V3	43,20														
	VGG19	20,25														
	Xception V1	35,40														
Cars 196	DenseNet201	116,40	1,73	4,65	4,60	4,62	4,60	4,62	4,67	3,66E-04	9,67	1,09	4,36	6,56	151,83	114,61
	Inc. ResNet V2	93,00														
	Inc. V3	68,60														
	VGG19	21,25														
	Xception V1	77,40														
Cassava	DenseNet201	81,20	0,81	0,86	0,86	0,87	0,87	0,86	0,88	8,02E-06	1,89	0,09	0,14	0,60	0,40	0,54
	Inc. ResNet V2	55,00														
	Inc. V3	45,00														
	VGG19	45,00														
	Xception V1	35,80														
Cats vs dogs	DenseNet201	72,40	1,43	5,67	5,66	5,64	5,57	5,66	5,63	1,89E-05	1,51	0,06	0,07	0,66	0,26	0,49
	Inc. ResNet V2	68,80														
	Inc. V3	71,40														
	VGG19	92,00														
	Xception V1	67,40														
Citrus leaves	DenseNet201	40,20	0,06	0,02	0,02	0,02	0,02	0,02	0,02	1,47E-06	0,11	0,02	0,03	0,59	0,02	0,03
	Inc. ResNet V2	17,00														
	Inc. V3	19,60														
	VGG19	7,50														
	Xception V1	15,20														
Deep weeds	DenseNet201	118,80	0,97	3,26	3,27	3,26	3,27	3,27	3,24	2,54E-05	12,49	0,17	0,31	0,63	1,04	1,17
	Inc. ResNet V2	127,40														
	Inc. V3	83,60														
	VGG19	93,75														
	Xception V1	54,40														
Malaria	DenseNet201	95,20	1,27	6,22	6,11	6,25	6,26	6,20	6,11	1,93E-05	1,80	0,10	0,07	0,70	0,28	0,65
	Inc. ResNet V2	104,80														
	Inc. V3	89,40														
	VGG19	68,50														
	Xception V1	74,60														
Oxford flowers 102	DenseNet201	45,40	0,72	0,17	0,17	0,17	0,17	0,17	0,17	6,54E-05	1,21	0,29	0,53	1,20	9,69	10,08
	Inc. ResNet V2	24,40														
	Inc. V3	25,00														
	VGG19	18,50														
	Xception V1	23,60														
Plant leaves	DenseNet201	133,20	15,09	0,32	0,32	0,32	0,32	0,32	0,32	1,63E-05	1,22	0,10	0,17	0,61	1,12	1,28
	Inc. ResNet V2	68,20														
	Inc. V3	75,60														
	VGG19	65,00														
	Xception V1	58,60														
Plant village	DenseNet201	232,80	2,75	37,84	37,86	38,92	38,17	37,72	37,94	2,04E-04	130,35	0,77	3,12	3,02	36,56	44,47
	Inc. ResNet V2	293,20														
	Inc. V3	212,80														
	VGG19	301,25														
	Xception V1	228,40														

Table 7: Duration in minutes of the different parts of the approach proposed for the different datasets, CNN architectures, classifiers used for building the statistical features based classifier and algorithms used for the final ensemble.