Undefined 1 (2009) 1–5
IOS Press

Content based image retrieval by ensembles of deep learning object classifiers

Safa Hamreras [a], Bachir Boucheham [a], Miguel A. Molina-Cabello [b], Rafaela Benítez-Rochel [b] and Ezequiel López-Rubio [b]

[a] *LRES (Laboratoire de Recherche en Électronique de Skikda) and Department of Computer Science, University of Skikda 20 Août 1955, Algeria*
E-mail: safahamreras@gmail.com; bachir.boucheham@hotmail.com
[b] *Department of Computer Languages and Computer Science. University of Málaga.*
Bulevar Louis Pasteur, 35. 29071 Málaga, Spain.
Biomedic Research Institute of Málaga (IBIMA)
C/ Doctor Miguel Díaz Recio, 28, 29010, Málaga, Spain
E-mail: {miguelangel,benitez,ezeqlr}@lcc.uma.es

Abstract. Ensemble learning has demonstrated its efficiency in many computer vision tasks. In this paper, we address this paradigm within content based image retrieval (CBIR). We propose to build an ensemble of convolutional neural networks (CNNs), either by training the CNNs on different bags of images, or by using CNNs trained on the same dataset, but having different architectures. Each network is used to extract the class probability vectors from images to use them as representations. The final image representation is then generated by combining the extracted class probability vectors from the built ensemble. We show that the use of CNN ensembles is very efficient in generating a powerful image representation compared to individual CNNs. Moreover, we propose an Averarge Query Expansion technique for our proposal to enhance the retrieval results. Several experiments were conducted to extensively evaluate the application of ensemble learning in CBIR. Results in terms of precision, recall, and mean average precision show the outperformance of our proposal compared to the state of the art.

Keywords: Content based image retrieval, Ensemble learning, convolutional neural networks

1. Introduction

Content Based Image Retrieval (CBIR) is the procedure of automatically identifying images by the extraction of their low-level visual features like color, texture, shape properties or any other features being derived from the image itself [1]. The well-known 'semantic-gap' issue that exists between low-level features of images and high-level semantic concepts perceived by humans has been addressed through a variety of techniques [2,3,4,5,6].

However, the huge diversity of semantical concepts contained in images suggest that many robust discrimination and learning techniques are needed. In that respect, deep learning has become a significant step forward in the already developing fields of computer vision. It is a technique which includes a family of machine learning algorithms that attempt to model high-level abstraction in data by employing deep architectures composed of multiple non-linear transformations [7,8]. These models are originated from computational models inspired by the structure and functions of the brain called artificial neural networks. The main reasons behind its success are the availability of large annotated datasets and the computational power and affordability of GPUs. A typical example of deep architectures are feed-forward neural networks with many hidden layers, and backpropagation [9] as a learning algorithm. Recent successes of deep learning techniques, especially Convolutional Neural Networks (CNNs) [10] in solving computer vision and image understanding tasks [11,12,13,14,15] have inspired us to

0000-0000/09/$00.00 © 2009 – IOS Press and the authors. All rights reserved

explore this approach with aim to end up with more robust and better performing CBIR systems.

In recent years, the study of CNN architectures for classification problems is mainly divided into two directions. One is to design a deep network structure to study the effect of depth on classification. Another direction is to optimize the network structure and integrate the outputs of different network structures or different training methods as the final result. This paper is based on the second idea because it is known that ensembles of neural networks are much more robust and accurate than individual networks [16,17]. Ensemble learning is a machine learning paradigm where multiple learners are trained to solve the same problem. In this work the ensemble is composed of CNNs as base learners. The employment of different base learners generation processes and/or different combination schemes leads to different ensemble methods. Unlike our previous work [18] which is based on the use of individual CNNs, in this paper we propose to build an ensemble of Convolutional Neural Networks to identify the most relevant images in the database for a query image. To describe in detail the proposed model as well as the experimental results, the article is organised as follows. Section 2 introduces some previous related works. The methodological aspects associated with the development of the mentioned neural network system are detailed in Section 3. Experimental setup and results are presented and discussed in Section 4. Finally, conclusions are given in Section 5.

2. Related work

After the success of deep convolutional neural networks in large scale image classification [10], CNNs have been applied successfully to many computer vision tasks [19,20,21,22,23,24,25] including image retrieval by content.

Early applications of CNNs in image retrieval consisted in extracting features from fully connected layers [26,27]. However, due to the lack of spatial information in deeper layers, later works used convolutional layers instead [28,29,30].

Features can be extracted from either a pretrained or a fine-tuned network. VGG [31] and RESNET [32] are examples of pretrained networks, which are widely used in the literature [33,34,35]. The experiments in [36] do not only show the improvements that can be achieved through deep learning in multimedia and computer vision, but also show how to apply and adapt an existing deep learning model trained in one domain to a new CBIR task in another domain. Fine-tuning allows adapting the network to a desired task such as landmark retrieval [37,38]. In [37] Babenko *et al.* introduced "The Landmarks dataset" to evaluate fine-tuned CNNs and obtained good results. In [38], another large-scale challenging dataset was introduced, labeled "Google-Landmarks". This dataset was utilized to retrain Resnet50 for effective feature extraction.

In general, works in CBIR that adopt deep learning focus on enhancing the discrimination power of extracted convolutional features. A variety of techniques for improving the final image representation was proposed. As examples, the work presented in [39] proposed a CBIR system based on a CNN and a SVM, where the CNN is used to extract the feature representations and the SVM is used to learn the similarity measures. In the training of the SVM, the generation of a validation set helps to tune the parameters. Another case can be found in [40], where a Triplet-CNN model is employed. The latter contains three identical CNNs sharing all the weights and biases. Additionally, it is trained considering intra-class and inter-class distances. Furthermore, features were regarded as heat sources and weighted with the heat equation in [41]. So that the "hot" features were assigned larger weights to have an important impact in the retrieval. An unsupervised strategy was proposed in [42] to select the discriminative regions from which features were extracted. These regional features were weighted according to their corresponding regions and aggregated into a single feature vector. Authors in [38] proposed to extract feature maps from an image pyramid, and then they fine tuned RESNET50 to improve these features. Moreover, features were filtered using an attention based keypoint selection mechanism so that only the most powerful ones participated in the retrieval. In [43] three masks were presented for features filtering: SIFT-mask, SUM-mask, and MAX-mask. In addition, the authors used recent embedding and aggregation methods for better performance.

Most of CNN CBIR methods extract features from a single CNN. Alternatively, features can be extracted from several CNNs and then combined, which is the idea of "ensemble learning". The motivation behind using an ensemble of CNNs is that the collaboration of several CNNs helps reducing the bias and the variance [44] and improving the accuracy.

Ensemble learning has been successfully applied to many problems [45,46,47,48,49,50,51]. An example

of this is the model called “Hydra" for land use classification in satellite images [52]. It uses two state-of-the-art CNN architectures, which are ResNet and DenseNet to create ensembles of CNNs, varying the training set. During testing, each classifier output is the probability of each class being the correct one. A majority voting is then used to determine the final label. Very recently, some works were interested in applying ensemble learning in image retrieval by content [53,54]. In [53] images were represented by combined hash codes, which were generated from multiple views. The hash codes were learned using an ensemble of classifiers, where each classifier was assigned a different image view. In [54] the authors proposed to learn images features using two different CNN architectures (NIN and Alexnet). Features vectors of each image were then combined using the weighted average of the outputs of each CNN.

Despite its demonstrated efficiency, ensemble learning is still poorly addressed in the area of image retrieval by content. However, it must be considered that the most recent works focused on the search of criteria to form a good ensemble: Dynamic ensemble learning algorithms [55]. This paper deals with the problem of automatically detecting NN ensemble architectures (number of NNs in the ensemble and number of hidden neurons in individual NNs) with the variation of training examples for each individual NN. This idea draws a promising future in the use of ensemble learning.

3. Methodology

We aim to design a Content Based Image Retrieval (CBIR) system based on deep learning neural networks. We propose to build an ensemble of Convolutional Neural Networks (CNNs), so that they collaborate to identify the most relevant images in the database for a query image. To this end, the CNNs to be combined must have an output layer with one output neuron for each object class in the training image database. They may have the same or different network architectures and parameter settings. Let us denote D the number of object classes. Before the ensemble is built, each CNN is trained on the images of the training database. One hot encoding is to be employed, i.e. the desired output associated to each image is a unit vector of size D with a one at the vector component corresponding to the correct class, and zeros elsewhere. Under this setup, each CNN is expected to approximate the probabilities that the input object belongs to each of the D classes under consideration:

$$f\left(\mathbf{X}\right) = \left(P\left(C_1|\mathbf{X}\right), ..., P\left(C_D|\mathbf{X}\right)\right) \in \left[0,1\right]^D \quad (1)$$

where $\mathbf{X}$ is the input image, and C_i is the i-th class, with $i \in \{1, ..., D\}$.

After all the CNNs are trained under the above specifications, an ensemble can be built from them. Let us denote M the number of CNNs to be combined, i.e. the ensemble size. Then the class probabilities can be estimated by the ensemble by averaging the estimations coming from each ensemble member:

$$f_{Mean}\left(\mathbf{X}\right) = \frac{1}{M}\sum_{j=1}^{M} f_j\left(\mathbf{X}\right) \in \left[0,1\right]^D \quad (2)$$

where $f_j\left(\mathbf{X}\right)$ is the output vector for the j-th CNN, given the input image $\mathbf{X}$. Another possibility is to use the component wise median, and then renormalize to a probability vector:

$$\hat{f}\left(\mathbf{X}\right) = \text{median}\left(\{f_j\left(\mathbf{X}\right) \mid j \in \{1, ..., M\}\}\right) \quad (3)$$

$$f_{Median}\left(\mathbf{X}\right) = \frac{1}{\left\|\hat{f}\left(\mathbf{X}\right)\right\|_1}\hat{f}\left(\mathbf{X}\right) \quad (4)$$

where $\|\cdot\|_1$ stands for the L1-norm (Manhattan norm) of a vector.

After the ensemble has been built, it can be employed to retrieve the most similar images in the database, given a user query. Let us denote $\mathbf{X}_{Query}$ the query image, and $\mathbf{X}_j$ the training database images, where $j \in \{1, ..., N\}$ and N is the number of images in the database. We propose to rank the images in the database according to the p-norm distances between the ensemble probability vector $f_h\left(\mathbf{X}_{Query}\right)$ which is obtained from the ensemble as it processes the query image, where $h \in \{Mean, Median\}$ is the ensemble type, and the probability vectors $f_h\left(\mathbf{X}_j\right)$ that are yielded by the ensemble as it processes the database images. Given these considerations, the image in the database with the best similarity to the query can be computed as follows:

$$s = \arg \min_{j \in \{1,...,N\}} \|f_h(\mathbf{X}_{Query}) - f_h(\mathbf{X}_j)\|_p \quad (5)$$

where $\|\cdot\|_p$ stands for the p-norm of a vector. Here p is a parameter which must be optimized by experimentation.

If we are interested in the k most similar database images given a query image, a ranking list can be obtained:

$$R_Q = \{\mathbf{X}_1^*, \mathbf{X}_2^*, ..., \mathbf{X}_k^*\} \quad (6)$$

R_Q is a subset of the training image database $\{\mathbf{X}_j : j = 1, ..., N\}$ where $\mathbf{X}_i^*$ is ranked before $\mathbf{X}_j^*$ when

$$\|f_h(\mathbf{X}_{Query}) - f_h(\mathbf{X}_i^*)\|_p < \|f_h(\mathbf{X}_{Query}) - f_h(\mathbf{X}_j^*)\|_p$$

Such a ranking process can be used to evaluate the performance of our CBIR system or when a parameter must be tuned.

If the number of results is not specified, then a similarity threshold ν should be used. In this case, the database images $\mathbf{X}_j \in R_Q$ are regarded as close to the input image if they satisfy the following condition :

$$\|f_h(\mathbf{X}_{Query}) - f_h(\mathbf{X}_j)\|_p < \nu \quad (7)$$

It is possible to add a simple way to refine the search of the closest images to a given query image using Average Query Expansion on the model proposed above. The proposed method is inspired by KNORA-UNION algorithm for dynamic ensemble selection [56]. The basic idea is described below.

Given an ensemble of M CNNs, and a query image $\mathbf{X}_{Query}$, after retrieving the most relevant images for this query (using our proposal: mean or median ensemble) we obtain $R_Q = \{\mathbf{X}_1^*, \mathbf{X}_2^*, ..., \mathbf{X}_k^*\}$.

We consider that a database image $\mathbf{X}_j^* \in R_Q$ which belongs to the class m is correctly classified by a CNN i in the ensemble if:

$$\max_{i=1,...,D} \{P^i(C_i|\mathbf{X}_j^*)\} = P^i(C_m|\mathbf{X}_j^*) \quad (8)$$

Let $\mathbf{V}_{ij}$ be the probability vector corresponding to the output of CNN$_i$ that correctly classifies the image $\mathbf{X}_j^* \in R_Q$, that is:

$$\mathbf{V}_{ij} = (P^i(C_1|\mathbf{X}_j^*), ..., P^i(C_D|\mathbf{X}_j^*)) = f_i(X_j^*) \quad (9)$$

In that situation we generate a new query image vector by taking the average of the original $f_h(\mathbf{X}_{query})$ and the selected vectors in the following set:

$$\{\mathbf{V}_{ij} : i \in S_1 \subset \{1, ..., M\}, j \in S_2 \subset \{1, 2, ..., k\}\} \quad (10)$$

Thus, the suppression of misclassified samples allows the controlled construction of extended queries.

The designed CBIR system is described in Fig. 1, where a flowchart allows to follow the process from training the model to testing it. In the training phase, if the used CNNs share the same architecture, diversity is achieved through splitting the training images into M bags, which are used to train M classifiers. However, if the CNNs have different architectures, each CNN is trained on the whole training set, since the variation of the architecture is supposed to make a diverse ensemble. After that, the extracted class probability vectors from the training images are combined and stored in the database. In the testing phase, the trained classifiers are used to extract the query class probability vectors. Similarly to the training images vectors, the query vectors are combined to generate a single query representation. The p-norm distance is then used for similarity measurement in order to retrieve the k most similar images. Optionally, we can enhance the returned list of similar images using Average Query Expansion. As explained above, the base vectors that belong to the ranked list of images are selected and combined with the query vector. Finally, the retrieval process is carried out again.

4. Experiments and results

4.1. Methods

We have considered several recent well-known methods from the literature in order to compare our proposal with them. We took into consideration the best results of each competitor. The first method that we have selected is the approach that we denote as *Wan* [36]. In this work, images features are extracted from the last three fully connected layers and refined

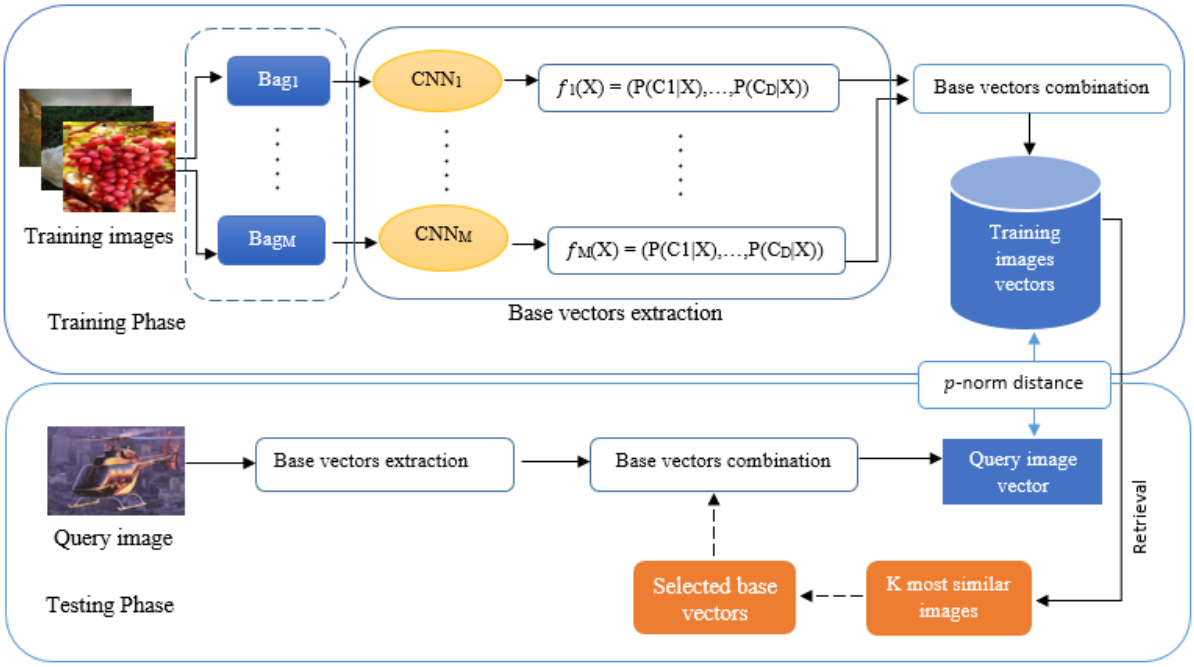


Fig. 1. Schema of our proposal. The dashed frame corresponds to the bagging technique which is carried out only if the ensemble is composed of the same CNN architecture. The dashed arrows correspond to the Average Query Expansion process which can be applied on the query one time.

by similarity learning using the Online Algorithm for Scalable Image Similarity Learning ("OASIS").

We also consider a method that we denote *Cai* [40]. In this algorithm, a triplet CNN-based architecture is used for feature extraction. First, the triplet CNN is re-trained on the used dataset. After that, the features are extracted from the last two fully connected layers and then combined in order to obtain an efficient representation with different semantic levels.

The last competitor method is denoted as *Fu* [39]. It uses a CNN to extract images features. After that, an SVM is trained on the dataset to differentiate between similar images and dissimilar ones.

Regarding our proposal, it uses several CNNs having the same or different architecture. This way, well-known CNNs have been employed in the experiments:

- *VGG* [57]: This architecture is characterized by the small size of convolutional filters receptive field (3x3). Some of the convolutional layers are followed by Max-pooling layers, which perform the spatial pooling. The used variants of this architecture are: VGG16 and VGG19. They consist of 16 and 19 weights layers, respectively, hence their names.
- *ResNet* [32]: The main characteristic of this network is its depth which is up to 152 layers. Moreover, it has residual connections that decrease the complexity of the network and make its training easier. Similar to VGG, this architecture has 3x3 filter size for convolutional layers. We use several variants of this architecture depending on the network depth: ResNet50 (50 layers), ResNet101 (101 layers), and ResNet152 (152 layers).
- *ResNetV2* [58]: ResNetV2 is an improved version of ResNet. The main difference is in use of identity skip connections and identity after-addition activation to fasten the propagation of information. Similar to ResNet, this architecture is available with different depths, resulting in three variants: Resnet50V2 (50 layers), ResNet101V2 (101 layers), and ResNet152V2 (152 layers).
- *Inception* [59]: The variants that we use are InceptionV3[60] and InceptionResNet [61]. The latter are successors of InceptionV1 (GoogleNet) and they inherit the use of special layers called "Inception layers/modules". Instead of doing a single operation on an input, an inception module allows several operations in parallel: 1x1 convolution, 1x1 convolution followed by 3x3 con-

volution, 1x1 convolution followed by 5x5 convolution, and max-pooling followed by 1x1 convolution. The outputs are then concatenated and passed through the network. Moreover, the use of convolutions of different filter sizes allows capturing patterns at different scales. While, InceptionV3 reduces the filter size to speed up the training, InceptionResNet benefits from both ResNet and InceptionV1 architectures by using residual inception layers, which reduces the architecture's complexity.

- *Xception* [62]: It is worth mentioning that Xception is inspired by Inception architecture, i.e. it allows using several operations in the same layer. However, unlike Inception these operations are restricted to convolutions. Thus, Inception modules are replaced by special convolutions called "depthwise separable convolutions", where, instead of having one convolution, several independent convolutions followed by a pointwise convolution are carried out. Note that a pointwise convolution has a filter size of 1x1. The replacement of Inception modules by depthwise separable convolutions led to a decrease in the computational cost. Moreover, this architecture defeats InceptionV3 on the ImageNet dataset in terms of accuracy.
- *MobileNet* [63]: We use the two versions of MobileNet architecture: MobileNet and MobileNetV2. This architecture is designed for mobiles and embedded vision applications. Similar to Xception, it uses depthwise separable convolutions to reduce the computational cost. However, the filter size is fixed to 3x3. All layers (except the fully connected layer) are followed by a batch normalization and a *relu* non linearity activation. In addition, MobilenetV2 [64] improves this architecture by the use of an inverted residual structure, and the suppression of the non linearity, which enhances the accuracy.
- *Densenet* [65]: The concept of Densenet architecture is based on residual connections introduced in ResNet. The idea is to connect each layer to all subsequent layers, in a manner that every layer has the outputs of all predecessor layers. This leads to a better propagation of information and speeds up the training. We use this architecture with different depths: DenseNet-121 (121 layers), DenseNet-169 (169 layers), and DensNet-201 (201 layers).
- *NasNet* [66]: The main novelty is that the architecture is learnt on a small dataset (Cifar [67]) using reinforcement learning, after that this architecture is used to train the network on a large scale dataset (ImageNet). Moreover, it uses "ScheduledDropPath" as a new regularization technique, which improves the model's accuracy. The concept of the variants NasNetMobile and NasNetLarge is the same, with the difference that NasNetMobile is conceived for mobile devices.

In order to code our approach, the used programming language was Python with its machine learning library Keras[1] running on top of Tensorflow[2].

The experiments have been conducted on Google colab[3] running on a NVIDIA Tesla k80 GPU.

4.2. Datasets

Our proposal is evaluated on two benchmark datasets:

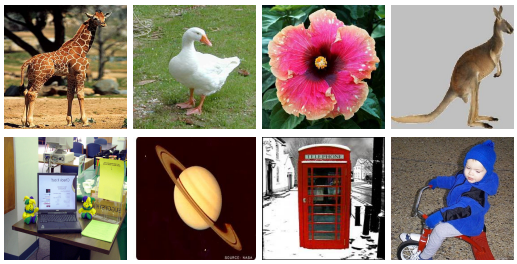
Caltech256[4] [57]: This dataset contains 30,608 images, grouped into 256 semantic categories. Moreover, it exhibits at least 80 images per category.

ImageNet[5] [68]: In order to evaluate our proposal on a large number of classes, we use the "ILSVRC-2010" dataset which is a subset from ImageNet. This dataset contains 1.2 million images for training, 50,000 images for validation, and 150,000 images for testing, all grouped into 1000 categories.

The great variety of categories included in both datasets can be observed in Fig. 2, where some samples are shown.

In regards to Caltech256 dataset, for a fair comparison with methods Wan [36], Cai [40] and Fu [39], we follow the same experimental setting by using subsets of 20 and 50 classes from Caltech256 dataset, and images from each class were randomly split into training and testing sets of 40 and 25 images, respectively. Moreover, 15 images from each category were randomly selected for validation. In each subset, the classes were selected as suggested in [69]. The motivation behind this selection is to include diverse categories in terms of their semantics and their classification difficulty as measured by [57], where it is shown

[1] https://keras.io
[2] https://tensorflow.org
[3] https://colab.research.google.com
[4] https://kaggle.com/jessicali9530/caltech256
[5] http://image-net.org/challenges/LSVRC/2010/index



(a) Caltech256

(b) ImageNet

Fig. 2. Samples of Caltech256 and ImageNet datasets

that the performance varies depending on the selected class.

For ImageNet dataset, the validation set is split into testing set and retrieval database. We select one query per category to evaluate our proposal, that is 1000 testing images. Moreover, we use 5000 images as a retrieval database, equally grouped into the 1000 categories. Images in both testing and validation sets were randomly selected.

4.3. Parameter selection

In order to achieve diversity, we evaluated two ensemble techniques to generate the pool of classifiers. The first one is the bagging technique [70]. In the corresponding experiments, M training different sets of equal size (60% from the training data) were randomly drawn with replacement, where M = 20. After that, each classifier was trained on a different bag of images with aim to generate an ensemble of M classifiers. This technique is evaluated on Caltech256 dataset. This way, several networks with the same architecture but different training data are considered in this first technique. The deep convolutional neural network architecture that we have employed using this technique is VGG16 [31]. We have employed a fine-tuning process, where we have removed the fully connected layer from the original network and added 4 layers: A flatten layer, a dense layer with 256 neurons and a *relu* activation function, a dropout layer with a coefficient of 0.5 to prevent overfitting, and a last dense layer to specify the number of classes where the used activation function was *softmax*. The fine-tuning process was performed by training the added layers on the experimented dataset. The experimental fine tuning setting consists of a batch size equals to 64, while the loss function is the categorical cross entropy. Additionally, the optimizer is the stochastic gradient descendant (SGD) with a learning rate of 10^{-4} and a momentum of 0.9.

The second ensemble technique is the use of classifiers having different architectures. All available pre-trained CNNs in Keras library are used in this second technique, that is 18 different architectures. This second technique is evaluated on ImageNet dataset. Herein, no training phase is carried out, since the used CNNs are already trained on this dataset.

In the rest of the paper, we denote both techniques with the name of the used dataset in each experiment. That is, Caltech256 (or ensemble with networks with the same architecture) and ImageNet (or ensemble with networks with different architectures).

4.4. Results

In order to evaluate our proposal from a quantitative point of view, we have employed some well known performance evaluation measures. These measures are: Precision at k (P@k), Recall at k (R@k), where k is the number of images to retrieve, Precision given a distance threshold (P_ν), Recall given a distance threshold (R_ν), where ν is the threshold value, and Mean Average Precision (mAP). They provide a real number between 0 and 1, where higher is better, and they are given in the following equations:

$$P@k, P_\nu = \frac{Number\ of\ relevant\ images\ retrieved}{Total\ number\ of\ retrieved\ images} \tag{11}$$

$$R@k, R_\nu = \frac{Number\ of\ relevant\ images\ retrieved}{Total\ number\ of\ relevant\ images} \tag{12}$$

$$mAP = \frac{1}{L}\sum_{l=1}^{L} AP_l \tag{13}$$

where AP_l is the average precision for a query l, and L is the total number of queries.

Additionally, the average precision is defined by:

$$AP_l = \sum_{k=1}^{K} P_l@k \times (R_l@k - R_l@(k-1)) \quad (14)$$

Please note that the number of the required iterations in equation (14) is the number of the retrieved images k. Generally, K is the number of the required iterations to achieve a perfect recall. In our proposal, setting K to that value for all the conducted experiments is very time consuming. Thus, we set K to the number of relevant images given a query, which is 40 for Caltech256 dataset, and 5 for ImageNet dataset.

If the number of images to retrieve is defined, precision and recall are denoted as $P@k$ and $R@k$, respectively. If a distance threshold is given, then the two measures are referred to as P_ν for precision, and R_ν for recall.

We have studied the impact of several parameters on the performance of our proposal. The first parameter p is used to compute the p-norm distance. In order to improve the retrieval results, several p-norm distances were evaluated in the retrieval step. Table 1 shows a comparison between the p-norm distances for the considered p values on both Caltech256 and ImageNet datasets. Depending on the parameter p, the considered distances are: Manhattan distance ($p = 1$), 1.5-norm distance ($p = 1.5$), Euclidean distance ($p = 2$), and Tchebychev distance ($p = \infty$). Results show that Manhattan distance outperformed other tested p-norm distances using both ensemble methods. Based on this, in all subsequent experiments Manhattan distance is considered to measure the similarity between the queries and the database images.

Moreover, we have studied the impact of the ensemble size M on the mAP. The results are shown in Fig. 3. For Caltech256 dataset, we can observe a positive correlation between the mAP and the parameter M, where for 20 classes $1 \leq M \leq 14$ using the mean, and $1 \leq M \leq 8$ using the median. For 50 classes, $1 \leq M \leq 18$ using the mean, and $1 \leq M \leq 13$ using the median. The highest mAP for 20 classes is 0.7370 using the mean, and 0.7442 using the median. The corresponding ensemble size is M = 18 for both methods.

For 50 classes, The highest mAP is 0.5749 using the mean, where M = 18. Using and the median, the most accurate ensemble is of size M = 17, the corresponding mAP is 0.5640.

P	Mean	Median
1	**0.7363**	**0.744**
1.5	0.7275	0.7206
2	0.7209	0.7194
∞	0.7079	0.7417

(a) Caltech256 20 classes

P	Mean	Median
1	**0.5726**	**0.558**
1.5	0.5597	0.5418
2	0.5514	0.5402
∞	0.5419	0.5526

(b) Caltech256 50 classes

P	Mean	Median
1	**0.4939**	**0.4904**
1.5	0.4593	0.4657
2	0.4549	0.4642
∞	0.4474	0.4649

(c) ImageNet

Table 1

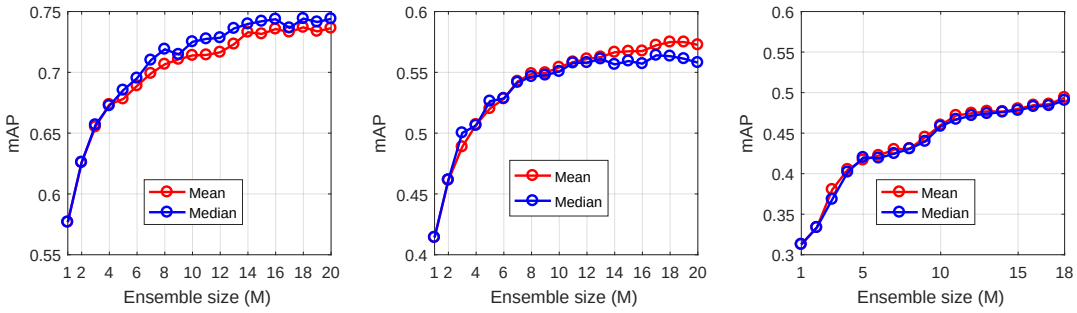
A comparison between several p-norm distances in terms of mAP on Caltech256 and ImageNet datasets, where M = 20 for Caltech256, and M = 18 for ImageNet. Both mean and median are considered in the comparison, best results are highlighted in **bold**.

For the same dataset, the most important margin in terms of mAP was recorded between ensemble sizes M=1 and M=2, where the mAP increased significantly. We refer the reason for this to the evolution of the architecture from individual CNN-based to CNN-based ensemble. Moreover, we can observe that the positive correlation is more significant for $1 \leq M \leq 10$. Outside this interval, the increase in performance is less important. Thus, we stop the evolution of the ensemble at M = 20.

Based on these results, the ensemble size M is set to the value that corresponds to the highest mAP in all the following experiments. That is, for 20 classes $M = 18$ using the mean and the median, for 50 classes $M = 18$ using the mean, and $M = 17$ using the median.

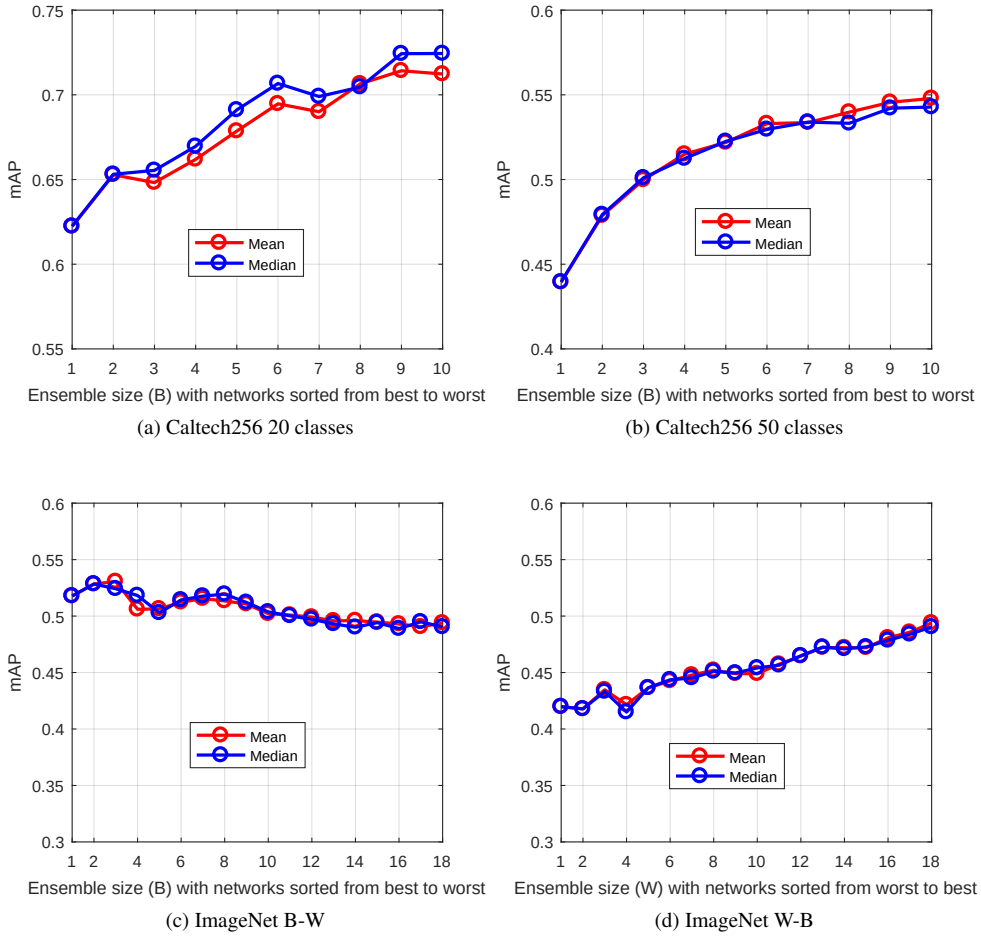
The general decrease in the classification performance when we pass from 20 classes to 50 classes is due to the greater difficulty in separating more classes. Our ensemble proposal is not related to this performance decrease, since it can be seen in Figure 3 that the effect is present even for just a single CNN (ensemble size M = 1), i.e. no ensemble. Moreover, according to the study presented in [57], a negative correlation is shown between the number of classes and the measured performance, where a significant decrease in performance was recorded between 20 classes and 50 classes.

Regarding ImageNet dataset, a significant improvement is observed for $1 \leq M \leq 5$ using the mean



(a) Caltech256 20 classes (b) Caltech256 50 classes (c) ImageNet

Fig. 3. Impact of the ensemble size on the mAP for Caltech256 and ImageNet datasets where the considered methods are mean and median.



(a) Caltech256 20 classes (b) Caltech256 50 classes

(c) ImageNet B-W (d) ImageNet W-B

Fig. 4. Impact of the ensemble size with respect to the classifiers accuracy on the mAP for Caltech256 and ImageNet datasets, where the considered methods are mean and median.

and the median. For $5 \leq M \leq 18$, the correlation between the ensemble size M and the mAP is still positive. However, the improvement is less important. Note that the highest mAP was reached with $M = 18$ for

both mean and median, where mAP = 0.4939 for the mean, and mAP = 0.4904 for the median. Even though the accuracy is generally increased after increasing the ensemble size M, the ensemble was outperformed by the single best classifier (NasNetLarge) whose mAP was 0.5178. Thus, we investigated two other ways for ensemble construction. In the first one, the ensemble was built with respect to the classifiers accuracy. B is set to the number of the most accurate classifiers. the B classifiers were sorted based on their top-5 accuracy on the ImageNet validation set, from best to worst (B-W). After that, several ensembles of size B were created, where B is set to the number of the most accurate classifiers. Moreover, $1 \leq B \leq M$. So that, B = 1 corresponds to the most accurate classifier, B = 2 corresponds to the two most accurate classifiers ensemble and so on. These ensembles are evaluated in terms of mAP as shown in Figure 4. This way, the ensemble could achieve a better accuracy and outperformed the single best classifier. The highest mAP reached by the mean was 0.5306, which corresponds to B = 3. In addition, the highest mAP reached by the median was 0.5284, where B = 2.

In the same way, we ordered the CNNs from worst to best (W-B), and created ensembles of size W. Results can be observed in Figure 4. Generally, there is an improvement in the mAP as the ensemble size W increases. The highest mAP value was reached with W=18 using the mean and the median, where it equals 0.4939 for the mean, and 0.4904 for the median. It should be highlighted how the ensemble B-W performance decreases when networks are added; on the other hand, the ensemble W-B performance is higher when networks are added. As it can be observed, the performance of the ensemble not only depends on the accuracy of the network which has been added to the ensemble; it also depends on the networks which belong to it.

Based on these results, for ImageNet dataset M is set to B in all subsequent experiments.

We repeated the experiment where classifiers are sorted from Best to worst on Caltech256 dataset. The classifiers are sorted based on their accuracy on the training and validation sets. We selected M classifiers from the pool, where M = 10. As shown in Figure 4, in most cases, there is a positive correlation between the ensemble of the most accurate classifiers B and the mAP, where even though a less performing classifier is added to the ensemble, the larger-sized ensemble was more accurate. However, for some cases, there was a slight degradation in the performance after increasing the ensemble size B, which is possibly due to the inclusion of a less accurate classifier in the pool. Note that the highest mAP for 20 classes was reached with B = 9 using the mean, where it equals 0.7142, and B = 10 using the median, where it equals 0.7244. For 50 classes, B = 10 corresponds to the best performing ensemble with 0.5428 for the mean and 0.5479 for the median.

The retrieval of similar images can be done either by defining the number of images to retrieve, or by considering a threshold for the distance between the query and the database images. After deciding the ensemble size M, we evaluated the distance threshold ν, which is another important parameter that affects the retrieval quality. In order to enhance the relevance of the returned ranked list, we have experimented 20 threshold values, where $0.1 \leq \nu \leq 2$. The impact of the value ν on both precision and recall is shown in Fig. 5. The threshold ν was set to 1 for Caltech256 and ImageNet datasets, which is a good compromise between precision and recall.

Taking into account these analyzed configurations, the performance of our CNN-based methods is shown in Table 2. We refer to the ensemble methods as Mean and Median, where the mean and the median were used to combine the extracted class probability vectors using each ensemble member, respectively. We refer to the base classifier as base. This classifier was trained on the whole training set and used individually to extract the class probability vectors from images. For the two datasets, the results show that both ensemble methods outperform significantly the base classifier in terms of all performance evaluation measures. The exception occurs in R_ν computed on ImageNet dataset, which does not necessarily reflect the outperformance of the base in terms of this performance evaluation measure, since it depends on the used threshold value ν.

The reason behind the outperformance of the ensemble architecture is that the latter takes advantage of the lack of correlation between the base learners. This lack of correlation is produced by training the CNNs on different bags of images. Thus, the CNNs are able to learn different patterns from images and make independent errors. For instance, if a CNN misclassifies a given input, then the collaboration of other CNNs may correct this error by combining their provided outputs for this image. Consequently, the ensemble is able to provide more powerful predictions, resulting in the improvement of the retrieval quality.

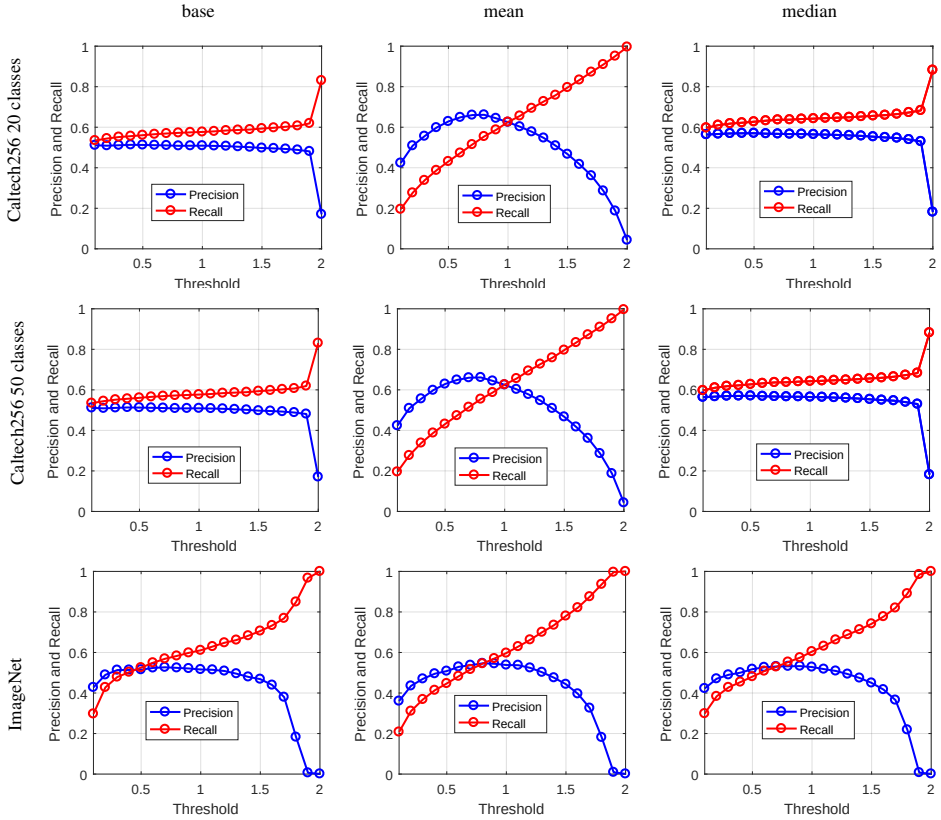
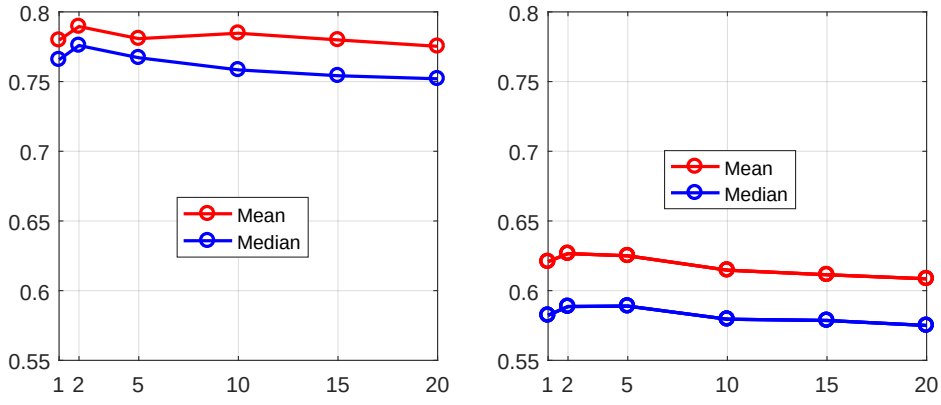


Fig. 5. Precision and recall in terms of distance threshold on Caltech256 and ImageNet datasets. The considered methods are: Base, Mean, and Median, where results are shown in the left, middle, and the right of the figure, respectively. Images from the first row correspond to Caltech256 with 20 classes, images from the second row correspond to Caltech256 with 50 classes, and images from the third row correspond to ImageNet dataset



(a) Caltech256 20 classes (b) Caltech256 50 classes

Fig. 6. Impact of the number of neighbours considered in the Average Query Expansion technique on the mAP for Caltech256 dataset, where the considered methods are mean and median.

Finally, we have applied the previously mentioned post processing technique called Average Query Expansion. We investigated the impact of the parameter k on the mAP. This parameter represents the number

Method	$P@1$	$P@10$	$P@50$	$P@100$	P_ν	$R@1$	$R@10$	$R@50$	$R@100$	R_ν	mAP
Base	0.692	0.6594	0.4925	0.2921	0.5401	0.0173	0.1649	0.6156	0.7302	0.5725	0.5265
Mean	**0.848**	**0.8398**	**0.6596**	**0.3613**	**0.7753**	**0.0212**	**0.2099**	**0.8246**	**0.9032**	0.7414	0.7370
Median	0.844	0.8296	0.6455	0.3554	0.7341	**0.0212**	0.2074	0.8068	0.8885	**0.7609**	**0.7442**

(a) Caltech256 20 classes

Method	$P@1$	$P@10$	$P@50$	$P@100$	P_ν	$R@1$	$R@10$	$R@50$	$R@100$	R_ν	mAP
Base	0.6408	0.6269	0.4685	0.2762	0.509	0.016	0.1567	0.5856	0.6905	0.5767	0.4909
Mean	**0.7456**	**0.7201**	**0.5483**	**0.3173**	**0.6248**	**0.0186**	**0.1800**	**0.6853**	**0.7932**	0.6258	**0.5749**
Median	0.728	0.6958	0.5198	0.2994	0.5646	0.0182	0.174	0.6498	0.7485	**0.6425**	0.5640

(b) Caltech256 50 classes

Method	$P@1$	$P@10$	$P@50$	$P@100$	P_ν	$R@1$	$R@10$	$R@50$	$R@100$	R_ν	mAP
Base	0.5970	0.3277	0.0761	0.0397	0.5161	0.1194	0.6554	0.7612	0.7950	**0.6108**	0.5178
Mean	**0.6280**	**0.3352**	**0.0791**	**0.0419**	**0.5381**	**0.1256**	**0.6704**	**0.7914**	**0.8390**	0.5972	**0.5306**
Median	0.6200	0.3332	0.0781	0.0412	0.5282	0.1240	0.6664	0.7814	0.8244	0.6046	0.5284

(c) ImageNet

Table 2

Image retrieval performance on the experimented datasets, best results are highlighted in **bold**.

of retrieved neighbours used to generate the new query vector. Evaluated k values were: 1, 2, 5, 10, 15, and 20. Fig. 6 shows the mAP in terms of k. For 20 classes, the highest mAP was reached with $k = 2$ using the mean and the median. For 50 classes, it was reached with k = 2 using the mean and $k = 5$ using the median. After applying the Average Query Expansion technique to our architecture, both ensemble methods achieved better results in terms of mAP, as shown in Table 3.

P	mAP
Mean	0.7370
Mean + QE	**0.7895**
Median	0.7442
Median + QE	0.7759

(a) 20 classes

P	mAP
Mean	0.5749
Mean + QE	**0.6266**
Median	0.5640
Median + QE	0.5889

(b) 50 classes

Table 3

mAP using the mean and the median with and without applying the Average Query Expansion to Caltech256 dataset, best results are highlighted in **bold**.

Results can be observed from a qualitative point of view. Fig. 7 shows an example of a query and the top 10 retrieved images. In the given example, for Caltech256, both mean and median performed equally in terms of precision, where ($p@10 = 0.9$). For ImageNet dataset, the same precision value was obtained by the mean and the median, where ($p@10 = 0.3$). Note that the Average Query Expansion is not applied in this retrieval example.

In order to further check the performance of our proposal, we compared it against relevant retrieval methods. The comparison is carried out on Caltech256 dataset in terms of mAP. In order to be comparable with the selected methods, the parameter k defined in equation 14 was set to the number of iterations required to achieve a perfect recall. In addition, no post processing technique is applied to our method. Results shown in Table 4 confirm the efficiency of our method where both mean and median outperform the competing methods. We can also observe that the mean achieved the best mAP with a slight difference compared to the median on 20 classes. However, when evaluated on 50 classes, this outperformance was more significant.

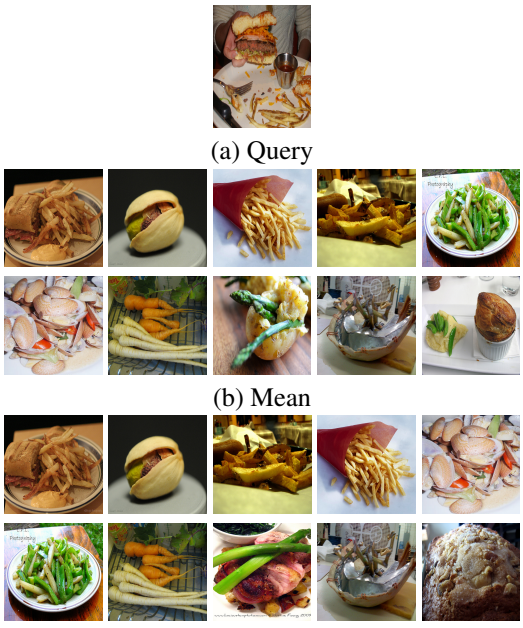
5. Conclusions

A novel architecture for image retrieval by content has been presented. This architecture is based on an ensemble of CNNs, which are trained on different bags

(a) Query

(b) Mean

(c) Median
Caltech256 dataset



(a) Query

(b) Mean

(c) Median
ImageNet dataset

Fig. 7. A query image and the top 10 returned images from both Caltech256 and ImageNet datasets using the mean and the median.

of images, so that a variety of class probability vectors could be acquired from each image. The obtained image probability vectors are then combined into a single vector as a final image representation. Finally, we applied an Average Query Expansion technique to our proposal to improve the retrieval quality.

Method	mAP
Wan [36]	0.7388
Cai [40]	0.6726
Fu [39]	0.6106
Mean	**0.8158**
Median	0.8126

(a) 20 classes

Method	mAP
Wan [36]	0.5410
Cai [40]	0.5532
Fu [39]	0.4264
Mean	**0.6713**
Median	0.6456

(b) 50 classes

Table 4

Comparison between our proposal (mean and median) and related methods in terms of mAP, best results are highlighted in **bold**. Results indicated in percentage in original papers are converted to decimal fractions.

The averaging of the class probability vectors coming from an ensemble of CNNs has the beneficial effect that the expected error of the average probability vector with respect to the true probability vector is lower than the expected error of any individual probability vector coming from a single CNN and the true probability vector.

Results show that our method outperforms the state of the art in terms of mAP. However, ensemble based architectures are computationally expensive, i.e. our architecture takes more time in the output prediction depending on the ensemble size. The delay in the response time will be more significant for larger-sized ensembles. Thus, employing the proposed architecture depends on how critical the response time is in the desired CBIR system.

The key idea of our contribution is to build an accurate classifier by combining several weak learners. In this regard, it should be emphasized that it is possible to address this issue within CBIR, i.e. enhancing the class probability vector using other techniques. Away from ensemble methods, EPNN [71] and NDS [72] are individual classifiers that aim at enhancing the classification accuracy. In EPNN, the improvement in the classification is achieved through considering local information and heterogeneity in the training data. In addition, NDS aims at discovering feature spaces that maximize margins between clusters and minimize them between classmates. Thus, our contribution leaves open the possibility of exploring other classification models within the CBIR area.

Finally, it can be interesting to study possible improvements in this proposed architecture. One idea could be a dynamic selection of an ensemble of classi-

fiers from the pool. Dynamic ensemble selection may empower the proposed architecture by selecting the most powerful classifiers for each image and reducing the computational cost.

Acknowledgments

This work is partially supported by the Ministry of Economy and Competitiveness of Spain under grants TIN2016-75097-P and PPIT.UMA.B1.2017. It is also partially supported by the Ministry of Science, Innovation and Universities of Spain [grant number RTI2018-094645-B-I00], project name Automated detection with low cost hardware of unusual activities in video sequences. It is also partially supported by the Autonomous Government of Andalusia (Spain) under project MA18-FEDERJA-084, project name Detection of anomalous behavior agents by deep learning in low cost video surveillance intelligent systems. All of them include funds from the European Regional Development Fund (ERDF). In addition, this work is partially supported by DGRSDT - Ministry of Higher Education and Scientific Research of the Algerian Government through a PRFU project [grant number A10N01UN2101201800001]. The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Málaga. They have also been supported by the Biomedic Research Institute of Málaga (IBIMA). They also gratefully acknowledge the support of NVIDIA Corporation with the donation of two Titan X GPUs used for this research. The authors also thankfully acknowledge the support of the Universidad de Málaga. Authors are also immensely grateful for ERASMUS+ program, CEI.MAR (Campus de Excelencia International del Mar), and University of 20 August 1955 for making this collaborative work possible.

References

[1] G.L. Ying Liu Dengsheng Zhang and W.-Y. Ma, A survey of content-based image retrieval with high-level semantics, *Pattern Recognition* **40**(1) (2007), 262–282.

[2] D. Zhang and G. Lu, Shape-based image retrieval using generic Fourier descriptor, *Signal Processing: Image Communication* **17**(10) (2002), 825–848.

[3] G.-H. Liu, L. Zhang, Y.-K. Hou, Z.-Y. Li and J.-Y. Yang, Image retrieval based on multi-texton histogram, *Pattern Recognition* **43**(7) (2010), 2380–2389.

[4] C.-C. Lai and Y.-C. Chen, A user-oriented image retrieval system based on interactive genetic algorithm, *IEEE transactions on instrumentation and measurement* **60**(10) (2011), 3318–3325.

[5] E. Rashedi, H. Nezamabadi-Pour and S. Saryazdi, A simultaneous feature adaptation and feature selection method for content-based image retrieval systems, *Knowledge-Based Systems* **39** (2013), 85–94.

[6] P. Liu, J.-M. Guo, K. Chamnongthai and H. Prasetyo, Fusion of color histogram and LBP-based features for texture image retrieval and classification, *Information Sciences* **390** (2017), 95–111.

[7] Y. Bengio, A.C. Courville and P. Vincent, Unsupervised feature learning and deep learning: A review and new perspectives, *CoRR* **abs/1206.5538**(52) (2012), 123–456.

[8] H. Chang and D.-Y. Yeung, Kernel-based distance metric learning for content-based image retrieval, *Image and Vision Computing* **25**(5) (2007), 695–703.

[9] D.H. Ackley, G.E. Hinton and T.J. Sejnowski, A learning algorithm for Boltzmann machines, *Cognitive science* **9**(1) (1985), 147–169.

[10] A. Krizhevsky, I. Sutskever and G.E. Hinton, Imagenet classification with deep convolutional neural networks, *NIPS* **25** (2012), 1106–1114.

[11] O.M. Manzanera, S.K. Meles, K.L. Leenders, R.J. Renken, M. Pagani, D. Arnaldi, F. Nobili, J. Obeso, M.R. Oroz, S. Morbelli et al., Scaled Subprofile Modeling and Convolutional Neural Networks for the Identification of Parkinson's Disease in 3D Nuclear Imaging Data, *International journal of neural systems* **29**(09) (2019), 1950010.

[12] Y. Gao and K.M. Mosalam, Deep transfer learning for image-based structural damage recognition, *Computer-Aided Civil and Infrastructure Engineering* **33**(9) (2018), 748–768.

[13] S. Bang, S. Park, H. Kim and H. Kim, Encoder–decoder network for pixel-level road crack detection in black-box images, *Computer-Aided Civil and Infrastructure Engineering* **34**(8) (2019), 713–727.

[14] R.-T. Wu, A. Singla, M.R. Jahanshahi, E. Bertino, B.J. Ko and D. Verma, Pruning deep convolutional neural networks for efficient edge computing in condition assessment of infrastructures, *Computer-Aided Civil and Infrastructure Engineering* **34**(9) (2019), 774–789.

[15] A.H. Ansari, P.J. Cherian, A. Caicedo, G. Naulaers, M. De Vos and S. Van Huffel, Neonatal seizure detection using deep convolutional neural networks, *International journal of neural systems* **29**(04) (2019), 1850011.

[16] L.K. Hansen and P. Salamon, Neural networks ensembles, *IEEE transactions on pattern analysis and machine intelligence* **12** (1990), 993–1001.

[17] A. Krogh and J. Vedelsby, Neural network ensembles, cross validation, and active learning, *NIPS* **7** (1995).

[18] S. Hamreras, R. Benítez-Rochel, B. Boucheham, M.A. Molina-Cabello and E. López-Rubio, Content Based Image Retrieval by Convolutional Neural Networks, in: *International Work-Conference on the Interplay Between Natural and Artificial Computation*, Springer, 2019, pp. 277–286.

[19] T. KarpathyA et al., Large-scale video classification with convolutional neural networks, in: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[20] O. Ronneberger, P. Fischer and T. Brox, U-net: Convolutional

networks for biomedical image segmentation, in: *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[21] Y.-z. Lin, Z.-h. Nie and H.-w. Ma, Structural Damage Detection with Automatic Feature-Extraction through Deep Learning, *Computer-Aided Civil and Infrastructure Engineering* **32**(12) (2017), 1025–1046.

[22] M. Koziarski and B. Cyganek, Image recognition with deep neural networks in presence of noise–Dealing with and taking advantage of distortions, *Integrated Computer-Aided Engineering* **24**(4) (2017), 337–349.

[23] P. Wang and X. Bai, Regional parallel structure based CNN for thermal infrared face identification, *Integrated Computer-Aided Engineering* **25**(3) (2018), 247–260.

[24] U.R. Acharya, S.L. Oh, Y. Hagiwara, J.H. Tan and H. Adeli, Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals, *Computers in biology and medicine* **100** (2018), 270–278.

[25] F. Vera-Olmos, E. Pardo, H. Melero and N. Malpica, DeepEye: Deep convolutional network for pupil detection in real environments, *Integrated Computer-Aided Engineering* **26**(1) (2019), 85–95.

[26] Y. Gong, L. Wang, R. Guo and S. Lazebnik, Multi-scale orderless pooling of deep convolutional activation features, in: *European conference on computer vision*, Vol. 8695, Springer, 2014, pp. 392–407.

[27] A.S. Razavian, J. Sullivan, S. Carlsson and A. Maki, Visual instance retrieval with deep convolutional networks, *ITE Transactions on Media Technology and Applications* **4**(3) (2016), 251–258.

[28] A. Babenko and V. Lempitsky, Aggregating deep convolutional features for image retrieval, *arXiv preprint arXiv:1510.07493* (2015).

[29] G. Tolias, R. Sicre and H. Jégou, Particular object retrieval with integral max-pooling of CNN activations, *arXiv preprint arXiv:1511.05879* (2015).

[30] Y. Kalantidis, C. Mellina and S. Osindero, Cross-dimensional weighting for aggregated deep convolutional features, in: *European conference on computer vision*, Vol. 9913, Springer, 2016, pp. 685–701.

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

[32] K. He, X. Zhang, S. Ren and J. Sun, Deep residual learning for image recognition, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[33] V. Chandrasekhar, J. Lin, Q. Liao, O. Morere, A. Veillard, L. Duan and T. Poggio, Compression of deep neural networks for image instance retrieval, in: *Data Compression Conference*, IEEE, 2017, pp. 300–309.

[34] P. Napoletano, Visual descriptors for content-based retrieval of remote-sensing images, *International journal of remote sensing* **39**(5) (2018), 1343–1376.

[35] F. Radenović, G. Tolias and O. Chum, Fine-tuning CNN image retrieval with no human annotation, *IEEE transactions on pattern analysis and machine intelligence* **41**(7) (2018), 1655–1668.

[36] J. Wan, D. Wang, S.C.H. Hoi, P. Wu, J. Zhu, Y. Zhang and J. Li, Deep learning for content-based image retrieval: A comprehensive study, in: *Proceedings of the 22nd ACM international conference on Multimedia*, ACM, 2014, pp. 157–166.

[37] A. Babenko, A. Slesarev, A. Chigorin and V. Lempitsky, Neural codes for image retrieval, in: *European conference on computer vision*, Springer, 2014, pp. 584–599.

[38] H. Noh, A. Araujo, J. Sim, T. Weyand and B. Han, Large-scale image retrieval with attentive deep local features, in: *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3456–3465.

[39] R. Fu, B. Li, Y. Gao and P. Wang, Content-based image retrieval based on CNN and SVM, in: *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, IEEE, 2016, pp. 638–642.

[40] Z. Cai, W. Gao, Z. Yu, J. Huang and Z. Cai, Feature extraction with triplet convolutional neural network for content-based image retrieval, in: *IEEE Conference on Industrial Electronics and Applications*, IEEE, 2017, pp. 337–342.

[41] S. Pang, J. Ma, J. Xue, J. Zhu and V. Ordonez, Deep Feature Aggregation and Image Re-ranking with Heat Diffusion for Image Retrieval, *IEEE Transactions on Multimedia* **21**(6) (2018), 1513–1523.

[42] J. Xu, C. Wang, C. Qi, C. Shi and B. Xiao, Unsupervised semantic-based aggregation of deep convolutional features, *IEEE Transactions on Image Processing* **28**(2) (2018), 601–611.

[43] T. Hoang, T.-T. Do, D.-K. Le Tan and N.-M. Cheung, Selective deep convolutional features for image retrieval, in: *Proceedings of the 25th ACM international conference on Multimedia*, ACM, 2017, pp. 1600–1608.

[44] T.G. Dietterich et al., Ensemble learning, *The handbook of brain theory and neural networks* **2** (2002), 110–125.

[45] M. Zareapoor, P. Shamsolmoali et al., Application of credit card fraud detection: Based on bagging ensemble classifier, *Procedia computer science* **48**(2015) (2015), 679–685.

[46] S. Sakthithasan, R. Pears, A. Bifet and B. Pfahringer, Use of ensembles of Fourier spectra in capturing recurrent concepts in data streams, in: *2015 international joint conference on neural networks (ijcnn)*, IEEE, 2015, pp. 1–8.

[47] C. Tekin, J. Yoon and M. van der Schaar, Adaptive ensemble learning with confidence bounds for personalized diagnosis, in: *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[48] I. Frías-Blanco, A. Verdecia-Cabrera, A. Ortiz-Díaz and A. Carvalho, Fast adaptive stacking of ensembles, in: *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, ACM, 2016, pp. 929–934.

[49] A. Ortiz, J. Munilla, J.M. Gorriz and J. Ramirez, Ensembles of deep learning architectures for the early diagnosis of the Alzheimer's disease, *International journal of neural systems* **26**(07) (2016), 1650025.

[50] A. Fernández, C.J. Carmona, M.J. del Jesus and F. Herrera, A Pareto Based Ensemble with Feature and Instance Selection for Learning from Multi-Class Imbalanced Datasets, *International Journal of Neural Systems* (2017).

[51] X. Yan, F. He, Y. Zhang and X. Xie, An optimizer ensemble algorithm and its application to image registration, *Integrated Computer-Aided Engineering* (2019), 1–17.

[52] R. Minetto, M. Pamplona Segundo and S. Sarkar, Hydra: An Ensemble of Convolutional Neural Networks for Geospatial Land Classification, *IEEE Transactions on Geoscience and Remote Sensing* **57**(9) (2019), 6530–6541, ISSN 1558-0644. doi:10.1109/tgrs.2019.2906883. `http://dx.doi.org/10.1109/TGRS.2019.2906883`.

[53] H. Li, Y. Li and Y. Zha, Image Retrieval Method based on Multi-View Generating and Ensemble Learning, *International Journal of Performability Engineering* **13**(5) (2017).

[54] S. Bai, Z. Zhou, J. Wang, X. Bai, L. Jan Latecki and Q. Tian, Ensemble diffusion for retrieval, in: *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 774–783.

[55] K. Alam, N. Siddique and Adeli, A dynamic ensemble learning algorithm for neural networks (2019). doi:10.1007/s00521-019-04359-7.

[56] A.H. Ko, R. Sabourin and A.S. Britto Jr, From dynamic classifier selection to dynamic ensemble selection, *Pattern recognition* **41**(5) (2008), 1718–1731.

[57] G. Griffin, A. Holub and P. Perona, Caltech-256 object category dataset (2007).

[58] K. He, X. Zhang, S. Ren and J. Sun, Identity mappings in deep residual networks, in: *European conference on computer vision*, Springer, 2016, pp. 630–645.

[59] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, Going deeper with convolutions, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[60] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, Rethinking the inception architecture for computer vision, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[61] C. Szegedy, S. Ioffe, V. Vanhoucke and A.A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[62] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[63] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, *arXiv preprint arXiv:1704.04861* (2017).

[64] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L.-C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[65] G. Huang, Z. Liu, L. Van Der Maaten and K.Q. Weinberger, Densely connected convolutional networks, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[66] B. Zoph, V. Vasudevan, J. Shlens and Q.V. Le, Learning transferable architectures for scalable image recognition, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.

[67] A. Krizhevsky, G. Hinton et al., Learning multiple layers of features from tiny images, Technical Report, Citeseer, 2009.

[68] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg and L. Fei-Fei, ImageNet Large Scale Visual Recognition Challenge, *International Journal of Computer Vision (IJCV)* **115**(3) (2015), 211–252. doi:10.1007/s11263-015-0816-y.

[69] G. Chechik, V. Sharma, U. Shalit and S. Bengio, Large scale online learning of image similarity through ranking, *Journal of Machine Learning Research* **11**(Mar) (2010), 1109–1135.

[70] J.R. Quinlan et al., Bagging, boosting, and C4. 5, in: *AAAI/IAAI*, Vol. 1, 1996, pp. 725–730.

[71] M. Ahmadlou and H. Adeli, Enhanced probabilistic neural network with local decision circles: A robust classifier, *Integrated Computer-Aided Engineering* **17**(3) (2010), 197–210.

[72] M.H. Rafiei and H. Adeli, A new neural dynamic classification algorithm, *IEEE transactions on neural networks and learning systems* **28**(12) (2017), 3074–3083.