applied optics

Categorization of digitized artworks by media with brain programming

Mariana Chan-Ley and Gustavo Olague*

[1]CICESE Research Center, EvoVisión Laboratory, Ensenada, Carretera Ensenada-Tijuana 3918, Zona Playitas, Ensenada B.C., 22860, Mexico
*Corresponding author: olague@cicese.mx

Received 12 December 2019; revised 13 April 2020; accepted 15 April 2020; posted 15 April 2020 (Doc. ID 385552); published 8 May 2020

This work describes the use of brain programming applied to the categorization problem of art media. The art categorization problem—from the standpoint of materials and techniques used by artists—presents a challenging task and is considered an open research area. Brain programming is a machine learning methodology successfully tested for the problem of object categorization; however, when working with art images, the objects in pictures of the same category may be different from each other regarding image content. Therefore, it is necessary to find the best set of functions that extract specific features to identify patterns among different techniques. In this study, we show a comparison with deep learning to understand the limits and benefits of our approach. We train and validate solutions with the Kaggle database and test the best results with the WikiArt database. The results confirm that brain programming matches or surpasses deep learning in three out of five classes (over 90%) while being close (less than 5%) in the remaining two with significantly simpler programs.

https://doi.org/10.1364/AO.385552

1. INTRODUCTION

The growing development of technologies has meant that almost all aspects of our lives are digitized. The case of art is no exception. Therefore, with the increasing volumes of art databases on the internet comes the daunting task of organization and retrieval of digitized artworks. Indeed, the topic of optics for arts is of high interest in the optical society [1,2]. The task of classifying pieces of fine art is hugely complex. Object categorization or generic object recognition seeks to model and recognize objects based on their coarse, prototypical shape [3]. Nowadays, the problem of object abstraction is arguably the most important and most challenging problem facing researchers in object categorization, and recognition of digitized artworks is no an exception. When examining an artwork, an art expert can usually determine its style, genre, and media.

A style of artwork refers to its distinctive visual elements, techniques, and methods. It usually corresponds to an art movement or a school (group) to which the author belongs. Recognizing art style is not necessarily correlated with the subject matter—in other words, the correspondence of existing specific objects in the artwork. Style is related mainly to the form and can be associated with features at different levels: low, medium, and high. A genre system divides artworks according to depicted themes and objects. The classical hierarchy of genres was developed in European culture by the 17th century. It ranked genres in high (history painting and portrait) and low (genre painting, landscape, and still life). This hierarchy was based on the notion of man as the measure of all things. Landscape and still life were the lowest because they did not involve human subject matter. History was highest because it deals with the noblest events of humanity. The genre system is not so relevant for a contemporary. Regarding these two aspects (style and genre), people dealing with the automated classification problem use style to refer to both terms. Indeed, research groups proposed several approaches that attempt to solve the problem of art style classification.

For example, in Ref. [4], we found one of the first approaches of image processing for artist identification on a dataset of 101 high-resolution grayscale scans of paintings with encouraging but not perfect results. The work reported in Ref. [5] deals with the problem of automatically classifying digital pictures of paintings gathered across internet sources rather than high-resolution data. The database consists of 353 paintings, belonging to five classes: Abstract, Impressionism, Cubism, Pop Art, and Realism. According to the authors, there is still much space for improvement in the proposed technique.

Kowaliw *et al*. proposed genetic programming (GP) to discover a useful collection of features for the description of individual artists' styles [6]. They introduce a database of comics and graphic novels, which is a collection of grayscale panel drawings (images of 200 × 200 pixels or less) organized by the artist, emphasizing aesthetic style. There are 150 training images and at least 80 validation images for each class, with 240 validation images in the control group. They attained high-accurate results (above 90%) regarding artists in the database, as well as randomly selected from search engine results. Some drawbacks are meaningful comparison with standard databases with more images and significantly harder information content.

In Ref. [7], the authors make a comparative study of different classification methodologies (based on computer vision techniques) for the classification of fine-art painting style. The dataset contains 70 paintings per class (Baroque, Abstract, Renaissance, Pop Art, Expressionism, Impressionism, Cubism), and the experiments propose five-fold cross-validation using 20% of images, with overall results around 50% for the seven classes. Condorovici *et al*. present an automatic system for the classification of digital representations of paintings [8]. The paper investigates seven classifiers (logistic regression, multilayer perceptron, sequential minimal optimization, bagging, logiboost, decision table, random forest) evaluated on a database containing more than 3400 paintings from six different classes (Renaissance, Baroque, Rococo, Romanticism, Impressionism, Cubism), from more than 600 authors. The experiments assess all possible combinations with a significant difference between the best (around 90%) and worst (around 60%) classification rate. These results correlated well if paintings are similar (Baroque, Rococo, Romanticism), while performance degrades if paintings are more distinct (Cubism, Renaissance, Romanticism).

In Ref. [9], the authors introduce a deep neural network for the task of identifying artistic styles in paintings. They suggest a compact binary representation combined with the PiCoDes descriptors, showing classification results on a large-scale collection of paintings. They use a convolutional neural network (CNN) trained on ImageNet using 4096 activations of the first fully connected layer and 9216 activations on the last convolutional layer with a final output of 1000 predictions. The authors use the WikiArt dataset, which is a collection of the visual art encyclopedia [10]. The collection is a complete and well-structured online repository of fine art with 40,724 unique digitized paintings with variable resolution. Style recognition differs from the task of object recognition, since two paintings can describe the same scene using very different artistic techniques. The results exhibit that there is still considerable room for improvement.

Saleh *et al.* [11] focus on the problem of automating the discovery of artistic influence. The proposed database contains a total of 1710 images of artworks by 66 artists chosen from Mark Harden's Artchive database of fine art [12]. Artists and curators use different concepts to describe art pieces. Some of them are basic features such as color, texture, form, and shape, but also more abstract elements such as movement, harmony, balance, proportion, and patterns. Even some physical attributes such as brush strokes or objects in the scene are used to categorize art images. Learning and judging such complex visual concepts is an impressive ability of human perception [13]. The research of Saleh *et al.* attempts to answer the question of finding influence between painters as a knowledge discovery problem and shows exciting results.

In Ref. [14], the authors propose a method of classifying painting styles by extracting various global features using color-based statistical computation and composition-based local features extracted through the segmentation of objects within the paintings. Based on extracted features, paintings are categorized by style using a self-organizing map. The database contains 1633 pieces of artwork painted by 19 painters collected from [15]. The authors follow a binary classification method of pairing four styles (Expressionism, Impressionism, Post-Impressionism, Surrealism). From the collected data, randomly selected 50% for each painter was used for training, and the remaining 50% was used for testing. Although there are differences related to the pairing of classes, the average train precision is about 93%, and the results agree with the test data.

Florea *et al.* address the problem of automatically recognizing artistic movement in digitized paintings [16]. They follow a computer vision approach, similar to previous works, composed of feature extraction [local binary patterns, histogram of oriented gradients (HOG), Gabor filters, scale-invariant feature transform (SIFT), colorHOG, colorSIFT, histogram of topography (HoT), and color names]. As a classifier, the authors use a boosted support vector machine (SVM). They also introduce a sizeable digitized painting database with annotations extracted mostly from WikiArt, with about 25% collected from other sources. They altered the original images by removing the painting framework and eliminating images of sculptures or 3D objects. Finally, an expert removed all images that were not considered as art. The final database consists of 18,040 images and 18 different movements. The authors evaluated the proposed system on the WikiArt database, and the overall accuracy was 82.41%, and even manages to outperform modern deep learning (DL) frameworks.

In Ref. [17], they address the problem using CNNs. In their study, they could see that the network grouped the characteristics according to specific guidelines; still, the best performance they obtained was only 62%, although they highlight the potential role of artificial intelligence and machine learning techniques to discover patterns and trends in the domain of art history.

In summary, we can observe from the reviewed works that the solutions to the classification problem have evolved in more complex systems, using more and more images, with methods based on DL, although there are no standards yet.

The organization of this paper is as follows. Section 2 presents the art media classification/categorization problem from the standpoint of GP. In Section 3, we outline the proposed methodology giving emphasis to the discovery of symbolic programs. Next, we present the experimental results divided into two main sections: first, training and validation with the Kaggle database, and second, testing with the WikiArt database. Finally, we provide a discussion and our conclusions.

2. ART MEDIA CATEGORIZATION PROBLEM

Art media refer to materials the artwork is made from, and to techniques used by the artist to create that artwork. Many modern works are made from a variety of materials by diverse techniques, and the term mixed media has had to be coined to take account of this. Lyu *et al.* [18] describe a computational technique for authenticating works of art, specifically paintings and drawings, from high-resolution digital scans of the original works. The approach builds a statistical model (based on first-and-higher-order wavelet statistics) of an artist from the scans of a set of authenticated works against which new works are then compared. Preliminary results based on 13 drawings and further tests to a painting confirm expert authentications.

As we review in previous work, it is possible to annotate art with a vast amount of meta-data for classification. Nevertheless, generating the meta-data by hand is time consuming and expensive; that is the reason people venture to create methods to categorize images through information extracted directly from pictures [19].

Recent advances in computer vision show the advantage of "learning" the features from data instead of engineering such features. However, such methods do not serve to mark the specific characteristics within an image. Neural networks operate as a black box with the drawback that after several layers of repetitive operations, we do not know what happens inside, nor how/why it chooses the features.

In Ref. [20], the authors successfully apply GP to extract formulas that describe vegetation coverage indices. This work is the first approach where some features are extracted from images to propose a model of functions to avoid classification, replacing it with the correlation of physical attributes.

Here, we use brain programming (BP) to discover a collection of functions for the description of different classes of art images. BP is a GP-like methodology selected as the proposed optimization approach, in which the main goal is the discovery of a set of evolutionary visual operators (EVOs) embedded within a hierarchical structure called the artificial visual cortex (AVC), explained in Section 3.B. This proposal extracts information from images and produces output functions to describe the studied art class. More formally, we describe the problem as follows.

In order to proceed with the analysis, we will introduce the problem of GP from the standpoint of data modeling, similar to the previous explanation, with some technicalities added that are necessary to understand the idea better. In general, a minimization problem requires to find a solution $\boldsymbol{P}_{\min} \in S$ such that $f(\boldsymbol{P}_{\min})$ is a global minimum on S. More specifically, it is required to find a $\boldsymbol{P}_{\min} \in S$ such that

$$\forall \boldsymbol{P} \in S : f(\boldsymbol{P}_{\min}) \leq f(\boldsymbol{P}).$$

Contrary to conventional approaches regarding the mere finding of best-fit parameters in GP, we would like to find a function that satisfies the task of fitting data. This process includes several stages, since a direct mapping between the domain and codomain are unknown or at least not well defined. In this way, the solution to the image classification problem requires to define the following problem:

$$\boldsymbol{y} = \min(f(\boldsymbol{x}, \boldsymbol{F}, \boldsymbol{T}, \boldsymbol{a})), \tag{1}$$

where the dataset is given by $(\boldsymbol{y}, \boldsymbol{x})$, $\boldsymbol{F}$ represents the set of functions, $\boldsymbol{T}$ defines the terminal set, and $\boldsymbol{a}$ are the parameters controlling the algorithm. Hence, two points need to be defined before attempting to solve the problem: (1) the method of feature extraction and (2) a suitable criterion $\boldsymbol{Q}$ for the minimization.

The AVC is the algorithm in charge of feature extraction from the input images. BP is the algorithm in charge of tuning $(\boldsymbol{F}, \boldsymbol{T}, \boldsymbol{a})$ for each visual operator embedded into the AVC. In the present work, the criterion is a classifier, implemented with an SVM. Therefore, an SVM is trained to learn a mapping $f(\mathbf{x})$ that associates descriptors $\mathbf{x}_i$ to labels y_i. We formulate our problem in terms of a binary classification task, whose main aim is to find a decision surface that best separates the elements of the class. In this work, we use a nonlinear SVM working with the discriminant hyperplane defined by

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b, \tag{2}$$

where the given training data are $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, l$, $y_i \in \{-1, 1\}$, and $\mathbf{x}_i \in \mathbf{R}^p$, and $K(\mathbf{x}_i, \mathbf{x})$ is the kernel function. The sign of the output indicates the class membership of $\mathbf{x}$. Thus, finding the best hyperplane is performed through an optimization process that uses the margin between the class and non-class as the search criteria.

In summary, we can say that the minimization problem works on the learning pentuple $((\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{F}, \boldsymbol{T}, \boldsymbol{a}, \boldsymbol{Q})$. In this work, we will focus on the problem of associating the domain given by the descriptors and the codomain given by the labels.

A. Dataset

BP has successfully been proved for classifying objects of the same class in [21]. To challenge this methodology, we move to a different problem that represents a compromise between GRAZ and ImageNet databases of object recognition. We download the digitized artwork from the Kaggle website [22], and Fig. 1 shows sample images for each class. This database provides a set of images divided into five categories of art media: drawing, painting, iconography, engraving, and sculpture. Art media categorization presents a big challenge since it departs from classical object recognition, where we usually try to recognize specific kinds of objects. Here, all images have something familiar from the standpoint of image content or composition—an object with similar characteristics—when classifying art media images. There could be almost anything in the picture, but similarities between classes will be materials and techniques,

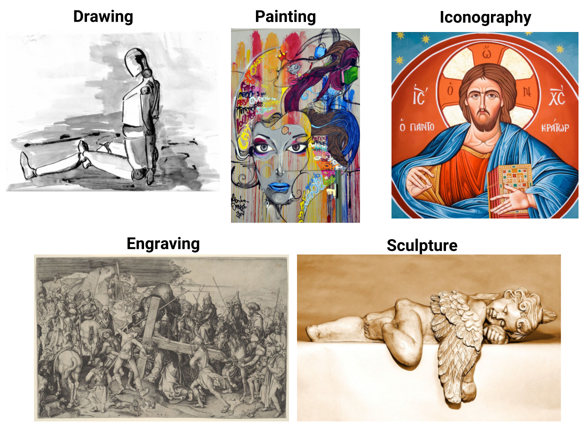


Fig. 1. These images were downloaded from the Pxhere site to illustrate the five art classes on the Kaggle dataset. This site provides free copyright images.

reflecting on style, color, and others. We seek to discover these similarities using the BP paradigm.

3. BRAIN PROGRAMMING

BP is an evolutionary paradigm that aims to emulate the behavior of the brain based on neuroscience knowledge for different vision problems, and adjusted according to the task we are going to perform. The first works introducing the technique [21,23] were focused on problems of automating the design of visual attention (VA) models to outperform previous human-made systems developed by VA experts. VA is a skill performed by the brain, and its functionality is to perceive salient visual features. The automatic design of the acquisition and integration steps of the artificial dorsal stream was engineered to emulate the selectivity and goal-driven behavior of the natural dorsal stream in the brain; this is a useful skill for the task of tracking objects from a video captured with a moving camera, as shown in Ref. [24].

The primary purpose of BP is to optimize complex models by adjusting the operations within them. Figure 2 depicts the entire process of this methodology.

BP is a long process consisting of several stages summarized in two key ideas; nevertheless, these ideas are not independent of each other. First, BP is an evolutionary process whose primary purpose is to discover some functions to optimize complex models by adjusting the operations within them; these functions are embedded and applied to a hierarchical model that best extracts the features. The second part is the hierarchical structure inspired by the human visual cortex that uses the concept of composition of functions to extract features from images. Depending on the task, we can change this model, between the focus of attention for saliency problems [23], or the complete AVC for categorization/classification problems. In this study, we are using the AVC, explained in detail in Section 3.B.

The final result of this study is the design of optimal object recognition programs that satisfies the object recognition tasks.

A. Initialization

BP, like any evolutionary process, begins with a randomized initial generation. It begins by creating the set of initialization variables to define the evolutionary process, e.g., population size, size of solutions or individuals, and crossing-mutation probabilities. In BP, an individual represents a computer program written with a set of syntactic trees embedded into hierarchical structures. In this work, individuals within the population contain four kinds of functions, one for each visual operator (VO). These functions are defined with expert knowledge to create trees whose nodes are selected from a previously defined pool of functions and terminals and updated through genetic operations. The list of functions and terminals used in this work for each VO or visual map (VM) are in Table 1. The table includes arithmetic functions between two images A and B, transcendental functions over an image, as well as square function, square root function, image complement, color opponencies (red-green and blue-yellow), dynamic threshold function, and arithmetic functions between an image A and a constant k. The table also includes round, half, floor, and ceil functions over an image A; dilation and erosion operators with the disk, square, and diamond structure element (SE); skeleton operator over the image A; find the perimeter of objects in the image A; and hit or miss transformation with the disk, square, and diamond structures. Also, we include morphological top-hat and bottom-hat filtering over the image A, opening and closing morphological operator on A, absolute value applied to A, and the addition and subtraction operators. Finally, we add the infimum and supremum functions between images A and B, the convolution of the image A, and a Gaussian filter with $\sigma = 1$ or 2, derivative of the image A along directions x and y. We also include the representation of an individual at the top of Fig. 2.

Table 1. Functions and Terminals for the EVO

Functions for EVO$_O$	**Terminals for EVO$_O$**
$A+B, A-B, A\times B, A/B, \lvert A\rvert, \lvert A+B\rvert, \lvert A-B\rvert, \log(A), (A)^2, \sqrt{A}, k+A, k-A, k\times A, A/k$, round$(A), \lfloor A\rfloor, \lceil A\rceil$, inf$(A, B)$, sup$(A, B)$, $G_{\sigma=1}(A), G_{\sigma=2}(A), D_x(A), D_y(A)$, thr$(A)$	$I_r, I_g, I_b, I_c, I_m, I_y, I_k, I_h, I_s, I_v, D_x(I_x), D_{xx}(I_x), D_y(I_x), D_{yy}(I_x), D_{xy}(I_x)$
Functions for EVO$_C$	Terminals for EVO$_C$
$A+B, A-B, A\times B, A/B$, $\log(A), \exp(A), (A)^2, \sqrt{A}, (A)^c$, thr$(A)$	$I_r, I_g, I_b, I_c, I_m, I_y, I_k, I_h, I_s, I_v, Op_{r-g}(I), Op_{b-y}(I)$
Functions for EVO$_S$	Terminals for EVO$_S$
$A+B, A-B, A\times B, A/B, k+A, k-A, k\times A, A/k$, round$(A), \lfloor A\rfloor, \lceil A\rceil, A\oplus SE_d, A\oplus SE_s, A\oplus SE_{dm}, A\ominus SE_d, A\ominus SE_s, A\ominus SE_{dm}, Sk(A)$, Perim$(A), A\circledast SE_d, A\circledast SE_s, A\circledast SE_{dm}, T_{\text{hat}}(A), B_{\text{hat}}(A), A\circledcirc SE_s, A\odot SE_s$, thr$(A)$	$I_r, I_g, I_b, I_c, I_m, I_y, I_k, I_h, I_s, I_v$
Functions for EVO$_{MM}$	Terminals for EVO$_{MM}$
$A+B, A-B, A\times B, A/B, \lvert A+B\rvert, \lvert A-B\rvert, \log(A), (A)^2, \sqrt{A}, G_{\sigma=1}(A), G_{\sigma=2}(A), D_x(A), D_y(A)$	$\mathrm{MC_d}, D_x(\mathrm{MC_d}), D_{xx}(\mathrm{MC_d}), D_y(\mathrm{MC_d}), D_{yy}(\mathrm{MC_d}), D_{xy}(\mathrm{MC_d})$

1. *Individual Representation*

The representation of individuals consists of a set of functions for each VO defined in Section 3.B, and encoded in a multi-tree representation, in addition to implementing evolutionary operations of crossover and mutation corresponding to this representation.

Each individual has a variable number of syntactic trees, ranging from four to 12, one for each EVO (EVO$_O$, EVO$_C$, EVO$_S$) regarding orientation, color, and shape; and at least one tree to merge the VMs produced after the center-surround process to generate the mental maps (MMs). Details about the usage of these VOs will be explained in detail in Section 3.B.1. Figure 2 provides a graphic representation of the complete BP workflow that creates symbolic solutions (individuals) to computer vision problems.

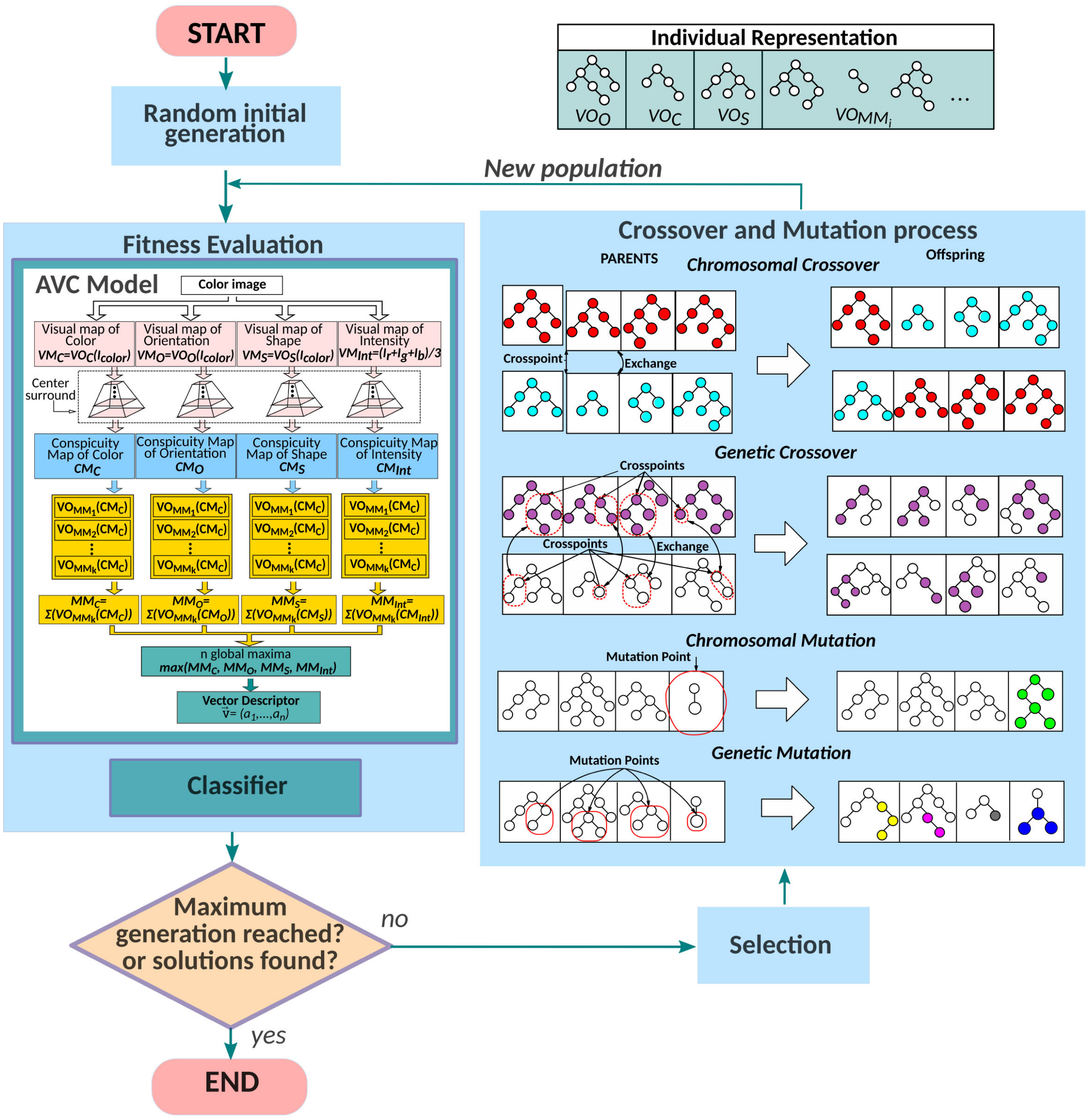


Fig. 2. Complete brain programming workflow.

After creating the first generation, we proceed to evaluate the fitness of the individuals in the population. As mention before, we are dealing with a categorization problem, so the best alternative is to model an AVC.

B. Fitness Evaluation: Artificial Virtual Cortex

Conventional evolutionary algorithms usually apply a fitness function previously defined to evaluate the quality of individuals. In BP, the evaluation consists of using the EVOs generated before to extract features from input images through the AVC hierarchical structure depicted in Fig. 2. We explain the detailed steps below.

The AVC models some aspects of the human visual cortex; each layer of the visual cortex represents a function through a set of mathematical operations, which represents a virtual bundle. It selects the visual features of the image in order to construct an abstract representation of the object of interest. Therefore, the system looks for salient points (at different dimensions) in the image to construct an image descriptor, which we use in the classification process.

The AVC consists of two main stages: the first acquires and transforms features that describe the object, while in the second stage, a descriptor encodes the object, which is classified based on the results of the first stage.

1. *Acquisition and Transformation of Features*

This stage follows the psychologic model of VA, proposed by Treisman and Gelade, called feature integration theory (FIT). According to this theory, different parts of the brain automatically separate basic features found in the visual field such as color, orientation, and shape. The entrance to our system is an RGB image that belongs to a predefined class. An image I is defined as the graphic of a function as follows [21].

Definition 1. Image as the graph of a function. Let f be a function $f: U \subset \mathbb{R}^2 \to \mathbb{R}$. The graph or image I of f is the subset of $\mathbb{R}^3$ that consists of the points $(x, y, f(x, y))$, in which the ordered pair (x, y) is a point in U, and $f(x, y)$ is the value at that point. Symbolically, the image $I = \{(x, y, f(x, y)) \in \mathbb{R}^3 | (x, y) \in U\}$.

This definition describes the fact that images are variations in the intensity of light along the two-dimensional plane of camera sensors. Regarding visual processing for feature extraction of the input image, multiple color channels are considered in order to build the set $I_{\text{color}} = \{I_r, I_g, I_b, I_c, I_m, I_y, I_k, I_h, I_s, I_v\}$, where each element corresponds to the color components of the RGB (red, green, blue), HSV (hue, saturation, and value) and CMYK (cyan, magenta, yellow, and black) color spaces. We define the optimization process through the formulation of an appropriate search space and evaluation functions, as well as the overall design of our evolutionary algorithm.

2. *Feature Dimensions*

The next step in the process is the decomposition of the image into relevant characteristics. Three EVOs transform the input images I_{color} through each VO defined as $\text{EVO}_d : I_{\text{color}} \to \text{VM}_d$ and applied independently to emphasize specific characteristics of the object. Note that VM_{Int} is not evolved, and it is calculated with the average of the RGB color bands. These EVOs are operators generated in Section 3.A. Individuals represent possible configurations for feature extraction that describe input images and are optimized using the evolutionary process. These transformations are performed to recreate the process of extracting information from the FIT. When applying each operator, a VM generated for each dimension represents a partial procedure within the overall process. Each VM is a topographic map that represents, in some way, an elementary characteristic of the image. The dimensions used in the AVC model are color, orientation, shape, and intensity.

3. *Center-Surround Process*

Once we obtain the VMs, a center-surround process is applied. At this stage, we extract features that are invariant to scale along the color bands. The result of this process is a conspicuity map (CM) and is calculated as the difference between different scales, which are obtained through a pyramid of nine levels $P_d^\sigma = \{P_d^{\sigma=0}, P_d^{\sigma=1}, P_d^{\sigma=2}, \ldots, P_d^{\sigma=8}\}$, where $d \in \{C, O, S, \text{Int}\}$ refers to color, orientation, shape, and intensity at each dimension. Each pyramid is calculated using a Gaussian smoothing filter on each VM, resulting in an image half the size of the input map. This process is repeated recursively eight times to obtain the nine levels of the pyramid. In the second step, we calculate the differences between each level of the pyramid P_d^σ using Eq. (3):

$$Q_d^j = P_d^{\sigma=\left\lfloor \frac{j+9}{2} \right\rfloor+1} - P_d^{\sigma=\left\lfloor \frac{j+2}{2} \right\rfloor+1}, \quad \textbf{(3)}$$

where $j = 1, 2, \ldots, 6$. Since the levels of P_d^σ have different sizes, each level is normalized and scaled to the dimension of the visual map VM_d using polynomial interpolation. The six levels generated are combined into a single map with a summation, generating a CM, for each dimension.

C. Description and Classification

The next step of the AVC is to synthesize all information obtained in a vector descriptor of the image, which will serve as input to a decision system, which in this case consists of an SVM.

1. *Computation of Mental Maps*

The objective at this stage is to build a map that discriminates the unwanted information from the CMs by focusing only on the classified object, thus highlighting the most salient features. This new map is the MM.

A set of MMs is applied to each CM generating a map for each dimension, using Eq. (4), where d is the dimension, and k represents the cardinality of the set VO_{MM}. We define these VOs through syntactic trees to create the individual that we evaluate later. The MMs correspond to the list from the fourth tree onwards. Table 1 shows the functions and terminals used to build the VO_{MM_k} operators:

$$MM_d = \sum_{i=1}^{k} \text{VO}_{MM_i}(CM_d). \quad \textbf{(4)}$$

Once we obtain the four mental maps and concatenate with the rest of syntactic trees, we apply the resulting program to each image on the database keeping the n highest values to define the descriptor vector $\vec{v}$ of the image.

2. *Label Assignment*

In Fig. 2, we can see the workflow of this process. The next step consists of training a decision system. For this research, after we gather all feature vectors from the image dataset, we learn an SVM. This classifier is trained to create a model $f(\mathbf{x})$ that maps a set of vectors $\mathbf{x}_i$, which in this case are the descriptor vectors, to a set of labels y_i. Therefore, the goal is to achieve solutions to Eqs. (1) and (2).

D. Selection and Reproduction

The selection function chooses parents to produce offspring for the next generation based on their scaled values from the fitness function. An individual can be selected more than once as a parent, in which case it contributes its genes to more than one child.

1. Genetic Operations

When considering a new individual representation, it is also necessary to propose new genetic operations. In our case, regular crossover and mutation operators are insufficient to solve this problem, we consider that the whole individual is similar to a chromosome, and each VO within the chromosome is a gene code; therefore, we apply four genetic operators:

- Chromosome level crossover. The algorithm randomly selects a crossing point within a parent's chromosome. The crossing point does not need to be the same in the second parent; then, the process builds a new offspring by the union of the left section resulting from the crossing point of the first parent and the right section of the crossing point in the second parent.
- Gene level crossover. Two VOs, chosen randomly, and for each function of both trees (genes), the system chooses a crossing point randomly, then the sub-trees under the crossing point are exchanged to generate two new VOs. Therefore, this operation creates two new children-chromosomes.
- Chromosome level mutation. The algorithm randomly selects a mutation point within a parent's chromosome by replacing the operator completely with a randomly generated operator.
- Gene level mutation. Within a VO, randomly chosen, the algorithm selects a node, and the mutation operation randomly alters the sub-tree that results below this point.

We show a graphic representation of this process in Fig. 2. Once we generate our new population, the evolutionary process continues, and we proceed to evaluate the new offspring.

E. Stop Criteria

For this approach, we use two different conditions to determine when to stop. The algorithm stops when the number of generations reaches a specific value. The algorithm could stop when the fitness has reached an optimal value—in other words, when the system classifies all training images correctly.

4. EXPERIMENTS

A. Training and Validation with Kaggle Database

The methodology for this problem follows the usual absent/present protocol as previous works, considering two image sets for learning and validation. The implementation is compiled in MATLAB running on a Dell Precision workstation with Linux OpenSuse Leap 15.0 OS. We use a population size of 30 individuals per function. In the same way, the programs run for only 30 generations. Such parameters follow the experiments made in previous works [21,23–25]. Note that we use atypical GP parameters, and this is due to the analysis of the visual problem and structure of the hierarchical model.

Our method spends 450 h or 18.75 days for the class drawing, 544 h or 22.66 days for the class engraving, 597 h or 24.87 days for the class painting, 810 h or 33.75 days for the class iconography, and 562 h or 23.41 days for the class sculpture. We use 10 workstations to speed the process of optimization. Table 2 shows a summary of statistical results after running 15 times our BP strategy per class, while Table 3 gives the size of the image sets used during training and validation. According to the reviewed works, we include the results of two deep learning methods for comparison: from scratch CNN and AlexNet [26]. The table shows that the best program for the class painting practically solves the problem, and its accuracy is better than AlexNet. It misses only about 40 images from the whole dataset. Regarding the class drawing, our program also surpasses AlexNet, while for sculpture and iconography, our best programs achieve around 90% and are slightly lower than AlexNet. Engraving is the class with the lowest score, and we take a closer look in Section 5. However, before this, we present a comparison with a popular database.

B. Testing with WikiArt Database

To corroborate our proposal, we decided to make a comparison using a standard database. We take as reference the WikiArt database [10]. This database contains approximately 250,000 images of works of art, made by more than 3000 artists. WikiArt arranges art images in different categories, either by style, genre, media, or by the artist, and within each one, there are many subcategories. Table 4 provides a summary of results and a comparison with AlexNet, while Table 5 gives the size of each dataset. Note that the ranking remains unchanged (between BP and AlexNet) despite the change. For the experiments described in this section, we select the best solutions (trained with the Kaggle database) discovered in the previous part, and test each of them with WikiArt. Nevertheless, to carry out our study with the WikiArt images, there were not the same classes that we had already studied, so we had to make a manual selection of the images corresponding to each of the categories. Figure 3 shows some samples of selected images from WikiArt.

Drawing was the class that represented a significant challenge to manually find the correspondences, since the Kaggle database includes images of watercolors and various handmade drawings. Thus, we managed to select from the category "caricature" of

Table 2. Summary of Results after Evolving the Best Set of Solutions with BP to the Categorization Problem of Digitized Artworks by Media[a]

	Drawings	Engraving	Painting	Iconography	Sculpture
Evolutionary process	80.48 ± 3.48	79.19 ± 3.19	91.40 ± 3.26	86.34 ± 2.02	84.70 ± 1.93
Best BP	86.01	83.46	98.24	89.37	89.37
Improved BP	**91.32**	84.51	**99.02**	91.18	89.73
From scratch CNN	76.18 ± 2.38	79.16 ± 4.88	91.78 ± 2.43	91.49 ± 0.51	81.14 ± 2.51
AlexNet	89.34 ± 0.89	92.50 ± 1.98	97.34 ± 0.42	96.46 ± 0.36	93.81 ± 0.91

[a]We include the results of deep neural networks for comparison.

Table 3. Total Number of Images per Class Obtained from the Kaggle Dataset[a]

	Drawings	Engraving	Painting	Iconography	Sculpture
Training	554	378	1021	1038	868
Validation	554	378	1021	1038	868

[a]For all experiments, we use 233 images from the background dataset of Caltech-101.

Table 4. Summary of Results Obtained after Testing the Best Individuals with Selected WikiArt Images per Class[a]

	Drawings	Engraving	Painting	Iconography	Sculpture
BP	96.08	87.28	100	92.21	85.34
AlexNet	82.61	92.24	91.82	97.52	89.74

[a]We include the results of deep neural networks for comparison.

Table 5. Total Number of Images per Class Obtained from the WikiArt Dataset[a]

	Drawings	Engraving	Painting	Iconography	Sculpture
Testing WikiArt	204	695	2089	251	117
WikiArt land-scapes			136		

[a]For all experiments, we use 233 images from the background dataset of Caltech-101.

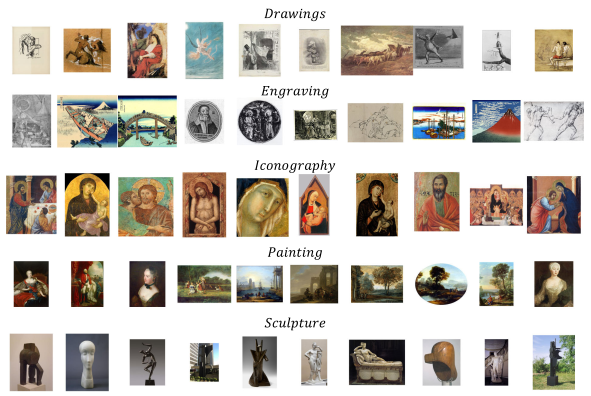


Fig. 3. These images were downloaded from the WikiArt site to test the best programs previously trained on the Kaggle dataset.

WikiArt, a set of 204 images similar to those of our problem resulting in a performance of 96.08%. We continued with the engraving database, 695 images of engravings in grayscale selected from WikiArt, all recorded by Albrecht Durer, and classified in the Northern Renaissance category. The discovered program achieves 87.28%, which was similar to the score achieved during validation. As for the images to test the class of painting, the complete set of 2089 images of the Rococo style was used due to their similarity with the images that had been used to train our program; in addition, a test was also made with the set of images of the category landscapes, in particular, the works of authors Claude Lorrain, Jan Dirksz Both, and Jan Van Goyen. Both tests scored 100%. For the iconography class, 251 works of art created by Ambrogio Lorenzetti, Cennino Cennini, Cimabue, Duccio, Luca di Tommè, Paolo Veneziano, Pietro Lorenzetti, and Simone Martini were used as a testbed, and our discovered best program achieved 92.21%. Despite WikiArt being dedicated mostly to paintings, the database includes some sculptures, from which we selected 117 images of works of art made by Antonio Canova, Constantin Brâncuşi, Donatello, Alexander Archipenko, Umberto Boccioni, John Chamberlain, and Louise Nevelson that we used to test the best program, which scored 85.34%.

5. DISCUSSION

In general, the use of random principles is overused in evolutionary computation. The idea is to adapt the procedure to avoid the unnecessary application of arbitrary or unplanned solutions within the algorithm to advance towards a more goal-oriented methodology. The idea first proposed in Ref. [25] is called hands-on evolution, where we use the best solutions discovered during the previous execution of the algorithm as the initial population for a new set of experiments of BP. Table 2 provides preliminary results achieved for the five classes using the hands-on method. We observe that the class drawing improved, surpassing AlexNet. Also, the strategy for the classes iconography and sculpture was slightly improved. Nevertheless, there was still one class remaining, which had not shown improvement, so we decided to give the dataset a closer look. Engraving represents a category of fine art or graphic art that usually impresses on a flat surface through the practice of incising a design onto a hard, usually flat surface by cutting grooves into it with a burin.

For the class engraving, on the Kaggle database, there were two different kinds of engravings (Fig. 4); most of them were engravings with only one color defining the art piece. Edges and shadows were made using different sizes of lines or incisions. The other style was Japanese engravings, which introduce color to the images, and the program could not find a generalization method to resolve both tasks. We approach the problem of dividing both sets of images as follows. Given an input image in RGB (color image), the goal is to know if the image is in grayscale, whether there are shades of the same color, such as the left image in Fig. 4, or when the bands represent different

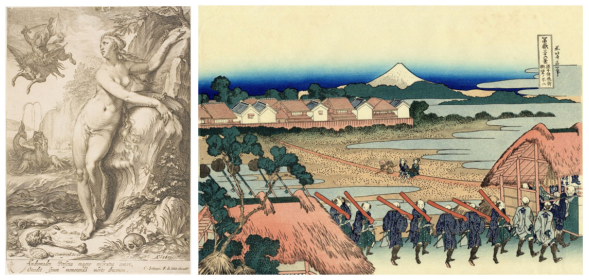
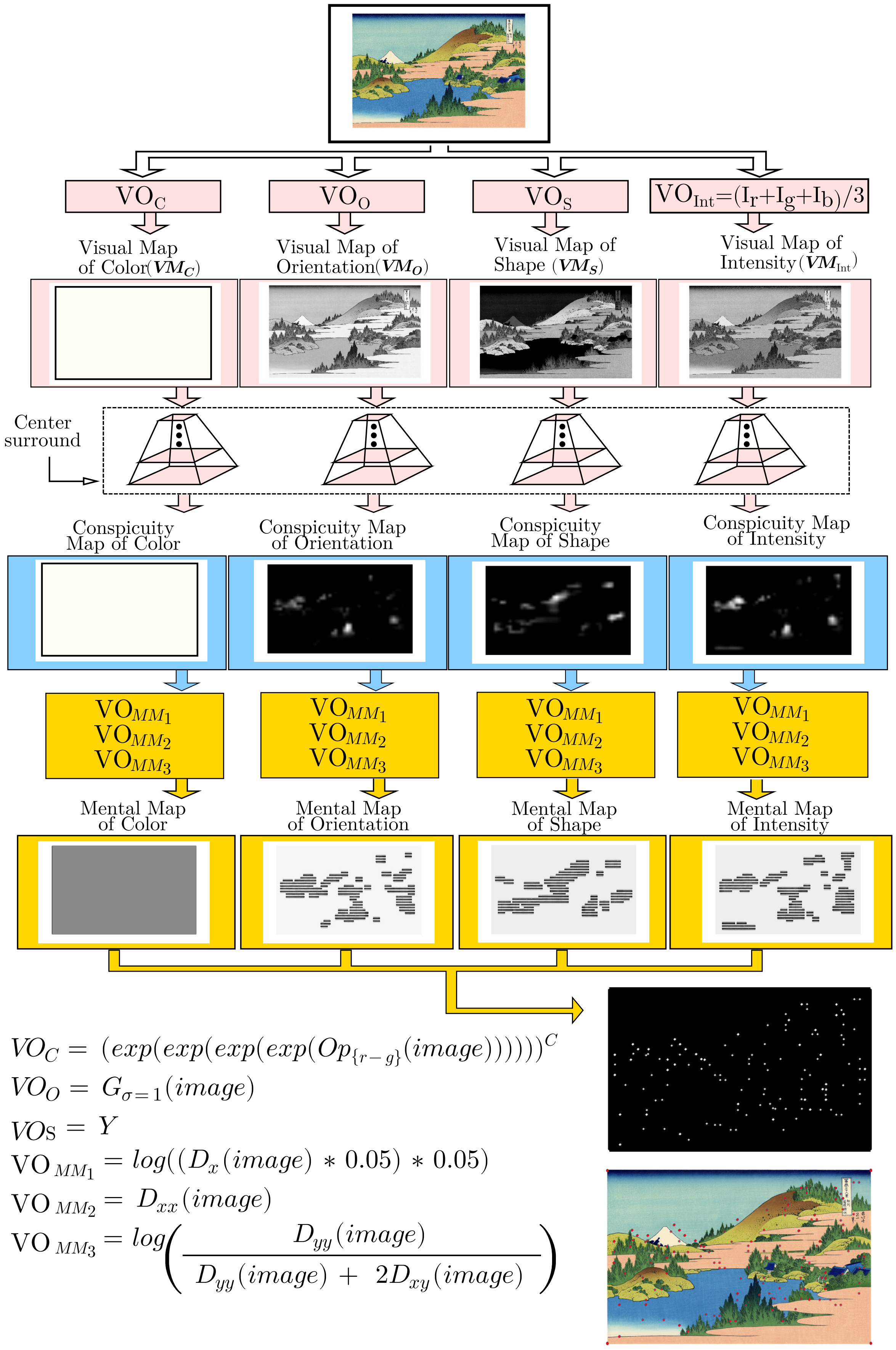


Fig. 4. This figure shows two different images from the same category, engraving. Note the differences between both images and the challenge it represents to a classification system.


VO_C
VO_O
VO_S
VO_Int=(I_r+I_g+I_b)/3
Visual Map of Color(VM_C)
Visual Map of Orientation(VM_O)
Visual Map of Shape (VM_S)
Visual Map of Intensity (VM_Int)
Center surround
Conspicuity Map of Color
Conspicuity Map of Orientation
Conspicuity Map of Shape
Conspicuity Map of Intensity
VO_MM1 VO_MM2 VO_MM3
Mental Map of Color
Mental Map of Orientation
Mental Map of Shape
Mental Map of Intensity

$$VO_C = (exp(exp(exp(exp(Op_{\{r-g\}}(image))))))^C$$
$$VO_O = G_{\sigma=1}(image)$$
$$VO_S = Y$$
$$\mathrm{VO}_{MM_1} = log((D_x(image) * 0.05) * 0.05)$$
$$\mathrm{VO}_{MM_2} = D_{xx}(image)$$
$$\mathrm{VO}_{MM_3} = log\left(\frac{D_{yy}(image)}{D_{yy}(image) + 2D_{xy}(image)}\right)$$

Fig. 5. Best AVC program for the class engraving.

colors. We use image indexing theory to solve the problem. It consists of mapping the values of each pixel in the input image to colormap values. Colormaps can be of any length with three columns—one for each RGB component specifying the color intensities of red, green, and blue. Images containing grayscale engravings tend to create fewer colormaps. As the final step, we create a decision rule acting over the input images to apply the right classifier. This experiment gives us the first clues about the possibility of considering visual indexing theory as a way to approach classification problems [27].

Therefore, we split the dataset into two classes considering these differences and train two new BP programs (obtained from five runs each), resulting in a classification accuracy of 92% for grayscale and 98.33% for color engraving images. Then, we apply the discovered best programs to the WikiArt database (Tables 4 and 5) obtaining 92.3%; note that these images are in grayscale. As a result, our best program trained with data from Table 3 matches the performance of AlexNet in both datasets. Regarding color, we perform another test using the ukiyo-e class (Japanese engravings) from WikiArt to test the best program scoring 89.71% with a total of 1167 selected images. This difference is due to the unbalanced sets, since Kaggle has only 79 color images; therefore, the achieved performance is high considering the few images used during training.

BP looks for the optimal combination within the search space of possible programs, thus designing multiple symbolic programs. Figure 5 shows the best program for the class engraving. Note that we can observe the step-by-step process similar to GOFAI (good old-fashioned artificial intelligence) systems. At the moment of making a comparison of the best individuals discovered through GP with solutions from the state of the art, it is necessary to use only the best solution. The average and standard deviations give information about the search process, but if we want to know the accuracy of a solution, this needs to be computed afterward. The final solutions are simple compared to AlexNet, so they can be adapted to low-cost computer systems.

6. CONCLUSION

The goal of this research was to outline a methodology called BP that can challenge DL in the categorization problem of art media. BP is an evolutionary system that automatically designs computational (symbolic) models that describe a specific class. BP's symbolic representation is similar to traditional computer vision systems (invariance to the viewpoint, occlusion, segmentation, grouping, integration, description, and the like) needed to recover prototypical shapes. Therefore, the proposed solutions are susceptible to being studied from the standpoint of conventional (machine vision–artificial intelligence) analysis. BP merges ideas from biological vision, cognitive neuroscience, computer vision, evolutionary computation, and psychology. We presented the results of applying this method for the problem of categorization of digitized artworks by media. The results confirm that BP matches or surpasses DL in three out of five classes with an overall accuracy score above 90%. This methodology, unlike DL, can add expert knowledge to the process to solve the problem. It represents an intermediate point between conventional methods for art recognition that spend a considerable amount of time in designing their feature extractors and DL that works like a black box in which the researcher cannot incorporate expert knowledge to choose the features. Our proposal can explain the operation of each of its stages, as exemplified in Fig. 5, unlike DL, where it is not evident to explain the reasons that the algorithm works. In future research, we propose to investigate other weaknesses that DL techniques have such as adversarial attacks.

Funding. Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California (634-135).

Acknowledgment. This research was partially funded by CICESE through the project 634-135, "Estudio de la programación cerebral en problemas de reconocimiento a gran escala y sus aplicaciones en el mundo real." The first author graciously acknowledges the scholarship paid by the Consejo Nacional de Ciencia y Tecnologa (CONACyT) under grant 25267-340078.

Disclosures. The authors declare no conflicts of interest.

REFERENCES

1. H. Liang, R. Groves, and P. Targowski, "Optics for arts, architecture, and archaeology vii," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* (2019), Vol. **11058**.
2. D. Curticapean, O. Vauderwange, and B. Heitz, "Art and photonics," Proc. SPIE **11143**, 395–405 (2019).
3. S. J. Dickinson, A. Leonardis, B. Schiele, and M. J. Tarr, *Object Categorization: Computer and Human Vision Perspectives*, 1st ed. (Cambridge University, 2009).
4. C. R. Johnson, E. Hendriks, I. J. Berezhnoy, E. Brevdo, S. M. Hughes, I. Daubechies, J. Li, E. Postma, and J. Z. Wang, "Image processing for artist identification," IEEE Signal Process Mag. **25**, 37–48 (2008).
5. J. Zujovic, L. Gandy, S. Friedman, B. Pardo, and T. N. Pappas, "Classifying paintings by artistic genre: an analysis of features classifiers," in *IEEE International Workshop on Multimedia Signal Processing* (2009), pp. 1–5.
6. T. Kowaliw, J. McCormack, and A. Dorin, "Evolutionary automated recognition and characterization of an individual's artistic style," in *IEEE Congress on Evolutionary Computation* (2010), pp. 1–8.
7. R. Arora and A. Elgammal, "Towards automated classification of fine-art painting style: a comparative study," in *21st International Conference on Pattern Recognition (ICPR 2012)* (IEEE Computer Society, 2012), pp. 3541–3544.
8. R. G. Condorovici, C. Florea, R. Vrânceanu, and C. Vertan, "Perceptually-inspired artistic genre identification system in digitized painting collections," in *Image Analysis*, J.-K. Kämäräinen and M. Koskela, eds. (Springer, 2013), pp. 687–696.
9. Y. Bar, N. Levy, and L. Wolf, "Classification of artistic styles using binarized features derived from a deep neural network," in *Computer Vision–ECCV 2014 Workshops*, L. Agapito, M. M. Bronstein, and C. Rother, eds. (Springer International Publishing, 2015), pp. 71–84.
10. "WikiArt: visual art encyclopedia," https://www.wikiart.org. Accessed: 2020-02-05.
11. B. Saleh, K. Abe, R. S. Arora, and A. Elgammal, "Toward automated discovery of artistic influence," Multimedia Tools Appl. **75**, 3565–3591 (2016).
12. "Artchive: Mark Hardenś database," http://www.artchive.com/cdrom.htm. Accessed: 2020-02-05.
13. B. Saleh and A. Elgammal, "A unified framework for painting classification," in *IEEE International Conference on Data Mining Workshop (ICDMW)* (IEEE, 2015), pp. 1254–1261.
14. S.-G. Lee and E.-Y. Cha, "Style classification and visualization of art painting's genre using self-organizing maps," Human-centric Comput. Inf. Sci. **6**, 7 (2016).

15. “Web gallery of art: Created by Emil Krén and Daniel Marx,” https://www.wga.hu/index1.html. Accessed: 2020-02-05.
16. C. Florea, C. Toca, and F. Gieseke, “Artistic movement recognition by boosted fusion of color structure and topographic description,” in *IEEE Winter Conference on Applications of Computer Vision (WACV)* (2017), pp. 569–577.
17. A. Elgammal, B. Liu, D. Kim, M. Elhoseiny, and M. Mazzone, “The shape of art history in the eyes of the machine,” in *Thirty-Second AAAI Conference on Artificial Intelligence* (2018).
18. S. Lyu, D. Rockmore, and H. Farid, “A digital technique for art authentication,” in *Proceedings of the National Academy of Sciences* (2004), pp. 17006–17010.
19. S. Agarwal, H. Karnick, N. Pant, and U. Patel, “Genre and style based painting classification,” in *IEEE Winter Conference on Applications of Computer Vision* (IEEE, 2015), pp. 588–594.
20. C. Puente, G. Olague, M. Trabucchi, P. D. Arjona-Villicaña, and C. Soubervielle-Montalvo, “Synthesis of vegetation indices using genetic programming for soil erosion estimation,” Remote Sens. **11**, 156 (2019).
21. G. Olague, E. Clemente, L. Dozal, and D. E. Hernández, “Evolving an artificial visual cortex for object recognition with brain programming,” in *EVOLVE–A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation III*, O. Schuetze, C. A. Coello Coello, A.-A. Tantar, E. Tantar, P. Bouvry, P. D. Moral, and P. Legrand, eds. (Springer, 2014), pp. 97–119.
22. “Art images: drawing/engraving/iconography/painting/sculpture,” https://www.kaggle.com/thedownhill/art-images-drawings-painting-sculpture-engraving. Accessed: 2019-11-30.
23. L. Dozal, G. Olague, E. Clemente, and D. E. Hernández, “Brain programming for the evolution of an artificial dorsal stream,” Cognit. Comput. **6**, 528–557 (2014).
24. G. Olague, D. E. Hernández, E. Clemente, and M. Chan-Ley, “Evolving head tracking routines with brain programming,” IEEE Access **6**, 26254–26270 (2018).
25. G. Olague and M. Chan-Ley, “Hands-on artificial evolution through brain programming,” in *Genetic Programming Theory and Practice XVII*, W. Banzhaf, E. Goodman, L. Sheneman, L. Trujillo, and B. Worzel, eds. (Springer, 2019).
26. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds. (Curran Associates, Inc., 2012), pp. 1097–1105.
27. C. Puente, G. Olague, S. V. Smith, S. H. Bullock, A. Hinojosa-Corona, and M. A. González-Botello, “A genetic programming approach to estimate vegetation cover in the context of soil erosion assessment,” Photogramm. Eng. Remote Sens. **77**, 363–376 (2011).