Universidad Autónoma de San Luis Potosí
Facultad de Ingeniería
Centro de Investigación y Estudios de Posgrado

Robustness Evaluation to Adversarial Perturbations in Image Classification Methods
Evaluación de la Robustez a las Perturbaciones Adversarias para los Métodos de Clasificación de Imágenes

TESIS

Que para obtener el grado de:

Doctorado en Ciencias de la Computación

Presenta:

M.I.E. Gerardo Ibarra Vázquez

Asesor:

Cesar Augusto Puente Montejano
Co-Asesor:
Gustavo Olague Caballero

San Luis Potosí, México
Enero 2022

©2022 – Gerardo Ibarra Vázquez
all rights reserved.

Abstract

Security concerns about the vulnerability of deep convolutional neural networks to adversarial attacks in slight modifications to the input image almost invisible to human vision make their predictions untrustworthy. Therefore, it is necessary to provide robustness to adversarial examples with an accurate score when developing a new classifier. In this thesis, we perform a comparative study of the effects of these attacks on two computer vision tasks: 1) art media categorization, which involves a sophisticated analysis of features to classify a fine collection of artworks, and 2) face recognition, which is the most popular biometric among others to recognize persons. We tested a prevailing bag of visual words approach from computer vision, four deep convolutional neural networks (AlexNet, VGG, ResNet, ResNet101), and brain programming for the art media categorization. The results showed that brain programming predictions' change in accuracy was below 2% using adversarial examples from the Fast Gradient Sign Method attack. With a multiple-pixel attack, Brain Programming obtained four out of seven classes without changes and the rest with a maximum error of 4%. Finally, Brain Programming got four categories without changes using adversarial patches and for the remaining three classes with an accuracy variation of 1%. The statistical analysis confirmed that Brain Programming predictions' confidence was not significantly different for each pair of clean and adversarial examples in every experiment. Lastly, adversarial training demonstrated diminishing the effect of the Fast Gradient Sign Method on deep convolutional neural networks but without providing any defense to the rest of the attacks. These results prove brain programming's robustness against adversarial examples compared to deep convolutional neural networks and the computer vision method for the art media categorization problem. For face recognition, we compare brain programming against a deep convolutional neural network (ResNet) using the facial accessories perturbations attack. In the experiments, brain programming could compete with ResNet without any influence on predictions when the adversarial attack was present. The brain programming accuracy change was below 3% compared to ResNet, which obtained up to 98.56% of accuracy change. A two-sample Kolmogorov-Smirnov test confirmed that ResNet and Brain Programming predictions confidences do not come from populations with the same distribution. Brain Programming's immunity to adversarial attacks demonstrated in this thesis is a significant breakthrough to the evolutionary computation community where this feature could be an edge compared to deep learning techniques. This example could be just the beginning of the secure era of evolutionary computation techniques.

Resumen

Las preocupaciones de seguridad sobre la vulnerabilidad de las redes neuronales convolucionales profundas a los ataques adversarios con pequeñas modificaciones hechas a la imagen de entrada casi invisibles para la visión humana hacen que sus predicciones no sean confiables. Por lo tanto, cuando se desarrolla un clasificador nuevo es necesario tener en mente la robustez a los ataques adversarios asi como un resultado preciso en la clasificación. En esta tesis, realizamos un estudio comparativo de los efectos de estos ataques en dos tareas de visión por computadora: 1) categorización de medios artísticos, que involucra un sofisticado análisis de características para clasificar una colección fina de obras de arte, y 2) reconocimiento facial, que es el biométrico más popular para reconocer personas. Para la categorización de medios artísticos probamos un enfoque tradicional del área de visión por computadora llamado bolsa de palabras visuales, cuatro redes neuronales convolucionales profundas (AlexNet, VGG, ResNet, ResNet101) y el algoritmo del programador de cerebros (por su nombre en inglés, brain programming). Los resultados mostraron que el cambio en la precisión de las predicciones de programador de cerebros estaba por debajo del 2% usando ejemplos contradictorios del ataque Fast Gradient Sign Method. Con un ataque multipíxel, el programador de cerebros obtuvo cuatro de siete clases sin cambios y el resto con un error máximo del 4%. Finalmente, el programador de cerebros obtuvo cuatro categorías sin cambios usando los parches adversarios y para las tres clases restantes con una variación de precisión del 1%. En el análisis estadístico el programador de cerebros mostró que la distribución en la confianza de las predicciones no fue significativamente diferente entre las imagenes limpias y los ejemplos adversarios de cada experimento. Por último, el entrenamiento adversario demostró disminuir el efecto del Fast Gradient Sign Method en las redes neuronales convolucionales profundas pero sin proporcionar ninguna defensa contra el resto de los ataques. Estos resultados demuestran la robustez del programador de cerebros frente a los ejemplos adversarios en comparación con las redes neuronales convolucionales profundas y el método de visión por computadora en el problema de categorización de medios artísticos. En el reconocimiento facial, comparamos el programador de cerebros con una red neuronal convolucional profunda (ResNet) utilizando el ataque de accesorios faciales. En los experimentos, el programador de cerebros pudo competir con ResNet sin la influencia en las predicciones del ataque adversario. El cambio en la precisión del programador de cerebros estuvo por debajo del 3% en comparación con ResNet, que obtuvo hasta un 98,56% de cambio de precisión. La prueba de Kolmogorov-Smirnov de dos muestras confirmó que las confianzas en las predicciones de ResNet y el programador de cerebros no provienen de poblaciones con la misma distribución. La inmunidad del programador de cerebros a los ataques adversarios demostrada en esta tesis es un avance significativo para la comunidad de computación evolutiva donde esta característica podría ser una ventaja en comparación con las técnicas de aprendizaje profundo. Este ejemplo podría ser solo el comienzo de la era segura de las técnicas de computación evolutiva.

Contenido

Contents

List of figures

List of Acronyms

AA Adversarial Attack

AE Adversarial Example

AMC Art Media Categorization

AVC Artificial Visual Cortex

BoV Bag of Visual words

BP Brain Programming

CM Conspicuity Map

CMYK Cyan Magenta Yellow Key

CV Computer Vision

DCNN Deep Convolutional Neural Networks

DL Deep Learning

DV Descriptor Vector

EC Evolutionary Computation

EP Evolutionary Paradigm

EVO Evolutionary Visual Operator

FGSM Fast Gradient Sign Method

FR Face Recognition

FV Fisher Vector

GMM Gaussian Mixture Model

GP Genetic Programming

HMAX Hierarchical MAX

HSV Hue Saturation Value

ML Machine Learning

MM Mental Map

RGB Red Green Blue

SI Swarm Intelligence

SIFT Scale Invariant Feature Transform

SVM Support Vector Machine

VM Visual Map

VO Visual Operator

List of Contributions

Published

Journal Citation Reports

- **“Brain Programming is Immune to Adversarial Attacks: Towards Accurate and Robust Image Classification using Symbolic Learning”**. *Gerardo Ibarra-Vázquez, Gustavo Olague, Mariana Chan-Ley, Cesar Puente, Carlos Soubervielle-Montalvo.* Swarm and Evolutionary Computation. 2022.

International Conferences

- **“A Local Image Feature Approach as a Step of a Top-Down RGBD Semantic Segmentation Method”**. *Gerardo Ibarra-Vázquez, Cesar A. Puente-Montejano and José I. Nuñez-Varela.* 2019 Research in Computing Science. https://doi.org/10.13053/rcs-148-11-3

- **“A Deep Genetic Programming based Methodology for Art Media Classification Robust to Adversarial Perturbations”**. *Gustavo Olague, Gerardo Ibarra-Vázquez, Mariana Chan-Ley, Cesar Puente, Carlos Soubervielle-Montalvo and Axel Martinez.* 2020 International Symposium on Visual Computing (ISVC). https://doi.org/10.1007/978-3-030-64556-4_6

- **“Automated design of accurate and robust image classifiers with brain programming”**. *Gerardo Ibarra-Vázquez, Gustavo Olague, Mariana Chan-Ley, Cesar Puente, Carlos Soubervielle-Montalvo.* 2021 Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO). https://doi.org/10.1145/3449726.3463179

Dedicated to my family, my fiancee and “kika” who encouraged me to pursue my doctoral degree, everything was possible with their support …

Acknowledgments

To my advisors Dr. Cesar Augusto Puente Montejano and Dr. Gustavo Olague Caballero for giving me the opportunity to do research and providing invaluable guidance throughout this work. Their vision, sincerity and motivation have deeply inspired me. The meetings and conversations were vital in motivate me to think outside the box, from multiple perspectives to form a comprehensive and objective critique. It was a great privilege and honor to work and study under their guidance. I would also like to thank them for their friendship, empathy, and great sense of humor.

To my parents for their love, prayers, caring and sacrifices for educating and preparing me for my future.

To my fiancee for her love, understanding, prayers and continuing support to complete this research work.

To the Consejo Nacional de Ciencia y Tecnología (CONACyT) for providing me the resources to pursue my PhD degree.

To my PhD colleagues who provided stimulating discussions as well as happy distractions to rest my mind outside of my research.

To the EvoVision team for their support and kindness during my research work.

To all the people who have supported me to complete the research work directly or indirectly.

Introduction

Image Classification is a current working area in Computer Vision (CV) applications. The objective is to analyze the contextual information or visual content of an image and assign the class or category to which the image belongs [4]. In other words, it is basically to build a computational classification method to predict the class with an associated probability that refers to the *trustworthiness* from the forecast, which requires sets of images for training, validation, and testing where the performance is measured through the *accuracy* metric [5]. Although outstanding results have been obtained in image classification, this problem still has challenges. Firstly, as the number of classes grows, the problem becomes more complex to build the model that generalizes each category. Secondly, when different classes appear in the same image. In that case, it is difficult to isolate one class in the image to train and validate the model and determine whether the image belongs to one class or another when the model is tested.

Two predominant methods have been among the most popular and successful approaches for solving image classification problems: 1) Bag-of-Visual words (BoV) [6] and 2) Deep Convolutional Neural Networks (DCNN), also known as Deep Learning (DL), a subdivision of Machine Learning (ML) [7, 8]. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is one of the main benchmarks for image classification. In 2011, a CV method based on BoV won the classification task at the ILSVRC. This method constructed a vector of occurrence counts of a vocabulary of local image features like dense Scale Invariant Feature Transform (SIFT) [9] extracted from the training images. These vectors were used to train a linear classifier such as Support Vector Machines (SVMs) [10] to predict the class over new vectors. The advantage of BoV is that it does not require labeled data to learn the *dictionary*. However, the process to learn the *dictionary* demands a high computational cost, and their complexity also limits the local image feature extraction.

Therefore, in 2012 a DCNN named AlexNet [1] demonstrated better performance on the classification task by bringing down the error rate by half, beating the predecessor CV approach. Since then, deep learning has achieved outstanding performance in different CV areas. For example, in 2014, a DCNN model named VGG increased the architecture deep and obtained with only 1-network 7.0% error rate, which is better than GoogLeNet, which has a 7.9% error rate with the same number of networks. However, at the submission of ILSVRC 2014, VGG has a 7.3% error rate only, and GoogLeNet with 7-networks obtained 6.7% which got the first runner-up at the moment. Also, VGG won the localization task in ILSVRC 2014. In 2015, the model named ResNet won the image classification task with a 152-layer architecture with 3.57% error rate. These DCNN architectures are explained in Section 2.2.

At the exact moment in 2014, intriguing properties about these models were found by Szegedy et al. [11]. The authors discovered a rare weakness about DCNN, in which the models can be fooled with small modifications almost imperceptible to the human vision on the input pixels. Also, these perturbations reported high confidence in the wrong prediction, and even worse, multiple networks were affected using the same modified image. In 2015, Goodfellow et al. [12] designed a method named Fast Gradient Sign Method (FGSM), which enables efficiently compute perturbations for a given image. Since then, these intentionally created perturbations have been named Adversarial Attacks (AA). At that moment, AA were not considered a serious threat since few works existed about these attacks.

Nowadays, there is a big concern about the security of DCNN, which has opened a new research area in charge of dealing with AA because they are generated through various forms, including making minor modifications to the input pixels, using spatial transformations, among others complex modifications [13, 14, 15, 16, 17, 18]. Some of these perturbations have been adapted to be imperceptible to human vision and can completely change the DCNN's prediction to drop its performance. In addition, there have been immense efforts to develop defense mechanisms to mitigate AA. Still, the perturbations have become more complex and highly efficient in fooling DCNN. The adversary attack's problem is covered in detail in Chapter 3.

Meanwhile, Evolutionary Computation (EC) and Swarm Intelligence (SI) have mainly contributed in two manners in image classification: 1) optimizing feature selection and 2) optimizing DCNN architectures. Genetic Programming (GP) has been one of EC's principal tools to optimize the selection of features and automatically extract the best characteristics to approach image classification tasks. However, the approaches made in this area work with outdated datasets with small images, and their comparison is made with non-state-of-the-art methods. In addition to optimizing feature selection, EC and SI have developed strategies to search for meaningful DCNN architectures for image classification [19]. Recent approaches such as [20, 21], which are summarized in [22], explore hybridization of swarm and evolutionary computation algorithms by aggregating hyper-parameters optimization during training. Despite the effort and interest made by the EC and SI communities to tackle image classification, they still are dealing with outdated problems using non-standard datasets while making comparisons against obsolete DCNN models. EC and SI have fallen short to be on par with DCNN models with minor works that do not exceed hand-crafted DCNN architectures. Therefore, to be competitive with state-of-the-art DCNN models the EC and SI community should propose innovative ways to solve the image classification problem.

Contrary to the approaches made by EC and SI, there is a Deep Genetic Programming Methodology called Brain Programming [23, 24] which has shown promising results in image classification, especially considering the robustness measures shown in this thesis. Brain programming is inspired by neuroscience knowledge that uses symbolic representations and incorporates rules from expert systems with a hierarchical structure inspired by the human visual cortex, which has achieved comparable performance with the renowned DCNN named AlexNet using high definition art images [25].

However, despite the progress made to build better image classifiers, a research opportunity not considered by this research community is the classifier's predictions' robustness. This thesis employs statistical strategies to measure robustness against AA beyond DL models. We evaluate classification approaches from three research areas to contrast performance and robustness to perturbations to guarantee predictions' trustworthiness while not focusing only on accuracy.

Research Question

Based on the previous problem statement and motivated by the emerging challenges on the image classification methods, this thesis proposes the following research question:

- How other image classification methods beyond deep learning respond to adversarial attacks?

Thesis Objectives

Based on the research question presented before, this thesis has the following general objective:

General Objective

- Evaluate the ability of an evolutionary computing method to resist several kinds of adversarial attacks
- Propose a methodology to measure robustness against adversarial examples to ensure the predictions' trustworthiness in image classification methods

Main Objectives

Derived from the stated general objective, the following specific objectives are presented:

1. Analyze the problem of adversarial attacks and the implication to the trustworthiness of image classification methods
2. Analyze the performance of image classification methods and their vulnerability to adversarial attacks using standard models
3. Propose a novel methodology that evaluate the robustness to adversarial examples from different classification algorithms
4. Test the methodology with two different image classification tasks (art media classification and face recognition)

Thesis Outline

Chapter 1 presents a literature review of image classification approaches and adversarial attacks. It reports the state of the art in image classification, where the most popular and successful approaches are highlighted to understand the progress conceived in this research area. It also presents the concerns about AAs and the research opportunity to study the classifier's predictions' robustness through different approaches to contrast between performance and robustness to adversarial examples to guarantee predictions' trustworthiness while not focusing only on accuracy. In Chapter 2, the mathematical modeling of the leading approaches in image classification is explained. Each approach's attributes are outlined to understand the main contribution, the techniques, and methods employed to contrast them. Chapter 3 presents the severe problem in the DCNN structure to AA, and the mathematical explanation from this dilemma is deeply explained. Examples of AA are illustrated in this chapter to understand the concerns about the security of the DCNN predictions and the implications of this vulnerability using different attack designs. Chapter 4 presents the novel methodology where it is proposed the robustness evaluation to adversarial examples to measure the classifier prediction's trustworthiness, allowing the analysis of distinct image classification approaches in complex and real-world applications. In Chapter 5, the details about the usage of the robustness evaluation in two experimental case studies (Art Media Classification and Face Recognition) are presented. The results obtained are accurately estimated under the assumption of different attempts to fool the systems, and they highlight the differences between the approaches by considering performance and robustness against adversarial examples.

1

State of the Art in Image Classification and Adversarial Attacks

There have been significant efforts to tackle the image classification problem in many research areas such as Computer Vision (CV), Machine Learning (ML), Evolutionary Computation (EC), and Swarm Intelligence (SI) [26, 27, 19]. Two predominant methods have been among the most popular and successful approaches for solving image classification problems. On the one hand, Bag of Visual words (BoV) from CV, and on the other hand, Deep Convolutional Neural Networks (DCNN), a mainstream from Deep Learning (DL), a subdivision of ML [7, 8]. EC and SI have mainly contributed in two manners: 1) optimizing feature selection via symbolic learning, and 2) optimizing DCNN architectures. In Figure 1.1 is shown a fishbone diagram, where contributions from these research areas to image classification are presented.

1.1 Computer Vision

The most popular approach for image classification used by CV was the BoV, which is explained in Section 2.1, it is inspired by the bag-of-words method [6]. Several variants of this framework have been tested. For example, the ones who use better coding techniques based on soft assignment [28, 29, 30], the ones that take into account spatial layout with spatial pyramids [31] and sparse coding [32, 33, 34].

Sparse coding is one of the most advanced methods from the BoV's frameworks [32]. The idea of sparse coding is to represent the description of the image, such as dense SIFT description [9, 35], HOG description [36], among others, approximately as a weighted linear combination of a small number of unknown basis vectors. These basis vectors capture the high-level patterns in the image description.

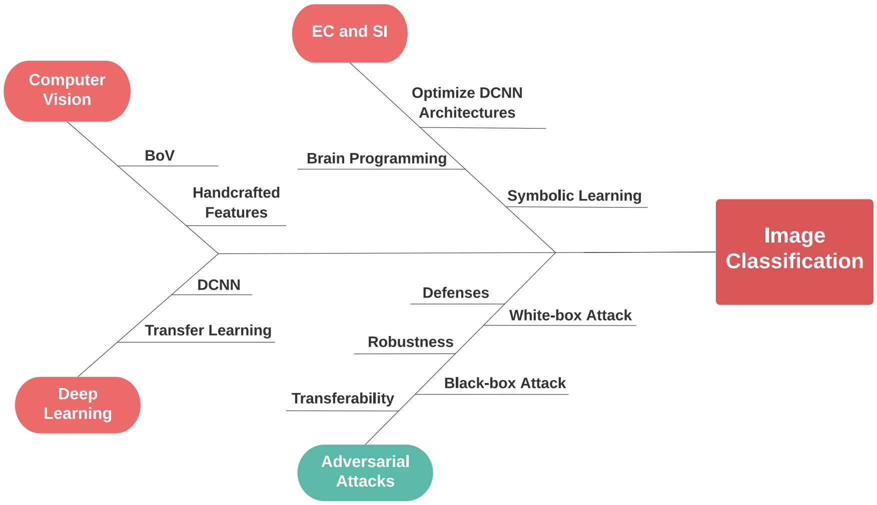


Figure 1.1: General overview of contributions to image classification from computer vision, deep learning, evolutionary computation, and swarm intelligence.

However, the sparse coding's performance relied principally on the hand-engineered features, and the CV target was to design better hand-engineering features. Over time, the complexity of these features started to become more challenging to design better features. In addition to the designing process, CV also focused on learning algorithm design, a completely independent research area. The advantage of using sparse coding is that is not require labeled data to learn the *dictionary*. So, it can work on limited labeled data situations. Also, the dictionary learning process can improve features quality by providing additional information of them [37, 38]. However, sparse coding is not capable of building features hierarchies, and the process is not simply stacked one method on top of another, even there have been attempts to make it deep [39, 40, 35]. Figure 1.1 shows the contributions to the image classification problem from CV.

1.2 Deep Learning

ML community was working in another direction by designing deep learning models (i.e., the neural network architecture) that build features from images. LeCun et al. [41] introduced the modern framework of Convolutional Neural Networks, which is explained in Section 2.2 and Figure 1.1 shows the general overview of contributions from Deep Learning. However, the first time that they started attracting attention was with the development of the AlexNet model [1] for the ILSVRC 2012. It could reduce by half the error rate on the image classification task.

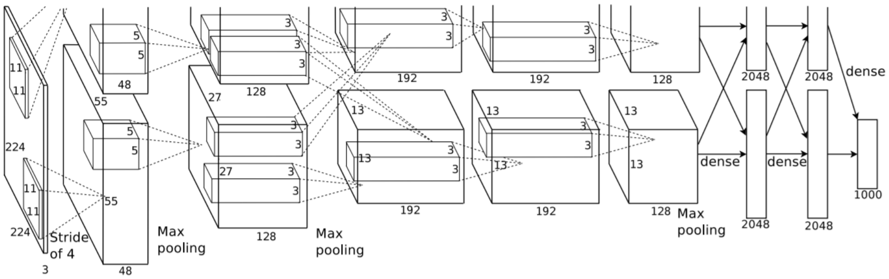
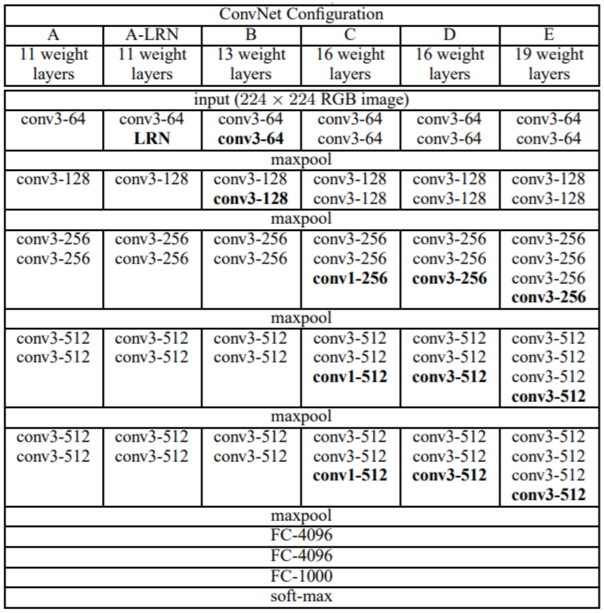


Figure 1.2: General overview of AlexNet architecture. The figure was extracted from the original article [1].

AlexNet layer architecture consists of 5 convolutional, three max-pooling, two normalizations, three fully connected layers (the last with 1000 softmax output), 60 Million parameters, and 500,000 neurons. Figure 1.2 illustrates the general overview of AlexNet architecture. Additionally, Alex et al. [1] introduced the use of ReLU (Rectified Linear Unit) Nonlinearity as an activation function with the benefits of much faster training than using tanh or sigmoid functions. To prevent overfitting, they also introduced the dropout method and data augmentation.

Another deep learning model that brought contributions to the state-of-the-art was the VGG network from the Visual Geometry Group of the University of Oxford [2]. VGG network increased the deep of these models by creating VGG-16 with 13 convolutional layers and three fully connected layers, and VGG-19 with additional three convolutional layers than VGG-16. In Figure 1.3 is observed all the configurations made from VGG architecture, the most used are VGG-16 and VGG19. They reduced the size of the filters to the smallest size to capture the notion of up/down, left/right, and center that is a 3x3 filter. VGG was distinguished for its state-of-the-art recognition and localization tasks on ILSVRC and other image recognition datasets.

ResNet [3] (Deep Residual Learning for Image Recognition) also contributed to redefining the layer as a residual learning function on the architecture. Figure 1.4 presents an overview of ResNet architecture and compare the 34-layer residual against 34-layer plain and VGG-19 architectures. This function helps mitigate the bottleneck problem of the training phase on DCNNs. ResNet showed its capacity to train its architecture with a depth of up to 152 layers and lower complexity than GoogLeNet. Also, ResNet won the ILSVRC 2015 on the classification task achieving for the first time the error rate to 3.57%.



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 1.3: General overview of VGG architectures. The figure was extracted from the original article [2].

1.3 Evolutionary Computation and Swarm Intelligence

Genetic Programming (GP) has been one of EC's principal tools to optimize the selection of features and automatically extract the best characteristics to approach image classification tasks. For example, in 2018, authors of [42] proposed a GP method that achieved simultaneously global and local feature extraction for image classification using the JAFFE (1998), YALE (1997), FLOWER (2007), and TEXTURE (2006) datasets. As can be seen, all datasets are outdated nowadays since no one uses them to test algorithms. Moreover, their approach is compared to standard hand-engineered features from CV like SIFT (Scale-Invariant Feature Transform), an image processing technique that follows the local feature paradigm. SIFT does not behave well for image categorization problems since different images with multiple attributes represent an object category. The solution demands a consensus of distinct characteristics in the form of a set of features. In 2019, the article [43] proposed a GP approach to automatically generate discriminative rich features for image classification using the MIT urban and nature scene datasets (2003). These image databases are also outdated, and the comparison is made with traditional CV classification methods like Histogram of Oriented Gradients (HOG) and Support Vector Machine (SVM), similarly to the previous work.

In 2019, Iqbal et al. [44] proposed a method for employing transfer learning in GP to extract and transfer knowledge to classify complex texture images. The proposed methodology uses the following texture datasets Kylberg (2011), Brodatz (1999), and Outex (2002), and all images are resized to 115×115 pixels to perform their experiments to avoid the computational cost and simplify the problem. In 2020, the article [45] proposed a GP-based feature learning approach to select and combine five methods automatically: Hist (Histogram features), DIF (Domain-Independent Features), SIFT, HOG, and LBP

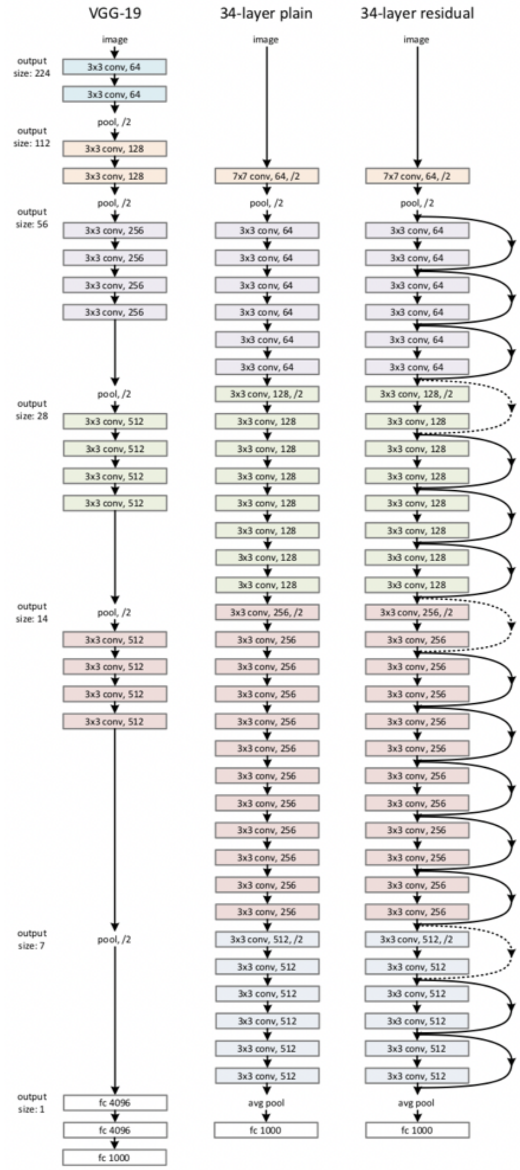


Figure 1.4: General overview of ResNet architecture, where its compared the 34-layer residual against 34-layer plain and VGG-19 architectures. The figure was extracted from the original article [3].

(Local Binary Patterns). The technique generates a compound solution that extracts high-level features to classify images from classical problems with low-resolution datasets–about 100×100 pixels up to 200×200 pixels. Authors compared their approach with other GP-based methods and DL methods like LeNet-5 (a CNN model with an input of grayscale images of 32×32 pixels, toy-method in comparison with the state-of-the-art) and two handcraft CNNs models of five- and eight-layers without providing the network parameters' information. Hence, it is not easy to judge the performance.

In 2021, authors from [46] proposed an instance selection-based surrogate-assisted GP for fast feature learning. They used 11 datasets FEI_1(2012), FEI_2(2012), KTH(2006), FS(2005), MB(2007), MRD(2007), MBR(2007), MBI(2007), Rectangle(2007), RI(2007), and Convex(2007) from 28×28 to up to 60×40 pixels. They compared their method with 32×32 input images DCNN such as evoCNN and two handcrafted CNNs models of five- and eight-layers among other non-state-of-the-art DCNN. Also, in 2021, the research work [47] proposes a GP-based approach with a dual-tree representation to learn image features for few-shot models. In this work, the methods and datasets used for comparison were not from the state-of-the-art of few-shot image classification.

Besides optimizing feature selection, EC and SI have developed strategies to search for meaningful DCNN architectures for image classification [19]. Also, recent approaches, summarized in [22], explore hybridization of the swarm and evolutionary computation algorithms by aggregating hyper-parameters optimization during training. To give an example, in 2019, authors from [20] proposed a novel method named evoCNN, which uses genetic algorithms for evolving DCNN architectures and connection values to address image classification problems. They based the experiments on nine datasets that use grayscale images of 28×28 pixels: MNIST, MNIST-RD, MNIST-RB, MNIST-BI, MNIST-RD + BI, Rectangles, Rectangles-I, Convex, and MNIST-Fashion. However, in 2019, authors from [48] proposed a novel algorithm based on particle swarm optimization (PSO) named psoCNN, capable of automatically searching DCNN architectures for image classification with fast convergence when compared with others evolutionary approaches like evoCNN, IPPSO, among others. The proposed experiments used the same nine datasets mentioned above. In 2021, the research article [49] proposed an evolutionary algorithm for searching DCNN architectures under multiple objectives, such as classification performance and floating-point operations (FLOPs). They find optimized DCNN architectures for CIFAR-10(2009), CIFAR-100(2009), and ImageNet(2009) datasets, but they still do not surpass state-of-the-art handcrafted DCNN. Table 1.1 shows that even recent works from the journal swarm and evolutionary computation optimize renowned DCNN models while excluding the best state-of-the-art methods for the studied datasets, denoting the relation with the latest approaches reached by the EC and SI research area.

Reference	DCNN models employed in the research work	Datasets used
He et al. [50]	AlexNet, VGG16, GoogleNet, SqueezeNet, ResNet-50, Inception-v3, DenseNet121, EvoCNN, among others	MB, MBI, MRB, MRD, MRDBI, RECT, RI, CS, FASHION, Real-World Xiangya-Derm
Singh et al. [51]	AlexNet, EvoCNN, IPPSO, among others	MNIST, CIFAR10, CIFAR100, CS, MDRBI
Darwish et al. [52]	VGG16, VGG19, Inception-v3, Xception	Subset of Plant Diseases
Wang et al. [53]	AlexNet, VGG16, VGG19, GoogleNet, ResNet52, ResNet101, DenseNet121	CIFAR10

Table 1.1: Recent works from the journal Swarm and Evolutionary Computation. For each work, it is presented the DCNN models employed and the datasets used.

Despite the effort and interest made by the EC and SI communities to tackle image classification, they still are dealing with outdated problems using classical datasets while making comparisons against obsolete DCNN models to the proposed problems. As a result, EC and SI have fallen short of being on par with DCNN models with minor works that do not exceed handcraft DCNN architectures. Nonetheless, a Deep Genetic Programming Methodology called Brain Programming (BP), inspired by neuroscience knowledge that uses symbolic representations and incorporates rules from expert systems with a hierarchical structure inspired by the human visual cortex was developed by the EvoVision research team. In Figure 1.1, we illustrate the general overview of contributions from EC and SI to image classification.

In 2016, EvoVision started evolving an Artificial Visual Cortex (AVC) for image classification and object detection. Hernández et al. used realistic images of medium size (VGA) using GRAZ-01 (2003), and GRAZ-02 (2004) datasets, which are the base for the Visual Object Challenge (VOC challenge)–both still relevant in CV literature–[23]. Authors compared the results with several feature extraction methods: Basic Moments (2006), Hierarchical MAX - Genetic Algorithm–HMAX-GA (2012), Enhanced Biologically Inspired Model–EBIM (2011), SIFT (2006), Similarity Measure Segmentation–SM (2006), and Moment Invariants (2006); most from CV and one including EC. In 2017, Hernández et al. [54] implemented a CUDA version of BP to speed up the original system's processing time. The experiment analyzed the performance using different image sizes, which started with 256×256 pixels, doubling the sizes to up to 4096×4096 pixels, demonstrating the possibility of real-time functionality and the application to high-definition images. Additionally, the authors compared the method regarding time performance with a CUDA implementation of HMAX and the CUDA version of a CNN with outstanding results.

In 2019, the article [24] proposes a random search to find best-fit programs for the AVC in image classification. The experiment found great individuals to classify GRAZ-01, GRAZ-02, and Caltech-101 (2004) datasets. GRAZ datasets have image sizes of 640×480 pixels, and Caltech-101 has images of 300×200 pixels. GRAZ images present a significant challenge due to the short object occurrence in the whole picture, becoming challenging to resize images for processing. In contrast, Caltech-101 presents a truly image recognition dataset. In 2020, BP was proposed as a technique to approach the complex problem of Art Media Categorization (AMC) [25]. The experiment consists of classifying high-resolution art datasets such as WikiArt (2016) and Kaggle Art Images (2018). Moreover, BP results were compared with a renowned DCNN model named AlexNet, obtaining a competitive outcome. Also, the authors evaluated BP on real-world problems of object tracking using standard datasets and algorithms like FRAGtrack and MILtrack. While also achieving outstanding results in real-working conditions compared to the method of Region-based Convolutional Neural Networks (R-CNN) [55, 56].

1.4 Adversarial Attacks

Despite the progress made to build better image classifiers, the study of the performance through vulnerabilities on the systems is a featured not considered in EC and SI. Nowadays, there is a big concern about the performance of DCNN, which has opened a new research area in charge of dealing with Adversarial Attacks (AA) that intentionally create small perturbations in the input image to mislead the model to predict the wrong class [13, 14, 15, 16, 17, 18]. Although, AA are a part of DL research area, they have contributed to the image classification problem as shown in Figure 1.1. Some of these perturbations are invisible to human vision and can completely change the DCNN's prediction to drop its performance. Researchers generate attacks through various forms, including making slight modifications to the input pixels, using spatial transformations, among others. In addition to the analysis of DCNN vulnerabilities, there have been immense efforts to develop defense mechanisms to mitigate AA. Still, the perturbations have become more complex and highly efficient in fooling DCNN. Further explanation about the behavior, use, and implementation of AA is made in Chapter 3.

Szegedy et al. [11] were the first who discovered a rare weakness of DCNN which with small perturbations almost imperceptible to the human vision on the input pixels can fool a convolutional neural network. Also, these attacks reported high confidence in the wrong prediction of the model and even worse, multiple networks were affected using the same perturbed image. Moosavi-Dezfooli et al. [57] discovered peculiar perturbations that can misclassify any image, they called it "universal perturbations". However, Szegedy et al. [22] found that the robustness of a DCNN against these adversarial attacks could be improved using these images in the training phase. So, Goodfellow et al. [12] design a method named Fast Gradient Sign Method (FGSM) which enables efficiently to compute perturbations for a given image.

Although there has been much progress in making defense methods against adversarial attacks by modifying its training process or modifying the input image during testing [12, 58, 59], or modifying the structure of the networks [60, 61, 62] or using external models to classify unseen examples [63, 13], the attacks have become more and more complex with high efficiency on the attacks . For example, Sarkar et al. [64] designed the Universal Perturbations for Steering to Exact Targets (UPSET) and the Antagonistic Network for Generating Rogue Images (ANGRI) for targeted attacks of CNNs. Baluja and Fischer [65] designed the Adversarial Transformation Networks (ATNs) which are feed-forward neural networks trained to generate perturbed images against other targeted CNN or set of CNNs. Even, Su et al. [66] designed an extreme case of an adversarial attack on which with the modification of one pixel in the image can make a CNN to misclassify an image. They obtained a 67.97% of success on the attacks using three different network models. Also, it was reported an average of 97.47% on the confidence of the misclassified images.

Table 1.2 shows that despite existing newer DCNN architectures, recent works still use renowned state-of-the-art models to find a solution to the problem of AA. The reason is that these models have been publicly available for research. They have been well studied and validated in different areas. In this way, it is easier to determine what happened experimentally due to the difficulty of obtaining a complete theoretical analysis to generate a general solution for all the attacks. These empirical studies are popular in the state of the art [67, 68, 69, 70, 71, 72].

AA is a hot topic regarding ML and DL since such algorithms suffer from this kind of perturbations. Figure 1.1 shows the contributions from AA and the connection to the three research areas mentioned above to the image classification problem. However, this vulnerability has not been proved that affect algorithms beyond these research areas. Hence, the evaluation of robustness to perturbations that guarantee predictions' trustworthiness while not focusing only on accuracy is of great importance for the research community. Mainly, there are two ways to tackle this problem: 1) solve the vulnerability from DCNN (which has not been elucidated yet), 2) find an alternative to DCNN that is not susceptible to these perturbations.

Reference	DCNN models employed in the research work	Datasets used
Chen et al. [67]	AlexNet, VGG16, ResNet50, Inception v4, Inception-ResNet, ResNeXt, DenseNet-121, PNAS-Net	AID, UC, NWPU, EuroSat-MS, MSTAR, SEN1-2(SAR-Summer)
Pestana et al. [73]	VGG, ResNet, DenseNet,Inception, Mobilenet, ShuffleNet, MnasNet	ImageNet-R
Duan et al. [74]	Inception v3, Inception v4,InceptionResNet v2, ResNet152, Inception v3$_{ens3}$, Inception v3$_{ens4}$, InceptionResNet v2$_{ens}$	1000 images from NIPS 2017: Defense Against Adversarial Attack
Hirano et al. [75]	Inception V3, VGG16, VGG19, ResNet50, Inception ResNet v2, DenseNet121, DenseNet169	Skin lesions, OCT, Chest X-ray
Lee et al. [76]	WideResNet-34-10, ResNet50	CIFAR10, CIFAR100, SVHN, Restricted ImageNet
Xie et al. [77]	ResNet50,EfficientNet B0-B7	ImageNet-C, ImageNet-A,Stylized-ImageNet
Zhang et al. [78]	AlexNet,GoogleNet,VGG16, VGG19, ResNet152	ImageNet, COCO, VOC, Places365
Kim et al.[79]	ResNet50	1000 random images from ImageNet
Oregi et al. [80]	Custom 3-layers CNN	MNIST, SVHN, GTSRB

Table 1.2: Recent works in adversarial examples. For each work, it is presented the DCNN models employed and the datasets used.

2

Data Modeling of Main Approaches in Image Classification

This chapter describes the data modeling of each method used in this work. However, here we explained the general modeling in image classification, which represents the data by fitting it into a model that establishes a relationship between the image $\mathbf{x}$ and the label y provided by a dataset as follows:

$$y = f(\mathbf{x}), \tag{2.1}$$

where the function $f()$ is the model that depends on adjustable parameters[81].

The following sections detail the SIFT + Fisher Vectors modeling as the last BoV method that won the image classification task on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2011 before DL models arose. Additionally, we describe the general overview of DCNN's architecture and its data modeling. Finally, we present the theory behind BP to introduce function-symbolic learning for data modeling and the system's workflow.

2.1 SIFT + Fisher Vectors

Fisher Vector (FV) is a vectorial representation of the gradient of the sample log-likelihood concerning a generative model of the data [82]. There are many advantages to the FV against the BoV. Sánchez et al. in [82] proved that BoV is a particular case of the FV where is restricted the gradient computation to the mixture weight parameters of the Gaussian Mixture Model (GMM) [83]. GMM is a probabilistic visual vocabulary, while FV incorporates additional gradients that improve accuracy. Also, it needs fewer vocabularies with lower computational costs than BoV, and it is easy to achieve good performance with simple linear classifiers. BoV is relatively sparse while the FV is almost dense, making FV impractical for large-scale applications due to storage problems. Nonetheless, researchers apply large-scale nearest neighbor search to mitigate this problem using a popular computer vision method named product quantization [84]. In practice, SIFT descriptors are used on a dense multi-scale grid to compute the FV image representation [82].

In order to construct the FV image representation, it is defined a set of D-dimensional descriptors extracted from an image $X = \{x_t, t = 1, \ldots, T\}$, a set of SIFT descriptors. FV is a sum of normalized gradient statistics $\delta_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(x_t)$ with the assumption that all descriptors are independent. Where $L_\lambda \nabla_\lambda \log u_\lambda(x_t)$ is the normalized gradient statistics computed for each descriptor. It can be understood that this operation is an embedding of the local descriptors $x_t \rightarrow \phi_{FK}(x_t) = L_\lambda \nabla_\lambda \log u_\lambda(x_t)$ in a higher-dimensional space which helps a linear classifier to model the data easier as in Equation (2.1).

In Figure 2.1, we illustrate the workflow overview of the SIFT+FV method. At the top, it is shown the dictionary learning procedure, where SIFT descriptors are extracted for all training images to learn the dictionary. In the middle, the training procedure is described, where for each image in the training dataset, SIFT descriptors are obtained to encode them into the FV with the dictionary. After all images are encoded into FVs, the SVM is trained. When testing, SIFT descriptors are extracted from the input image to encode them into the FV and send it to the SVM for the forecast.

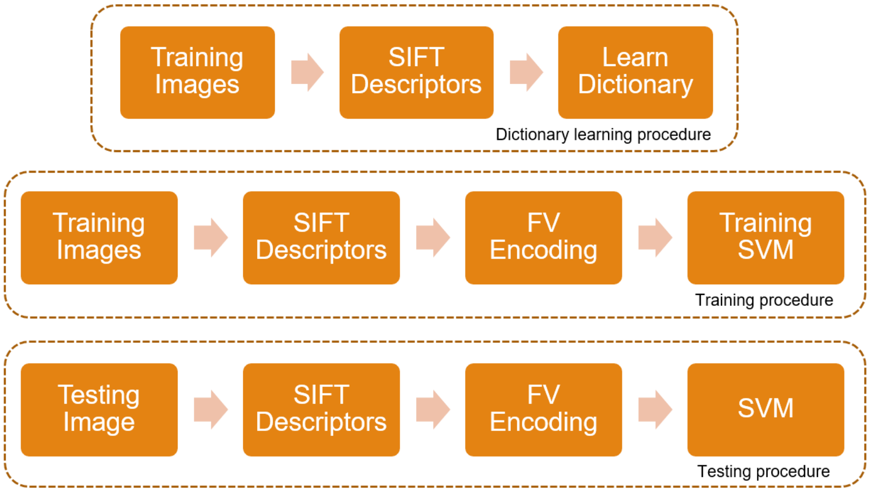


Figure 2.1: Workflow overview of SIFT+FV method. The top of the image shows the dictionary learning procedure. In the middle is illustrated the training procedure. The bottom of the image displays the testing procedure.

These algorithms' advantage is that they do not require labeled data to learn the *dictionary*. BoV algorithms can work on cases with limited labeled data. The dictionary learning process can also improve feature quality by providing additional information of them [37, 38]. However, they are not capable of building feature hierarchies, and the process is not merely stacked one method on top of another even there have been attempts to make it deep [39, 40, 35].

2.2 Deep Convolutional Neural Networks

Deep Convolutional Neural Networks is a deep learning architecture inspired by biological animal visual perception [85]. The DCNN architecture is divided into multiple learning stages comprised of convolutional layers, activation functions, sub-sampling or pooling layers, and fully-connected layers [86, 87]. Convolution layers help extract valuable features from the input image and are composed of several convolution kernels used to compute different feature maps. Activation functions help to learn abstractions and integrate non-linearities in the feature space. The non-linearities facilitate learning semantic differences in the images by generating different activation patterns for different responses. Sub-sampling or pooling layers help to get shift invariance by reducing the resolution of the feature maps. Fully-connected layers are found at the end of the architecture. These layers perform high-level reasoning to generate global semantic information. The DCNN's output layer for the classification task is commonly a softmax function that performs the forecast.

DCNN models the data using Equation (2.1) employing $f_{DNN}()$ as a particular form of a nested function, and each one called a layer [81].

$$y = f_{DNN}(x) = f_3(\mathbf{f}_2(\mathbf{f}_1(\mathbf{x}))) \quad , \tag{2.2}$$

in such a way that $\mathbf{f}_1$ and $\mathbf{f}_2$ are vector functions of the following form:

$$\mathbf{f}_l(\mathbf{z}) = \mathbf{g}_l(\mathbf{W}_l\mathbf{z} + \mathbf{b}_l) \quad , \tag{2.3}$$

with l denoting the index of the layer. $\mathbf{g}_l$ is the activation function that usually is a nonlinear function, and the model parameters consist of $\mathbf{W}_l$ the weights matrix and $\mathbf{b}_l$ the bias vector. Hence, the minimization problem is defined by the loss function $J(\theta, \mathbf{x}, y)$ where the goal is to find the best model parameters for all the layers Θ that fits the data $\mathbf{x}$ to the label y.

2.3 Brain Programming

Before we explain the algorithm of BP, we make a brief introduction to GP algorithms since this is the core of the search process for the models created by BP. GP is an evolutionary computation technique inspired by biological evolution principles [88]. It is considered a derivative of genetic algorithms that evolve individuals' populations in a tree or computer program (formulas or mathematical expressions). Each computer program is generated depending on the terminal and function sets established by the user. First, the system evaluates programs in terms of how well it performs in a particular problem. Then, using the Darwinian principle of reproduction and survival of the fittest and the genetic operators of crossover and mutation, individuals are evolved to find a better fit solution to the problem.

BP is an evolutionary paradigm for solving CV problems[*]. This methodology extracts characteristics from images through a hierarchical structure inspired by the brain's functioning. BP proposes a GP-like method, using a multi-tree representation for individuals. The main goal is to obtain a set of evolutionary visual operators ($EVOs$), also called visual operators (VOs), embedded within a hierarchical structure called the artificial visual cortex. The AVC is based primarily on two models: a psychological model called feature integration theory [89] and a neurophysiological model called the two pathway cortical model [90]. According to the brain's neurological ventral-dorsal model, the AVC attempts to emulate the natural process along the visual cortex. This two-stream model states that the process of acquiring visual information in the brain follows two main pathways.

The dorsal stream is known as the "where" or "how" stream. This pathway is where actions and recognizing objects' location in space are involved and where visual attention occurs. BP follows the most popular theory of feature integration for the dorsal stream from [89], whose principles of the first computational model for visual attention are in [91]. The theory states that visual attention in human beings proceeds in two stages. The first one is called the preattentive stage, where the natural system computes visual information processing in parallel over different feature dimensions that compose the scene: shape, color, orientation, spatial frequency, brightness, and motion direction. The second stage, called focal attention, integrates the extracted features from the previous stage to highlight a region of the scene. In the computational model, the image is decomposed into several dimensions to obtain a set of conspicuity maps, and finally, a single map called the saliency map integrating the four dimensions.

The ventral stream is known as the "what" stream. This pathway is mostly associated with object recognition and shape representation tasks. Proposed ventral stream models like neocognitron system [92], convolutional neural networks [41], and HMAX model [93] (the Max principle is used in BP), start by decomposing the image into a set of alternating "S" and "C" layers. The "S" or simple layers define a set of local filters applied to find higher-order features, and the "C" complex layers increase the features' invariance by combining units of the same kind. However, BP follows a function-driven approach instead of a data-driven paradigm. In the function-driven process, a set of visual operators–fused by synthesis–describe the image's properties. Through a set of experiments, we will show that the discovered solutions do not rely directly on the data but on specific characteristics; hence, making the solutions reliable.

BP consists of two steps: first, the evolutionary process, whose primary purpose is to discover functions to optimize complex models by adjusting its operations. Second, the AVC, a hierarchical

[*] For more details about the inspiration of Brain Programming, please consult the following research works [23, 24, 56].

structure inspired by the human visual cortex, uses the concept of function composition to extract features from images. The model can be adapted depending on the task, whether it is trying to solve the focus of attention for saliency problems or the complete AVC for categorization/classification problems. BP differs from the data-driven models using a function-driven approach to extract and combine the relevant information that solves a specific visual task. The overall function-driven process requires the input in a suitable representation; thus, we define in the mathematical form below an image I as the graph-of-a-function, which refers to the graph as the triplet values of x-coordinate, y-coordinate, and the pixel value at x, y coordinates.

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 consist 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 highlights that images result from the impression of variations in light intensity along the two-dimensional plane. Therefore, functions are optimized to imitate the functionality of specialized areas of the brain through a set of operators.

2.3.1 Data Modeling with BP

BP proposes to solve the problem of image classification from the standpoint of data modeling through GP. Therefore, to understand the learning process of BP, we start defining the minimization problem, which requires finding a solution $\mathbf{P}_{min} \in S$ such that:

$$\forall \mathbf{P}_{min} \in S : f(\mathbf{P}_{min}) \leq f(\mathbf{P}) \quad . \tag{2.4}$$

The strategy takes several steps because the direct mapping between the domain and codomain is unknown or not well defined. Hence, instead of conventional approaches to finding best-fit parameters, we would like to fit the data by discovering functions that perform a classification task in BP. In this manner, the solution to the image classification problem through BP requires to define the following equation:

$$min(y - f(\mathbf{x}, \mathbf{F}, \mathbf{T}, \mathbf{a})) \quad , \tag{2.5}$$

where $(y, \mathbf{x})$ are the label and the image respectively, given by the dataset; $f(\cdot)$ represents the classifier, $\mathbf{F}$ and $\mathbf{T}$ represents the function and terminal sets respectively from the feature extraction, and $\mathbf{a}$ are the parameters controlling the evolutionary process. Therefore, BP is the algorithm in charge of tuning $(\mathbf{F}, \mathbf{T})$ looking for optimal feature extraction from the input images using the visual operators embedded into the artificial visual cortex (AVC). The feature selection process works as a wrapper method based on the BP algorithm for fitting the whole AVC to the dataset. The criterion for minimization in terms of a classification task helps discover an optimal solution to the problem. In this particular case, we use an SVM to learn a mapping $f(\cdot)$ that associates descriptors $\mathbf{d_i}$ created by the AVC to labels y_i. Here, we define the BP algorithm in terms of a binary classification task, whose primary purpose is to find a boundary that best separates the class elements.

Evolving an Artificial Visual Cortex (AVC)

Each individual consists of syntactic trees defining the VOs that constructs the AVC structure to extract features from color images. In this procedure, the AVC designs a descriptor vector that encodes salient characteristics from the image. The descriptor concatenates the information from the four dimensions, resulting in a n global maxima vector. Then, an SVM performs the image classification that addresses individual fitness by calculating the accuracy of a given training image database. BP uses an evolutionary loop presented in Algorithm 1 to evolve the entire population represented by a set of AVCs. In Figure 2.2 we illustrate the AVC model that is optimized in the evolutionary loop through GP to find the best individual (best AVC model) for the image classification task. The AVC model follows a procedure that is detailed next to extract the features to build the image descriptor to be classified.

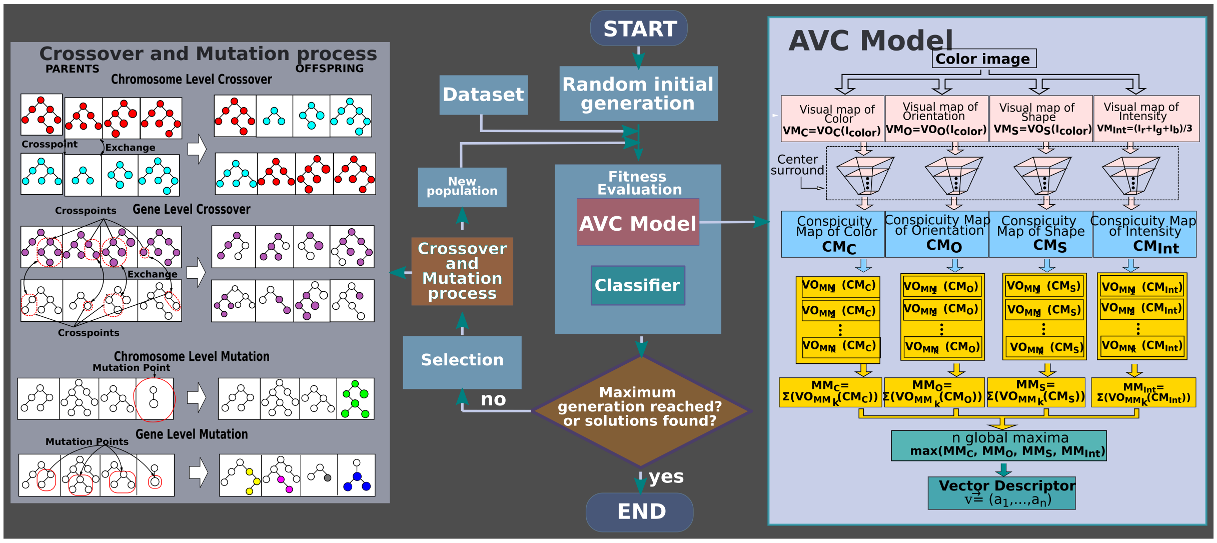


Figure 2.2: Brain Programming workflow. The left side shows the genetic operations; in the middle, we observe the BP's flow diagram, and the right side illustrates the individual representation.

Structure Representation and Genetic Operations

In BP, an individual is a computer program represented by syntactic trees embedded into a hierarchical structure. Individuals within the population contain a variable number of syntactic trees, ranging from four to 12, one for each evolutionary visual operator (VO_O, VO_C, VO_S, VO_I) regarding orientation, color, shape, and intensity; and at least one tree to merge the resulting Visual Maps, and finally, generate the Mental Maps (MM). All functions within each VO are defined according to expert knowledge

Algorithm 1: BP evolutionary process

```
Input  : Training images, Algorithm parameters (see Table 2.5)
Output: The updated population AVCs
Generate a random initial population $P_0$;
$i = 0$;
while the termination criterion is not satisfied do
    Evaluate each individual fitness (AVC) in $P_i$ ;
    Selection using lexicographic parsimony pressure;
    Generate offspring by crossover and mutation;
    $i = i + 1$;
return The updated population $P_{final}$
```

to highlight characteristics related to the respective feature dimension and updated through genetic operations.

- Visual Maps

Each input image is transformed to build the set I_{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), CMYK (Cyan, Magenta, Yellow, and black) and HSV (Hue, Saturation, and Value) color spaces. Elements on I_{color} are the inputs to four VOs defined by each individual. Each VO is a mapping function applied to the input image to extract specific features from it, along with information streams of color, orientation, shape, and intensity; each of these properties is called a dimension. The output from the VO is an image called Visual Map (VM) for each dimension. It is important to note that each solution in the population represents a complete system and not only a list of tree-based programs. Individuals represent a possible configuration for feature extraction describing input images and optimized through the evolutionary process. Next, we explain the process of VOs to extract features on each dimension to obtain a resulting VM.

The first tree of the individual mimics the orientation. We evolve this visual operator (VO_O) through a set of specially selected elements to highlight edges, corners, and other orientation-related features using the set of terminals and functions provided in Table 2.1. The input for the functions can be any of the terminals, and the composition among the functions; G_σ are Gaussian smoothing filters with $\sigma = 1, 2$; and D_u represents the image derivatives along the direction $u \in \{x, y, xx, yy, xy\}$. Finally, D_x represents an approximation to the discrete derivative of image A in the direction of the x axis. We calculate these operations with a convolution between a kernel and an image. The convolution kernel results from deriving the Gaussian function along x:

$$
\begin{aligned}
D_x(A) =& A \star \frac{\partial G_\sigma(x)}{\partial x} \\
=& A \star \left(\frac{-x}{\sigma^2} * e^{\frac{x^2}{2\sigma^2}} \right),
\end{aligned}
\tag{2.6}
$$

where $\sigma = 1$. The objective of this operation is to highlight changes in intensity along the x-axis, which emphasizes vertical edges. On the other hand, D_y uses $\frac{\partial G_\sigma(y)}{\partial y}$ as the convolution kernel, which makes it possible to highlight horizontal edges. Applying D_x and D_y recursively, we obtain D_{xx}, D_{yy} and D_{xy}. These operators emulate the functionality of the V1 region presented in the primary visual cortex.

Functions	Description	Terminals	Description
Element-wise operators			
$A + B$, $A - B$, $A \times B$, A/B, $k + A$, $k - A$, $k \times A$, A/k, $\|A\|$, $\|A + B\|$, $\|A - B\|$, $log(A)$, $(A)^2$, $\sqrt{A}$, $round(A)$, $\lfloor A \rfloor$, $\lceil A \rceil$, $inf(A, B)$, $sup(A, B)$, $thr(A)$	Arithmetic functions between images or constants k, absolute values, trascendental functions, square, square root, rounding functions, infimum, supremum, and threshold applied to images A and/or B	I_r, I_g, I_b, I_c, I_m, I_y, I_k, I_h, I_s, I_v, $D_x(I_x)$, $D_{xx}(I_x)$, $D_y(I_x)$, $D_{yy}(I_x)$, $D_{xy}(I_x)$	Elements of I_{color} and its derivatives
Convolution operators			
$G_{\sigma=1}(A)$, $G_{\sigma=2}(A)$, $D_x(A)$, $D_y(A)$	Convolution with a Gaussian filter, and derivatives applied to image A		

Table 2.1: Functions and terminal list for the visual operator VO_O.

The second operator encodes the color dimension emulating the color-sensitive cells in the visual cortex. The visual operator of color(VO_C) reproduces the color perception process to find prominent regions with color properties in the image. Note that some functions of VO_C are the same as those in VO_O plus the function $complement()$ that provides a negative image that complements an intensity or RGB value (see Table 2.2). Regarding the output image, dark areas become lighter, and light areas become dark. Opponent terminals perform a fixed operation between the color bands that build a new image with maximum values. For example, $Op_{\ r,g}$ accentuates the difference between the red and green bands.

Functions	Description	Terminals	Description
Element-wise operators			
$A+B$, $A-B$, $A\times B$, A/B, $k+A$, $k-A$, $k\times A$, A/k, $log(A)$, $exp(A)$, $(A)^2$, $\sqrt{A}$, $(A)^c$, $round(A)$, $\lfloor A \rfloor$, $\lceil A \rceil$, $thr(A)$	Arithmetic functions between images or constants k, trascendental functions, square, square root, image complement, rounding functions and threshold applied to images A and/or B	I_r, I_g, I_b, I_c, I_m, I_y, I_k, I_h, I_s, I_v, $Op_{r-g}(I)$, $Op_{b-y}(I)$	Elements of I_{color} and color opponencies: red-green and blue-yellow

Table 2.2: Functions and terminal list for the visual operator VO_C.

The third tree is the visual operator of shape. The method that extracts visual information from the object's shape employing VO_S from Table 2.3 utilizes the artifacts' morphological information in the image. BP proposes to create compound operators by the composition of basic morphological operators such as dilation, erosion, open, close with disk, square, and diamond structural elements. Indeed, it is possible to create more complex operators from these operators. The goal of extracting shape information is to highlight valuable information for object recognition.

Functions	Description	Terminals	Description
Element-wise operators			
$A+B$, $A-B$, $A\times B$, A/B, $k+A$, $k-A$, $k\times A$, A/k, $round(A)$, $\lfloor A \rfloor$, $\lceil A \rceil$, $thr(A)$	Arithmetic functions between images or constants k, rounding functions, and threshold	I_r, I_g, I_b, I_c, I_m, I_y, I_k, I_h, I_s, I_v	Elements of I_{color}
Morphological operators			
$A \oplus SE_d$, $A \oplus SE_s$, $A \oplus SE_{dm}$, $A \ominus SE_d$, $A \ominus SE_s$, $A \ominus SE_{dm}$, $A \circledcirc SE_s$, $A \odot SE_s$, $Sk(A)$, $Perim(A)$, $A \circledast SE_d$, $A \circledast SE_s$, $A \circledast SE_{dm}$, $T_{hat}(A)$, $B_{hat}(A)$	Dilation, erosion, open, close with disk, square, and diamond structural element; skeleton, hit or miss, bottom-hat, and top-hat		

Table 2.3: Functions and terminal list for the visual operator VO_S.

Finally, the intensity measure corresponds to the amount of light perceived by a photosensitive device. In humans, specialized ganglion cells in the retina measure the intensity. Then, the following formula is applied to compute the visual map of intensity.

$$VM_{Int} = \frac{I_r + I_g + I_b}{3} \quad . \tag{2.7}$$

- Conspicuity Maps

The following procedure is the center-surround process; it efficiently combines the VMs and helps detect scale invariance in each of the dimensions. This process applies a Gaussian smoothing over its corresponding VM_d at nine scales $P_d^{\sigma} = \{P_d^{\sigma=0}, P_d^{\sigma=1}, ..., P_d^{\sigma=7}, P_d^{\sigma=8}\}$; this processing reduces the visual map's size by half on each level forming a pyramid. Subsequently, the six levels of the pyramid are extracted and combined.

$$Q_d^j = P_d^{\sigma=\lfloor \frac{j+9}{2} \rfloor + 1} - P_d^{\sigma=\lfloor \frac{j+2}{2} \rfloor + 1} \quad , \tag{2.8}$$

where $j = 1, 2, ..., 6$. Since the levels P_d^{σ} have different sizes, each level is normalized and scaled to the visual map's dimension using polynomial interpolation. This technique emulates the center-surround process of the biological system. After extracting features, the brain receives stimuli from the vision center and compares it with the receptive field's surrounding information. The goal is to process the images so that the results are independent of scale changes. The entire process ensures that the image regions are responding to the indicated area. The algorithm computes this process for each characteristic dimension (VM_d); the results are the Conspicuity Maps (CM), focusing only on the searched object by highlighting the most salient features. This early stage of the system follows the psychological model of visual attention, which involves the objects' location in space as the artificial dorsal stream pathway.

- Mental Maps

After obtaining the most saliency features, the next stage along the AVC is to compute the Mental Maps (MMs) to define a compound descriptor vector used as input to a classifier for categorization purposes. This procedure is analogous to the artificial ventral stream pathway. The synthesized information from CMs enters the set of MMs, which discriminates against unwanted information. The AVC model uses a set-of-functions to extract the images' discriminant characteristics (see Table 2.4); it uses a functional approach. The algorithm applies a set of k VOs to the CMs for the construction of the MMs. These VOs correlate with the remaining trees of the individual representation corresponding to the last step in the feature descriptor construction.

Unlike the specialized VOs used to obtain the CMs, the algorithm's next step is to simulate the ventral stream known as the "what" stream, which is mainly associated with object recognition and shape representation. The idea is to concatenate the highest values per dimension (four) into a single vector of n global maxima using Equation (2.9), where d is the dimension, and k represents the cardinality of the set of VO_{MM_k}. Therefore, MMs uses the same VOs set in all dimensions to gather the salience information obtained from the CMs to construct the descriptor that characterizes the object of interest.

$$MM_d = \sum_{i=1}^{k} VO_{MM_i}\left(CM_d\right) \tag{2.9}$$

Functions	Description	Terminals	Description
Element-wise operators			
$A+B$, $A-B$, $A \times B$, A/B, $\lvert A+B \rvert$, $\lvert A-B \rvert$, $log(A)$, $(A)^2$, $\sqrt{A}$	Arithmetic functions between images or constants k, absolute values, transcendental functions, square, and square root	CM_d, $D_x(CM_d)$, $D_{xx}(CM_d)$, $D_y(CM_d)$, $D_{yy}(CM_d)$, $D_{xy}(CM_d)$	Conspicuity Maps and its derivatives
Convolution operators			
$G_{\sigma=1}(A)$, $G_{\sigma=2}(A)$, $D_x(A)$, $D_y(A)$	Convolution with a Gaussian filter, and derivatives applied to image A		

Table 2.4: Functions and terminal list for the set VO_{MM}.

- Genetic Operations

The imitation of Darwin's natural selection consists of assigning to each solution a selection probability proportional to their fitness value while preferring smaller trees when fitness is equal. Individuals are selected from the population using a tournament with lexicographic parsimony pressure [94] to participate in the genetic recombination from the individuals' multi-tree representation. The algorithm retains the best individuals after applying genetic operators to create the new offspring.

Like genetic algorithms, BP executes the crossover between two selected parents at the chromosome level using a "cut-and-splice" crossover. We consider the entire individual, the array of $EVOs$, similar to a chromosome. Each operator within the chromosome is a gene. Each function or terminal is analogous to the nucleotides from the gene anatomy. The algorithm swaps all data beyond the selected crossover point between both parents A and B. The result of applying a crossover at the gene level is performed by randomly selecting two subtree crossover points between both parents. The selected genes match with the corresponding subtree in the other parent. The chromosome level mutation leads to selecting a given parent's random gene to replace such substructure with a new randomly mutated gene. The algorithm calculates the mutation at the gene level by applying a subtree mutation to a probabilistically selected gene; the subtree after that point is removed and replaced with a new subtree. These genetic operators allow the variation of the genetic material while promoting individuals' genetic innovation through all levels and maintaining the diversity of the population.

Fitness Function

The following stage in the model is the construction of the image descriptor vector (DV). The system concatenates the four MMs and uses a max operation to extract the n highest values; these values are used to construct the DV. Once we get the DVs from images in the database, a classifier associates the domain given by the descriptors to the labels' codomain. In this work, we use an SVM working with the discriminate hyperplane defined by:

$$f\left(x\right)=\sum_{i=1}^{l}\alpha_i y_i K\left(x_i,x\right)+b, \tag{2.10}$$

where the given training data is $(\mathbf{x}_i, y_i)$, $i=1,\ldots,l$, $y_i \in \{-1,1\}$, $\mathbf{x}_i \in \mathbf{R}^p$ 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 locates the margin between the class and non-class as the search criteria. The accuracy obtained by the SVM indicates the fitness of the individual, which minimize the learning problem from Equation (2.5) †.

Initialization, GP parameters, and Solution Designation

Once we define the AVC structure from each individual, we set the parameters of the BP evolutionary process (see Table 2.5) and establish the image database. Next, the algorithm creates a random initial population using a ramped half-and-half technique, selecting half of the individuals with the grow method and half with the full method. According to the maximum initial depth, the full method makes balanced trees, while the grow method makes unbalanced trees allowing branches of varying lengths. Here we set a limit of maximum depth to avoid uncontrolled growth of trees over time. The algorithm dynamically sets the tree depth using two maximum values to limit the individual's size within the population. The dynamic max depth is a maximum value that may not be surpassed unless the individual's fitness is better than the best solution found so far. If it occurs, the dynamic max depth value is updated to the new fittest individual. The real max depth is a hard limit that no individual may surpass under any circumstances. Selection uses a tournament with lexicographic parsimony pressure for keeping the best individual. Finally, the algorithm terminates the evolutionary process when it reaches one of

[†] In this section, The meaning of accuracy has the purpose of optimizing BP; nevertheless, the accuracy indicated in Section 4.1 refers to the metric to measure the attack responses.

these two conditions: 1) an acceptable classification rate or 2) the total number of generations. Thus, the evolutionary process reaches an optimum population that contains the best solution to the problem.

Parameters	Description
Generations	30
Initial Population	30
Crossover at chromosome level	0.4
Crossover at gene level	0.4
Mutation at chromosome level	0.1
Mutation at gene level	0.1
Tree depth	Dynamic depth selection
Dynamic max depth	7 levels
Real max depth	9 levels
Selection	Tournament with lexicographic parsimony pressure
Survival	Elitism

Table 2.5: Initialization parameters for each GP applied in the BP algorithm.

2.3.2 Hands-on Artificial Evolution

The use of random principles is overused in evolutionary computation. Thus, Olague and Chan-Ley adopted a methodology to avoid the unnecessary application of arbitrary or unplanned solutions within an algorithm to advance towards a more goal-oriented methodology [95]. It is not feasible to leave a methodology to discover the best solution when they have complex structures, but helping it with previous discoveries will guide the search in a better direction. Hence, the idea was to use the best solutions discovered during previous searches as the initial population to set a new experiment to find a better solution. Most of the time, this strategy improves the performance of an algorithm to be competitive in a task. In this research work, it is employed in Section 5.2.4.

Brain programming is a highly demanding computational paradigm. A balance should be found to create programs that can solve non-trivial problems within the state-of-the-art in a reasonable amount of time. The idea is to continue the evolution from the best local minimum discovered so far. The proposed technique significantly improves the performance of previous results. The idea of hands-on evolution works for computationally demanding problems. It is a simple strategy that saves computational time because this kind of results cannot be obtained by simply continuing the random initial population's approach.

3
Adversarial Attacks

Adversarial attacks are a severe threat to the exemplar performance of DCNN. Therefore, we outlined the explanation of how these attacks works. First, given an input image $\mathbf{x}$ in an input subspace $\mathbf{X}$ such that $\mathbf{x} \in \mathbf{X}$ and its corresponding label y, DCNN model establishes a relationship within the data using the following equation:

$$y = f(\mathbf{x}) = A(\mathbf{w}^{\intercal}\mathbf{x}) \quad , \tag{3.1}$$

where function $f(\cdot)$ is the DCNN model, whose associated weights parameters are $\mathbf{w}$ and $A(\cdot)$ is an activation function. However, an erroneous behavior is notable when the input image suffers a small change in its pixels $\mathbf{x}_{\boldsymbol{\rho}} = \mathbf{x} + \boldsymbol{\rho}$ such that:

$$f(\mathbf{x}) \neq f(\mathbf{x}_{\boldsymbol{\rho}}) \;\; \mathbf{s.t.} \;\; ||\mathbf{x} - \mathbf{x}_{\boldsymbol{\rho}}||_p < \alpha \tag{3.2}$$

where $p \in N \mid p \geq 0$, $\alpha \in R \mid \alpha \geq 0$, $||\cdot||_p$ denotes the l_p-norm, which the most commonly used are $||\cdot||_0$, $||\cdot||_2$ and $||\cdot||_\infty$. The scalar value α limits the image modification depending on the l_p-norm and the design of the attack. For example, Fast Gradient Sign Method [12] limits the intensity of the attack by using standard ϵ values that restricts the image's modification to do not overpass the norm. In the predecessor of the adversarial patch attack [57] uses $\alpha = 2000$ for l_2-norm and $\alpha = 10$ for l_∞-norm. So, it can be defined an Adversarial Example (AE) as an intentionally modified input $\mathbf{x}_{\boldsymbol{\rho}}$

that is classified differently than $\mathbf{x}$ by the DCNN model, with a limited level of change in the pixels of $||\mathbf{x} - \mathbf{x}_{\boldsymbol{\rho}}||_p < \alpha$, so that it may be imperceptible to a human eye.

The simplest explanation of how AEs work to attack a DCNN is that most digital images use 8-bit per channel per pixel. So, each step of 1/255 limits the data representation; thus, information in between is unused. Therefore, if every element of a perturbation $\boldsymbol{\rho}$ is smaller than the data resolution, it is coherent for the linear model to predict distinct given an input $\mathbf{x}$ than to an adversarial input $\mathbf{x}_{\boldsymbol{\rho}} = \mathbf{x} + \boldsymbol{\rho}$. We assume that forasmuch as $||\boldsymbol{\rho}||_\infty < \alpha$, where α is too small to be discarded, the classifiers should predict the same class to $\mathbf{x}$ and $\mathbf{x}_{\boldsymbol{\rho}}$.

Nonetheless, after applying the weight matrix $\mathbf{w} \in \mathbf{R}^{M \times N}$ to the AE, we obtain the dot product defined by $\mathbf{w}^\intercal \mathbf{x}_{\boldsymbol{\rho}} = \mathbf{w}^\intercal \mathbf{x} + \mathbf{w}^\intercal \boldsymbol{\rho}$. The AE will grow the activation by $\mathbf{w}^\intercal \boldsymbol{\rho}$. Note that the dimensionality of the problem does not grow with $||\boldsymbol{\rho}||_\infty$; thus, the activation change caused by perturbation $\boldsymbol{\rho}$ can grow linearly with n. In high dimensional problems, the numerous imperceptible changes to the input sum to obtain immense output changes.

The linear interpretation of AEs implies easy ways to generate them. Authors from [12] hypothesize that neural networks are too linear to resist linear perturbations. Networks such as long short-term memory, ReLUs, and maxout are intentionally designed to perform linearly so that they are easier to optimize. Moreover, nonlinear models such as sigmoid networks work in the non-saturating state, becoming more like a linear model. Hence, every perturbation as challenging or straightforward to compute of a linear model should affect DCNNs. Therefore, when a model is affected by an AE, this image often affects another model, whether the two models have different architectures or were trained with other databases. Then, they only have to be set up for the same task to change the result [12].

In this manner, the AE generation finds an image $\mathbf{x}_{\boldsymbol{\rho}}$ in the input subspace $\mathbf{X}'$ such that $\mathbf{x}_{\boldsymbol{\rho}} \in \mathbf{X}'$ and $f(\mathbf{x}) \neq f(\mathbf{x}_{\boldsymbol{\rho}})$. Nevertheless, we denote robustness in terms of function continuity. Given a model's function $f()$ in an image input subspace $\mathbf{X}$ is said to be robust at $\mathbf{x} \in \mathbf{X}$, if $\mathbf{x}_k \rightarrow \mathbf{x}$ then $f(\mathbf{x}_k) \rightarrow f(\mathbf{x})$. Equivalently, $f(\mathbf{x})$ is robust at $\mathbf{x}$, for all $\mathbf{y} \in \mathbf{X}$, if given a $\epsilon > 0$, there is a $\delta > 0$ such that $||\mathbf{y} - \mathbf{x}|| < \delta$ implies $|f(\mathbf{y}) - f(\mathbf{x})| < \epsilon$. Hence, if $f(\mathbf{x})$ is robust for every $\mathbf{x}$, then $f(\mathbf{x})$ is said to be robust on $\mathbf{X}$.

Adversarial attacks are usually established as constraint optimization problems. The objective is to find a perturbation ϵ such that $f(\mathbf{x} + \epsilon)$ predicts $y_t \neq y_{original}$, where $\mathbf{x}$ is the input image which is classified as $f(\mathbf{x})$, and f is the target image classifier. The perturbation ϵ is limited to be as imperceptible as possible with maximum modification constraint L measured by the length of vector ϵ. For targeted attack, y_t is an specified target class, and for non targeted attack, y_t is not specified, as long as it is not the correct label. Therefore, targeted attacks find an optimal solution $\epsilon*$ for the following equation:

$$\begin{aligned} \min_{\epsilon*} \quad & J(f(\mathbf{x} + \epsilon), y_t) \\ \text{s.t.} \quad & ||\epsilon|| \leq L \quad . \end{aligned} \tag{3.3}$$

It minimizes the cost function J over the target class y_t. In a non-targeted attack, the goal is to find a perturbation $\epsilon*$ that maximizes the cost function's values J over the original predicted class $y_{original}$.

That means to minimize the probability of the class $y_{original}$, and the optimization is defined as follows:

$$\begin{aligned} \max_{\epsilon *} \quad & J(f(\mathbf{x}+\epsilon), y_{original}) \\ \text{s.t.} \quad & ||\epsilon|| \leq L \quad . \end{aligned} \tag{3.4}$$

Adversarial attacks are classified according to the model's available information and the desired attack to predict a specific class. The literature [13] divides attacks into targeted and untargeted approaches. Targeted attacks refer to the ability to fool a model with a specific label, while untargeted attacks induce an error without a precise label. Also, the literature refers to white-box attacks where it is assumed complete knowledge of the model, including parameter values, architecture, training method, and sometimes training data. Finally, a black-box attack feeds a targeted model with adversarial examples without the model's knowledge.

This thesis analyzes in Section 3.1 a white box untargeted attack (Fast Gradient Sign Method) to determine the impact of an easy and direct threat to DCNN by knowing its parameters. We study the AE transferability property, which means generating AEs and performing an attack with the misclassification on DL systems with no access to the model, extending the analysis to different architectures like BP and SIFT+FV. In Section 3.2, a black box untargeted attack (multiple pixel attack) analyzes the hazard from an attack that tries to find locations and pixel values to build a perturbation that changes the model's prediction from an artwork image. In Section 3.3 a targeted attack is analyzed, which uses an adversarial patch to challenge the robustness of such modified image patches, which can be rotated, put on random locations, and printed to appear in real-world conditions in the artwork to cause a misleading prediction of the target class. We also analyze the AE transferability of such patches through all models. In Section 3.4, a white box untargeted attack (facial accessories perturbations) that utilizes a physically realizable and inconspicuous pair of eyeglass frames to evade recognition is detailed. Lastly, in Section 3.5 we include an analysis of a DCNN defense mechanism to test a solution proposed to the AA problem. We verify the feasibility of using a defense mechanism in the real world to make DCNN secure. We detailed each of the adversarial attacks and the defense mechanism mentioned above in the following sections.

3.1 Fast Gradient Sign Method

The Fast Gradient Sign Method proposed in [12] is the most widely used method for computing AEs due to its easy implementation. The FGSM exploits the gradient, which is the weights correction in the backpropagation process of a neural network to build an adversarial example. FGSM computes the gradient of a loss function with respect to the input image and then uses the sign of the gradient to create the image that maximizes the loss. FGSM proposes to increase the loss of the classifier by solving the following equation: $\rho = \epsilon \ \text{sign}(\nabla J(\theta, \mathbf{x}, y_l))$, where $\nabla J()$ computes the gradient of the cost function around the current value of the model parameters θ with the respect to the image $\mathbf{x}$, and the target label y_l. sign() denotes the sign function, which maximizes the magnitude of the loss and ϵ is a small scalar value that restricts the norm L_∞ of the perturbation.

The perturbations generated by FGSM take advantage of the linearity of the DL models in the higher dimensional space to make the model misclassify the image. The linearity of DL models discovered by FSGM implies the transferability between models. Authors in [96] reported that with the ImageNet dataset, the top-1 error rate using the perturbations generated by FGSM is around 63-69% for $\epsilon \in [2, 32]$. Figure 3.1 illustrates AEs from the FGSM where the intensity of the perturbation is controlled with ϵ. We observed that even for the largest ϵ value is difficult to notice the perturbation.

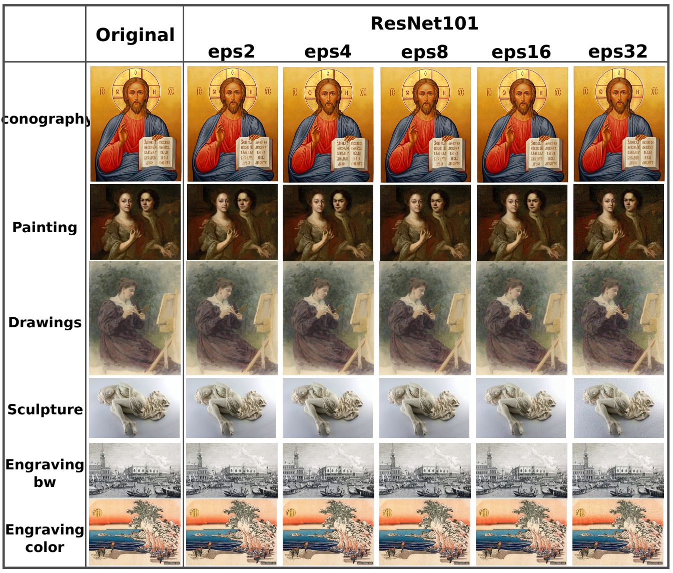


Figure 3.1: Example images of computing the FGSM using ResNet101 from each class with a scale factor of $\epsilon = 2, 4, 8, 16, 32$.

3.2 One Pixel Attack

The one-pixel attack considers a minimal scenario where only one pixel is changed in the image to fool the DL models using images of a reduced size of 32×32 pixels. With these limitations, Su et al. successfully fool three different CNN models on 67.97% of the testing images with the modification of just one pixel per image [66]. Furthermore, the authors reported that the average confidence of the CNNs on the wrong prediction on the pictures was 97.47%.

The one-pixel adversarial perturbations are black-box attacks since they do not require knowledge of the model. The attack uses a population-based optimization algorithm for solving complex multi-modal optimization problems named Differential Evolution [97] to generate the damage. First, the method searches a solution from a vector space $\mathbb{R}^5$ that contains (*x*,*y*) coordinates limited by the image size and the three bands of the RGB color values. Then, within a population, it randomly modifies the five-dimensional individuals' elements to create new offspring such that they compete in the current iteration to obtain better fitness. For example, in the case of two pixels, an individual has a vector space $\mathbb{R}^{10}$ that contains a pair of (*x*,*y*) coordinates and a pair of RGB colors values, and so on for individuals with more pixels. During the run, the algorithm used the probability of the predicted label to compute the fitness criterion. Finally, the last surviving individual is used to modify the pixels in the image.

In summary, let the vector $\mathbf{x} = (x_1, \ldots, x_n)$ be a *n*-dimensional image, which is the input of the target classifier f that predict correctly the class *t* from the image. The probability of $\mathbf{x}$ associated to the class *t* is $f_t(\mathbf{x})$. It builds an additive adversarial perturbation vector $e(\mathbf{x}) = (e_1, \ldots, e_n)$ according to $\mathbf{x}$ and the limitation of maximum modifications *d*, a small number that express the dimensions that are modified while other dimensions of $e(\mathbf{x})$ left as zeros. For untargeted attacks, the main purpose is to find the optimal solution $e(\mathbf{x})*$ that solves the following equation:

$$\begin{aligned} \min_{e(\mathbf{x})*} \quad & f_t(\mathbf{x} + e(\mathbf{x})) \\ \text{s.t.} \quad & ||e(\mathbf{x})||_0 \leq d \quad . \end{aligned} \tag{3.5}$$

The case of a one-pixel attack is $d = 1$, and it is possible to extend it to multiple pixels by increasing d. Note that the one-pixel attack was tested initially on DL models with inputs from CIFAR 10 dataset. So, it represents a considerable modification of such tiny images; nevertheless, it is insignificant with the databases studied in the present work. Therefore, we use a multiple-pixel attack $d >> 1$ in order to work with real-size images. Notice that increasing the number of pixels in this attack will raise the perturbation risk to be noticeable. In Figure 3.2, we illustrate AEs from the multiple-pixel attack where the perturbation can be noticeable.

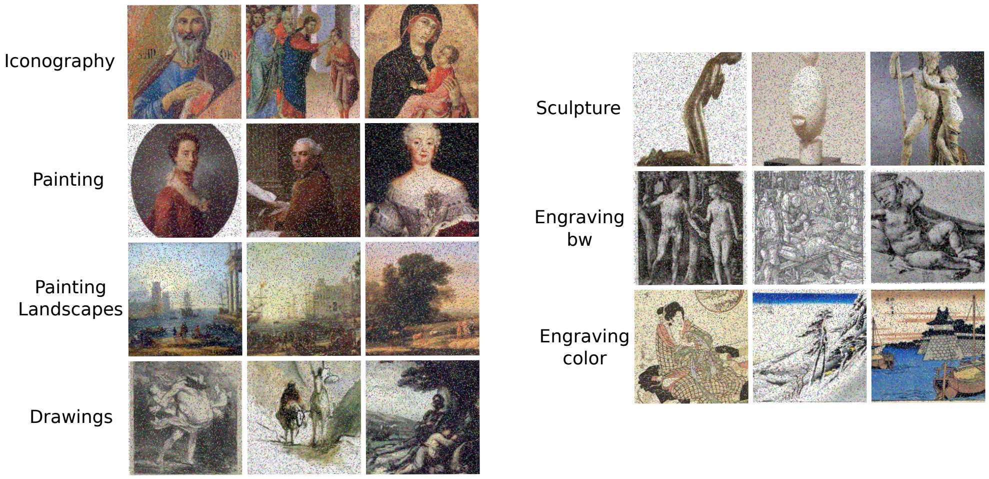


Figure 3.2: Example images of the multiple pixel attack using $d = 10,000$ for each class. Each column shows three sample images from the Wikiart database.

3.3 Adversarial Patch

The adversarial patch, in contrast to the traditional strategy (FGSM), is a method to replace a perturbation on the whole image with a patch, see Figure 3.3. The adversarial patch consists of creating universal, robust, targeted adversarial image patches by finding a perturbation $\hat{p}$ that maximizes $f_{target}(\mathbf{x} + \hat{p}))$. The robustness of these patches resides on the wide variety of transformations on which they can attack any image and target the classifier prediction to the desired class. Also, they work in real work environments where they can be printed, photographed, or even when the patch is too small; they can ignore the whole scene to predict the target class.

The method builds a patch $\hat{p}$, using a variant of the Expectation over Transformation (EOT) framework, for which the algorithm trains the patch to optimize the following equation:

$$\hat{p} = \arg\max_{\hat{p}} \mathbb{E}_{x \in X. t \in T. l \in L}[\log f(y_{target}, A(p, \mathbf{x}, l, t))] \quad , \tag{3.6}$$

where X is a training set of images, T is a distribution over transformations of the patch, L is a distribution over locations in the image, y_{target} is the target label, and $\mathbf{x}$ the image vector. The expectation over the training images improves the patch's effectiveness, regardless of the background. It was proved by [98] the patch's universality using several images with different backgrounds. A variation of this method is to add a constraint of the form $||p - p_{orig}||_\infty < \epsilon$ to the patch objective in order to camouflage it. The constraint enforces the final patch to be within ϵ in the L_∞ norm of some starting patch p_{orig}.

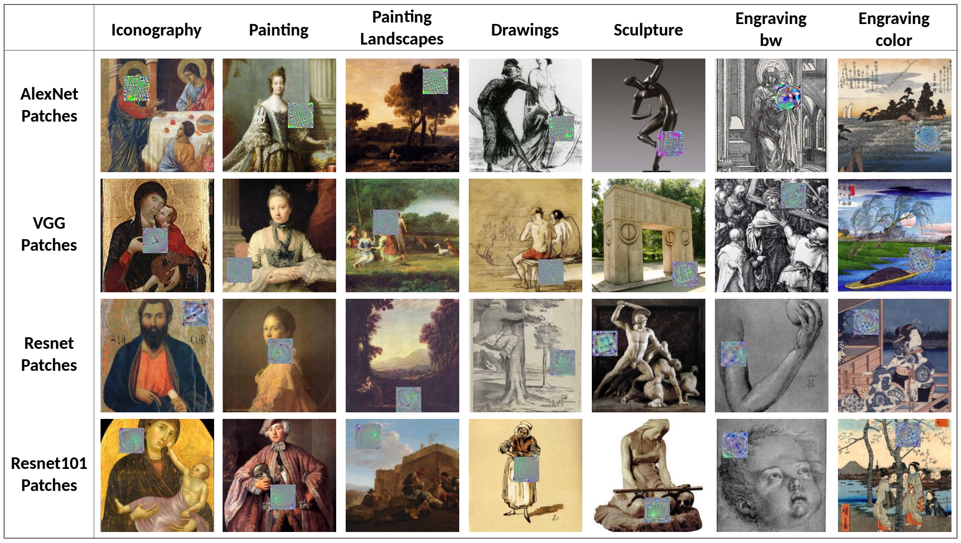


Figure 3.3: Example images of the adversarial patch. Each column represents the classes from the Wikiart database, and the rows show the adversarial patches obtained with the corresponding DCNN models.

3.4 Facial Accessories Perturbations

The Facial Accessories Perturbations infer that the attack is made with the targeted model's knowledge to evade the recognition [99]. In this manner, facial accessories are used to perform the attacks, which in this case are eyeglasses frames. The advantage of facial accessories is that they can be easily realizable in real-world conditions. Furthermore, eyeglasses are an everyday facial accessory that is natural for people to wear, helping the attacks be feasible.

Hence, a set of eyeglasses frames is employed to physically realize the attack, ensuring that the perturbation effectively misclassifies more than one image. In order to find a perturbation that performs the attack, the following optimization problem needs a solution:

$$\min_{\rho} \sum_{x \in X} -softmaxloss(f(x+\rho), c_x), \tag{3.7}$$

where the perturbation ρ would maximize the $softmaxloss(f(x+\rho), c_x)$ value to minimize the probability of the class c_x. To guarantee the generality of perturbations, we need to look for complex models that can cause any image in a set of inputs to be misclassified. Hence, the attack requires a set of images, X, and finds a single perturbation that optimizes her objective for every image $x \in X$. Figure 3.4 shows example images from the eyeglasses frame perturbation.

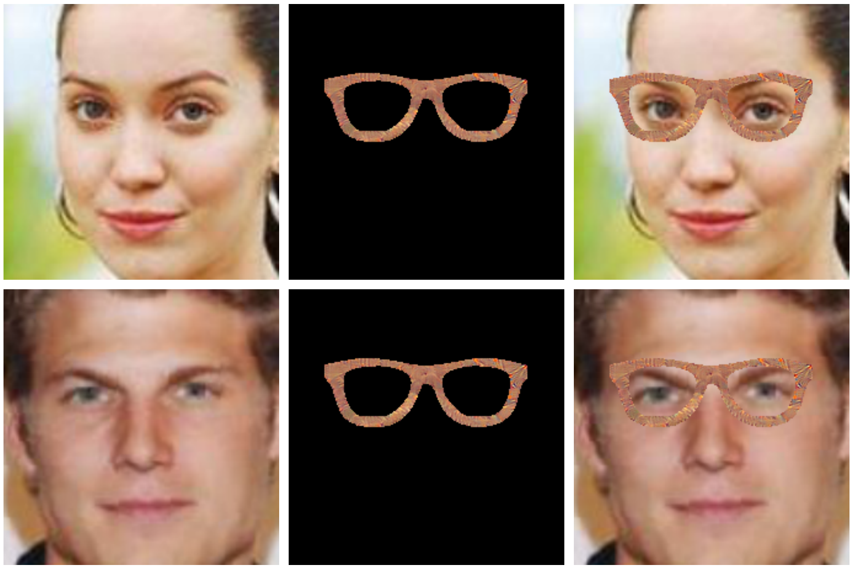


Figure 3.4: These images illustrate adversarial examples computed with the facial accessories perturbations. The first column shows the clean face images, the second column presents the precomputed ResNet glasses frame, and the third column shows the resulting adversarial examples.

3.5 Defense Mechanisms

Currently, the defenses against the adversarial attacks are being developed along with three main directions [13]:

- Modifying the training process during learning or the input during testing.
- Changing the structure of the networks, e.g., by adding more layers/subnetworks, changing loss/activation functions, and so on.
- Employing external resources as network add-on when classifying unseen examples.

Adversarial training is the process of explicitly training a model on adversarial examples to make it more robust to attack or reduce its test error on clean inputs (See Algorithm 2). The most common method used is to modify the training process during learning because there has been a consensus in the literature that the robustness of neural networks is improved against adversarial examples with adversarial training. For example, in the research works [11, 12] besides the introduction of new AAs, they propose using AEs generated by these methods in the learning process as the first line of defense against those attacks.

Although adversarial training helps make a network robust to AAs, it is a non-adaptive strategy requiring training to be performed using solid attacks. The results from the adversarial training have been commonly observed in the literature [12, 100] in regularizing the network to reduce over-fitting, which improves the robustness against the AAs. Even though few works mention that AAs may not be a serious concern on DL, a large number of research works indicates otherwise [13].

The profound implications of the vulnerability of deep neural networks to adversarial perturbations have made a highly active research area in AAs and their defenses. For example, Authors from [101] demonstrated that ten different defenses against AEs could be defeated by new attacks designed using different loss functions. At the same time, techniques are being proposed to defend deep neural networks against the known AA, but more complex and compelling attacks are being designed. Therefore, it is definitely that the problem of AAs has not been solved yet to became DL a secure method. We need to verify if the solutions provided to solve the problem of AAs make a real difference to make DCNN trustworthy.

In this Chapter, we have presented the security concerns about DCNN models that their predictions can be manipulated with small perturbations to the image to change their behavior. Also, we show how these attacks work to fool the DCNN models and the transferability effect that causes to different architectures to be affected by these perturbations. Four kind of attacks were detailed in this Chapter: 1) FGSM (white-box untargeted), 2) Multiple-pixel attack (black-box untargeted), 3) Adversarial patch (white-box targeted), and 4) Facial accessories perturbations (white-box targeted). Finally, the defense mechanisms that the research community has developed were explained to notice that the problem of adversarial attacks is still unsolvable.

Algorithm 2: Adversarial training for network *N*
Size of the training minibatch is *m*. Number of adversarial images in the minibatch is *k*. Procedure extracted from [96]

```
Randomly initialize network N
repeat
    Read minibatch $B = \{X^1, \dots, X^m\}$ from training set
    Generate k adversarial examples $\{X^1_{adv}, \dots, X^k_{adv}\}$ from corresponding clean
      examples $\{X^1, ..., X^k\}$ using current state of the network N
    Make new minibatch $B_0 = \{X^1_{adv}, \dots, X^k_{adv}, X^{k+1}, \dots, X^m\}$
    Do one training step of network N using minibatch $B_0$
until training converged;
```

4

Robustness Evaluation to Adversarial Examples

This chapter presents the novel robustness evaluation methodology proposed in this thesis. Figure 4.1 illustrates at the top of the image the traditional methodology to evaluate robustness against adversarial attacks, which is limited to deep learning models because this vulnerability has been studied only in the context of these approaches. In this methodology, several models are selected to be competitive in a task. After that, different attacks are considered to evaluate the robustness through metrics that quantify the performance's change. However, the proposed methodology extends the robustness evaluation beyond the deep learning systems (see Figure 4.1 at the bottom of the image). We consider to submit a competitive method against deep learning in a defined task in the first steps. Consequently, we contemplate the threats generalization by choosing contrasting attack strategies (e.g., white-box, black-box, targeted, and untargeted). Standard metrics to rate the performance's change are used to determine the attack's damage, but we add a new step to measure collateral effects through a statistical analysis that measures the prediction's confidence variation.

Evaluation metrics play a crucial role in assessing the performance of classification models to demonstrate competence in a task. Typically the performance measure involves training a model on a dataset, using the model to make predictions on a different dataset not used during training, then comparing the predictions to the expected values in this separate dataset.

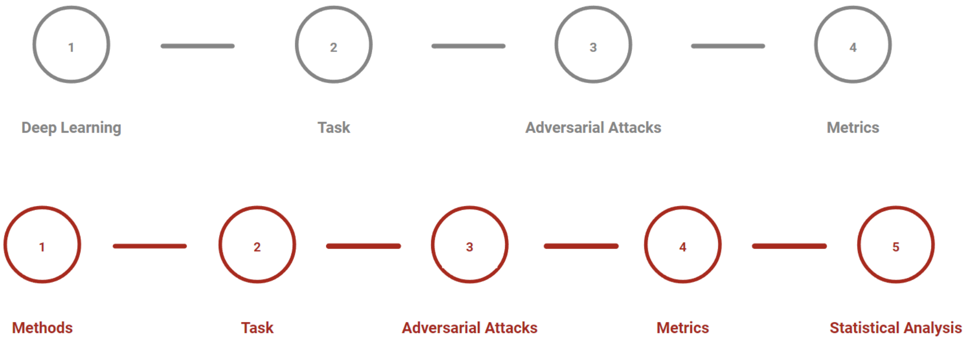


Figure 4.1: Comparative of the traditional methodology to evaluate robustness against adversarial attacks in deep learning at the top of the image, and the proposed methodology that extends the robustness evaluation beyond deep learning systems at the bottom of the image.

Choosing an appropriate metric is generally challenging in computer vision tasks but is particularly difficult for classification problems when adversarial attacks are present because most of the standard metrics that are widely used assume no intentions to fool the system. For classification problems, metrics involve comparing the expected class label to the predicted class label or interpreting the predicted confidence for the class labels for the problem. The most commonly used measure for this purpose is accuracy.

We analyze the algorithms' performance using standard metrics such as accuracy and the ratio between adversarial examples and clean images to quantify robustness. These metrics measure immediate changes in the algorithms' predicted labels. However, these measures do not consider the change in the predicted confidence for the class labels, which determines the foresee label. Therefore, we propose to use a series of statistical tests to measure significant differences in each classifier's predictions' confidence, which we considered an effect of the adversarial attack. Additionally, we use multiple comparisons of group means with the Bonferroni method between the performances from all the algorithms.

4.1 Standard Metrics

We employ classification accuracy as a measure of performance for the classifiers, which is simply the rate of correct classifications given by the following formula:

$$Accuracy = \frac{1}{N}\sum_{n=1}^{N} d(y'_n, y_n) \quad , \tag{4.1}$$

where N is the total of test images, y'_n is the predicted label for the image n, y_n is the original label for the image n, and $d(x,y) = 1$ if $x = y$ and 0 otherwise. Additionally, as a robustness measure, we used the accuracy ratio between adversarial examples and clean images implemented by [96] (see eq. 4.2). This metric means that if the ratio reaches one, the accuracy of AEs and the clean images is the same. Nevertheless, if it tends to zero, that means that the AA worked to fool the classifier. If this ratio exceeds 1, it implies that the AA is helping to correct misclassified images. The following equation calculates the ratio:

$$Ratio = \frac{acc_{adv}}{acc_{clean}} \quad , \tag{4.2}$$

where acc_{adv} is the classification accuracy on AEs, and acc_{clean} is the classification accuracy on the clean images

4.2 Statistical Analysis of Robustness

We see that differences among experiments seem striking, particularly when images suffer a subtle perturbation. Nevertheless, statistical analysis allows us to be more confident regarding the robustness of each method's predictions. Nowadays, the nonparametric statistical analysis is bringing researchers' attention to measure the performance through a rigorous comparison among algorithms, considering independence, normality, and homoscedasticity [102, 103]. Such procedures perform both pairwise and multiple comparisons for multiple-problem analysis. In our case, we apply pairwise statistical procedures to perform individual comparisons between each method's predictions' confidence from clean and attacked images based on the statistical procedure described in [104].

When the designed algorithms' results for the same problem achieved the conditions expressed before, the most common test is the ANOVA. In case that the distributions are not normal, we must use a nonparametric test like Kruskal-Wallis. If the distributions are normal but do not achieve the property of homoscedasticity, the analysis required is the Welch test. The statistical tests enable comparisons of the sample distributions, attending to the required conditions, and applying a suitable assessment a posteriori to contrast the results. As a result, we have first studied data normality (Lilliefors, Kolmogorov-Smirnov) and homoscedasticity (Levene test); then, according to the results, we have applied the appropriate statistical test (Kruskal-Wallis, Welch, Anova) to determine if the differences are significant, using a p-value < 0.05. Therefore, if the predictions' confidence is statistically different, it will illustrate the rejection of the null hypothesis Ho. If the statistical analysis accepts Ho, it will define that the predictions' confidence from the pair of clean and perturbed images is not significantly different; hence we can conclude that the method is robust to the AEs.

The Bonferroni method [105] can be used to compare different groups at the baseline, study the relationship between variables, or examine one or more endpoints in experiments. It is applied as a post-hoc test in many statistical procedures [106, 107]. The Bonferroni method is more rigorous than the Tukey test [108], which tolerates type I errors, and more generous than the very conservative Scheffé's method[109]. A simple main effect analysis between all classifiers using the Bonferroni method as a post-hoc test from the previous procedure considerations was employed with the FGSM testing data. Then, we use multiple comparisons to determine which group means are different from others using the Bonferroni method.

5
Experimental Results

In this chapter, we present the results from the evaluation to adversarial perturbations to determine the image classification models' robustness beyond machine learning systems. We explore to study robustness to adversarial attacks through the challenging image classification task of art media categorization and a first proposal to solve the face recognition task. We want to determine whether the hypothesis that the vulnerability of adversarial attacks is also affecting image classification methods beyond machine learning is valid. In this manner, we made an empirical study on the image classification robustness intersecting the state-of-the-art of image classification, adversarial attacks, art media categorization, and face recognition.

5.1 Art Media Categorization Problem

The Art Media Categorization (AMC) problem in CV has arisen from the increasing volumes of art databases publicly available to have automatic systems for identifying valuable artwork pieces. Therefore, recognizing art media from digital images has several important purposes. Its primary insight is to understand artworks through the analysis of complex features that can not be subjective as humans are prone to be [110]. For example, classifying fine art pieces involves a sophisticated selection of features that distinguish each medium, which is extremely difficult to find where usually an art expert analyzes the style, genre, and media from artworks to identify the artist and detect forgeries [111, 112, 25].

Furthermore, researchers use extracted features from art media to generate synthetic artistic effects in digital images [113, 114]. Therefore, the development of automated systems that makes an accurate and robust analysis of features to recognize artworks is a critical issue. In this manner, to make a complete analysis of art media, high-resolution images must provide enough information to maximize carefulness based on the artwork details. The art style, usually associated with the author's school, describes the artists' distinctive artifacts, visual elements, techniques, and methods. The form is related to the localization of features at different levels. For example, the classical hierarchy of genres ranks history-painting and portrait as high, while landscapes and still-life are classified as low because they did not contain persons.

Researchers studied AMC from three perspectives: 1) handcrafted feature extraction, 2) deep convolutional neural networks, and 3) genetic programming methodologies. First, handcrafted engineered features were the principal method to develop formulas that can extract features to obtain an image representation to classify an image effectively.

One of the first works that employ handcrafted features was [115]; here, the authors proposed a Discrete Cosine Transform (DCT) coefficients scheme used for feature extraction painter identification by classifying the artist's style. They build a custom database of approximately 300 grayscale images from five painters (Rembrandt, Van-Gogh, Picasso, Magritte, and Dali) to experiment. Li and Wang [116] proposed using a two-dimensional multi-resolution hidden Markov model to analyze brush strokes to provide reliable information to distinguish artists from ancient Chinese paintings. Their database consists of 276 grayscale images from five Chinese artists at a resolution of 3000×2000 pixels but scaled to 512 on the shorter dimension, maintaining the aspect ratio. Also, authors in [117] present a comparative study of different classification methodologies based on handcrafted engineered features. They contrasted semantic-level features with an SVM, color SIFT and opponent SIFT with BoV, and latent Dirichlet allocation with a generative BoV topic model for fine-art genre classification. In their study, a database of seven categories of paintings (Abstract, Baroque, Renaissance, Popart, Expressionism, Impressionism, and Cubism) was used from the Artchive fine-art dataset using 70 images from each class. Rosado [118] employed a BoV implemented using a dense-SIFT method for feature extraction and Probabilistic Latent Semantic Analysis (PLSA) make an image analysis of 434 digitized images from paintings, drawings, books, and engravings by Antoni Tàpies. In general, we note that using handcrafted engineered features makes it possible to obtain encouraging but not perfect results. Over time, the complexity of these characteristics started to become more challenging to design. In addition to the designing process of features, the learning algorithm development was a completely independent research area needed to match the feature extraction.

DCNN has been a breakthrough in many areas of image processing, and recent works on AMC have presented approaches based on state-of-the-art DCNN architectures. Authors in [119] introduced the use of deep convolutional activation features from a DCNN model trained for object recognition to recognize the style. These learned characteristics achieve high performance identifying styles in painting images and outperform most handcrafted engineered features. Bar et al. [120] proposed a compact binary representation combining the PiCoDes descriptors and the deep convolutional activation features from a DCNN model to identify artistic styles in paintings showing exceptional results to classify artwork images from WikiArt using 27 classes. Noord et al. [121] employed an adaptation of AlexNet to classify artwork styles from Rijks Museum images. They could visualize the regions with a heatmap from the artwork that impacts the prediction of style. Cetinic and Grgic [122] utilized the features extracted from VGG to classify WikiArt database images into seven genre classes such as portrait, landscape, city, still life, nude, flower, and animal. They outperform handcrafted engineered features such as SIFT, gist descriptor, HOG, Gray Level Co-occurrence Matrix (GLCM), and HSV color histograms with their classification method. Seguin et al. [123] propose to extract from VGG similar components shared by various artworks named visual link. These links try to find similitude from the paintings of the same creators or the same schools. The experiment used images from the Web Gallery of Art database reporting that their method achieves better performance than handcrafted engineered features such as SIFT.

Sun et al. [124] employed AlexNet and VGG to construct a structure with two pathways to obtain object and texture features. The DCNN performs the object computation, and the texture pathway uses the Gram matrices of intermediate features. Authors used in their experiments WikiPaintings, Flickr Style, and AVA Style databases. Elgammal et al. [125] proposed an analysis of strokes in line drawings using a database of 300 digitized drawings with over 80 thousand strokes. They employ handcrafted engineered features, deep learned features, and the combination of both to discriminate between artists at the stroke level with high accuracy. Also, their work serves to discover forgeries made by artists. Cetinic et al. [126] performed an extensive CNN fine-tuning experiment using five Caffe models (CaffeNet, Hybrid-CNN network, MemNet network, Sentiment network, and Flickr network) for five different art-related classification tasks (artist, genre, style, period, and association with a specific national artistic context) on three large fine art datasets (WikiArt, Web Gallery of Art, and TICC Printmaking Dataset). In [127], authors employed pre-train DCNN models (AlexNet, VGG, GoogLeNet, ResNet, DenseNet) to recognize basic artistic media from artworks. They collected about 1000 artwork images per class (oil-paint brush, pastel, pencil, and watercolor) through various search engines and websites to classify them. They obtained comparable results with that of trained humans.

Finally, a GP-like methodology called BP obtained competitive results compared to a DCNN model for the AMC task [25]. This technique aims to emulate the brain's behavior based on neuroscience learning processes with new symbolic learning via genetic programming. In the experiments, Chan-Ley and Olague use two renowned high-resolution artwork databases (Kaggle and WikiArt) to classify five art media classes (drawings, engraving, painting, iconography, and sculpture). The proposed technique achieves comparable results to AlexNet on a binary classification problem.

Although DCNN has obtained exemplary results in solving a wide variety of computer vision tasks, small perturbations named adversarial attacks on the input image turn the learning model's decision to change its prediction completely. Researchers generate these perturbations in several forms, including slight modifications to the input pixels and using spatial transformations, among others. These attacks' primary purpose is to fool the DL model's prediction intentionally and remain unnoticed to human perception. Szegedy et al. [11] were the first who discovered an unusual weakness where small perturbations almost invisible to the human vision on the input pixels can fool a CNN. These attacks also reported high confidence in the model's wrong prediction, and even worse, multiple networks were affected using the same perturbed image. Later, they found that CNN's robustness against AA could be improved using these images in the training phase. However, recent studies have highlighted the lack of robustness in well-trained DCNNs [128, 129]. Goodfellow et al. [12] designed a method named Fast Gradient Sign Method (FGSM), which enables efficient computing perturbations for a given image. Another threat consists of an extreme and straightforward attack proposed by Su et al. [66], which consists of modifying one pixel in the image to fool a CNN. A drawback, however, is that it only works for icon images.

Nevertheless, they successfully attacked three different network models under this strategy with high confidence. Moosavi-Dezfooli et al. [57] discovered singular perturbations that can misclassify any image; they called them universal perturbations. In this way, Brown et al. [98] proposed creating universal, robust, targeted adversarial image patches. These patches are so compact that they can be printed and used in real-world scenes to fool a CNN. Despite significant efforts in making defense methods against AAs, the research works have focused on modifying its training process or modifying the input image during testing [12, 58, 130], also on changing the structure of the networks [60, 61, 62] or through external models to classify unseen examples [63, 131]. Zhang et al. [132] discussed the limitation of the adversarial training because the attacks have become more and more challenging with high efficiency on the damage. Hence, it is difficult to fight against all the new and more complex AA. Even if DL architectures have classified large-scale sets of images with multiple classes with outstanding results, this paradigm's security concerns make the solutions unreliable. The brittleness is because there is a chance that hackers can intentionally fool such machine learning systems.

AMC is a complex problem to solve. Its solution involves a complicated analysis of features and demands accurate and robust decisions, primarily when curators work with precious art pieces. The performance of handcrafted engineered features methods does not compete with DCNN through their inability to extract complex features from the artworks to build a better image representation. DCNN has outperformed handmade methods and has established the leading for the AMC. Nevertheless, BP demonstrates its competence against DCNN performance in this area [25]. AA is an open research area that is a hot topic due to the risk of affecting DL architectures regardless of image databases because attacks have successfully worked in critical areas such as face recognition and object recognition for autonomous vehicles, among others[13]. Therefore, AMC is not the exception, and automation process designers need to avoid using vulnerable systems in critical tasks performed by art experts in museums and galleries, such as artist identification, art valuation, and forgery detection.

5.1.1 Experiments

Robust classification is a key characteristic regarding automatic system development, security, and confidence in art pieces' predictions. In this study, we analyze the algorithms' performance using accuracy. Besides, we use the accuracy ratio between adversarial examples and clean images to measure robustness. Moreover, we use a series of statistical analyses to corroborate the results. Firstly, we propose to determine significant differences in the change of each classifier's predictions' confidence. Secondly, we use multiple comparisons of group means with the Bonferroni method between the performances from all the algorithms. This experiment consists of studying the accuracy and robustness against AAs using three of the main approaches for image classifications:

- Traditional handcrafted features algorithm (SIFT+FV)

- Deep Genetic Programming Methodology (BP)

- DCNN models (AlexNet, VGG, ResNet18, and ResNet101).

We consider unconventional training, validation, and test datasets since we apply two different image databases compiled by experts for AMC. Training and validation datasets are constructed from the Kaggle database, while testing uses a standard database WikiArt (See Table 5.1). The aim is to emulate a real-world scenario where we test the best models against standard databases.

In this experiment, we analyze the threat of using three types of AA to the models mentioned above. The white box untargeted (FGSM) determines the impact from an easy and direct threat to DCNN by knowing its parameters. Also, we study the AE transferability property, which means generating AEs and performing an attack with the misclassification on DL systems with no access to the model, extending the analysis to different architectures like BP and SIFT+FV. We analyze the behavior of such perturbations from these architectures, which can cause wrong predictions with the addition of subtle texture to the artworks.

The black box untargeted (multiple-pixel attack) analyzes the hazard from an attack that tries to find locations and pixel values to build a perturbation that changes the model's prediction from an artwork image. The targeted attack uses the adversarial patch to challenge the robustness of such modified image patches, which can be rotated, put on random locations, and printed to appear in real-world conditions in the artwork to cause a misleading prediction of the target class. We also analyze the AE transferability of such patches through all models. Lastly, we include an analysis of a DCNN defense mechanism to test a solution proposed to the AA problem. We verify the feasibility of using these defense mechanisms in the real world to make DCNN secure.

5.1.2 Datasets

We use the same datasets from the experiment of AMC reported in [25]. We obtained training and validation images from the Kaggle website. This digitized artwork dataset comprises five categories of art media: drawing, painting, iconography, engraving, and sculpture. The engraving class consists of two different kinds; most of them were black and white art pieces. The other style was Japanese engravings, which introduce color to the images. We split the engraving class into engraving black and white and engraving color. We used a standard database obtained from WikiArt for testing, where we selected the images that match the same categories of Kaggle. Since Wikiart engraving is in grayscale, we use the ukiyo-e class (Japanese engravings) from Wikiart as the engraving color class. Also, the set of images of the category landscapes, which are painting from renowned artists, is added to test the evolved programs of the painting class. Table 5.1 provides the number of artworks for each dataset.

	Iconography	Painting	Drawings	Sculpture	Engraving BW	Engraving Color	Caltech Background
Train	1038	1021	553	868	426	30	233
Validation	1038	1021	553	868	283	19	233
Wikiart	251	2089	204	116	695	1167	233
Wikiart Landscapes		136					

Table 5.1: Total number of images per class obtained from Kaggle and Wikiart databases.

5.1.3 Implementation details

In this subsection, we outline the implementation details for all learned models:

- Brain Programming: was implemented on Matlab R2018b using a modified version of GP Lab [133] and the libsvm v3.25 [134], both are external Matlab libraries for genetic programming and support vector machines, respectively
- SIFT+FV: was implemented on Matlab(R2018b) using VLFeat v0.9.21 [135], which is an open source library that implements popular computer vision algorithms such as the SIFT description, GMM, and Fisher Vectors. It was used the SVM provided by Matlab R2018b
- DCNN: for the implementation of the four models (AlexNet, VGG, ResNet18, and ResNet101), we use the pre-trained models from the Python library PyTorch v1.1 [136]. These models were retrained using transfer learning for the art media problem

also, we outline each of the AA:

- FGSM: was implemented in PyTorch v1.1 [136] using the validation and test datasets to compute AEs with standard values for scale $\epsilon = 2, 4, 8, 16, 32$ for all the DCNN models
- Multiple-pixel attack: was implemented using 100 random images from the test dataset (50 from each class) in Matlab R2018b and Python v3.6.5. Python version was programmed using the

differential evolution from the Pygmo v2.11.3 library [137], and Matlab's version used the differential evolution library available from their file exchange website [138]. Both implementations used the same settings of 50 individuals, 30 generations, a crossover probability of 0.9, and $d = 10,000$ pixels

- Adversarial Patch: was implemented using 100 images from the training dataset for each DCNN model in PyTorch v1.1 [136] with the following parameters set to build the patch: patch size of 50×50 pixels, a max of 100 iterations per image with a stop criteria of 0.9 posterior probability of the target class. As we defined the binary classification problem, we choose the background class as the target prediction to measure the number of class images that predict the model as the target class

And the statistical analysis is explained next:

- Prediction confidence: We used the prediction confidence data from all the models to measure significant differences from each pair of clean and AEs from the FGSM and adversarial patch. The Matlab R2018b's Statistics and Machine Learning Toolbox was used to determine independence, normality, and homoscedasticity from data and use the proper test (ANOVA, Kruskal-Wallis or Welch test) as described in Section 4.2

- Multiple comparisons: We used the accuracy testing data from FGSM and adversarial patch to make multiple comparisons of group means using the Bonferroni test from Matlab R2018b's Statistics and Machine Learning Toolbox

all experiments were run in a computer with Intel Core i7-6700HQ with 24 GB of RAM and graphic card NVIDIA GeForce GTX 1070. Following their respective articles, all the methods were trained to build the classification models. For BP, it was used pre-trained models from [25]. However, in the face recognition problem, we run several evolutionary loops to optimize BP models explained in Section 5.2.4. Adversarial attacks were run independently to test each method using the parameter mentioned above adapted to the art media dataset. The FGSM used $\epsilon = 2, 4, 8, 16, 32$ to control the intensity of the attack over the validation and testing datasets. The multiple-pixel attack used 100 random images from the testing dataset (50 of each class) using $10,000$ pixels. The adversarial patch used the parameters mentioned above to train the patch, and it was put in 100 random images in a random location and orientation. The statistical analysis was programmed to follow the procedure from Chapter 4 to evaluate the predictions' confidence over the testing data and the multi-comparison over the FGSM and adversarial patch data. However, we outline the contribution of this thesis made in our experimental design to measure vulnerabilities beyond deep learning models as mentioned in Chapter 4.

5.1.4 Results

In the following subsections, we present and discuss the results obtained from the experiments introduced earlier.

FGSM

Table 5.2 presents the results for the training and validation datasets from Kaggle along with the AEs computed using FGSM for all DCNN models. We report classification accuracy at each stage of training and validation next to all models' accuracy tested with the AEs. Here, we want to measure the influence in the prediction of the FGSM in two manners: 1) direct, since we know the model's parameters and perturbation, and 2) indirect, through the AE transferability property, which different works report that AEs generated from a model can perform a misclassification attack on a different DCNN trained for the same task with no access to the model information [96, 139]. We consider the transferability property as a type of black-box attack due to the lack of model information. Still, we want to extend the analysis to different architectures such as BP and SIFT+FV that could be affected by these subtle perturbations to the digitized artworks.

First, from Table 5.2 we notice a big gap between the training and validation accuracies from SIFT+FV in comparison to the rest of the models, which obtained comparable results between both datasets. The high variability among results was the reason to suspect possible overfitting in the SIFT+FV models. We employed two overfitting verification procedures presented in Table 5.3. We use the hyperparameters optimizer and the *crossval* function from Matlab. We set in the SVM training process the hyperparameters optimizer with a *maximum objective evaluations*= 10 to return the best model for each class after ten runs. We list the accuracy results from the best model found over the training and validation datasets in the *optimizer* column at Table 5.3. The *crossval* function validates the SVM model using 10-fold cross-validation that randomly partitions the data into ten sets of equal size and trains an SVM classifier using nine sets to finalize after repeating the process ten times. After that, we computed the mean accuracy considering the ten experiments for each class's training and validation datasets. We present the results in the *cross-validation* column at Table 5.3. We obtained in both columns the same results–high variability between training and validation–as the original experiment. In summary, the results showed us that the data do not overfit the models.

Iconography

			AlexNet					VGG					ResNet18					ResNet101				
	train	val	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	92.84	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42	91.42
SIFT+FV	99.92	95.91	95.91	95.91	95.91	95.59	94.26	95.83	95.83	96.07	95.52	94.57	95.75	95.75	96.07	95.52	94.34	95.99	95.99	96.07	95.67	94.73
AlexNet	99.61	98.66	96.3	96.3	83.24	52.56	38.39	98.51	98.51	98.43	98.03	97.64	98.58	98.58	98.66	98.03	97.4	98.51	98.51	98.51	98.03	97.48
VGG	100	99.21	99.29	99.29	99.06	98.82	96.85	91.9	91.9	47.05	17.7	16.76	99.21	99.21	98.98	98.74	95.83	99.21	99.21	98.98	98.35	97.32
ResNet18	100	98.9	98.66	98.66	98.66	98.9	97.95	98.66	98.66	98.66	98.03	95.83	90.24	90.24	52.01	29.03	32.1	98.66	98.66	98.43	97.17	95.75
Resnet101	100	99.37	99.21	99.21	99.21	99.06	97.72	99.29	99.29	99.06	98.9	97.01	99.37	99.37	99.21	97.95	95.28	94.34	94.34	67.98	50.04	51.3

Painting

			AlexNet					VGG					ResNet18					ResNet101				
	train	val	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	99.68	99.04	98.25	98.25	98.48	98.41	98.48	98.78	98.8	98.64	98.33	98.41	98.8	98.8	98.56	98.64	98.56	98.41	98.41	98.56	98.8	97.69
SIFT+FV	99.76	92.24	92.08	92.08	92.00	89.84	87.84	92.16	92.16	92.08	90.48	88.08	91.92	91.92	91.76	90.08	88.00	92.00	92.00	91.84	89.76	87.60
AlexNet	98.96	97.69	93.46	93.46	83.01	66.99	69.3	97.53	97.53	97.13	96.89	96.41	97.45	97.45	96.89	96.97	96.49	97.45	97.45	97.21	97.05	96.73
VGG	99.92	98.17	97.93	97.93	97.53	96.73	92.82	89.31	89.31	32.14	14.27	14.91	97.69	97.69	97.05	95.45	88.28	97.69	97.69	96.81	95.14	88.12
ResNet18	100	97.85	97.93	97.93	97.93	97.45	96.33	97.77	97.77	97.05	96.33	93.22	86.92	86.92	43.94	31.82	40.75	97.69	97.69	97.13	95.77	92.9
Resnet101	100	98.56	98.72	98.72	98.48	98.17	96.65	98.64	98.64	98.25	96.49	93.86	98.72	98.72	98.17	95.85	92.58	91.15	91.15	55.42	43.94	49.68

Drawings

			AlexNet					VGG					ResNet18					ResNet101				
	train	val	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	96.56	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59	90.59
SIFT+FV	99.87	83.84	83.84	83.84	83.97	83.46	81.30	84.22	84.22	84.48	83.59	81.93	84.22	84.22	84.22	82.95	81.68	84.10	84.10	84.35	82.95	80.79
AlexNet	96.44	91.35	85.75	85.75	66.79	44.91	35.62	90.84	90.84	91.22	90.59	88.55	91.09	91.09	91.09	90.59	89.06	90.71	90.71	91.09	91.09	90.08
VGG	99.75	95.42	95.29	95.29	94.78	93.51	87.02	74.43	74.43	28.75	15.78	14.38	94.78	94.78	93.13	88.68	77.86	94.78	94.78	93.77	90.59	83.46
ResNet18	99.87	94.44	94.27	94.27	93.64	92.37	86.9	93.38	93.38	91.22	86.77	77.48	72.9	72.9	31.04	23.41	22.77	93.64	93.64	92.37	88.17	80.28
Resnet101	99.87	95.8	95.8	95.8	95.42	93.89	89.31	95.55	95.55	93.89	90.84	83.33	95.29	95.29	93.13	88.68	80.79	76.08	76.08	47.96	41.48	38.55

Sculpture

			AlexNet					VGG					ResNet18					ResNet101				
	train	val	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	93.19	93.26	92.79	92.79	92.79	92.79	92.79	92.79	92.79	92.79	92.7	92.79	92.88	92.88	92.79	92.79	92.7	92.88	92.88	92.79	92.88	92.7
SIFT+FV	99.55	87.35	87.44	87.44	85.79	85.15	83.68	87.26	87.26	86.34	85.15	84.42	87.35	87.35	85.98	84.97	85.06	87.44	87.44	85.98	85.24	85.15
AlexNet	99.36	95.78	90.93	90.93	63.24	27.50	14.57	95.78	95.78	95.42	94.68	89.55	95.88	95.88	95.78	94.13	89.09	95.97	95.97	96.06	94.68	90.10
VGG	100	97.62	98.26	98.26	97.89	94.87	78.28	84.69	84.69	37.76	17.87	14.21	98.08	98.08	97.07	91.38	72.59	97.98	97.98	96.98	93.31	78.00
ResNet18	100	96.88	97.25	97.25	96.88	95.05	80.66	96.88	96.88	96.15	92.39	77.54	84.88	84.88	45.92	25.30	19.07	96.70	96.70	95.69	92.58	79.65
Resnet101	100	97.89	98.44	98.44	98.17	96.06	87.08	98.44	98.44	98.08	95.42	84.88	98.35	98.35	96.98	92.30	77.45	89.00	89.00	60.49	44.18	37.86

Engraving BW

			AlexNet					VGG					ResNet18					ResNet101				
	train	val	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	89.76	92.05	92.23	92.23	92.23	91.70	91.87	91.70	91.70	92.06	91.87	91.70	91.70	91.70	92.23	92.05	91.53	91.70	91.70	91.87	91.87	92.05
SIFT+FV	100	93.99	94.35	94.35	94.70	94.17	92.76	94.35	94.35	94.35	94.17	93.64	94.35	94.35	94.52	94.17	93.46	94.35	94.35	94.88	94.35	93.46
AlexNet	99.76	99.29	96.11	96.11	78.62	56.71	47.88	99.12	99.12	99.12	98.94	98.41	99.12	99.12	99.12	98.94	98.06	99.12	99.12	99.12	98.94	98.41
VGG	100	100	99.82	99.82	99.82	99.65	99.29	98.53	97.53	73.14	49.29	47.17	99.82	99.82	99.82	99.82	99.12	99.82	99.82	99.82	99.82	99.29
ResNet18	100	100	100	100	99.82	99.82	98.94	99.82	99.82	99.82	99.65	98.23	95.58	95.58	78.98	64.49	63.07	100	100	100	100	98.41
Resnet101	100	100	100	100	100	99.82	99.47	100	100	99.82	99.82	99.47	100	100	99.65	99.65	98.76	98.94	98.94	94.70	89.75	88.16

Engraving Color

			AlexNet					VGG					ResNet18					ResNet101				
	train	val	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	98.33	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37	97.37
SIFT+FV	100	50.00	44.74	44.74	44.74	44.74	50.00	47.37	47.37	47.37	47.37	50.00	47.37	47.37	44.74	47.37	47.37	50.00	50.00	47.37	47.37	50.00
AlexNet	100	100	73.68	73.68	23.68	13.16	15.79	100	100	100	100	94.74	100	100	100	100	94.74	100	100	100	94.74	92.11
VGG	100	100	100	100	100	100	97.37	97.37	97.37	26.32	15.79	13.16	100	100	100	100	100	100	100	100	100	100
ResNet18	95.00	100	97.37	97.37	97.37	97.37	81.58	97.37	97.37	97.37	89.47	81.58	52.63	52.63	13.16	02.63	21.05	97.37	97.37	97.37	94.74	78.95
Resnet101	100	100	100	100	100	100	97.37	100	100	100	100	97.37	100	100	100	97.37	94.74	94.74	94.74	81.58	65.79	68.42

Table 5.2: Results using training and validation datasets from Kaggle. Each method presents its classification accuracy for training, validation, and the AEs using the FGSM computed at $\epsilon = \{2, 4, 8, 16, 32\}$. The AEs generated by the four DCNN models (AlexNet, VGG, ResNet18, and ResNet101) are in their respective columns.

	Optimizer		Cross-validation	
SIFT+FV	train	val	mean train	mean val
Iconography	100	95.28	99.28	95.28
Painting	99.76	92.72	98.84	92.83
Drawings	100	83.84	98.28	83.44
Sculpture	100	86.71	98.63	86.48
Engraving Bw	100	93.64	99.32	93.87
Engraving Color	100	50.00	92.00	47.11

Table 5.3: This table shows the results of using the SVM hyperparameters optimizer method from Matlab and the cross-validation function to verify overfitting on SIFT+FV.

In Table 5.2 as well as in Figure 5.1, we observed how drastically can be dropped the performance of DCNNs in the presence of a direct treat using the FGSM. The worst-case was in sculpture class, where VGG's performance went from 97.62% to 14.38%, AlexNet dropped from 95.78% to 14.57%, ResNet18 diminished from 96.88% to 19.07%, and ResNet101 decreased from 97.89% to 37.86%. We detected the transferability property between DCNN models in this experiment, which is more significant at $\epsilon = 32$. This effect can be perceived in the most straightforward artwork classification experiments. Therefore, we can assume that it can be present in more extensive experiments with thousands of classes in a more effective manner converting the models more vulnerable to this effect as reported in [12, 96, 139]. The drawings class presents almost the same behavior as the sculpture class, where other networks are affected by AEs. For all other classes, the effect is unnoticeable, but the accuracy is significantly affected when the model matches the AE.

In some cases, SIFT+FV was affected by AEs from FGSM. For example, in the drawing class, the performance was reduced by almost 8%. Furthermore, for the painting, the accuracy was decreased approximately by 4%. This result demonstrated a partial AE transferability to SIFT+FV because regardless of the applied DCNN, the perturbation compromised these two classes' performance. However, BP maintains its performance in almost every test; the accuracy variation through all the analyses was lower than 2%.

Figure 5.1 presents the results of Table 5.2 using the accuracy ratios between adversarial examples and clean images. We observe that the variation of BP is imperceptible in comparison with SIFT+FV and DCNN models. Also, we noted that the performance of DCNNs drastically dropped in almost all classes reaching less than 20% of its original accuracy when the perturbation matches the network design. In all other cases, the attack reduces the accuracy to about 20% of the actual performance considering clean images for Sculpture, Engraving BW, and Engraving Color. Figure 5.4 illustrates an example showing that the generated maps from the AVC do not suffer any change in their responses with the FGSM.

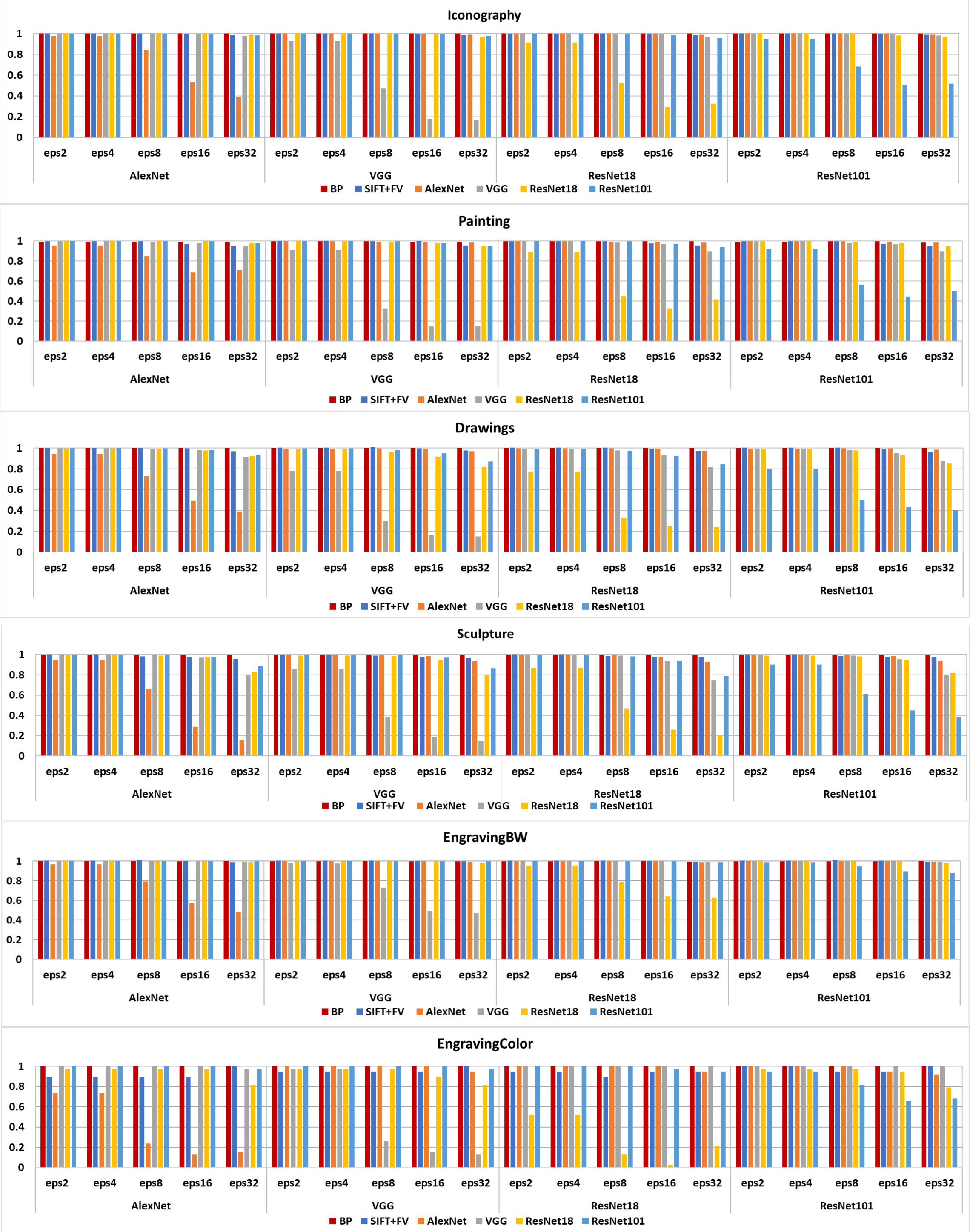


Figure 5.1: Comparative graph of the computed accuracy ratios between adversarial examples and clean images from each method using the validation dataset from Kaggle.

Iconography

		AlexNet					VGG					ResNet18					ResNet101				
	test	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	91.74	91.66	91.66	91.82	91.74	91.74	91.66	91.66	91.74	91.74	91.74	91.66	91.66	91.66	91.58	91.5	91.58	91.58	91.58	91.58	91.58
SIFT+FV	86.16	85.54	85.54	84.71	83.26	77.69	85.54	85.54	84.92	83.47	77.48	85.95	85.95	84.71	83.06	76.24	86.16	86.16	84.71	83.06	75.62
AlexNet	96.07	93.39	93.39	70.04	37.4	28.72	95.87	95.87	95.04	94.42	92.98	96.07	96.07	95.87	94.83	93.18	96.07	96.07	95.45	94.63	92.15
VGG	95.87	95.45	95.45	94.83	91.32	80.99	76.65	76.65	36.98	23.97	21.69	95.66	95.66	94.21	87.81	76.86	95.87	95.87	95.87	90.91	82.44
ResNet18	96.49	95.87	95.87	94.83	94.21	87.81	95.66	95.66	94.42	90.5	83.88	76.86	76.86	38.64	25.21	21.49	96.07	96.07	94.21	90.29	85.12
Resnet101	95.25	95.25	95.25	94.83	92.77	89.88	95.45	95.45	94.63	91.94	88.02	95.45	95.45	92.56	87.6	83.26	79.96	79.96	49.38	36.16	36.36

Painting

		AlexNet					VGG					ResNet18					ResNet101				
	test	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	100	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65	95.65
SIFT+FV	94.83	94.83	94.83	94.70	94.57	93.63	94.92	94.92	94.88	94.57	93.63	94.88	94.88	94.79	94.44	93.28	94.88	94.88	94.70	94.32	93.20
AlexNet	94.06	90.57	90.57	64.64	41.04	41.00	94.10	94.10	93.90	94.01	94.92	94.10	94.10	94.06	94.32	95.35	94.10	94.10	94.06	94.06	95.00
VGG	93.37	93.28	93.28	92.64	87.47	60.12	61.15	61.15	13.14	10.42	10.68	92.89	92.89	91.17	80.10	47.55	92.59	92.59	90.78	81.05	44.96
ResNet18	94.23	94.19	94.19	94.40	94.40	92.64	94.01	94.01	93.63	91.30	81.91	64.86	64.86	15.25	13.01	15.07	93.80	93.80	92.72	89.84	80.19
Resnet101	95.91	95.82	95.82	95.78	94.62	90.09	95.82	95.82	95.69	90.44	79.03	95.61	95.61	94.66	88.33	73.47	75.24	75.24	30.62	19.04	19.98

Painting Landscapes

		AlexNet					VGG					ResNet18					ResNet101				
	test	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
SIFT+FV	75.34	75.07	75.07	72.90	70.46	62.6	75.61	75.61	73.71	70.46	62.60	75.34	75.34	73.17	68.83	60.70	75.34	75.34	72.90	68.29	59.62
AlexNet	93.77	86.99	86.99	61.25	41.46	35.77	93.50	93.50	92.68	91.06	90.24	93.50	93.50	92.68	91.60	90.51	93.77	93.77	92.68	91.33	90.51
VGG	94.58	94.58	94.58	94.31	90.79	73.71	80.76	80.76	42.01	28.73	30.89	94.31	94.31	94.31	88.08	72.63	94.04	94.04	93.22	87.53	70.19
ResNet18	95.12	94.85	94.85	93.77	92.95	90.79	94.31	94.31	92.41	89.43	81.84	72.36	72.36	42.82	33.60	38.48	94.31	94.31	91.6	88.62	79.95
Resnet101	95.39	95.12	95.12	95.12	93.77	89.43	94.58	94.58	94.58	93.50	83.74	94.31	94.31	92.95	88.89	78.86	80.76	80.76	50.96	43.90	44.72

Drawings

		AlexNet					VGG					ResNet18					ResNet101				
	test	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	94.05	94.28	94.28	93.59	93.81	94.5	93.82	93.82	94.05	93.59	94.73	93.81	93.81	93.59	93.59	94.05	93.81	93.81	93.59	93.59	93.81
SIFT+FV	73.61	68.42	68.42	66.36	62.7	56.52	68.42	68.42	67.51	62.7	57.89	68.42	68.42	66.82	62.01	56.75	68.19	68.19	67.28	62.01	56.75
AlexNet	86.73	77.8	77.8	57.21	41.19	32.72	85.81	85.81	86.27	84.67	79.18	85.81	85.81	85.35	83.75	80.09	85.81	85.81	85.58	84.67	81.69
VGG	91.99	91.99	91.99	90.89	88.79	80.78	72.77	72.77	35.7	20.59	18.99	91.3	91.3	89.02	83.3	73.91	91.53	91.53	89.7	83.98	76.89
ResNet18	90.85	90.85	90.85	89.7	88.79	81.46	90.39	90.39	87.41	81.69	74.37	71.85	71.85	36.16	24.49	24.49	90.16	90.16	86.96	81.92	74.83
Resnet101	93.59	93.36	93.36	93.14	91.53	85.81	93.14	93.14	90.62	85.58	76.43	93.14	93.14	90.16	83.07	75.29	72.27	72.27	45.54	35.24	33.41

Sculpture

		AlexNet					VGG					ResNet18					ResNet101				
	test	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	90.54	90.83	90.83	90.83	90.83	90.83	85.96	85.96	85.96	85.96	85.96	90.83	90.83	90.83	90.83	90.83	90.83	90.83	90.83	90.83	90.83
SIFT+FV	60.47	52.80	52.80	52.80	53.10	51.62	53.39	53.39	53.10	52.51	52.51	52.80	52.80	52.51	52.51	51.33	53.39	53.39	52.51	52.51	50.74
AlexNet	91.45	87.61	87.61	65.49	44.25	36.87	91.15	91.15	90.56	89.38	87.32	91.45	91.45	91.45	89.09	89.38	91.45	91.45	91.45	90.27	88.20
VGG	94.69	94.99	94.99	94.99	92.33	84.37	79.06	79.06	45.43	32.74	34.51	95.28	95.28	94.10	88.20	82.01	94.69	94.69	93.81	91.74	86.73
ResNet18	92.63	91.74	91.74	90.86	89.38	83.19	91.74	91.74	87.91	84.96	80.24	75.81	75.81	46.61	34.81	33.92	91.15	91.15	89.38	86.14	83.19
Resnet101	92.92	93.22	93.22	92.63	90.86	87.61	92.92	92.92	92.92	89.97	86.14	93.22	93.22	91.15	88.20	83.48	80.53	80.53	56.64	50.44	56.34

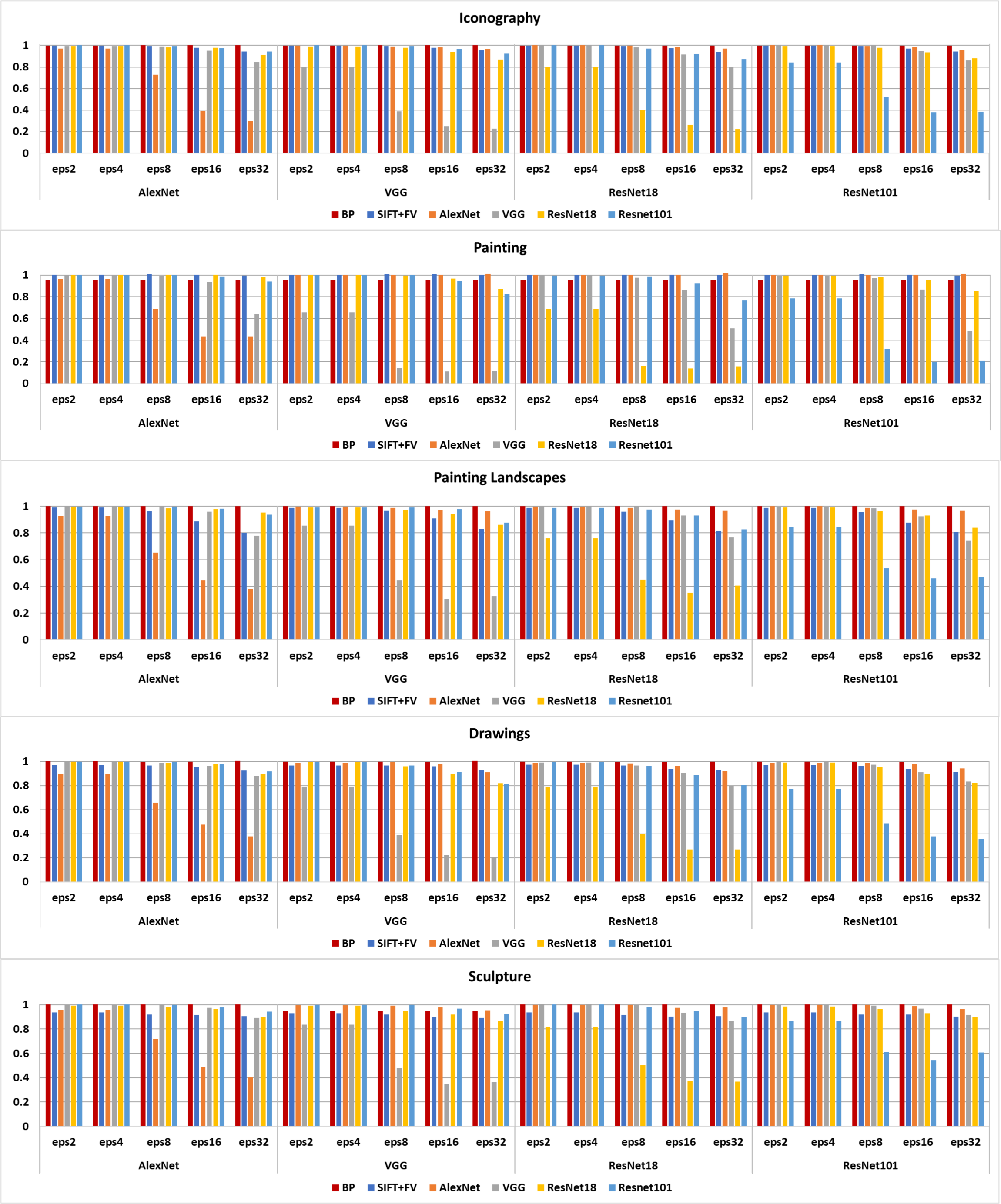
Engraving BW

		AlexNet					VGG					ResNet18					ResNet101				
	test	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	91.55	92.64	92.64	91.97	91.72	91.63	92.30	92.30	92.05	92.05	91.63	91.97	91.97	91.80	91.97	91.80	92.13	92.13	91.97	91.80	91.63
SIFT+FV	89.79	89.79	89.79	89.71	89.87	90.96	89.79	89.79	89.79	89.37	90.96	89.87	89.87	89.62	89.54	90.88	89.96	89.96	89.46	89.54	90.96
AlexNet	98.58	94.06	94.06	75.06	57.32	54.64	98.66	98.66	98.66	98.49	97.32	98.66	98.66	98.66	98.33	97.15	98.66	98.66	98.66	98.58	97.49
VGG	99.58	99.83	99.83	99.67	99.50	99.16	91.05	91.05	62.85	45.94	49.87	99.58	99.58	98.74	98.41	97.91	99.58	99.58	99.25	99.00	98.83
ResNet18	99.83	99.92	99.92	99.83	99.67	99.16	99.75	99.75	99.41	98.83	97.49	93.22	93.22	71.55	59.41	61.09	99.83	99.83	99.67	98.91	97.82
Resnet101	99.67	99.75	99.75	99.75	99.83	99.75	99.83	99.83	99.50	99.25	98.74	99.67	99.67	99.50	99.08	98.16	95.90	95.90	90.13	85.77	83.01

Engraving Color

		AlexNet					VGG					ResNet18					ResNet101				
	test	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	89.92	89.68	89.68	89.74	89.86	89.80	89.92	89.92	89.74	89.86	89.62	89.68	89.68	89.74	89.98	89.80	89.92	89.92	89.86	89.50	90.16
SIFT+FV	66.95	66.77	66.77	66.59	66.89	68.09	66.83	66.83	66.59	67.19	68.09	66.89	66.65	66.95	68.33	66.71	66.71	66.53	66.53	66.95	66.95
AlexNet	94.72	73.55	73.55	25.49	12.30	17.22	94.78	94.78	94.90	94.48	93.64	94.72	94.72	94.66	95.14	94.24	94.54	94.54	95.02	94.66	94.00
VGG	99.40	99.46	99.46	99.46	99.28	96.52	79.90	79.90	16.02	05.46	06.06	99.52	99.52	99.22	99.10	97.18	99.40	99.40	99.10	98.50	95.98
ResNet18	96.40	95.98	95.98	96.16	95.02	89.14	95.92	95.92	95.50	93.88	89.50	49.13	49.13	06.84	05.58	10.74	95.62	95.62	95.02	92.68	86.98
Resnet101	99.88	99.76	99.76	99.76	99.52	98.92	99.70	99.70	99.70	99.22	98.44	99.82	99.82	99.76	99.40	98.56	92.86	92.86	61.91	49.19	54.53

Table 5.4: Results using testing datasets from Wikiart. Each method presents its classification accuracy for training, validation, and the AEs using the FGSM computed at $\epsilon = \{2, 4, 8, 16, 32\}$. The AEs generated by the four DCNN models (AlexNet, VGG, ResNet18, and ResNet101) are in their respective columns.


Iconography

Painting

Painting Landscapes

Drawings

Sculpture

Figure 5.2: Comparative graphs of the computed accuracy ratios between adversarial examples and clean images from each method using Iconography, Painting, Painting Landscapes, Drawings, and Sculpture classes from the testing dataset from Wikiart.

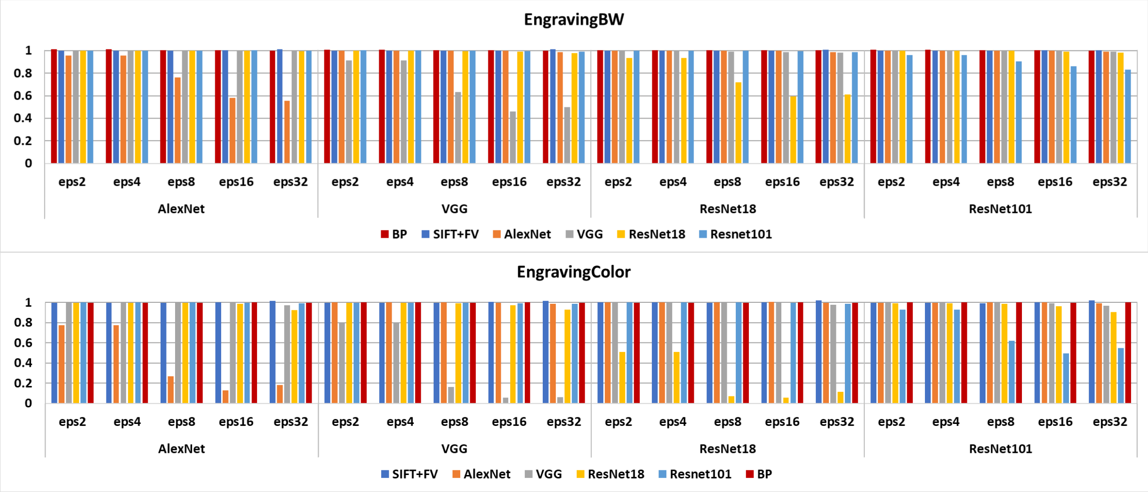


Figure 5.3: This figure shows comparative graphs of the computed accuracy ratios between adversarial examples and clean images from each method using Engraving BW and Engraving Color classes from the testing dataset from Wikiart.

The testing stage exhibited a worse scenario compared with the validation dataset for the DCNN and SIFT+FV. Table 5.4 shows that the accuracy was compromised in all DCNN models for three classes: Painting, Drawings, and Engraving Color. For example, the worst-case is Engraving Color (see Figure 5.3), where AlexNet fell to 17.22% from a clean score of 94.72%, VGG and ResNet18 diminished their performance by 5% of accuracy after scoring 99% and 96%, respectively, and ResNet101 achieves 49%, which was the less affected in the accuracy. Moreover, the experimental results in Table 5.4 demonstrated a drop in the performance due to the transferability property. Despite, it is not a big concern the drop in accuracy of approximately 10-20% when we transfer AEs at $\epsilon = 32$ contrary to the approximate drop of 80% when the model matches the attack; attackers have almost up to 20% possibility to succeed. Also, we observed an asymmetric transferability of AEs, which means that AEs generated on a DCNN model tend to decrease the performance of a different model highly, but it does not occur equally in the other direction. Therefore, we have two concerns with these results: 1) the possibility to use the transferability properties as a black box attack and asymmetric transferability to find the best DCNN model to manipulate a targeted model; 2) the reachability of this effect to manipulate the results in the most straightforward artwork classification experiments pointing out to a situation when the number of classes accentuates this vulnerability. Also, the test showed the poor performance of SIFT+FV considering clean images. In four out of seven classes (Painting Landscapes, Drawings, Sculpture, and Engraving color), the accuracy is way below to compete with DCNNs. Additionally, SIFT+FV was affected by AEs in Iconography, Painting Landscapes, and Sculpture, where approximately 10% of its original score reduced the performance. Finally, BP demonstrated high quality and steady results keeping its scores from clean images after AEs with minimal to zero changes for all classes.Additionally, it is noticeable that, as opposed to SIFT+FV, BP reaches comparable results to DCNNs' scores. Besides, we present in Figures 5.2-5.3 the accuracy ratio on AEs for the testing classes. We observed graphically the same behavior, at least for BP, whose rate for all experiments remains almost one. In contrast, we see a drastic drop in DCNN models' performance when the perturbation matches the network's architecture and influences AEs' transferability to other DCNN models and SIFT+FV.

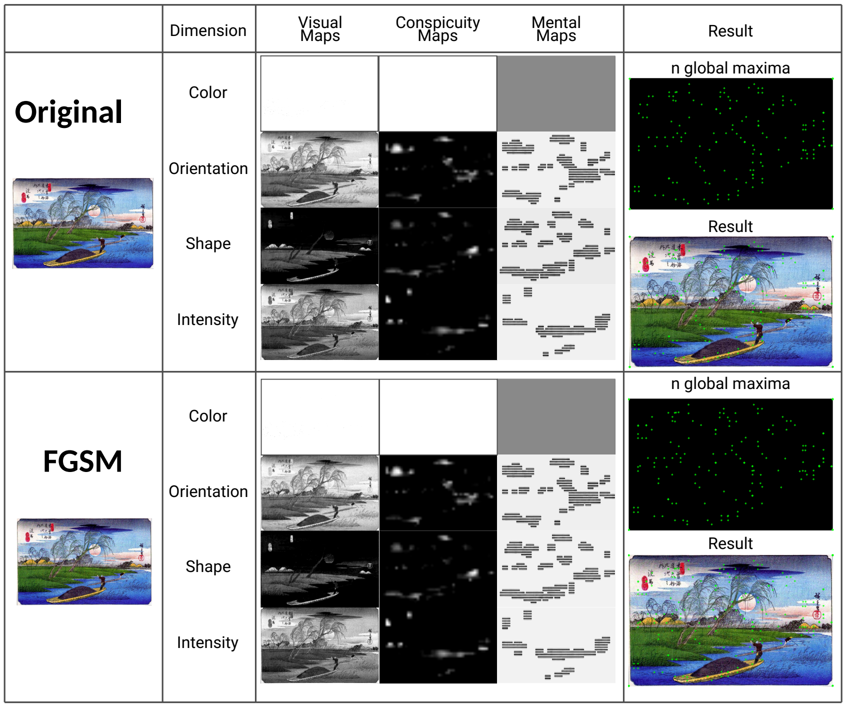


Figure 5.4: Maps generated in each phase of the AVC extracted from the original image and the AE computed with FGSM using ResNet101 and $\epsilon = 32$. The last column illustrates the final stage of n global maxima and their superposition with the original image. Note that despite the attack, the point locations do not change much in the generated map.

Multiple-pixel Attack

The one-pixel attack experiment exhibit that one pixel does not perturb high-resolution images to change the model's prediction. We experiment with modifying one pixel to fool the models over 100 selected images, and the results indicate no score changes. Therefore, we experimentally found that when we set the attack with 8000-10,000 pixels, DCNN models have a massive change in their prediction. Thus, we set a multiple-pixel attack experiment with 10,000-pixels. In Table 5.5, we present the pixel change ratios from the multiple-pixel attack experiment. Since art images present a high size variation, we denote the ratios from each class's smallest and largest images to indicate pixel rate change in the experiment.

Class	Iconography	Painting	Painting Land.	Drawings	Sculpture	Engraving BW	Engraving Color
Min size (pixels)	2276×1804	400×335	789×800	768×595	549×500	768×629	875×600
Ratio	0.24%	7.46%	1.58%	2.18%	3.64%	2.07%	1.90%
Max size (pixels)	2697×4386	3039×2400	1100×1377	1424×1348	2484×1564	3000×1934	3518×2348
Ratio	0.08%	0.13%	0.66%	0.52%	0.25%	0.17%	0.12%

Table 5.5: Pixel ratios from the multiple-pixel attack.

Table 5.6 presents the results from the multiple-pixel attack experiment. Under the success rate row, we observed the number of images that changed their forecast. The confidence row shows the mean posterior probability over the new predictions, which is the parameter that all classifiers give when a label is predicted to measure the confidence in the forecast. Also, we report the mean processing time in seconds. We observed that DCNN changes by a considerable amount of their predictions with high confidence by modifying multiple-pixels. SIFT+FV was also misled in five out of seven classes achieving the same number of images as DCNN models with lower confidence. In this way, only two categories resisted the attack. On the contrary, BP was robust to this attack having four out of seven classes without changes and the rest with a maximum error of 4%. Notice that the amount of pixels modified in this experiment fails the motivation of AA in which the perturbation should be unnoticeable to human vision and its processing time makes this attack unfeasible to perform in real-time applications like video streams.

It is remarkable to notice that this attack attempts to threaten image classification methods regardless of their architecture directly. The differential evolution algorithm [97] employed in this attack proposes one of the most powerful stochastic optimization strategies for solving complex multi-modal problems. In this manner, we want to highlight that even this method had evolved populations to find 10,000 pixels' location and their RGB values, but it could not find in their entire search space (image size and RGB values) perturbations that represented a real threat to BP. Contrarily to DCNN and SIFT+FV, which were affected in a significant manner. We illustrate in Figure 5.5 an example of BP-generated maps using a multiple-pixel attack to compare it against the original response in Figure 5.4. We can observe that the response of the n global maxima, where the image descriptor is built, does not change with the perturbed image.

Iconography	**BP**	**SIFT+FV**	**AlexNet**	**VGG**	**ResNet18**	**ResNet101**
Original Acc.	92.00	88.00	96.00	94.00	96.00	92.00
Success Rate	0.00	32.00	32.00	44.00	46.00	42.00
Confidence	NA	64.96	85.09	85.72	76.34	77.61
Time (seconds)	94.22	301.21	138.51	147.72	152.37	237.73

Painting	**BP**	**SIFT+FV**	**AlexNet**	**VGG**	**ResNet18**	**ResNet101**
Original Acc.	100	78.00	94.00	90.00	92.00	94.00
Success Rate	2.00	0.00	54.00	60.00	64.00	64.00
Confidence	51.83	NA	78.11	97.34	99.37	98.06
Time (seconds)	90.16	598.12	119.78	122.59	111.14	242.58

Painting Landscapes	**BP**	**SIFT+FV**	**AlexNet**	**VGG**	**ResNet18**	**ResNet101**
Original Acc.	100	78.00	88.00	88.00	92.00	92.00
Success Rate	2.00	40.00	54.00	60.00	64.00	66.00
Confidence	54.06	62.04	75.70	97.25	99.26	97.37
Time (seconds)	98.83	585.69	141.85	163.51	143.62	205.53

Drawings	**BP**	**SIFT+FV**	**AlexNet**	**VGG**	**ResNet18**	**ResNet101**
Original Acc.	88.00	70.00	80.00	90.00	86.00	92.00
Success Rate	0.00	38.00	68.00	68.00	74.00	78.00
Confidence	NA	66.53	83.91	91.94	95.11	94.24
Time (seconds)	118.85	462.92	110.18	111.48	128.07	220.69

Sculpture	**BP**	**SIFT+FV**	**AlexNet**	**VGG**	**ResNet18**	**ResNet101**
Original Acc.	86.00	62.00	88.00	98.00	96.00	96.00
Success Rate	4.00	60.00	62.00	54.00	56.00	54.00
Confidence	58.14	67.61	92.65	98.60	97.45	96.93
Time (seconds)	71.20	601.53	121.22	130.06	137.16	181.14

Engraving BW	**BP**	**SIFT+FV**	**AlexNet**	**VGG**	**ResNet18**	**ResNet101**
Original Acc.	94.00	94.00	100	100	100	100
Success Rate	0.00	0.00	40.00	50.00	32.00	20.00
Confidence	NA	NA	77.63	68.07	71.86	61.25
Time (seconds)	88.71	599.41	148.90	169.56	152.11	177.61

Engraving Color	**BP**	**SIFT+FV**	**AlexNet**	**VGG**	**ResNet18**	**ResNet101**
Original Acc.	94.00	74.00	98.00	100	92.00	100
Success Rate	0.00	60.00	40.00	50.00	46.00	22.00
Confidence	NA	55.98	73.80	66.15	62.96	65.31
Time (seconds)	87.01	600.82	150.70	174.51	154.52	186.13

Table 5.6: This table shows the results from the experiment of computing the multiple-pixel attack with $d = 10,000$ on 100 random images from the testing dataset. The original accuracy refers to the score of the clean images. Success rate means the percentage of images that change the prediction with a mean confidence value of the posterior probabilities over the new predicted classes and the mean processing time in seconds.

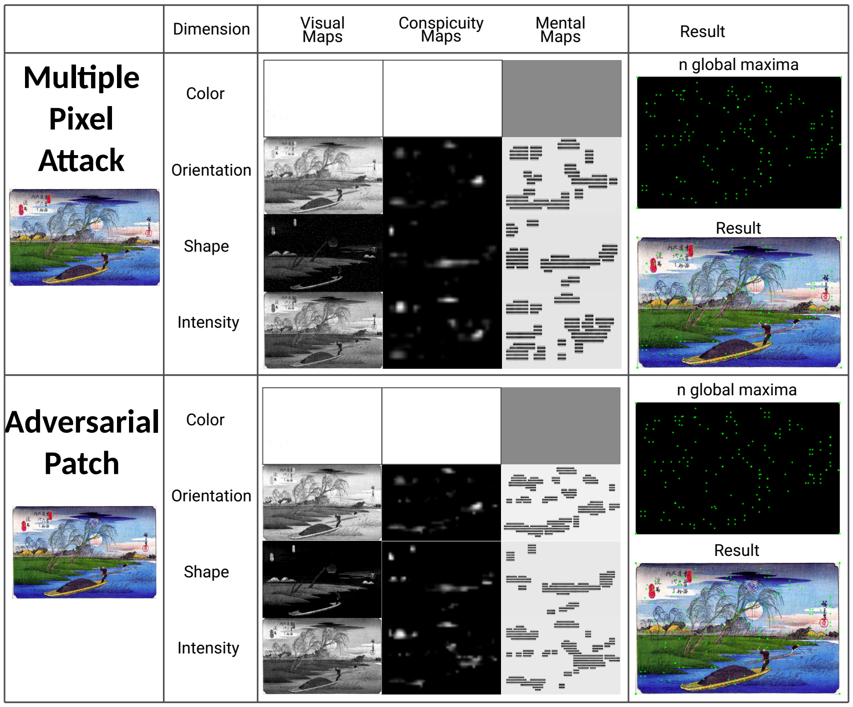


Figure 5.5: Maps generated in each phase of the AVC extracted from an AE of the multiple-pixel attack and the image with the adversarial patch. The last column illustrates the final stage of n global maxima and their superposition with the original image. Note that despite the attack, the point locations do not change much in the generated map.

Adversarial Patch

We present the accuracy of the adversarial patch experiment in Table 5.7 and the accuracy ratios in a graphical manner in Figure 5.6. This experiment analyzes the change in the model's predictions by adding the trained patches from DCNN models using 100 images from each class in a random location and orientation. The results from Table 5.7 show that these patches affect in a significant manner DCNN models in most experiments. Also, we discovered that the patches could be transferable to other DCNNs.

The painting landscapes experiment showed the worst-case scenario for DCNN models, on which we observed a considerable AE transferability between the models. We observed that VGG, ResNet18, and ResNet101 were affected by all the patches. DCNN models dropped their performance to approximately half of their original accuracy and, in some cases, is less to 50%. ResNet18 was fooled in all images using its trained patch. All other classes did not show a similar behavior; the patches can fool DCNN models. In contrast, SIFT+FV and BP demonstrated a robust control over the adversarial patches, showing almost an unchangeable performance. Figure 5.5 illustrates the BP-generated maps using an image with the adversarial patch.

Iconography	Original Acc.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	99.00	99.00	99.00	99.00	99.00
SIFT+FV	92.00	89.00	93.00	93.00	92.00
AlexNet	98.00	74.00	97.00	97.00	98.00
VGG	94.00	91.00	45.00	82.00	81.00
ResNet18	94.00	87.00	90.00	58.00	90.00
ResNet101	93.00	87.00	87.00	78.00	70.00

Painting	Original Acc.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	100.00	100.00	100.00	99.00	100.00
SIFT+FV	97.00	98.00	97.00	98.00	96.00
AlexNet	96.00	54.00	94.00	94.00	94.00
VGG	92.00	71.00	48.00	73.00	61.00
ResNet18	94.00	67.00	76.00	23.00	43.00
ResNet101	97.00	72.00	72.00	69.00	56.00

Painting Land.	Original Acc.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	100.00	100.00	100.00	100.00	100.00
SIFT+FV	87.00	81.00	78.00	84.00	81.00
AlexNet	94.00	24.00	85.00	86.00	77.00
VGG	95.00	41.00	19.00	48.00	23.00
ResNet18	95.00	22.00	39.00	0.00	9.00
ResNet101	96.00	43.00	41.00	35.00	22.00

Drawings	Original Acc.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	91.00	91.00	91.00	91.00	91.00
SIFT+FV	72.00	67.00	68.00	69.00	67.00
AlexNet	94.00	30.00	85.00	80.00	73.00
VGG	98.00	81.00	69.00	74.00	62.00
ResNet18	96.00	82.00	91.00	66.00	79.00
ResNet101	99.00	88.00	90.00	85.00	75.00

Sculpture	Original Acc.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	85.00	85.00	85.00	85.00	85.00
SIFT+FV	95.00	92.00	94.00	94.00	95.00
AlexNet	97.00	32.00	92.00	89.00	86.00
VGG	97.00	93.00	72.00	85.00	85.00
ResNet18	95.00	92.00	86.00	66.00	89.00
ResNet101	94.00	87.00	89.00	86.00	87.00

Engraving BW	Original Acc.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	90.00	90.00	91.00	91.00	91.00
SIFT+FV	91.00	94.00	93.00	95.00	92.00
AlexNet	100.00	99.00	100.00	100.00	100.00
VGG	100.00	99.00	83.00	96.00	97.00
ResNet18	100.00	100.00	96.00	71.00	96.00
ResNet101	100.00	100.00	100.00	100.00	100.00

Engraving Color	Original Acc.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	93.00	92.00	93.00	92.00	92.00
SIFT+FV	94.00	94.00	96.00	95.00	95.00
AlexNet	97.00	67.00	98.00	95.00	93.00
VGG	100.00	100.00	99.00	100.00	100.00
ResNet18	98.00	99.00	100.00	98.00	99.00
ResNet101	100.00	100.00	100.00	100.00	100.00

Table 5.7: This table shows the results from the adversarial patch experiment. Each column presents the score obtained for the original 100 images per class (Wikiart) when adding the adversarial patch.

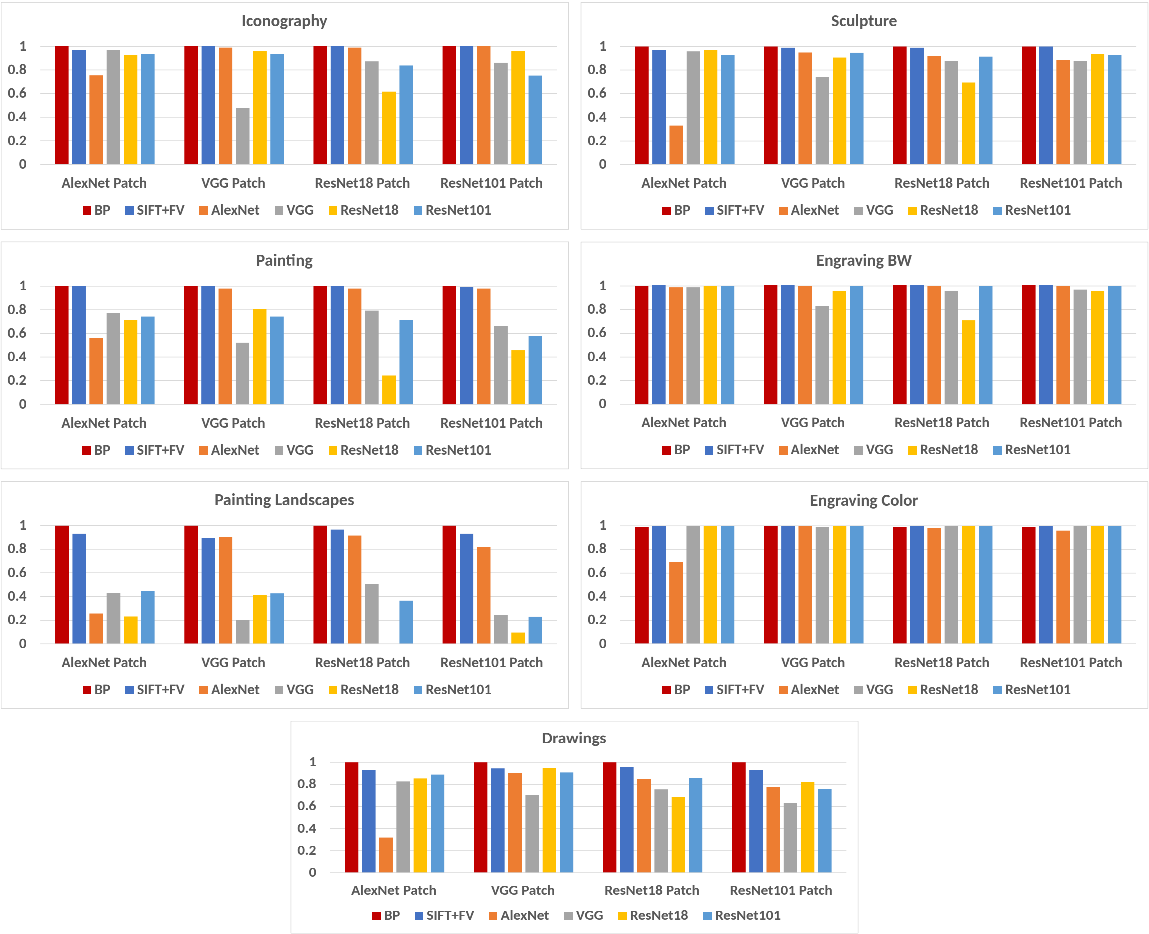


Figure 5.6: Comparative graph of the computed accuracy ratios between adversarial examples from the patches and clean images from each method using 100 images per class from the testing dataset.

Statistical Analysis of Robustness

The statistical analysis from Tables 5.8-5.9 shows that the predictions' confidence from BP is not significantly different in every experiment of the test dataset using FGSM. That means that the confidence is not affected by the subtle perturbations added to the images. The majority of p-values from SIFT+FV demonstrate to be not significantly different between the predictions' confidences. Nonetheless, the analysis from all DCNN architectures showed that, in most cases, the rejection of the null hypothesis Ho. The rejection illustrates the damage of the AEs to the DCNN's predictions' confidence by making them statistically different.

We observe the same behavior in the statistical analysis at Table 5.10, which shows the method's predictions' confidence to the adversarial patch. The study showed the same rejection of the null hypothesis Ho from all DCNN architectures in a significant part of the experiments for all classes. Conversely, BP accepted the null hypothesis Ho in every experiment. SIFT+FV showed similar behavior to BP, but the sculpture class and the VGG patch obtained significantly different predictions' confidence.

A simple main effect analysis between all classifiers using the Bonferroni method as a post-hoc test from the previous procedure considerations was employed with the FGSM testing data. First, we obtained a rejection of the null hypothesis that all group means are equal with a p-value$= 4.2063e-15$. Then, we use multiple comparisons to determine which group means are different from others using the Bonferroni method. Figure 5.7 shows the multiple comparisons of group means on which BP obtained significant differences from the rest of the classifiers.

Also, we applied this statistical analysis to the adversarial patch data to compare the performances from all classifiers. We obtained a p-value$= 0.0013$, establishing a rejection of the null hypothesis and demonstrating the differences between all classifiers. Figure 5.8 presents the multiple comparisons of group means; in this case, BP shows significant differences to VGG and ResNet18. This result can be explained because the authors from this attack claimed to be robust to the location and orientation of the patch [98]; however, it is still dependant on these variables. In this experiment, we observed two scenarios where two DCNN models were drastically affected and two in a limited manner. Despite BP and SIFT+FV exhibited higher group means, BP highlight among all groups.

Iconography

	AlexNet					VGG				
Testing Vs.	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$
BP	0.99743	0.99889	0.99599	0.9958	0.9958	0.99828	0.99634	0.99517	0.9937	0.99371
SIFT+FV	0.95119	0.95118	0.80859	0.21374	0.00012958	0.93022	0.93022	0.73482	0.10472	9.0663e-06
AlexNet	1.7665e-12	1.7665e-12	1.8922e-54	5.2622e-73	1.5556e-74	0.99483	0.99483	0.99702	0.84401	0.83964
VGG	0.58235	0.58245	0.0087004	1.884e-09	3.1824e-32	3.2201e-30	3.2201e-30	3.1832e-64	1.3611e-174	7.6365e-176
ResNet18	0.63062	0.63063	0.024056	8.0612e-06	3.3087e-17	0.52182	0.52182	0.00035193	3.2009e-10	2.4186e-22
ResNet101	0.6259	0.62599	0.054054	5.9926e-05	2.5752e-11	0.54501	0.54501	0.0017698	1.0084e-06	2.232e-12

Painting

	AlexNet					VGG				
Testing Vs.	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$
BP	0.9136	0.89874	0.89634	0.85943	0.12557	0.99068	0.99108	0.78341	0.75145	0.59411
SIFT+FV	0.21866	0.21866	0.00012449	1.6086e-24	1.65e-86	0.16502	0.16502	3.9752e-06	1.516e-31	3.6495e-98
AlexNet	3.4962e-105	3.4962e-105	0	0	0	0.98106	0.98106	0.90152	0.47591	0.050199
VGG	0.64622	0.64621	9.6649e-14	2.4111e-137	0	0	0	0	0	0
ResNet18	0.9711	0.97111	0.92035	0.13873	3.8914e-111	0.37065	0.37065	2.124e-25	1.5714e-103	1.1938e-291
ResNet101	0.90558	0.90557	0.37347	3.1411e-66	4.1528e-240	0.35338	0.35338	3.3466e-53	1.3622e-216	0

Painting Landscapes

	AlexNet					VGG				
Testing Vs.	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$
BP	1	1	1	1	1	1	1	1	1	1
SIFT+FV	0.79505	0.79505	0.30022	0.0024393	3.3167e-09	0.79599	0.79599	0.3353	0.0019852	3.6761e-09
AlexNet	7.9291e-10	7.9291e-10	1.4342e-29	5.265e-33	8.3707e-33	0.9905	0.9905	0.95413	0.95212	0.70978
VGG	0.87445	0.87447	0.38584	3.5473e-12	7.3903e-34	5.0409e-29	5.0409e-29	5.5977e-37	3.0354e-37	1.0586e-38
ResNet18	0.89967	0.89966	0.7306	0.30211	1.4175e-05	0.68713	0.68713	0.030198	1.4375e-05	7.5887e-15
ResNet101	0.9671	0.96696	0.74545	5.8498e-08	2.5216e-22	0.87901	0.87901	0.33332	2.368e-17	3.0898e-30

Drawings

	AlexNet					VGG				
Testing Vs.	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$
BP	0.98405	0.98405	0.97876	0.95426	0.97483	0.98745	0.98745	0.99322	0.98889	0.97186
SIFT+FV	0.67854	0.67854	0.43269	0.04983	2.6184e-06	0.69315	0.69315	0.46697	0.064349	1.0753e-05
AlexNet	1.5308e-05	1.5308e-05	3.0146e-26	5.9791e-41	6.2069e-47	0.92991	0.92989	0.83881	0.84904	0.0052409
VGG	0.87066	0.87066	0.058863	9.9068e-08	3.7992e-30	1.5939e-20	1.5939e-20	1.7229e-53	3.4676e-58	6.0516e-61
ResNet18	0.90781	0.90781	0.50143	0.0023366	5.16e-16	0.59103	0.59103	0.0056782	6.2775e-08	9.5421e-26
ResNet101	0.97912	0.97912	0.55583	3.2526e-06	1.313e-22	0.79099	0.79099	0.00070622	4.8929e-12	1.5062e-33

Sculpture

	AlexNet					VGG				
Testing Vs.	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$
BP	0.99772	0.99772	0.99742	0.99748	0.99701	0.99736	0.99736	0.99717	0.99702	0.99753
SIFT+FV	0.89198	0.89198	0.4632	0.1714	0.086323	0.93114	0.93114	0.5201	0.20871	0.14017
AlexNet	0.00013513	0.00013513	2.0152e-20	8.8035e-31	8.2946e-34	0.89484	0.89484	0.54423	0.00026161	1.6579e-12
VGG	0.75594	0.75594	0.0082156	5.2489e-08	1.6714e-21	6.7774e-13	6.7774e-13	1.0801e-30	7.3611e-34	2.2997e-36
ResNet18	0.76905	0.76905	0.053247	2.3723e-06	7.1788e-18	0.6501	0.6501	0.0032158	1.3364e-08	4.5108e-21
ResNet101	0.83153	0.83153	0.018017	6.1378e-07	2.1775e-14	0.73046	0.73046	0.0025386	2.3318e-07	3.531e-15

Engraving BW

	AlexNet					VGG				
Testing Vs.	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$
BP	0.1314	0.1314	0.75089	0.7213	0.76281	0.1184	0.1184	0.69932	0.72718	0.75278
SIFT+FV	0.89606	0.89606	0.9213	0.77775	0.01329	0.87447	0.87447	0.85628	0.96241	0.086893
AlexNet	5.358e-33	5.358e-33	2.205e-116	3.5771e-171	2.9344e-187	0.61108	0.61108	0.79384	0.66623	1.2529e-07
VGG	0.94146	0.94146	0.62982	0.019374	0.092515	1.7458e-85	1.7458e-85	2.3367e-202	6.0154e-219	3.4579e-219
ResNet18	0.35122	0.35122	0.020752	6.4226e-13	4.1326e-56	0.11378	0.11378	2.9188e-05	5.2976e-22	1.1364e-73
ResNet101	0.14262	0.14262	0.70567	0.0010867	5.2621e-31	0.4537	0.4537	0.19041	4.4468e-06	1.2934e-39

Engraving Color

	AlexNet					VGG				
Testing Vs.	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$	$\epsilon2$	$\epsilon4$	$\epsilon8$	$\epsilon16$	$\epsilon32$
BP	0.89591	0.89591	0.94823	0.93453	0.98186	0.87265	0.87265	0.92126	0.93354	0.99179
SIFT+FV	0.83715	0.83715	0.7306	0.52109	0.95255	0.78356	0.78356	0.78447	1	0.82826
AlexNet	1.6503e-133	1.6479e-133	4.924e-273	2.106e-290	1.6302e-274	0.56852	0.56852	0.36771	0.041684	1.0253e-05
VGG	3.66e-32	3.66e-32	1.8892e-35	1.7963e-34	1.0132e-18	0	0	0	0	0
ResNet18	1.105e-08	1.105e-08	1.8668e-09	4.1096e-08	6.1218e-06	1.6772e-10	1.6772e-10	1.7619e-14	1.0486e-16	1.5263e-15
ResNet101	5.1941e-16	5.1941e-16	1.1333e-18	6.1089e-25	5.416e-42	8.2181e-23	8.2181e-23	1.968e-41	3.066e-70	3.3115e-107

Table 5.8: This table shows the results from the statistical tests applied to each method's predictions' confidence from clean and attacked images using test datasets and AEs from AlexNet and VGG. Each value represents the corresponding p-value from the statistical test.

Iconography

	ResNet18					ResNet101				
Testing Vs.	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	0.99767	0.99672	0.99739	0.99585	0.99585	0.99992	0.99593	0.99825	0.99589	0.99589
SIFT+FV	0.95457	0.95457	0.78501	0.098796	1.296e-05	0.9245	0.92449	0.71244	0.06041	2.3283e-06
AlexNet	0.99776	0.99776	0.96299	0.74961	0.18328	0.9903	0.9903	0.98065	0.79478	0.62045
VGG	0.45019	0.45018	1.3824e-05	7.6022e-17	2.9553e-39	0.60766	0.60768	0.0020685	3.2656e-09	6.6815e-28
ResNet18	2.8465e-35	2.8465e-35	1.2532e-66	1.0059e-161	1.2666e-66	0.48618	0.48626	0.0005933	1.195e-08	9.7675e-18
ResNet101	0.36979	0.36979	1.161e-05	8.3498e-14	2.6481e-22	5.2645e-31	5.2645e-31	1.3736e-58	7.7789e-61	7.2518e-59

Painting

	ResNet18					ResNet101				
Testing Vs.	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	0.84987	0.78268	0.54929	0.99293	0.9937	0.73759	0.94754	0.87769	0.88685	0.83682
SIFT+FV	0.15684	0.15685	1.1729e-06	9.2235e-34	4.2265e-102	0.1448	0.1448	4.1104e-07	1.6025e-34	6.9011e-99
AlexNet	0.97537	0.97538	0.83884	0.0002221	0.056476	0.98499	0.98501	0.8621	0.39927	0.11822
VGG	1.5516e-06	1.5524e-06	1.4289e-58	1.8098e-237	0	8.3796e-06	8.3735e-06	5.48e-50	1.1918e-222	0
ResNet18	0	0	0	0	0	9.7705e-05	9.7674e-05	3.6504e-28	2.1311e-107	9.0869e-291
ResNet101	0.20263	0.20266	7.6609e-70	1.041e-267	0	0	0	0	0	0

Painting Landscapes

	ResNet18					ResNet101				
Testing Vs.	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	1	1	1	1	1	1	1	1	1	1
SIFT+FV	0.81073	0.81072	0.31041	0.0013292	9.4406e-10	0.79544	0.79544	0.27643	0.0011679	3.2551e-09
AlexNet	0.97698	0.97699	0.93679	0.92671	0.66599	0.99458	0.9946	0.97321	0.97355	0.81807
VGG	0.71643	0.71643	4.7994e-05	1.6994e-17	4.5878e-38	0.73709	0.73709	0.00021444	4.9845e-16	8.8656e-38
ResNet18	1.3222e-24	1.3435e-24	1.8235e-34	5.3646e-34	2.2238e-35	0.60616	0.60616	0.021885	2.1956e-05	1.1705e-16
ResNet101	0.83154	0.83154	1.4205e-06	5.6908e-22	5.671e-33	8.0509e-29	8.0509e-29	7.9778e-37	9.0441e-38	5.7158e-41

Drawings

	ResNet18					ResNet101				
Testing Vs.	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	0.98646	0.98646	0.98853	0.96403	0.97252	0.99539	0.99539	0.99544	0.98031	0.98827
SIFT+FV	0.69545	0.69544	0.45485	0.052022	4.3891e-06	0.68791	0.68791	0.43034	0.039834	2.482e-06
AlexNet	0.91835	0.91835	0.86539	0.77007	0.0013548	0.9221	0.92208	0.86883	0.94347	0.29188
VGG	0.61463	0.61463	0.00064209	2.228e-13	1.8494e-35	0.5806	0.58052	0.0061323	1.4804e-10	5.7541e-34
ResNet18	1.7101e-24	1.7101e-24	1.9766e-51	1.4251e-54	1.2956e-56	0.50031	0.50028	0.00714	1.1845e-06	2.6572e-20
ResNet101	0.59366	0.59366	0.0001009	4.6116e-12	1.5527e-34	4.7426e-30	4.7426e-30	3.8037e-54	1.3584e-57	3.9703e-60

Sculpture

	ResNet18					ResNet101				
Testing Vs.	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	0.9975	0.9975	0.99702	0.99693	0.9969	0.99659	0.99659	0.99686	0.99662	0.99609
SIFT+FV	0.91246	0.91245	0.45842	0.19418	0.23847	0.91303	0.91303	0.40899	0.11917	0.0022884
AlexNet	0.89915	0.89911	0.57069	0.00029538	2.4554e-12	0.92807	0.92807	0.63864	0.0012897	1.0607e-10
VGG	0.71495	0.71495	0.00025963	4.4615e-10	1.6644e-23	0.83568	0.83568	0.0062375	1.065e-06	8.4707e-20
ResNet18	5.4653e-13	5.4653e-13	1.885e-26	2.7286e-29	2.1562e-32	0.8016	0.8016	0.01385	2.2828e-06	1.1366e-17
ResNet101	0.63809	0.63809	0.00097214	2.3315e-09	1.5158e-17	7.4724e-11	7.4724e-11	1.7682e-19	8.4807e-21	1.7814e-25

Engraving BW

	ResNet18					ResNet101				
Testing Vs.	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	0.62139	0.62139	0.76279	0.72754	0.73081	0.23525	0.23525	0.73697	0.75743	0.76744
SIFT+FV	0.88575	0.88575	0.8536	0.95828	0.14382	0.86903	0.86904	0.88072	0.92791	0.065097
AlexNet	0.88255	0.88255	0.82907	0.12103	5.201e-14	0.63177	0.63178	0.85028	0.51736	0.026006
VGG	0.3549	0.3549	0.011117	1.113e-11	1.2312e-46	0.61302	0.61311	0.14793	1.5966e-08	1.9818e-32
ResNet18	2.0619e-81	2.0619e-81	8.5802e-191	2.2357e-211	4.2645e-218	0.10882	0.10882	0.0001122	9.641e-23	5.6127e-80
ResNet101	0.5792	0.5792	0.0092807	5.08e-14	5.8058e-77	4.4317e-56	4.4223e-56	1.1988e-119	1.3737e-152	3.8698e-193

Engraving Color

	ResNet18					ResNet101				
Testing Vs.	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32	ϵ2	ϵ4	ϵ8	ϵ16	ϵ32
BP	0.96553	0.96553	0.93342	0.99733	0.99721	0.96275	0.96275	0.9169	0.98099	0.83034
SIFT+FV	0.89023	0.89023	0.67872	0.66967	0.8903	0.89075	0.89075	0.72948	0.51895	0.8264
AlexNet	0.61061	0.61064	0.4212	0.095421	0.0014581	0.56698	0.56703	0.33301	0.017158	1.6441e-06
VGG	8.8075e-36	8.8007e-36	1.3014e-44	7.2256e-54	7.1826e-59	1.3175e-40	1.323e-40	6.4181e-58	1.2544e-79	1.8917e-105
ResNet18	3.076e-307	3.076e-307	0	0	0	3.907e-11	3.907e-11	5.3445e-16	1.6623e-17	1.5303e-14
ResNet101	4.2e-20	4.1976e-20	4.9361e-32	4.2834e-52	6.6939e-89	2.6025e-310	2.6206e-310	0	0	0

Table 5.9: This table shows the results from the statistical tests applied to each method's predictions' confidence from clean and attacked images using test datasets and AEs from ResNet and ResNet101. Each value represents the corresponding p-value from the statistical test.

Iconography

Testing Vs.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	0.99608	0.99019	0.99812	0.99901
SIFT+FV	0.60131	0.89837	0.61682	0.97909
AlexNet	7.1039e-20	0.79567	0.46915	0.56209
VGG	0.0003743	1.3363e-23	9.9987e-07	1.0847e-07
ResNet18	2.5542e-05	0.00040313	2.092e-18	0.00029613
ResNet101	0.0010507	8.4448e-05	3.9575e-08	1.358e-12

Painting

Testing Vs.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	1	1	1	1
SIFT+FV	0.54497	0.74837	0.55432	0.82595
AlexNet	1.3914e-27	2.7344e-07	4.1336e-07	7.5744e-12
VGG	4.5007e-16	3.0804e-22	5.2464e-14	3.4408e-20
ResNet18	1.2746e-19	1.1991e-15	2.269e-29	4.9202e-26
ResNet101	3.855e-20	1.9447e-20	5.876e-18	9.657e-26

Painting Landscapes

Testing Vs.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	1	1	1	1
SIFT+FV	0.71106	0.35442	0.76126	0.32247
AlexNet	2.3132e-27	1.4598e-08	1.3614e-06	4.8304e-13
VGG	3.4926e-25	5.7961e-30	1.6104e-22	3.0764e-29
ResNet18	1.5553e-28	1.3091e-24	1.1945e-33	5.1204e-32
ResNet101	1.868e-27	5.565e-27	3.9203e-28	1.1313e-30

Drawings

Testing Vs.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	1	1	1	1
SIFT+FV	0.49234	0.24792	0.40336	0.20665
AlexNet	2.0788e-27	2.61e-05	2.1405e-08	7.0741e-12
VGG	3.9051e-15	2.6781e-20	1.1136e-18	3.3552e-21
ResNet18	5.2286e-15	3.8956e-09	1.3091e-24	5.042e-17
ResNet101	3.8469e-17	9.1359e-12	5.7602e-14	1.0681e-22

Sculpture

Testing Vs.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	0.99544	0.99701	0.99823	0.99557
SIFT+FV	0.093228	0.048075	0.14621	0.49341
AlexNet	9.0533e-28	1.6758e-06	1.6128e-08	8.3985e-08
VGG	6.2478e-06	4.5278e-19	1.1145e-10	3.8384e-09
ResNet18	9.011e-06	5.6851e-07	3.0015e-20	6.7596e-08
ResNet101	3.1949e-07	0.00045029	2.9946e-07	1.2838e-08

Engraving BW

Testing Vs.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	0.84032	0.9292	0.83695	0.79371
SIFT+FV	0.17899	0.68646	0.63549	0.75774
AlexNet	6.8616e-16	0.33203	0.13546	0.80088
VGG	2.7464e-15	6.3779e-28	1.1879e-13	1.0857e-15
ResNet18	4.2622e-24	1.3125e-21	5.7495e-32	6.0573e-26
ResNet101	1.5343e-07	1.7745e-10	4.6019e-14	4.6746e-23

Engraving Color

Testing Vs.	AlexNet Patch	VGG Patch	ResNet18 Patch	ResNet101 Patch
BP	0.81987	0.96904	0.82055	0.81363
SIFT+FV	0.51466	1	1	1
AlexNet	4.7445e-15	0.42195	0.6454	0.32844
VGG	0.00057199	5.3585e-08	0.11452	0.000143
ResNet18	0.00012499	0.01838	0.00042032	0.31804
ResNet101	0.33861	0.29463	1.2204e-05	8.898e-10

Table 5.10: This table shows P-values obtained using the adversarial patch on the statistical tests. Each column presents the score obtained for the 100 images per pair (Clean and AE) using the adversarial patch.

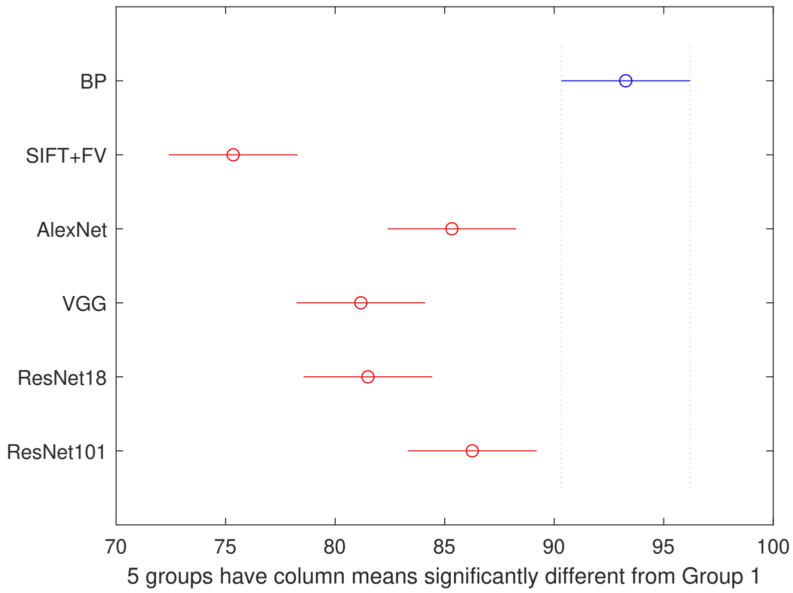


Figure 5.7: Multiple comparisons of group mean from testing data considering the FGSM experiment.

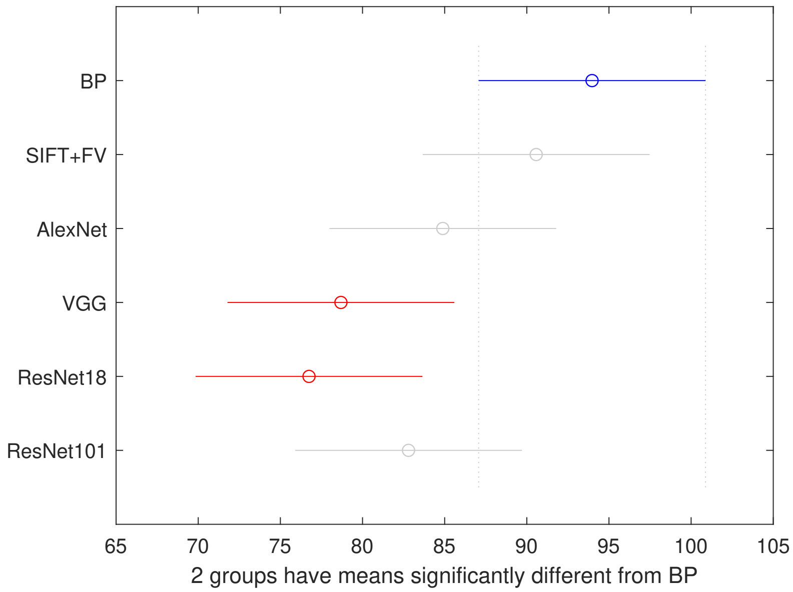


Figure 5.8: Multiple comparisons of group mean from testing data considering the adversarial patch experiment.

Defense mechanisms

In this experiment, we generated new models from the DCNN in three different classes using adversarial training. Next, we computed new AEs from FGSM using the new models to verify the direct attack and the AE transferability. We also test them using the multiple-pixel attack and the previously computed adversarial patch to each model. Table 5.11 presents the results from the experiments. We observed in the first stage that the direct impact from FGSM is reduced, but it still decreases the DCNN's performance in a great manner. In contrast, the AE transferability property between the models was not perceptible even in the strong perturbations. However, the success rate from the multiple-pixel attack obtained almost the same performance as previous models with no defense. In addition, the previously computed adversarial patches affected the new models. The adversarial training diminished the influence of the patch, but it did not provide proof of solving this vulnerability.

	Clean images			AlexNet					VGG					Multiple-pixel		Adversarial patch	
Paintings	train	val	test	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	previous model	new model	previous score	new score
AlexNet	98.48	96.80	92.81	87.73	87.73	73.9	60.34	57.92	92.81	92.81	92.85	92.55	92.68	54	46	54	76
VGG	99.84	97.52	94.01	94.57	94.57	94.53	94.40	93.63	92.16	92.16	84.24	73.21	68.04	60	96	48	87

	Clean images			AlexNet					ResNet18					Multiple-pixel		Adversarial patch	
Drawings	train	val	test	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	previous model	new model	previous score	new score
AlexNet	93.26	90.71	86.96	77.57	77.57	60.18	43.48	35.93	86.73	86.73	85.35	83.52	80.32	68	54	30	56
ResNet18	99.87	93.51	88.56	88.56	88.56	88.56	88.1	86.27	82.84	80.78	65.45	52.86	47.83	74	52	55	86

	Clean images			AlexNet					ResNet101					Multiple-pixel		Adversarial patch	
Sculpture	train	val	test	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	previous model	new model	previous score	new score
AlexNet	98.46	96.36	90.56	84.96	84.96	66.37	43.95	36.28	91.15	91.15	91.74	91.74	89.97	62	60	32	47
ResNet101	100	98.53	94.1	95.28	95.28	94.40	93.81	91.15	83.19	83.19	64.01	55.16	59.29	54	58	87	90

Table 5.11: This table shows the results using adversarial training in all DCNNs models in three different classes. Each method presents its classification accuracy for training, validation, testing, and the adversarial examples using the FGSM computed from the testing dataset at $\epsilon = \{2, 4, 8, 16, 32\}$. The multiple-pixel column presents the attack success rate between the original and the new model (a higher value means worse). The adversarial patch column shows the classification accuracy between the original and the new model to the first precomputed adversarial patches.

5.2 Face Recognition Problem

Face Recognition (FR) is an important research area in Computer Vision (CV), where security is crucial. The face is the most popular biometric among others to recognize persons since it can be acquired in unconstrained environments and, in turn, provide excellent discriminative features for recognition. Hence, FR's security is necessary because it is an essential tool in tasks like video-surveillance, security systems and access control, and many applications in our everyday.

The recent progress of Machine Learning (ML) in many areas of CV has enabled ML algorithms to be adaptable to many research areas like Face Recognition. Commonly, these algorithms are known to achieve exemplary performance in many areas. However, recent studies have demonstrated that

Adversarial Attacks (AA) pose a predicting threat to their success because, with perturbations intentionally created in the input image, they could lead to a wrong prediction.

In this matter, the study of the attack architectures and defense mechanisms to diminish the damage has been a popular research topic. Nevertheless, despite significant efforts to solve this problem, attacks have become more complex and challenging to defend [13]. The attack effects have been only exposed for ML algorithms, unlike Evolutionary Paradigms (EPs), which have not been demonstrated to have such vulnerability yet. Therefore, an EP could function as a solution to AA due to the inability to solve the problem in current ML algorithms such as DCNNs.

Recently, there has been enormous progress in solving the FR problem by introducing massive face databases useful for training very deep architectures of DCNNs [140]. Although FR shares similarities with object recognition, a particular aspect is characteristic of a face: they have a well-structured shape that can be modeled very well. Hence, in FR, the data is preprocessed to appropriately modify the input image to easily learn the face representations [141]. DCNNs have achieved exemplary results in popular databases for FR even some of them were pretrained for generic object recognition [2, 142, 3]. The advantage of the DCNN architectures against other methods is that they can be optimized end-to-end to develop features that amplify the identity signal, improving the identification ability of face recognition systems exploiting the vast training databases available.

Face recognition systems demand reliability and security in their predictions because they are the most popular biometric used for person recognition. Nevertheless, they have been traditionally evaluated with the implicit assumption of no threats that actively attempt to fool the system [143, 144]. Recent studies have encountered on AA a predicting threat to CNN's success because with perturbations intentionally created (some of them are imperceptible to human vision), can completely change the DCNN's prediction to drop its performance, and face recognition systems based on DCNNs are not the exception [11, 12, 145].

For example, subjects can be effectively impersonated using 3d-printed masks or face images downloaded from social networks [146, 147]. Fredrikson et al. presented model inversion attacks, in which face images of enrolled users are reconstructed from neural networks trained compromising the privacy of users enrolled in the system [148]. Feng and Prabhakaran proposed in [149] the use of makeup and hair designs to dodge face recognition systems (including non-CNN architecture such as Eigenfaces [150] and Fisherfaces [151]). Sharif et al. proposed in [99] a physically realizable and inconspicuous attack through printing a pre-computed pair of eyeglass frames to evade recognition.

There have been immense efforts to develop defense mechanisms to mitigate AA [147, 152, 153]. Still, the perturbations have become more complex and highly efficient in fooling CNNs [140]. However, despite the ML community's progress, EC have mostly contributed to developing strategies to search for meaningful DCNN architectures for image classification [19]. Nevertheless, these approaches have fallen short to be on par with hand-craft DCNNs architectures. Additionally, AAs' problem is inherent to the CNN structure, making them susceptible to these attacks.

Moreover, Genetic Programming (GP) has been one of EC's principal tools to optimize the selection of features and automatically extract the best characteristics to approach image classification tasks [43, 45]. However, they are still dealing with outdated problems using classical datasets while making comparisons against CV methods based on handcrafted features and obsolete CNNs. EPs have not demonstrated to be vulnerable against AA yet, and FR could be a suitable application for EPs on which reliability and security in the predictions are required.

5.2.1 Experiment

Reliable predictions are a highly valuable characteristic regarding face recognition system development following security and confidence of the recognition. We propose the assumption of an attempt to fool the system and highlight the differences between the renowned DCNN who has performed well in object and face recognition (ResNet [3]), and an Evolutionary Paradigm (BP) who has obtained comparable results with AlexNet[1]. We take into consideration performance and security against adversarial attacks in the most straightforward face recognition experiment. We use training, validation, and testing stages. The aim is to emulate a real-world scenario where the proposed models employ standard benchmark procedures.

In this experiment, we analyze a threat to evade recognition using a pair of inconspicuous eyeglasses frame. The facial accessories perturbation is a white-box targeted attack that challenges the robustness of such eyeglasses, which can be printed to appear in real-world conditions to cause a misleading prediction of a target class. The analysis helps us to determine whether this attack could be used to evade recognition or can be used for impersonation. We also analyze the AE transferability of such eyeglasses to BP. Lastly, we include a statistical analysis to determine significant differences between each pair of prediction confidences between DCNN and BP at each stage.

5.2.2 Dataset

We use a widely used face recognition dataset named CelebFaces Attributes (CelebA)[154]. It consists of a large-scale face attributes dataset with more than 200K celebrity images, each with 40 attribute annotations. The images in this dataset cover significant pose variations and background clutter. Additionally, it has enormous diversities, large quantities, and rich annotations, including 10,177 identities, 202,599 face images, five landmark locations, 40 binary attributes annotations per image.

Class	training	validation	testing
Faces	1000	300	800
Background	1000	300	800

Table 5.12: Dataset construction from CelebA images.

As the experiment required a significant number of images, in Table 5.12, we provide the number of images for each set of images for training, validation, and testing stages. Hence, we randomly construct three sets of images from the CelebA dataset and manually fine-tune changing images to preserve the face diversity and front-facing images. Additionally, we use a Multi-task Cascaded Convolutional Networks (MTCNN) to perform face detection and alignment using the DeepFace library from [155]. We constructed the background class from landscape images from the ml5 project datasets [156].

5.2.3 Implementation details

In this subsection, we outline the implementation details for the models used:

- Brain Programming: was implemented on Matlab R2018b using a modified version of GP Lab and the libsvm v3.25 [134] library for the SVM.

- DCNN: for the implementation of ResNet, we use the pre-trained model from PyTorch v1.1 [136]. This model was retrained using transfer learning for face recognition.

Also, we outline each of the AA:

- Facial accessories perturbation: was implemented using 100 images from the training dataset in PyTorch v1.1 [136].

The BP algorithm was run in a server with an Intel Xeon Silver 4114 CPU and 32 Gb of RAM and ResNet as well as the facial accessories perturbation were run in a computer with Intel core i7-6700HQ with 24 GB of RAM and graphic card NVIDIA GeForce GTX 1070.

5.2.4 Results

We show in Figure 5.9 the fitness evolution progress of BP in the training phase. It is seen that most of the runs converged to approximately 75% of the validation accuracy, two runs achieved around 80%, and one run obtained approximately 90% (see Table 5.13). Hence, we can see that it is not easy to reach satisfactory solutions in the search space due to the complex structure of the AVC departing from random individuals, but it was possible to obtain an excellent individual. Nevertheless, due to the BP's high computational cost, it was possible only to execute 15 runs with a mean execution time of 40.18 hours and a standard deviation of 1.04 hours in a server with an Intel Xeon Silver 4114 CPU and 32 Gb of RAM.

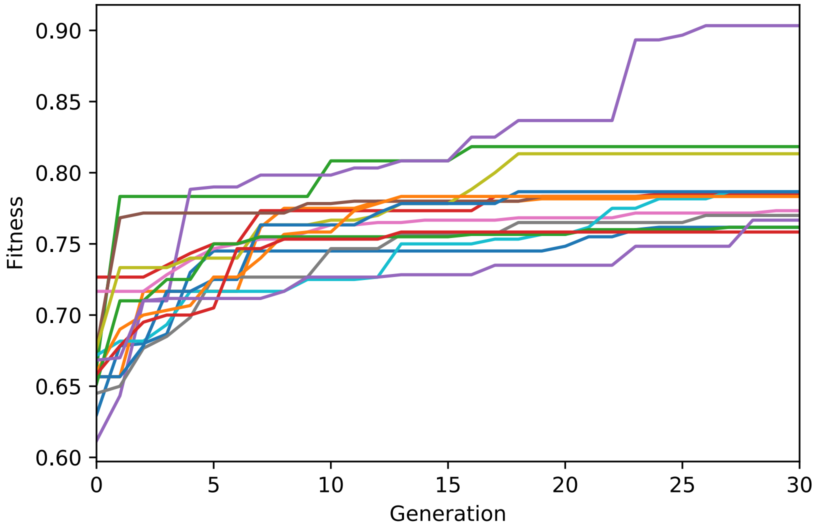


Figure 5.9: Fitness evolution progress for the best solution during the validation stage of BP. Each plot represents one of the 15 runs.

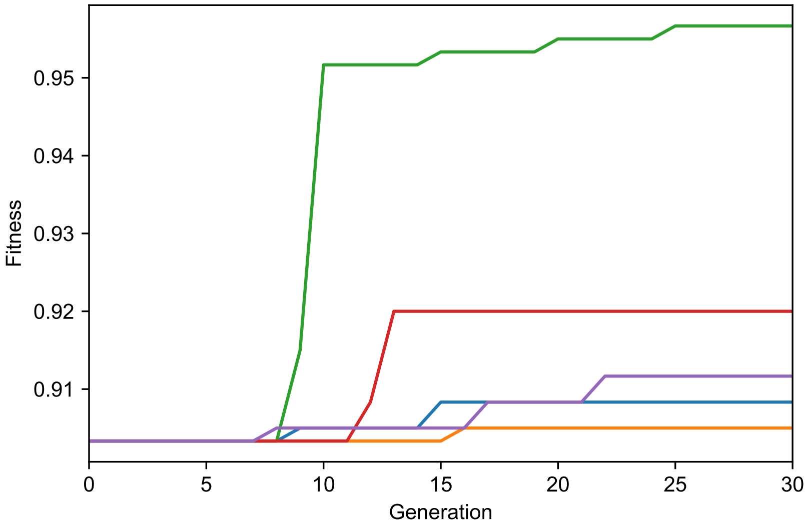


Figure 5.10: Fitness evolution progress for the best solution during the five runs of the hands-on artificial evolution.

Run	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Training	0.7175	0.7705	0.7810	0.7620	0.8925	0.7700	0.7165	0.7605	0.8420	0.7800	0.8065	0.7840	0.7195	0.7180	0.7500
Validation	0.7617	0.7833	0.8183	0.7850	0.9033	0.7833	0.7733	0.7700	0.8133	0.7867	0.7867	0.7833	0.7617	0.7583	0.7667

Table 5.13: Performance of the best individuals of BP in each run.

Therefore, we follow the hands-on artificial evolution strategy from [95] in which we selected the best two individuals from each run to construct an initial population to search for new individuals. Figure 5.10 shows the fitness evolution progress of five runs from this strategy. It can be seen the guide from the previous experiments delivered an increase of up to 5% in the performance. Hence, we validate the advantage of the hands-on evolution strategy to get out of local minima, thus helping the methodology to discover better solutions.

Next, we present in Table 5.14 the outcome of each model for the clean images, and when it is applied, the eyeglasses frame perturbation to the training, validation, and testing datasets. Each method presents its accuracy to each dataset. We observed that ResNet surpassed BP in all sets of clean images (training, validation, and testing). However, as we add the eyeglasses frame perturbation to all sets of face images, the AA's effect becomes enormous. It is shown that ResNet completely drops its outstanding performance as the eyeglasses frame is present in the face images, making almost every face image in the three datasets evade the recognition.

	Clean Images			ResNet Eyeglasses		
	train	val	test	train	val	test
BP	94.1	95.67	93.19	94.3	94.33	95.75
ResNet	99.75	99.67	99.94	4.70	5.67	1.38
h	1	1	1	1	1	1

Table 5.14: Results obtained using the ResNet eyeglass attack. Each method presents its classification accuracy for training, validation, and testing, and the adversarial examples using the pre-computed glasses from ResNet. The third row present results of a two-sample Kolmogorov-Smirnov test between both methods.

The powerful effect of the eyeglasses frame perturbation is seen on the training dataset, where even though ResNet has identified the faces images from the training stage, the perturbation makes them evade the recognition. Meanwhile, BP demonstrated to remain with a maximum variation of 2.56% of its original score, proving its security to recognize faces even the images are perturbed with an eyeglasses frame. Additionally, Table 5.14 shows a two-sample Kolmogorov-Smirnov test between each pair of prediction confidences at each stage between DCNN and BP. h is one if the test rejects the null hypothesis at the 5% significance level and 0 otherwise. Hence, all the prediction confidences at each stage between both methods denoted significant differences by rejecting the null hypothesis.

This experiment innovates by introducing the assumption of an attempt to fool the system and highlighting the differences between a renowned DCNN (ResNet) and the AVCs generated by BP. We also considered contrast performance and security against adversarial attacks in the most explicit face recognition problem. ResNet's performance was extremely weakened using the facial accessories perturbation. In contrast, the AVC models from BP resist the attempt to mislead the system. Additionally, a two-sample Kolmogorov-Smirnov test confirmed that DCNN and the AVC models designed with BP are statistically different. These results open the possibility of using evolutionary computation in the face recognition pipeline to protect the predictions. Furthermore, we validate the hands-on strategy beyond a pure random initialization that helped get out from the local minima to discover better solutions.

Conclusions

Robustness against AA must be the primary concern when we are developing an automatic recognition system. From now on, a classifier's performance should not only focus on accuracy but also on robustness to AAs. In this thesis, we present an empirical study for AMC and FR subject to AA. We compare several methods to analyze the performance and their reliability to predict a class using adversarial perturbations.

For the AMC problem, we selected six models using three of the main approaches for image classification: 1) handcrafted features approach (SIFT+FV), 2) deep genetic programming approach (BP), and 3) DCNN approach (AlexNet, VGG, ResNet18, and ResNet101). The comparative study consists of analyzing three different attacks:

1. Analyze the direct threat's impact and transferability considering the white box untargeted attack–FGSM. This perturbation adds a subtle texture to the artwork, which can cause a misleading prediction.

2. Find a set of localization and pixel values to modify the artwork to fool the classifier using a black box untargeted attack–multiple pixel attack.

3. Apply precomputed patches–adversarial patch–robust to transformations located randomly in the artwork to predict a targeted class.

In this sense, this study has confirmed that AA is a severe threat to the performance of DCNN considering the AMC problem. Using FGSM showed that if the attacker knows the model, it can make the DCNN decrease its performance up to less than 20% of its original score. Additionally, we corroborated the AE transferability property between DCNN models, which is not severe for the binary classification, but it can reduce up to 20% of the performance. On the other hand, SIFT+FV also was affected by some of the classes but by a minor amount. However, the added texture caused by the FGSM leads to a decrease in its performance in a significant manner when testing the algorithm, having encouraging results but not suitable to compete with DCNNs in the testing phase. Finally, BP exhibits comparable performance (efficiency) to DCNN in both validation and testing phases. Furthermore, it has an almost imperceptible variant on its accuracy to these perturbations proving no direct transferability from other models. Figures 5.4-5.5 show the output of each stage of BP from clean and AEs with almost no variation on its outcomes.

The study about a one-pixel attack confirms this type of attack's poor design due to a minimal scenario contrived with an input image of size 32×32 pixels. We conclude that it is challenging to apply single-pixel attacks on real-world conditions. Also, it is hard to apply it in multiple-pixel attacks due to the following factors. On the one hand, when we extend to multiple pixels, the perturbation loses the attack's intention of imperceptible to human vision. On the other hand, a massive amount of processing time. The robustness of BP shows that it is a challenge to make it fail against these attacks even by increasing the number of pixels per attack by five times compared to the SIFT + FV and DCNN models. Finally, the adversarial patch showed that a precomputed perturbation positioned in a random location and orientation in the artwork could fool DCNN models with excellent transferability between them; meanwhile, BP and SIFT+FV remain in their original score. It is remarkable the BP robustness to the multiple pixel attack and the adversarial patch. However, these two attacks are harsh perturbations, and BP remained steady in its performance, leading to the reliability of BP's predictions in no human supervision cases.

The statistical analysis from the predictions' confidence supports the study of robustness by illustrating the change in the posterior probability complementing the results from the accuracy's standpoint. In this manner, BP demonstrated to have not significantly different predictions' confidence compare to DCNN models, which showed in most cases the rejection of the null hypothesis Ho. Conversely, SIFT+FV obtained good results, with most of the test scoring a not significant difference in the predictions' confidence. Also, the comparison between all models using the accuracy proves a rejection of the null hypothesis that all group means are equal in the FGSM and the adversarial patch experiments. Specifically, BP exhibits significant differences from the rest of the classifiers using the multiple comparisons with Bonferroni method in the FGSM experiment. BP highlighted among all groups in the adversarial patch data means showing significant differences to VGG and ResNet18.

Lastly, defense mechanisms proposed particular solutions to the AA problem. The adversarial training showed to diminish the effect of FGSM. However, it could not provide any defense to the multiple-pixel and adversarial patch. Defense mechanism implies that we must add a defense for every attack, making it impractical and difficult to implement when new and complex attacks are made. Therefore, we present BP as a solution to the AA problem, which proposes a different approach that competes with DCNN and does not suffer this vulnerability.

Art media categorization is a complex problem that involves high-resolution images and the inclusion of many artifacts and textures, where it is difficult to outperform DCNN performance. BP has obtained comparable results to DCNN models, but it demonstrated immunity to these adversarial attacks with no direct transferability of such perturbations to the model. On the other hand, SIFT+FV proves to be robust for a limited number of experiments with moderate results. Therefore, BP arises as an alternative to DCNN for an art media classifier approach without the vulnerabilities of AA because it takes advantage of the symbolic representations and incorporates expert systems rules in a hierarchical structure to solve the AMC problem. Additionally, BP opens the possibility of being explainable within each stage, unlike DCNN, an important research area to precisely know the model's inner workings.

For the FR problem, AA confirm to be a severe threat to the security of DCNNs. Their performance can be extremely manipulated with a physically realizable and inconspicuous eyeglasses frame to evade recognition. However, BP has demonstrated to automatically design AVC models capable of competing with DCNN and safeguarded the predictions' integrity by remaining steady in its performance. This thesis innovates by introducing the assumption of an attempt to fool the system and highlighting the differences between a renowned DCNN and an Evolutionary Paradigm by considering performance and security against adversarial attacks in the most straightforward face recognition experiment.

The Facial Accessories Perturbation demonstrated the vulnerability of a DCNN in contrast to the EP. ResNet's performance was extremely weakened in this experiment. In contrast, the AVC models from BP resist the attempt to mislead the system. Additionally, a two-sample Kolmogorov-Smirnov test confirmed that DCNN and the AVC models designed with BP are statistically different. These results open the possibility of using evolutionary computation in the face recognition and artwork classification pipeline to protect the predictions. Furthermore, we validate the hands-on strategy beyond a pure random initialization that helped get out from the local minima to discover better solutions.

The BP's immunity to AA is a significant breakthrough to the EC community where this feature could be an edge compared to machine learning techniques. Due to the enormous advance that DL has brought to the state-of-the-art, many research areas have not been on par with DL. However, demonstrating trustworthiness is another manner to compete with such models. This example could be just the beginning of the secure era of EC techniques.

Future Work

As future work from this thesis, we consider the following research topics. First, as we made an empirical study of robustness beyond deep learning systems, we plan to work on the theoretical reasons why BP is immune to adversarial attacks. We expect to find the main features in the BP architecture that make it immune to employing it in other classification algorithms. Secondly, we want to extend the robustness study to demonstrate the BP's immunity to other research areas that involve security solutions such as person re-identification, pedestrian detection, among others.

Genetic programming algorithms' limitation of binary classification is a significant constraint in developing image classification systems. Therefore, we want to improve the BP architecture to solve multi-class problems with the advantage of robustness to adversarial attacks. Finally, we want to explore the possibility of designing adversarial attacks to algorithms beyond deep learning either in other research areas or for BP to discover vulnerabilities.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, p. 14, 2015.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[4] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach.* Prentice Hall, 2020.

[5] C. M. Bishop, "Machine learning and pattern recognition," *Information science and statistics. Springer, Heidelberg*, 2006.

[6] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on Statistical Learning in Computer Vision, European Conference on Computer Vision (ECCV)*, vol. 1, pp. 1–2, Prague, 2004.

[7] P. Druzhkov and V. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recognition and Image Analysis*, vol. 26, pp. 9–15, 2016.

[8] B. Zhao, J. Feng, X. Wu, and S. Yan, "A survey on deep learning-based fine-grained object classification and semantic segmentation," *International Journal of Automation and Computing*, vol. 14, pp. 119–135, 2017.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[10] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2001.

[11] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014, Conference Track Proceedings*, p. 10, 2014.

[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, p. 11, 2015.

[13] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[14] Y. Li and Y. Wang, "Defense against adversarial attacks in deep learning," *Applied Sciences*, vol. 9, p. 76, Dec 2018.

[15] M. Ozdag, "Adversarial attacks and defenses against deep neural networks: A survey," *Procedia Computer Science*, vol. 140, pp. 152 – 161, 2018. Cyber Physical Systems and Deep Learning Chicago, Illinois November 5-7, 2018.

[16] T. Chen, J. Liu, Y. Xiang, W. Niu, E. Tong, and Z. Han, "Adversarial attack and defense in reinforcement learning-from ai security view," *Cybersecurity*, vol. 2, no. 1, p. 11, 2019.

[17] H. Xu, Y. Ma, H.-C. Liu, D. Deb, H. Liu, J.-L. Tang, and A. K. Jain, "Adversarial attacks and defenses in images, graphs and text: A review," *International Journal of Automation and Computing*, vol. 17, no. 2, pp. 151–178, 2020.

[18] K. Ren, T. Zheng, Z. Qin, and X. Liu, "Adversarial attacks and defenses in deep learning," *Engineering*, vol. 6, no. 3, pp. 346 – 360, 2020.

[19] A. Darwish, A. Hassanien, and S. Das, "A survey of swarm and evolutionary computing approaches for deep learning," *Artificial Intelligence Review*, vol. 53, pp. 1767–1812, 2019.

[20] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "Evolving deep convolutional neural networks for image classification," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 394–407, 2020.

[21] M. Suganuma, S. Shirakawa, and T. Nagao, "A genetic programming approach to designing convolutional neural network architectures," in *Proceedings of the genetic and evolutionary computation conference*, pp. 497–504, 2017.

[22] T. Nakane, B. Naranchimeg, H. Sun, X. Lu, T. Akashi, and C. Zhang, "Application of evolutionary and swarm optimization in computer vision: a literature survey," *IPSJ Transactions on Computer Vision and Applications*, vol. 12, pp. 1–34, 2020.

[23] D. E. Hernández, E. Clemente, G. Olague, and J. L. Briseño, "Evolutionary multi-objective visual cortex for object classification in natural images," *Journal of Computational Science*, vol. 17, pp. 216 – 233, 2016.

[24] G. Olague, E. Clemente, D. E. Hernández, A. Barrera, M. Chan-Ley, and S. Bakshi, "Artificial visual cortex and random search for object categorization," *IEEE Access*, vol. 7, pp. 54054–54072, 2019.

[25] M. Chan-Ley and G. Olague, "Categorization of digitized artworks by media with brain programming.," *Applied Optics*, vol. 59 14, pp. 4437–4447, 2020.

[26] R. Szeliski, *Computer vision: algorithms and applications.* Springer, 2020.

[27] G. Olague, *Evolutionary computer vision: the first footprints.* Springer, 2016.

[28] J. Farquhar, S. Szedmak, H. Meng, and J. Shawe-Taylor, "Improving" bag-of-keypoints" image categorisation: Generative models and pdf-kernels," 2005.

[29] F. Perronnin, C. Dance, G. Csurka, and M. Bressan, “Adapted vocabularies for generic visual categorization,” in *European Conference on Computer Vision (ECCV)*, pp. 464–475, Springer, 2006.

[30] J. C. Van Gemert, C. J. Veenman, A. W. Smeulders, and J.-M. Geusebroek, “Visual word ambiguity,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1271–1283, 2009.

[31] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 2169–2178, IEEE, 2006.

[32] H. Lee, A. Battle, R. Raina, and A. Y. Ng, “Efficient sparse coding algorithms,” in *Advances in Neural Information Processing Systems*, pp. 801–808, 2007.

[33] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, “Locality-constrained linear coding for image classification,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3360–3367, Citeseer, 2010.

[34] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, “Learning mid-level features for recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2559–2566, Citeseer, 2010.

[35] Y. He, K. Kavukcuoglu, Y. Wang, A. Szlam, and Y. Qi, “Unsupervised feature learning by deep sparse coding,” in *SIAM International Conference on Data Mining*, pp. 902–910, SIAM, 2014.

[36] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” 2005.

[37] Q. Zhang and B. Li, “Discriminative k-svd for dictionary learning in face recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2691–2698, IEEE, 2010.

[38] Z. Jiang, Z. Lin, and L. S. Davis, “Learning a discriminative dictionary for sparse coding via label consistent k-svd,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1697–1704, IEEE, 2011.

[39] A. Coates, A. Ng, and H. Lee, “An analysis of single-layer networks in unsupervised feature learning,” in *14th International Conference on Artificial Intelligence and Statistics*, pp. 215–223, 2011.

[40] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep fisher networks for large-scale image classification,” in *Advances in Neural Information Processing Systems*, pp. 163–171, 2013.

[41] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[42] Y. Bi, M. Zhang, and B. Xue, “Genetic programming for automatic global and local feature extraction to image classification,” *2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, 2018.

[43] S. R. Price, D. Anderson, and S. Price, “Goofed: Extracting advanced features for image classification via improved genetic programming,” *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1596–1603, 2019.

[44] M. Iqbal, H. Al-Sahaf, B. Xue, and M. Zhang, "Genetic programming with transfer learning for texture image classification," *Soft Computing*, pp. 1–13, 2019.

[45] Y. Bi, B. Xue, and M. Zhang, "An effective feature learning approach using genetic programming with image descriptors for image classification [research frontier]," *IEEE Computational Intelligence Magazine*, vol. 15, pp. 65–77, 2020.

[46] Y. Bi, B. Xue, and M. Zhang, "Instance selection-based surrogate-assisted genetic programming for feature learning in image classification," *IEEE Transactions on Cybernetics*, pp. 1–15, 2021.

[47] Y. Bi, B. Xue, and M. Zhang, "Dual-tree genetic programming for few-shot image classification," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2021.

[48] F. E. F. Junior and G. Yen, "Particle swarm optimization of deep neural networks architectures for image classification," *Swarm Evol. Comput.*, vol. 49, pp. 62–74, 2019.

[49] Z. Lu, I. Whalen, Y. Dhebar, K. Deb, E. D. Goodman, W. Banzhaf, and V. N. Boddeti, "Multiobjective evolutionary design of deep convolutional neural networks for image classification," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 2, pp. 277–291, 2021.

[50] X. He, Y. Wang, X. Wang, W. Huang, S. Zhao, and X. Chen, "Simple-encoded evolving convolutional neural network and its application to skin disease image classification," *Swarm and Evolutionary Computation*, vol. 67, p. 100955, 2021.

[51] P. Singh, S. Chaudhury, and B. K. Panigrahi, "Hybrid mpso-cnn: Multi-level particle swarm optimized hyperparameters of convolutional neural network," *Swarm and Evolutionary Computation*, vol. 63, p. 100863, 2021.

[52] A. Darwish, D. Ezzat, and A. E. Hassanien, "An optimized model based on convolutional neural networks and orthogonal learning particle swarm optimization algorithm for plant diseases diagnosis," *Swarm and Evolutionary Computation*, vol. 52, p. 100616, 2020.

[53] Y. Wang, H. Zhang, and G. Zhang, "cpso-cnn: An efficient pso-based algorithm for fine-tuning hyper-parameters of convolutional neural networks," *Swarm and Evolutionary Computation*, vol. 49, pp. 114–123, 2019.

[54] D. E. Hernández, G. Olague, B. Hernández, and E. Clemente, "Cuda-based parallelization of a bio-inspired model for fast object classification," *Neural Computing and Applications*, vol. 30, pp. 3007–3018, 2017.

[55] G. Olague, D. E. Hernández, E. Clemente, and M. Chan-Ley, "Evolving head tracking routines with brain programming," *IEEE Access*, vol. 6, pp. 26254–26270, 2018.

[56] G. Olague, D. E. Hernández, P. Llamas, E. Clemente, and J. L. Briseño, "Brain programming as a new strategy to create visual routines for object tracking," *Multimedia Tools and Applications*, vol. 78, no. 5, pp. 5881–5918, 2019.

[57] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1765–1773, 2017.

[58] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," in *6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings*, p. 20, 2018.

[59] T. Miyato, A. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," *5rd International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*, 2017.

[60] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," in *3rd International Conference on Learning Representations, ICLR, Conference Track Proceedings*, p. 9, 2015.

[61] A. S. Ross and F. Doshi-Velez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients," in *32nd AAAI Conference on Artificial Intelligence*, 2018.

[62] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 582–597, IEEE, 2016.

[63] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *ACM SIGSAC Conference on Computer and Communications Security*, pp. 135–147, 2017.

[64] S. Sarkar, A. Bansal, U. Mahbub, and R. Chellappa, "Upset and angri: breaking high performance image classifiers," *arXiv preprint arXiv:1707.01159*, 2017.

[65] S. Baluja and I. Fischer, "Adversarial transformation networks: Learning to generate adversarial examples," *arXiv preprint arXiv:1703.09387*, 2017.

[66] J. Su, D. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, pp. 828–841, 2019.

[67] L. Chen, Z. Xu, Q. Li, J. Peng, S. Wang, and H. Li, "An empirical study of adversarial examples on remote sensing image scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 9, pp. 7419–7433, 2021.

[68] G. Ughi, V. Abrol, and J. Tanner, "An empirical study of derivative-free-optimization algorithms for targeted black-box attacks in deep neural networks," *Optimization and Engineering*, Jun 2021.

[69] X. Li, S. Ji, M. Han, J. Ji, Z. Ren, Y. Liu, and C. Wu, "Adversarial examples versus cloud-based detectors: A black-box empirical study," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 4, pp. 1933–1949, 2021.

[70] Z. Zeng and D. Xiong, "An empirical study on adversarial attack on NMT: Languages and positions matter," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, (Online), pp. 454–460, Association for Computational Linguistics, Aug. 2021.

[71] J. Yoon, K. Kim, and J. Jang, "Propagated perturbation of adversarial attack for well-known cnns: Empirical study and its explanation," *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 4226–4234, 2019.

[72] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, (New York, NY, USA), p. 473–480, Association for Computing Machinery, 2007.

[73] C. Pestana, W. Liu, D. G. Glance, and A. S. Mian, "Defense-friendly images in adversarial attacks: Dataset and metrics for perturbation difficulty," *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 556–565, 2021.

[74] Y. Duan, X. Zhou, J. Zou, J. Qiu, J. Zhang, and Z. Pan, "Mask-guided noise restriction adversarial attacks for image classification," *Computer Security*, vol. 100, p. 102111, 2021.

[75] H. Hirano, A. Minagi, and K. Takemoto, "Universal adversarial attacks on deep neural networks for medical image classification," *BMC Medical Imaging*, vol. 21, p. 9, 2021.

[76] H. Lee, H. Bae, and S. Yoon, "Gradient masking of label smoothing in adversarial robustness," *IEEE Access*, vol. 9, pp. 6453–6464, 2021.

[77] C. Xie, M. Tan, B. Gong, J. Wang, A. L. Yuille, and Q. V. Le, "Adversarial examples improve image recognition," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 816–825, 2020.

[78] C. Zhang, P. Benz, T. Imtiaz, and I. S. Kweon, "Understanding adversarial examples from the mutual influence of images and perturbations," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14509–14518, 2020.

[79] E. J. Kim, J. Rego, Y. Z. Watkins, and G. T. Kenyon, "Modeling biological immunity to adversarial examples," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4665–4674, 2020.

[80] I. Oregi, J. D. Ser, A. P. Martínez, and J. A. Lozano, "Robust image classification against adversarial attacks using elastic similarity measures between edge count sequences," *Neural networks : the official journal of the International Neural Network Society*, vol. 128, pp. 61–72, 2020.

[81] B. Mehlig, *Machine Learning with Neural Networks: An Introduction for Scientists and Engineers.* Cambridge University Press, 2021.

[82] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *International Journal of Computer Vision*, vol. 105, no. 3, pp. 222–245, 2013.

[83] D. M. Titterington, A. F. Smith, and U. E. Makov, *Statistical analysis of finite mixture distributions.* Wiley, 1985.

[84] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2325–2383, 1998.

[85] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *The Journal of physiology*, vol. 195, no. 1, pp. 215–243, 1968.

[86] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.

[87] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, pp. 1–62, 2020.

[88] R. Poli, W. Langdon, and N. McPhee, "A field guide to genetic programming," 2008.

[89] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive Psychology*, vol. 12, no. 1, pp. 97–136, 1980.

[90] M. A. Goodale and A. Milner, "Separate visual pathways for perception and action," *Trends in Neurosciences*, vol. 15, no. 1, pp. 20 – 25, 1992.

[91] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry.," *Human Neurobiology*, vol. 4 4, pp. 219–27, 1985.

[92] K. Fukushima, "Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193–202, 1980.

[93] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, nov 1999.

[94] S. Luke and L. Panait, "Lexicographic parsimony pressure," in *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, GECCO'02, (San Francisco, CA, USA), p. 829–836, Morgan Kaufmann Publishers Inc., 2002.

[95] G. Olague and M. Chan-Ley, *Hands-on Artificial Evolution Through Brain Programming*, pp. 227–253. Cham: Genetic Programming Theory and Practice XVII, Springer International Publishing, 2020.

[96] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*, p. 17, 2017.

[97] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2010.

[98] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," *31st Conference on Neural Information Processing Systems, NIPS*, p. 6, 2017.

[99] M. Sharif, S. Bhagavatula, L. Bauer, and M. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[100] S. Sankaranarayanan, A. Jain, R. Chellappa, and S.-N. Lim, "Regularizing deep networks using efficient layerwise adversarial training," in *AAAI*, 2018.

[101] N. Carlini and D. A. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017.

[102] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, pp. 3–18, 2011.

[103] S. García, A. Fernández, J. Luengo, and F. Herrera, "A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability," *Soft Computing*, vol. 13, pp. 959–977, 2009.

[104] F. F. de Vega, G. Olague, D. Lanza, O. F.Chavezdela, W. Banzhaf, E. Goodman, J. Menendez-Clavijo, and A. Martínez, "Time and individual duration in genetic programming," *IEEE Access*, vol. 8, pp. 38692–38713, 2020.

[105] J. M. Bland and D. G. Altman, "Multiple significance tests: the bonferroni method," *BMJ*, vol. 310, no. 6973, p. 170, 1995.

[106] S. Lee and D. Lee, "What is the proper way to apply the multiple comparison test?," *Korean Journal of Anesthesiology*, vol. 71, pp. 353 – 360, 2018.

[107] S. Chen, Z. Feng, and X. Yi, "A general introduction to adjustment for multiple comparisons.," *Journal of thoracic disease*, vol. 9 6, pp. 1725–1729, 2017.

[108] J. Tukey, "Comparing individual means in the analysis of variance.," *Biometrics*, vol. 5 2, pp. 99–114, 1949.

[109] H. Scheffé, "A method for judging all contrasts in the analysis of variance," *Biometrika*, vol. 40, no. 1-2, pp. 87–110, 1953.

[110] Z. Falomir, L. Museros, I. Sanz, and L. Gonzalez-Abril, "Categorizing paintings in art styles based on qualitative color descriptors, quantitative global features and machine learning (qart-learn)," *Expert Systems with Applications*, vol. 97, pp. 83 – 94, 2018.

[111] L. Kong, J. Lv, M. Li, and H. Zhang, "Extracting generic features of artistic style via deep convolutional neural network," in *International Conference on Video and Image Processing*, ICVIP 2017, p. 119–123, 2017.

[112] A. Elgammal, M. Mazzone, B. Liu, D.-E. Kim, and M. Elhoseiny, "The shape of art history in the eyes of the machine," in *32nd AAAI Conference on Artificial Intelligence*, 2018.

[113] J. Fišer, O. Jamriška, D. Simons, E. Shechtman, J. Lu, P. Asente, M. Lukáč, and D. Sýkora, "Example-based synthesis of stylized facial animations," *ACM Trans. Graph.*, vol. 36, July 2017.

[114] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423, 2016.

[115] D. Keren, "Painter identification using local features and naive bayes," in *Object recognition supported by user interaction for service robots*, vol. 2, pp. 474–477 vol.2, 2002.

[116] J. Li and J. Z. Wang, "Studying digital imagery of ancient paintings by mixtures of stochastic models," *IEEE Transactions on Image Processing*, vol. 13, no. 3, pp. 340–353, 2004.

[117] R. S. Arora and A. Elgammal, "Towards automated classification of fine-art painting style: A comparative study," in *21st International Conference on Pattern Recognition (ICPR)*, pp. 3541–3544, IEEE, 2012.

[118] P. Rosado, "Computer vision models to categorize art collections according to the visual content: A new approach to the abstract art of antoni tàpies," *Leonardo*, vol. 52, pp. 255–260, 2019.

[119] S. Karayev, M. Trentacoste, H. Han, A. Agarwala, T. Darrell, A. Hertzmann, and H. Winnemöller, "Recognizing image style," in *British Machine Vision Conference*, 2014.

[120] Y. Bar, N. Levy, and L. Wolf, "Classification of artistic styles using binarized features derived from a deep neural network," in *European Conference on Computer Vision (ECCV)*, pp. 71–84, Springer, 2014.

[121] N. van Noord, E. Hendriks, and E. Postma, "Toward discovery of the artist's style: Learning to recognize artists by their artworks," *IEEE Signal Processing Magazine*, vol. 32, pp. 46–54, 2015.

[122] E. Cetinic and S. Grgic, "Genre classification of paintings," in *International Symposium ELMAR*, pp. 201–204, 2016.

[123] B. Seguin, C. Striolo, I. diLenardo, and F. Kaplan, "Visual link retrieval in a database of paintings," in *European Conference on Computer Vision (ECCV)*, pp. 201–204, 2016.

[124] T. Sun, Y. Wang, J. Yang, and X. Hu, "Convolution neural networks with two pathways for image style recognition," *IEEE Transactions on Image Processing*, vol. 26, pp. 4102–4113, 2017.

[125] A. Elgammal, Y. Kang, and M. D. Leeuw, "Picasso, matisse, or a fake? automated analysis of drawings at the stroke level for attribution and authentication," in *32nd AAAI Conference on Artificial Intelligence*, 2018.

[126] E. Cetinic, T. Lipic, and S. Grgic, "Fine-tuning convolutional neural networks for fine art classification," *Expert Systems With Applications*, vol. 114, pp. 107–118, 2018.

[127] H. Yang and K. Min, "Classification of basic artistic media based on a deep convolutional approach," *The Visual Computer*, vol. 36, no. 3, pp. 559–578, 2020.

[128] S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the robustness of deep neural networks via stability training," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4480–4488, 2016.

[129] D. Su, H. Zhang, H. Chen, J. Yi, P.-Y. Chen, and Y. Gao, "Is robustness the cost of accuracy? – a comprehensive study on the robustness of 18 deep image classification models," in *European Conference on Computer Vision (ECCV)* (V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, eds.), (Cham), pp. 644–661, Springer International Publishing, 2018.

[130] J.-Y. Baek, Y.-S. Yoo, and S.-H. Bae, "Adversarial learning with knowledge of image classification for improving gans," *IEEE Access*, vol. 7, pp. 56591–56605, 2019.

[131] N. Akhtar, J. Liu, and A. Mian, "Defense against universal adversarial perturbations," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3389–3398, 2018.

[132] H. Zhang, H. Chen, Z. Song, D. Boning, I. S. Dhillon, and C.-J. Hsieh, "The limitations of adversarial training and the blind-spot attack," in *7th International Conference on Learning Representations, ICLR 2019, Conference Track Proceedings*, 2019.

[133] S. Silva and J. Almeida, "Gplab-a genetic programming toolbox for matlab," *Proceedings of the Nordic MATLAB Conference*, 07 2008. Software available at http://gplab.sourceforge.net/download.html.

[134] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[135] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms." http://www.vlfeat.org/, 2008.

[136] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019. https://pytorch.org/.

[137] F. Biscani and D. Izzo, "A parallel global multiobjective framework for optimization: pagmo," *Journal of Open Source Software*, vol. 5, no. 53, p. 2338, 2020.

[138] M. Buehren, "Differential evolution." https://www.mathworks.com/matlabcentral/fileexchange/18593-differential-evolution, 2020.

[139] N. Papernot, P. Mcdaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," *ArXiv*, vol. abs/1605.07277, 2016.

[140] I. Masi, Y. Wu, T. Hassner, and P. Natarajan, "Deep face recognition: A survey," *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pp. 471–478, 2018.

[141] F. H. B. Zavan, N. Gasparin, J. Batista, L. P. e Silva, V. Albiero, O. Bellon, and L. Silva, "Face analysis in the wild," *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, pp. 9–16, 2017.

[142] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, 2014.

[143] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *BMVC*, 2015.

[144] R. Chaturvedi and K. Waghmare, "Techniques for facial recognition system:survey," *Journal of emerging technologies and innovative research*, 2020.

[145] F. V. Massoli, F. Carrara, G. Amato, and F. Falchi, "Detection of face recognition adversarial attacks," *Computer Vision and Image Understanding*, vol. 202, p. 103103, 2021.

[146] N. Erdogmus and S. Marcel, "Spoofing in 2d face recognition with 3d masks and anti-spoofing with kinect," *2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pp. 1–6, 2013.

[147] Y. Li, K. Xu, Q. Yan, Y. Li, and R. Deng, "Understanding osn-based facial disclosure against face authentication systems," *Proceedings of the 9th ACM symposium on Information, computer and communications security*, 2014.

[148] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015.

[149] R. Feng and B. Prabhakaran, "Facilitating fashion camouflage art," in *MM '13*, 2013.

[150] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, pp. 71–86, 1991.

[151] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, pp. 711–720, 1997.

[152] Z. Yan, Y. Guo, and C. Zhang, "Deep defense: Training dnns with improved adversarial robustness," in *NeurIPS*, 2018.

[153] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, OpenReview.net, 2017.

[154] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[155] S. I. Serengil and A. Ozpinar, "Lightface: A hybrid deep face recognition framework," in *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pp. 23–27, IEEE, 2020.

[156] M. Project, "Landscapes dataset," 2020.

UASLP
Universidad Autónoma
de San Luis Potosí

FACULTAD DE
INGENIERÍA

Thesis presented to the Engineering Faculty of the Universidad Autónoma de San Luis Potosí to obtain the degree of Doctor in Computer Science.

Gerardo Ibarra Vázquez
gerardo.ibarra@alumnos.uaslp.edu.mx.

January, 2021