Universidad Autónoma de Querétaro

Facultad de Ingeniería

Maestría en Ciencias en Inteligencia Artificial

Análisis de redes neuronales profundas con capas monogénicas para robustecer su seguridad frente ataques adversarios para la clasificación de imágenes

TESIS

Que como parte de los requisitos para obtener el grado de
Maestro en Ciencias en Inteligencia Artificial

Presenta:

Ing. Oscar Garcia Porras

Dirigido por:

M. en C. Sebastián Salazar Colores

Centro Universitario, Querétaro, QRO, México.
Noviembre 2023

Universidad Autónoma de Querétaro

Facultad de Ingeniería

Maestría en Cinecias en Inteligencia Artificial

Análisis de redes neuronales profundas con capas monogénicas para robustecer su seguridad frente ataques adversarios para la clasificación de imágenes

TESIS

Que como parte de los requisitos para obtener el grado de
Maestro en Ciencias en Inteligencia Artificial

Presenta:

Ing. Oscar Garcia Porras

Dirigido por:

M. en C. Sebastián Salazar Colores

SINODALES

M. en C. Sebastián Salazar Colores
Presidente

Dr. E. Ulises Moya Sánchez
Secretario

Dr. Saúl Tovar Arriaga
Vocal

Dr. Marco Antonio Aceves Fernández
Suplente

Dr. Juan Manuel Ramos Arreguín
Suplente

Centro Universitario, Querétaro, QRO, México.
Noviembre 2023

Esta tesis la dedico a mis seres queridos, en agradecimiento por su apoyo y soporte durante este breve, pero significativo trayecto recorrido.

Agradecimientos

En agradecimiento a mi familia y seres queridos que han sido el pilar de mi crecimiento y mis triunfos, y que también permanecen ahí en mis fracasos y dolencias. Les debo todo, y estaré siempre agradecido.

Así mismo, añado un inmensurable agradecimiento al Dr. Sebastián Salazar Colores, al Dr. E. Ulises Moya Sánchez y al Dr. Saúl Tovar Arriaga por el apoyo, los conocimientos compartidos, dedicación y empuje para la realización de este trabajo de investigación. No existe mejor persona que la que gusta de lo que hace y trabaja.

Esta investigación académica fue soportada por el instituto gubernamental mexicano Consejo Nacional de Humanidades Ciencias y Tecnologías (CONAHCyT), en la facultad de posgrado de ingeniería de la Universidad Autónoma de Querétaro (UAQ), específicamente con el soporte financiero para el autor (CVU:1176674) en los estudios para la Maestría en Ciencias en Inteligencia Artificial.

Abstract

The purpose of adversarial attacks is to modify the data supplied to Artificial Intelligence (AI) models triggering failures in their designed purpose. Convolutional Neuronal Networks (ConvNet) are one of the AI models developed for computer vision tasks (e.g. detection, segmentation, and classification). State-of-art ConvNets have improved their performance increasing their reliability for real-life tasks. However, multiple studies have exposed ConvNet's vulnerability to adversarial attacks causing severe failures in critical tasks (e.g. medical purposes, self-driving, facial recognition, and finance purposes). Hence, complementary defensive methods are required for ConvNets to increase their robustness.

This research continues the implementation of the novelty bio-inspired monogenic layer [1, 2] now as a defensive method to confront adversarial attacks, whose intended purpose for this research is to minimize Projected Gradient Descent (PGD) adversarial attack disruptions working with a L_{inf} distance metric. The defensive method is based on the use of a deterministic monogenic layer at the beginning of a classification ResNet50 pipeline, and the CIFAR10 data set. The results show better performance for the ConvNet Resnet50 with the monogenic layer than the regular ConvNet ResNet50, and a second model using robust training as a defensive method. The obtained results may be interpreted as increasing resistance to performance disruptions caused by the adversarial attack, and reliable usage for AI models. Structural Similarity Index Measure (SSIM) and Peak Signal-to-Noise Ratio (PSNR) metrics were applied in the monogenic layer activations intending to get a quantitative interpretation of the improvement.

The obtained results from this research show the potential usage of the monogenic layer to confront adversarial attacks and encourage the continuity of research to expand knowledge of the monogenic signal for AI purposes.

Keywords: Artificial intelligence · Adversarial attacks · Deep neural networks · Image classification · Monogenic signal

Resumen

El propósito de los ataques adversarios es modificar los datos suministrados a modelos de Inteligencia Artificial (IA) para causar fallas en la tarea para la que fueron diseñados. Las redes neuronales convolucionales (ConvNet) son un los modelos de IA desarrollados para tareas de visión por computadora (e.g. detección, segmentación, y clasificación). ConvNets del estado del arte han mejorado su desempeño, permitiendo confiar en su uso para tareas de la vida real. Sin embargo, se ha demostrado que las ConvNet son vulnerables ante los ataques adversarios, y esto puede ocasionar fallas graves en tareas que requieren alta seguridad en su ejecución (e.g. fines médicos, conducción autónoma, reconocimiento facial, y finanzas). Por lo cual, es necesario complementar las ConvNet con métodos de defensa que las hagan más robustas.

Esta tesis continua la implementación de la novedosa capa monogénica bioinspirada [1, 2], ahora como método de defensa contra ataques adversarios, cuyo propósito en esta investigación es contrarrestar las afectaciones causadas por un ataque adversario Descenso de Gradiente Proyectado (PGD, por sus siglas en inglés) con métrica de distancia $Linf$. El método de defensa esta basado en el uso de una capa monogénica determinista al frente de una ConvNet ResNet50 utilizada para la clasificación de imágenes con el conjunto de datos CIFAR10. Los resultados muestran mejor desempeño para la ConvNet ResNet50 con capa monogénica que una ConvNet ResNet50 común, y que un segundo modelo con entrenamiento robusto como método de defensa. Los resultados obtenidos pueden ser interpretados como un incremento en la resistencia al ataque adversario, y fiabilidad en su uso para modelos de IA. Las métricas Medición del Índice de Similitud Estructural (SSIM, por sus siglas en inglés) y Proporción Máxima Señal-Ruido (PSNR, por sus siglas en inglés) fueron aplicadas en las activaciones de la capa monogénica con el fin de obtener una explicación cuantitativa de la mejora.

Los resultados obtenidos de esta investigación exponen el uso potencial de la capa monogénica para confrontar ataques adversarios, y motiva la continuidad de expansión del conocimiento de la señal monogénica utilizada para propositos de IA.

Palabras claves: Inteligencia artificial · Ataques adversarios · Redes neuronales profundas · Clasificación de imágenes · Señal monogénica

Índice general

Índice de Figuras

Índice de Cuadros

Abreviaturas y Siglas

AI - Artificial Intelligence
IA - Inteligencia Artificial
ConvNet - Convolutional Neural Network
PGD - Projected Gradient Descent
SSIM - Structural Similarity Index Measure
PSNR - Peak Signal-to-Noise Ratio
DNN - Deep Neural Network
SVM - SUpport Vector Machine
ILSNRC - ImageNet Large Scale Visual Recognition Challenge
QAFW - Quaternionic Atomic Function Wavelet
QPHCNN - Quaternionic Phase Convolutional Neuronal Network
MNIST- Modified National Institute of Standards and Technology
CIFAR - Canadian Institute For Advanced Research
ANN - Artificial Neuronal Network
ReLU - Rectified Linear Unit
KNN - K-Nearest Neighbors
GAN - Generative Adversarial Network
JPEG - Joint Photographic Experts Group
RAM - Random Access Memory
GPU - Graphics Processing Unit
CPU - Central Processing Unit
RGB - Red, Green and Blue
ReseNet - Residual Network

CAPÍTULO 1

Introducción

El propósito de la visión por computadora es intentar semejar la habilidad de ver y entender la información visual como lo hacen los seres humanos, este tipo de tareas requieren el uso de métodos y técnicas para poder analizar, entender, y extraer información relevante de los datos visualizados. Las redes neuronales profundas (DNN, por sus siglas en inglés) y específicamente las redes neuronales convolucionales (ConvNets) son ampliamente utilizadas en tareas de visión por computadora debido a la capacidad que tienen para poder extraer características superiores de la información, procesarla, y completar las tareas requeridas [13].

El estado del arte de las redes neuronales está alcanzando un desempeño sin precedentes para tareas de visión por computadora, el cual ha sido expuesto por múltiples estudios e investigaciones, un gran numero se enfocan en tareas relacionadas a la clasificación de imágenes [14, 15, 16]. La clasificación de imágenes puede ser utilizada en tareas cotidianas de la vida real, para lo cual se requiere disponer de bases de datos, es por ello que el trabajo de múltiples investigadores se ha enfocado en construir conjuntos de datos de imágenes para poder llevar acabo el entrenamiento de las redes neuronales. Algunos conjuntos de datos son relacionados a números, animales, plagas, tópicos afines a diversos temas, y algunos son desarrollados con fines críticos, como es el área médica [17, 18, 19, 20, 21].

El desempeño del estado del arte de las ConvNets puede ser afectado mediante el suministro de imágenes procesadas por ataques adversarios (ejemplos adversarios), los cuales llevan a la ConvNet a generar clasificaciones erróneas, y posterior falla en el propósito principal para la cual fue desarrollada [22]. El impacto que ocasionan los ataques adversarios al desempeño de las ConvNets es un problema que debe ser afrontado y minimizado.

En la actualidad múltiples investigaciones han explorado alternativas para mejorar el desempeño de las ConvNets al confrontar las afectaciones ocasionadas por los ataques adversarios, sin embargo, no se ha llegado a una solución general efectiva para este problema, lo cual deja un hueco que permite ahondar más en el área de investigación relacionada a minimizar las afectaciones causadas por ataques adversarios en el desempeño de las ConvNets [12, 23, 24, 25, 26, 27, 28, 29, 30].

La presente tesis expone el uso de una capa monogénica diseñada e inspirada en el concep-

to de las propiedades biológicas de la corteza primaría V1 del sistema visual de los mamíferos [31], la cual es usada como acercamiento en el diseño de un método de defensa que contrarrestar las afectaciones ocasionadas a una ConvNet (diseñada para tareas de clasificación de imágenes) por ataques adversarios.

La experimentación es basada en la generación de ejemplos adversarios mediante las librerías Foolbox [32], Robustness [33] y el conjunto de datos CIFAR-10 [20]. La clasificación de las imágenes se realizo mediante el uso de la ConvNet ResNet-50 [34]. Los resultados obtenidos de las evaluaciones realizadas exponen una mejora en el desempeño y robustez de la ConvNet con la capa monogénica.

1.1. Definición del Problema

Múltiples trabajos de investigación se han enfocado en el desempeño de las ConvNet para tareas de clasificación de imágenes, sin embargo, se ha demostrado que las ConvNet son vulnerables a los ataques adversarios. Los ataques adversarios afectan su desempeño y ocasionan graves fallas en su funcionamiento, por lo tanto, es un problema que debe ser afrontado y minimizado. Investigaciones relacionadas a métodos de defensa para confrontar los ataques adversarios han obtenido resultados favorables, no obstante, no se ha llegado a una solución general efectiva para este problema.

La información disponible e investigaciones generadas sobre este tema no es suficiente para poder satisfacer las necesidades en cuanto a la seguridad de redes neuronales profundas, esta restricción detiene gradualmente los avances sobre el uso de redes neuronales profundas para el uso de aplicaciones que requieren de fiabilidad en su desempeño, y de un estricto uso de seguridad.

Por lo tanto, se requiere expandir el conocimiento de métodos de defensa que minimicen los daños ocasionados por ataques adversarios mediante el incremento de investigación sobre el tema.

1.2. Justificación

El uso de algoritmos de redes neuronales profundas para el procesamiento de datos y ejecución de tareas relacionadas con visión por computadora se ha incrementado en la actualidad. Algunas aplicaciones de las redes neuronales profundas pueden incluirse en procesos críticos de seguridad, como: conducción autónoma, detección de *malware*, navegación de drones, uso de robótica, procesos médicos inteligentes, procesos de seguridad mediante el reconocimiento facial, entre otros. Sin embargo, se ha comprobado que los modelos de redes neuronales actuales son vulnerables ante la interacción con agentes externos, como lo son los ataques adversarios, los cuales pueden llegar a ser muy complejos y forzar a un modelo entrenado a producir errores específicos en la salida, desencadenando problemas de seguridad para las redes neuronales y su funcionamiento [35].

Actualmente el principio de generación de ataques adversarios no ha sido explicado con

una base científica, por lo cual, los investigadores se han dado a la tarea de proveer una base teórica y practica para incrementar la seguridad de los modelos basados en redes neuronales profundas, mediante la exploración de algoritmos para generación de ataques adversarios [12].

Es necesario expandir la investigación en el tema de ataques adversarios y métodos de defensa contra ataques adversarios para diseñar algoritmos más robustos, y poder garantizar la seguridad de las redes neuronales profundas.

El presente proyecto de investigación esta dirigido al análisis y exploración de modelos de redes neuronales profundas que podrían incluir el uso de capas monogénicas. El uso de las capas monogénicas como método de defensa podrían hacer a los modelos más robustos, podrían incrementar su capacidad para afrontar de manera exitosa ataques adversarios, y su desempeño no se vería afectado. De igual forma, este proyecto de investigación podría servir como una base para futuras investigaciones en el diseño de redes neuronales profundas más robustas y seguras.

1.3. Hipótesis

El uso de capas monogénicas en una red neuronal de aprendizaje profundo mejoraría las capacidades de seguridad del modelo haciéndolo más robusto al reducir la afectación causada por ataques adversarios, y proporcionaría un desempeño equiparable o mejor en comparación con modelos neuronales del estado del arte utilizados para la clasificación de imágenes.

1.4. Objetivos

1.4.1. Objetivo general

Experimentar y analizar el uso de las capas monogénicas en redes neuronales de aprendizaje profundo, con el fin de validar su uso como método de mejora para robustecer su seguridad al reducir la afectación causada por ataques adversarios, y mejorar o equipara su desempeño en la tarea de clasificación de imágenes en comparación con modelos neuronales del estado del arte.

1.4.2. Objetivos específicos

Los objetivos específicos para la ejecución del presente proyecto de investigación, son los siguientes:

- Obtener y adecuar: Base de datos, modelos de ataques adversarios y los modelos de defensa contra ataques adversarios que serán utilizados para la experimentación.
- Definir la metodología de pruebas para la obtención de valores cuantitativos requeridos.

- Desarrollar los modelos neuronales orientados a la tarea de clasificación de imágenes basados en la capa monogénica.
- Ejecutar entrenamiento, validación y pruebas de la red de aprendizaje profundo.
- Analizar y validar los resultados obtenidos para determinar las conclusiones.

1.5. Alcance del proyecto

El alcance del proyecto de investigación fue realizado con base en los recursos disponibles para la ejecución del mismo.

- Exploración, análisis y ejecución del uso de capas monogénicas para redes neuronales profundas como método de defensa contra ataques adversarios para la clasificación de imágenes.
- Diseño y ejecución de la metodología de pruebas sobre el uso de las capa monogénicas para redes neuronales profundas como método de defensa contra taques adversarios para la clasificación de imágenes.
- Análisis y validación de resultados concluyentes.

CAPÍTULO 2

Antecedentes

2.1. Antecedentes

El inicio de la inteligencia artificial se remonta a la década de 1950, su continuo desarrollo se atribuye a las aportaciones de numerables personajes, situaciones históricas, y descubrimientos. En 1950 Alan Turing propuso la prueba total de Turing que engloba las disciplinas: procesamiento de lenguaje natural, representación del conocimiento, razonamiento automatizado, aprendizaje automático, visión por computadora, y robótica. Disciplinas con las cuales la mayoría de los actuales modelos de IA están construidos [36].

Frank Rosenblatt *et al.* [37] en 1957 desarrollo el modelo perceptrón de Rossenblatt. El modelo de Ronssenblatt se convertiría en el prototipo de las redes neuronales artificiales modernas [38].

Rumelhart *et al.* [39] en el año 1986 crean el algoritmo *back-propagation*, formalmente introduce un método de aprendizaje para entrenar redes neuronales. Esta aportación impulsaría más investigaciones en el área de inteligencia artificial.

LeCunn [40] propone en 1989 una red neuronal convolucional, su investigación consta de la clasificación de 480 imágenes de números escritos a mano, obteniendo una generalización de 98.4 %. Su aportación formaría parte esencial para el área de visión por computadora.

Cortes y Vapnik [41] en 1995 crean una nueva versión del algoritmo SVM (*Support Vector Machine*), al cual llaman *support-vector networks*, el cual tuvo un impacto significativo en teoría, y aplicación del aprendizaje automático.

Diversas investigaciones son llevadas a cabo para desarrollar métodos alternativos que permitan procesar la información, y expandir el conocimiento.

Felsberg y Sommer [42] en el año 2001 introducen una metodología de análisis de imágenes llamada *señal monogénica* haciendo uso del álgebra de cuaternión. La señal monogénica es la generalización de la señal analítica de una dimensión (1D) a dos dimensiones (2D). Esta investigación permitiría obtener características complejas de las señales 2D, para su posterior y procesamiento.

Los avances tecnológicos facilitaron el procesamiento de mayor cantidad de datos, lo cual impulso la expansión en la investigación del aprendizaje automático y visión por computadora [38].

Krizhevsky *et al.* [43] en el año 2010 diseñaron un modelo de red convolucional (AlexNet) que en el año 2012 obtuvo el primer lugar en la competición ILSVRC, la cual consiste en clasificar 1.2 millones de imágenes con distintos tipos de resolución para 1,000 tipos de clases diferentes.

Grandes avances se generan y los modelos comienzan a ser más complejos, sin embargo, no son lo suficientemente robustos ni seguros, lo cual los hace vulnerables ante ataques de agentes externos, como lo son los ataques adversarios. [12].

La aparición de algoritmos de ataques adversarios comenzó en el año 2013 con Szegedy *et al.* [44], comprueban que pueden causar falla en la clasificación de una red neuronal mediante perturbaciones a los datos de entrada. La vulnerabilidad de las redes neuronales expuesta por Szegedy *et al.* [44] da un empuje a la investigación de los ataques adversarios, y su afectación en el desempeño de las redes neuronales.

Hinton *et al.* [45] en el año 2015 fue el primero en proponer un método de defensa contra ataques adversarios (*distillation*), este método de defensa reduce la capacidad del atacante para obtener información del modelo debido a la base de su funcionamiento relacionado a la transferencia de conocimiento entre una red neuronal compleja y una de menor complejidad.

Las investigaciones continúan exponiendo la vulnerabilidad de las redes neuronales, y de los métodos de defensa desarrollados.

Carlini y Wagner [46] en el año 2016 proponen el método de ataque adversario Descenso de Gradiente Proyectado (PGD, por sus siglas en inglés). Este ataque adversario hace uso de los parámetros del modelo, la métrica de distancia (L_0, L_2 y L_∞), y del descenso de gradiente para maximizar la pérdida y generar los ejemplos adversarios. Hacen uso de los conjuntos de datos MNIST, CIFAR-10 e ImageNet, causando mayor afectación que los métodos de ataque del estado del arte.

2.2. Antecedentes de la Capa Monogénica

Moya y Bayro [3] en el año 2010 introdujeron la teoría y experimentación de la *Quaternionic Atomic Function Wavelet (QAFW)* para el procesamiento de imágenes, la experimentación da como resultado que la información extraída mediante el uso de la QAFW es independiente al cambio de iluminación de las imágenes (Figura 2.1).

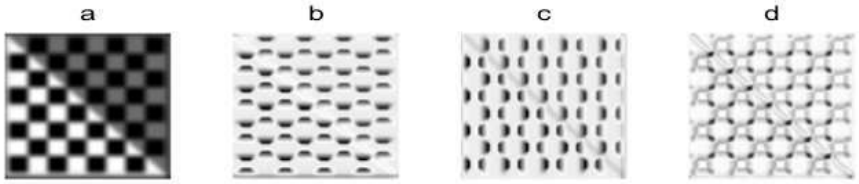

Figura 2.1: a) Amplitud de la imagen filtrada, y sus tres fases (ϕ, θ, φ)[3].

Moya y Bayro [4] en el año 2014 proponen un método para extraer características de la simetría de los objetos contenidos en imágenes, las cuales podrían ser usadas para la clasificación de objetos, esto mediante el uso del concepto de fase local del álgebra de cuaternión y el uso de la señal monogénica (Figura 2.2).

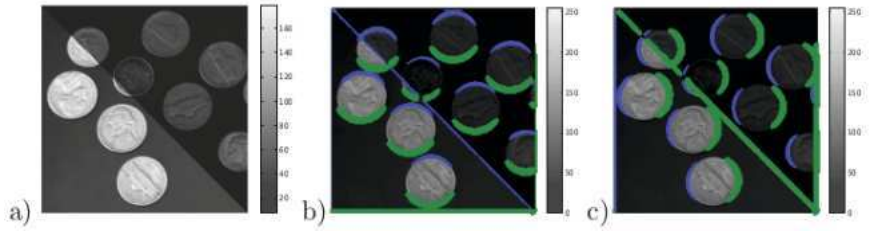

Figura 2.2: a) Imagen original y los bordes detectados por las fases: b) ϕ y c) θ [4].

Moya *et al.* [5] en el año 2018 presentan el primer acercamiento para crear una red neuronal convolucional usando la fase local de cuaternión, la denominan *Quaternion Phase Convolutional Neural Network (QPHCNN)*. Ejecutan la experimentación modificando la iluminación y rotación de las imágenes de la base de datos *MNIST*, y obtienen una exactitud y valor de pérdida similar al de una CNN común (Figura 2.3).

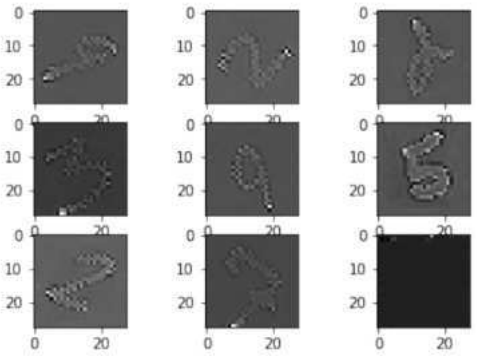

Figura 2.3: Ejemplo de imágenes de MNIST rotadas y con modificación en la iluminación [5].

Moya *et al.* [6] en el año 2018 presentan la red neuronal convolucional *Monogenic Convolutional Neural Network (MCNN)*, hacen uso de la base de datos *dogs and cats*, y los resultados exponen que la red MCNN reduce el sobre-ajuste ante la rotación de las imágenes, por lo cual provee una invarianza rotacional local dentro de cierto rango (Figura 2.4).

Moya *et al.* [1] en el año 2019 implementan una capa frontal monogénica para ConvNets denominada *M6*, clasifican imágenes con las bases de datos *Dogs and Cats* y *CIFAR-10*, aplican diferentes grados de iluminación y entrenan diferentes modelos de ConvNets con y sin la capa M6. Los resultados confirman que las ConvNets con la capa M6 tienen substancialmente mayor exactitud que las ConvNets comunes ante el cambio de iluminación en las imágenes (Figura 2.5).

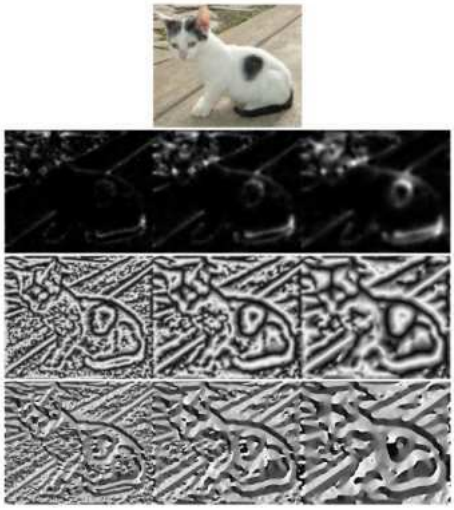

Figura 2.4: Ejemplo de la imagen original (superior), su amplitud (izquierda), fase (centro) y orientación (derecha) obtenidas en tres diferentes escalas [6].

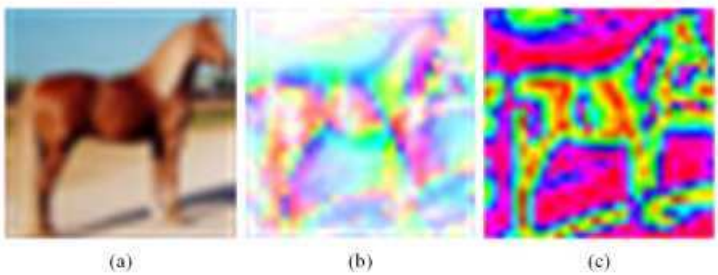

Figura 2.5: a)Imagen RGB original, b)RGB_θ y c)RGB_ϕ [1].

2.3. Estado del Arte

En la actualidad los ataques adversarios pueden ser clasificados de acuerdo a sus características y efectos causados. Las dos categorías principales son: ataque de caja negra y ataque de caja blanca [12].

Mao *et al.* [47] en el año 2020 presentan el modelo GAP++, el cual realiza ataques específicos y direccionados. Este ataque puede causar errores en la clasificación, y desviar la predicción a la clase que sea requerida por el atacante.

Duan *et al.* [48] en el año 2022 presentan un método de ataque de caja negra, que nombran *MsGN*, su funcionamiento consiste en generar ejemplos adversarios antes de ser suministrados a la ConvNet, realizan la inferencia con el modelo objetivo, y al final analizan los resultados para tratar de obtener información del modelo usado para clasificación. La información del modelo posteriormente puede dar base a un ataque de caja blanca.

Así como se han dado a la tarea de clasificar los ataques adversarios, de la misma forma, en la actualidad los métodos de defensa contra ataques adversarios se pueden clasificar en:

optimización de modelo, optimización de datos y adición de redes [12].

Wu *et al.* [49] en el año 2019 presentan el método de defensa de tipo optimización de datos. Su método de defensa consiste en realizar el entrenamiento mediante el uso de imágenes limpias y ejemplos adversarios para intentar que la ConvNet aprenda a distinguir entre ambos datos. Este tipo de defensa es conocido como entrenamiento robusto.

Engstrom *et. al* [33] en el año 2019 crean la librería *robustness* para python. La librería es utilizada para generar ataques adversarios de tipo Descenso de Gradiente Proyectado PGD, y hace uso de los ejemplos adversarios para un entrenamiento robusto como método de defensa. Hacen uso de la métrica de exactitud para medir el desempeño del método de defensa, y usan los conjuntos de datos ImageNet y CIFAR-10.

Dapello *et al.* [50] en el año 2020 implementan una ConvNet con la capa V1OneBlock frontal dentro de la red neuronal. La capa es inspirada en las propiedades conocidas de la corteza primaria V1 del sistema visual de los primates. Usan un modelo lineal-no lineal de poisson, un filtro de Gabor, y un generador V1 neuronal estocástico. Hacen uso de la métrica de exactitud, y de ImageNet como el conjunto de datos. Obtienen resultados superiores a 18 % más que las ConvNets base, y 3 % más que los métodos del estado del arte.

Ma *et al.* [51] en el año 2021 presenta un método de defensa de tipo optimización de modelo, el cual reduce la afectación de ataques adversarios mediante un proceso interno de regularización de imágenes. Hacen uso de la métrica de exactitud y el conjunto de datos CIFAR-10. La experimentación se realiza con los modelos ResNet-50 y Wide ResNet.

Abusnaina *et al.* [52] en el año 2021 desarrollan el método de defensa de tipo adición de red, su método se basa en usar un discriminador para detectar imágenes perturbadas mediante el uso de KNN, y así evitar su procesamiento en el modelo usado para clasificar.

Moya *et al.* [2] en el año 2021 presentan la nueva capa frontal CNN entrenable M6, que genera una representación geométrica 3D del valor de cada pixel mediante la codificación de la señal monogénica de cuaternión en el dominio de Fourier. Se hizo uso de la base de datos *MNIST*, *fashion MNIST*, *CIFAR-10*, y *dogs and cats*. Se realizaron pruebas utilizando la Medición de Índice de Similitud Estructural SSIM para medir de forma cuantitativa las modificaciones generadas antes y después de la M6, los resultados demostraron que existe una baja discrepancia entre las imágenes, así como también, se comprobó que la M6 mejora la exactitud en la clasificación de imágenes afectadas por altos cambios de contraste.

Resumen del estado del arte para métodos de defensa contra ataques adversarios (Cuadro 2.1).

Cuadro 2.1: Resumen del estado del arte para métodos de defensa contra ataques adversarios.

Autor	Método de defensa / ConvNet	Ataque adversario	Parámetros p/ataque	Conjunto de datos	Resultados
Wu *et al.* [49]	Optimización de datos — Entrenamiento robusto —— VGGFace LISA CNN	PGD L_∞ [22]	ϵ = [4,8,16,32] pasos = 50 tam. de paso = $\epsilon/4$	VGGFace LISA	∼68 % acc. (entrenamiento) -ϵ = 8- 48 % acc. (prueba)
Engstrom *et al.* [33]	Optimización de datos — Entrenamiento robusto —— ResNet-50	PGD L_∞ [33]	ϵ = [0.0313,0.0627] pasos = 20 tam. de paso = [0.00391, 0.00784]	CIFAR-10	87.03 % acc. (entrenamiento) -ϵ = 0.0313- 53.49 % acc. (prueba) -ϵ = 0.0627- 18.13 % acc. (prueba)
Dapello *et al.* [50]	Optimización de datos — Entrenamiento robusto —— ResNet-50	PGD L_∞ [53]	ϵ = [1/1020,1/255] pasos = 64 tam. de paso = $\epsilon/32$	ImageNet	71.7 % acc. (entrenamiento) -ϵ = 1/1020- ∼67 % acc. (prueba) -ϵ = 1/255- ∼30 % acc. (prueba)
Ma *et al.* [51]	Optimización de modelo — Regularización de imágenes —— ResNet10	PGD L_∞ [22]	ϵ = 8/255 pasos = 20 tam. de paso = 2/255	CIFAR-10	87.95 % acc. (entrenamiento) -ϵ = 8/255- ∼56.06 % acc. (prueba)
Abusnaina *et al.* [52]	Adición de red — Detección de ejemplos adversarios —— ResNet110	PGD L_∞ [22]	ϵ = 0.02 pasos = 50 tam. de paso = 0.002	ImageNet CIFAR-10 STL-10	91.39 % AUC (detección)

2.4. Fundamentación Teórica

2.4.1. Aprendizaje automático

El aprendizaje automático surge de la siguiente pregunta: ¿puede una computadora ir más allá de lo que se sabe sobre cómo ordenarle que lo haga y que aprenda por sí misma como realizar una tarea especifica? Con el uso del aprendizaje automático los humanos suministran datos con las respuestas esperadas, y como resultado salen las reglas. Estas reglas pueden ser aplicadas a nuevos datos de entrada para producir respuestas originales. Un sistema de aprendizaje automático es entrenado en lugar de ser explícitamente programado (Figura 2.6) [7].

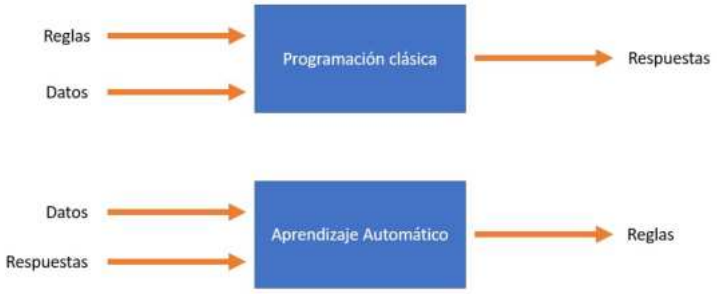

Figura 2.6: Aprendizaje automático [7].

El aprendizaje automático convencional comúnmente hace uso de métodos de optimización que son usados para el aprendizaje de modelos parametrizados (e.g. regresión lineal, *Support Vector Machines* (SVM), regresión logística, reducción de dimensionalidad, y factorización de matrices).[8].

Aprendizaje profundo

Desde la década de 1940 el aprendizaje automático a mejorado su desempeño constantemente a partir de las Redes Neuronales Artificiales (ANN, por sus siglas en inglés). Las ANNs son redes interconectadas que asemejan el funcionamiento biológico de las neuronas del cerebro. Las entradas de datos suministradas a las neuronas son atenuadas o amplificadas por los valores de entrada que son provenientes del uso de otras neuronas vecinas. El principio de funcionamiento de cada neurona artificial se basa en transmitir las señales de entrada a la neurona, sumarlas, aplicar una función de activación, y una toma de decisión basada en los resultados obtenidos. En la actualidad existen diferentes tipos de ANNs, y su variabilidad reside principalmente en el tipo de activación que utilizan [10].

Las redes neuronales artificiales fueron renombradas como redes neuronales de aprendizaje profundo en el año 2006 por Geoffrey Hinton y colaboradores cuando introdujeron la idea del entrenamiento supervisado y las redes profundas. El aprendizaje profundo está establecido como un subconjunto del área de aprendizaje automático (Figura 2.7) [10].

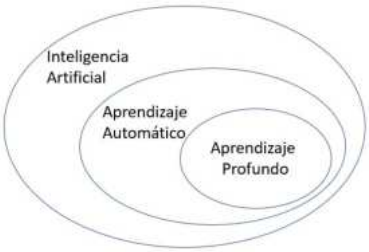

Figura 2.7: Inteligencia artificial, aprendizaje automático, y aprendizaje profundo [7].

Los modelos de redes neuronales requieren de entrenamiento basado en las características de los datos de entrada, posteriormente el modelo predice un valor de salida cuando se suministra cualquier otro dato. El proceso de entrenamiento involucra el uso de [7]:

- Capas neuronales, la combinación de capas genera la red neuronal densa.
- Datos de entrada y etiquetas correspondientes.
- Función de pérdida, la cual define el valor de retroalimentación para el aprendizaje.
- Optimizador, el cual determina como procederá el entrenamiento.

En la Figura 2.8 se pueden observar ejemplos de la estructura de una neurona de tipo perceptrón, un ejemplo de la estructura de una red neuronal profunda, y un diagrama de la interacción de los elementos que conforman una red neuronal y su funcionamiento.

El diseño de las redes neuronales profundas puede cambiar en su estructura, función de pérdida, optimizador, función de activación, entre otras características. El diseño y complejidad de los modelos depende del propósito final para los cuales son construidos (e.g. tareas de visión por computadora, lenguaje natural, entre otras).

2.4.2. Visión por computadora

El propósito de la visión por computadora es extraer información relevante de los datos visualizados para poder analizarlos y entenderlos [54].

Las redes neuronales profundas (DNN, por sus siglas en inglés) y en especial las redes convolucionales (ConvNets) son modelos diseñados que pertenecen al aprendizaje profundo, y una de sus aplicaciones es la extracción de características relevantes de datos visuales. Este tipo de red puede ser usada en conjunto con redes neuronales profundas para tareas de visión por computadora, como los son: clasificación, detección, y segmentación de objetos [8].

Una de las diferencias principales entre una DNN y una ConvNet recae en el hecho de que la red neuronal profunda aprende patrones globales, y la ConvNet aprende patrones locales de los datos de entrada. Asignando a las ConvNet dos propiedades [7]:

- Los patrones que aprenden son invariantes de traslación. Lo que significa que el patrón que aprende en determinada ubicación de la imagen lo puede reconocer en cualquier parte.

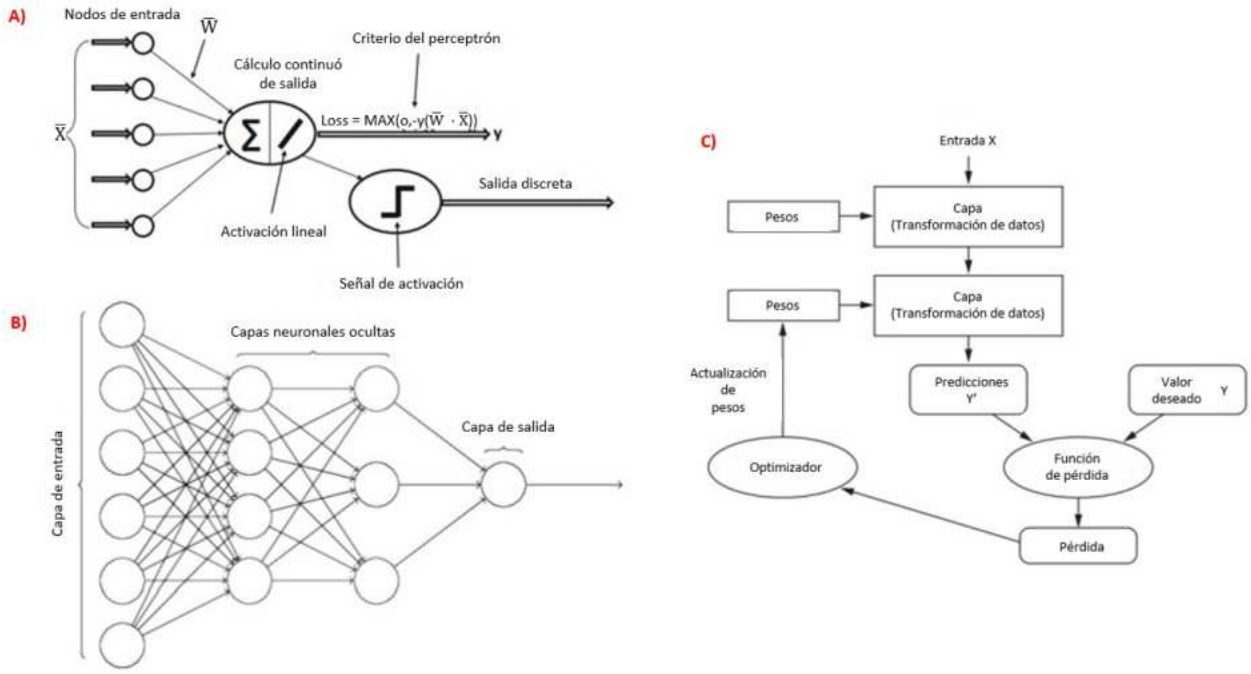

Figura 2.8: A) La estructura de una neurona de tipo perceptrón (continua o discreta) [8]. B) La estructura de una red neuronal [9]. C) Diagrama de funcionamiento de una red neuronal [7].

- Pueden aprender jerarquías espaciales de patrones. La primera capa aprenderá pequeños patrones locales (e.g. bordes), la segunda capa aprenderá patrones más grandes generados por los patrones de la primer capa, y así continuamente con el uso de más capas. Esto le permite a las ConvNets aprender de manera eficiente conceptos visualmente más complejos y abstractos (Figura 2.9).

Los componentes básicos para una ConvNet (Figura 2.10), son [7]:

- Capa de entrada, contiene la intensidad de los píxeles de la imagen.
- Capa convolucional, toma las imágenes procedentes de la capa anterior y realiza la convolución del número especificado de filtros para crear el mapa de objetos de salida.
- Función de activación, usualmente es ReLU, la cual añade no-linealidad a la red y al mismo tiempo evita la saturación de gradientes para entradas positivas a la red.
- Capa de reagrupación, reduce la dimensión de las imágenes contenidas en el mapa de objetos. La cantidad de filtros permanece sin modificación.
- Capa totalmente conectada, contiene la red neuronal densa que realiza la tarea para la cual fue diseñada la red neuronal convolucional.

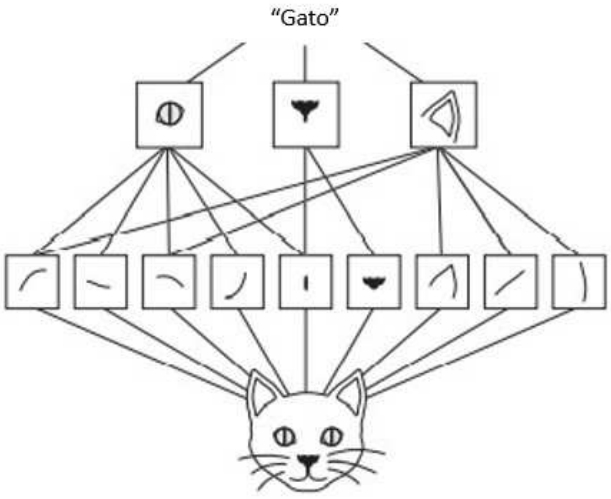

Figura 2.9: Jerarquía espacial de patrones [7].

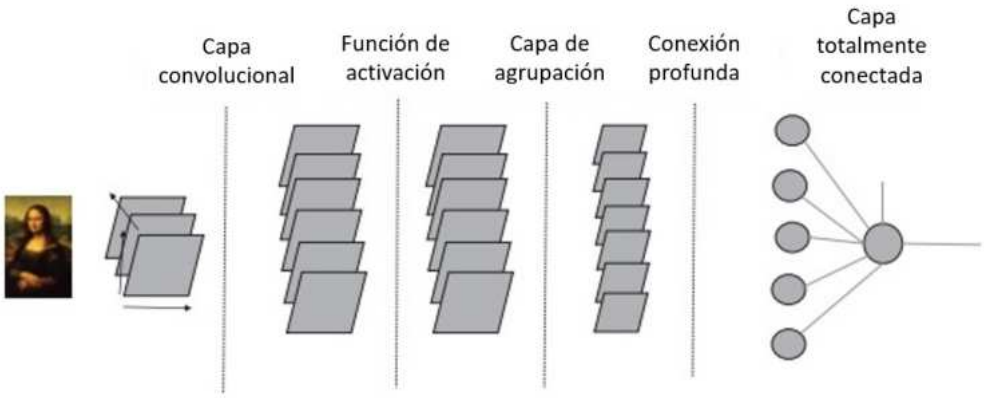

Figura 2.10: Ejemplo de la estructura básica de una ConvNet [10].

El uso de las ConvNets para tareas de visión por computadora ha facilitado el análisis y procesamiento de la información, sin embargo, la complejidad de algunos problemas requiere de métodos que puedan ayudar a aumentar el desempeño de los modelos de redes neuronales. El procesamiento de imágenes puede incrementar el desempeño de los modelos de redes neuronales aplicando técnicas o métodos a los datos de entrada (e.g. transformaciones, filtros, entre otros).

2.4.3. Señal Analítica y Señal Monogénica

La señal analítica es la representación compleja de una señal de una dimensión (1D), y es la base de la señal monogénica. La señal monogénica, como comúnmente se conoce, es la generalización más aceptada de dos dimensiones (2D) de la señal analítica, y es ampliamente usada en procesamiento de imágenes [42].

Señal Analítica

En el área de procesamiento y reconocimiento de imágenes, la señal analítica puede ser vista como la separación de identidades. En su representación polar se pueden extraer dos características de la señal compleja: la amplitud local (medición cuantitativa de la información) como el módulo, y la fase local (medición cualitativa de la información) como el argumento [42].

Las ecuaciones que definen a la señal analítica (f_A) de una función inicial (f) [55] son:

$$f_A = f - if_{Hi} \tag{2.1}$$

donde, f_{Hi} es la transformada de Hilbert:

$$f_{Hi}(x) = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{f(x)}{\tau - x}d\tau \tag{2.2}$$

Obteniendo la amplitud (A) y fase (ϕ) de f_A:

$$A = |f_A(x)| = \sqrt{[f(x)]^2 + [f_{Hi}(x)]^2} \tag{2.3}$$

$$\phi(x) = \arctan\left(\frac{f_H(x)}{f(x)}\right) \tag{2.4}$$

Estas dos características extraídas de la señal compleja son usadas como representación alternativa de la misma, y comúnmente usadas para procesamiento de señales. Sin embargo, en gran mayoría de los casos existen diferentes tipos de señales y en especial las que trabajan a diferentes escalas, por lo cual, el área de trabajo necesita enfocarse a un rango especifico de la señal analizada. Comúnmente para tratar de solucionar este problema se hace uso de un filtro que se coloca en el centro de las frecuencias que requieren del proceso de filtrado. Gabor, log-gabor, poisson, cauchy, y la derivada gaussiana son algunos filtros utilizados que pueden funcionar como solución [56].

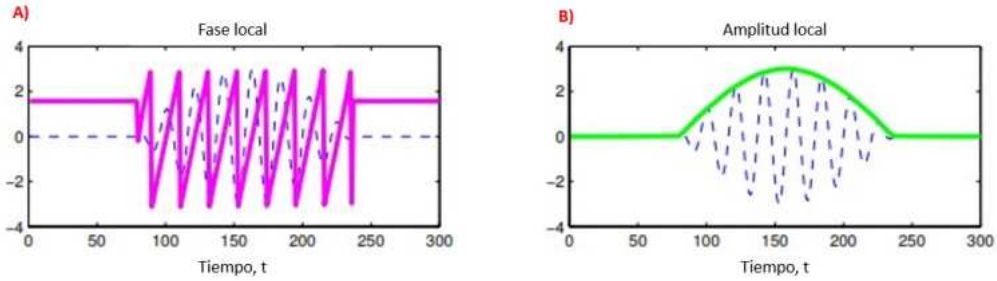

Figura 2.11: A) Fase local (linea rosa) y B) Amplitud local (linea verde) de una onda senoidal modulada por un pulso de coseno [11].

En la Figura 2.11 se puede observar un ejemplo de la fase local y amplitud local de una señal 1D.

Señal Monogénica

La señal analítica proporciona un único valor imaginario, y para el caso de una señal 2D se requieren dos valores imaginarios para representar cada dimensión. Para obtener estos dos valores imaginarios se requiere del uso del álgebra de cuaternión y la transformada de riesz (generalización de la transformada de Hilbert).

En su forma básica general, la señal monogénica puede ser expresada como [42]:

$$f_M = f + if_{R1} + jf_{R2} \tag{2.5}$$

donde, f_R son las transformadas de riesz. En este trabajo se hace la aproximación a la solución mediante un filtro de cuadratura, $f\,' = g * f$ donde * es el operador de convolución y $g = g(x, y)$ es la función de Gabor. Reescribiendo la función original:

$$J_M = \mathcal{F}^{-1}\left(J + J_1 + J_2\right) \tag{2.6}$$

donde, $J = \mathcal{F}(f)\mathcal{F}(g)$, $J_1 = \mathcal{F}(f)\mathcal{F}(g)f_{R_1}$ y $J_2 = \mathcal{F}(f)\mathcal{F}(g)f_{R_2}$.

La transformada de riesz está definida por:

$$f_{R_1}(u) = \frac{u_1}{||u||} \tag{2.7}$$

$$f_{R_2}(u) = \frac{u_2}{||u||} \tag{2.8}$$

donde, $||u|| = \sqrt{{u_1}^2 + {u_2}^2}$, u_1 y u_2 son los componentes de la frecuencia.

Regresando a las tres principales representaciones de cada pixel, la amplitud local de la señal monogénica puede ser definida como:

$$A_M = \sqrt{J^2 + {J_1}^2 + {J_2}^2} \tag{2.9}$$

Así, la fase local (J_ϕ) y la orientación local (J_θ) asociadas a J son definidas por:

$$J_\phi = \arctan 2\left(\frac{J}{|J_R|}\right) \tag{2.10}$$

$$J_\theta = \arctan\left(\frac{-J_2}{J_1}\right) \tag{2.11}$$

donde, $J_R = J_1 + J_2$.

El resultado final de aplicar las ecuaciones proporciona ambas características requeridas: la fase local (ϕ) y la orientación local (θ). Ambas representadas de forma visual en la Figura 2.12.

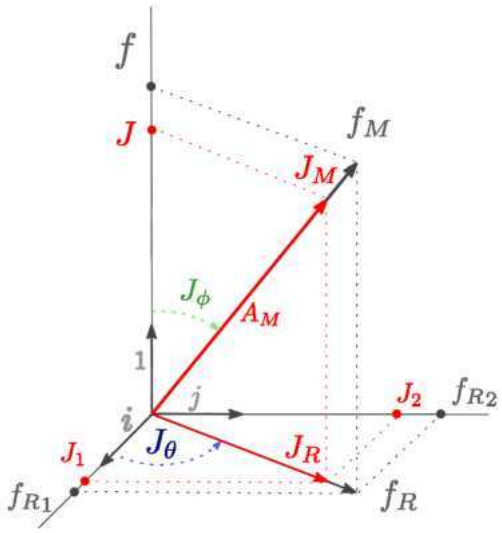

Figura 2.12: Representación geométrica de la señal monogénica obtenida de un pixel.

En la Figura 2.13 se puede observar la representación polar esférica de los componentes de la señal monogénica.

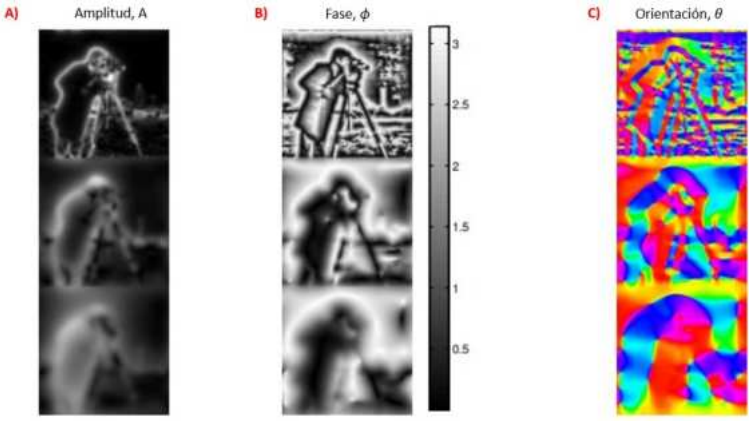

Figura 2.13: A) Amplitud local, B) Fase y C) Orientación, de imágenes obtenidas mediante la señal monogénica. La imagen de orientación codifica el ángulo como una rueda *hue* de colores [11].

2.4.4. Ataques Adversarios

En la actualidad las DNNs tienen un extenso uso y son implementadas frecuentemente en actividades diarias, donde las tareas de visión por computadora son necesarias. Sin embargo, las DDNs son vulnerables a los ataques adversarios, lo cual desencadena resultados incorrectos, y falla en el propósito de diseño del modelo [57]. Actualmente los ataques adversarios pueden ser identificados en tres distintos tipos: ataques de caja negra, ataques de caja

gris y ataques de caja blanca, los cuales son clasificados con base en el nivel de información que utilizan del modelo objetivo para ejecutar los ataques adversarios, y poder generar los ejemplos adversarios [12, 58].

- Ataque de caja negra: Tiene acceso a los datos de entrada para poder generar los ejemplos adversarios. Este tipo de ataque no tiene conocimiento de la estructura o parámetros del modelo objetivo (e.g. ruido gaussiano [59]).

- Ataque de caja gris: Tiene acceso a los datos de entada y tiene conocimiento de la estructura del modelo objetivo para poder generar los ejemplos adversarios (e.g. MI-FGSM [60], PGD [46], entre otros.).

- Ataque de caja blanca: Tiene acceso a los datos de entrada, tiene conocimiento de la estructura, y puede hacer uso de los parámetros del modelo objetivo para poder generar los ejemplos adversarios. En su método más avanzado puede generar ejemplos adversarios para desviar la salida de clasificación a una clase en especifico (e.g. PGD [46], Ataque Adversario Distribuido (DDA, por sus siglas en inglés) [61], entre otros.).

En la Figura 2.14 se puede observar la generación de un ejemplo adversario, y el proceso de un ataque adversario.

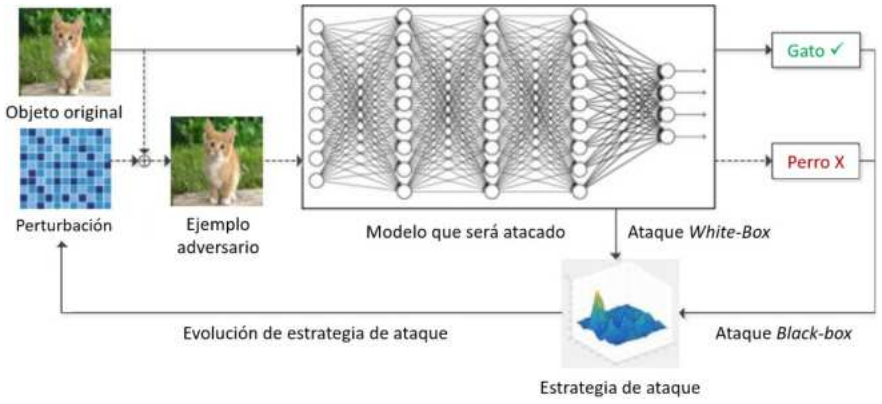

Figura 2.14: Ejemplo de la generación de un ejemplo adversario, y el proceso de un ataque adversario [12].

Debido al tiempo limitado para la realización de esta investigación y los múltiples tipos de ataques adversarios existentes, este trabajo de investigación se enfoca en el ataque adversario de tipo caja blanca PGD con métrica de distancia L_{inf}(Figura 2.15) [46].

Figura 2.15: Ejemplo adversario generado mediante el uso de la librería Foolbox y CIFAR-10, con la métrica de distancia $L_{inf}PGD$, epsilon=0.5, pasos=20, y tamaño de paso=0.0333.

2.4.5. Métodos de Defensa

Los ataques adversarios son un problema real que debe de ser considerado como un riesgo para la seguridad de los modelos de redes neuronales [23]. Por lo tanto, es necesario ahondar en temas de investigación para diseñar métodos de defensa que minimicen las afectaciones a las ConvNets. Múltiples métodos de defensa han sido diseñados en investigaciones para tratar de contrarrestar estas afectaciones. Los métodos de defensa pueden ser categorizados en: optimización del modelo, optimización de los datos, y adición de red [12].

- Optimización del modelo: Tiene diferentes tipos de técnicas, donde la robustez del modelo es implementado directamente en la estructura principal de la DNN. Su objetivo es complicar o evitar que el atacante obtenga la estructura, parámetros o información de los datos de entrada con los cuales fue entrenado el modelo. Algunos de los acercamientos para estos métodos utilizan la idea de transferencia de conocimiento entre redes, lo cual reduce la capacidad del ataque adversario de obtener información de la DNN, este método es conocido como defensa por destilación [24]. Dentro de los métodos que utilizan enmascaramiento de gradiente está el uso de codificadores y decodificadores para esconder los gradientes, y evitar que el atacante obtenga los parámetros originales [25], mientras que otras investigaciones incluyen el uso de una red *U-net* como método de reducción de ruido, cuya intención es tratar de reducir el ruido generado por el atacante, y restaurar la imagen a sus valores iniciales previo al ataque [26].

- Optimización de los datos: Tiene como objetivo confrontar el problema ocasionado por los ataques adversarios mediante el procesamiento de las imágenes previo al modelo de red neuronal o en algunos casos se trata de resolver el problema durante el entrenamiento. El entrenamiento adversario es un método orientado a entrenar la red neuronal con una mezcla de imágenes limpias y ejemplos adversarios, para así intentar que la red aprenda de ambos casos, y poder diferenciar las imágenes limpias de los ejemplos adversarios [27]. Algunas investigaciones preprocesan el ejemplo adversario mediante la compresión de características usando compresión de imágenes de 24-bit as 12-bit, posteriormente un segundo proceso ejecuta la reconstrucción de las imágenes

[28]. La reconstrucción de los datos de entrada mediante la transformación de imágenes es otro método utilizado, donde se usan transformaciones como: recorte de imágenes, escalamiento de imágenes, reducción de profundidad, compresión JPEG, minimización de variación total, y acolchonado de imágenes, donde la intención es obtener múltiples características de los datos de entrada que pueden ser efectivas para minimizar las afectaciones causadas por los ataques adversarios [29]. Algunos otros investigadores recurren al uso de métodos basados en redes de tipo GAN, la cuales trabajan removiendo el ruido mediante la generación de una imagen con ruido aleatorio que al final del proceso aproxima su distribución a la distribución de las imágenes limpias [30].

- Red adicional: Tiene como objetivo detectar ejemplos adversarios mediante el uso de una red adicional que procesa los datos de entrada, y proporciona resultados que ayudan a detectar si un dato es una imagen limpia o un ejemplo adversario. Uno de los trabajos de investigación utiliza el método KNN y una función de influencia para medir la correlación de un dato de entrada con el vecino más cercano de los datos de entrada limpios, y de esta manera detectar los ejemplos adversarios [62]. Un método ocupa una red detectora y otra red reformadora, donde la red detectora busca los datos de entrada que se encuentran lejos del área popular de la clase más cercana, y reconstruye los datos para cambiar la clasificación [63].

En la Figura 2.16 se puede observar el diagrama de clasificación para métodos de defensa contra ataques adversarios, propuesto por [12] .

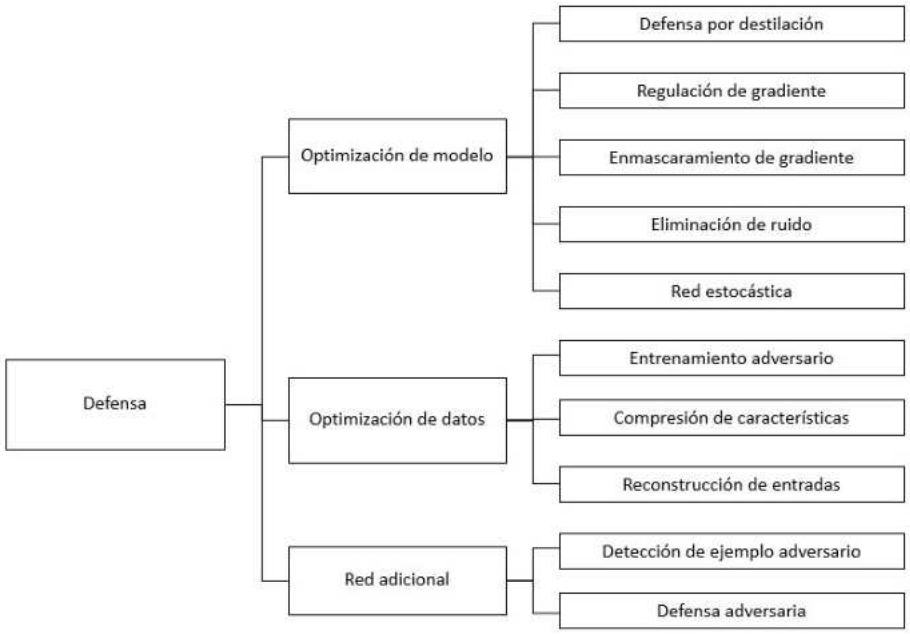

Figura 2.16: Métodos de defensa para redes neuronales profundas contra ataques adversarios [12].

Los métodos actuales demuestran una efectividad relativa mediante el uso de diferentes técnicas, sin embargo, no existe una solución universal efectiva, lo cual deja espacio para poder investigar más acerca de métodos para contrarrestar las afectaciones causadas por los ataques adversarios.

CAPÍTULO 3

Metodología

3.1. Hardware y Software

La sección de configuración de la experimentación 3.5 se conforma de diferentes tipos de modelos utilizados durante la experimentación, para los cuales se hizo uso del siguiente *hardware* y *software*:

- Modelo de tipo caja gris (subsección 3.5.1): El código fue realizado mediante el uso del lenguaje de programación Python 3.10.6, el modelo de la red neuronal profunda fue diseñada y desarrollada con el uso de las librerías Pytorch 1.12 y Pytorch-lightning 1.9.5. Se utilizo una computadora con 64GB de RAM, 12GB de memoria GPU (NVIDIA GeForce RTX3080Ti), y Ryzen 7 con ocho núcleos de CPU.

- Modelo de tipo caja blanca (subsección 3.5.2): Se utilizo el repositorio de [33], que hace uso del lenguaje de programación Python 3.7.16, generando el modelo de red neuronal profunda mediante Pytorch 1.13.1. Se utilizo la misma computadora descrita para el modelo de tipo caja gris y una segunda computadora con 128GB de RAM, 4 GPUs de 24GB de memoria GPU (NVIDIA GeForce RTX3090), y Ryzen Threadripper pro 7945WX con 24 núcleos de CPU. Esta ultima computadora se utilizo para realizar cuatro pruebas en paralelo de los modelos con distintos valores de hiperparámetros, esto permitía realizar el análisis y evaluaciones en menor tiempo.

3.2. Base de Datos

La base de datos utilizada para la experimentación en los diferentes tipos de modelos utilizados fue CIFAR-10. Esta base de datos es de acceso libre, y consiste de las siguientes características:

- 10 clases conformadas por: aviones, automóviles, aves, gatos, venados, perros, ranas, caballos, barcos, y camiones.

- 60,000 imágenes a color (3 canales) con un tamaño de 32x32 pixeles.
- 6,000 imágenes por clase.
- 50,000 imágenes para entrenamiento y 10,000 imágenes para prueba como lo recomienda el autor [20].

3.3. Métrica de Exactitud para la Clasificación de Imágenes

El desempeño de clasificación para los modelos de redes neuronales profundas puede ser medido mediante distintas métricas. En este trabajo de investigación se hizo uso de la métrica de exactitud Top-1.

La métrica de exactitud Top-1 fue seleccionada con base en las investigaciones relacionadas, cuya métrica utilizada para medir el desempeño de la DNN en tareas de clasificación es comúnmente la exactitud Top-1 [49, 33, 50, 51].

La exactitud fue calculada mediante el método de torchmetrics [64], la cual se define como:

$$Accuracy = \frac{1}{N}\sum_{i}^{N} 1(y_i = \hat{y}_i) \tag{3.1}$$

, donde y es el tensor de etiquetas verdaderas, y $\hat{y}$ es el tensor de las predicciones.

3.4. Métricas de Calidad de Imagen

El desempeño de la capa monogénica puede ser analizado y evaluado de manera cuantitativa mediante la comparación de la modificación estructural de las imágenes y sus activaciones monogénicas (fase local y orientación local) antes del ataque adversario y las obtenidas posterior al ataque adversario. Los resultados cuantitativos pueden ser obtenidos mediante el uso de métricas de calidad de imagen.

Algunas de las métricas comúnmente utilizadas para medir la fidelidad de las imágenes son la Medición del Índice de Similitud Estructural (SSIM, por sus siglas en inglés) y la Proporción Máxima Señal-Ruido (PSNR, por sus siglas en inglés).

Medición del Índice de Similitud Estructural

La métrica SSIM toma en consideración la estructura de la imagen y sus dependencias. Su proceso de funcionalidad está basado en las características individuales de las imágenes y su fuerte dependencia con sus vecinos. La métrica SSIM puede ser definida como [65]:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{({\mu_x}^2 + {\mu_y}^2) + C_1)({\sigma_x}^2 + {\sigma_y}^2 + C_2)} \tag{3.2}$$

donde x y y son dos arreglos de tamaño N x N, μ_x, μ_y, σ_x^2 y σ_y^2 son los valores medios y varianzas respectivamente, σ_{xy} es la covarianza de x y y, mientras que C_1 y C_2 son pequeños valores contantes positivos que tienen la función de estabilizar cada termino.

Proporción Máxima Señal-Ruido

La métrica PSNR es otro método comúnmente utilizado para medir la fidelidad de las imágenes, sin embargo, esta métrica tiende a tener un bajo desempeño debido a la limitación que tiene para medir distorsiones estructurales [66].

$$PSNR = 10 \log_{10} \frac{L^2}{MSE} \tag{3.3}$$

donde L es el rango de intensidad de los píxeles de las imágenes analizadas.

Ambas métricas (SSIM y PSNR) fueron implementadas mediante el uso computacional de los métodos suministrados por la librería Scikit-image [67].

Ataque Adversario PGD

El Descenso de Gradiente Proyectado (PGD, por sus siglas en inglés) es un ataque adversario con métricas de distancia L_0, L_2 y L_∞, las cuales son utilizadas para cuantificar la similitud de la imagen y el ejemplo adversario. L_0 hace uso de la medición de la cantidad de pixeles reemplazados en la imagen, L_2 usa la distancia Euclideana estandar (error cuadrático medio) de la imagen original y el ejemplo adversario, y L_∞ mide el cambio absoluto máximo a cualquier pixel [46].
La métrica L_∞ puede definirse como:

$||x - x'||_\infty = max(|x_1 - x'_1|, ..., |x_n - x'_n|)$

donde x es la imagen inicial y x' es la imagen modificada.

Este trabajo de investigación se basa en el uso de la métrica de distancia L_∞ para generar los ejemplos adversarios debido a su popular uso y argumentación en investigaciones sugieren que es la mejor métrica para usar [46].

3.5. Configuración para la Experimentación

La experimentación se ejecuto utilizando distintos modelos de la capa monogénica, donde únicamente cambia la selección de los canales que salen después del procesamiento de la capa monogénica, y que son suministrados a la red neuronal profunda para su posterior proceso de clasificación. Las ecuaciones [2.1-2.11] fueron utilizadas para codificar la capa monogénica en todos los modelos utilizados.

La capa monogénica fue puesta a prueba como método de defensa de optimización de datos contra ataques de caja gris y caja blanca PGD con métrica de distancia L_{inf}. La descripción de las configuraciones de los experimentos se describe en las subsecciones 3.5.1 y 3.5.2.

3.5.1. Modelo de Defensa de Capa Monogénica Confrontando un Ataque Adversario PGD L_{inf} de Caja Gris

El ataque adversario de tipo caja gris hace uso de la estructura de la red neuronal profunda que será atacada para poder generar los ejemplos adversarios, sin embargo, este tipo de ataque no tiene conocimiento de los pesos de la red entrenada [58].

La selección de este tipo de ataque se debe a las características de la capa monogénica, la capa monogénica se utiliza como método de defensa de optimización de datos, por lo cual se encuentra fuera de la red neuronal densa. Cuando se realiza el entrenamiento, la red neuronal densa espera una entrada de seis canales debido a la configuración de canales de salida de la capa monogénica, posterior al entrenamiento los ejemplos adversarios hacen uso de la estructura de la red, sin embargo, no se puede utilizar la estructura de la red preentrenada debido a que espera una entrada de seis canales, y las imágenes limpias únicamente cuentan con los ya conocidos tres canales de las imágenes (RGB).

Para solucionar el problema de la generación de ejemplos adversarios con la estructura de la red se hace uso de la estructura común de la ResNet-50 (sin pesos precargados) que espera una entrada de tres canales, las pruebas se realizan de la forma común, haciendo uso de los ejemplos adversarios que son introducidos a la capa monogénica, y suministra los seis canales a la red preentrenada para obtener una clasificación.

Base de Datos

La base de datos CIFAR-10 fue obtenida de la librería torchvision y configurada mediante el método *DataLoader* de la misma librería usando las siguientes configuraciones de transformación:

- Conjunto de datos de entrenamiento:
  - *Random crop* = 32
  - *Padding* = 4
  - *Random horizontal flip*
  - Normalización : media = (0.4914, 0.4822, 0.4465), y desviación estándar = (0.2023, 0.1994, 0.2010)
- Conjunto de datos de prueba:
  - Normalización : media = (0.4914, 0.4822, 0.4465), y desviación estándar = (0.2023, 0.1994, 0.2010)

Ataque Adversario

La librería *Fool box* [32] fue utilizada para codificar el ataque adversario L_{inf} PGD de tipo caja gris, y generar los ejemplos adversarios. La configuración para generar los ejemplos adversarios fue la siguiente:

- Pasos = 20
- Inicio aleatorio.
- Rango de limites = [-2.42906, 2.75373]
- Tamaño de paso = 0.03333
- Valores de ϵ (epsilon)= [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]
- Estructura de la red neuronal profunda: Estructura común de la red neuronal profunda ResNet-50 sin pesos precargados.

El valor de 20 para la configuración de pasos e inicio aleatorio fueron seleccionados como valores comunes utilizados en algunas investigaciones [68, 69, 33], el rango de limites fue seleccionado de acuerdo a los valores mínimo y máximo de las imágenes del conjunto de datos después de la transformación inicial, el valor de tamaño de paso corresponde al valor predefinido por el método de la librería *Fool box*, y los valores de epsilon fueron seleccionados para cubrir un amplio rango de valores (Figura 3.1).

Figura 3.1: Ejemplo de una imagen procesada por diferentes valores de epsilon. A es la imagen limpia, A1 es la orientación local (θ) de A, A2 es la fase local (ϕ) de A, B es el ejemplo adversario de A producido por su correspondiente valor de epsilon, B1 es la orientación local (θ) de B y B2 es la fase local (ϕ) de B como corresponde con su valor de epsilon.

.

Red Neuronal Profunda para la Clasificación de Imágenes

ResNet-50 [34] sin pesos precargados fue la red neuronal seleccionada para esta configuración de pruebas. Se realizo una modificación al modelo original para permitir la entrada de los seis canales que son generados posterior al proceso de la capa monogénica (fase local y orientación local). Las características principales de la estructura de la red neuronal profunda están resumidas en el cuadro 3.1, y la representación gráfica de la estructura del proceso completo puede observarse en la Figura 3.2.

Cuadro 3.1: Resumen de la estructura de la red ResNet-50 utilizada para el atacante de caja gris.

Nombre de la capa	Canales de entrada a la capa	Tamaño de salida	Contenido	
Capa 1	6	32 x 32	Conv 1x1, 64 Conv 3x3, 64 Conv 1x1, 256	x3
Capa 2	256	16 x 16	Conv 1x1, 128 Conv 3x3, 128 Conv 1x1, 512	x4
Capa 3	512	8 x 8	Conv 1x1, 256 Conv 3x3, 256 Conv 1x1, 1024	x6
Capa 4	1024	4 x 4	Conv 1x1, 512 Conv 3x3, 512 Conv 1x1, 2048	x3
		1 x 1	AvgPool2d, 10-d fc, softmax	

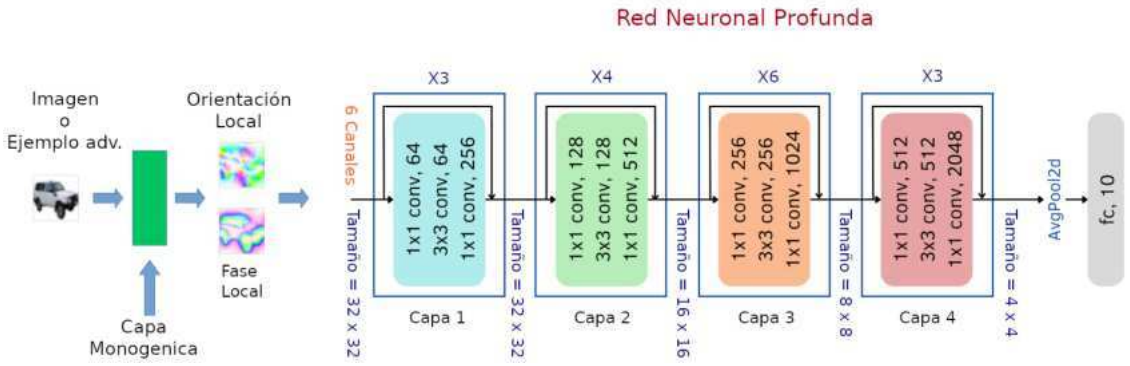

Figura 3.2: Ilustración de la estructura de la red neuronal densa con la capa frontal monogénica.

Entropía cruzada y Descenso de Gradiente Estocástico (SGD, por sus siglas en inglés) fueron implementados en esta red neuronal profunda como la función de pérdida y la función de optimización respectivamente. Ambas implementaciones fueron codificadas mediante el uso de la librería Torch [70]. La cantidad total de parámetros entrenables resultantes de la DNN fueron 23.5 millones.

El desempeño de la capa monogénica en tareas de clasificación de imágenes fue obtenido mediante la comparación de la tasa de clasificación Top-1 de ambos modelos utilizados:

- Modelo de la DNN sin la capa monogénica (sin método de defensa).
- Modelo de la DNN con la capa monogénica (con método de defensa).

Entrenamiento

Ambos modelos fueron entrenados con las 50,000 imágenes de entrenamiento y probados con las 10,000 imágenes de prueba previamente categorizadas por el autor de CIFAR-10 [20].

El proceso de entrenamiento utilizado para el modelo de red neuronal profunda con la capa monogénica puede observarse en el diagrama de flujo de la Figura 3.3.

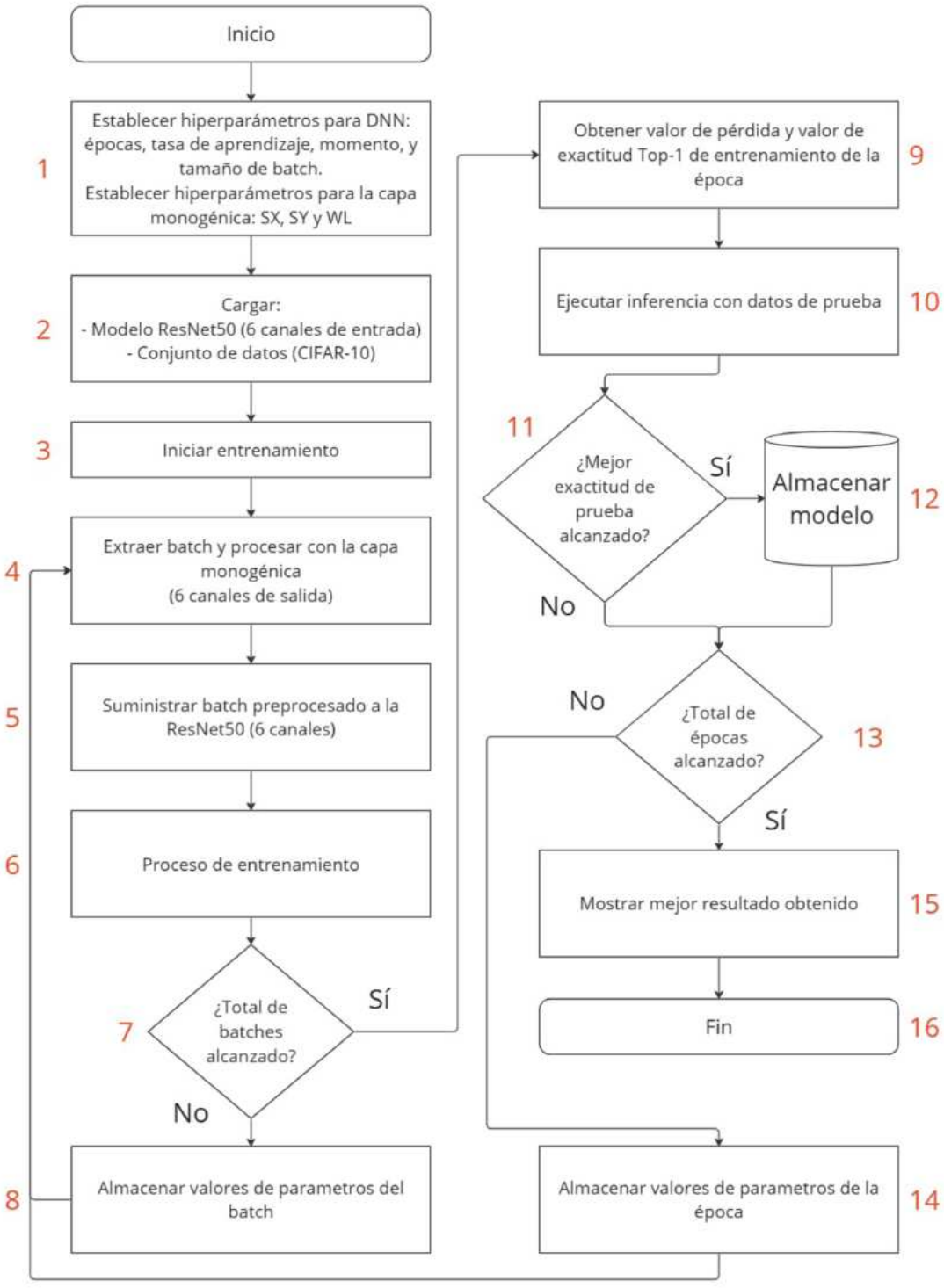

Figura 3.3: Diagrama de flujo del proceso de entrenamiento para le modelo con la capa monogénica con seis canales de salida.

Descripción a detalle del proceso de entrenamiento del modelo con capa monogénica:

- 1. Definición de los hiperparámetros requeridos por la red neuronal profunda y para la capa monogénica.
- 2. Definición de la estructura de la red ResNet-50 con la modificación de seis canales a la entrada, y obtención del conjunto de datos CIFAR-10 (conjunto de entrenamiento y conjunto de pruebas).
- 3. Inicio del proceso de entrenamiento.
- 4. Obtención del batch_n, y procesamiento de las imágenes limpias con la capa monogénica. El proceso de la capa monogénica genera una salida de tres canales (orientación local y fase local).
- 5. Suministro a la ResNet-50 de los tres canales correspondientes a la orientación local y fase local de las imágenes obtenidas del batch_n.
- 6. Proceso regular de entrenamiento para redes neuronales profundas con los hiperparámetros, función de pérdida, y el optimizador establecidos.
- 7. Toma de decisión para asegurar que el entrenamiento por época procese todas la imágenes del conjunto de datos.
- 8. Almacenamiento de los valores de parámetros actuales para reanudar el proceso con el batch_{n+1}.
- 9. Obtención del valor de pérdida y valor de exactitud obtenidos en la época_n de entrenamiento mediante la función de pérdida establecida.
- 10. Ejecución de inferencia con el conjunto de prueba para obtener el desempeño del entrenamiento.
- 11. Toma de decisión para almacenar la estructura de la red y sus parámetros de entrenamiento, si se alcanza una mejor exactitud en la inferencia.
- 12. Almacenamiento de la estructura de la red y sus parámetros de entrenamiento de la época_n.
- 13. Toma de decisión para asegurar que se ha alcanzado el numero N de épocas establecido en los hiperparámetros.
- 14. Almacenamiento de los valores de parámetros obtenidos al finalizar la época_n para reanudar el proceso con la época_{n+1}.
- 15. Impresión en pantalla de los valores de pérdida y exactitud obtenidos de la época con mejor desempeño.
- 16. Termino del proceso de entrenamiento.

Prueba

La experimentación realizada para el proceso de pruebas con los ejemplos adversarios puede observarse en el diagrama de flujo de la Figura 3.4:

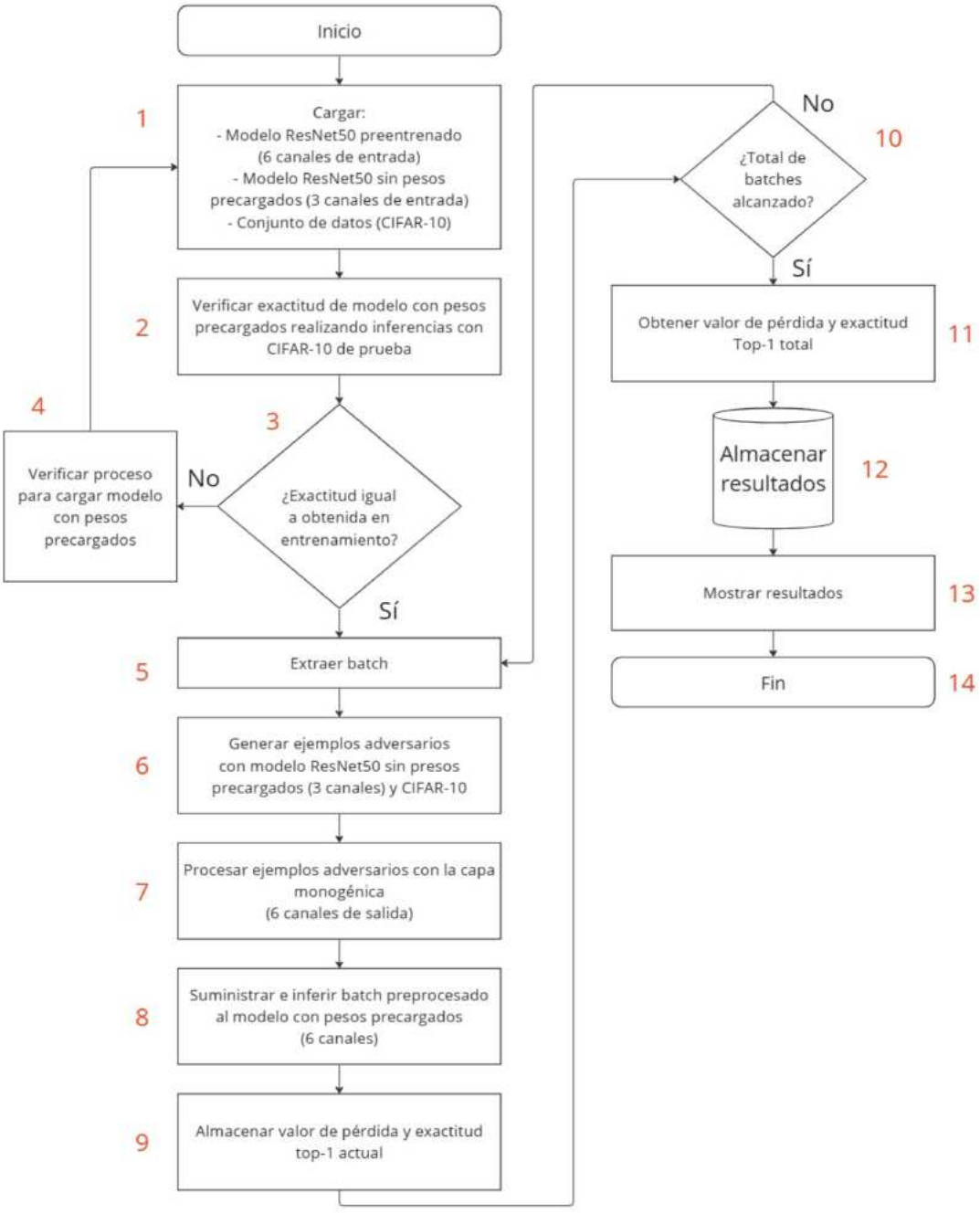

Figura 3.4: Diagrama de flujo del proceso de prueba para el modelo con capa monogénica con seis canales de salida.

En las pruebas con ejemplos adversarios para ambos modelos las mismas imágenes fueron atacadas con el mismo ataque adversario (PGD L_{inf} de caja gris), los mismos valores de epsilon ([0.1 - 1.0]), el mismo número de pasos (20), y la configuración descrita en 3.5.1-*Ataque Adversario*.

Descripción a detalle del proceso de prueba del modelo con la capa monogénica:

- 1. Obtención del modelo ResNet-50 (seis canales de entrada) con pesos precargados obtenidos del entrenamiento para ejecutar las inferencias de prueba, obtención del modelo ResNet-50 (tres canales) sin pesos precargados para generar los ejemplos adversarios, y obtención del conjunto de datos para prueba de CIFAR-10.
- 2. Verificación del correcto cargado del modelo preentrenado mediante la ejecución de inferencias con el conjunto de datos de prueba. Este proceso requiere pasar cada imagen limpia de prueba por la capa monogénica y realizar las inferencias, el resultado de exactitud debe de ser el mismo al obtenido en el entrenamiento.
- 3. Toma de decisión para corroborar el correcto cargado del modelo y parámetros obtenidos en el entrenamiento.
- 4. Verificación de proceso para cargar el modelo con parámetros obtenidos en el entrenamiento, si la exactitud obtenida de la inferencia no es la esperada.
- 5. Obtención del batch_n.
- 6. Generación de los ejemplos adversarios mediante los métodos de la librería *Fool box*, con la configuración descrita en 3.5.1-*Ataque Adversario*, y el uso de la estructura de la ResNet-50 con tres canales de entrada. La ResNet-50 con tres canales de entrada permitirá generar los ejemplos adversarios con las imágenes de entrada (tres canales).
- 7. Procesamiento de los ejemplos adversarios con la capa monogénica (activación del método de defensa). El objetivo de este proceso es remover las afectaciones causadas a las imágenes (ejemplos adversarios) por el ataque adversario para suministrar a la red de clasificación imágenes limpias. Las imágenes de entrada a la capa monogénica son de tres canales y el proceso de la capa monogénica genera una salida de seis canales (orientación y fase local).
- 8. Ejecución de inferencia con las imágenes procesadas por la capa monogénica (seis canales), y la red ResNet-50 de seis canales con los pesos precargados.
- 9. Almacenamiento del valor de pérdida y exactitud obtenidos en el batch_n para reanudar con el batch_{n+1}.
- 10. Toma de decisión para asegurar que se han procesado los N batches.
- 11. Obtención de valores de pérdida y exactitud finales.
- 12. Almacenamiento de los resultados para su posterior análisis.

- 13. Impresión en pantalla de los valores obtenidos en el proceso de inferencia de los ejemplos adversarios procesados por la capa monogénica.
- 14. Fin del proceso de prueba.

3.5.2. Modelo de Defensa de Capa Monogénica Confrontando un Ataque Adversario PGD L_{inf} de Caja Blanca

El ataque adversario de tipo caja blanca a diferencia del ataque de tipo caja gris, cuenta con la estructura y los pesos de la red preentrenada que será atacada [12].

La configuración del atacante para este conjunto de pruebas utiliza la capa monogénica como método de defensa de optimización de datos, por lo cual se encuentra fuera de la red neuronal densa. La configuración de los canales de salida de la capa monogénica fue limitada a tres canales, esto para permitir la generación de los ejemplos adversarios usando el ataque de tipo caja blanca (estructura y pesos del modelo preentrenado). Los ejemplos adversarios fueron generados mediante el uso de la librería [33].

Base de Datos

La librería [33] obtiene el conjunto de datos CIFAR-10 de la librería torchvision, y configura los datos mediante el método *DataLoader* de la misma librería usando las siguientes configuraciones de transformación:

- Conjunto de datos de entrenamiento:
  - *CenterCrop* = 32
  - *Random horizontal flip.*
  - *ColorJitter* = 0.25, 0.25, 0.25
  - *Random rotation* = 2
  - *ToTensor()*
- Conjunto de datos de prueba:
  - *CenterCrop* = 32
  - *ToTensor()*

Ataque Adversario

La configuración para general los ejemplos adversarios fue la siguiente:

- Pasos = 20
- Inicio aleatorio.

- Tamaños de paso = [0.003921, 0.007843]
- Valores de ϵ (epsilon)= [0.0, 0.0313, 0.0627]
- Estructura de la red neuronal profunda: Red neuronal profunda ResNet-50 con estructura común y pesos precargados obtenidos del entrenamiento con la capa monogénica.

El valor de 20 para la configuración de pasos, inicio aleatorio, el valor de tamaño de paso, y los valores de epsilon fueron los mismos valores de configuración utilizados en [33]. La muestra de las imágenes obtenidas de los ejemplos adversarios generados con los parámetros utilizados se puede observar en la Figura 3.5

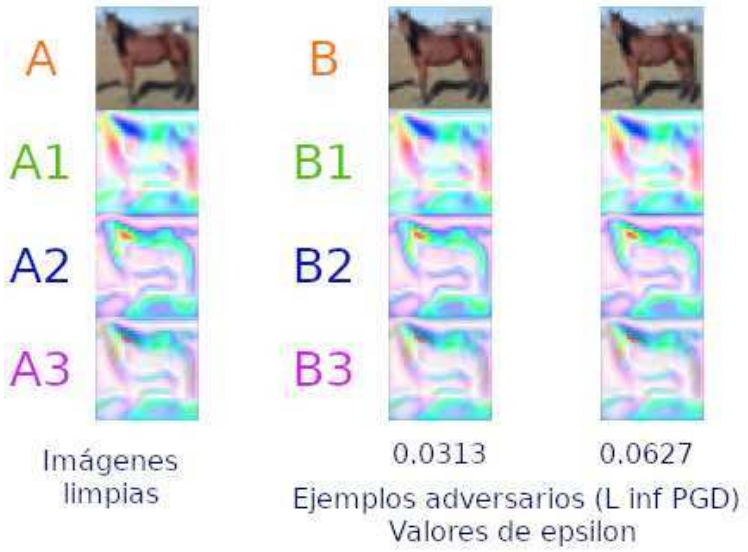

Figura 3.5: Ejemplo de una imagen procesada por diferentes valores de epsilon. A es la imagen limpia, A1 es la orientación local (θ) de A, A2 es la fase local (ϕ) de A, A3 es la media de la orientación y fase local, B es el ejemplo adversario de A producido por su correspondiente valor de epsilon, B1 es la orientación local (θ) de B, B2 es la fase local (ϕ) de B, y B3 es la media de la orientación y fase local de B como corresponde con su valor de epsilon.

.

Red Neuronal Profunda para la Clasificación de Imágenes

ResNet-50 [34] con pesos precargados del entrenamiento con el proceso de la capa monogénica fue la red neuronal seleccionada para esta configuración de pruebas. Las características principales de la estructura de la red neuronal profunda están resumidas en el Cuadro 3.2.

La experimentación de la capa monogénica como método de defensa confrontando un ataque adversario de tipo caja blanca requiere de una entrada de tres canales a la red neuronal preentrenada, por lo tanto, la experimentación se llevo acabo con el uso de tres configuraciones distintas en los canales de salida de la capa monogénica.

Cuadro 3.2: Resumen de la estructura de la red ResNet-50 utilizada para el atacante de caja blanca.

Nombre de la capa	Canales de entrada a la capa	Tamaño de salida	Contenido	
Capa 1	3	32 x 32	Conv 1x1, 64 Conv 3x3, 64 Conv 1x1, 256	x3
Capa 2	256	16 x 16	Conv 1x1, 128 Conv 3x3, 128 Conv 1x1, 512	x4
Capa 3	512	8 x 8	Conv 1x1, 256 Conv 3x3, 256 Conv 1x1, 1024	x6
Capa 4	1024	4 x 4	Conv 1x1, 512 Conv 3x3, 512 Conv 1x1, 2048	x3
		1 x 1	AvgPool2d, 10-d fc, softmax	

Las tres configuraciones utilizadas en este conjunto de experimentos nos permite analizar el desempeño del modelo con el uso de la fase local, la orientación local y la media de ambas imágenes. Para casos prácticos denominaremos cada uno de los modelos de la siguiente forma: CM_{ori} Figura 3.6 , CM_{fase} Figura 3.7, y CM_{media} Figura 3.8. De igual forma, para casos prácticos el modelo propuesto por [33] será denominado $Madry_R$.

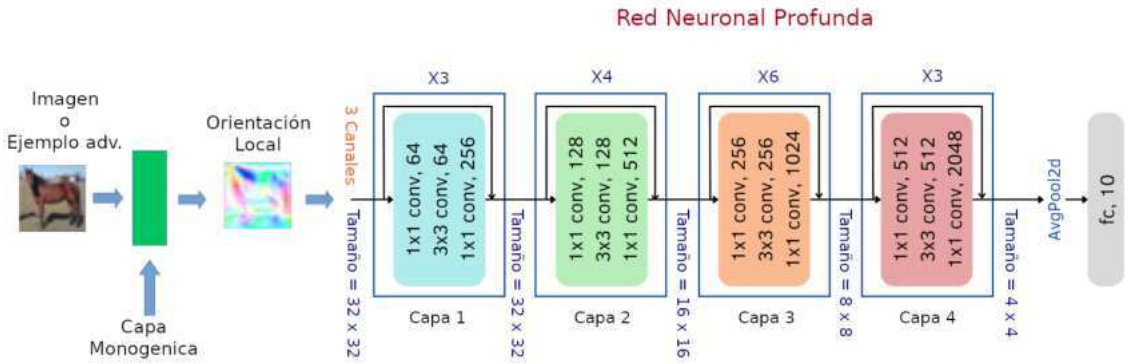

Figura 3.6: Ilustración de la estructura de la DNN con la capa frontal monogénica con la orientación local como salida (CM_{ori}).

Cross-entropy y el Descenso de Gradiente Estocástico (SGD, por sus siglas en inglés) son la función de pérdida y la función de optimización respectivamente utilizadas en estos experimentos. Ambas implementadas mediante el uso de la librería Torch [70]. La cantidad total

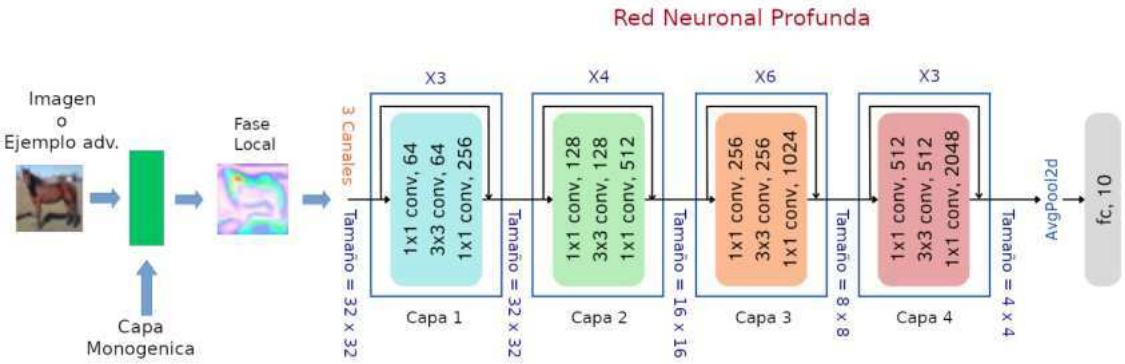

Figura 3.7: Ilustración de la estructura de la DNN con la capa frontal monogénica con la fase local como salida (CM_{fase}).

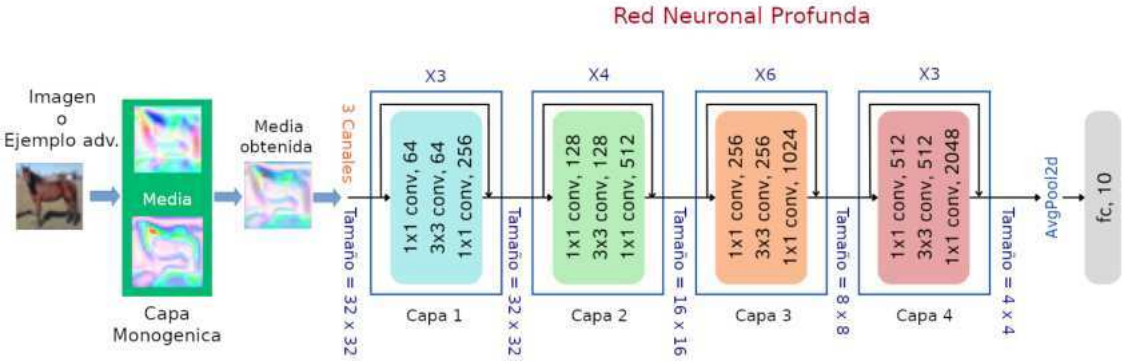

Figura 3.8: Ilustración de la estructura de la DNN con la capa frontal monogénica con la media obtenida de la orientación y fase local como salida CM_{media}.

de parámetros entrenables para cada una de las redes neuronales profundas es 23,499,466.

El desempeño de clasificación se obtuvo mediante la comparación de la tasa de clasificación top-1 de los modelos correspondientes a las Figuras 3.6, 3.7 y 3.8 contra los resultados obtenidos por el modelo $Madry_R$.

Los modelos de defensa utilizados para este conjunto de experimentos tienen las siguientes características:

- Modelos de red neuronal profunda con capa monogénica:
  - Modelos con tipo de defensa de optimización de datos.
  - Preprocesamiento de los datos mediante una capa monogénica que obtiene la fase local y la orientación local de las imágenes mediante el uso de la señal monogénica y un filtro Gabor. Los procesos ejecutados dentro de la capa monogénica son realizados en el dominio de Fourier.
  - El atacante no tiene acceso al método de defensa.

- Modelo de entrenamiento robusto $Madry_R$:
  - Modelo con tipo de defensa de optimización de datos.
  - Entrenamiento robusto: Uso de imágenes limpias y ejemplos adversarios.
  - El entrenamiento se realiza con ejemplos adversarios generados con valores únicos de configuración: epsilon (0.0313), pasos (20), tamaño de paso (0.003921), e inicio aleatorio.
  - El atacante tiene acceso al método de defensa. El modelo de la red es entrenado con los ejemplos adversarios, por lo cual, el atacante tiene conocimiento de las características completas de las imágenes utilizadas para el entrenamiento.

Entrenamiento

Los tres modelos fueron entrenados con las 50,000 imágenes de entrenamiento y probados con las 10,000 imágenes de prueba previamente categorizadas por el autor de CIFAR-10 [20].

El proceso de entrenamiento para cada uno de los tres modelos utilizados puede observarse en el diagrama de flujo de la Figura 3.9.

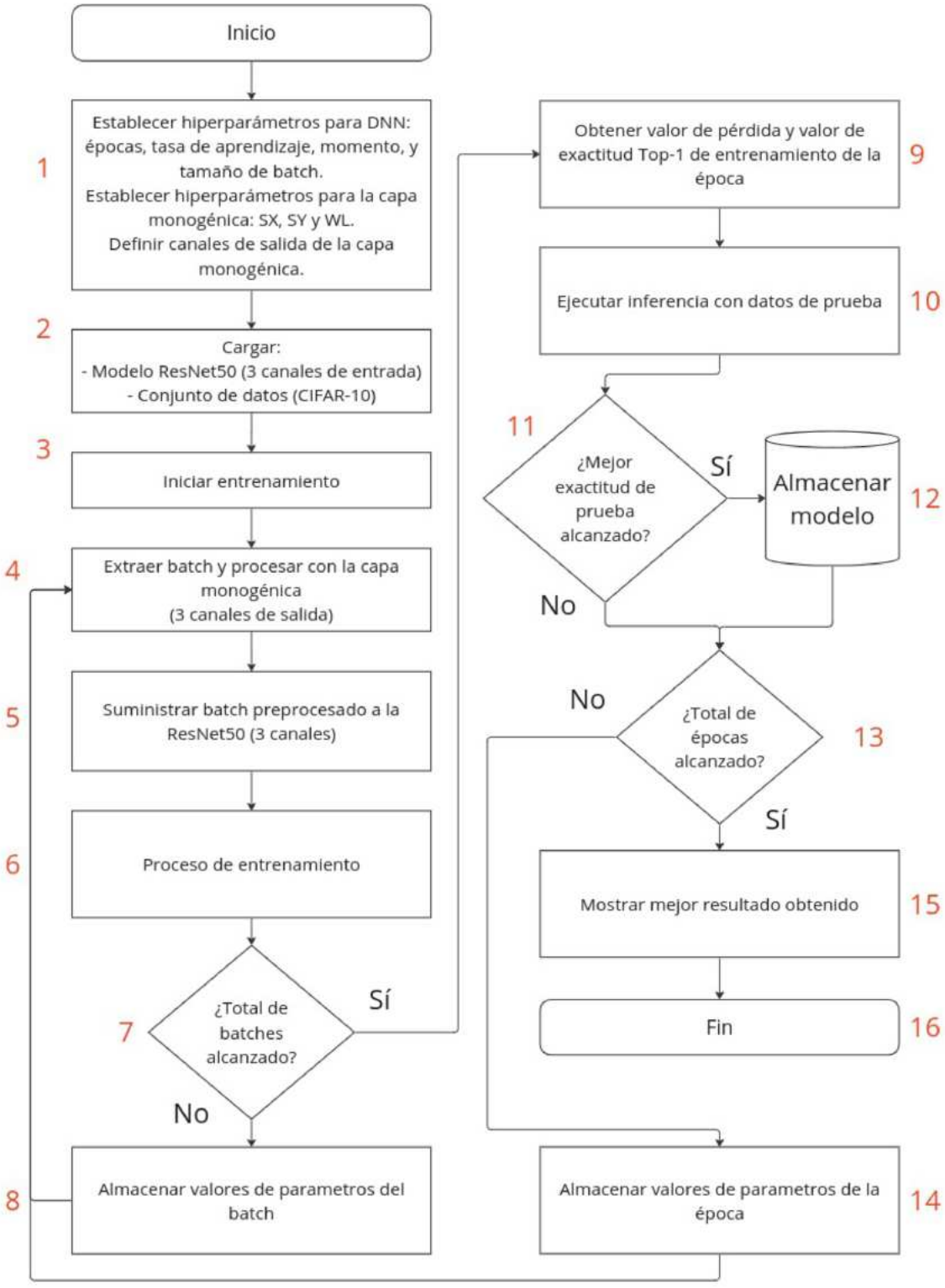

Figura 3.9: Diagrama de flujo proceso de entrenamiento para el modelo con capa monogénica con tres canales de salida.

Como referencia de la descripción a detalle del entrenamiento puede referirse a la descrita en 3.5.1-*Entrenamiento*, con modificaciones en los siguientes puntos:

- 1. Se configura la capa monogénica para definir los canales de salida.
- 2. El modelo que se carga es la ResNet-50 con tres canales de entrada.
- 4. El procesamiento de la capa monogénica genera una salida de tres canales según la configuración del modelo utilizado (orientación local, fase local o media de la fase y orientación local).
- 5. Se suministran tres canales a la ResNet-50 de tres canales de entrada.

La experimentación ejecutada para el proceso de pruebas con los ejemplos adversarios puede observarse en el diagrama de flujo de la Figura 3.10:

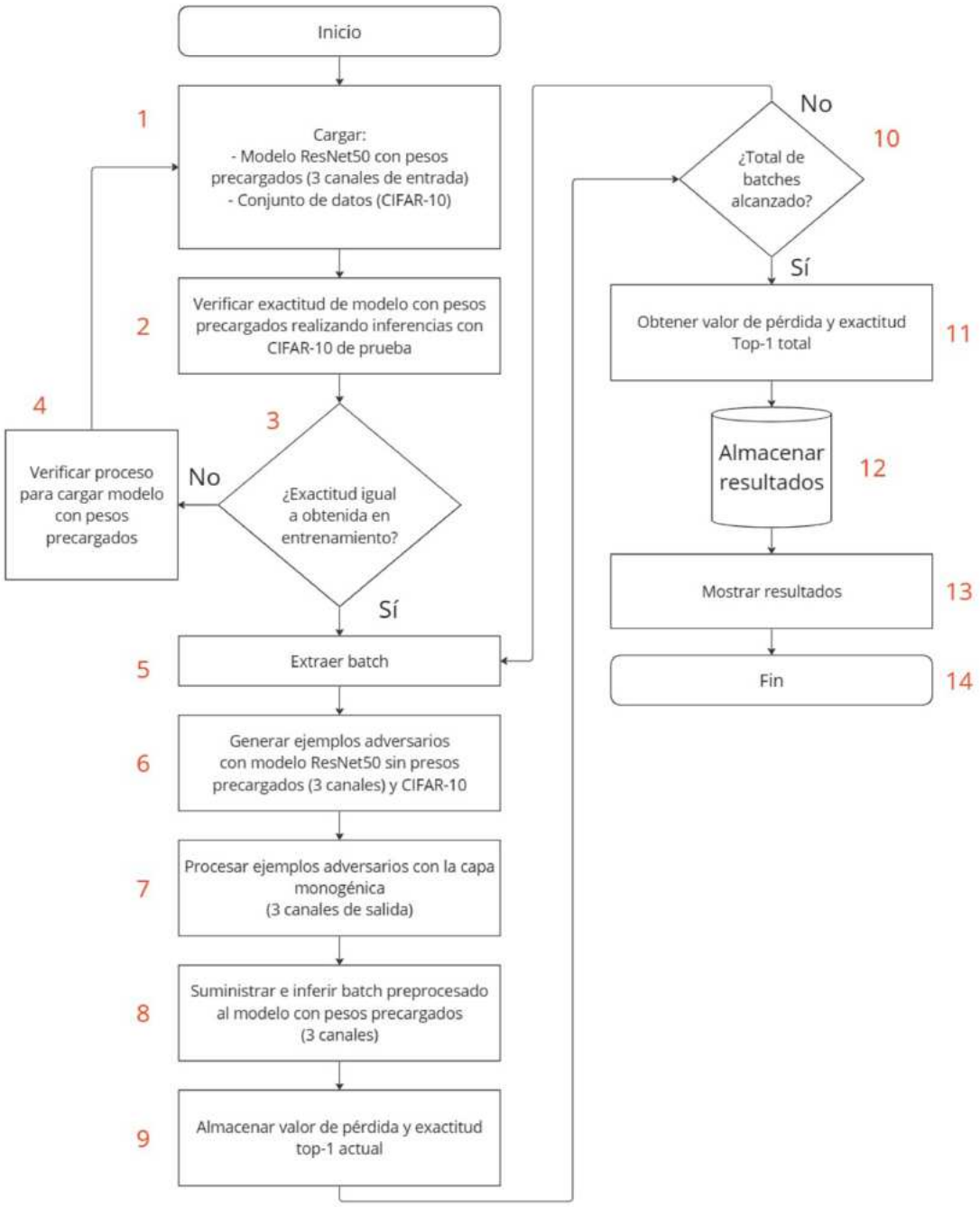

Figura 3.10: Diagrama de flujo del proceso de prueba para el modelo con capa monogénica con tres canales de salida.

Prueba

La experimentación se llevo acabo con el uso de los mismos ejemplos adversarios (PGD L_{inf} de caja gris), mismos valores de epsilon ([0.0313, 0.0627]), mismo número de pasos (20), y mismas características descritas en 3.5.2-*Ataque Adversario* para los tres modelos.

Como referencia de la descripción a detalle del proceso de prueba puede referirse a la descrita en 3.5.1-*Prueba*, con modificaciones en los siguientes puntos:

- 1. Únicamente se carga el modelo ResNet-50 (tres canales de entrada) con los pesos obtenidos en el entrenamiento.
- 2. El proceso de inferencia para verificar el correcto cargado del modelo preentrenado implica el proceso de la capa monogénica que genera una salida de tres canales de acuerdo a su configuración.
- 6. Los ejemplos adversarios son generados mediante métodos de [33] con la configuración descrita en 3.5.2-*Ataque Adversario*, y con la ResNet-50 preentrenada de tres canales de entrada.
- 7. Tres canales son generados después del proceso de la capa monogénica.
- 8. Tres canales son suministrados a la red de clasificación ResNet-50 preentrenada de tres canales.

CAPÍTULO 4

Resultados y Discusión

4.1. Resultados

Se utilizaron dos métricas para medir el desempeño de la capa monogénica, y una tercer métrica para la tarea de clasificación de la DNN. PSNR y SSIM fueron aplicadas para medir la fidelidad de la imagen, y exactitud top-1 para el desempeño de clasificación.

4.1.1. Modelo con Capa Monogénica Confrontando un Ataque Adversario PGD L_{inf} de Caja Gris

Los valores obtenidos de las métricas PSNR y SSIM, muestran un mejor desempeño para los pares de imágenes obtenidos de la fase local y de la orientación local que de los obtenidos por los pares de imágenes iniciales (imagen limpia y ejemplo adversario).

Inicialmente se hizo la medición de fidelidad de imágenes SSIM con un solo par de imágenes (imagen inicial y su ejemplo adversario con la fase local y orientación local correspondientes), los resultados obtenidos de los pares de imágenes de la fase local y la orientación local mostraron un mejor desempeño que los pares de imágenes iniciales (Figura 4.1).

De forma similar los valores obtenidos de los pares de imágenes de la fase local y la orientación local de la métrica PSNR mostraron un mejor desempeño que los pares de imágenes iniciales (Figura 4.2).

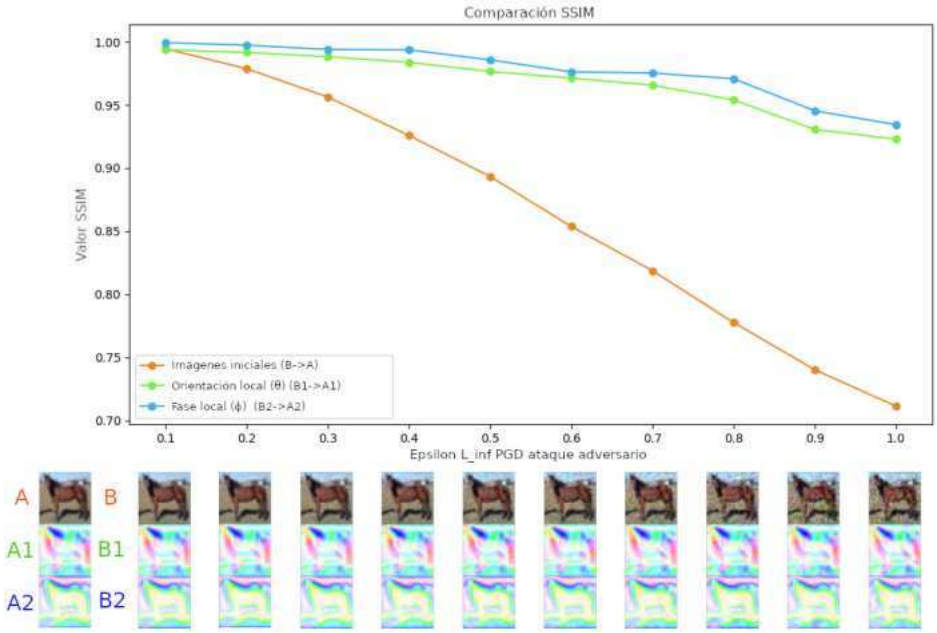

Figura 4.1: Métrica SSIM calculada para una imagen, su fase local y su orientación local del conjunto de pruebas con múltiples valores de epsilon para los ejemplos adversarios.

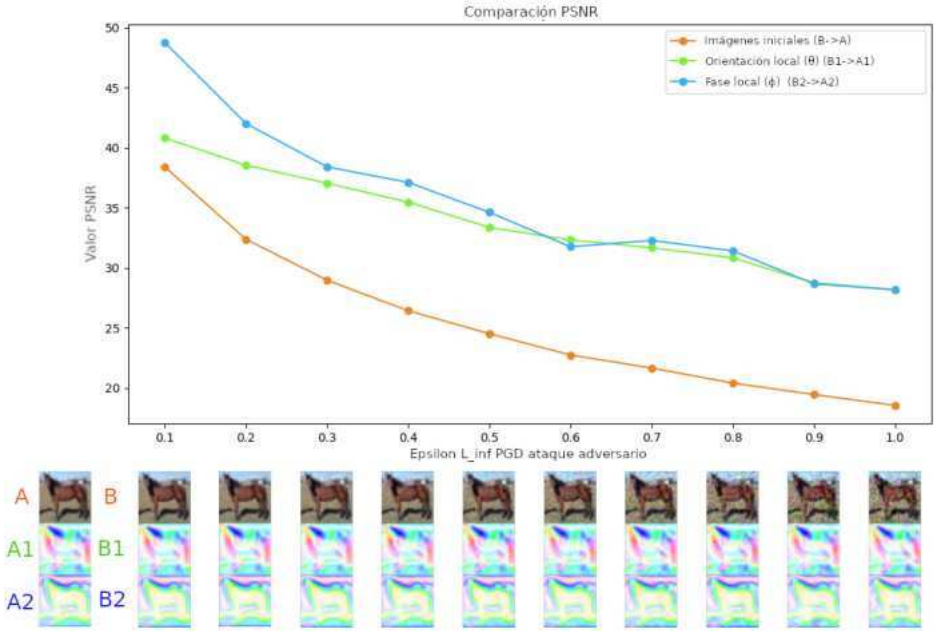

Figura 4.2: Métrica PSNR calculada para una imagen, su fase local y su orientación local del conjunto de pruebas con múltiples valores de epsilon para los ejemplos adversarios.

Los resultados mostrados en las Figuras 4.1 y 4.2 pueden interpretarse como un buen

desempeño de la capa monogénica. Sin embargo, es requerido aumentar la muestra de la experimentación para obtener datos más confiables. 3,000 imágenes seleccionadas aleatoriamente, y sin repetición, fueron tomadas de las 10,000 imágenes del conjunto de pruebas para realizar las mismas mediciones.

Los resultados de la métrica SSIM que se observan en el gráfico de cajas (Figura 4.3) muestran una distribución más compacta para los pares de imágenes de la fase local y la orientación local, valores de media mayores, y menor cantidad de valores atípicos que los obtenidos de los pares de imágenes iniciales.

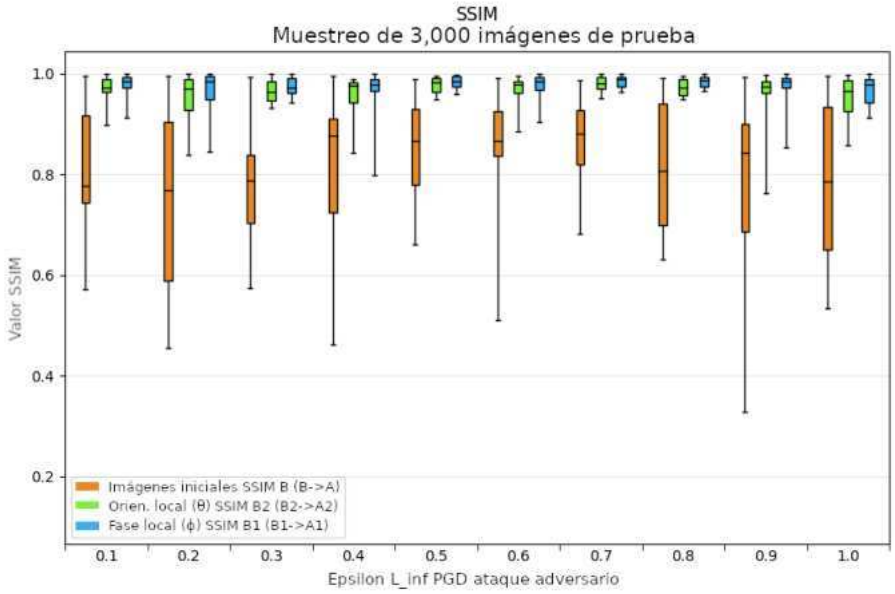

Figura 4.3: Gráfico de cajas de la métrica SSIM obtenida de 3,000 imágenes seleccionadas aleatoriamente y sin repetición con múltiples valores de epsilon para el atacante de caja gris.

Por otra parte, los resultados obtenidos de la métrica PSNR que se pueden observar en el gráfico de cajas (Figura 4.4) muestran una distribución más dispersa, sin embargo, se puede observar que la fase local y la orientación local tienen una distribución más centrada, y valores de media mayores que las imágenes iniciales.

Los resultados de la tarea de clasificación son calculados mediante la exactitud top-1. El modelo sin método de defensa alcanzo un valor máximo de 0.8674 de exactitud durante el entrenamiento, y el modelo con la capa monogénica alcanzo un valor máximo de 0.7641 durante el entrenamiento. Es posible que la exactitud máxima alcanzada por el modelo con la capa monogénica pueda incrementarse mediante técnicas de ajuste fino o algún otro método que ayude a mejorar su desempeño, sin embargo, estos resultados pueden servir para comparar el desempeño de ambos modelos.

Los resultados muestran valores de exactitud más estables al incrementar los valores de epsilon para el modelo con la capa monogénica que el modelo sin método de defensa (Figura 4.5).

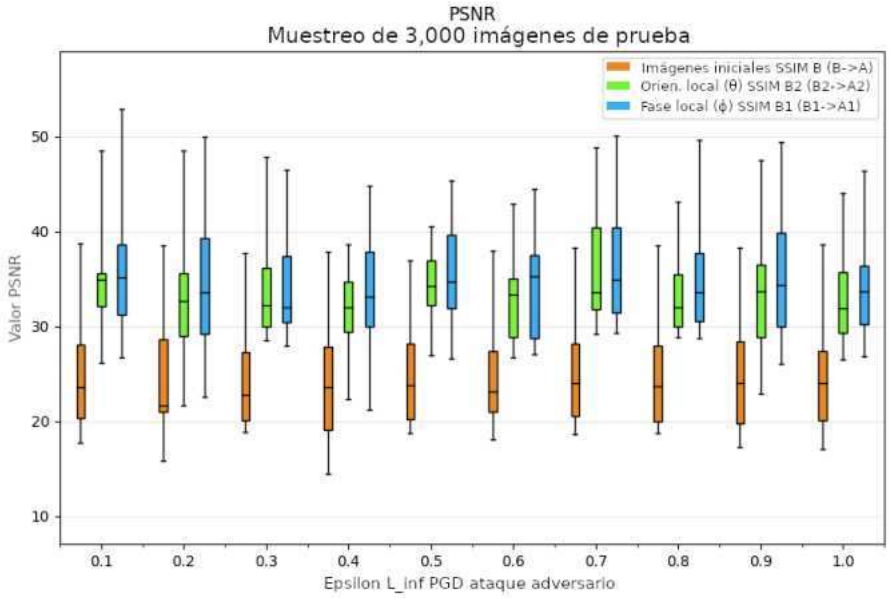

Figura 4.4: Gráfico de cajas de la métrica PSNR obtenida de 3,000 imágenes seleccionadas aleatoriamente y sin repetición con múltiples valores de epsilon para el atacante de caja gris.

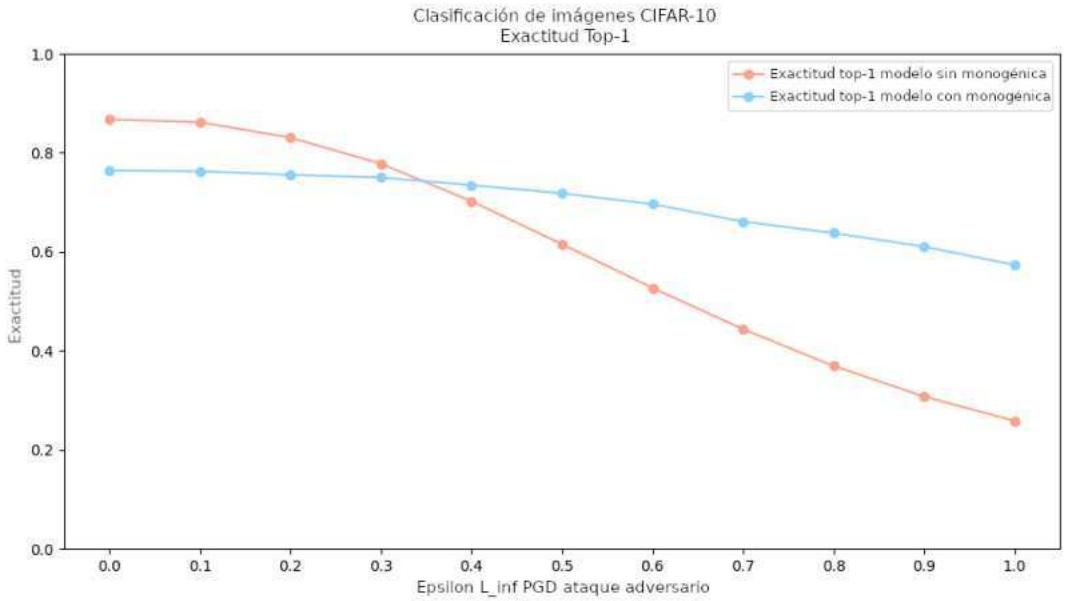

Figura 4.5: Exactitud top-1 de la tarea de clasificación para el atacante de caja gris.

La comparación de la tasa de decremento basado en la exactitud máxima alcanzada para ambos modelos podría proporcionar una manera más fácil de observar el nivel de afectación para cada uno de ellos al incrementar los valores de epsilon. El gráfico de barras de la

Figura 4.6 muestra valores de exactitud más estables y con menor nivel de afectación para el modelo con la capa monogénica. La afectación para el modelo sin método de defensa es más notable, su métrica de exactitud cayo a un 30 % para el valor máximo de epsilon evaluado (1.0), mientras que el desempeño del modelo con la capa monogénica sufrió menor nivel de afectación manteniendo su desempeño en 75 %.

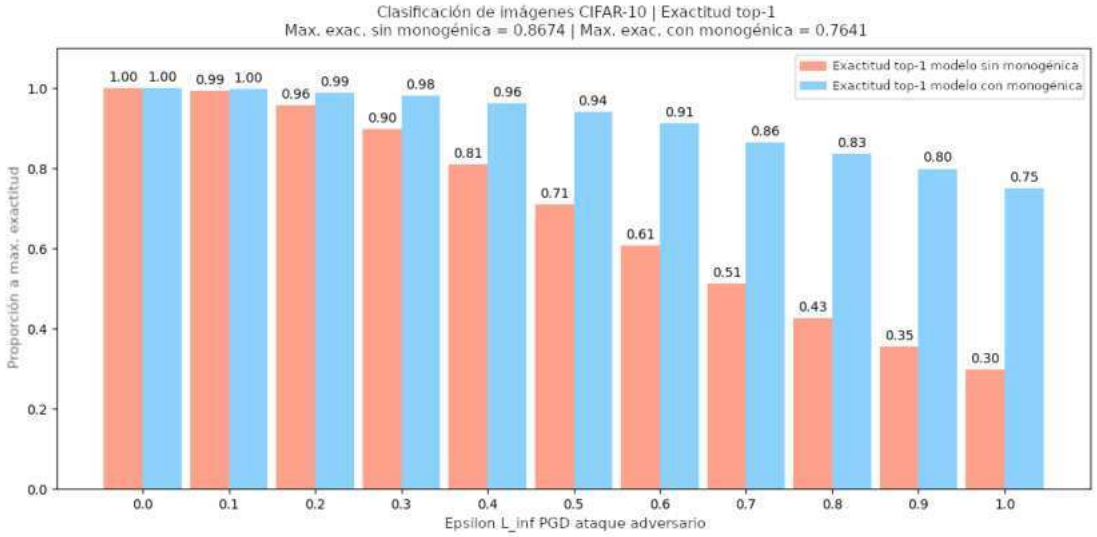

Figura 4.6: Tasa de decremento basado en la exactitud máxima alcanzada para ambos modelos con un atacante de caja gris.

4.1.2. Modelo con Capa Monogénica Confrontando un Ataque Adversario PGD L_{inf} de Caja Blanca

Siguiendo el mismo proceso de análisis para los resultados de la experimentación, se hace uso de las métricas SSIM y PSNR para medir la fidelidad de las imágenes, y la exactitud top-1 para medir el desempeño de clasificación de la DNN. Sin embargo, en esta sección se muestran los resultados obtenidos para cada una de las tres configuraciones distintas utilizadas para la salida de la capa monogénica.

Resultados de la Capa Monogénica con la Orientación Local como Salida (CM_{ori})

Los resultados de la métrica SSIM que se pueden observar en el gráfico de cajas (Figura 4.7) muestra una distribución más compacta y valores de media mayores para las imágenes procesadas por la capa monogénica que los obtenidos de las imágenes iniciales.

Los valores obtenidos de la métrica PSNR generó distribuciones más amplias que las de la métrica SSIM, lo cual nos indica que hay mayor dispersión en los resultados obtenidos, no obstante, se puede observar que la distribución de las imágenes de la capa monogénica tiene un valor de media mayor que los de las imágenes iniciales (Figura 4.8).

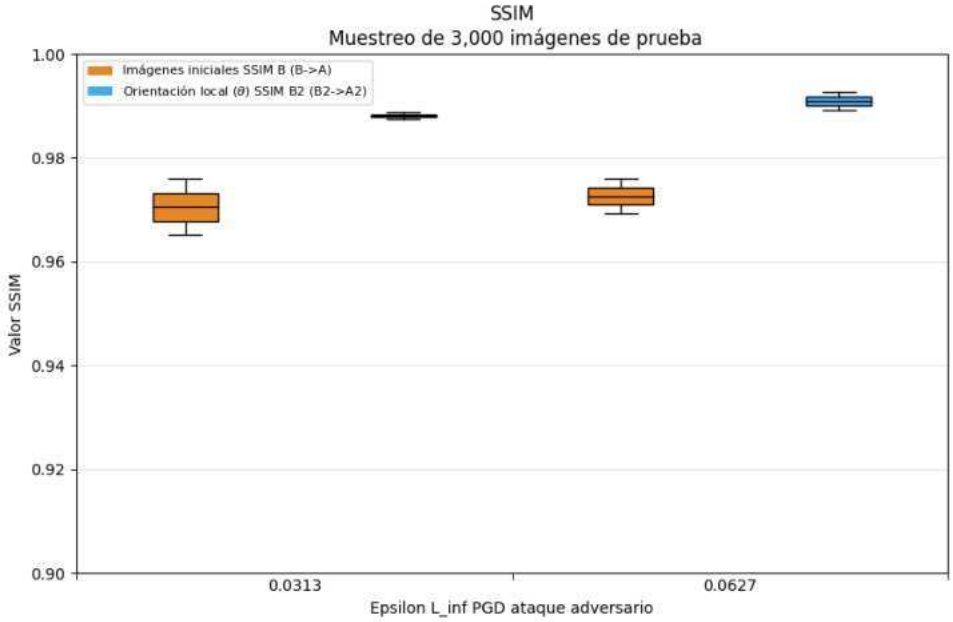

Figura 4.7: Gráfico de cajas de la métrica SSIM para 3,000 imágenes aleatorias y sin repetición con valores de espilon (0.0313, 0.0627) para el modelo CM_{ori} con un atacante de caja blanca.

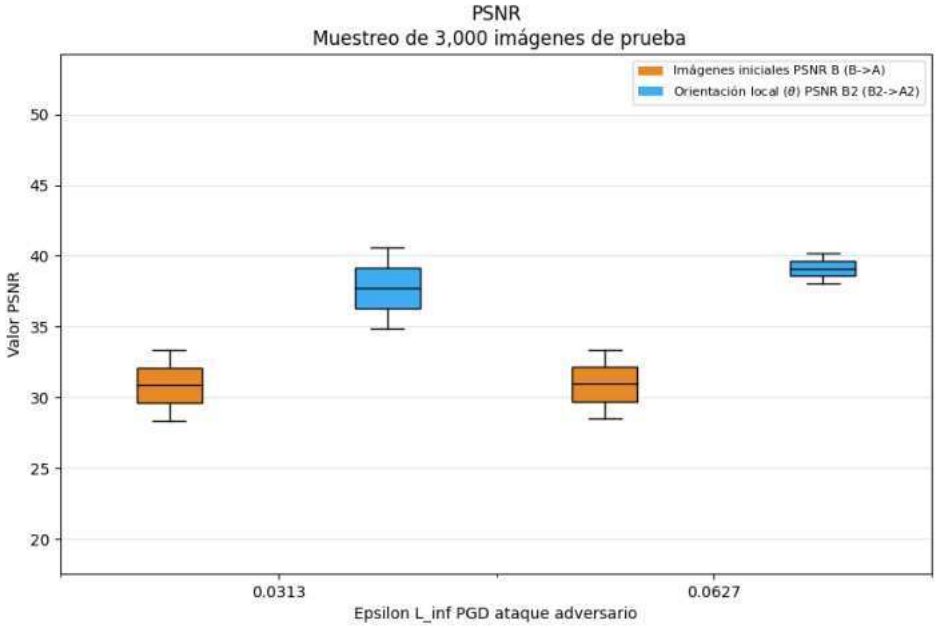

Figura 4.8: Gráfico de cajas de la métrica PSNR para 3,000 imágenes aleatorias y sin repetición con valores de espilon (0.0313, 0.0627) para el modelo CM_{ori} con un atacante de caja blanca.

El modelo CM_{ori} alcanzo 0.7495 de exactitud top-1 durante el entrenamiento, cuyo valor es menor al alcanzado por $Madry_R$ (0.8703), sin embargo, el desempeño del modelo CM_{ori} muestra poca afectación al incrementar los valores de epsilon (Figura 4.9).

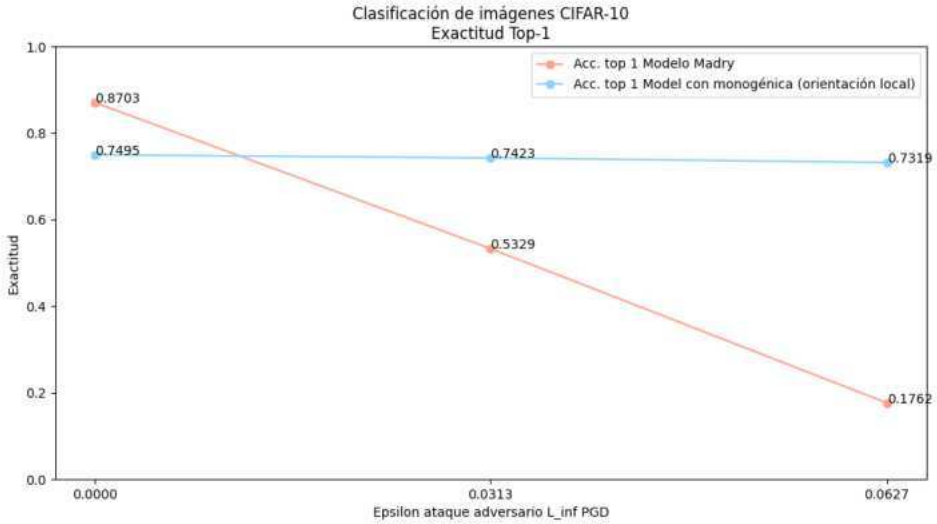

Figura 4.9: Clasificación de imágenes con la métrica exactitud top-1 para ambos modelos ($Madry_R$ y CM_{ori}) con un atacante de caja blanca.

Siguiendo el mismo proceso de comparación para el desempeño de los modelos, se obtuvo la tasa de decremento basado en la exactitud máxima alcanzada para cada uno de ellos. Los resultados corroboran la poca afectación al modelo CM_{ori} al incrementar los valores de epsilon (Figura 4.10).

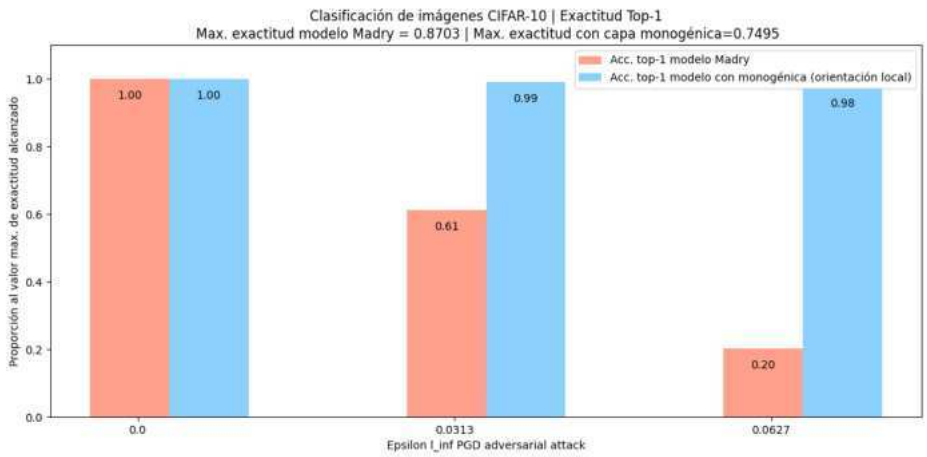

Figura 4.10: Tasa de decremento basado en la exactitud máxima alcanzada para ambos modelos ($Madry_R$ y CM_{ori}) con un atacante de caja blanca.

Resultados de la Capa Monogénica con la Fase Local como Salida (CM_{fase})

El gráfico de cajas (Figura 4.11) de la métrica SSIM muestra distribuciones más compactas y con valores de media mayores a los obtenidos por las imágenes iniciales.

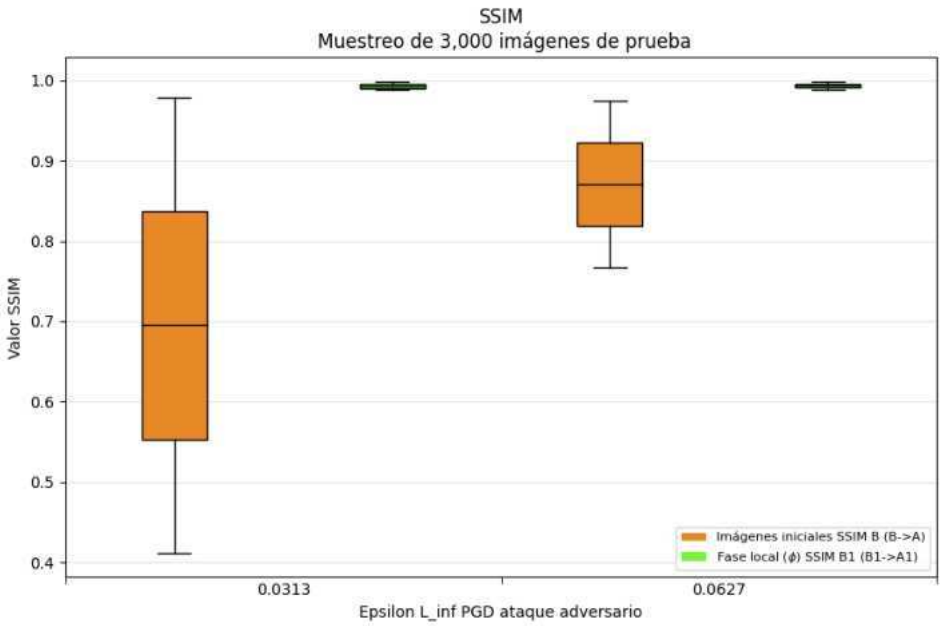

Figura 4.11: Gráfico de cajas de métrica SSIM para 3,000 imágenes aleatorias y sin repetición con valores de espilon (0.0313, 0.0627) para el modelo CM_{fase} con un atacante de caja blanca.

Por otra parte, el gráfico de cajas (Figura 4.12) obtenido de la métrica PSNR mantiene

la misma tendencia con distribuciones más dispersas, no obstante, se puede observar que los valores de media son mayores para las imágenes de la fase local que las correspondientes a las imágenes iniciales.

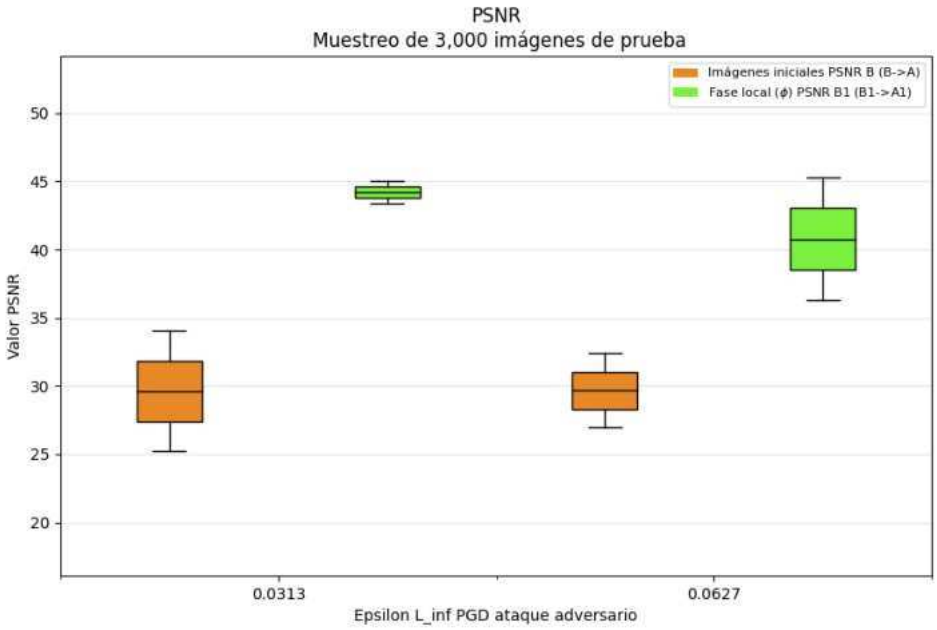

Figura 4.12: Gráfico de cajas de métrica PSNR para 3,000 imágenes aleatorias y sin repetición con valores de espilon (0.0313, 0.0627) para el modelo CM_{fase} con un atacante de caja blanca.

El modelo CM_{fase} alcanzo 0.7496 de exactitud durante el entrenamiento, valor más bajo al alcanzado por el modelo $Madry_R$ (0.8703). Sin embargo, el desempeño del modelo CM_{fase} muestra menor afectación al incrementar el valor de espilon que el modelo $Madry_R$ (Figura 4.13).

El gráfico de barras (Figura 4.14) muestra la tasa de decremento del desempeño para ambos modelos, y hace evidente la diferencia de afectación causada por el ataque adversario. Se puede observar que el modelo CM_{fase} manejo satisfactoriamente los ataques realizados.

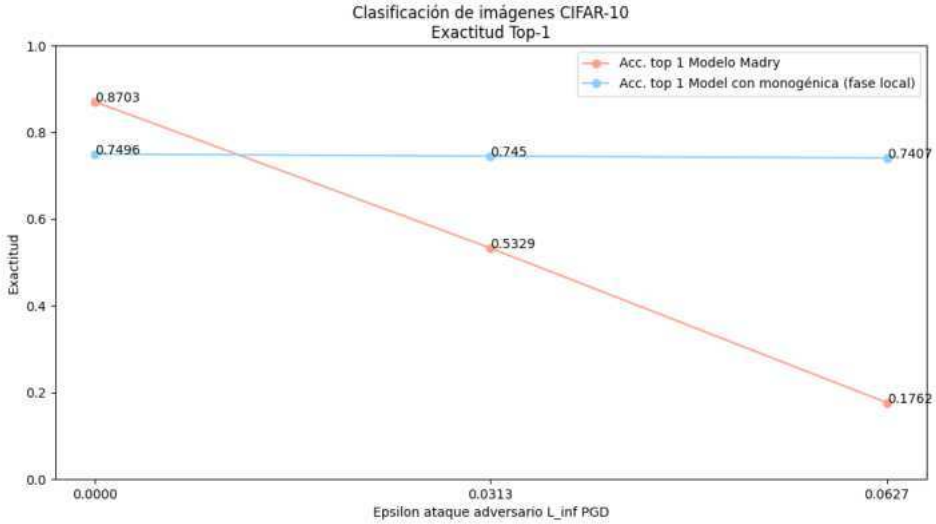

Figura 4.13: Clasificación de imágenes con la métrica exactitud top-1 para ambos modelos ($Madry_R$ y CM_{fase}) con un atacante de caja blanca.

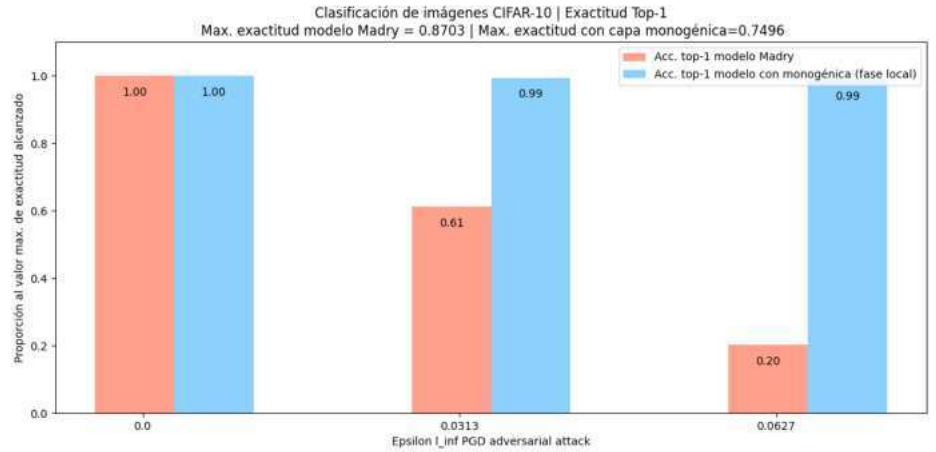

Figura 4.14: Tasa de decremento basado en la exactitud máxima alcanzada para ambos modelos ($Madry_R$ y CM_{fase}) con un atacante de caja blanca.

Resultados de la Capa Monogénica con la Media de la Fase y Orientación Local como Salida (CM_{media})

Por su parte, el modelo CM_{media} también mostró un buen desempeño confrontando el ataque adversario.

Los resultados obtenidos de la métrica SSIM muestran distribuciones más compactas y valores de media mayores a los obtenidos por las imágenes iniciales (Figura 4.15).

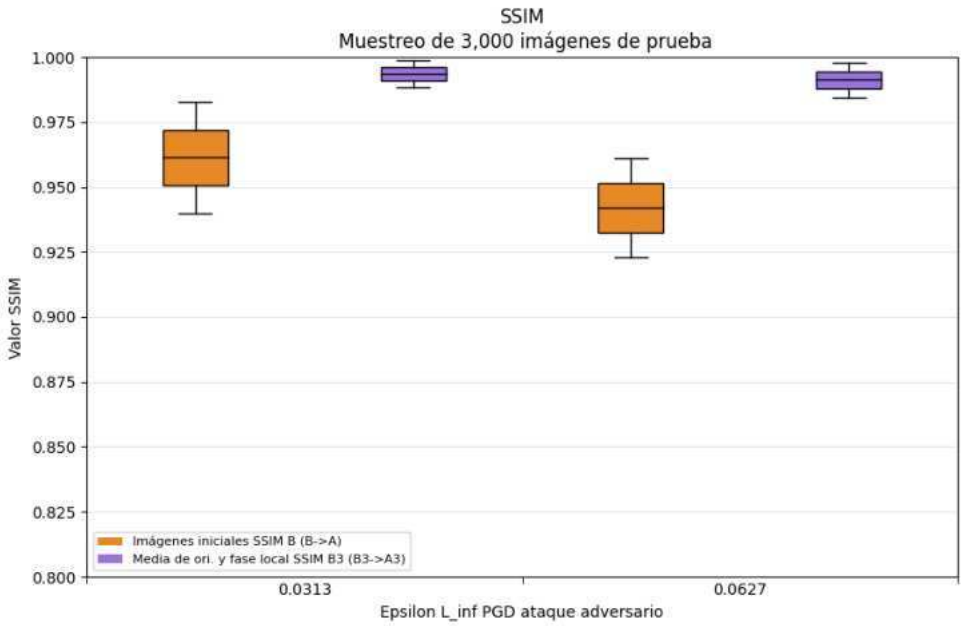

Figura 4.15: Gráfico de cajas de métrica SSIM para 3,000 imágenes aleatorias y sin repetición con valores de espilon (0.0313, 0.0627) para el modelo CM_{media} con un atacante de caja blanca.

Los gráficos de caja correspondientes a la métrica PSNR muestran distribuciones más dispersas para las imágenes obtenidas por la media de la fase y orientación local, sin embargo, los valores de media están por encima a los obtenidos por las imágenes iniciales (Figura 4.16).

Ambas métricas (SSIM y PSNR) corroboran el buen desempeño de la capa monogénica para procesar los ejemplos adversarios.

En cuanto a los resultados obtenidos para la tarea de clasificación se tiene un mejor desempeño para el modelo CM_{media} que el modelo $Madry_R$. El modelo $Madry_R$ aún mantiene una exactitud más alta, sin embargo, se ve más afectado por los ataques realizados (Figura 4.17).

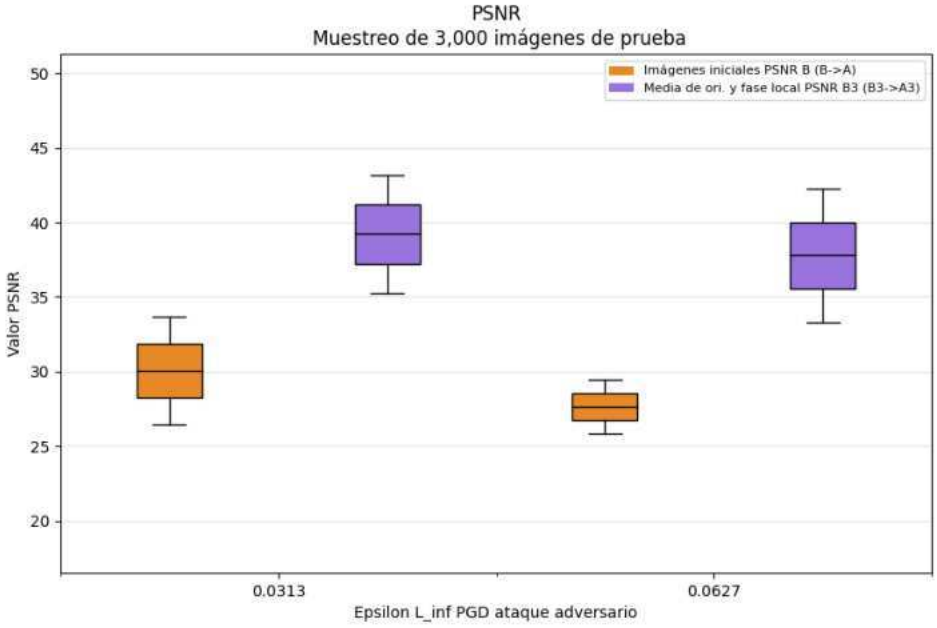

Figura 4.16: Gráfico de cajas de métrica PSNR para 3,000 imágenes aleatorias y sin repetición con valores de espilon (0.0313, 0.0627) para el modelo CM_{media} con un atacante de caja blanca.

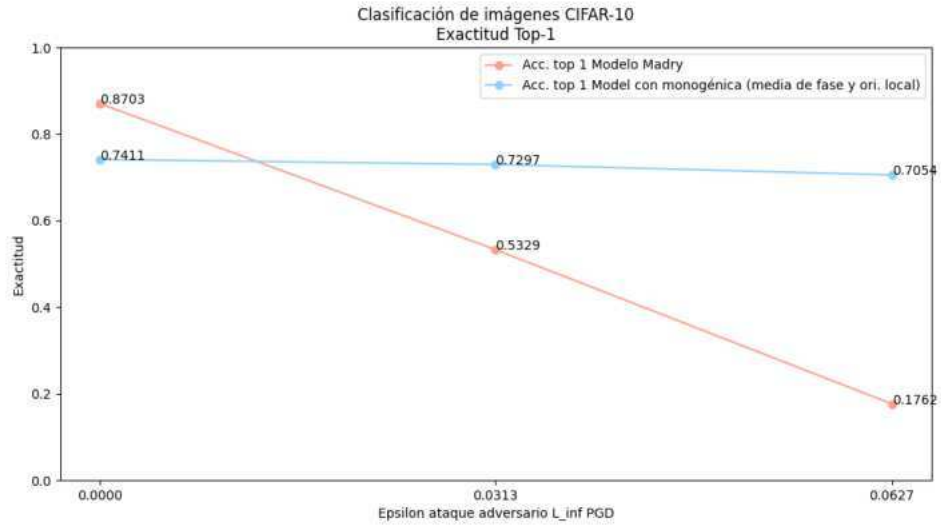

Figura 4.17: Clasificación de imágenes con la métrica de exactitud top-1 para ambos modelos ($Madry_R$ y CM_{media}).

Los valores obtenidos de la tasa de decremento corroboran el buen desempeño obtenido por el modelo CM_{media} para contrarrestar las afectaciones causadas por los ataques adversarios (Figura 4.18).

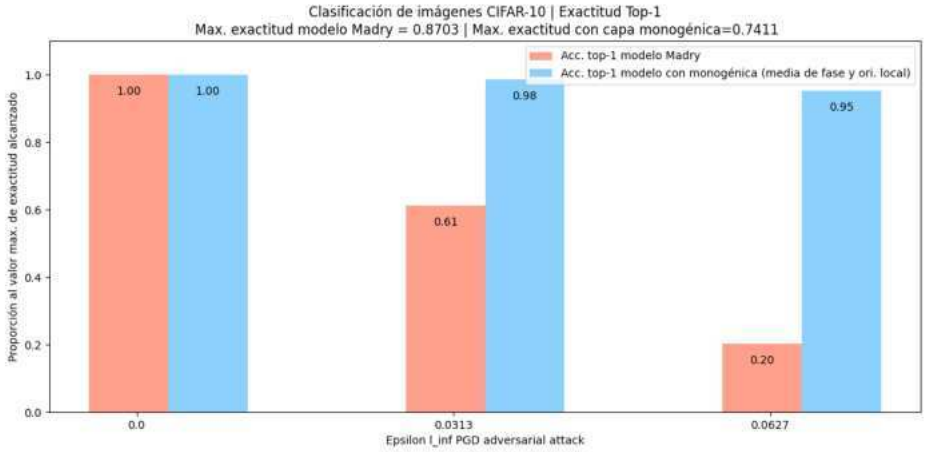

Figura 4.18: Tasa de decremento basado en la exactitud máxima alcanzada para ambos modelos ($Madry_R$ y CM_{media}) con un atacante de caja blanca.

4.2. Discusión

Las redes neuronales profundas son vulnerables a los ataques adversarios [22], desencadenando errores de predicción [57], en consecuencia, surge la necesidad de métodos de defensa para mantener la fiabilidad del uso de los modelos de inteligencia artificial para tareas de visión por computadora en la vida real.

Este trabajo de investigación llevo a cabo la experimentación y análisis de la novedosa capa monogénica [1, 2] como método de defensa contra un atacante PGD con métrica de distancia L_{inf} de caja gris y otro de caja blanca.

Los resultados obtenidos por las métricas de fidelidad de imágenes proporcionan una interpretación cuantitativa del desempeño de la capa monogénica para manejar las afectaciones de los ejemplos adversarios. Los resultados expuestos en los gráficos de caja de las Figuras [4.3, 4.4, 4.7, 4.8, 4.11, 4.12, 4.15, y 4.16] dan evidencia del exitoso procesamiento de los ejemplos adversarios para contrarrestar las afectaciones causada por el atacante.

Este trabajo de tesis incluye la comparativa del desempeño medido con la exactitud top-1 de la clasificación de las imágenes del modelo con la capa monogénica y otros dos modelos (ResNet-50 sin método de defensa y el modelo $Madry_R$ [33]).

Los resultados obtenidos para la primera comparativa 3.5.1 se muestran en el Cuadro 4.1, y los resultados obtenidos para la segunda comparativa 3.5.2 se muestran en el Cuadro 4.2. Los mejores resultados obtenidos para la clasificación son resaltados en negro.

Cuadro 4.1: Resultados de la comparativa del modelo ResNet-50 con una capa monogénica y el modelo ResNet-50 sin método de defensa contra un ataque PGD L_{inf} de caja gris.

Nombre del modelo	Método de defensa	Tipo de ataque	Epsilon/ Exactitud Top-1
ResNet-50	Sin método de defensa	PGD caja gris	ϵ (0.0) = **0.8674** ϵ (0.1) = **0.8618** ϵ (0.2) = **0.8304** ϵ (0.3) = **0.7778** ϵ (0.4) = 0.702 ϵ (0.5) = 0.6151 ϵ (0.6) = 0.5269 ϵ (0.7) = 0.4435 ϵ (0.8) = 0.3692 ϵ (0.9) = 0.3075 ϵ (1.0) = 0.2582
ResNet-50 con capa monogénica	Capa monogénica	PGD caja gris	ϵ (0.0) = 0.7641 ϵ (0.1) = 0.7631 ϵ (0.2) = 0.7598 ϵ (0.3) = 0.7483 ϵ (0.4) = **0.7346** ϵ (0.5) = **0.7137** ϵ (0.6) = **0.6905** ϵ (0.7) = **0.6677** ϵ (0.8) = **0.6402** ϵ (0.9) = **0.6037** ϵ (1.0) = **0.5723**

El análisis e interpretación de los resultados dan un desempeño favorable para los modelos con capa monogénica, sin embargo, aún existen áreas por mejorar para hacer más robusto este método.

La explicación al buen desempeño del modelo con capa monogénica para confrontar ataques adversarios podría atribuirse a que el atacante no tiene conocimiento del método de defensa, únicamente de la estructura y parámetros de entrenamiento. Por otra parte, el modelo $Madry_R$ hace uso de un método de defensa que no tiene otra opción más que incluir el método de defensa dentro del modelo, y hace más vulnerable al modelo ante los ataques adversarios.

Cuadro 4.2: Resultados de la comparativa del modelo $Madry_R$ y los tres modelos ResNet-50 con una capa monogénica (CM_{ori}, CM_{fase} y CM_{media}) contra un ataque PGD L_{inf} de caja blanca.

Nombre del modelo	Método de defensa	Tipo de ataque	Epsilon/ Exactitud Top-1
$Madry_R$ [33]	Entrenamiento robusto	PGD caja blanca	ϵ (0.0) = **0.8703** ϵ (0.0313) = 0.5329 ϵ (0.0627) = 0.1762
ResNet-50 con capa monogénica (CM_{ori})	Capa monogénica	PGD caja blanca	ϵ (0.0) = 0.7495 ϵ (0.0313) = 0.7423 ϵ (0.0627) = 0.7319
ResNet-50 con capa monogénica (CM_{fase})	Capa monogénica	PGD caja blanca	ϵ (0.0) = 0.7496 ϵ (0.0313) = **0.745** ϵ (0.0627) = **0.7407**
ResNet-50 con capa monogénica (CM_{media})	Capa monogénica	PGD caja blanca	ϵ (0.0) = 0.7411 ϵ (0.0313) = 0.7297 ϵ (0.0627) = 0.7054

4.3. Publicaciones

Durante el desarrollo de este trabajo de tesis se realizo la publicación del artículo [71], presentado en el congreso *Telematics and Computing WITCOM 2023*.

4.4. Trabajo futuro

Como trabajo futuro existen innumerable posibilidades para continuar con la investigación, sin embargo, se sugieren los siguietnes puntos:

- Incrementar la exactitud obtenida por el modelo con la capa monogénica.
- Experimentar con más conjuntos de datos.
- Someter la capa monogénica a un ataque adversario con método de caja blanca sugerido por [52], donde el atacante también tiene acceso al método de defensa.
- Realizar experimentación de la capa monogénica con parámetros entrenables [2].
- Explorar el uso de la capa monogénica en métodos de defensa como tipo optimización de modelo o adición de red.

CAPÍTULO 5

Conclusión

Este trabajo se enfoco en exponer la vulnerabilidad de la red neuronal ResNet-50 ante el ataque adversario PGD con métrica de distancia L_{inf}, así como también, llevar a cabo la comparativa del modelo presentado contra uno del estado del arte. El conjunto de datos CIFAR-10 fue utilizado para generar los ejemplos adversarios y la clasificación de imágenes. La propuesta expuesta en este trabajo para controlar las afectaciones causadas por un ataque adversario PGD L_{inf} es un método de defensa de tipo optimización de datos que procesa las imágenes previo al modelo DNN mediante el uso de una capa monogénica.

Los resultados obtenidos de esta investigación exponen el potencial desempeño de la capa monogénica para confrontar ejemplos adversarios, añadiendo robustez a la red neuronal profunda y mejorando o equiparando su desempeño ante otros métodos de defensa. Aunque no se ha encontrado un método de defensa completamente eficiente para contrarrestar las afectaciones causadas por los ataques adversarios de forma efectiva, este trabajo muestra el potencial uso de la señal monogénica para propósitos de defensa.

Los resultados favorable que se han obtenido de este trabajo, en conjunto con los obtenidos en [1, 2], son evidencia para alentar la continuidad de la investigación sobre el uso de la señal monogénica para propósitos de Inteligencia Artificial.

Bibliografía

[1] E. U. Moya-Sánchez, S. Xambó-Descamps, S. Salazar Colores, A. Sánchez Pérez, and U. Cortés, "A quaternion deterministic monogenic cnn layer for contrast invariance," in *Systems, Patterns and Data Engineering with Geometric Calculi* (S. Xambó-Descamps, ed.), (Cham), pp. 133–152, Springer International Publishing, 2021.

[2] E. U. Moya-Sanchez, S. Xambo-Descamps, A. S. Perez, S. Salazar-Colores, and U. Cortes, "A trainable monogenic ConvNet layer robust in front of large contrast changes in image classification," *IEEE Access*, vol. 9, pp. 163735–163746, 2021.

[3] E. U. Moya-Sánchez and E. Bayro-Corrochano, "Quaternion atomic function wavelet for applications in image processing," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pp. 346–353, Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[4] E. Ulises Moya-Sánchez and E. Bayro-Corrochano, "Symmetry feature extraction based on quaternionic local phase," *Adv. Appl. Clifford Algebras*, vol. 24, no. 2, pp. 333–354, 2014.

[5] E. U. Moya-Sánchez, S. Colores, A. Sánchez, and U. Cortés, "Quaternion phase cnn," in *Artificial Intelligence Research and Development*, pp. 89–95, IOS Press, 2018.

[6] E. U. Moya-Sanchez, S. Salazar-Colores, A. Sánchez, U. Cortés, and S. Xambó, "Deep learning monogenic cnn," in *Proc. Applied Geometric Algebras in Computer Science and Engineering*, pp. 147–154, 07 2018.

[7] F. Chollet, *Deep learning with python*. New York, NY: Manning Publications, 2017.

[8] C. C. Aggarwal, *Neural networks and deep learning: A textbook*. Cham, Switzerland: Springer International Publishing, 1 ed., 2018.

[9] M. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 1 ed., 2015.

[10] S. Pattanayak, *Pro deep learning with TensorFlow: A mathematical approach to advanced artificial intelligence in python*. New York, NY: APRESS, 1 ed., 2017.

[11] C. P. Bridge, "Introduction to the monogenic signal," *CoRR*, vol. abs/1703.09199, 2017.

[12] H. Liang, E. He, Y. Zhao, Z. Jia, and H. Li, "Adversarial attack and defense: A survey," *Electronics*, vol. 11, no. 8, 2022.

[13] Y. Bengio, *Learning Deep Architectures for AI*, vol. 2. Dept. IRO, Université de Montréal, Canada: Now Publishers Inc., jan 2009.

[14] M. Tan and Q. V. Le, "EfficientNetV2: Smaller models and faster training," 2021.

[15] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby, "Big transfer (bit): General visual representation learning," in *Computer Vision – ECCV 2020* (A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds.), (Cham), pp. 491–507, Springer International Publishing, 2020.

[16] T. Ridnik, H. Lawen, A. Noy, E. Ben, B. G. Sharir, and I. Friedman, "Tresnet: High performance gpu-dedicated architecture," in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1399–1408, 2021.

[17] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[18] H. Song, M. Kim, and J.-G. Lee, "SELFIE: Refurbishing unclean samples for robust deep learning," in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 5907–5915, PMLR, 09–15 Jun 2019.

[19] D. Amarathunga, M. N. Ratnayake, J. Grundy, and A. Dorin, "Image dataset of two morphologically close thrip species: Western flower thrips and plague thrips," Nov. 2022.

[20] A. Krizhevsky, "Learning multiple layers of features from tiny images," pp. 32–33, 2009.

[21] R. S. Lee, F. Gimenez, A. Hoogi, K. K. Miyake, M. Gorovoy, and D. L. Rubin, "A curated mammography data set for use in computer-aided detection and diagnosis research," *Sci. Data*, vol. 4, p. 170177, 2017.

[22] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2019.

[23] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *2018 IEEE Symposium on Security and Privacy (SP)*, pp. 19–35, 2018.

[24] 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, 2016.

[25] J. Folz, S. Palacio, J. Hees, and A. Dengel, "Adversarial defense based on structure-to-signal autoencoders," in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 3568–3577, 2020.

[26] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided denoiser," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1778–1787, 2018.

[27] H. Kannan, A. Kurakin, and I. Goodfellow, "Adversarial logit pairing," 2018.

[28] X. Jia, X. Wei, X. Cao, and H. Foroosh, "Comdefend: An efficient image compression model to defend adversarial examples," pp. 6077–6085, 2019.

[29] C. Guo, M. Rana, M. Cissé, and L. van der Maaten, "Countering adversarial images using input transformations," *CoRR*, vol. abs/1711.00117, 2017.

[30] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," 2018.

[31] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *J. Physiol.*, vol. 160, no. 1, pp. 106–154, 1962.

[32] J. Rauber, R. Zimmermann, M. Bethge, and W. Brendel, "Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax," *Journal of Open Source Software*, vol. 5, no. 53, p. 2607, 2020.

[33] L. Engstrom, A. Ilyas, S. Santurkar, and D. Tsipras, "Robustness (python library)," 2019.

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

[35] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial attacks and defences: A survey," *CoRR*, vol. abs/1810.00069, 2018.

[36] S. Russell and P. Norvig, *Artificial intelligence: A modern approach*. Upper Saddle River, NJ, Estados Unidos de América: Pearson, 3 ed., 2009.

[37] F. Rosenblatt, *The Perceptron, a Perceiving and Recognizing Automaton*. Report: Cornell Aeronautical Laboratory, Cornell Aeronautical Laboratory, 1957.

[38] A. L. Fradkov, "Early history of machine learning," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 1385–1390, 2020.

[39] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[40] Y. Lecun, *Generalization and network design strategies*. Elsevier, 1989.

[41] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[42] M. Felsberg and G. Sommer, "The monogenic signal," *IEEE Trans. Signal Process.*, vol. 49, no. 12, pp. 3136–3144, 2001.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[44] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv [cs.CV]*, 2013.

[45] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015.

[46] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," pp. 39–57, 2017.

[47] X. Mao, Y. Chen, Y. Li, Y. He, and H. Xue, "GAP++: learning to generate target-conditioned adversarial examples," *CoRR*, vol. abs/2006.05097, 2020.

[48] M. Duan, K. Li, J. Deng, B. Xiao, and Q. Tian, "A novel multi-sample generation method for adversarial attacks," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 18, mar 2022.

[49] T. Wu, L. Tong, and Y. Vorobeychik, "Defending against physically realizable attacks on image classification," *CoRR*, vol. abs/1909.09552, 2019.

[50] J. Dapello, T. Marques, M. Schrimpf, F. Geiger, D. D. Cox, and J. J. DiCarlo, "Simulating a primary visual cortex at the front of cnns improves robustness to image perturbations," *bioRxiv*, 2020.

[51] A. Ma, F. Faghri, and A. Farahmand, "Adversarial robustness through regularization: A second-order approach," *CoRR*, vol. abs/2004.01832, 2020.

[52] A. Abusnaina, Y. Wu, S. Arora, Y. Wang, F. Wang, H. Yang, and D. Mohaisen, "Adversarial example detection using latent neighborhood graph," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7667–7676, 2021.

[53] M. Nicolae, M. Sinn, T. N. Minh, A. Rawat, M. Wistuba, V. Zantedeschi, I. M. Molloy, and B. Edwards, "Adversarial robustness toolbox v0.2.2," *CoRR*, vol. abs/1807.01069, 2018.

[54] S. Prince, *Computer Vision: Models Learning and Inference*. Cambridge University Press, 2012.

[55] G. H. Granlund and H. Knutsson, *Signal Processing for Computer Vision*. New York, NY: Springer, 2010.

[56] C. P. Bridge, "Introduction to the monogenic signal," 2017.

[57] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "A survey on adversarial attacks and defences," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 1, pp. 25–45, 2021.

[58] 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.

[59] E. Rusak, L. Schott, R. S. Zimmermann, J. Bitterwolf, O. Bringmann, M. Bethge, and W. Brendel, "A simple way to make neural networks robust against diverse image corruptions," 2020.

[60] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 9185–9193, IEEE Computer Society, jun 2018.

[61] T. Zheng, C. Chen, and K. Ren, "Distributionally adversarial attack," *CoRR*, vol. abs/1808.05537, 2018.

[62] G. Cohen, G. Sapiro, and R. Giryes, "Detecting adversarial samples using influence functions and nearest neighbors," *CoRR*, vol. abs/1909.06872, 2019.

[63] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," *CoRR*, vol. abs/1705.09064, 2017.

[64] N. S. Detlefsen, J. Borovec, J. Schock, A. Harsh, T. Koker, L. D. Liello, D. Stancl, C. Quan, M. Grechkin, and W. Falcon, "Torchmetrics - measuring reproducibility in pytorch," 2022.

[65] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[66] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? a new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, 2009.

[67] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and scikit-image contributors, "scikit-image: image processing in python," *PeerJ*, vol. 2, p. e453, 2014.

[68] S. Addepalli, B. S. Vivek, A. Baburaj, G. Sriramanan, and R. Venkatesh Babu, "Towards achieving adversarial robustness by enforcing feature consistency across bit planes," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1017–1026, 2020.

[69] G. Jin, X. Yi, W. Huang, S. Schewe, and X. Huang, "Enhancing adversarial training with second-order statistics of weights," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15252–15262, 2022.

[70] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, 2011.

[71] O. Garcia-Porras, S. Salazar-Colores, E. U. Moya-Sánchez, and A. Sánchez-Pérez, "Adversarial attack versus a bio-inspired defensive method for image classification," in *Telematics and Computing* (M. F. Mata-Rivera, R. Zagal-Flores, and C. Barria-Huidobro, eds.), (Cham), pp. 533–547, Springer Nature Switzerland, 2023.

Anexos

Carta de aceptación para la publicación del artículo [71].

ACCEPTANCE LETTER

Conferences and Workshops in Telematics and Computing
WITCOM 2023

Dear
Oscar Garcia Porras, Sebastían Salazar Colores, E. Ulises Moya Sánchez and Abraham Sánchez Pérez
Paper ID: 9817
Title: Adversarial attack versus a bio-inspired defensive method for image classification

We are delighted to inform you that the paper referenced above has been accepted for presential presentation at WITCOM 2023, subject to fulfilling the requirements listed below.

Congratulations on your acceptance!

Please carefully review this email as it contains important information regarding the inclusion of your paper in the Proceedings of WITCOM 2023, which will be published in the Springer CCIS Series of Journals (https://www.springer.com/series/7899)

Yours faithfully

Félix Mata

Chair WITCOM conferences 2023

Portada del artículo [71] publicado durante la investigación realizada.

Adversarial Attack Versus a Bio-inspired Defensive Method for Image Classification *

Oscar Garcia-Porras[1][0009−0003−1095−9303], Sebastián Salazar-Colores[2][0000−0002−6353−0864], E. Ulises Moya-Sánchez[3][0000−0002−5123−4881], and Abraham Sánchez-Pérez[3]

[1] Universidad Autónoma de Querétaro, Santiago de Querétaro, Querétaro, México, ogarcia41@alumnos.uaq.mx
[2] Centro de Investigaciones en Óptica, León, Guanajuato, México sebastian.salazar@cio.mx
[3] Dirección de Inteligencia Artificial del Gobierno de Jalisco, Guadalajara, Jalisco, Mexico {eduardo.moya,abraham.sanchez}@jalisco.gob.mx

Abstract. Adversarial attacks are designed to disrupt the information supplied to Artificial Intelligence (AI) models causing a failure in their intended purpose. Despite late state-of-art performance on computer vision tasks like image classification, AI models can be vulnerable to adversarial attacks. Hence, to mitigate this risk the AI models require additional steps. This research presents the implementation of a novel bio-inspired defense method against adversarial attacks. The defense is based on the use of a deterministic monogenic layer at the top of a ResNet-50 convolutional neural network model used for image classification with the CIFAR-10 data set. The results show the ConvNet with the bio-inspired defense approach outpacing the regular convolutional architectures, this can be translated into increased resistance to adversarial attacks and greater reliability of artificial intelligence models. The Structural Similarity Index Measure (SSIM) and the Peak Signal-to-Noise Ratio (PSNR) metrics were measured on the bio-inspired layer activations to obtain a quantitative explanation of the improvement. The results obtained from this research expose the potential performance of the monogenic layer in confronting adversarial attacks and encourage further expansion of the knowledge in the field of the monogenic signal for AI purposes.

Keywords: Adversarial attacks · Deep neural networks · Image classification · Monogenic signal

1 Introduction

Computer vision intends to similarly accomplish the ability to see and understand visual information as humans, this task for computer vision requires the use of methods and techniques to analyze, understand, and extract relevant information from the visualized data. Deep neural networks (DNN) and specifically

* Supported by the Mexican governmental institute Consejo Nacional de Humanidades Ciencias y Tecnologías (CONAHCyT)