UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

FACULTAD DE INGENIERÍA

Algoritmo para la detección de vehículos y peatones combinando CNN´s y técnicas de búsqueda.

TESIS

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

Presenta
Ing. Gerardo Treviño Valdés

Dirigido por:
Dr. Jesús Carlos Pedraza Ortega

Co-dirigido por:
Dr. Jesús Alejandro Navarro Acosta

Querétaro, Qro. a 7 de junio de 2022

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE INGENIERÍA
MAESTRÍA EN CIENCIAS EN INTELIGENCIA ARTIFICIAL

Algoritmo para la detección de vehículos y peatones combinando CNN´s y técnicas de búsqueda.

TESIS

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

Presenta
Gerardo Treviño Valdés

Dirigido por:
Jesús Carlos Pedraza Ortega

Co-dirigido por:
Jesús Alejandro Navarro Acosta

Dr. Jesús Carlos Pedraza Ortega
Presidente

Dr. Jesús Alejandro Navarro Acosta
Secretario

Dr. Saúl Tovar Arriaga
Vocal

Dr. Marco Antonio Aceves Fernández
Suplente

Dr. Juan Manuel Ramos Arreguín
Suplente

Centro Universitario, Querétaro, Qro.
7 de junio de 2022
México

A mis padres por su apoyo incondicional.

Agradecimientos

A mi director y codirector de tesis por guiarme a lo largo de todo el programa y compartirme sus experiencias profesionales y personales que me han servido en mi desarrollo profesional.

A mis profesores y sinodales por su dedicación al compartir su conocimiento.

A la Universidad Autónoma de Querétaro y la Facultad de Ingeniería.

Al Consejo Nacional de Ciencia y Tecnología por apoyar con el financiamiento del programa de posgrado.

RESUMEN

El reconocimiento de objetos es una de las tareas básicas de la visión por ordenador. Es el proceso de localizar o identificar objetos en imágenes o vídeo. Los métodos de reconocimiento de objetos suelen utilizar características extraídas y algoritmos de aprendizaje para identificar instancias de objetos o imágenes que pertenecen a la clase de objeto. El objetivo del reconocimiento de objetos es identificar un objeto en una clase o categoría determinada, mientras que el objetivo de la detección de objetos es encontrar el objeto de interés en imágenes o vídeos.

En esta tesis se estudia la combinación de dos técnicas de inteligencia artificial para la detección de objetos en imágenes de carretera: las redes neuronales convolucionales y los algoritmos de búsqueda. Se pretende mejorar la identificación en el algoritmo neuronal implementando un algoritmo meta heurístico y de esta manera crear un algoritmo híbrido de estas técnicas.

El análisis de las redes neuronales convolucionales y los diferentes algoritmos metaheurísticos es un paso importante para determinar como podría combinarse estos dos métodos. El ajuste de los parámetros e hiperparámetros de una red neuronal juega un papel muy importante en la evaluación de la misma.

A través de las pruebas realizadas se determina que al emplear los algoritmos metaheurísticos para un ajuste fino de los parámetros en la red neuronal YOLOv3 podremos obtener una mejoría en la detección de vehículos, este algoritmo mejora esta detección en diversas clases y en la precisión media global.

ABSTRACT

Object detection or recognition is one of the basic tasks of computer vision. It is the process of locating or identifying objects in images or video. Object recognition methods typically use extracted features and learning algorithms to identify instances of objects or images that belong to the object class. The goal of object recognition is to identify an object in a given class or category, while the goal of object detection is to find the object of interest in images or videos.

In this thesis we study the combination of two artificial intelligence techniques for object detection in road images: convolutional neural networks and search algorithms. The aim is to improve the identification in the neural algorithm by implementing a meta heuristic algorithm and thus create a hybrid algorithm of these techniques.

The analysis of the convolutional neural networks and the different metaheuristic algorithms is an important step to determine how these two methods could be combined. The adjustment of the parameters and hyperparameters of a neural network plays a very important role in the evaluation of the neural network.

Through the tests performed it is determined that by using the metaheuristic algorithms for fine tuning the hyperparameters in the YOLOv3 neural network we can obtain an improvement in the detection of vehicles, this algorithm improves this detection in various classes and also in mean average precision.

ÍNDICE

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

1.1. Introducción

Las redes neuronales convolucionales (por sus siglas en inglés CNN) han demostrado un rendimiento en el procesamiento de imágenes al aumentar su desempeño en tareas como clasificación, detección de objetos, entre otras. Así como la capacidad de adaptación de sus modelos [23].

Los métodos de detección, clasificación e identificación de objetos que se fundamentan en la extracción de características han tomado un auge en el campo de la inteligencia artificial, y no es por menos ya que han sido muy beneficiosos e interesantes [12]. Estos métodos se pueden aplicar a diferentes tareas y adaptarse a entornos nunca antes vistos basados en las características adquiridas previamente, para esto se tiene que pasar por un debido ajuste en sus parámetros y un correcto entrenamiento [47].

Las redes neuronales convolucionales, en los últimos años, han propiciado un importante avance en tareas que involucran visión artificial, tales como clasificación, localización, detección y segmentación de objetos, descripción de escenas, entre otras, ya sea en imágenes o vídeo. Los resultados que se obtienen actualmente se puedan emplear en una gran variedad de aplicaciones.

Sin embargo, el desempeño de algoritmos como las CNN en la detección de objetos depende en gran medida de la elección y ajuste de diversos parámetros que determinan la funcionalidad de estas, por tal motivo en esta tesis se presentan dos técnicas de inteligencia artificial como lo son los algoritmos metaheurísticos y las redes neuronales convolucionales para realizar un entrenamiento más robusto y fino en comparación del entrenamiento estándar de estas redes [40].

1.2. Justificación

El reconocimiento de objetos es una de las tareas fundamentales en la visión por computadora. Es el proceso de encontrar o identificar instancias de objetos (por ejemplo, caras, perros o edificios) en imágenes digitales o vídeos. Los métodos de reconocimiento de objetos utilizan con frecuencia características extraídas y algoritmos de aprendizaje para reconocer instancias de un objeto o imágenes que pertenecen a una categoría de objeto. El reconocimiento de objetos tiene como objetivo identificarlo en una determinada clase o categoría, mientras que la detección de objetos tiene como objetivo localizar un objeto de interés en imágenes digitales o vídeos. Cada "objeto" o "clase" tiene sus propias características particulares que se reconocen y las diferencian del resto, ayudando en el reconocimiento de los mismos objetos o similares en otras muestras. El reconocimiento de objetos se aplica en muchas áreas de la visión por computadora, incluyendo recuperación de imágenes, seguridad, vigilancia, sistemas automáticos de vehículos y maquinaria industrial. En este documento abordamos dos tareas de reconocimiento de objetos [25].

- Clasificación: Dada una entrada de imagen, decida cuál de las múltiples posibles categorías está presente.

- Detección y localización: Dada una imagen con distintos objetos, decida si un objeto específico de interés se encuentra en algún lugar de esta imagen y proporciona una ubicación precisa información sobre el objeto.

Hoy la complejidad y sofisticación de los procesos, demanda estrategias para la detección de objetos de forma asertiva, minimizando los errores de identificación, ya que en algunos escenarios se pueden dar tareas muy puntuales donde se requiere de alta precisión en identificar vehículos y peatones. Por lo tanto, tener sistemas seguros y confiables es primordial en dicha área, donde se implementan sistemas para prevenir accidentes mediante una detección eficaz de automóviles, ciclistas y peatones.

1.3. Planteamiento del problema

En los últimos años, las redes neuronales convolucionales han logrado importantes avances en tareas relacionadas con la visión por ordenador, como clasificación, localización, detección y segmentación de objetos, descripción de escenas y otras tareas de visión artificial en imágenes y vídeo. Los resultados obtenidos hoy pueden utilizarse en muchos ámbitos diferentes y pueden utilizarse en una variedad de aplicaciones en las que estas tareas son hacen que el problema está sustancialmente resuelto.

Aunque la CNN ha demostrado ser adecuada para estos problemas, tiene la desventaja de que requiere una unidad aritmética potente debido al gran número de operaciones de multiplicación y acumulación.

Esto también fue una limitación para los investigadores en el campo en la década de 1990. Por ello, para resolver esta dificultad, decidieron implementar el algoritmo con respecto al hardware adecuado y no el software, que actualmente seguimos con esta limitación a priori. Con el desarrollo de dispositivos como GPUs (unidades de procesamiento gráfico), CPUs multinúcleo y otros dispositivos como el procesamiento de alto rendimiento mediante clusters, se ha hecho posible implementar y probar estos algoritmos.

Sin embargo, el desempeño de algoritmos como las CNN en la detección de objetos depende en gran medida de la elección y ajuste de diversos hiper parámetros que pueden determinar la tasa de aprendizaje de las mismas, por tal motivo en este trabajo de investigación se busca abordar el problema de aumentar el desempeño de una CNN para la detección de objetos mediante el hibridismo de algoritmos con el fin de lograr un mejor resultado de la CNN, logrando así un entrenamiento más robusto en comparación con los algoritmos de entrenamiento estándar.

La implementación de algoritmos como CNN y la mejora de algoritmos existentes representa un reto en el campo de la IA, que al reducir los esfuerzos de cómputo da pie a la inclusión de diferentes técnicas, como son los algoritmos híbridos, a fin de mejorar la resolución de la red, posibilitando aplicaciones en las que por su naturaleza se proyectan importantes desarrollos.

1.4. Hipótesis

Mediante la implementación de algoritmos de búsqueda para el ajuste de los hiper parámetros en el proceso de entrenamiento de las CNN´s, se mejorará su porcentaje de detección en imágenes de vehículos y/o peatones.

1.5. Objetivos

1.5.1 Objetivo General

Desarrollar e implementar un algoritmo de aprendizaje profundo mediante la combinación de diversas técnicas de visión por computadora, para la detección de vehículos y peatones en imágenes de carretera mejorando su identificación en las CNN estándar.

1.5.2 Objetivos Específicos

- Definir una arquitectura CNN apropiada para la extracción de características en la etapa de entrenamiento.
- Definir un algoritmo de optimización para el ajuste en los hiper parámetros en la red neuronal en la etapa de entrenamiento.
- Desarrollo e implementación de un algoritmo de aprendizaje profundo para la detección de vehículos implementando la combinación de estas técnicas.

1.6. Estructura de la tesis

En el presente proyecto de tesis se estudian diversos temas relacionados con inteligencia artificial, vision por computadora, aprendizaje profundo y algoritmos de busqueda para su realizacion, ademas de distintas pruebas de estos topicos para llegar a cumplir los objetivos de la misma. La estructuracion de la tesis se plantea de la siguiente forma:

- En el segundo capitulo se abordan los antecedentes de esta para tomar un poco de referencia de la historia en la inteligencia artificial llegando hasta el estado del arte donde exponemos el estado actual de nuestro proyecto de tesis donde encontramos diversas arquitecturas funcionales y competentes para ayudar en la realizacion del proyecto.

- En el tercer capitulo nos enfocamos en los fundamentos que nos permitien la realisacion del proyecto de tesis, donde exponemos temas primordiales como: redes neuronales, redes neuronales convolutivas, algoritmos de busqueda, inteligencia de enjambre/-manada.

- El cuarto capitulo se abordan los metodos y materiales utilizados para el desarrollo de esta, la metodologia y algoritmos primoridales utilizados para la realizacion de pruebas.

- Proseguimos con el penultimo capitulo, el quitno capitulo, se muestran los resultados de la experimentacion, como el entrenamiento de las redes, exploracion de los parametros a través de los algoritmos de búsqueda y donde discutimos su comparación.

- Por ultimo, en el sexto capitulo, se exponen las conclusiones del proyecto de tesis, así como los proyectos futuros que podrían aprovecharse de este trabajo.

2. ANTECEDENTES

Hoy en día los avances en el mundo de la tecnología avanzan con pasos agigantados, estamos en una era de constante cambio para cualquier industria que quiera prevalecer en vanguardia, tareas que para principios del milenio eran vistas en solo ciencia ficción hoy son reales. Uno de los ejemplos es ver cómo la inteligencia artificial (IA) ha llegado para quedarse en nuestro entorno. tal que las aplicaciones en este campo son muy extensas y aún nos queda mucho trabajo por realizar.

Las tecnologías de visión por computadora una de las áreas dentro de la IA ha evolucionado rápidamente en la última década. Los primeros sistemas funcionales de visión por computadora se basaron en imágenes binarias (en blanco y negro) procesadas en bloques, ventanas o píxeles. El siguiente paso en el desarrollo de la visión por computadora fue la introducción de sistemas de densidad gris. Con esta técnica, cada elemento de la imagen o píxel se representa mediante un número que es proporcional a la intensidad del color gris del elemento. La característica principal de esta técnica es la corrección de las fluctuaciones de la iluminación local. Los sistemas de sonido gris se pueden usar en cualquier tipo de iluminación, ya que pueden encontrar contornos de objetos buscando cambios en los valores de densidad de píxeles. Los sistemas avanzados de visión por computadora funcionan con estructuras, no con píxeles. Estos sistemas requieren potentes procesadores de imágenes para manejar grandes entradas de datos y calidad digital [4] .

2.1. Inicio Redes Neuronales Convolucionales (CNN)

Las arquitecturas clásicas para las tareas de clasificación y detección se han actualizado significativamente desde 2012, especialmente en AlexNet. [23] para las tareas de clasificación. Se basa en una red neuronal convolucional (CNN) que consta de cinco capas

convolucionales y tres capas totalmente conectadas. Desde entonces, varios estudios han seguido explorando el uso de las CNN en tareas de clasificación, incluyendo las redes conocidas como ZF [48], VGG [39] y ResNet [19].

Con la llegada de los sistemas de clasificación basados en CNN, también se han desarrollado sistemas de detección, así como enfoques específicos de dominio basados en la idea de que cada objeto de interés en una imagen tiene características que lo distinguen. Por lo tanto, no es necesario buscar en toda la imagen, como en el paradigma de la ventana deslizante.

Se presentan tres sistemas de detección de gran éxito, que utilizan en mayor o menor medida la idea de la señalización regional.

1. R-CNN: Este sistema usa un método de región específica como *selective search* para generar muestras que se utilizan en el entrenamiento de las capas completamente conectadas de una red convolucional [42]. Generó un aumento de 30 por ciento en el mapeo con respecto a los mejores métodos de detección de VOC2012 [7]. Demora 13 s/imagen en GPU y 52 s/imagen en CPU en la etapa de prueba.

2. Fast R-CNN: Al igual que el sistema anterior usa *region proposal* y CNN, sin embargo, solamente la imagen completa pasa por las capas convolucionales y no cada región especifica cómo se hace en R-CNN, producto de esto se logra una considerable disminución en la detección de objetos en una imagen. Los *proposals* se utilizan para identificar regiones de interés sobre los mapas de características de salida desde la última capa convolucional, estas regiones son en las que se concentran las capas completamente conectadas. Demora 0.3 s/imagen en GPU [16].

3. Faster R-CNN: Este sistema de detección se diferencia de los dos anteriores, ya que, utiliza la misma red convolucional para generar los *proposals*, localizar y clasificar, por lo que, se puede entrenar de extremo a extremo sin depender de ningún método externo. Toma en total 198 ms/imagen en GPU [35].

2.2. Implementación de algoritmos de búsqueda

Los problemas del mundo real, modelados como problemas de optimización, constan de tres partes: variables que necesitan optimización, una o más funciones objetivas que deben ser maximizadas o minimizadas, y una serie de restricciones y condiciones que limitan dichas funciones [32]. Por lo tanto, el uso de metaheurísticas para buscar las mejores soluciones es una de las alternativas más destacadas. En general, las metaheurísticas se definen como procedimientos de búsqueda para una solución casi óptima dentro de un espacio de solución para un problema de optimización. Los factores que afectan la calidad de los resultados de búsqueda son el tipo de problema, el algoritmo metaheurístico aplicado, el tamaño del espacio de búsqueda y la capacidad computacional. La reputación actual del aprendizaje profundo se debe implícitamente a mejorar drásticamente las habilidades de procesamiento de chips, disminuye significativamente el costo del hardware e investigación avanzada en aprendizaje automático y procesamiento de señales [8]. En general, los modelos de aprendizaje profundo pueden clasificarse en modelos discriminativos, generativos e híbridos [8]. Los modelos discriminatorios, por ejemplo, son CNN, aprendizaje profundo y red neuronal recurrente. Algunos ejemplos de los modelos genéticos son red de creencia profunda (Deep belief network),restricted machine Boltzmann, autoencoders regularizados y deep machine Boltzmann. Por otro lado, modelo híbrido se refiere a la arquitectura profunda utilizando la combinación de un modelo discriminativo y generativo. Un ejemplo de este modelo es DBN para entrenar una CNN, lo que puede mejorar El rendimiento de la CNN profunda sobre la inicialización aleatoria [36].

2.3. Estado del arte.

Actualmente las personas que pueden mirar una imagen, pueden reconocer instantáneamente qué objetos se encuentran en esta y dónde se encuentran situados dentro de la imagen. La capacidad de detectar objetos rápidamente, combinada con el conocimiento de una persona ayuda a emitir un juicio preciso sobre la naturaleza del objeto. Un sistema que simula la capacidad del sistema visual humano para detectar objetos es algo en lo que los

científicos han estado investigando [12]. Rapidez y precisión son los dos requisitos previos para los que se examina un algoritmo de detección de objetos. La detección de objetos es uno de los problemas clásicos de la visión por computadora. No sólo clasifica el objeto en la imagen, sino que también localiza ese objeto. En décadas anteriores, los métodos utilizados para abordar este problema consistían en extraer diferentes áreas en la imagen utilizando cajas de diferentes tamaños y aplicar el problema de clasificación para determinar a qué clase pertenecen los objetos. Estos enfoques tienen la desventaja de exigir una gran cantidad de cálculos y de estar divididos en varias etapas [47].

En cambio, las recientes tendencias en aprendizaje profundo (Deep Learning) se han colocado en la mira de muchos investigadores por desafíos y competencias[23] relativamente recientes incluyendo detección de objetos [21].

Inclusive en la época actual diversos modelos básicos aplicables a detección de objetos se comparten públicamente y se pueden utilizar para lo conveniencia del usurario y tener un buen entrenamiento y listo para distintas tareas en clasificación o detección.

KITTI dataset [13] se posiciona como un conjunto de entrenamiento muy popular en la detección para diversas clases sobre imágenes de carretera en diferentes ámbitos. Para la detección de peatones en concreto también es muy popular el la base de datos de Caltech [9].

Lo que las redes neuronales convolucionales permiten es una mejora importante tanto de rendimiento como velocidad para el área de visión en detección de objetos. En un inicio estas se basaban en ventanas deslizantes [38]. Es un gran trabajo para la localización debido a la imensa cantidad de campos y variables.

Posteriormente, en el [15], se propusieron R-CNNs para resolver el problema de localización con un paradigma de reconocimiento basado en regiones. Generaron sub-teoremas por búsqueda selectiva [42], utilizaron redes convolucionales para extraer un vector de características de longitud fija para cada teorema, y utilizaron SVMs lineales para clasificar cada región. Las redes de clasificación por regiones y dominio son costosas, pero se han propuesto varias mejoras para hacerlas mas eficientes computacionalmente hablando [14].

La Faster R-CNN [35] utiliza una red de propuesta de región (RPN) que comparte

la operación de convolución de la imagen de tamaño con la red de detección y, por lo tanto, no aumenta el coste computacional.Las RPN se entrenan de extremo a extremo para producir una clasificación de regiones de alta calidad, que luego es utilizada por la Faster R-CNN para la clasificación. Además de estar muestreada, existe una implementación robusta de esta red en Tensorflow, y en particular varias redes preentrenadas de código abierto en el repositorio [17], lo que la convierte en una candidata adecuada para iniciar una red de detección de objetos.

En MONO3D[6], se presenta un método para generar regiones de objetos 3D específicos de una clase a partir de imágenes monoculares utilizando el aprendizaje de modelos contextuales y semánticos.

La arquitectura SPD+RPN [46], ofrece un enfoque alternativo. Para los objetos pequeños, la fuerte activación de las neuronas convolucionales se produce con más frecuencia en la primera capa. La fusión dependiente de la escala se utiliza para representar los fotogramas delimitadores que son candidatos utilizando características convolucionales con la escala adecuada. Y se exponen clasificadores de rechazo en cascada por multicapa, en los que las características convolucionales de la primera capa se consideran clasificadores menos predominantes para eliminar los candidatos a objetos negativos.

En la MS-CNN [5], se propone una red convolucional multiescala compuesta por una subred de sugerencia de rango y una subred de identificación. La red propuesta realiza la detección en múltiples capas de salida y estos detectores de escala adicionales se combinan para formar una eficiente detección de objetos a múltiples escalas. El detector incrementa su porcentaje de detección con referencia a todos los demás métodos de KITTI en la detección de peatones y ciclistas, y ocupa el tercer lugar en la detección de vehículos.

En la red YOLOv4 [1] es diseñada con una operación rápida de un detector de objetos en sistemas de producción y optimización para cálculos paralelos, en lugar del indicador teórico de bajo volumen de cálculo (BFLOP) esta red está basada en una arquitectura con una propuesta de convoluciones a la par, propone que el algoritmo diseñado puede ser fácilmente entrenado y utilizado por cualquiera que use una GPU convencional Nvidia 1080 Ti para el entrenamiento, y así lograr en tiempo real, alta calidad y observación convincente.

Tabla 2.1: Precisión en porcentaje en la clase automóvil de KITTI dataset.

Método de detección	**Media porcentual**	**Tiempo de ejecución**
Mono3D (2017)	88.66	4.2 s / GPU
Faster R-CNN (2015)	81.84	2 s / GPU
MS-CNN (2016)	89.02	0.4 s / GPU
SDP+RPN (2016)	85.88	0.4 s / GPU
SDP+CRC (2016)	83.53	0.6 s / GPU
YOLOv4 (2020)	**92.13**	0.02 s / GPU

Tabla 2.2: Precisión en porcentaje en la clase peatón de KITTI dataset.

Método de detección	**Media porcentual**	**Tiempo de ejecución**
Mono3D (2017)	66.68	4.2 s / GPU
Faster R-CNN (2015)	65.90	2 s / GPU
MS-CNN (2016)	**73.70**	0.4 s / GPU
SDP+RPN (2016)	64.19	0.6 s / GPU
SDP+CRC (2016)	62.15	0.6 s / GPU
YOLOv4 (2020)	50.62	0.02 s / GPU

YOLOv4 logra una gran detección de automóviles en muy poco tiempo Tabla4.1, pero aun en la identificación de otras clases, está por debajo a comparación de las aquí propuestas con la referencia de la base de datos KITTI.

En los últimos años algunos de los principales desafíos relacionados con visión computacional han sido las tareas de clasificación, detección y segmentación de objetos [11] como ya lo vimos en este apartado tenemos oportunidad de mejora, nos centraremos en la tarea de detección de objetos combinando técnicas híbridas de redes neuronales convulocionales.

Tabla 2.3: Precisión en porcentaje en la clase ciclista de KITTI dataset.

Método de detección	**Media porcentual**	**Tiempo de ejecución**
Mono3D (2017)	66.68	4.2 s / GPU
Faster R-CNN (2015)	63.35	2 s / GPU
MS-CNN (2016)	**75.46**	0.4 s / GPU
SDP+RPN (2016)	73.74	0.4 s / GPU
SDP+CRC (2016)	61.31	0.6 s / GPU
YOLOv4 (2020)	54.30	0.02 s / GPU

3. FUNDAMENTACIÓN TEÓRICA

En la presente sección se discutirán los fundamentos teóricos sobre los cuales se basa la metodología propuesta.

3.1. Visión por computadora

La visión por computador es el conjunto de herramientas y procedimientos que permiten alcanzar, procesar y examinar escenas del mundo real, consiste en la extracción automatizada de información de las imágenes. Esto permite automatizar una amplia tonalidad de ocupaciones al suministrar a las máquinas la consultoría que necesitan para la toma de decisiones correctas en cada una de las ocupaciones en las que han sido asignadas. Los integrantes principales de un método de visión por computador son un sensor de imagen y un digitalizador. La consecución de la imagen en un determinado instante de tiempo conlleva dos pasos. En primer sitio, el muestreo de la escena, prohibida de fase continua, obteniendo un conjunto discreto de aciertos. En segundo punto, la cuantización de la muestra, en otras palabras, aplicar a cada punto un valor discreto representativo del rango en el que varía la muestra [25].

3.2. Aprendizaje automático

El aprendizaje automático es fundamentalmente una forma de estadísticas aplicadas con mayor énfasis en el uso de computadoras para estimar estadísticamente funciones complicadas y un menor énfasis en los intervalos de confianza en torno a estas funciones. Aprendizaje automático se podría definir como un programa de computación aprende de la experiencia E con respecto a una tarea T y alguna medida de rendimiento P, si es que el rendimiento en T, medido por P, mejora con la experiencia E. La mayoría de los algoritmos de

aprendizaje automático se pueden dividir en las categorías de aprendizaje supervisado donde requiere la intervención humana y aprendizaje no supervisado donde no ocupa intervención [30] .

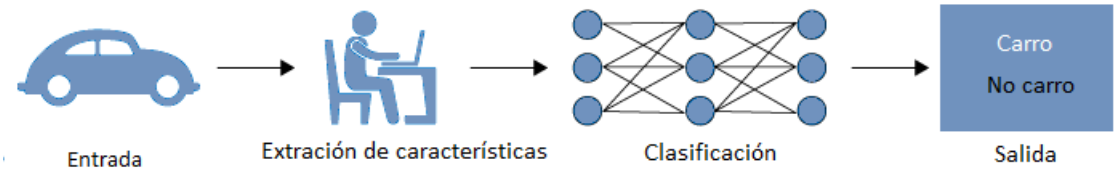


Figura 3.1: Aprendizaje automático

3.2.1 Aprendizaje profundo

Una arquitectura de aprendizaje profundo es una pila de múltiples capas de módulos simples, todos o la gran parte, los cuales están sujetos a aprendizaje, y muchos de los cuales computan asignaciones no lineales de entrada y salida. Cada módulo en la pila transforma su entrada para aumentar tanto la selectividad como la invariancia de la representación. Algunas de las diferencias entre aprendizaje automático y profundo, como se muestra en la figura 3, es la forma del aprendizaje, mientras aprendizaje automático requiere supervisión, se tiene que guiar al programa en todas las fases del sistema para que sepa identificar cada categoría automáticamente, el aprendizaje profundo requiere supervisión, es una técnica mejorada donde los sistemas alcanzan niveles de aprendizaje en un grado más detallado [8].

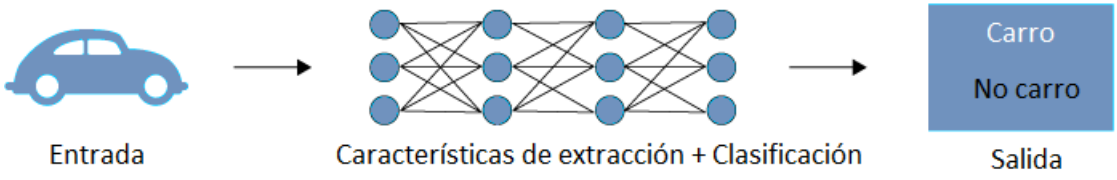


Figura 3.2: Aprendizaje profundo

3.3. Redes Neuronales

3.3.1 Neurona

La neurona es el elemento que da origen a las redes neuronales. Se origina en el perceptrón que fue propuesto en la década de 1950. Un perceptrón tiene muchas entradas y una salida como se muestra en la figura 3.3. Pondera las entradas y compara la suma de un umbral dado con valor binario de salida. La ecuación 3.1 da la expresión matemática del perceptrón.

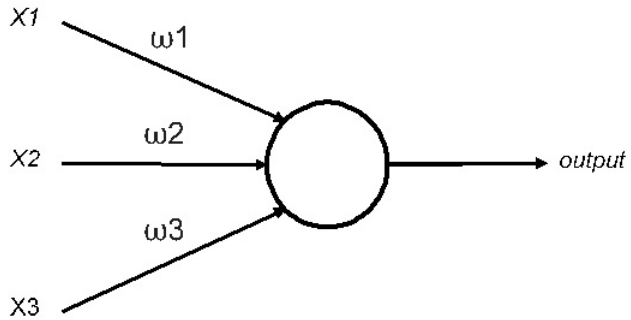


Figura 3.3: Neurona simple, donde x_i son las entradas y w_i los pesos a modificar.

$$\text{output} = \begin{cases} 0 \text{ si } \sum x_i w_i > \text{límite} \\ 1 \text{ si } \sum x_i w_i \leq \text{límite} \end{cases} \tag{3.1}$$

Dado que las salidas de un perceptrón utilizan la curva de la función escalonada. Solo cuando la suma de entradas cambia cerca del umbral, la salida saltará entre 0 y 1. De cualquier forma, un pequeño cambio de entrada no se reflejará en la salida, para producir un cambio notable en las salidas, se introduce la función de activación para cambiar el modelo del perceptrón. La Figura 3.4 nos presenta dos funciones de activación (función Sigmoide y función ReLU) de las varias que se pueden proponer. A través de la curva que agrega la función de activación, nosotros podemos encontrar como un pequeño cambio en las entradas

se refleja en las salidas con mayor facilidad [49].

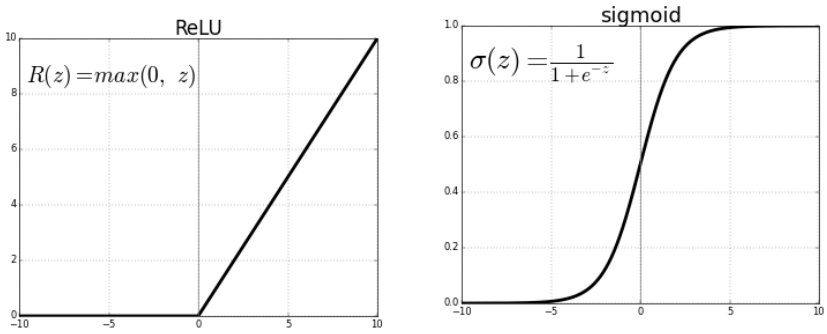


Figura 3.4: Funciones de activación

3.3.2 Estructura de Redes Neuronales

Las redes neuronales generalmente consisten en múltiples capas como se muestra en la Figura 3.5. La primera la capa se llama capa de entrada, la última capa se llama capa de salida y todas las neuronas en el centro son capas ocultas. Cada capa tiene neuronas. Las neuronas entre las capas adyacentes están conectadas para pasar información. En general cuando hablamos de redes neuronales de total conexión, las neuronas están completamente conectados como se muestra en la Figura 3.5. Entre dos capas adyacentes, cada par de neuronas tiene una conexión. Por ejemplo, si dos capas adyacentes respectivamente tienen 'm' y 'n' neuronas, el número total de conexiones será m x n.

Matemáticamente la representación de una capa sería la siguiente:

$$y = \varphi(b + \sum_{i=1}^{m} x_i w_i) \tag{3.2}$$

Donde w son los pesos a modificar, x las entradas y b el termino independiente conocido como 'bias' activada por la función φ.

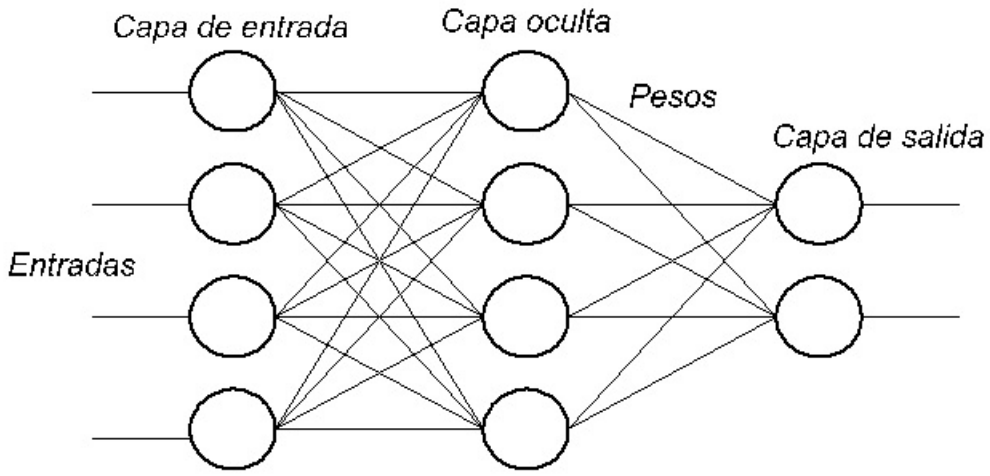


Figura 3.5: Red Neuronal con una capa oculta

En una red neuronal, las salidas de las capas anteriores se convierten en las entradas de la siguiente disposición. Cuando la red está en estado activo, la primera capa sólo toma decisiones simples y pasa a la segunda capa como entradas. Basado en las decisiones simples, la segunda capa puede tomar decisiones más inteligentes. De esta manera, a medida que la información pasa por más capas, la red puede tomar mejores decisiones [18].

3.3.3 Redes Neuronales Convolucionales

Las redes convolucionales, también conocidas como redes neuronales convolutivas (CNN por sus siglas en inglés), son un tipo especializado de red neuronal para procesar datos que tiene una topología similar a una cuadrícula (matriz). Las redes convolucionales han tenido un éxito extraordinario en aplicaciones prácticas. Las redes convolucionales son simplemente redes neuronales que usan la convolución en lugar de la multiplicación general de matrices en al menos una de sus capas, son muy potentes para todo lo que tiene que ver con el análisis de imágenes, debido a que son capaces de detectar características simples como por ejemplo detección de bordes, líneas, etc. y componer en características más complejas hasta detectar lo que se busca. Consta de capas convolucionales y de reducción alternadas, y al finalmente tiene capas de conexión total como una red perceptrón multicapa Figura 4.4.

En la convolución se realizan operaciones de productos y sumas entre la capa de partida y los n filtros que genera un mapa de características Figura 3.7. Las características

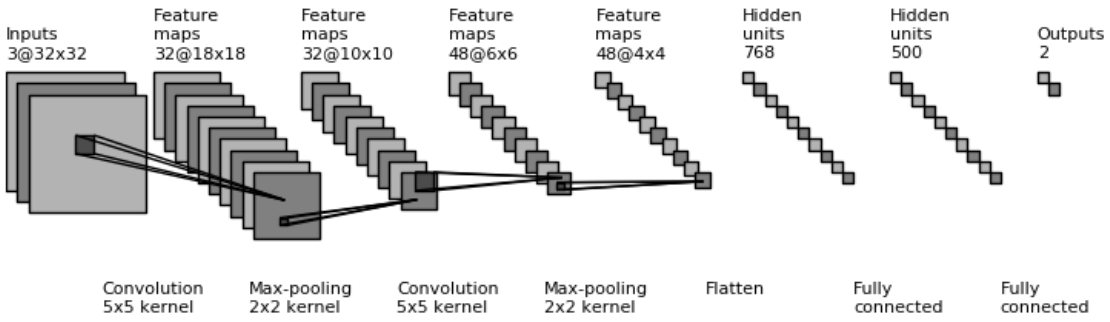


Figura 3.6: Arquitectura original de las CNN [24].

extraídas corresponden a cada posible ubicación del filtro en la imagen original.

La ventaja es que el mismo filtro sirve para extraer la misma característica en cualquier parte de la entrada, con esto que consigue reducir el número de conexiones y el número de parámetros a entrenar en comparación con una red multicapa de conexión total [5].

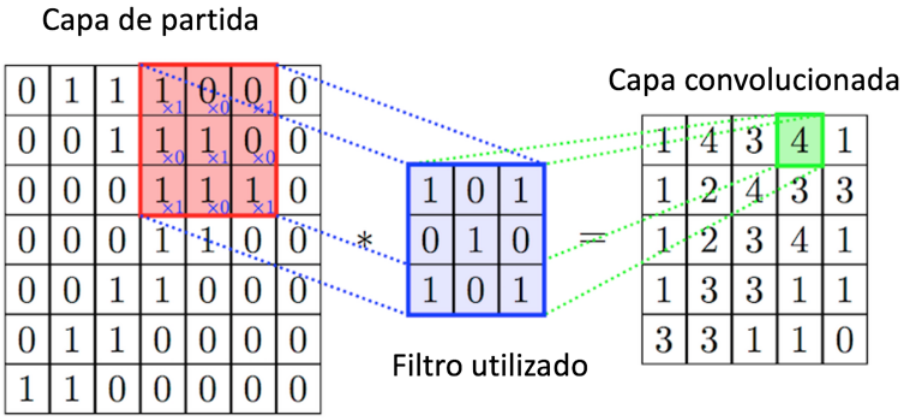


Figura 3.7: Salida de una convolución de un kernel 3x3

3.4. Algoritmos de búsqueda y entrenamiento

Los múltiples parámetros que existen en una red neuronal han de ser ajustados para acercar con mayor precisión a la función que se desea modelar. Para dicha labor se utilizan los cálculos de aprendizaje. Este apartado se centra comúnmente en tres técnicas; la optimización por descenso de gradiente, la optimización por desprecio de gradiente estocástico y la propagación hacia atrás. Estas técnicas son las mas comúnmente explotadas para en-

caminar redes neuronales, en este trabajo se experimenta con algoritmos poco utilizados en las CNNs estándar, debido a que, el entrenamiento puede entenderse como un conflicto de optimización, por ello es inevitable concretar la función que se empleará[25].

Variando los grados de los hiperparámetros de la red (pesos y sesgos) se tratará de averiguar a ser posible un valor optimo [8] [32].

3.5. Metaheurísticas

Las técnicas de optimización metaheurísticas se han inspirado principalmente en conceptos muy simples. Estos algoritmos suelen basarse en fenómenos físicos, comportamientos de animales o conceptos evolutivos. Tienen mayor flexibilidad a diferentes problemas sin ningún cambio especial en la estructura del algoritmo, ya que en su mayoría asumen los problemas como cajas negras. En otras palabras, solo las entradas y salidas de un sistema son importantes para una metaheurística [26].

Las metaheurísticas tienen capacidades superiores para evitar los óptimos locales en comparación con las técnicas de optimización convencionales. Esto se debe a la naturaleza estocástica de las metaheurísticas que les permiten evitar el estancamiento en las soluciones locales y adentrarse extensamente en todo el espacio de búsqueda. El cual para problemas reales suele ser desconocido o muy complejo y con una gran cantidad de óptimos locales, por lo que las metaheurísticas tienen buen desempeño únicamente teniendo claro el objetivo [31].

3.5.1 Algoritmos Genéticos (GA)

El algoritmo genético es una metaheurística inspirada en el proceso de selección natural creado por John Henry Holland en el año 1970, surgió con este algoritmo base de muchas representaciones metaheurísticas [45]. Un algoritmo genético estándar requiere dos requisitos previos, es decir, una representación genética del dominio de la solución y una función de aptitud para evaluar a cada individuo. La idea central del algoritmo genético es permitir que los individuos evolucionen a través de algunas operaciones genéticas como se muestra en el algoritmo. Las operaciones populares incluyen selección, mutación, cruce. El

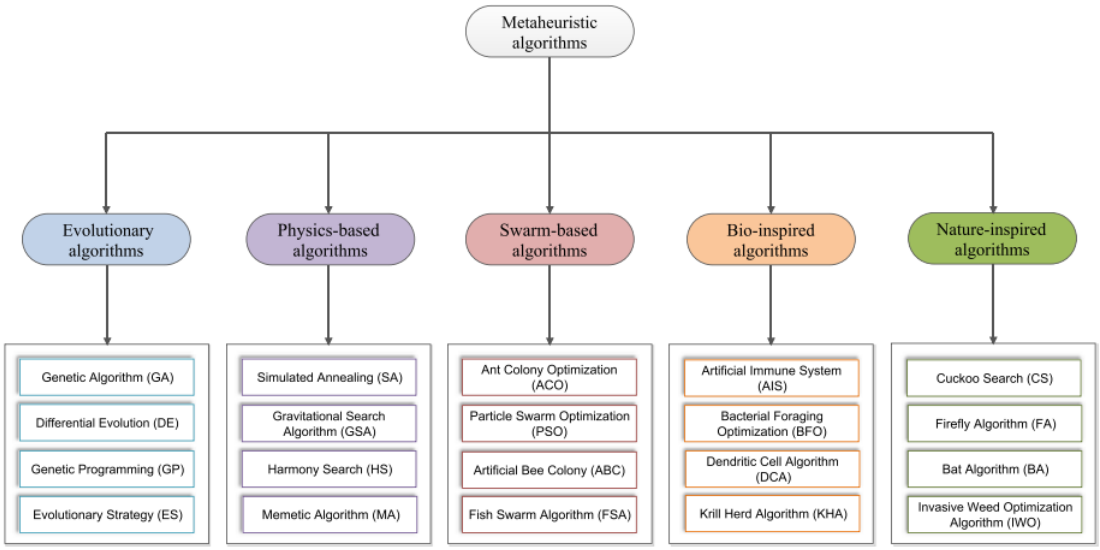


Figura 3.8: Clasificación de los algoritmos metaheurísticos [28] .

proceso de selección nos permite preservar a los individuos fuertes mientras eliminamos a los débiles. Las formas de realizar la mutación y el cruce a menudo se basan en las propiedades del problema específico [27].

3.5.2 Optimización por enjambre de partículas (PSO)

El algoritmo de optimización con enjambre de partículas (PSO) fue desarrollado por J. Kennedy y R. C. Eberhart [10], el cual se basa del comportamiento de parvadas de aves, colonias de abejas, bancos de peces, entre otros. Se puede utilizar para resolver problemas de optimización que carecen de conocimiento del dominio. La población está constituida por una serie de partículas. Cada uno de ellas representa un individuo. Busca la mejor solución actualizando velocidad y vector de partículas de acuerdo con las ecuaciones (1) y (2). Donde v_{id} representa la velocidad de la partícula i en la d-ésima dimensión, x_{id} representa la posición de la partícula i. P_{id} y P_{gd} son los mejores locales y el mejor global, r_1, r_2 son números aleatorios entre 0 y 1, mientras que w, c_1 y c_2 son peso de inercia y coeficientes de aceleración para explotación y aceleración para los coeficiente de exploración, respectivamente.

$$V_{id}(t+1) = w * v_{id}(t) + c_1 * r_1 * (P_{id} - x_{id}(t)) + c_2 * r_2 * (P_{gd} - x_{id}(t)) \tag{3.3}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \tag{3.4}$$

3.5.3 Optimizador lobo gris (GWO)

El algoritmo metaheurístico del lobo gris salió a la luz en 2014 por obra de Seyedali Mirjalili [29]. Donde se muestra el comportamiento de este animal su forma de caza y su conducta social y de particular interés es que tienen una jerarquía social dominante muy estricta donde llamamos a estos grupos como alfa, beta, delta y omega, cada una de estos grupos juega un papel importante en la manada. Para modelar matemáticamente la jerarquía social de los lobos, consideramos la solución más adecuada como la alfa (a). En consecuencia, la segunda y tercera mejores soluciones se nombran beta (b) y delta (d) respectivamente. Se supone que el resto de las soluciones candidatas como omega (x). En el algoritmo GWO la búsqueda está guiada por a, b y d. Los lobos x siguen a estos tres lobos y así estos rodean a sus presas durante la caza. Matemáticamente se representa en la ecuación (3) y (4) Donde t indica la iteración actual, $\vec{A}$ y $\vec{C}$ son vectores de coeficientes, $\vec{X_p}$ es el vector de posición de la presa, $\vec{X}$ indica el vector de posición de un lobo gris.

$$\vec{D} = |\vec{C} \cdot \vec{X_p}(t) - \vec{X}(t)| \tag{3.5}$$

$$\vec{X}(t+1) = \vec{X_p}(t) - \vec{A} \cdot \vec{D} \tag{3.6}$$

Donde los componentes de $\vec{a}$ se reducen linealmente de 2 a 0 en el transcurso de las iteraciones y r_2, r_2 son vectores aleatorios en [0, 1].

$$\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a} \tag{3.7}$$

$$\vec{C} = 2 \cdot \vec{r_2} \tag{3.8}$$

3.5.4 Algoritmo de optimización de ballenas (WOA)

El algoritmo de optimización de la ballena jorobada se presenta en 2017 por Seyedali Mirjalili [28]. Se puede interpretar como una modificación al algoritmo del lobo gris (GWO) donde en este caso representa de igual manera su comportamiento de caza, las ballenas jorobadas pueden reconocer la ubicación de sus presas y rodearlas. Las ecuaciones principales son las descritas en el algoritmo GWO a diferencia del este método, una ecuación en espiral es creado entre la posición de la ballena y la presa para imitar el movimiento en forma de hélice de las ballenas jorobadas dada la siguiente ecuación (9).

$$\vec{X}(t+1) = \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X}*(t) \tag{3.9}$$

Donde $\vec{D'}$ indica la distancia de la ballena a la presa (la mejor solución obtenida hasta ahora), b es una constante para definir la forma de la espiral logarítmica, l es un valor aleatorio de $[-1, 1]$ y $\cdot$ es una multiplicación elemento por elemento. Aquí se tiene en cuenta el vector donde una ballena crea un círculo que se contrae para llegar a su presa, se asume una probabilidad del 50 por ciento para elegir esta distinción al modelo circular GWO.

$$\vec{X}(t+1) = \begin{cases} \vec{X}*(t) - \vec{A} \cdot \vec{D} & \text{si} \quad p < 0{,}5 \\ \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X} * (t) & \text{si} \quad p \geq 0{,}5 \end{cases} \tag{3.10}$$

Donde p es un número aleatorio uniforme de [0,1].

3.6. Detección de objetos

Los detectores de objetos basados en CNN actuales tienen una estructura similar, que tiene un componente de Region Proposal seguido de un clasificador CNN como se muestra en la Figura 3.9. Los investigadores utilizan métodos de propuesta de región para producir un montón de regiones candidatas, cada uno de los cuales puede contener un tipo de objeto. Luego, deje que cada región pase por el CNN para hacer la clasificación. La idea esencial

detrás de esta estructura es convertir un problema de detección de múltiples objetos en un solo problema de clasificación de objetos [46] [12].

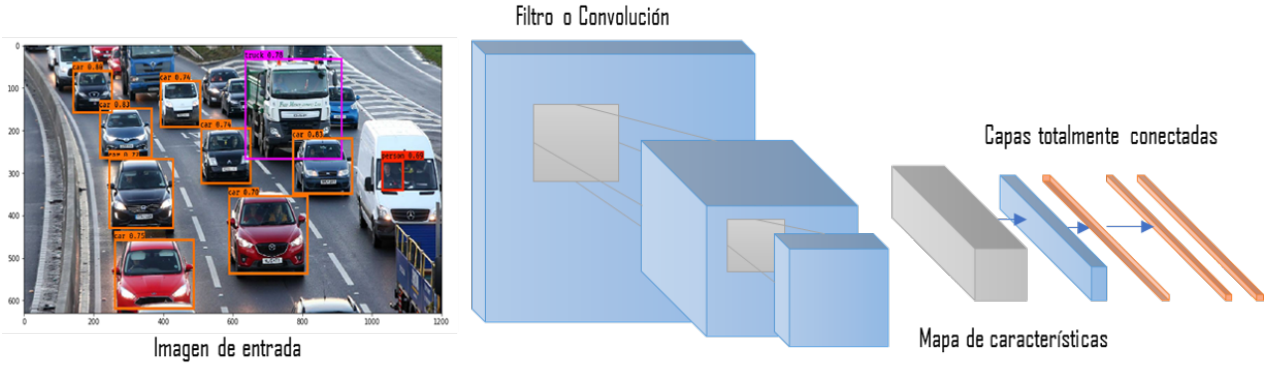


Figura 3.9: Estructura del modelo de detección de objetos basado en CNN. Region Proposal para generar regiones candidatas, luego pasa para que cada región pase por el clasificador CNN.

3.7. Sobre-entrenamiento y Sub-entrenamiento

La creación de un modelo funcional y robusto es un desafió constante, ya que este debería de ser capaz de funcionar con datos desconocidos, en otras palabras, datos fuera del conjunto de entrenamiento del modelo entrenado. Esto se le conoce como generalización. El sobre y sub-entrenamiento o ajuste (overfitting, underfitting) son un termino utilizado para encontrar la viabilidad entre ambos para evaluar un modelo en referencia a datos nunca antes vistos. Un modelo entrenado con un conjunto de entrenamiento de poca densidad de datos y con distinciones poco relevantes entre los datos, aunado a esto, una cantidad de ciclos suficientes para tener un alto valor de precisión dejando de lado otros parámetros de evaluación , el modelo tenderá a sufrir un sobre-ajuste, donde a datos nunca antes vistos, el resultado que tuvimos en las pruebas no se reflejara a estos nuevos datos. Por lo contrario si tendemos a entrenar un modelo con datos muy dispersos, con mucho ruido o un entrenamiento deficiente, produce lo contrario llamado sub-ajuste, donde al introducir datos nuevos o desconocidos sufrirá una muy baja generalización provocando que reconozca muchos más datos positivos a los verdaderamente positivos [22].

Podemos evitar el sobre y sub entrenamiento tomando las siguientes recomendacio-

nes:

- Aumentar la cantidad de datos.
- Crear datos artificiales correspondientes a los datos reales.
- Tener muestras consistentes y equilibradas por clase o categoría.
- Dividir los datos para entrenamiento, prueba y validación.
- Aplicar técnicas de validación cruzada

3.8. Red Neuronal You Only Look Once (YOLO)

You Only Look Once (YOLO) es una arquitectura dentro del estado del arte, que aprovecha y modifica ciertas características de una red neuronal convolucional. [33].

La red neuronal YOLO en su primera versión se publicó en 2015, por su gran desempeño a continuado en constante desarrollo e investigación para tener una evolución o mejoría en la velocidad, detección, precisión o requerir menor poder computacional, actualmente existen 5 versiones de esta red neuronal convolutiva sobre la capa Darknet.

La base de esta arquitectura YOLO utiliza una red neuronal a la imagen completa y aumenta su rendimiento computacional. Esta arquitectura de red trabaja por secciones sobre la imagen y haces predicciones múltiples de las cajas delimitadoras (bounding box o bbox) y crea una probabilidad de detección por categoría por cada caja delimitadora. Este emplea un método para suprimir los no-máximos, elimina las cajas delimitadoras múltiples detectadas sobre un mismo objeto.

En el presente trabajo de investigación, se empleó la versión numero 3 de YOLO publicada en 2018 (YOLOv3) [34]. Se profundizará brevemente en los componentes de YOLOv3 para comprender su funcionamiento.

3.8.1 Arquitectura Neuronal

La arquitectura de YOLOv3 está compuesta por 53 capas convolucionales, razón por la cual recibe el nombre de Darknet-53 3.10. Cada capa convolucional es seguida de una

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figura 3.10: Arquitectura base de la red neuronal YOLO: Darknet-53 . [33]

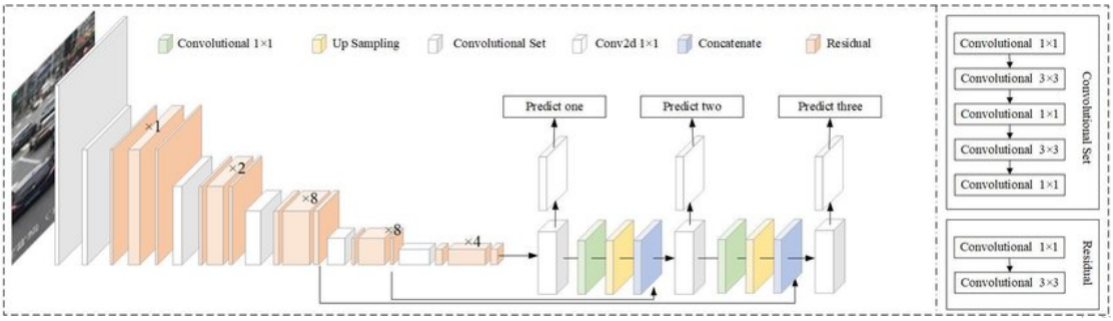


Figura 3.11: Arquitectura YOLOv3 [34] .

normalización de lote (batch normalization) y la función de activación Leaky ReLU. No se utiliza ninguna capa de reducción, en su lugar, se utilizan capas convolucionales con paso igual a dos. Se reduce con esto la dimensión del mapa de características, de esta manera evitamos perder los atributos del nivel anterior que sustituyen las capas kernel de redimensión. Como resultado se obtiene una red neuronal convolutiva eficiente y eficaz en la detección y clasificación [34].

3.8.2 Procesamiento

YOLOv3 puede trabajar con diferentes estructuras de pixeles en cada parte del entrenamiento, pero se requiere que las imágenes de entrada se transformen en datos de la misma longitud, preferiblemente efectúan un mejor trabajo con un múltiplo de 32, por ejemplo; 128, 256, 416, etc. Si estas no se cumple desde un inicio la arquitectura darknet no trabaja de manera óptima. Se configura la entrada como:

$$Entrada(m, h, w, d) \tag{3.11}$$

Donde:

- m es el tamaño del lote de entrenamiento (batch).
- h es la altura de la imagen (height).
- w es el ancho de la imagen (width).
- d son los canales de la imagen de entrada (depth).

Dependiendo del tamaño de la memoria gráfica, el tamaño del lote de entrenamiento puede ser subdividido. Ejemplo, si contamos con un lote de 64, estas a su vez una subdivision de 16, solo 4 imágenes por lote en paralelo se procesaran en la red neuronal convolucional. Sustituyendo los valores para este ejemplo y empleando una imagen de 3 canales de color RGB de 416$*$416, en YOLOv3 tendremos:

$$Entrada(64, 416, 416, 3) \tag{3.12}$$

YOLOv3 emplea un método de detección a tres diferentes escalas, donde redimensiona la imagen de entrada en factores de 32, 16 y 8. Con el ejemplo anterior de los datos de $Entrada(64, 416, 416, 3)$ se reduce la dimensionalidad de la imagen por el multiplicador de 32 lo cual hace un mapa de características de 13x13. A cada elemento del mapa se le conoce como celda. En cada celda se hace la predicción de un numero fijo de bbox. Donde en cada una de las detecciones podríamos tener un valor muy elevado de bboxes predichas, para esto se pueden reducir con una supresión de no-máximos donde compararíamos la detección múltiple del objeto en cuestión y se utilizaría una métrica llamada intersección sobre unión (IoU) para descartar las bboxes con menor probabilidad, si cumple con nuestros criterios de umbral solo una bbox es admitida por objeto.

4. MATERIALES Y MÉTODOS

En esta sección veremos el procedimiento y los métodos empleados para la construcción de nuestro algoritmo de aprendizaje profundo que se utilizaron para cumplir el objetivo general y los objetivos específicos de esta investigación. Se presentan las herramientas empleadas, los conjuntos de entrenamiento para las pruebas preliminares y finales así como las métricas de evaluación que se utilizan para llegar a los resultados deseados.

4.1. Metodología

En la figura 4.1 vemos de manera gráfica la metodología empleada en este trabajo de investigación, la metodología consta de 7 fases principales para llegar a la conclusión de esta tesis. De manera general se describe las fases de esta a continuación:

- La primera etapa, investigación del estado del arte, consta de empaparnos con los diferentes métodos y arquitecturas disponibles de redes neuronales, comprender su uso y replicar algunas CNN para proseguir con una selección en base a su rendimiento. De igual forma se seleccionan posibles algoritmos de búsqueda a replicar y evaluar, centrándonos en los algoritmos de búsqueda basados en la naturaleza comúnmente llamados algoritmos metaheurísticos.

- En paralelo de la primera etapa se investigan las bases de datos disponibles para emplear, ya que existen diferentes bases de datos posibles a utilizar como lo son Kitti, Waymo, Oxford, etc., de acceso libre, donde se muestran imágenes de carretera, se pretende recolectar información para elegir la idónea o tomar las imágenes convenientes.

- Se reduce la búsqueda en base al estado del arte a las posibles redes neuronales convolu-

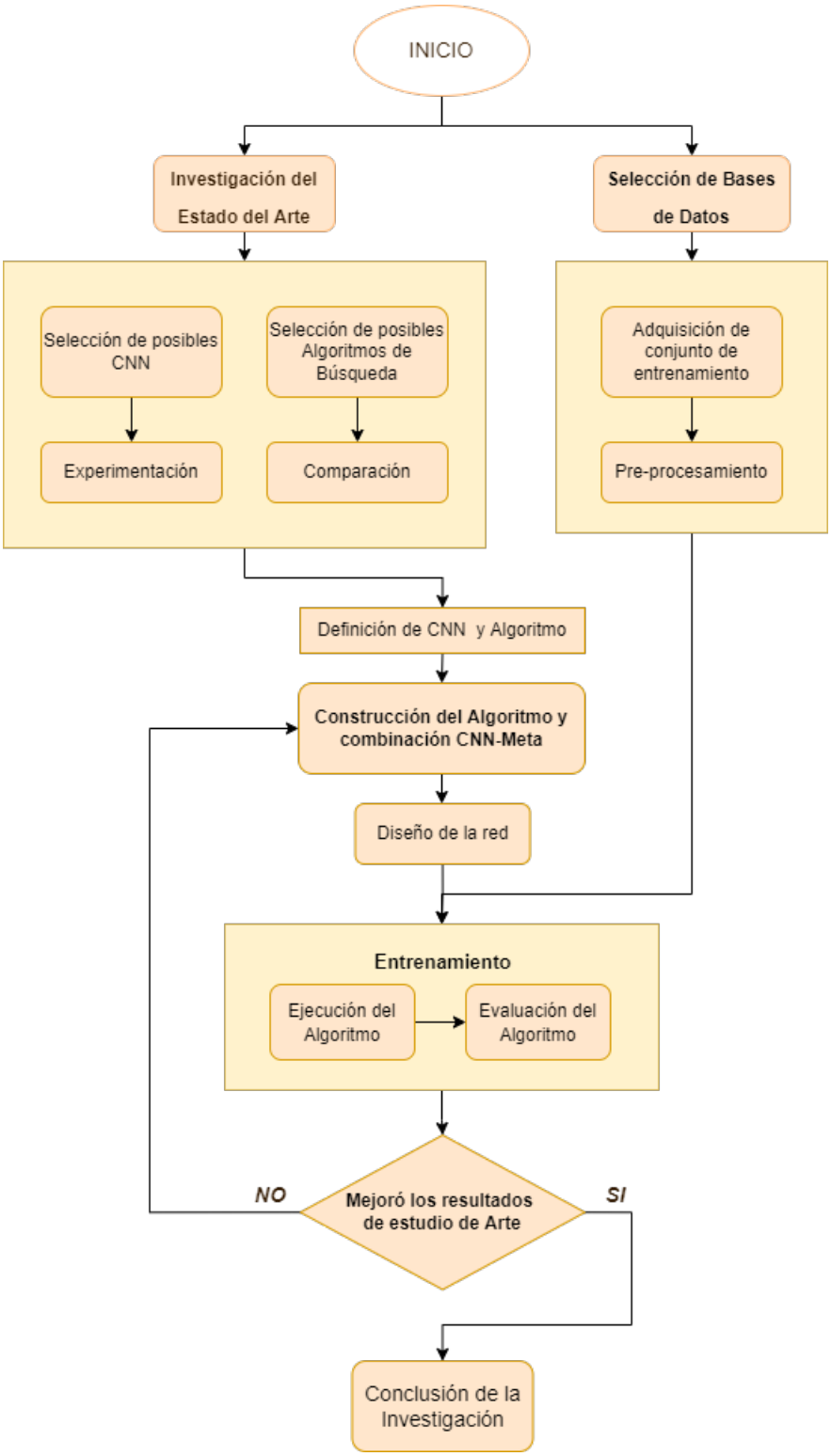


Figura 4.1: Metodología propuesta para el trabajo de tesis .

tivas, se experimenta con estas redes para validar su rendimiento, de la misma manera con los posibles algoritmos heurísticos comparando su relación con su capacidad de procesamiento de grande cantidad de datos y su eficacia.

- Se realiza un pre-procesamiento y un mapeo a los objetos a trabajar. Se di-mencionan y homogeneizan las imágenes para no perder su dimensiónalidad al hacer el re-escalado a menos pixeles.

- En la tercera etapa procedemos a la construcción del algoritmo final, donde pondremos a prueba la hipótesis de este proyecto la cual se centra en combinar los dos tópicos antes mencionados de la inteligencia artificial y así podremos mejorar la identificación en las imágenes a evaluar. Se evalúan distintas variables a tener en cuenta al momento de hacer un hibridismo CNN-Metaheurístico y se define la mejor forma de llevarlo a cabo. Se inicializan todas las variables antes de proceder con el algoritmo.

- En la etapa de entrenamiento se ejecutará el algoritmo de búsqueda y la red neuronal convolutiva sobre las imágenes procesadas para efectuar la detección de los elementos de interés; en esta parte se entrenan y evalúan las imágenes pasadas en lote, se calcula el error y se ajustan los pesos, todo este proceso es un ciclo hasta lograr los mejores índices de rendimiento y se realiza una interpretación de los resultados obtenidos.

- Se evalúan los resultados y el modelo híbrido empleado, si este modelo no cumple con las expectativas deseadas en comparación con el estado del arte, regresamos a la construcción del algoritmo donde re-evaluaremos las fallas y los posibles cambios a realizar para su nueva ejecución y evaluación. Si el algoritmo final cumple en la mejoría se analizan los resultados para la conclusión de la investigación.

4.2. Conjunto de entrenamiento

Actualmente existen bases de datos abiertas a la comunidad de investigación para el desarrollo de aplicaciones como lo son la detección de objetos enfocados en los vehículos

autónomos/imágenes de carretera que los trabajos mencionados en el estado del arte utilizan como referencia:

- **KITTI:** KITTI Vision Benchmark Suite está especializado para la conducción autónoma. Al conducir el automóvil equipado con múltiples sensores en ciudades medianas, áreas rurales y en en carreteras, recolectaron datos enriquecidos, incluidas imágenes y vídeo. Esta base de datos extrae características para la detección de objetos 2D / 3D, seguimiento de objetos y estimación de pose. Estas características incluye 7481 imágenes etiquetadas de 80 clases. Entre estas clases, sólo en los objetos importantes (automóviles, camiones, peatones, ciclistas, tranvías y personas sentadas) están etiquetados independientemente, todas las otras clases están etiquetadas como 'Misc' o 'DontCare'. En la Figura 4.3 podemos ver un collage de algunos ejemplos de este conjunto y la variedad de escenarios que contiene. En total contiene mas de 80k objetos etiquetados Figura 4.2 donde en su mayoría encontramos automóviles y peatones. En la referencia de detección de objetos y estimación de orientación, los objetos se subdividen en tres niveles de dificultad, fácil, moderado y difícil por su oclusión, truncamiento y distancia [13].

- **Natural Images:** La base de datos Natural Images consta de 6899 imágenes distintas divididas en 8 clases diferentes [37]. En la Figura 4.4 se muestra una imagen representativa de las imágenes a trabajar observamos que son de diferentes tamaños y estilos. Las clases que contiene este conjunto de entrenamiento se muestran en la Tabla 4.4, con su respectivo nombre y el número de imágenes por clase, estas base de imágenes fue construida a partir de diversas fuentes publicas para su uso en problemas usualmente de clasificación.

4.3. Herramientas de desarrollo

Con la finalidad de cumplir con los objetivos establecidos en el presente proyecto de tesis se trabaja sobre Python 2.7/3.0, Matlab r2020a y se ejecutó sobre una computadora con un CPU Intel Core i7-6700 a 3.4GHz, 16 GB de RAM y una GPU Nvidia 1060 de 3 GB.

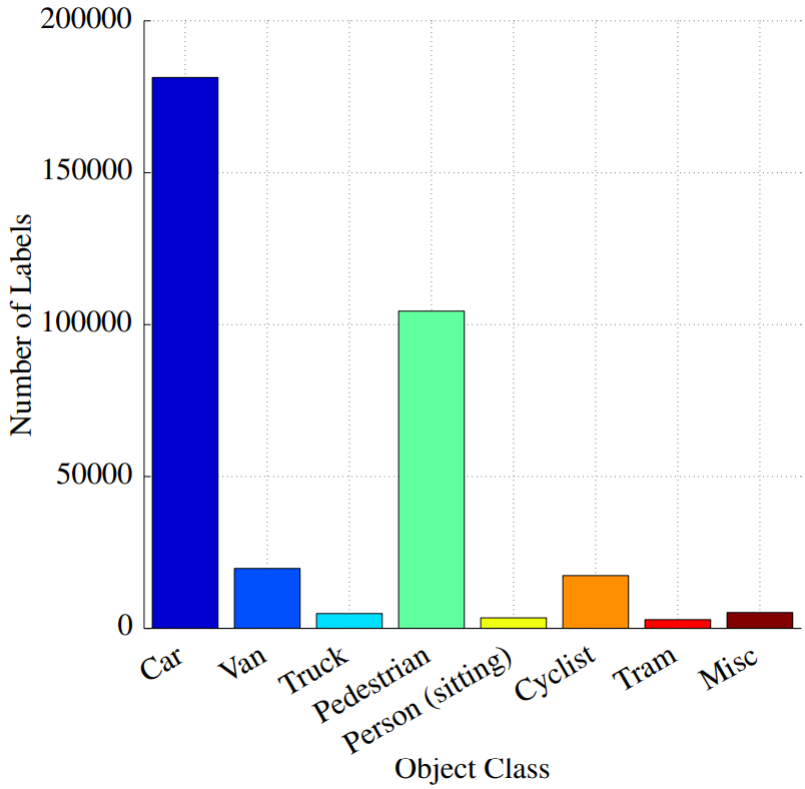


Figura 4.2: Distribucion de datos KITTI Vision Benchmark Suite [13] .

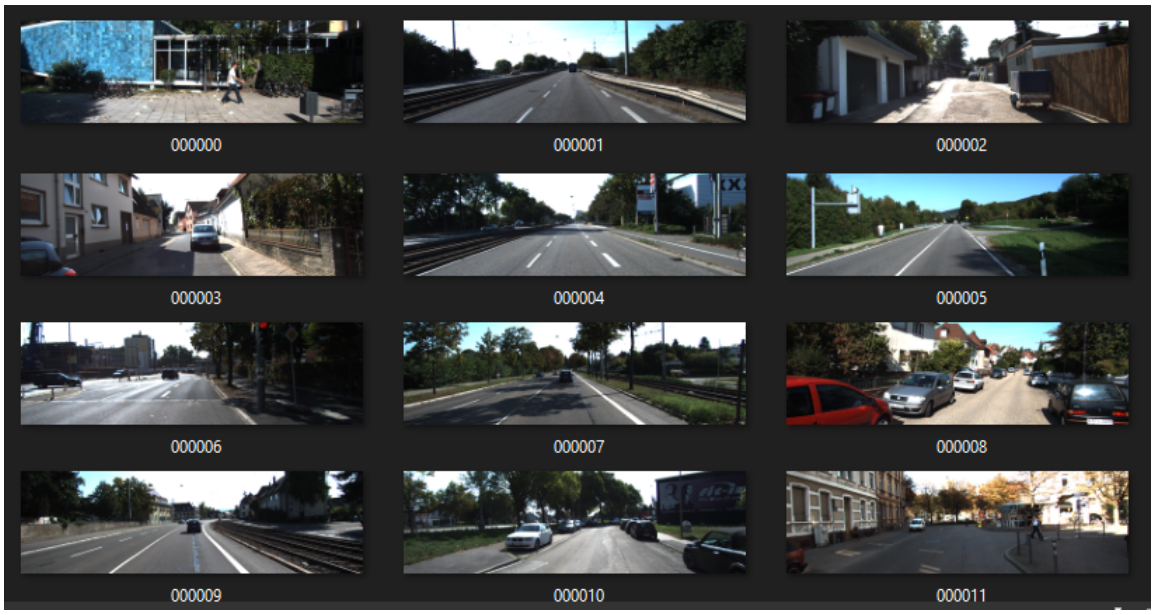


Figura 4.3: Muestra de la base de datos KITTI Vision Benchmark Suite [13] .

Figura 4.4: Muestra de la base de datos Natural Images [37] .

Tabla 4.1: Clases y numero de imágenes del dataset Natural Images [37].

Clase	Nombre	Imágenes por clase
1	Avión	727
2	Automóvil	968
3	Gato	885
4	Perro	702
5	Flor	843
6	Fruta	1000
7	Motocicleta	788
8	Persona	986

4.4. Implementación

4.4.1 Redes Neuronales Convoutivas

A través de la investigación en el estado del arte se descubrieron diversas redes neuronales convolutivas que podrían ser de gran utilidad para el desarrollo de esta tesis debido a su buen desempeño y su implementación en la detección de objetos. Algunas de las redes propuestas son:

- Faster R-CNN

- Single Shot
- YOLOv2
- YOLOv3

4.4.2 Algoritmo Metaheurístico

Dentro de los algoritmos de búsqueda existe una rama que destaca por su rendimiento en problemas de alto nivel a bajo coste computacional: los algoritmos metaheurísticos. Existe una amplia gama dentro de los algoritmos meta heurísticos, lo cual implica un reto para el correcto entendimiento de estos.

En nuestros criterios de evaluación decidimos basarnos en algoritmos que puedan tener buen desempeño con un gran número de individuos o partículas iniciales, de igual manera, tomaremos la implementación como rubro de elección, nos centraremos principalmente en comparar la categoría de enjambre sin embargo incluiremos uno de los algoritmos evolutivos más conocidos, el algoritmo genético, por su mención en el estado del arte.

Estos son los algoritmos seleccionados para las pruebas y entrenamientos:

- Optimización de partículas por enjambre
- Algoritmo Genético
- Optimizador de ballena
- Optimizador de lobo gris

Según describe el autor Nguyen Van Thieu [41] en el cual nos basamos para la elección de los algoritmos antes mencionados podemos visualizar en la figura 4.5 donde debido a su investigación clasifica los algoritmos dependiendo del grupo, rendimiento, escala de parámetros iniciales, parámetros a modificar, la versión del algoritmo si es que fue modificada para su correcto funcionamiento o quedo en su versión original y la dificultad de implementación de este donde aquí los clasifica de 4 formas:

- Fácil: Pocos parámetros, pocas ecuaciones, Código fuente corto

Groupo	Nº	Nombre	Clave	Año	Versión	Rendimiento	Gran escala	Param	Dificultad
Enjambre	1	Optimización de partículas por enjambre	PSO	1995	original	Bueno	Si	6	Fácil
Evolución	3	Algoritmo Genético	GA	1992	original	Bueno	No	4	Fácil
Enjambre	20	Optimizador de ballena	WOA	2016	original	Muy bueno	Si	2	Fácil
Enjambre	14	Optimizador de lobo gris	GWO	2014	original	Muy bueno	Si	2	Fácil

Figura 4.5: Tabla de comparación entre algoritmos metaheurísticos [41] .

- Medio: Mayor numero de ecuaciones que el nivel fácil, Código fuente más largo que el nivel fácil
- Difícil: Muchas ecuaciones, Código fuente más largo que el nivel Medio, el documento es difícil de leer.
- Muy difícil: Muchas ecuaciones, Código fuente demasiado largo, el documento es muy difícil de leer.

Debido a la complejidad de diversos algoritmos metaherusiticos y el tiempo de implementación requerido como antes se ha mencionado tomamos la dificultad como un gran factor en la elección.

4.5. Procesamiento de imágenes

Todos los métodos desarrollados para el análisis de imágenes u otros objetos constan de tres fases: Preprocesamiento, procesamiento y posprocesamiento. La fase de preprocesamiento se considera importante porque afecta la calidad de las fases posteriores y el rendimiento general.

Las imágenes dadas por la base de datos antes mencionadas pasan por una etapa de redimencionamiento para que el trabajo de procesamiento y tiempo de ejecución sea menor, se estandarizan a $416*416$ pixeles por tres capas de color, de igual manera se trabaja con

Figura 4.6: Ejemplo de re-escalado y re-dimensión de las imágenes. .

la dimesionalidad de estas imágenes para no perder congruencia en la perspectiva real de la imagen Figura 4.6.

4.6. Hibridismo: Limitaciones

Experimentos planteados de este silogismo:

Optimizador nativo de la red neuronal convolucional. Los optimizadores actuales de las redes neuronales se han estado acentuando de forma sólida como métodos efectivos y eficientes en la retro-propagación, a continuación se muestra una gráfica Figura 4.7 con los optimizadores más populares, por mencionar algunos [43]:

1. Descenso de gradiente estocástico (SGD).

2. Optimizador de monumento.

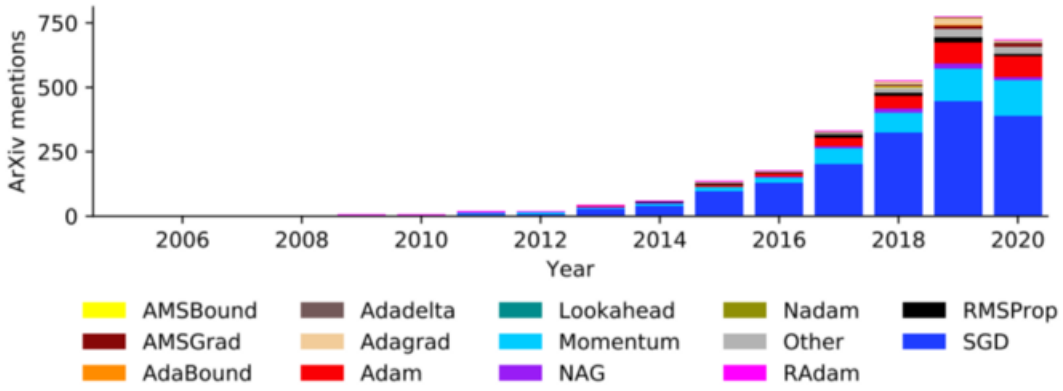


Figura 4.7: Optimizadores populares según las menciones en ArXiv 2020 [43] .

3. Propagación cuadrática media (RMSProp).

4. Estimación de torque adaptativo (Adam).

Elección idónea de híperparámetros. La elección de los hiperparámetros iniciales de una red neuronal convencional ha sido un reto ya que la elección de estos determina en gran parte la convergencia de la red y la modificación de sus parámetros para lograr los resultados favorables.

Ajuste en los parámetros por época. Internamente en una red neuronal los pesos y sesgos se modifican cada época o ciclo de entrenamiento, una vez iniciado el entrenamiento es complejo modificar estos hiperparámetros para hacer un reajuste y de estos depende la elección de los parámetros adecuados para obtener el resultado esperado como antes se mencionó.

Ajuste fino de la red neuronal. Al terminar el entrenamiento de una red neuronal ocasionalmente obtenemos resultados no convenientes para nuestro fin. Un ajuste fino en los parámetros puede darnos un mejor resultado acercándonos más a nuestro objetivo y revaluar nuestro experimento.

Al concluir estos experimentos en las limitaciones de realizar el hibridismo de una red neuronal convolucional y un algoritmo metaheurístico se decidió hacer pruebas preliminares para decantarse en el ajuste de parámetros y se tomaron en cuenta los siguientes puntos:

- Se acotó a pruebas de ajuste en los parámetros.
- Se limitó a únicamente clasificación
- Se elige una base de datos especial para clasificación
- Se definen las métricas a evaluar
- Tiempos de entrenamiento cortos para realizar la mayor cantidad de pruebas

4.7. Metaheurístico adaptado para el ajuste fino de parámetros CNN

El objetivo del algoritmo de este proyecto de investigación es aprovechar el beneficio de la búsqueda exhaustiva de los algoritmos metaheurísticos sobre los hiperparámetros de la red neuronal convolucional centrándose en las capas densas donde se trabaja con una extensa cantidad de parámetros.

Las capas convolucionales en los algoritmos de clasificación como detección de objetos son de suma relevancia. Estas capas juegan un papel importante en la parametrización de las entradas (imágenes) y dan paso a las capas densas para realizar las operaciones necesarias y llegar a la clasificación. Los algoritmos metaheurísticos tienen mecanismos libres de derivación. A diferencia de los enfoques de optimización basados en gradientes que por su practicidad se utilizan en redes neuronales y redes neuronales convolucionales. Las metaheurísticas optimizan estocásticamente los problemas. El proceso de optimización comienza con soluciones aleatorias y no es necesario calcular la derivada de los espacios de búsqueda para encontrar el óptimo. Esto hace que la metaheurística sea adecuada para problemas reales con información desconocida o compleja [2].

Se tiene claro que la combinación de estos métodos puede realizarse de diversas formas, como se mencionó antes: utilizar el metaheurístico después de cada etapa de entrenamiento o época, en determinadas épocas del entrenamiento, únicamente para su ajuste final. Todo esto se puede determina arbitrariamente dependiendo de la complejidad de la arquitectura, efectividad de convergencia, el tiempo de entrenamiento, entre otras [44].

En este caso se examina únicamente el algoritmo descrito en el pseudocódigo del Algoritmo 1 donde al finalizar las épocas se ejecutan los algoritmos de optimización para el ajuste fino de estos hiperparámetros sumado las operaciones que se ejecutan a través de las épocas en la CNN.

Algoritmo 1: Algoritmo híbrido CNN-Metaheurístico

Entrada: Base de datos: $clases\ Imagenes$, épocas CNN $epoch$, iteraciones Metaheurística it ;

1. Preprocesamiento $\leftarrow clases\ Imagenes$
2. modelo $\leftarrow$ Se crea el modelo CNN;
3. **while** i $<= epoch$
4. Época CNN $\leftarrow$ Ejecución de una época en el entrenamiento CNN
5. $get\ weights\rightarrow$ hiperparámetros
6. Inicio Metaheurístico:
7. Parámetros iniciales. $\leftarrow it$
8. Ingreso de hiperparámetros al algoritmo $\leftarrow$ se evita empeorar la salida
9. **while** j $<= it$
10. Generación aleatoria de individuos
11. Ejecución
12. Aptitud de los individuos $\leftarrow Max\ \ f(x)$
13. Mejores individuos
14. **end while**
15. **return:** Hiperparámetros
16. $set\ weights\leftarrow$ hiperparámetros
17. **end while**
18. modelo entrenado
19. Evaluación
20. **return:** clasificación de imagen

Tabla 4.2: Arquitectura utilizada en su implementación

Capa	Kernel	Parámetros
Conv2d	3x3	640
Max Pooling	2x2	-
Conv2d	3x3	36928
Max Pooling	2x2	-
Flatten	-	-
Dense1	32	663584
Dense2	32	1056
Dense3	8	264

4.8. Implementación del algoritmo híbrido en clasificación

4.8.1 Parámetros.

La arquitectura utilizada en este proyecto se basa en la propuesta original ilustrada en la Figura 4.1 y se representa en la Tabla 4.2 de manera detallada, similar a la LeNet-5 propuesta por Yann LeCun [24]. Se realizará el ajuste de parámetros en las capas densas durante 10 épocas de entrenamiento. Por lo tanto, los algoritmos descritos requieren parámetros de inicialización para su funcionamiento. Cada uno de los metaheurísticos se estableció con 30 individuos, 10 iteraciones por cada época de la red neuronal convolucional, una ventana de 25, y una longitud del problema igual a la cantidad de parámetros a evaluar después de las capas convolucionales en este caso particular sería de 664,896 variables.

4.8.2 Función Objetivo

En este trabajo se propone una función objetivo a maximizar (Ec. 9) por parte de los enfoques híbridos presentados. Esta, integra la exactitud Ex (Ec. 10) y la medición F1-score $F1$ (Ec. 13), las cuales son dos de las métricas más utilizadas en el campo de la clasificación.

$$Max \;\; f = w1 * Ex + w2 * F1 \tag{4.1}$$

Donde $w1 = 0.4$ y $w2 = 0.6$, son ponderaciones de la importancia de dichas métricas. Estos valores se obtuvieron mediante pruebas exhaustivas donde se observó que la CNN presenta mejor desempeño usando dicha configuración. Para fines de comparación, en los resultados se presentan por separado los valores de Exactitud, F1-score, así como los de precisión Pre (Ec. 11) y sensitividad $Sens$ (Ec. 12). Métricas que componen la F1-Score [20]. Donde TP es Verdadero Positivo (abreviado por sus siglas en inglés), TF es Verdadero Negativo, FP es Falso Positivo y FN corresponde a Falso Negativo [3].

$$Ex = \frac{TP + TF}{TP + TF + FP + FN} \tag{4.2}$$

$$Pre = \frac{TP}{TP + FP} \tag{4.3}$$

$$Sens = \frac{TP}{TP + FN} \tag{4.4}$$

$$F1 = \frac{2 * Pre * Sens}{Pre + Sens} = \frac{2 * TP}{2 * TP + FP + FN} \tag{4.5}$$

4.9. Función objetivo en detección

La evaluación de la etapa de entrenamiento se lleva acabo utilizando las métricas mAP 4.7 (mean Avarage Precision) e IoU (Intersection Over Union), las cuales evalúan la clasificación y la detección de objetos respectivamente.

$$AP = \frac{TP}{TP + FP} \tag{4.6}$$

$$mAP = \frac{\sum_{i=1}^{m} AP}{clases} \tag{4.7}$$

Siendo AP la precisión promedio por clase, TP los verdaderos positivos y FP los falsos positivos.

$$IoU(A, B) = \frac{A \cap B}{A \cup B} \tag{4.8}$$

Siendo A el conjunto de pixeles predichos por la Red Neuronal y B el conjunto de pixeles del objeto del conjunto de entrenamiento sobre. Donde IoU se clasifica de la siguiente manera:

- Si IoU > 0.5 es TP
- Si IoU < 0.5 es FP
- Si IoU > 0.5 pero la clase no corresponde es FN

5. RESULTADOS Y DISCUSIÓN

En este apartado se exponen las pruebas y resultados realizados para concluir este proyecto de tesis. Se genera un punto de partida para homogeneizar las condiciones iniciales de cada algoritmo híbrido iniciando con una época base para todos los algoritmos comparados.

5.1. Pruebas a las arquitecturas de la red neuronal convolucional

Como lo marca la metodología propuesta se implementaron algoritmos de detección de objetos sobre imágenes de carretera en Matlab sobre las redes neuronales convolucionales antes mencionadas, en la tabla 5.1 observamos la comparación de las pruebas realizadas sobre la precisión de cada una de las arquitecturas. Se destaca cómo la red neuronal YOLO V3 obtiene un mejor resultado en precisión media y tiempo evaluadas a una sola clase, por este motivo se decide trabajar en base a esta red.

5.2. Selección de metaheurísticos

En primera instancia se experimentaron con algunos metaheuristicos ya conocidos de buen rendimiento, estos algoritmos se probaron con funciones de alta dimensionalidad como lo son la función Levy y Rastrinig, estas dos funciones que son representadas en un

Tabla 5.1: Valores en los experimentos iniciales en detección de objetos.

Arquitectura CNN	**Precisión**	**Tiempo de ejecución**
Faster R-CNN (2015)	76.56 %	3.225 seg
YOLOv2 (2016)	85.68 %	3.196 seg
YOLOv3	87.62 %	1.256 seg
SDP+CRC (2016)	82.79 %	2.982 seg

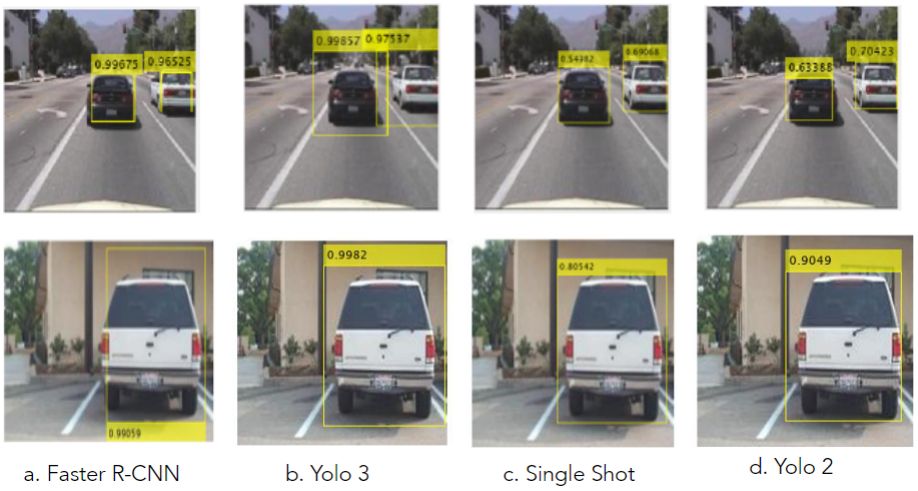


Figura 5.1: Imágenes comparativas en detección de vehículos .

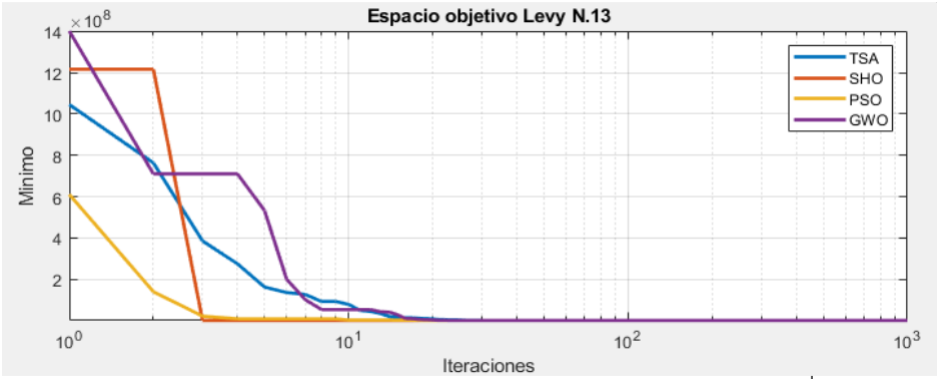


Figura 5.2: Comparativa de los primeros heurísticos implementados en la investigación: Levy.

espacio tridimensional en la figura 5.4 y 5.5. Gracias a estos algoritmos en los que se experimentó: TSA, SHO, PSO, GWO, se determino su buen rendimiento en dichas funciones, pero debido a las implicaciones de los parámetros iniciales de una red neuronal estos algoritmos fueron descartados y sustituidos por los cuatro algoritmos metaheurísticos antes mencionados en la figura 4.5 gracias a como el autor Van Thieu menciona.

5.3. Hibridismo en CNN y Metaheurístico

5.3.1 Clasificación

Antes de implementar la metodología propuesta con los algoritmos de detección de objetos sobre imágenes de carretera, se realizan pruebas a menor escala sobre clasificación,

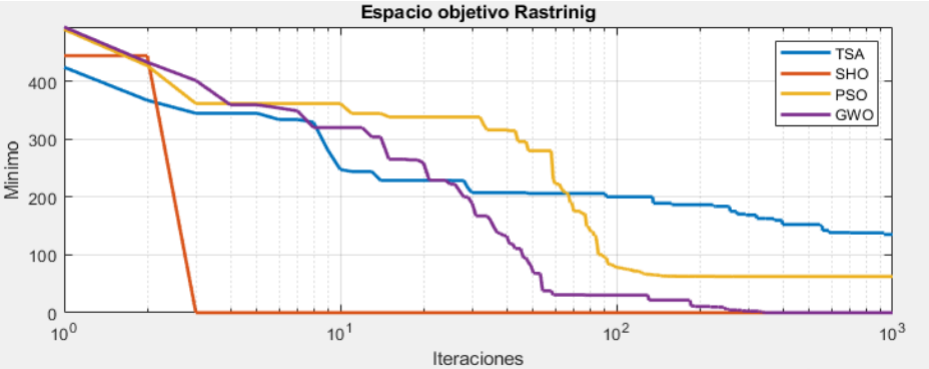


Figura 5.3: Comparativa de los primeros heurísticos implementados en la investigación: Rastrinig.

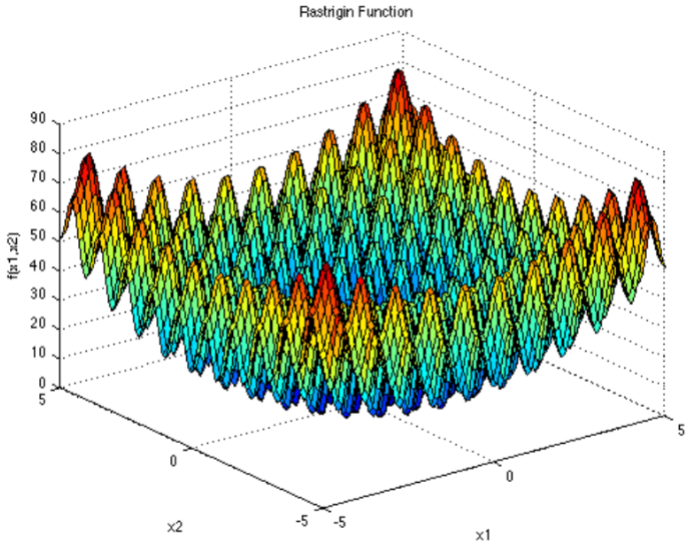


Figura 5.4: Rastringin Function en 3 dimensiones .

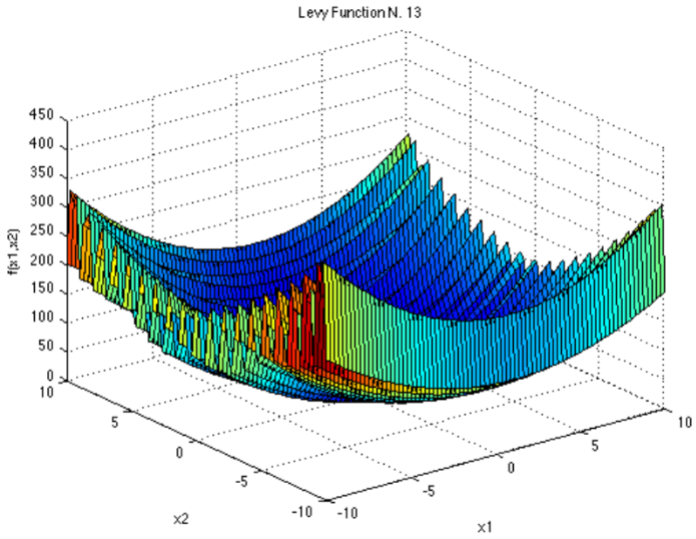


Figura 5.5: Levy Funtion N13 en 3 dimensiones .

se obtienen los datos presentados en la Tabla 5.2 para los máximos valores alcanzados por los 4 algoritmos híbridos y la CNN estándar de las métricas antes descritas.

El entrenamiento de la red neuronal convolucional se realizó con una relación 80/20 en los datos de entrenamiento y prueba, se realizaron 10 distintos experimentos hasta alcanzar los máximos valores que podemos apreciar en dicha tabla.

En la Figura 5.7 se representa gráficamente la convergencia al mayor valor de exactitud obtenido en la experimentación a través de las épocas establecidas para cada uno de los métodos. El comportamiento de los algoritmos durante las épocas se intercala utilizando el método CNN y el algoritmo metaheurístico. Podemos notar que existen fluctuaciones en estos cambios siendo la mas notoria en CNN-PSO, a través de las épocas cae la puntuación y en la siguiente iteración es recuperada. Esto podría deberse a la elección de los hiperparámetros de la entrada a la CNN, no logra aprovechar las propiedades del algoritmo híbrido por sus bajas iteraciones en la etapa metahuerística. En la Figura 5.6 observamos un ejemplo de la convergencia en la primera época de la CNN evaluada por los metaheurísticos, desde esta instancia se aprecia como destaca el algoritmo WOA.

Comparando los modelos híbridos con el método CNN obtenemos que en la precisión los mejores resultados fueron logrados por el método estándar de la CNN con un

Tabla 5.2: Evaluación de los algoritmos

Método	**Exactitud**	**Precisión**	**Sensitividad**	**F1-score**
CNN	0.8202	**0.9962**	0.1178	0.2054
CNN-GA	0.8224	0.9318	0.0901	0.1605
CNN-PSO	0.8260	0.9740	0.1079	0.1911
CNN-GWO	**0.8355**	0.9740	0.2450	0.3864
CNN-WOA	0.8347	0.9956	**0.4403**	**0.6053**

diferencia de $0.06\,\%$ al segundo mejor y del $6.44\,\%$ al peor. La sensitividad fue mejorada por CNN-WOA en $32.25\,\%$ respecto a CNN. En la evaluación F1-score se destaca nuevamente WOA con una diferencia de $40.0\,\%$. Finalmente para la exactitud se mejoró por el algoritmo CNN-GWO un $1.53\,\%$ al modelo estándar CNN.

Como se mostró en los resultados al modificar la arquitectura de la CNN utilizando la metodología híbrida propuesta en este trabajo se observa una mejora en algunas de las métricas pero la más notoria es F1-score lo cual nos indica que el utilizar algoritmos híbridos de inteligencia artificial puede mejorar la robustez del modelo final por lo tanto mejorar la clasificación de imágenes.

Analizando los resultados obtenidos podemos concluir que algunos de los resultados en los modelos hibridos no superaron el modelo estándar, esto podría deberse a la naturaleza de los mismos, debido a las pocas iteraciones en las pruebas realizadas. Y se concluye para la tarea de clasificación aquí presentada el algoritmo con mejor desempeño es CNN-WOA.

5.3.2 Detección de objetos

Con los resultado obtenidos en las pruebas de clasificación se determinó implementar únicamente el algoritmo metaheurístico WOA junto a la red neuronal YOLO V3. Debido a la convergencia mostrada en la figura TAL, se implementará el ajuste fino de los parámetros únicamente en el último ciclo o época de la red neuronal convolucional minimizando los tiempo de ejecución del algoritmo metaheurístico a diferencia de ejecutarlo cada ciclo.

Se ejecutaran pruebas con la Yolo V3 sin modificaciones con los mismos hiperparámentros iniciales que nuestra YOLO híbrida. Se utiliza un optimizador ADAM, una tasa de

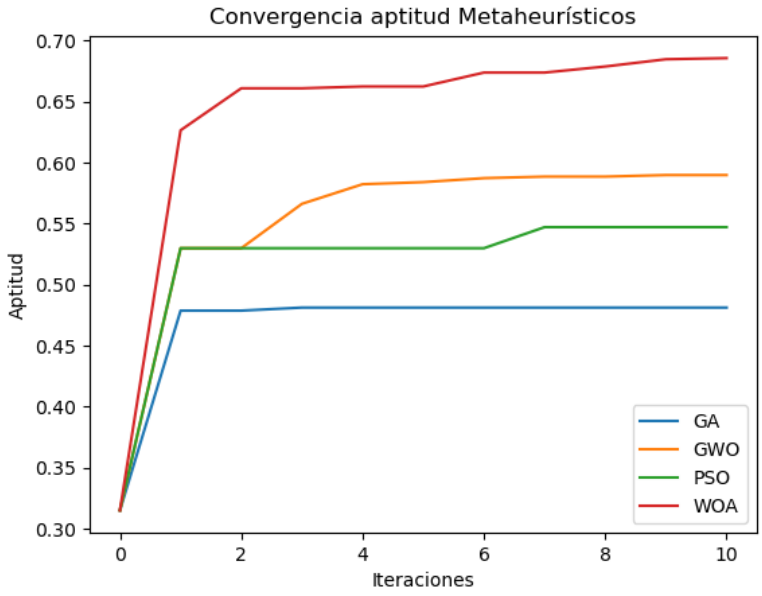


Figura 5.6: Comparación de convergencia de métodos metaheurísticos, 1 de 10 épocas.

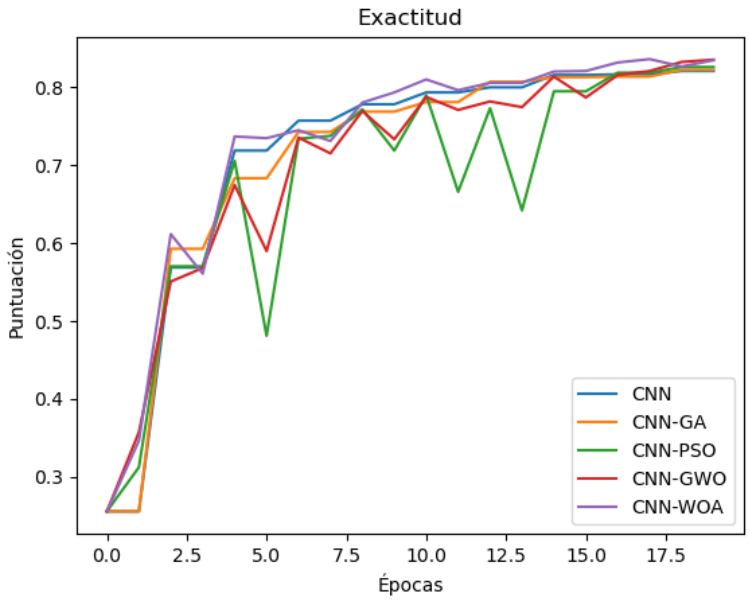


Figura 5.7: Gráfica sobre épocas de la Exactitud.

Tabla 5.3: Valores de la precisión por categoría en la detección $1/2$.

Método de detección	Automovil	Van	Camioneta	Peatón
YOLO v3	60.6 %	0.18 %	17.0 %	23.4 %
YOLOv3-Meta	58.8 %	17.7 %	37.5 %	16.3 %

Tabla 5.4: Valores de la precisión por categoría en la detección $2/2$.

Método de detección	Pea. Sentado	Ciclista	Tranvia	Otros	mAP
YOLO v3	0 %	0.55 %	0 %	0.36 %	11.8 %
YOLOv3-Meta	0 %	0.47 %	0.5 %	0 %	17.5 %

aprendizaje de 0.001, un lote de 2 y épocas totales de 30 en ambos casos con la diferencia que la red neuronal híbrida parte de la época anterior para ejecutar el algoritmo metaheurístico. El entrenamiento de ambas redes neuronales convolucionales se realizaron sobre una base de 60 % en imágenes de entrenamiento con un total de 4488, 20 % en imágenes de prueba que equivalen a 1496 imágenes y 20 % (1496 imágenes) para imágenes de validación sobre la base de datos KITTI. El tiempo de entrenamiento total para YOLO V3 simple fue de 22hrs, mientras que en la red híbrida tuvo una diferencia de 4hrs, teniendo un tiempo de entrenamiento total de 26hrs.

En la tabla 5.3 vemos como similar a nuestras pruebas de clasificación algunas de las clases son beneficiadas en la YOLO V3 simple, pero de igual manera la YOLO V3 híbrida destaca en otras categorías obteniendo un mejor resultado en la categoría Van, camioneta y no despegándose por un gran porcentaje de la categoría de automóvil. El mAP es superado por el YOLOV3 meta por una diferencia de 5.7 % con respecto a la YOLO V3 simple que obtiene un total de 11.8 %.

5.4. Imágenes comparativas

A continuación se mostrara una comparación entre la detección de las dos redes neuronales utilizadas, donde la parte superior tendremos la versión de YOLOV3 simple, la parte

(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.8: Imágenes comparativas finales entre arquitecturas .

media la YOLOV3 híbrida y en la parte inferior la imagen original sin detección. En alguunas de las imágenes mostradas apreciamos como el ajuste fino realizado por el hibridismo muestra una mejoría en detección en casos particulares donde la imagen tiene mayor información o muy pocos pixeles para la identificación de los vehículos y peatones algunos ejemplos puntuales los vemos en las figuras 5.15, 5.17, 5.21 y 5.22. Sin embargo como vemos mejoría en varias imágenes en algunas otras tenemos pequeños errores donde el algoritmo confunde señalamientos con vehículos por ejemplo en la figura 5.9 y 5.11.

(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.9: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.10: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

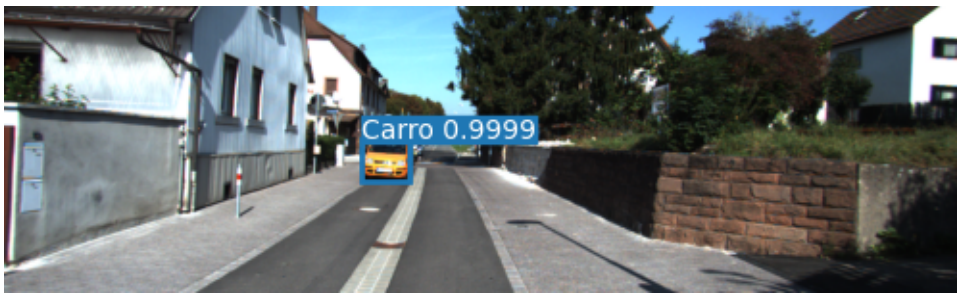
Figura 5.11: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.12: Imágenes comparativas finales entre arquitecturas .



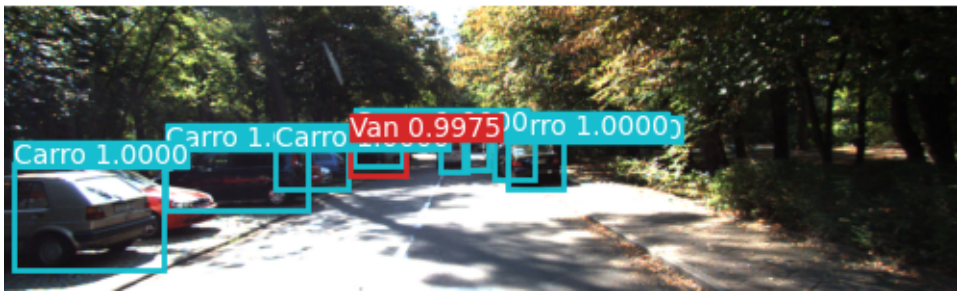
(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.13: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3



(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.14: Imágenes comparativas finales entre arquitecturas .

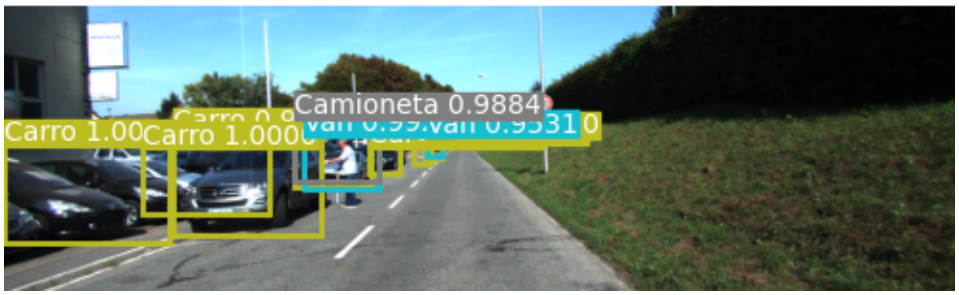
(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.15: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3



(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.16: Imágenes comparativas finales entre arquitecturas .

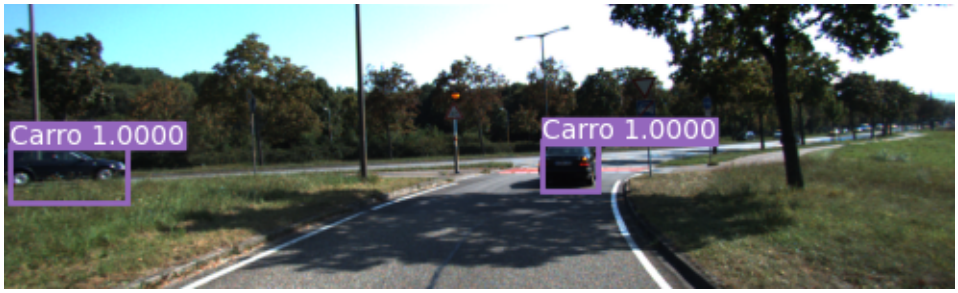
(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.17: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3

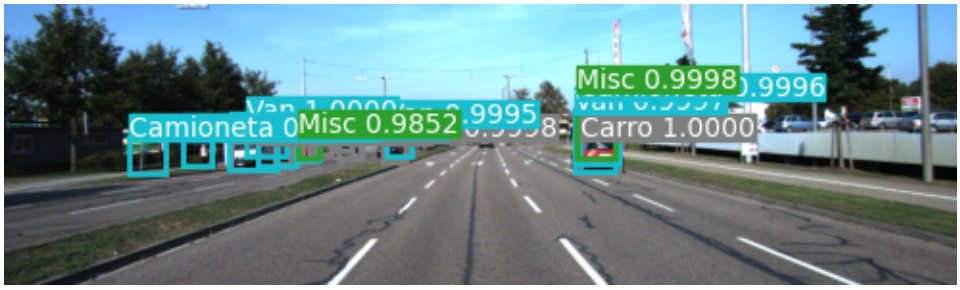


(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.18: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3



(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.19: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.20: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.21: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3

(b) YOLOv3-Híbrida

(c) Imagen sin detección

Figura 5.22: Imágenes comparativas finales entre arquitecturas .

(a) YOLOv3

(b) YOLOv3-Híbrida

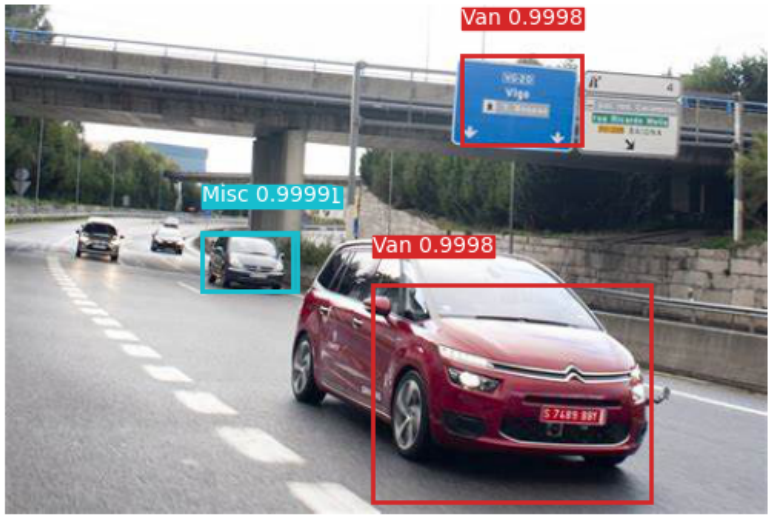
(c) Imagen sin detección

Figura 5.23: Imágenes comparativas finales entre arquitecturas .

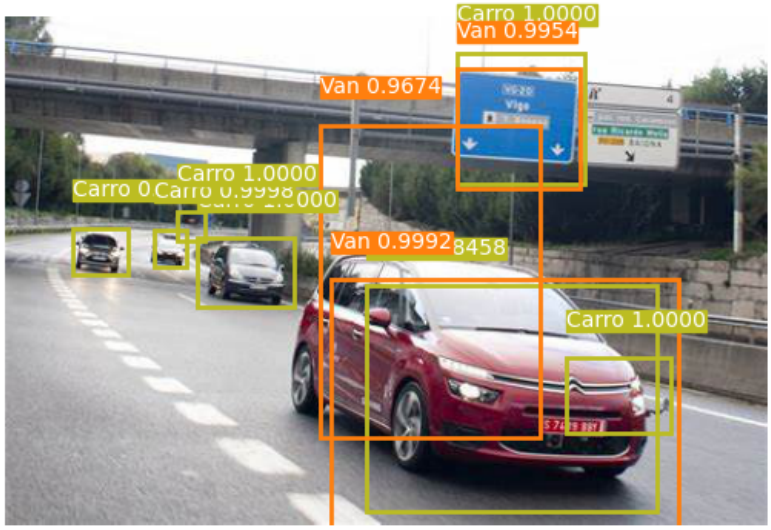
Figura 5.24: Imágenes comparativas entre YOLOv3/ YOLOv3-Híbrida .

5.4.1 Imágenes comparativas fuera de la validación

Se introdujeron imágenes de prueba al algoritmo en diferentes condiciones y vemos que la YOLO-hibrida detecta la mayoría de automóviles a diferencia que la YOLO simple, pero aun así teniendo algunos falsos positivos en la imagen figura 5.25 y figura 5.24.



(a) YOLOv3



(b) YOLOv3-Híbrida

Figura 5.25: Imágenes comparativas finales entre arquitecturas .

6. CONCLUSIONES Y TRABAJO FUTURO

6.1. Conclusiones

En base a los resultados y el trabajo realizado para llevar a cabo este proyecto de tesis se tiene las siguientes conclusiones:

Mediante los años han avanzado las técnicas propuestas inicialmente por Yen en 1990 hasta llegar a un punto sumamente importante para la inteligencia artificial donde a día de hoy existen diversas técnicas derivadas de las redes convolutivas. Este trabajo partía del silogismo donde dos métodos de inteligencia artificial podrían aportar y sumar para mejorar la detección de objetos en imágenes de carretera.

Las redes neuronales convolutivas investigadas en el estado del arte muestran en la practica los resultados competitivos esperados donde se partía a elegir el mejor candidato en base de estos resultados. Se determino que la red con mejores resultados, YOLOv3 con un mAP de 11.8 % en detección se utilizaría como arquitectura base para las pruebas y experimentos posteriores.

Los algoritmos de búsqueda fueron un reto inminente ya que al igual como las arquitecturas de redes convolutivas, existen cientos de estos algoritmos y se tuvo que indagar en sus cualidades y desventajas de cada uno de ellos, nos centramos en los algoritmos de búsqueda de gran escala respecto a sujetos iniciales, aunado a esto nos centramos en los algoritmos estocásticos como lo son las actualmente conocidos como “algoritmos metaheurísticos” basados en la naturaleza de los cuales se definieron los algoritmos de enjambre/horda. Al decidir sobre este tipo de algoritmos de búsquedas se eligieron 4 posibles candidatos basados en el estado del arte: Optimización por enjambre de partículas (PSO), Optimización de ballena jorobada (WOA), Algoritmos Genéticos (GA), Optimizador de lobo gris (GWO).

Se realizaron pruebas de menor escala para la creación de un nuevo algoritmo donde se crearon y probaron distintas formas combinadas de estas redes convolutivas y se concluyo en un algoritmo de ajuste fino de los parámetros de la red neuronal para aumentar las métricas de evaluación deseadas.

A lo largo de esta investigación se analizó el uso de metaheurísticas para el ajuste de parámetros en las redes neuronales convolucionales. Dichos modelos híbridos presentan una mejora considerable para algunas de las métricas utilizadas en el campo de la clasificación como sensitividad, precisión, exactitud y F1. Siendo esta última una de las más relevantes en la clasificación de imágenes por medio de CNNs la cual demostró una diferencia significativa favorable en el modelo CNN-WOA. La metodología aquí implementada se aventaja de la búsqueda exhaustiva de estos algoritmos de optimización ampliando el espacio de búsqueda. En este trabajo de investigación se experimentó con la intercalación de capas utilizando el método CNN y el algoritmo de optimización con el fin de mejorar el desempeño de la red por época, finalmente se decidió hacer el ajusto fino en la última época del entrenamiento.

Se vio una mejora en la detección de objetos de las imágenes propuestas y una mejora en el mAP de la red YOLO híbrida de un 5.7 % teniendo un total de 17.5 % , se concluye que los algoritmos metaheurísticos pueden utilizarse de diversos métodos para aumentar la precisión en las redes neuronales dependiendo de la arquitectura y algoritmo a combinar, como de la base de datos tratada.

6.2. Trabajo futuro

Como trabajo futuro, se plantean algunas mejoras o pruebas extra a los algoritmos propuestos:

- Realizar pruebas con diferentes arquitecturas neuronales y distintos algoritmos metaheurísticos, y comparar su rendimiento en función de los diversos beneficios que las arquitecturas y algoritmos nos puedan proporcionar.
- Aumentar el tamaño de imagen en la entrada del modelo neuronal y re-evaluar estos algoritmos con técnicas de validación cruzada.

- Conjunto de entrenamiento. Entrenar y evaluar con distintas imágenes, categorías y áreas de interés y comparar los resultados globales.
- Validar diferentes modelos CNN de menor costo computacional para implementación en un sistema embebido y evaluar el costo beneficio de este.

BIBLIOGRAFÍA

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv*, pages arXiv–2004, 2020.

[2] Eric Bonabeau, Directeur de Recherches Du Fnrs Marco, Marco Dorigo, Guy Théraulaz, Guy Theraulaz, et al. *Swarm intelligence: from natural to artificial systems*. Number 1. Oxford university press, 1999.

[3] Jason Brownlee. Machine learning mastery with python. *Machine Learning Mastery Pty Ltd*, 527:100–120, 2016.

[4] D. Erhan C. Szegedy, A. Toshev. “deep neural networks for object detection". *NIPS*, 2013.

[5] Zhaowei Cai, Quanfu Fan, Rogerio S Feris, and Nuno Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *European conference on computer vision*, pages 354–370. Springer, 2016.

[6] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals using stereo imagery for accurate object class detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(5):1259–1272, 2017.

[7] J Deng, A Berg, S Satheesh, H Su, A Khosla, and L Fei-Fei. Ilsvrc-2012, 2012. *URL http://www. image-net. org/challenges/LSVRC*, 3, 2012.

[8] Li Deng and Dong Yu. Deep learning for signal and information processing. *Microsoft Research Monograph*, 2013.

[9] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):743–761, 2011.

[10] Russell Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 39–43. Ieee, 1995.

[11] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[12] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[14] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[15] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[16] Georgia Gkioxari, Ross Girshick, and Jitendra Malik. Contextual action recognition with r* cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1080–1088, 2015.

[17] Google. Tensorflow object detection model zoo. *Tensorflow,*, 2019.

[18] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Mohammad Hossin and MN Sulaiman. A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2):1, 2015.

[21] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7310–7311, 2017.

[22] Will Koehrsen. Overfitting vs. underfitting: A complete example. *Towards Data Science*, 2018.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[24] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[25] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020.

[26] Michael Lones. Sean luke: essentials of metaheuristics, 2011.

[27] John McCall. Genetic algorithms for modelling and optimisation. *Journal of computational and Applied Mathematics*, 184(1):205–222, 2005.

[28] Seyedali Mirjalili and Andrew Lewis. The whale optimization algorithm. *Advances in engineering software*, 95:51–67, 2016.

[29] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. Grey wolf optimizer. *Advances in engineering software*, 69:46–61, 2014.

[30] Tom M Mitchell et al. Machine learning, 1997.

[31] Ibrahim H Osman and Gilbert Laporte. Metaheuristics: A bibliography, 1996.

[32] Singiresu S Rao. *Engineering optimization: theory and practice*. John Wiley & Sons, 2019.

[33] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[34] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[35] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[36] LM Rere, Mohamad Ivan Fanany, and Aniati Murni Arymurthy. Metaheuristic algorithms for convolution neural network. *Computational intelligence and neuroscience*, 2016, 2016.

[37] Prasun Roy, Subhankar Ghosh, Saumik Bhattacharya, and Umapada Pal. Effects of degradations on deep neural network architectures. *arXiv preprint arXiv:1807.10108*, 2018.

[38] Pierre Sermanet, Koray Kavukcuoglu, Soumith Chintala, and Yann LeCun. Pedestrian detection with unsupervised multi-stage feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3626–3633, 2013.

[39] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[40] Arie Rachmad Syulistyo, Dwi Marhaendro Jati Purnomo, Muhammad Febrian Rachmadi, and Adi Wibowo. Particle swarm optimization (pso) for training optimization on convolutional neural network (cnn). *Jurnal Ilmu Komputer dan Informasi*, 9(1):52–58, 2016.

[41] Nguyen Van Thieu. A collection of the state-of-the-art meta-heuristics algorithms in python: Mealpy, 2020.

[42] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[43] Cornell University. arxiv in numbers 2020. urlhttps://arxiv.org/, May 2020.

[44] Bin Wang, Bing Xue, and Mengjie Zhang. Particle swarm optimisation for evolving deep neural networks for image classification by evolving and stacking transferable blocks. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2020.

[45] Daniel S Weile and Eric Michielssen. Genetic algorithm optimization applied to electromagnetics: A review. *IEEE Transactions on Antennas and Propagation*, 45(3):343–353, 1997.

[46] Fan Yang, Wongun Choi, and Yuanqing Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2129–2137, 2016.

[47] J Javier Yebes, Luis M Bergasa, and Miguel García-Garrido. Visual object recognition with 3d-aware features in kitti urban scenes. *Sensors*, 15(4):9228–9250, 2015.

[48] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[49] Guoqiang Peter Zhang. Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30(4):451–462, 2000.

1. ANEXOS

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE LENGUAS Y LETRAS

A QUIEN CORRESPONDA:

La que suscribe, Directora de la Facultad de Lenguas y Letras, hace **C O N S T A R** que

TREVIÑO VALDES GERARDO

Presentó el **Examen de Manejo de la Lengua** efectuado el día diez de noviembre de dos mil veintiuno, en el cual obtuvo la siguiente calificación:

8-

Se extiende la presente a petición de la parte interesada, para los fines escolares y legales que le convengan, en el Campus Aeropuerto de la Universidad Autónoma de Querétaro, el día veinticinco de noviembre de dos mil veintiuno.

A t e n t a m e n t e,
"Enlazar Culturas por la Palabra"

DRA. ADELINA VELÁZQUEZ HERRERA

AVH/japa*CL*FLL-C.-2303

Figura A.1: Constancia de manejo de lengua extranjera

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO
FACULTAD DE LENGUAS Y LETRAS

A QUIEN CORRESPONDA:

La que suscribe, Directora de la Facultad de Lenguas y Letras, hace **C O N S T A R** que

TREVIÑO VALDES GERARDO

Presentó y acreditó el **Examen de Comprensión de Textos en Inglés** efectuado el día dieciocho de octubre de dos mil veintiuno.

Se extiende la presente a petición de la parte interesada, para los fines escolares y legales que le convengan, en el Campus Aeropuerto de la Universidad Autónoma de Querétaro, el día veinticuatro de noviembre de dos mil veintiuno.

A t e n t a m e n t e,
"Enlazar Culturas por la Palabra"

DRA. ADELINA VELÁZQUEZ HERRERA

AVH/japa*CL*FLL-C.-2192

Figura A.2: Constancia de comprensión de textos de lengua extranjera

CENTRO DE INVESTIGACIÓN EN MATEMÁTICAS APLICADAS
UNIVERSIDAD AUTÓNOMA DE COAHUILA

"2021, Año del reconocimiento al trabajo del personal de salud por su lucha contra el COVID-19"

Saltillo, Coahuila a 12 de agosto 2021

Dr. Saúl Tovar Arriaga
Coordinador de la Maestría en Ciencias en Inteligencia Artificial
Facultad de Ingeniería
Universidad Autónoma de Querétaro
Presente;

Por medio de la presente se hace CONSTAR que el estudiante **Gerardo Treviño Valdés** perteneciente a la Maestría en Ciencias en Inteligencia Artificial de la Universidad Autónoma de Querétaro, cursó satisfactoriamente la materia de ***Tópicos Selectos (Optimización)*** en el Centro de Investigación en Matemáticas Aplicadas de la Universidad Autónoma de Coahuila, impartida por el Dr. Jesús Alejandro Navarro Acosta obteniendo una calificación aprobatoria de 10.0

Sin otro particular por el momento, le envío un cordial saludo.

A T E N T A M E N T E
"EN EL BIEN FINCAMOS EL SABER"

CENTRO DE INVESTIGACIÓN EN MATEMÁTICAS APLICADAS

DRA. IRMA DELIA GARCÍA CALVILLO
DIRECTORA DEL CIMA

Figura A.3: Constancia de estancia académica en CIMA

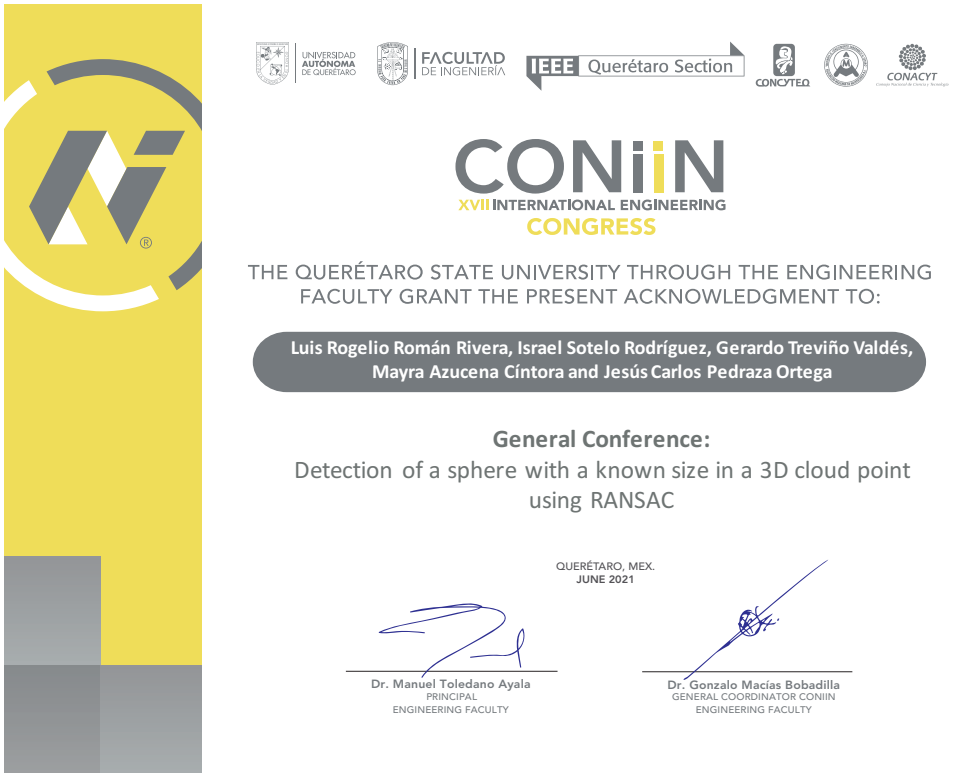


Figura A.4: Constancia de presentación del articulo en CONIIN 2021

Figura A.5: Constancia de presentación del articulo en COMIA 2021

Metaheurísticas y CNN: comparación de modelos híbridos para mejorar la clasificación de imágenes

Gerardo Treviño-Valdés[1], Jesús Alejandro Navarro-Acosta[2], Jesús Carlos Pedraza-Ortega[1], Marco Antonio Aceves-Fernández[1], Saúl Tovar-Arriaga[1]

[1] Universidad Autónoma de Querétaro, Facultad de Ingeniería, Querétaro, Querétaro, México.
[2] Universidad Autónoma de Coahuila, Centro de Investigación en Matemáticas Aplicadas, Saltillo, Coahuila, México
geratrevino115@gmail.com, alexnav24@gmail.com, caryoko@yahoo.com, marco.aceves@gmail.com, saul.tovar@uaq.mx

Resumen. En este artículo presenta la comparación entre una red neuronal convolucional estándar y una variante híbrida de la misma incluyendo algoritmos metaheurísticos para la clasificación de imágenes. Se pretende mejorar la clasificación del algoritmo implementando un ajuste en los hiperparámetros de la red neuronal convolucional con el algoritmo híbrido propuesto. Los algoritmos metaheurísticos aquí presentados son: Algoritmos Genéticos, Optimización por enjambre de partículas, Optimización de Lobo Gris y Optimización de Ballena Jorobada. Los resultados muestran que la metodología implementada es capaz de aumentar la exactitud en la clasificación, como de algunas otras de las métricas para evaluar la clasificación.

Palabras clave: Red Neuronal Convolucional, Hibridismo, Metaheurísticas, Clasificación de imágenes.

Metaheuristics and CNN: Hybrid model comparison to improve image classification

Abstract. In this paper presents the comparison between a standard convolutional neural network and a hybrid variant of it, including metaheuristic algorithms for image classification. The aim is to improve the classification of the algorithm by implementing an adjustment in the hyperparameters of the convolutional neural network with the proposed hybrid algorithm. The metaheuristic algorithms presented here are: Genetic Algorithms, Particle Swarm Optimization, Gray Wolf Optimization, and Humpback Whale Optimization. The results show that the methodology implemented is able for increasing the accuracy in the classification, as well as some other of the metrics to evaluate the classification.

Keywords: Convolutional Neural Network, Hybridism, Metaheuristics, Image clasification.

Figura A.6: Articulo presentado en COMIA 2021: pagina 1

1. Introducción

Las redes neuronales convolucionales (CNN) han demostrado un rendimiento en el procesamiento de imágenes al aumentar su desempeño en tareas como clasificación, detección de objetos, entre otras. Así como la capacidad de adaptación de sus modelos [1].

Los enfoques de detección de objetos basados en características y clasificadores de aprendizaje automático han sido muy fructíferos hasta tiempos recientes. Cuando se aplican a diferentes tareas o se adaptan para desafíos adicionales, estos requieren de un ajuste de parámetros y una reducción dimensional para lograr un rendimiento aceptable.

Las redes neuronales convolucionales, en los últimos años, han propiciado un importante avance en tareas que involucran visión artificial, tales como clasificación, localización, detección y segmentación de objetos, descripción de escenas, entre otras, ya sea en imágenes o vídeo. Los resultados que se obtienen actualmente se puedan emplear en una gran variedad de aplicaciones.

Sin embargo, el desempeño de algoritmos como las CNN en la detección de objetos depende en gran medida de la elección y ajuste de diversos hiperparámetros que pueden determinar la tasa de aprendizaje de las mismas, por tal motivo en este artículo se presenta la combinación de dos técnicas de inteligencia artificial como lo son los algoritmos metaheurísticos y las redes neuronales convolucionales para realizar un entrenamiento más robusto en comparación del entrenamiento estándar de estas redes [2]. Y de esta forma lograr un modelo eficiente y eficaz para la clasificación de objetos. Con el fin de ajustar los hiperparámetros de la CNN se implementan y comparan cuatro metaheurísticas ampliamente utilizados en el estado del arte como son optimización por enjambre de partículas (PSO), algoritmos genéticos (GA), optimización de lobos grises (GWO) y optimización de ballena jorobada (WOA).

2. Marco Teórico

2.1. Redes Neuronales Convolucionales

Las redes convolucionales, son un tipo especializado de red neuronal para procesar datos que tiene una topología similar a una cuadrícula (matriz). Las redes convolucionales han tenido un éxito extraordinario en aplicaciones prácticas. Las redes convolucionales son simplemente redes neuronales que usan la convolución en lugar de la multiplicación general de matrices en al menos una de sus capas. Estas son muy potentes para todo lo que tiene que ver con el análisis de imágenes, debido a que son capaces de detectar características simples como por ejemplo detección de bordes, líneas, etc. Y extraer características más complejas hasta llegar a su objetivo. Consta de capas convolucionales y de reducciones alternadas, y en sus capas finales tiene capas de conexión total como una red perceptrón multicapa Figura 4.

En la convolución se realizan operaciones de productos y sumas entre la capa de partida y los n filtros que genera un mapa de características. Las

Figura A.7: Articulo presentado en COMIA 2021: pagina 2

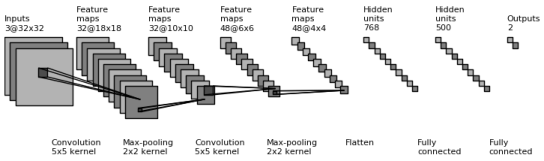


Fig. 1. Arquitectura de las CNN [3].

características extraídas corresponden a cada posible ubicación del filtro en la imagen original [3].

La ventaja es que el mismo filtro sirve para extraer la misma característica en cualquier parte de la entrada, con esto que consigue reducir el número de conexiones y el número de parámetros a entrenar en comparación con una red multicapa de conexión total [4].

2.2. Metaheurísticas

Las técnicas de optimización metaheurísticas se han inspirado principalmente en conceptos muy simples. Estos algoritmos suelen basarse en fenómenos físicos, comportamientos de animales o conceptos evolutivos. Tienen mayor flexibilidad a diferentes problemas sin ningún cambio especial en la estructura del algoritmo, ya que en su mayoría asumen los problemas como cajas negras. En otras palabras, solo las entradas y salidas de un sistema son importantes para una metaheurística [5].

Las metaheurísticas tienen capacidades superiores para evitar los óptimos locales en comparación con las técnicas de optimización convencionales. Esto se debe a la naturaleza estocástica de las metaheurísticas que les permiten evitar el estancamiento en las soluciones locales y adentrarse extensamente en todo el espacio de búsqueda. El cual para problemas reales suele ser desconocido o muy complejo y con una gran cantidad de óptimos locales, por lo que las metaheurísticas tienen buen desempeño únicamente teniendo claro el objetivo [6].

Algoritmos Genéticos (GA) El algoritmo genético es una metaheurística inspirada en el proceso de selección natural creado por John Henry Holland en el año 1970, surgió con este algoritmo base de muchas representaciones metaheuristicas [7]. Un algoritmo genético estándar requiere dos requisitos previos, es decir, una representación genética del dominio de la solución y una función de aptitud para evaluar a cada individuo. La idea central del algoritmo genético es permitir que los individuos evolucionen a través de algunas operaciones genéticas

Figura A.8: Articulo presentado en COMIA 2021: pagina 3

como se muestra en el algoritmo. Las operaciones populares incluyen selección, mutación, cruce. El proceso de selección nos permite preservar a los individuos fuertes mientras eliminamos a los débiles. Las formas de realizar la mutación y el cruce a menudo se basan en las propiedades del problema específico [8].

Optimización por enjambre de partículas (PSO) El algoritmo de optimización con enjambre de partículas (PSO) fue desarrollado por J. Kennedy y R. C. Eberhart [9], el cual se basa del comportamiento de parvadas de aves, colonias de abejas, bancos de peces, entre otros. Se puede utilizar para resolver problemas de optimización que carecen de conocimiento del dominio. La población está constituida por una serie de partículas. Cada uno de ellas representa un individuo. Busca la mejor solución actualizando velocidad y vector de partículas de acuerdo con las ecuaciones (1) y (2). Donde v_{id} representa la velocidad de la partícula i en la d-ésima dimensión, x_{id} representa la posición de la partícula i. P_{id} y P_{gd} son los mejores locales y el mejor global, r_1, r_2 son números aleatorios entre 0 y 1, mientras que w, c_1 y c_2 son peso de inercia y coeficientes de aceleración para explotación y aceleración para los coeficiente de exploración, respectivamente.

$$V_{id}(t+1) = w * v_{id}(t) + c_1 * r_1 * (P_{id} - x_{id}(t)) + c_2 * r_2 * (P_{gd} - x_{id}(t)) \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (2)$$

Optimizador lobo gris (GWO) El algoritmo metaheurístico del lobo gris salió a la luz en 2014 por obra de Seyedali Mirjalili [10]. Donde se muestra el comportamiento de este animal su forma de caza y su conducta social y de particular interés es que tienen una jerarquía social dominante muy estricta donde llamamos a estos grupos como alfa, beta, delta y omega, cada una de estos grupos juega un papel importante en la manada. Para modelar matemáticamente la jerarquía social de los lobos, consideramos la solución más adecuada como la alfa (a). En consecuencia, la segunda y tercera mejores soluciones se nombran beta (b) y delta (d) respectivamente. Se supone que el resto de las soluciones candidatas como omega (x). En el algoritmo GWO la búsqueda está guiada por a, b y d. Los lobos x siguen a estos tres lobos y así estos rodean a sus presas durante la caza. Matemáticamente se representa en la ecuación (3) y (4) Donde t indica la iteración actual, $\vec{A}$ y $\vec{C}$ son vectores de coeficientes, $\vec{X_p}$ es el vector de posición de la presa, $\vec{X}$ indica el vector de posición de un lobo gris.

$$\vec{D} = |\vec{C} \cdot \vec{X_p}(t) - \vec{X}(t)| \quad (3)$$

$$\vec{X}(t+1) = \vec{X_p}(t) - \vec{A} \cdot \vec{D} \quad (4)$$

Donde los componentes de $\vec{a}$ se reducen linealmente de 2 a 0 en el transcurso de las iteraciones y r_2, r_2 son vectores aleatorios en [0, 1].

Figura A.9: Articulo presentado en COMIA 2021: pagina 4

$$\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a} \tag{5}$$

$$\vec{C} = 2 \cdot \vec{r_2} \tag{6}$$

Algoritmo de optimización de ballenas (WOA) El algoritmo de optimización de la ballena jorobada se presenta en 2017 por Seyedali Mirjalili [11]. Se puede interpretar como una modificación al algoritmo del lobo gris (GWO) donde en este caso representa de igual manera su comportamiento de caza, las ballenas jorobadas pueden reconocer la ubicación de sus presas y rodearlas. Las ecuaciones principales son las descritas en el algoritmo GWO a diferencia del este método, una ecuación en espiral es creado entre la posición de la ballena y la presa para imitar el movimiento en forma de hélice de las ballenas jorobadas dada la siguiente ecuación (9).

$$\vec{X}(t+1) = \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X}*(t) \tag{7}$$

Donde $\vec{D'}$ indica la distancia de la ballena a la presa (la mejor solución obtenida hasta ahora), b es una constante para definir la forma de la espiral logarítmica, l es un valor aleatorio de $[-1, 1]$ y $\cdot$ es una multiplicación elemento por elemento. Aquí se tiene en cuenta el vector donde una ballena crea un círculo que se contrae para llegar a su presa, se asume una probabilidad del 50 por ciento para elegir esta distinción al modelo circular GWO.

$$\vec{X}(t+1) = \begin{cases} \vec{X}*(t) - \vec{A} \cdot \vec{D} & \text{si} \quad p < 0{,}5 \\ \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X} * (t) & \text{si} \quad p \geq 0{,}5 \end{cases} \tag{8}$$

Donde p es un número aleatorio uniforme de [0,1].

3. Materiales y métodos

3.1. Metaheurístico adaptado para el ajuste fino de hiperparámetros CNN

El objetivo del algoritmo en este artículo es aprovechar el beneficio de la búsqueda exhaustiva de los algoritmos metaheurísticos sobre los hiperparámetros de la red neuronal convolucional centrándose en las capas densas donde se trabaja con una extensa cantidad de hiperparámetros.

Las capas convolucionales en los algoritmos de clasificación como detección de objetos son de suma relevancia. Estas capas juegan un papel importante en la parametrización de las entradas (imágenes) y dan paso a las capas densas para realizar las operaciones necesarias y llegar a la clasificación. Los algoritmos metaheurísticos tienen mecanismos libres de derivación. A diferencia de los enfoques de optimización basados en gradientes que por su practicidad se utilizan en redes neuronales y redes neuronales convolucionales. Las metaheurísticas

Figura A.10: Articulo presentado en COMIA 2021: pagina 5

optimizan estocásticamente los problemas. El proceso de optimización comienza con soluciones aleatorias y no es necesario calcular la derivada de los espacios de búsqueda para encontrar el óptimo. Esto hace que la metaheurística sea adecuada para problemas reales con información desconocida o compleja [12].

Se tiene claro que la combinación de estos métodos puede realizarse de infinitas formas, por ejemplo: utilizar el metaheurístico después de cada etapa de entrenamiento o época, en determinadas épocas del entrenamiento, únicamente para su ajuste final, por mencionar algunas. Todo esto se determina arbitrariamente dependiendo de la complejidad de la arquitectura, efectividad de convergencia, el tiempo de entrenamiento, entre otras [13].

En este caso se examina únicamente el algoritmo descrito en el pseudocódigo del Algoritmo 1 donde al finalizar cada época se ejecutan los algoritmos de optimización para el ajuste fino de estos hiperparámetros sumado las operaciones que se ejecutan a través de las épocas en la CNN.

Se genera un punto de partida para homogeneizar las condiciones iniciales de cada algoritmo híbrido iniciando con una época base para todos los algoritmos comparados.

Algoritmo 1: Algoritmo híbrido CNN-Metaheurístico

Entrada: Base de datos: *clases Imagenes*, épocas CNN *epoch*, iteraciones Metaheurística *it* ;

1. Preprocesamiento ← *clases Imagenes*
2. modelo ← Se crea el modelo CNN;
3. **while** i <= *epoch*
4. Época CNN ← Ejecución de una época en el entrenamiento CNN
5. *get weights*→ hiperparámetros
6. Inicio Metaheurístico:
7. Parámetros iniciales. ← *it*
8. Ingreso de hiperparámetros al algoritmo ← se evita empeorar la salida
9. **while** j <= *it*
10. Generación aleatoria de individuos
11. Ejecución
12. Aptitud de los individuos ← $Max\ \ f(x) = w1 * Ex + w2 * F1$
13. Mejores individuos
14. **end while**
15. **return:** Hiperparámetros
16. *set weights*← hiperparámetros
17. **end while**
18. modelo entrenado
19. Evaluación
20. **return:** clasificación de imagen

Figura A.11: Articulo presentado en COMIA 2021: pagina 6

Tabla 1. Clases y numero de imágenes del dataset Natural Images [14].

Clase	Nombre	Imágenes por clase
1	Avión	727
2	Automóvil	968
3	Gato	885
4	Perro	702
5	Flor	843
6	Fruta	1000
7	Motocicleta	788
8	Persona	986

3.2. Base de datos

La base de datos Natural Images empleada en este artículo consta de 6899 imágenes distintas divididas en 8 clases diferentes [14]. En la Figura 2 se muestra una imagen representativa de las imágenes a trabajar observamos que son de diferentes tamaños y estilos. Las clases que contiene el dataset se muestran en la Tabla 2, con su respectivo nombre y el número de imágenes por clase, para poder emplear esta base de datos se toma el total de imágenes y aplica un pequeño pre-procesamiento al normalizar a 80 x 80 pixeles.

Fig. 2. Muestra de la base de datos Natural Images [14] .

3.3. Implementación del algoritmo híbrido

Parámetros. La arquitectura utilizada en este artículo se basa en la propuesta original ilustrada en la Figura 1 y se representa en la Tabla 2 de manera detallada, similar a la LeNet-5 propuesta por Yann LeCun [3]. Se realizará el ajuste de

Figura A.12: Articulo presentado en COMIA 2021: pagina 7

Tabla 2. Clases y numero de imágenes del dataset

Capa	Kernel	Parámetros
Conv2d	3x3	640
Max Pooling	2x2	-
Conv2d	3x3	36928
Max Pooling	2x2	-
Flatten	-	-
Dense1	32	663584
Dense2	32	1056
Dense3	8	264

hiperparámetros en las capas densas durante 10 épocas de entrenamiento. Por lo tanto, los algoritmos descritos requieren parámetros de inicialización para su funcionamiento. Cada uno de los metaheurísticos se estableció con 30 individuos, 10 iteraciones por cada época de la red neuronal convolucional, una ventana de 25, y una longitud del problema igual a la cantidad de hiperparametros a evaluar después de las capas convolucionales en este caso particular sería de 664,896 variables.

3.4. Función Objetivo

En este trabajo se propone una función objetivo a maximizar (Ec. 9) por parte de los enfoques híbridos presentados. Esta, integra la exactitud Ex (Ec. 10) y la medición F1-score $F1$ (Ec. 13), las cuales son dos de las métricas más utilizadas en el campo de la clasificación.

$$Max \quad f = w1 * Ex + w2 * F1 \tag{9}$$

Donde $w1 = 0.4$ y $w2 = 0.6$, son ponderaciones de la importancia de dichas métricas. Estos valores se obtuvieron mediante pruebas exhaustivas donde se observó que la CNN presenta mejor desempeño usando dicha configuración. Para fines de comparación, en los resultados se presentan por separado los valores de Exactitud, F1-score, así como los de precisión Pre (Ec. 11) y sensitividad $Sens$ (Ec. 12). Métricas que componen la F1-Score [15]. Donde TP es Verdadero Positivo (abreviado por sus siglas en inglés), TF es Verdadero Negativo, FP es Falso Positivo y FN corresponde a Falso Negativo.

$$Ex = \frac{TP + TF}{TP + TF + FP + FN} \tag{10}$$

$$Pre = \frac{TP}{TP + FP} \tag{11}$$

$$Sens = \frac{TP}{TP + FN} \tag{12}$$

Figura A.13: Articulo presentado en COMIA 2021: pagina 8

$$F1 = \frac{2 * Pre * Sens}{Pre + Sens} = \frac{2 * TP}{2 * TP + FP + FN} \tag{13}$$

4. Resultados experimentales y discusión

Una vez implementada la metodología propuesta se obtienen los datos presentados en la Tabla 3 para los máximos valores alcanzados por los 4 algoritmos híbridos y la CNN estándar de las métricas antes descritas.

El entrenamiento de la red neuronal convolucional se realizó con una relación 80/20 en los datos de entrenamiento y prueba, se realizaron 10 distintos experimentos hasta alcanzar los máximos valores que podemos apreciar en dicha tabla. Comparando los modelos híbridos con el método CNN obtenemos que en la precisión los mejores resultados fueron logrados por el método estándar de la CNN con un diferencia de 0.06 % al segundo mejor y del 6.44 % al peor. La sensitividad fue mejorada por CNN-WOA en 32.25 % respecto a CNN. En la evaluación F1-score se destaca nuevamente WOA con una diferencia de 40.0 %. Finalmente para la exactitud se mejoró por el algoritmo CNN-GWO un 1.53 % al modelo estándar CNN.

Como se mostró en los resultados al modificar la arquitectura de la CNN utilizando la metodología híbrida propuesta en este trabajo se observa una mejora en algunas de las métricas pero la más notoria es F1-score lo cual nos indica que el utilizar algoritmos híbridos de inteligencia artificial puede mejorar la robustez del modelo final por lo tanto mejorar la clasificación de imágenes.

Analizando los resultados obtenidos podemos concluir que algunos de los resultados en los modelos hibridos no superaron el modelo estándar, esto podría deberse a la naturaleza de los mismos, debido a las pocas iteraciones en las pruebas realizadas. Y se concluye para la tarea de clasificación aquí presentada el algoritmo con mejor desempeño es CNN-WOA.

En la Figura 4 se representa gráficamente la convergencia al mayor valor de exactitud obtenido en la experimentación a través de las épocas establecidas para cada uno de los métodos. El comportamiento de los algoritmos durante las épocas se intercala utilizando el método CNN y el algoritmo metaheurístico. Podemos notar que existen fluctuaciones en estos cambios siendo la mas notoria en CNN-PSO, a través de las épocas cae la puntuación y en la siguiente itración es recuperada. Esto podría deberse a la elección de los hiperparámetros de la entrada a la CNN, no logra aprovechar las propiedades del algoritmo híbrido por sus bajas iteraciones en la etapa metahuerística. En la Figura 3 observamos un ejemplo de la convergencia en la primera época de la CNN evaluada por los metaheurísticos, desde esta instancia se aprecia como destaca el algoritmo WOA.

5. Conclusiones y trabajo futuro

A lo largo de esta investigación se analizó el uso de metaheurísticas para el ajuste de hiperparámetros en las redes neuronales convolucionales. Dichos modelos híbridos presentan una mejora considerable para algunas de las métricas

Figura A.14: Articulo presentado en COMIA 2021: pagina 9

Tabla 3. Evaluación de los algoritmos

Método	Exactitud	Precisión	Sensitividad	F1-score
CNN	0.8202	**0.9962**	0.1178	0.2054
CNN-GA	0.8224	0.9318	0.0901	0.1605
CNN-PSO	0.8260	0.9740	0.1079	0.1911
CNN-GWO	**0.8355**	0.9740	0.2450	0.3864
CNN-WOA	0.8347	0.9956	**0.4403**	**0.6053**

utilizadas en el campo de la clasificación como sensitividad, precisión, exactitud y F1. Siendo esta última una de las más relevantes en la clasificación de imágenes por medio de CNNs la cual demostró una diferencia significativa favorable en el modelo CNN-WOA. La metodología aquí implementada se aventaja de la búsqueda exhaustiva de estos algoritmos de optimización ampliando el espacio de búsqueda. En este trabajo de investigación se experimentó con la intercalación de capas utilizando el método CNN y el algoritmo de optimización con el fin de mejorar el desempeño de la red por época.

Como trabajo futuro se plantea realizar el ajuste únicamente en la última capa de la red neuronal convolucional implicando cambios relevantes de los hiperparámetros para mejorar la eficiencia general en la red neuronal. Acceder a equipo de computo robusto en la cual se puedan realizar mayor número iteraciones en los algoritmos híbridos presentados obteniendo mejor desempeño en las variantes GA y PSO. De igual forma se plantea la implementación de estos modelos híbridos en distintas arquitecturas complejas de las CNN, y aplicarlo en diversas áreas como lo son la detección de objetos en imágenes y vídeo.

Figura A.15: Articulo presentado en COMIA 2021: pagina 10