UNIVERSIDAD DEL PAPALOAPAN

Campus Loma Bonita

INGENIERÍA EN COMPUTACIÓN

Metodología semi-automática para la clasificación de aves basada en características morfológicas

TESIS

QUE PARA OBTENER EL TÍTULO DE

INGENÍERA EN COMPUTACIÓN

PRESENTA

MARÍA MERCEDES VIDAL RAMÍREZ

DIRECTORA DE TESIS:

DRA. NANCY PÉREZ CASTRO

LOMA BONITA, OAXACA, 2023

Universidad del Papaloapan

FECHA:	22 de Junio del 2023
ÁREA:	Vice-Rectoría Académica
OFICIO NÚMERO:	UNPA/VRA/172/2023
ASUNTO:	Autorización de Impresión de tesis.

C. MARIA MERCEDES VIDAL RAMIREZ
P R E S E N T E:

En base al artículo 120 del reglamento de alumnos, por medio de la presente se aprueba la impresión de la tesis titulada **"*Metodología Semi Automática para la Clasificación de Aves basada en Características Morfológicas*"** así como la programación del examen profesional bajo la dirección de la Dra. Nancy Pérez Castro.

Sin más por el momento aprovecho la ocasión para enviarle un cordial saludo.

Atentamente.
terra ubérrima, mens aperta
Bou Lo-tama, chi jí jú

MC. HÉCTOR LÓPEZ ARJONA
Vice-Rector Académico.

VICE-RECTORIA ACADEMICA

C.c.p. Dr. Sergio Fabián Ruiz Paz. Jefe de Carrera de la Ing. en Computación
C.c.p. L.P. Yesenia Barrientos Arenal. Jefa del Departamento de Servicios Escolares
C.c.p. Dra. Nancy Pérez Castro. Directora de Tesis.
C.c.p. Archivo.

UNIVERSIDAD DEL PAPALOAPAN

Campus Loma Bonita

Oficio: ICOM/06/002/23
Loma Bonita, Oaxaca, a 21 de junio de 2023

M.E. Yesenia Barrientos Arenal
Jefa del Departamento de Servicios Escolares
PRESENTE

Mediante la presente, le informo que la jefatura de carrera a mi cargo, con visto bueno de la Vice-Rectoría Académica, ha designado a los siguientes profesores como sinodales para examen profesional de la alumna **C. María Mercedes Vidal Ramírez**, quien defenderá su trabajo de tesis titulado ***"Metodología Semi Automática para la Clasificación de Aves basada en Características Morfológicas"***, para obtener el título de Licenciada en Ingeniería en Computación.

Titulares:

Presidente: **Dr. Sergio Fabián Ruiz Paz**
Secretario: **M.C. Ariel López Rodríguez**
Vocal: **Dra. Nancy Pérez Castro**

Suplentes:

Dra. Carolina G. Maldonado Méndez
M.C. José Domingo Juárez Hernández

UNIVERSIDAD DEL PAPALOAPAN
23 JUN 2023
SERVICIOS ESCOLARES

A t e n t a m e n t e
terra uberrima, mens aperta
Bou lo-tama, chi, jí, jú

UNIVERSIDAD DEL PAPALOAPAN
UNPA
terra uberrima, mens aperta
Bou Lo-tama, chi jí jú
OAXACA
INGENIERIA EN COMPUTACION

Dr. Sergio Fabián Ruiz Paz
Jefe de la Carrera de Ingeniería en Computación

M.C. Héctor López Arjona
Vice-Rector Académico

c.c.p. M.C. Héctor López Arjona. Vice-Rector académico. Para su conocimiento
c.c.p. Archivo

UNIVERSIDAD DEL PAPALOAPAN

Campus Loma Bonita

INGENIERÍA EN COMPUTACIÓN

La presente tesis titulada "**METODOLOGÍA SEMI AUTOMÁTICA PARA LA CLASIFICACIÓN DE AVES BASADA EN CARACTERÍSTICAS MORFOLÓGICAS**" presentada por la pasante **MARÍA MERCEDES VIDAL RAMÍREZ**, bajo la dirección de la Dra. Nancy Pérez Castro, ha sido aprobada por la comisión revisora y aceptada como requisito parcial para obtener el título de Ingeniero en Computación.

COMISIÓN REVISORA

DRA. NANCY PÉREZ CASTRO
DIRECTORA DE TESIS

DR. SERGIO FABIAN RUIZ PAZ
PRESIDENTE

M.C. ARIEL LÓPEZ RODRÍGUEZ
SECRETARIO

Loma Bonita, Oaxaca, México. 2023

Índice general

1 Introducción **7**

1.1. Introducción 7
1.2. Planteamiento del problema 8
1.3. Objetivos 9
1.3.1. Objetivo General 9
1.3.2. Objetivos Específicos 9
1.4. Hipótesis 9
1.5. Justificación 10

2 Revisión de la literatura **11**

3 Marco teórico **15**

3.1. Pipeline machine learning 15
3.2. Procesamiento de imágenes 16
3.2.1. Qué es una imagen 16
3.2.2. Qué es el procesamiento digital de imágenes 16
3.2.3. Filtrado 17
3.2.4. Filtro Gaussiano 17
3.2.5. Filtro u Operador Sobel 17
3.3. Segmentación 18
3.3.1. Umbralización 18
3.4. Series de tiempo 19
3.5. Transformaciones de series temporales 20
3.5.1. Aproximación Agregada por Partes (PAA) 20
3.5.2. Aproximación Agregada Simbólica (SAX) 20
3.6. Clasificación 21
3.6.1. Algoritmo KNN 22
3.7. Algoritmo Genético 25
3.8. Ingeniería de software dentro de la IA 26

4 Propuesta de solución **30**

4.1. Adquisición de imágenes 30
4.2. Algoritmo Genético para optimizar hiperparámetros y tareas del modelo de clasificación 32
4.2.1. Representación del cromosoma 32
4.2.2. Codificación Binaria 34
4.2.3. Codificación Real 34
4.3. Función de evaluación o fitness 35
4.4. Selección de operadores 35
4.4.1. Operador de Selección 35
4.4.2. Operador de Cruza 36
4.4.3. Operador de Mutación 36
4.4.4. Recombinación o cambio generacional 36

5 Experimentos y resultados **39**
5.1. Configuración experimental 40
5.2. Comparación del rendimiento de algoritmos con diferentes técnicas de codificación. 40
5.3. Análisis de convergencia 44
5.4. Análisis de la solución real 46

6 Conclusiones y trabajos futuros **54**
6.1. Conclusiones 54
6.2. Trabajos futuros 55

Anexos **56**

A Anexo I: Gráficas de convergencia codificación real **56**

B Anexo II:Gráficas de convergencia codificación binaria **59**

Bibliografia **62**

Agradecimientos

Sin duda estoy completamente agradecida con Dios, porque Él es quien hace todo posible, sin Él nada puedo. Me ha dado el don de la vida, salud, familia y amigos generosos, ha sido mi fortaleza en cada instante y situación de mi vida, antes de rendirme me inspiró a confiar en Él.

A mis padres Norberto y Josefina, por sus grandes esfuerzos en darme una carrera profesional. Gracias papá por ambientar con tu música las noches mientras realizábamos tareas. Gracias por tu inspiración a seguir, por ser un buen maestro, por estar presente, me ha tocado culminar esta etapa sin tí, pero siempre estarás conmigo.

A mis hermanos, Tali, Mary y Gladys les agradezco por su compañia. Mary, gracias por tu apoyo muy valioso, por ayudarme con tus asesorías y por desvelarte en varias ocasiones conmigo, Gladys, gracias por tu apoyo, comprenderme y soportarme. Tus ocurrencias me animan cada día.

A mis queridas señoras, que siempre me guian hacia Dios y con sus oraciones me han acompañado hasta este momento tan especial para mí.

A mi asesora, la Dra. Nancy porque más allá de sus conocimientos compartió conmigo su gran calidad humana, gracias por su apoyo, comprensión, paciencia y generosidad durante todo este tiempo, siempre estaré agradecida con usted.

A mis profesores de la carrera de Ingeniería en Computación por compartir sus conocimientos conmigo durante mi formación en la Universidad del Papaloapan campus Loma Bonita.

Resumen

En esta investigación se utilizó una metodología semi-automática para la clasificación de imágenes de aves a través de algoritmos metaheurísticos, especificamente Algoritmos Genéticos bajo dos tipos de codificación: binaria y real, haciendo una comparación de su rendimiento para determinar cuál de las dos proporciona el modelo más eficiente. Las imágenes fueron extraídas de dos repositorios públicos, CONABIO (Comisión Nacional para el Conocimiento y Uso de la Biodiversidad) y Flickr obteniendo un total de 228 imágenes. En los resultados se obtiene que no hay diferencia significativa entre ambas codificaciones, las gráficas de convergencia indican que ambas codificaciones cumplen con el objetivo de minimizar el error de clasificación, finalmente, se analizaron los métodos y parámetros dados por los modelos del mejor error, el peor error, el próximo a la media y mediana, estos tres últimos casos se muestra que los métodos generados por el algoritmo no varían mucho entre las dos codificaciones, aunque si sus parámetros, en cambio en el mejor error hay variación en ambas codificaciones. En conclusión, la codificación binaria tiene el mejor desempeño al minimizar el error, así como en tiempo de ejecución, se logró hacer en ambas codificaciones la selección de los métodos más óptimos, y la calidad en ambas codificaciones proporcionaron resultados favorables sobre la selección de la cabeza del ave, aunque visualmente la codificación real es la que más se ajusta a este tipo de problemas.

Abstract

This research used a semi-automatic methodology to classify bird images using metaheuristic algorithms, specifically Genetic Algorithms. The performance of two encoding types, binary and mixed, was compared to determine which provides the most efficient model. The images were obtained from two public repositories, CONABIO and Flickr, resulting in 228 images. The results showed that there is no significant difference between the two encodings. The convergence graphs indicated that both encodings successfully minimized the classification error.

Additionally, the methods and parameters of models with the best error, worst error, and those close to the mean and median were analyzed. It was observed that the methods generated by the algorithm did not vary much between the two encodings, although their parameters did. However, in the case of the best error, there was variation in both encodings. In conclusion, the binary encoding performed better in minimizing error and execution time. The most optimal methods were successfully selected in both encodings, and both encodings provided favorable results in the selection of the bird's head, although visually, the mixed encoding better suited this type of problem.

Capítulo 1

Introducción

1.1. Introducción

Las Ciencias de la Computación poseen diversas áreas que proponen alternativas de solución cuando se trata de procesar, analizar e identificar patrones en imágenes. Algunas de las áreas más socorridas son el procesamiento digital de imágenes, visión artificial y el reconocimiento de patrones.

El procesamiento digital de imágenes mediante algoritmos de filtrado, suavizado, escalado, busca mejorar la imagen, eliminando el mayor ruido posible causado por el sensor al momento de obtenerla. Posteriormente la visión por computadora aplicando técnicas de detección de bordes, texturas, etc., extrae las características o atributos que mejor describen al objeto obteniendo así un patrón de la imagen para un posterior procesamiento; ésta salida se complementa con el reconocimiento de patrones en el que, básicamente se busca "saber" qué contiene la imagen, clasificándolo en categorías o clases y asignando etiquetas para asociarlas a un grupo de pertenencia. Estas áreas cuando se aplican como complemento obtienen sistemas capaces de percibir y entender de forma semi-automática el contenido de la imagen [8].
Estos sistemas semi-autónomos son aplicados en algunos campos de acción como en la industria, para automatizar la inspección de líneas de ensamblaje; en la medicina para diagnosticar enfermedades a partir de datos médicos como tomografías, electrocardiogramas, electroencefalogramas, etc.

El uso de la biometría animal es una base fundamental en la biología, por ejemplo, permite la identificación de cada individuo a partir del estudio de las manchas en la piel de cada especie [18]. Dentro de estos estudios se encuentra el caso de la clasificación de imágenes de aves, que busca extaer las características que más identifican a una especie, obteniendo así aspectos como textura, color o la morfología completa del ave.

En la mayoría de los trabajos donde se aplican técnicas de procesamiento de imágenes el ambiente es controlado, es decir, el fondo es diferente al color del ave o no hay una superposición de objetos (ramas, hojas, etc.), de tal forma que la limpieza o el procesamiento de la imagen es menos compleja y se facilita la aplicación de técnicas de extracción de características.

En esta investigación se realiza el tratamiento de imágenes en ambientes reales, utilizando para ello propiedades de su estructura morfológica como el pico, cabeza y alas. Así como la aplicación de técnicas de procesamiento de imágenes para lograr representar la morfología de la cabeza del ave en una serie de tiempo. Con lo anterior se puede mejorar la metodología que hasta ahora se ha aplicado en este proceso y se construiría una base de datos para evaluarlos con la aplicación de diferentes modelos de clasificación.

Este trabajo podrá proporcionar una base para construir un detector de aves como herramienta de apoyo para personas no expertas que participan en actividades sobre observación de aves, principalmente en programas de ciencia ciudadana de la ciudad de Loma Bonita. De tal manera que se despierte el interés de los ciudadanos en su colaboración para conservar las aves de la localidad.

1.2. Planteamiento del problema

La identificación de aves es una tarea importante en muchas actividades de monitoreo de ciencia ciudadana, donde se solicita a los voluntarios que tomen y compartan fotografías de aves en su entorno [23]. El objetivo de estas actividades es recopilar datos que se puedan utilizar para estudios de conservación de la biodiversidad y para comprender mejor la distribución y el comportamiento de las aves [23].
Sin embargo, la identificación de aves a partir de imágenes puede ser difícil, especialmente para los no expertos en el área. Entre los principales factores que pueden complicar la tarea de identificar aves en imágenes digitales se encuentran: la calidad de la imagen, variaciones y diversidad de las especies, variaciones en la iluminación y el fondo de las imágenes y principalmente la experiencia del observador [37].
En la actualidad no existen o son limitadas las aplicaciones que puedan ser utilizadas para ayudar en la identificación de aves a partir de imágenes digitales. En este sentido, el procesamiento de imágenes digitales es un área de la ciencia de la computación que busca mejorar los aspectos de una imagen a partir de los pixeles que la constituyen, mediante la aplicación de diferentes técnicas las cuales se seleccionan de acuerdo al problema planteado.
A partir de estos factores surge la siguiente pregunta de investigación:

¿Es posible diseñar una metodología semi-automática para mejorar la extracción de la forma de la cabeza, en imágenes digitales de tres especies de aves, a través del uso de algoritmos genéticos con representación real y binaria en una secuencia de procesamiento de imágenes digitales para obtener un valor competitivo de clasificación?

Se plantea diseñar una metodología semi-automática basado en procesamiento de imágenes digitales a partir de un algoritmo metaheurístico que puede ser considerada como parte de la caja de herramientas para los investigadores, conservacionistas y científicos ciudadanos para mejorar la identificación de aves, y no como una solución que reemplace la necesidad de experiencia y conocimiento humano en ornitología.

1.3. Objetivos

1.3.1. Objetivo General

Clasificar con precisión las especies de aves, a partir de su morfología, al diseñar y validar una metodología semi-automática basada en algoritmos metaheurísticos.

1.3.2. Objetivos Específicos

- Analizar los trabajos relacionados con la extracción de características.
- Obtener imágenes digitales a partir de repositorios públicos.
- Evaluar diferentes técnicas de procesamiento de imágenes digitales para la extracción de la forma de la cabeza de las tres especies de aves.
- Implementar algoritmos genéticos con representación real y binaria para optimizar el proceso de extracción de la forma de la cabeza de las tres especies de aves.

1.4. Hipótesis

La siguiente investigación define la hipótesis:
La utilización de algoritmos genéticos con representación real y binaria en una secuencia de técnicas de procesamiento de imágenes digitales puede mejorar la precisión y eficiencia de la extracción de la forma de la cabeza de una especie de ave a partir de imágenes digitales.

1.5. Justificación

Herramienta de soporte: busca ayudar a mejorar la precisión y eficacia en la identificación y clasificación de especies de aves, aprovechar la aplicación como una herramienta tecnológica para mejorar el conocimiento [30] [5]. Esta herramienta no busca sustituir la capacidad humana para la identificación y clasificación de aves, sino que, permite valorar la importancia que tiene la experiencia humana en la observación y el estudio de la fauna silvestre [5].

Innovación tecnológica: Al desarrollar esta técnica y aplicarla en la clasificación de especies de aves, se podría abrir nuevas oportunidades para el desarrollo de sistemas automatizados de monitoreo y seguimiento de la fauna silvestre. Siendo especialmente útil en áreas remotas o de difícil acceso, donde la implementación de tecnologías avanzadas puede tener un gran impacto en la conservación de la biodiversidad [3]. Además, la técnica propuesta es novedosa, llegando a contribuir al desarrollo de algoritmos genéticos más avanzados y eficientes que puedan tener aplicaciones en otros campos de la ciencia y la tecnología.

Impacto en la educación ambiental: La investigación propuesta contribuye en la educación ambiental y conciencia sobre la importancia de la biodiversidad y la conservación de las aves [12] [27] [5]. También su desarrollo daría mayor disponibilidad, accesibilidad y precisión a la información para los educadores, estudiantes y el público en general interesados en aprender sobre la fauna silvestre y su conservación [10].

Participación de la ciencia ciudadana: Al contar con una metodología clara y sencilla para la extracción de la forma de la cabeza de las aves, los investigadores pueden involucrar a los ciudadanos interesados en la observación de aves y la conservación de la biodiversidad en el proceso de clasificación de las imágenes digitales. Esto puede generar un impacto social significativo, ya que permite a la ciudadanía participar activamente en los estudios sobre fauna silvestre [10]. Además, ayuda a aumentar la cantidad de datos disponibles para la investigación [23] [15], lo que propicia una mejor precisión y eficacia de la clasificación de especies de aves. En definitiva, esto contribuiría a fortalecer la colaboración entre la comunidad científica y la sociedad en general en la conservación de la biodiversidad [10].

Capítulo 2

Revisión de la literatura

En el siguiente capítulo se describen algunos trabajos relacionados con el procesamiento de imágenes siguiendo una metodología semi-automática.

La revisión de la literatura se enfocó en consultar propuestas que planteen el uso de metaheurísticas para optimizar la secuencia de las tareas de procesamiento de imágenes, así como los hiperparámetros correspondientes. Esto debido a que existen escasos trabajos donde se proponen una metodología semi-automática para la clasificación de aves que involucren varias tareas en el procesamiento de imágenes.

1. Rojas, Carballido, Olivera y Vidal [36]. Propusieron que para obtener un mayor rendimiento de clasificación y generalización de una máquina de soporte de vectores SVM (Support Vector Machine), era necesario encontrar una configuración de calidad de sus hiperparámetros, en el cual utilizaron un conjunto de datos de Rinopatía Diabética perteneciente a la universidad de Debrecen. Logrando encontrar en los algoritmos metaheurísticos una forma eficiente para lograr la optimización del SVM. De los cuales tomaron cuatro para evaluarlos: (Algoritmo Genético (AG), Evolución diferencial (ED), Algoritmo Genético Celular (AGC) y Recocido simulado (RS)), con la finalidad de que se optimizara cuatro hiperparámetros, que mejoraron la exactitud de predicción de cada instancia al momento de clasificarlos. Dentro de la división del conjunto de datos no tomaron más que una porción para entrenamiento y prueba, esta selección pudo provocar un sobreajuste, y clasificar de manera incorrecta nuevos datos, una solución a ello podría ser el considerar la aplicación de validación cruzada (Cross-validation).

2. Para Morales, Viear, Rodriguez y Serrano [13], los algoritmos de procesamiento de imágenes y visión computacional para la detección de obstáculos presentan múltiples parámetros que necesitan ser ajustados para un funcionamiento eficiente según las condiciones del entorno donde opera el robot. Las imágenes

contienen diferentes obstáculos presentes en un entorno y que fueron generadas por el equipo; algunas fueron segmentadas por las personas para compararlas con las que generó el algoritmo. Se compararon dos metaheurísticas (algoritmo óptimo de Enjambre de Partículas y búsqueda de población mínima) en las cuales se logró optimizar la segmentación de la imagen buscando la configuración de los hiperparámetros de dos tareas (filtrado y segmentación). La propuesta debía aplicar una comparación pixel a pixel a través de imágenes previamente procesadas por el usuario, para poder determinar la calidad del algoritmo, esto pudo ser una desventaja sobre el método, ya que interviene de manera significativa el humano, en este caso pudo ser conveniente aplicar algún clasificador.

3. Carvaloh, Silva, Rabelyo y Cavalho Filho [6], proponen que para poder dar un diagnóstico efectivo sobre el COVID-19 a partir de imágenes de tomografía computarizada obtenidas de repositorios públicos, era necesario generar una metodología donde se lograra extraer las características que mejor tasa de acierto proporcionaran, para ello definieron un algoritmo genético cuya función de evaluación era determinada a través del índice kappa, a partir de las características optimizadas por el AG, lograron obtener una precisión del 97 %.
   En la implementación del método de cruza de un sólo punto, se pudo considerar una cruza uniforme para expandir la diversidad de individuos en las nuevas generaciones.

4. En cambio, Hoang y Nguyen [21] generaron una metodología basada en el procesamiento de imágenes para detectar grietas en las superficies de las paredes de concreto. Las imágenes son de edificios de la ciudad de Da Nang (Vietnam) y recopiladas por inspectores quienes las etiquetan como grietas o no grietas. Dentro de la metodología realizan algunas tareas de limpieza y segmentación que requieren de una buena configuración de sus hiperparámetros, lograron comprobar que, a través de la aplicación de metaheurísticas, concretamente de un algoritmo de polinización diferencial de flores (DFP) se pudo resolver este problema de optimización. Cabe señalar que fue evaluado bajo cuatro ejecuciones sobre el conjunto, una por cada método de bordes para obtener el que mejor representa la grieta. Haber realizado la tarea de esta manera conllevó consumo de tiempo y recurso computacional, a diferencia de, si se hubiera dejado que el algoritmo DFP realizara esta tarea de manera automática. Además la evaluación se realizó bajo una función de costo determinada a partir de los falsos positivos y los falsos negativos obtenidos de la clasificación.

En la Tabla 2.1 se muestra una síntesis de las propuestas descritas anteriormente.

Trabajos Relacionados	Metaheurísticas	Datos	Tareas a optimizar	Total de parámetros
Carballido, Olivera y Vidal [19]	Algoritmo Genético (AG) Evolutivo Diferencial (ED) Algoritmo Genético Celular (AGC) Recocido Simulado (RS)	1151 instancias sobre información de enfermedad, con 19 atributos	Clasificación	4 parámetros de un SVM
Morales, Viear, Rodriguez y Serrano [13]	Optimización de Enjambre de Partículas (PSO) Método de población mínima (MPS)	Imágenes capturadas manualmente	Preprocesamiento y segmentación	10 parámetros; 1 para preprocesamiento y 9 de segmentación
Carvaloh, Silva, Rabelyo y Cavalho Filho [6]	Algoritmo Genético (AG)	2482 imágenes de repositorios públicos y etiquetadas	Extracción de características	Considera las características obtenidas de una red neuronal
Hoang y Nguyen[21]	Polinización Diferencial de Flores (PDF)	1880 imágenes de grietas de edificios tomadas manualmente	preprocesamiento y segmentación	1 parámetro para preprocesamiento y 6 para segmentación

Tabla 2.1: Síntesis de los métodos de clasificación de imágenes de aves mediante una metodología semiautomática

Los estudios anteriores han demostrado de qué manera la aplicación de las metaheurísticas son una opción eficiente cuando se trata de optimizar hiperparámetros o tareas que implican procesamiento de imágenes y clasificación.

Capítulo 3

Marco teórico

En este capítulo se definen las distintas técnicas y métodos que se utilizaron en el desarrollo de este documento de tesis, con la finalidad de una mejor comprensión de ésta.

3.1. Pipeline machine learning

Pipeline machine learning o canalización de datos es un proceso de automatización que actúa como una tubería por donde los datos fluyen desde un origen a un destino. Durante este proceso se pasa por diferentes etapas, las cuales son [33] [1]:

- Origen de datos: Estos pueden proceder de bases de datos, de sistemas CRM, sensores IoT, archivos, etc.
- Procesamiento o transformación: En esta etapa se incluyen las operaciones que procesan los datos en el formato requerido por el repositorio de datos destino. Dentro de las operaciones que se pueden realizar se encuentran transporte, traducción, clasificación, deduplicación, validación y análisis.
- Almacenamiento: Todos los datos se almacenan para que los usuarios puedan acceder a ellos.

En el caso del procesamiento de imágenes, algunas tareas que pueden incluirse en una canalización de datos son:

- Seleccionar o recolectar las imágenes, las cuales pueden ser obtenidas de diferentes repositorios, estos suelen estar abiertos al público para su uso.
- Pre-procesamiento, el cual incluye transformaciones como normalización, estandarización, corrección de color, reescalado. Selección de áreas de interés, mediante recortes, detección de objetos o segmentación de imágenes, entre otras técnicas.

- Extracción de características, que puede ser, representación vectorial, imagen transformada, entre otros.
- Clasificación y predicción, a partir de las características extraídas se aplica un clasificador.
- Validación, para lograr evitar un sobreajuste se implementa algún método de validación por ejemplo cross-validation.
- Implementación, desplegar de acuerdo al fin que se persigue.

3.2. Procesamiento de imágenes

3.2.1. Qué es una imagen

Una imagen es la representación del mundo físico que tiene información importante, misma que es obtenido a través de diferentes medios electrónicos, como pueden ser cámaras fotográficas o de video, sensores, etc [38]. Los cuales involucran un proceso de captura, muestreo, cuantificación y codificación [2].

Una imagen puede definirse como una función bidimensional que cuantifica la intensidad de luz [38] [2] en cada punto de la imagen y que esta asociada a un sistema de coordenadas espaciales x, y. Una imagen se expresa matemáticamente como en la ecuación 3.1:

$$I = f(x, y) \tag{3.1}$$

Donde f representa el nivel de brillantez o intensidad de la imagen en las coordenadas (x, y) [38].

Una imagen digital puede considerarse como una matriz de n x m, cuyos índices de filas y columnas representan un punto en la imagen y el valor correspondiente a ese punto indica el nivel de gris en esa ubicación. A este punto se le denomina pixel.

3.2.2. Qué es el procesamiento digital de imágenes

Cuando se adquiere una imagen mediante diferentes dispositivos de captura o sensores como cámaras fotográficas, cámaras de televisión, sensores de rangos etc., las imágenes pueden contener efectos no deseados.

Al realizar trabajos de reconocimiento o clasificación de imágenes, es necesario que las imágenes sean mejoradas, es decir, eliminar el mayor ruido posible o acentuar algunas partes para poder lograr una buena clasificación.

El procesamiento de imágenes es un área de las ciencias de la computación que proporciona diferentes técnicas, que se enfocan en mejorar la imagen, básicamente

con su aplicación se obtiene una "nueva" imagen que será de mayor utilidad en las siguientes etapas. Las técnicas que se desarrollan para el mejoramiento de la imagen dependerán del tipo de análisis a emplear. Dentro de las más utilizadas se encuentran suavizado, reducción de ruido, detección de bordes, segmentación, etc. esta investigación tratará la *filtración y segmentación* [38].

3.2.3. Filtrado

El objetivo de un filtro es suavizar y eliminar el mayor ruido posible de la imagen así como resaltar o destacar detalles, sin perder información. De ahí que se tengan dos formas de realizar el filtrado: bajo el dominio del espacio y bajo el dominio de la frecuencia. Los filtros bajo el dominio del espacio trabajan directamente sobre los pixeles de la imagen [38], ya que estos están basados en máscaras de convolución o kernel, las cuales son matrices de coeficientes de $n \times m$; operan sobre un pixel central de interés y sus vecinos. El tamaño de las matrices dispondrá la cantidad de vecinos.

3.2.4. Filtro Gaussiano

El filtro Gaussiano actúa como una convolución, definida por:

$$g(x,y) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} k(i,j) \times f(x+i, y+j) \tag{3.2}$$

donde una máscara o kernel $k(i,j)$ hace un barrido sobre la imagen $f(x+i, y+j)$.

Llamaremos *máscara* a una matriz de coeficientes construida a partir de la función gaussiana

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x+y)^2}{2\sigma^2}} \tag{3.3}$$

Donde el valor dado a la desviación estándar σ y el tamaño de la matriz determinan que tanto se conservará el contraste de la imagen. Ya que, un valor de sigma mayor atenua más la imagen eliminando el ruido, pero perdiendo detalles de la imagen. Mientras un valor menor elimina ruido, pero sin perder información.

Por tanto es necesario determinar un valor estándar óptimo para evitar perder detalles de la imagen, que provoquen dificultades en las etapas posteriores.

3.2.5. Filtro u Operador Sobel

El objetivo de este filtro es resaltar detalles de la imagen sobre todo, los bordes. Ya que son una forma de delimitar al objeto pues generalmente un borde presenta un cambio brusco de color, trabajando de pixel en pixel. El operador Sobel está basado

en las derivadas de primer orden o gradiente de una función $f(x, y)$ definido como:

$$\nabla f(x,y) = \begin{bmatrix} \dfrac{\partial f(x,y)}{\partial x} \\ \dfrac{\partial f(x,y)}{\partial y} \end{bmatrix}$$

Calculando la magnitud del gradiente por:

$$|\nabla f| = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2}$$

Este gradiente determina el cambio de intensidad del pixel, y determina si es o no un borde. Para que el operador obtenga el gradiente hace uso de dos máscaras de convolución generalmente impares, una para obtener cambios de intensidad en dirección vertical y otra para la dirección horizontal [38].

Ejemplo 3.2.1. *Consideremos dos matrices de* 3×3*,* G_x *y* G_y*, definidas como:*

$$G_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad y \quad G_y = \begin{pmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

En este ejemplo G_x y G_y representan la dirección vertical y horizontal respectivamente. Este filtro trabaja mediante imágenes en escalas de grises, cada pixel constituye la intensidad o nivel de gris. La imagen que recibe debe de ser previamente filtrada, es necesario que se evite perder detalles de la imagen ya que esto podría provocar que el operador Sobel no los detecte.

3.3. Segmentación

La segmentación es una de las etapas previas al reconocimiento de patrones, ya que su objetivo es aislar el objeto de interés [16], de tal manera se obtienen los rasgos más significativos de la imagen. Una forma de segmentar las imágenes es mediante los contornos o bordes, para lograrlo se hace uso de diferentes características de la imagen un ejemplo de ello es el color [38].

3.3.1. Umbralización

Es una técnica simple de segmentación que transforma una imagen de entrada en escala de grises (donde sus pixeles representan los niveles de gris) a una imagen de salida binarizada, agrupando los pixeles a los diversos objetos.

- Umbral simple

- Umbral adaptativo

El ***umbral simple*** es el método más básico [38]; ya que a partir de un umbral con valor T (threshold) de intensidad se determina el valor del pixel de la imagen de salida. Obteniendolo a partir de:

$$S[x, y] = 1, \quad E[x, y] > T,$$

$$S[x, y] = 0, \quad E[x, y] \leq T$$

Donde $E[x, y]$ es la imagen de entrada, $S[x, y]$ la imagen de salida y T es el valor de umbral. Si el pixel en la posición (x, y) de la imagen de entrada es menor al valor de T entonces la imagen de salida en la misma posición obtiene el valor de cero. Y de uno en caso de que el nivel de gris del pixel sea mayor a T.

El método descrito anteriormente da buenos resultados para imágenes donde hay poca variación en la intensidad de fondo. Sin embargo, no daría los mismos resultados ante imágenes donde el fondo presenta una gran variedad de intensidad. Ante esta problemática se desarrolló un método adaptativo que se conoce como ***umbral adaptativo***. Este método tiene como base la umbralización a través de un valor T pero no se hace de manera global para toda la imagen, si no que obtiene umbrales locales a partir de los pixeles vecinos, aunque para ello se debe de contemplar la manera en que se realizará la búsqueda de esos valores ya que puede adoptar un método gaussiano o un método basado en la mediana de los vecinos del pixel.

3.4. Series de tiempo

Una serie de tiempo se puede decir que son colecciones de observaciones sobre un determinado fenómeno, realizadas en sucesivos momentos del tiempo, y generalmente equiespaciados. Es decir que corresponde a la realización de un proceso generador de datos.

Definición 3.4.1. Una serie temporal t (o simplemente una serie) es una secuencia de n observaciones (datos) ordenadas y equidistantes cronológicamente sobre una característica (serie univariante o escalar) o sobre varias características (serie multivariante o vectorial) de una unidad observable en diferentes momentos.

Comúnmente definiremos a la serie como $t = t_1, t_2, \ldots, t_n$ [32]. Donde cada t_n representa un elemento que constituye a la serie t.

3.5. Transformaciones de series temporales

Las series de tiempo suelen tener una alta dimensionalidad, por lo que hace que su tratamiento sea más costoso en términos de almacenamiento y procesamiento. Es por ello que, para que sean más manejables se han desarrollado diferentes técnicas de reducción de la dimensionalidad. Estas técnicas mantienen las características de los datos pero con una dimensión menor a la original [11].
Entre las técnicas para la representación de series mostradas se encuentran:

- Análisis de componentes principales (PCA),
- Transformación discreta por Wavelets (DWT),
- Aproximación agregada por partes (PAA),
- Aproximación agregada simbólica (SAX).

3.5.1. Aproximación Agregada por Partes (PAA)

Sea $X = x_1, x_2, \ldots, x_n$ una serie de tiempo y el conjunto de series temporales que constituyen la base de datos como $Y = Y_1, Y_2, \ldots, Y_n$. Sin pérdida de generalidad, supongamos que cada sucesión de Y tiene una longuitud de n unidades. Sea N la dimensión del espacio transformado que queremos indexar ($1 \leq N \leq n$). Por conveniencia asumimos que N es un factor de n. Una Serie de tiemo X de tamaño n es representada en el espacio N por el vector $\bar{X} = \bar{x_1}, \bar{x_2}, \ldots, \bar{x_N}$, el $i-$ésimo elemento de $\bar{X}$ es calculado por la ecuación 3.4

$$x_i = \frac{N}{n} \sum_{j=\frac{n}{N(i-1)+1}}^{\left(\frac{n}{N}\right)i} x_j \tag{3.4}$$

Vale la pena señalar dos casos especiales. Cuando $N = n$ la representación transformada es idéntica a la representación original. Cuando $N = 1$, la representación transformada es simplemente la media de la secuencia original. Más generalmente, la transformación produce una aproximación constante por partes de la secuencia original. Por lo tanto, llamamos a este enfoque *aproximación agregada por partes (PAA)* [24].

3.5.2. Aproximación Agregada Simbólica (SAX)

El método *Aproximación Agregada Simbólica (SAX)*, es un método de reducción de dimensionalidad de una serie temporal de longitud n a una cadena de longitud w

con $w < n$ donde w es el número de segmentos que representa la serie transformada por PAA. *El método SAX* define un alfabeto que es un número entero arbitrario a donde $a > 2$ [26].
Por lo tanto, este método consta de dos fases: la primera es la transformación de los datos en una representación de aproximación agregada por partes (PAA). La segunda es simbolizar la representación PAA mediante cadenas.

A continuación, se describirá la representación simbólica propuesta en [26], ya que la reducción mediante la técnica de PAA, fue descrita en la sección 3.5.2.
Sea $\bar{X} = \bar{x_1}, x_2, \ldots, \bar{x_n}$ obtenida por PAA; a esta reducción se aplica una transformación más para obtener una representación discreta normalizada. Así tenemos símbolos que cuenten con la misma probabilidad, esto es posible ya que las series temporales normalizadas tienen distribución gaussiana. Dado que las series de tiempo normalizadas tienen una distribución altamente gaussiana, podemos simplemente determinar los " puntos de ruptura " que producirán áreas de igual tamaño bajo la curva gaussiana [25].

Definición 3.5.1 (Puntos de ruptura)**.** Los puntos de ruptura son una lista ordenada de números $B = \beta_1, \beta_2, \ldots, \beta_{\alpha-1}$ tal que el área bajo $N(0,1)$ la curva gaussiana desde β_i hasta $\beta_{i+1} = \dfrac{1}{a}$ (β_0 y β_α son definidas como $-\infty$ y ∞ respectivamante).

Después de obtener los puntos de ruptura se asignan los símbolos. Todos los coeficientes PAA debajo del punto de ruptura más pequeño se asigna el símbolo a; todos los coeficientes mayores o iguales al punto de ruptura más pequeños y menores al segundo punto se asignan al símbolo b y así sucesivamente. En este ejemplo los tres símbolos, "a", "b" y "c" son aproximadamente equiprobables. Llamamos *palabra* a la concatenación de símbolos que representan una subsecuencia.

Definición 3.5.2 (Palabra)**.** Una subsucesión C de tamaño n se puede representar como una *palabra* de la siguiente manera $\hat{C} = \hat{c_1}, \hat{c_2}, \ldots, \hat{c_n}$. α_i representa el $i-$ésimo elemento del alfabeto, es decir, $\alpha_1 = a$ y $\alpha_2 = b$. Entonces, un mapeo de una aproximacón PAA $\bar{X}$ a una palabra $\hat{C}$ es obtenida como sigue

$$\hat{c_i} = \alpha_j \text{ si y solo si } \beta_{j-1} \leq \hat{c_i} < \beta_j$$

3.6. Clasificación

El objetivo principal de la clasificación es asignar objetos en un número específico de categorías o clases. Dependiendo de la aplicación estos objetos pueden ser imágenes, sonidos, colores, etc [35]. Se tiene diferentes tipos de clasificación, supervisada y no supervisada. La clasificación supervisada, es cuando se tiene un conocimiento previo

de los datos, y la no supervisada es aquella donde no se conoce la información, y se generan "cluster" basado en la similitud de la información.

3.6.1. Algoritmo KNN

Es un método no paramétrico de clasificación supervisada. Estima la clase de un individuo de prueba en función de los $k-$vecinos más cercanos, cada elemento se determina mediante una métrica de cálculo de distancia entre cada objeto [39], se toman los k elementos más cercanos y se determina cual es la clase que más se repite, asignándola al nuevo elemento o instancia.
Dado un conjunto de muestras de entrenamiento con tuplas $D = \{(x_1, c_1), (x_2, c_2), \ldots, (x_N, c_N)\}$ donde x_N es el conjunto de características y c_N la clase. Sea $y = \{y_1, y_2, \ldots, y_i, c_i\}$ el nuevo patrón desconocido que se asignará a la clase c_m tal que sea la muestra del conjunto de entrenamiento más cercana a x de acuerdo con alguna métrica. En general se considera los k vecinos más cercano de x dentro del conjunto de entrenamiento y se considera c_m como la clase que tiene más muestras entre los k vecinos considerados [31].

Algoritmo 1 Algoritmo KNN

Entrada: Entrada: $D = (x_1, c_1), ...(x_1, x_N), k$

1: $y = (y_1, ..., y_n)$ nuevo caso a clasificar perteneciente a D
2: **for** (x_i, c_i) perteneciente a D **do**
3: Calcular la distancia $d_i = d(x_i, y)$
4: Ordenar $d_i (i = 1, ..., N)$ en manera descendente
5: Elegir los k casos de d_i donde sean menores o más cercanos a y
6: Contar la frecuencia de aparición de la c_i asociada a cada d_i
7: Asignar a y la clase más frecuente.
8: **end for**

Como anteriormente se describió, para realizar el cálculo de los k vecinos más cercanos se determina a través de una métrica de distancia, en este caso se consideraron las métricas de distancia Euclidiana y deformación dinámica del tiempo (DTW).

Distancia temporal dinámica o por sus siglas en inglés Dynamic Time Warping (DTW), originalmente esta técnica se había empleado para el reconocimiento de patrones de voz automático [14]. Sin embargo, tiene otras aplicaciones como es el caso del algoritmo de KNN, ya que se utiliza como métrica para medir la distancia entre la instancia de prueba y y la instancia de entrenamiento x. El algoritmo permite llevar a cabo una alineación óptima entre dos series de tiempo de diferentes longitudes haciendo uso de programación dinámica para calcular el coste. Este alineamiento se obtiene a partir de una medida de distancia entre dos patrones temporales, como se puede observar en la Figura 3.1.

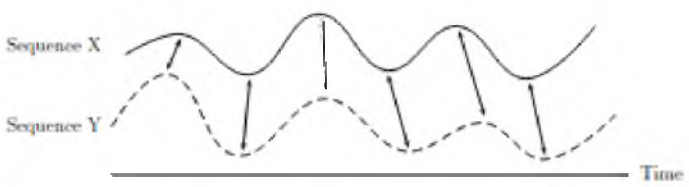


Figura. 3.1: Alineación temporal de dos secuencias dependientes del tiempo. Los puntos alineados se indican con flechas [14]

Definición 3.6.1 (Distancia Euclidiana). Sean $x, y \in \mathbf{R}^2$. la distancia entre x y y se denota como $d(x, y)$ y se calcula como

$$(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2} \tag{3.5}$$

donde x_1, x_2 son coordenadas de x y y_1, y_2 coordenadas de y, es decir $x = (x_1, x_2)$, $y = (y_1, y_2)$

Ejemplo 3.6.2. *Sean* $X = \{2, 3, 4, 4, 5, 5, 6, 7, 5, 5\}$ *y* $Y = \{2, 2, 2, 2, 2, 2, 2, 2, 3, 3\}$ *dos series de tiempo y* $P = \{1, 1, 1, 1, 1, 2, 2, 3, 4, 4\}$.
Tomamos a X *y a* Y *como las series pruebas y a* P *como el patrón. El objetivo es encontrar que serie se parece más al patrón* P.

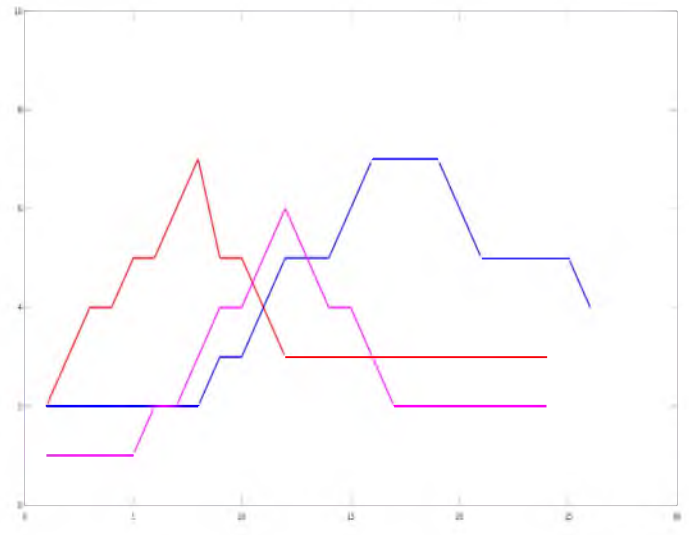


Figura. 3.2: Representación de las series de tiempo X, Y, P. Donde $X-$línea roja, $Y-$línea azul y $P-$línea rosa

A simple vista se puede decir que la más parecida es la línea azul, es decir, X. *Pero para una computadora o robot no es tan fácil de determinar.*

En [20] podemos ver un ejemplo más extenso de este método. En este ejemplo darémos sólo un resumen de los pasos y cálculos necesarios que fueron aplicados.

1	1	1	1	1	0	0	1	2	2
2	2	2	2	2	1	1	0	1	1
3	3	3	3	3	2	2	1	0	0
3	3	3	3	3	2	2	1	0	0
4	4	4	4	4	3	3	2	1	1
4	4	4	4	4	3	3	2	1	1
5	5	5	5	5	4	4	3	2	2
6	6	6	6	6	5	5	4	3	3
4	4	4	4	4	3	3	2	1	1
4	4	4	4	4	3	3	2	1	1

Figura. 3.3: X es el lado vertical y P el lado horizontal

- Matriz de distancia.
  Aplicamos 3.5 a X y a P.

- Matriz de costos y camino de menor costo.
  Estos cálculos estan relacionados a un problema de *Teoría de Grafos*. Un método para obtener la menor distancia de un punto a otro, es el de *Dijkstra* el cual consiste en ir encontrando distancias acumuladas mínimas en cada nodo. Este método permite explicar el concepto de la matriz de costo y el "warping path". Se puede decir que la matriz de costo es una matriz de distancias acumuladas en el que cada elemento está definido por la siguiente expresión

$$c_{i,j} = d_{i,j} + \text{mín}(c_{i-1}, c_{i-1,j}, c_{i,j-1}) \tag{3.6}$$

Después de aplicar $c_{i,j}$ la matriz de costo es de la forma Figura 3.4. Para hallar

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
A1	[01]	[02]	[03]	[04]	[05]	[05]	[05]	06	08	10
A2	03	03	04	05	06	06	06	[05]	06	07
A3	05	05	05	06	07	07	07	[05]	06	07
A4	08	08	08	08	09	09	09	06	[05]	[05]
A5	12	12	12	12	12	12	12	08	06	06
A6	16	16	16	16	16	15	15	10	07	07
A7	21	21	21	21	21	19	19	13	09	09
A8	27	27	27	27	27	24	24	17	12	12
A9	32	32	32	32	32	28	28	20	14	14
A10	36	36	36	36	36	31	31	22	15	15

Figura. 3.4: Matriz de costo después de aplicar $c_{i,j}$[20]

los elementos que conforman el "warping path" primero se toma el elemento P_n, A_m y los demás elementos se encuentran aplicando la siguiente fórmula en la matriz de los costos de manera sucesiva: $c_{i,j} = d_{i,j} + \text{mín}(c_{i-1}, c_{i-1,j}, c_{i,j-1})$

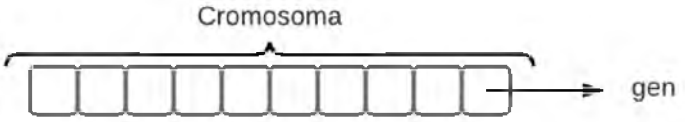


Figura. 3.5: Composición de un individuo

- Distancia DTW
  Como resultado se tiene que la distancia DTW entre X y P es 0,45$[cm]$ la cual se obtiene al aplicar el "warping path" en la matriz de distancias y sacar el promedio entre todos sus elementos [20].

3.7. Algoritmo Genético

Los algoritmos genéticos son parte de los algoritmos evolutivos; estos son métodos robustos de búsqueda, que permiten tratar problemas de optimización donde el objetivo es encontrar un conjunto de parámetros que minimizan o maximizan una función de adaptación (fitness) [20].

Están basados en la teoría biológica de la evolución, donde en una población inicial los individuos evolucionan hacia una mejor población.
Estos algoritmos operan con una población de individuos $P(t) = x_1^t, \ldots, x_n^t$, para la iteración t, donde cada individuo x_i representa un punto de búsqueda en el espacio de las soluciones potenciales a un problema dado [40]. Cada individuo de la población es una posible solución al problema, el conjunto de parámetros de cada individuo (genes) se codifica en una cadena de valores denominada cromosomas, como se muestra en la Figura 3.5. El conjunto de los parámetros de un cromosoma particular se llama fenotipo; y el fenotipo es la información necesaria para la construcción del organismo, es decir, la solución real al problema. Cada individuo tiene una medida de evaluación (fitness) que determina que tan bueno es con respecto a los demás individuos. Hay diferentes formas de representar o codificar los parámetros de un vector solución, estas pueden ser directamente en valores enteros, reales, punto flotante o binario, esta última es la representación que comúnmente se utiliza y es denominada también codificación binaria. La codificación binaria consiste en, dado un posible vector solución $X_k = x_{k1}, \ldots, x_{kn}$ cada uno de los componentes x_{kn} se codifica en una cadena en binario de b bits, la cantidad de bits dependerá del grado de ajuste, finalmente cada cadena se concatena en un solo vector, obteniendo así $l = b$ bits.
Para la decodificación, el vector solución (fenotipo), convierte la cadena de bits según

la cadena de bits que representa cada parámetro al entero correspondiente entre $0, 2^b - 1$, para finalmente reescalarlo al dominio correspondiente.

- Población inicial: Se realiza de manera aleatoria de M individuos y cada uno de longitud l. A partir de esta población se realizan los mecanismos de selección, cruza (crossover) y mutación.

- Selección de individuos: Consiste en determinar que individuos (padres) serán los encargados de reproducirse y generar un nuevo individuo, la selección se obtiene mediante el valor fitness o valor de ajuste. Puede haber algoritmos determinísticos y probabilísticos.

- Selección por ruleta: Determina la suma S de las adaptaciones de toda la población, genera un número aleatorio entre $[0, S]$ y selecciona el individuo cuyo segmento cubre el número aleatorio, el proceso se repite constantemente hasta obtener el número de muestras deseada.

El proceso del algoritmo de programación genética puede ser implementado de la siguiente manera:

1. Inicializar la población.
2. Determinar la aptitud para cada individuo.
3. Realizar la reproducción de acuerdo a los valores de aptitud y la probabilidad de reproducción.
4. Realizar el cruzamiento de los individuos seleccionados como padres.
5. Regresar al paso dos si la condición de terminación no se ha cumplido.

3.8. Ingeniería de software dentro de la IA

La aplicación de la inteligencia artificial (IA) se ha ido incrementando constantemente en los últimos años para resolver problemas en diferentes áreas, de tal manera que es más común ver desde traductores de texto o voz hasta autos autónomos basados en IA [29]. Estos sistemas, como se describe en [29], es software con funcionalidades asistidas por al menos un componente de IA y su aplicación puede estar orientada a software actuando en un mundo virtual o integrados en dispositivos de hardware [29].
Debido a la constante aplicación de los sistemas basados en IA en diversos sectores, y con la finalidad de poder construir, mantener y operarlos se ha propuesto adoptar la ingeniería de software para su desarrollo. La IEEE publicó en el documento Software

Áreas de la ingeniería de software
Requerimiento de software
Diseño de software
Construcción de software
Pruebas de software
Gestión de la configuración de software
Gestión de la ingeniería de software
Proceso de ingeniería de software
Modelos y métodos de ingeniería de software
Calidad del software
Práctica profesional de la ingeniería de software
Economía de la ingeniería de software
Fundamentos de la informática
Fundamentos matemáticos
Fundamentos de la ingeniería

Tabla 3.1: 15 tareas tratadas en el desarrollo de software por la IEEE en SWEBOK [22]

Engineering — Guide to the software engineering body of knowledge (SWEBOK) quince áreas base para el proceso de desarrollo de un producto de software [22], las cuales se enlistan en la Tabla 3.1.

Gorken Giray menciona que el desarrollo de los sistemas basados en IA difieren de la forma en que tradicionalmente se llevan a cabo [17], de la misma forma otros autores mencionan que debido al enfoque de este tipo de software, no se cubren todas las áreas mencionadas en SWEBOK [17],[29], [28]; además de que el análisis de la ingeniería de software dentro de los sistemas basados en IA aún estan en desarrollo. Es por ello que sólo se definirán las que más se adaptan a este tipo de sistemas, según coinciden en la literatura:

- Requerimientos de software:
  Esta fase implica la obtención, el análisis, la especificación y la validación de los requisitos que representan la finalidad prevista de un sistema de software[22]. En estos sistemas los requisitos dependen en gran medida de la disponibilidad del conjunto de datos, es decir, consideran tanto los requisitos funcionales como no funcionales, centrándose más en el análisis de este último. Estos sistemas resultan más complejos al momento de cumplir los cambios de los clientes, debido a las métricas que se utilizan para los datos, las librerías, evaluarlos, etc. [29].

- Diseño de software:
  Es la fase para definir la arquitectura, los componentes, las interfaces y otras características, a partir de los requisitos del software [22]. Esta fase se lleva

a cabo a partir de la definición de diferentes modelos, seleccionando el más adaptable a los requisitos, aunque es un área que está siendo recientemente estudiada, ya se han propuestos modelos para su diseño.

- Construcción de software:
  Una vez definidas las etapas anteriores, se continua con el desarrollo de software a través de una combinación de codificación, verificación, pruebas unitarias, de integración y depuración, la cual se encuentra relacionada con el área de diseño y pruebas; en esta última no se trata de llevar todo el proceso sino sólo algunas actividades [22]. La dificultad, en esta fase para los sistemas de software basados en IA, consiste en su dependencia a los datos, librerías e incluso el lenguaje.

- Pruebas de software:
  Aunque en la fase de desarrollo se llevaron a cabo algunas pruebas, esta área busca verificar dinámicamente que un programa proporcione los comportamientos esperados en un conjunto finito de casos de pruebas [22]. Estas tareas se hacen más complicadas en sistemas basados en IA ya que, se correría el riesgo de un sobreajuste de datos debido a que su desarrollo se realiza a través del entrenamiento de los datos, además del tiempo que se lleva en el entrenamiento. En la literatura este es el mayor problema en este tipo de sistemas.

- Mantenimiento de software:
  En esta fase al software, como producto finalizado y posterior a su implementación, se da un tiempo de garantía o comúnmente llamado prueba de soporte. Durante este tiempo en el que el software se encuentra en funcionamiento se descubren defectos, cambios de entorno y/o surgen nuevas necesidades por parte de los usuarios. Cabe mencionar que esta área está relacionada con las demás. El mantenimiento en sistemas basados en IA, tiene diferentes problemas, se mencionan tres según Giray [17];

  - Gestión de los datos y los modelos de ML: dado al cambio constante de los datos y modelos.
  - Un seguimiento de los experimentos: debido a los elementos que participan como hardware, código, sistema operativo, datos de entrenamiento, etc.
  - Rendimiento y despliegue: a causa del cambio externo de los datos, mantener el rendimiento en medida de la precisión es complicado.

- Proceso de ingeniería de software:
  Son los procesos que los ingenieros de software realizan para desarrollar, mantener y operar software tales como requisitos, diseño, construcción, pruebas,

gestión de configuración. Este desarrollo puede llevarse a cabo de manera lineal, secuencial, con retroalimentación e iteración. El proceso que un sistema basado en IA realiza es distinto al del software tradicional, algunos autores los definen en tres o cinco etapas. Es importante lograr la unificación entre estas y las propuestas por el estándar de la IEEE.

- Modelos y métodos de ingeniería de software:
  Proponen diferentes métodos y herramientas, por ejemplo métodos heurísticos, formales, prototipado y ágiles, con el objetivo de llevar a cabo el desarrollo de software de manera eficiente, orientándolo al éxito, esta área no implica abordar todas las fases de desarrollo de software, aunque tampoco se descartaría.
  Obtener un modelo ajustado para lograr estabilidad ya que tanto los datos, cómo los modelos usados para desarrollar software basado en IA, cambian constantemente.

- Calidad de software:
  Aunque en la fase de desarrollo ya se implementaron algunas prácticas de esta etapa, no se da por culminada, a causa de que se deben de realizar pruebas más estructuradas, de tal manera que un sistema, componente o proceso cumpla con los requisitos especificados y las necesidades del cliente o usuario. Este es uno de los problemas con mayor recurrencia en los sistemas basados en IA sobre todo porque se llevan a cabo a través de entrenamiento y su medida es la precisión dentro de la predicción.

Capítulo 4

Propuesta de solución

En este capítulo de describe la metodología utilizada, tal como se muestra en el Figura 4.1, para llevar a cabo el proceso en este trabajo de investigación. A continuación se describe con detalle las etapas que la conforman.

4.1. Adquisición de imágenes

Para construir el conjunto de datos, se trabajó con tres especies diferentes de aves, Luis Bienteveo (Pitangus Sulphuratus), Carpintero (Dryocopus Lineatus), Zopilote Común (Coragyps Atratus), ya que estas aves son las que predominan o son más populares en la región.

Para la generación del conjunto de imágenes se buscó extraerlas de repositorios

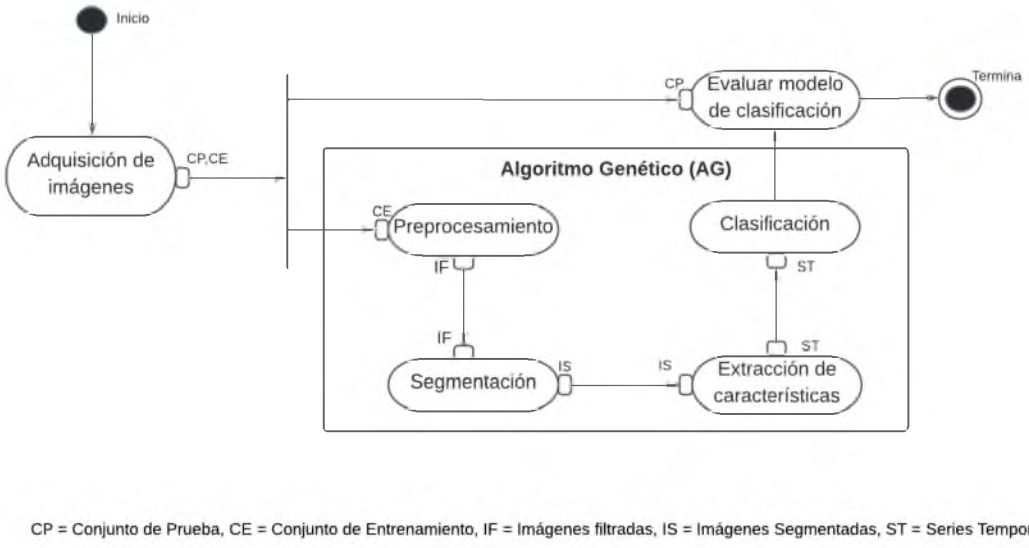


Figura. 4.1: Metodología seguida

públicos donde se encuentran disponibles para su descarga, las páginas que se consultaron para esta tarea fue la del CONABIO (Comisión Nacional para el Conocimiento y Uso de la Biodiversidad), página del gobierno mexicano cuya finalidad es reunir, generar y sintetizar información sobre la biodiversidad mexicana. El banco de imágenes de las tres especies que proporcionan es pequeño con una total de 95 imágenes; de las cuales sólo se descargaron 44, con una dimensión de 800px X 600px y 900px X 700px. Se consideró esta cantidad debido a que las aves se deben de encontrar lo más centrada posible. Cabe mencionar que cada imagen contiene una marca de agua de CONABIO, para proteger los derechos del autor.

Ante la poca diversidad de imágenes que ofrecia el CONABIO, se buscó en la pagina web flickr, cuya finalidad es permitir a los usuarios compartir imágenes con amigos, familia, etc., de aquí se obtuvo la mayor parte, un total de 184 imágenes y bajo las dimensiones de 800px x 600px, nuevamente buscando que el ave estuviera lo más centrada, ya que estas se recortarían, antes de enviarlas a la etapa de procesamiento.

En la Figura 4.2 se muestra un ejemplo de las imágenes de CONABIO y en la Figura 4.3 las de flickr.

((a)) Carpintero Lineado (Dryocopus Lincatus) ((b)) Luis Bienteveo (Pitangus Sulphuratus) ((c)) Zopilote Común (Coragyps Atratus)

Figura. 4.2: Ejemplo de imágenes obtenidas de CONABIO

((a)) Carpintero Lineado (Dryocopus Lincatus) ((b)) Luis Bienteveo (Pitangus Sulphuratus) ((c)) Zopilote Común (Coragyps Atratus)

Figura. 4.3: Ejemplo de imágenes obtenidas de Flickr

El conjunto de datos de imágenes quedó constituido con un total de 228, de las cuales se generarán dos conjuntos; uno de entrenamiento con 117 imágenes (que se usará para entrenar el modelo) y otro para la etapa de prueba y se destinaron 111 imágenes para evaluar el modelo creado.

4.2. Algoritmo Genético para optimizar hiperparámetros y tareas del modelo de clasificación

Dado que los algoritmos genéticos se basan en la teoría de la evolución, donde el individuo de una generación es el más adecuado si ha sido, de toda la población, el que mejor se adaptó a los cambios. Para ello se explicará de que manera puede ayudar en problemas de procesamiento de imágenes.

4.2.1. Representación del cromosoma

Un cromosoma representa *al vector solución*, el cual contiene los valores que darán solución a un problema. Para este caso la cadena representa dos métodos para preprocesamiento; dos para la segmentación, tres para la extracción de características y dos para la clasificación. De esta manera cada método ocupa una porción del vector solución, así para el preprocesamiento va desde $x_i, x_{i+1}, \ldots x_p$, la segmentación $x_j, x_{j+1}, \ldots x_s$, la extracción de características $x_k, x_{k+1}, \ldots x_e$ y la clasificación x_{c_1}, x_{c_2}. En la Figura 4.4 se muestra un ejemplo de como se representa.

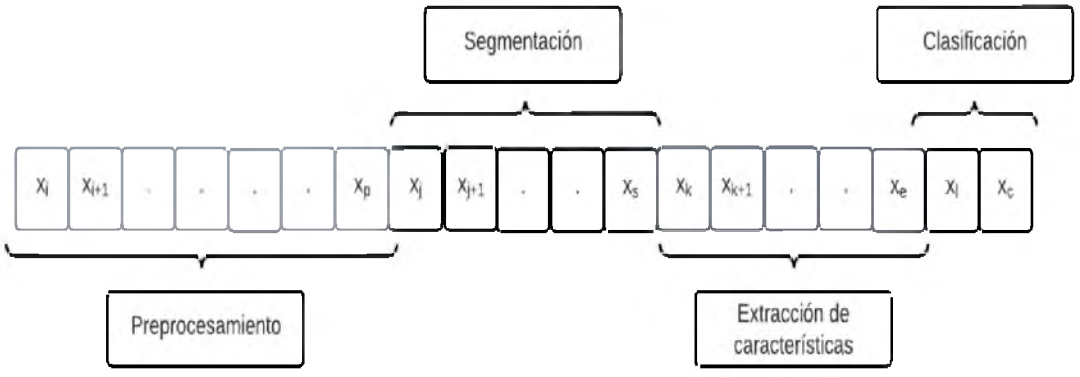


Figura. 4.4: Cromosoma (vector solución)

En la Tabla 4.1 se muestran las tareas.

Para la codificación del individuo se contemplaron dos representaciones una representación binaria y una real.

Tarea	Método	Parámetros	Rangos
Filtrado	Gaussiano (imagen binaria previa)	Kernel	[n x m] impar [3 - 7]
		Desviación estándar en x	[0 - 10]
		Desviación estándar en y	[0 - 10]
	Sobel (filtrado previo)	Size Kernel	[1, 3, 5, 7]
		Bordes X	[1, 0]
		Bordes Y	[1, 0]
Segmentación	Umbral adptative (filtrado previo)	Tipo umbral adaptativo	[1,2]
		Método de cálculo de umbral	[1,2]
		Tamaño de área de vecindad	[3,21]
	Threshold local (filtrado previo)	Umbral mínimo	[0 - 255]
Extracción del contorno	FindContourns (previo detección de bordes)	Modo de recuperación del contorno	[1 - 2]
Transformación	PAA	Tamaño de la ventana	[30 % - 70 %]
	SAX	Tamaño de la ventana	[30 % - 70 %]
		Número de bins	[2 - 26]
Clasificación	Knn distancia Euclidiana	K vecinos	[30 % - 80 %]
	Knn distancia DTW	K vecinos	[30 % - 80 %]

Tabla 4.1: Tareas y parámetros seleccionados

4.2.2. Codificación Binaria

Para *codificar* el vector solución bajo una representación binaria se debe de determinar cual es la cantidad de bits que representará cada valor del vector; para ello se aplica la fórmula (4.1).

$$len(x_i) = \frac{\log(up(x_i) - low(x_i) * (10^P))}{\log(2) + c} \tag{4.1}$$

donde *up* es valor máximo, *low* el valor mínimo del rango definido para cada hiper-parámetro; y P es el valor de precisión.
Para tener el tamaño del vector solución se realiza a través de la suma de cada tamaño de cadena. Y se rellena de manera aleatoria con valores de 0 ó 1. En la Figura 4.6 podemos ver un ejemplo de codificación.

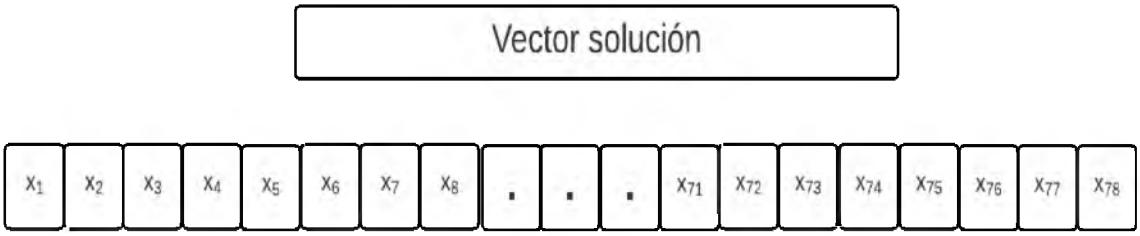


Figura. 4.5: Codificación binaria

Sobre esta representación se aplicarán los diferentes operadores de *cruza, mutación, reemplazo generacional.* Sin embargo, para evaluar la calidad de la solución se debe de *decodificar* a través de la ecuación 4.3. Desde el aspecto biológico a esta decodificación del genotipo se le conoce como *fenotipo.*

$$dec = \frac{low + [x_{int} \times (up - low)]}{2^{len} - 1} \tag{4.2}$$

donde X_{int} es la conversión de binario a real y *len* es el tamaño de la cadena en bits obtenida en la codificación por cada parámetro.

Una de las dificultades dentro de este tipo de representación, es que hay una serie de restricciones de los parámetros mostrados en la Tabla 4.1 por las librerías, ya que especifican que algunos deben de ser impares como por ejemplo los métodos que aplican kernel; provocando que los vectores se salgan del espacio de soluciones factibles; para ello es necesario realizar una reparación del vector, en la literatura el más utilizado es el método de proyección [7], el cual si un valor x_i, es un componente no factible se reasigna a los límites definidos ya sea inferior o superior.

4.2.3. Codificación Real

A diferencia de la representación binaria esta no requiere más que la codificación la cual se lleva acabo de manera aleatoria, dado los rangos definidos en la Tabla 4.1

se construye el vector solución, en la Figura 4.6 se muestra una representación a manera de ejemplo de codificación real.

Figura. 4.6: Codificación

Este método a menudo está más cerca del espacio de soluciones; sin embargo, en este trabajo se decidió aplicar estas dos representaciones. La convencional que es la representación binaria y la real, bajo el objetivo de determinar cual representación es la mejor para aplicar en casos como el de este estudio.

4.3. Función de evaluación o fitness

Para determinar cual es el mejor individuo de una población, es necesario definir una función que evalue la calidad de cada individuo que llamamos solución.
Como el objetivo en este problema es que, dado un conjunto de datos de imágenes se le aplique un conjunto de métodos e hiperparámetros del área de procesamiento de imágenes y reconocimiento de patrones se pueda obtener la mejor configuración que minimice el error de clasificación; sin embargo, para poder evitar un sobreajuste de los datos en el entrenamiento se aplicó validación cruzada. El cual se calcula a partir del promedio de cada error de clasificación obtenida con la expresión 4.3.

$$ER = \frac{1}{k}\sum_{i-1}^{k}\frac{a}{b}, \tag{4.3}$$

donde k es el valor de los k-folds de validación cruzada, a es la porción de instancias clasificadas erróneamente y b es el total de instancias dentro de la base de datos.

4.4. Selección de operadores

4.4.1. Operador de Selección

Para generar una mejor solución, se implementará el método de la ruleta, ya que este buscará al mejor valor de aptitud de un individuo según la subdivisión hecha como si se simulara una ruleta. Este método se consideró el más apto para su aplicación, ya que, una de las ventajas es que suele elegirse el mejor fitness. Cabe mencionar que este operador fue aplicado para ambas codificaciones.

4.4.2. Operador de Cruza

Para los algoritmos genéticos este es el aspecto fundamental [9], ya que de esta combinación de información entre dos individuos se espera obtener el de mejor fitness o valor de adaptación, en este caso se utilizó la cruza uniforme; para ambas representaciones. Cabe mencionar que dentro de este operador es necesario especificar el porcentaje de cruza *pc*. Para la selección se realizaron varios experimentos con distintos valores, esto se explicará en la siguiente sección de experimentos y resultados. El funcionamiento de este operador consiste en recorrer el vector y generar un número aleatorio por cada uno de sus elementos, si el número es menor al *pc* se intercambian los valores en esa posición entre los vectores padres.

4.4.3. Operador de Mutación

Un operador de mutación contribuye a proporcionar una variación en la información de los individuos de tal manera que se expanda el espacio de soluciones. En este caso su aplicación no afecta en gran medida el valor de aptitud ya que su valor de ajuste es mínimo. Por tanto, se propone aplicar un operador de mutación uniforme para ambas representaciones.
Para que en un individuo se suscite la mutación uniforme se generará un número x de manera aleatoria, este debe de cumplir que sea menor a *pm* (porcentaje de mutación).

4.4.4. Recombinación o cambio generacional

Una vez obtenidos los nuevos individuos de la población, es necesario determinar quién o quienes pasarán a la siguiente generación; para esto puede aplicarse, un modelo generacional o un modelo elitista. Este último tiene el inconveniente de llegar a una convergencia más rápidamente; es por ello que se planteó aplicar un modelo generacional, ya que por cada generación crea una nueva población (de descendientes) del mismo tamaño que la inicial; ambas son mezcladas y ordenadas de acuerdo a su valor de aptitud (fitness), de manera que los primeros n individuos, donde n es el tamaño de la población inicial, pasarán a la siguiente generación. En este modelo los individuos que pasan a formar parte de la siguiente población pueden ser tanto los descendientes como los padres.

Algoritmo 2 Algoritmo genético

Entrada: pc, pm , npop, ngen, k-folds

1: Cargar conjunto de datos de entrenamiento
2: Generar índices de *train* y *test* con los $k - folds$
3: Generar población inicial *pob*
4: Evaluar en la función $FITT(\texttt{setImagenes}, \texttt{vecSol})$ cada individuo
5: Seleccionar el mejor individuo según su fitt
6: $i \leftarrow 0$
7: **for** $i \neq ngen$ **do**
8: **for** $x \neq$ (npop/2) **do**
9: Seleccionar p_1 y p_2 de *pob* con selección por ruleta()
10: Generar un valor r aleatorio entre [0,1]
11: **if** $r \leq$pc **then** ▷ Existe o no cruzamiento
12: realizar cruzamiento uniforme
13: **else**
14: p_1 y p_2 no se cruzan
15: **end if**
16: Generar r_2 aleatorio entre [0,1]
17: **if** $r_2 \leq pm$ **then** ▷ Existe o no mutación
18: Realizar mutación uniforme
19: **end if**
20: Evaluar p_1 en la función $FITT(\texttt{setImagenes}, \texttt{vecSol})$
21: Evaluar p_2 en la función $FITT(\texttt{setImagenes}, \texttt{vecSol})$
22: **if** $p_1.fit \leq bestSol.fit$ **then**
23: $bestSol = p_1$
24: **end if**
25: **if** $p_2.fit \leq bestSol.fit$ **then**
26: $bestSol = p_1$
27: **end if**
28: Agregar p_1 y p_2 a *pobc* como nuevos descendientes.
29: **end for**
30: Mezclar y ordenar *pobc* con *pob*
31: Reemplazo generacional $pob = pob[0 : npop]$
32: Almacenar el mejor fit por iteración
33: **end for**

Algoritmo 3 Procesamiento

Entrada: setImagenes, vecSol

function FITT(`setImagenes`, `vecSol`)
  while el archivo csv permanece abierto a escritura **do**
    while $i \leq tImag$ **do**
      Leer imagen (img)
      Recortar img 400 x 400
      Cambiar img a escala de grises
      ▷ Selección de métodos e hiperparámetros
      if $vecSol[0] = 1$ **then**
        Gaussian(img,vecSol[1:3])
      else
        Sobel(img,vecSol[4:6])
      end if
      if $vecSol[7] = 1$ **then**
        umb. Adap(img,vecSol[8:10])
      else
        umb. Simple(img,vecSol[11])
      end if
      Aplicar operación de erosión sobre img
      extraercontorno(img, vecSol[12])
      Guardar contornos en archivo
    end while
    Cerrar archivo
  end while
  ▷ Reducir dimensionalidad
  while el archivo csv permanece abierto a escritura **do**
    Abrir archivo de contorno
    while $lineas \leq null$ **do**
      Leer serie
      if $vecSol[13] = 1$ **then**
        paa($serie$, vecSol[14])
      else
        sax ($serie$, vecSol[15:16])
      end if
      Guardar serie reducida en archivo
    end while
    Cerrar archivo
  end while
  ▷ Clasificación
  if $vecSol[17] = 1$ **then**
    kNN($'Euclidean'$, $vecSol[18]$, $setSerTrain$,$setSerTest$)
  else
    kNN($'DTW'$, $vecSol[18]$, $setSeries$,,$setSerTest$)
  end if
  ▷ Error promedio
  for $i \leq totEtiq$ **do**
    if $y_p[i] \leq y_r$ **then**
      cont ++
    end if
  end for
  `prom = cont/totEtiq`
  return `index`
end function

Capítulo 5

Experimentos y resultados

En esta capítulo se describirá la etapa experimental y los resultados obtenidos de la metodología propuesta. El capítulo se compone de cuatro secciones que son: configuración experimental, comparación del rendimiento del algoritmo con diferentes técnicas de codificación, análisis de convergencia y análisis de la solución final.

Características del equipo

Dentro de las primeras pruebas realizadas para la configuración de parámetros y los experimentos sobre el conjunto de entrenamiento, se ejecutaron en un equipo de escritorio con las siguientes características de software: sistema operativo Ubuntu versión 22.04 y Visual Studio Code versión 1.75.1 como entorno de desarrollo. En cuanto al hardware, el equipo presenta una capacidad en RAM de 8 GB y un procesador Intel Core i5.
En la fase de pruebas para validar los modelos obtenidos, se utilizó Colab (Colaboratory), un producto proporcionado por Google Research que es un entorno de ejecución de código de Python en la nube, diseñado principalmente para el aprendizaje automático, el análisis de datos y la educación [34]; dado que Colab es un servicio en la nube, su ejecución se realizó en el navegador web Firefox. El entorno Colab proporcionó los siguientes recursos: 12.7 GB de RAM y un disco de 107.7 GB. Se decidió trabajar en Colab para las pruebas debido a que, el objetivo de esta tesis no es demostrar que la ejecución en una computadora personal es más rápida que en una virtual, ya que esto ha sido demostrado en otras investigaciones [4]. Más bien, se busca probar que un modelo generado a través de un algoritmo genético para tareas de reconocimiento de patrones puede mejorar la clasificación de imágenes de aves, considerando que la morfología de la cabeza es la que mejor la describe.

5.1. Configuración experimental

Los algoritmos genéticos dependen en gran medida del tipo de representación, de la configuración de los parámetros generales; tales como, el tamaño de población, porcentaje de cruza y porcentaje de mutación. Dichos valores se determinaron según el espacio de búsqueda proporcionado por los rangos de los hiperparámetros del problema, por tanto, el tamaño de población para esta investigación se determinó de 50 individuos y 50 generaciones obteniendo así 2500 iteraciones. Aunque los valores de configuración dependían de los rangos de los métodos, no es el caso para la configuración del porcentaje de cruzamiento (PC) y mutación (PM), para ello fue necesario realizar tres experimentos con valores distintos, tomados de los propuestos en la literatura [19].
Para este experimento previo a la generación de los modelos, se utilizaron las imágenes del conjunto de entrenamiento, y se consideró la codificación real para su implementación. En la Tabla 5.1 se muestran las configuraciones de cada ejecución.

Para el análisis de los resultados se aplicaron medidas de tendencia central; media, mediana y desviación estándar por cada ejecución, también se obtuvo el mejor y peor error. Estos resultados se muestran en la Tabla 5.1, donde se observa que las configuraciones de la tercera ejecución obtienen un mejor error que las ejecuciones uno y dos, y lo mismo ocurre con la media y mediana; sin embargo, la diferencia en la media y mediana de la ejecución uno y dos es mínima. Para hacer una comparación basada en pruebas estadísticas entre los datos de cada ejecución, se aplicó la prueba de Wilcoxon, una prueba no paramétrica que determina si las muestras están relacionadas o si existe una diferencia, se consideró un nivel de confianza del 95 %, es decir, una significancia de 0.05. Esta prueba demostró que sí existe una diferencia significativa entre las ejecuciones, pero la variación entre la ejecución uno y dos con respecto a la tres es mayor; por lo tanto, convencionalmente, las configuraciones de la tercera ejecución se determinaron como los hiperparámetros del algoritmo genético, con porcentajes de cruzamiento y mutación de 0.7 y 0.2, respectivamente, ver Tabla 5.2.

5.2. Comparación del rendimiento de algoritmos con diferentes técnicas de codificación.

Debido a que se consideraron dos técnicas de codificación, es necesario hacer una comparación de su desempeño en cuestión del porcentaje de error y tiempo de ejecución, para ello se implementaron diez ejecuciones sobre cada codificación, proporcionando diversos errores obtenidos de la clasificación; dado que se trata de la etapa de entrenamiento se dispuso del conjunto de imágenes previamente destinadas

	Parámetros	Ejec.1	Ejec.2	Ejec. 3
Porcentaje de PC y PM	Gen.	50	50	50
	Pob.	50	50	50
	PC	0.9	0.8	0.7
	PM	0.1	0.15	0.2
b	Mejor	0.3659	0.3652	0.3485
	Peor	0.4438	0.4090	0.4438
	Media	0.3801	0.3725	0.3589
	Mediana	0.3659	0.3652	0.3485
	Desv. Est.	0.020	0.014	0.020

Tabla 5.1: Medidas de tendencia central

	Parámetros
Gen.	50
Pob.	50
Pc.	0.7
Pm	0.2

Tabla 5.2: Configuración final del Algoritmo Genético

para esta tarea, en la Figura 5.1, se muestran algunas imágenes pertenecientes a dicho conjunto, donde las aves se encuentran en ambientes reales.
Durante el entrenamiento cada codificación obtuvo el rendimiento mostrado en la Tabla 5.6 y 5.5 respectivamente; donde se puede observar que la codificación binaria tiene el menor y mayor error, con respecto a la real, sin embargo sus errores son más variados que en la otra codificación.
Para tener una mejor visión del desempeño se aplicaron medidas de tendencia central a cada resultado, a partir de estos se muestran la media, mediana, desviación estándar, el mejor y peor en la Tabla 5.7. En ellos se hace notorio que, durante el entrenamiento la codificación binaria, produce un buen desempeño puesto que el error y mediana, son mejores, aunque, en la media hay diferencia con la codificación real. Para comparar los datos de cada representación, se aplicó la prueba no paramétrica de Wilcoxon con 0.05 de significancia obteniendo que, el valor p es de 0.0353. Esta prueba demostró que se cumple la condición $p < 0{,}05$ existiendo diferencia significativa.

En cuestión del tiempo de ejecución para la codificación real durante el entrenamiento, presentados en la Tabla 5.3, se observa que; las ejecuciones uno, dos y cuatro fueron las más rápidas, aunque el mejor error se ubique en la ejecución siete con un tiempo de procesamiento de 333 minutos. Esto podría deberse a que a los métodos que se seleccionan según el modelo obtenido, son costosos al procesar las imágenes y además que durante la ejecución se tenían otras aplicaciones en ejecución.
Para la codificación binaria, la Tabla 5.4 muestra su desempeño basado en el tiempo de entrenamiento, en este caso podemos observar que en la mayoría son menores a la

((a)) Luis Bienteveo (Pitangus Sulphuratus) ((b)) Luis Bienteveo (Pitangus Sulphuratus) ((c)) Luis Bienteveo (Pitangus Sulphuratus)

((d)) Carpintero Lineado (Dryocopus Lineatus) ((e)) Carpintero Lineado (Dryocopus Lineatus) ((f)) Carpintero Lineado (Dryocopus Lineatus)

((g)) Zopilote común (Coragyps Atratus) ((h)) Zopilote común (Coragyps Atratus) ((i)) Zopilote común (Coragyps Atratus)

Figura. 5.1: Ejemplo de imágenes del conjunto de entrenamiento

ejecución real, y aunque el menor error está ubicado en la ejecución siete el tiempo de este es menor en comparación al tiempo del real; a diferencia de la codificación real en la binaria el mayor tiempo coincide con el peor error durante el entrenamiento, esto causado, posiblemente a la combinación de métodos y los hiperparámetros dados por el modelo.

De la fase de entrenamiento se obtuvieron diez modelos por cada codificación; estos modelos usados para el entrenamiento fueron evaluados sobre un conjunto de imágenes de pruebas, calculando también el error de clasificación usando cross-validation.

Los resultados obtenidos por cada uno se muestran en la Tabla 5.6 y 5.5, en la columna tres. A simple vista se puede observar cómo el error producido por la codificación binaria es mayor que en la codificación real, sin embargo, el que produce

Ejecución	Tiempo (min)
1	295
2	288
3	314
4	295
5	331
6	303
7	333
8	328
9	313
10	336

Tabla 5.3: Tiempo de ejecución en minutos de la codificación real del entrenamiento

Ejecución	Tiempo (min)
1	367
2	594
3	370
4	228
5	215
6	211
7	219
8	211
9	215
10	194

Tabla 5.4: Tiempo de ejecución para la codificación binaria en el entrenamiento

Ejecución	Error de entrenamiento	Error de prueba
1	0.3655	0.5584
2	0.3655	0.5584
3	0.3572	0.54321
4	0.3655	0.5561
5	0.3518	0.5511
6	0.3768	0.5524
7	0.3514	0.5468
8	0.3688	0.5381
9	0.3688	0.5353
10	0.3518	0.5349

Tabla 5.5: Tabla de error por mejor modelo de la representación real

un mejor resultado es la representación binaria.

Las pruebas estadísticas de la Tabla 5.8 muestran como la media y la mediana tanto en la representación binaria como en la representación real se aproximan; en la desviación estándar los errores producidos por los modelos binarios muestran que

Ejecución	Error de entrenamiento	Error de prueba
1	0.3768	0.4766
2	0.4264	0.5537
3	0.3775	0.5491
4	0.3507	0.5672
5	0.3684	0.5764
6	0.3507	0.5795
7	0.3597	0.5867
8	0.3594	0.5888
9	0.3416	0.5685
10	0.3927	0.5675

Tabla 5.6: Tabla de error por mejor modelo de la representación binaria

	Vector real	Vector binario
Mejor	0.351	0.342
Peor	0.377	0.426
Media	0.362	0.37
Mediana	0.366	0.364
Desv_Es	0.009	0.025

Tabla 5.7: Tabla de error en el entrenamiento por cada representación

	Vector_Real	Vector_Binario
Mejor	0.535	0.477
Peor	0.558	0.589
Media	0.548	0.561
Mediana	0.549	0.568
Desv_Es	0.009	0.032

Tabla 5.8: Tabla de error en la prueba por cada representación

están más dispersos o que hay más variación, por lo tanto, si los errores producidos por la codificación real están menos dispersos la mayoría de los modelos obtenidos no varian en los hiperparámetros utilizados, en cada tarea. La prueba de Wilcoxon demuestra que no hay diferencia significativa en ambos conjuntos, ya que el valor de p es de 0.1054, por lo tanto, no hay diferencia en entre ellos.

5.3. Análisis de convergencia

Para el análsis de convergencia de las técnicas de codificación, se consideró tomar de las diez ejecuciones el error que más se aproxima a la mediana, realizando la gráfica de convergencia de acuerdo a su índice de ejecución; así mismo se planteó graficar el menor error del conjunto de ejecuciones. El gráfico se compone del error

dado por la validación cruzada en el eje de las y, con respecto al total de iteraciones en el eje de las x.

En la Figura 5.2, se muestra la gráfica de convergencia del más próximo a la mediana, en ella se observa que la codificación binaria inicia con un error alto pero descendiendo continuamente, aunque se estanca en un óptimo local temporalmente, alcanza el óptimo en las últimas iteraciones; en cambio la codificación real se estanca en un óptimo local desde las primeras iteraciones, pero iniciando con un error bajo en comparación del binario.
La Figura 5.3 muestra la convergencia del mejor error de cada técnica; se observa inicalmente en la codificación binaria que el error esta sobre el 0.48 aun así se desempeña bien hasta que, aproximadamente en la iteración doce convergen más rápido, estancandose en un óptimo local. En el caso de la codificación real ésta inicia con un error relativamente bajo al binario minimizando el error de manera progresiva, quedando en un óptimo local cerca de las 28 iteraciones, indicando una mayor exploración del espacio de búsqueda que la codificación binaria.
Las gráficas de convergencia muestran que ambas técnicas de codificación cumplen con la función de minimizar el error de clasificación. Aun cuando esto es posible, los mejores resultados son dados por la codificación binaria en los dos casos expuestos. Tomando en cuenta estos errores se propuso definirlos como modelos de solución y

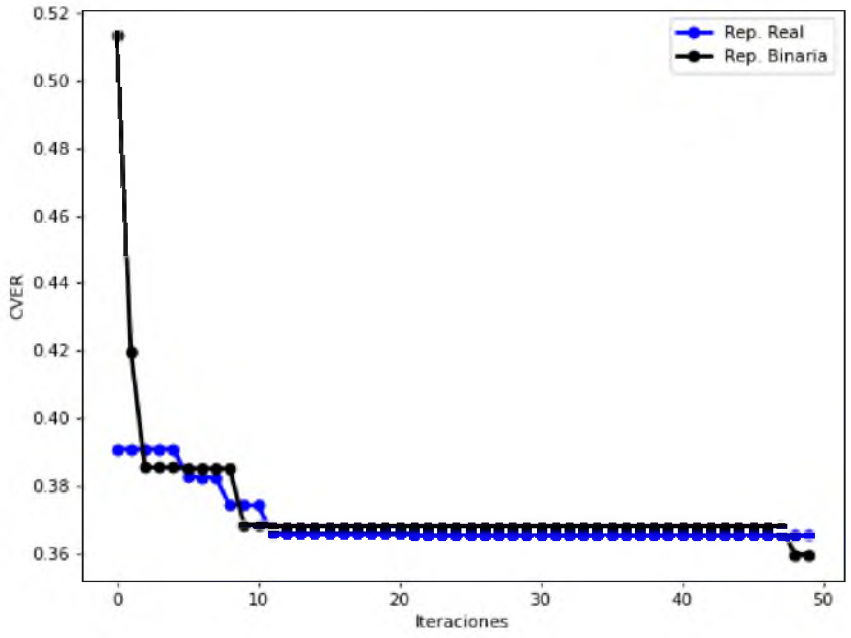


Figura. 5.2: Convergencia de los modelos aproximados a la mediana en ambas codificaciones

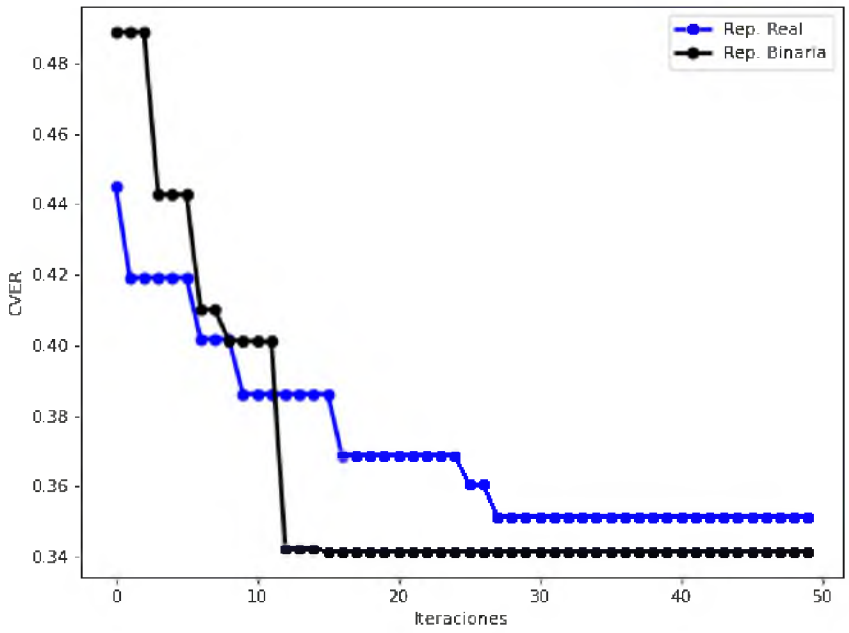


Figura. 5.3: Convergencia el mejor error por codificación

de esta manera realizar un análisis sobre los modelos obtenidos.

5.4. Análisis de la solución real

En esta sección se analizarán los métodos e hiperparámetros que proporcionó el modelo evolutivo propuesto, los cuales se muestran en la Tabla 5.9 donde los primeros cuatro modelos son tomados del mejor y peor error de cada codificación, los siguientes cuatro son considerados del error más próximo a la media y mediana obtenidos de la fase de entrenamiento.

Dentro del modelo del mejor error; se visualiza que, para la tarea de filtrado hay una diferencia entre la codificación real y binaria, ya que para la codificación real el método que mejora más la imagen es el filtro Gaussiano; y para la codificación binaria es el filtro Sobel; también se observa que hay una diferencia entre los métodos para la segmentación, ya que el primero selecciona la umbralización adaptativa donde se optimizan tres hiperparámetros: el método del cálculo de umbral, el tipo de umbralización y por último un kernel; en cambio, la codificación binaria implementa la umbralización simple que sólo emplea el limite de umbral, para detectar un cambio del valor de los pixeles; en el caso de la extracción de características y en la clasificación difieren en los métodos de aproximación de contornos y del valor optimizado en el hiperparámetro tamaño de la ventana del método PAA, así como, en el tipo de cálculo de la distancia entre los k vecinos. En cuestión del tiempo de

ejecución, el modelo dado por la codificación binaria tuvo el mejor desempeño.

Para los modelos dados en el peor error, los métodos optimizados fueron los mismos para las tareas de segmentación, y dentro de la extracción de características sólo difieren en el tipo de aproximación de contornos en ambas codificaciones; en el filtrado y clasificación los métodos son distintos. Para este caso el modelo de la codificación binaria fue la más costosa en tiempo de ejecución y tuvo el peor error.
En los modelos más próximos a la mediana y media nos muestran cómo; los mismos métodos para las tareas de filtrado, segmentación, extracción de características son óptimos en ambas codificaciones, claro está, que con distintos hiperparámetros; sin embargo, en el caso de la clasificación en el modelo real es más óptimo aplicando distancia Euclidiana (ED); y el modelo binario aplicando Dynamic Time Warping (DTW).

De manera general los métodos más utilizados en la tarea de filtrado es el filtro Gaussiano, e incluso se puede observar que, aunque los seis modelos seleccionaron este filtro, proporcionan diferentes errores, esto a causa de sus hiperparámetros; en la segmentación, la umbralización adaptativa es la más óptima; y en el caso de la tarea de extracción de características, para los ocho modelos el más óptimo es (PAA). Es evidente que para la clasificación, en la codificación real es más óptimo aplicar ED (distancia Euclidiana) y para la binaria DTW (Dynamic Time Warping), ya que éstas se mantuvieron constantes en los ocho modelos.

Para ampliar el conocimiento de qué métodos fueron mayormente seleccionados, se realizó un gráfico de frecuencia a partir de los modelos obtenidos en cada ejecución durante la fase de entrenamiento; teniendo en cuenta ambas técnicas de codificación, esto a su vez, nos podrá ayudar para hacer una comparación entre las codificaciones, de la misma manera en observar que métodos son los más óptimos. La gráfica se muestra en la Figura 5.4, donde, para ambas codificaciones el filtro Gaussiano es el más óptimo, así también, en el caso de la segmentación, la umbralización adaptativa es la que tiene mayor frecuencia, sobre todo en la codificación real ya que todos los modelos lo aplicaron; caso contrario a la umbralización simple, que fue seleccionada muy escasas veces por la codificación binaria. Para las tareas de extracción de características, en lo que se refiere al modo de recuperación de contornos, fue más eficiente el método uno para la codificación real en todas las ejecuciones, aunque en la codificación binaria es más óptima la segunda opción. En el caso de la reducción de series temporales el método PAA es el que mejor se adapta en ambas técnicas real y binaria, a diferencia de las métricas de distancias en la clasificación ya que, para la codificación binaria la más frecuente es la distancia Dynamic Time Warping

Codificación	Filtrado	Segmentación	Extracción de características	Clasificación	Error	Tiempo de ejecución (min)
		Mejor error				
Real	Gauss: 1, Kernel: 3, desvX: 8.44, desvY: 4.63	UmbAdp:1, TCU: 1, TU: 2, kernel: 11	Aprox. C.:1, PAA:1, w:42	ED:1. n:35	0.351	336
Binaria	Sob: 2, Kernel: 3, BordX: 0, BordY: 1	UmbS.:2, Umbral: 38	Aprox. C: 2, PAA: 1, w:66	DTW: 2, n:45	0.342	215
		Peor error				
Real	Gauss: 1, kernel: 3, desvX: 10:, desvY: 8.44	UmbAdp:1, TCU: 1, TU: 2, kernel: 21	Aprox.C.: 1, PAA: 1, w:51	ED: 1, n: 23	0.377	303
Binaria	Sob: 2, kernel: 7, BordX: 0, BordY: 1	UmbAdp:1, TCU: 2, TU: 1, kernel: 3	Aprox.C.: 2, PAA: 1, w:48	DTW: 2, n: 63	0.426	594
		Error próximo a la media				
Real	Gauss: 1, kernel: 3, desvX: 10:, desvY: 3.78	UmbAdp:1, TCU: 1, TU: 2, kernel: 11	Aprox.C.: 1, PAA: 1, w:30	ED: 1, n: 44	0.357	314
Binaria	Gauss: 1, kernel: 7, desvX: 7:, desvY: 8	UmbAdp:1, TCU: 2, TU: 2, kernel: 3	Aprox.C.: 2, PAA: 1, w:61	DTW:2, n: 23	0.368	215
		Error próximo a la mediana				
Real	Gauss: 1, kernel: 3, desvX: 8.44:, desvY: 4.63	UmbAdp:1, TCU: 1, TU: 2, kernel: 21	Aprox.C.: 1, PAA: 1, w: 55	ED:1, n:67	0.366	295
Binaria	Gauss: 1, kernel: 3, desvX: 1:, desvY: 3	UmbAdp:1, TCU: 1, TU: 2, kernel: 3	Aprox.C.: 2, PAA: 1, w:34	DTW:2, n:55	0.364	219

Tabla 5.9: Métodos e hiperparámetros seleccionados por el Algoritmo Genético

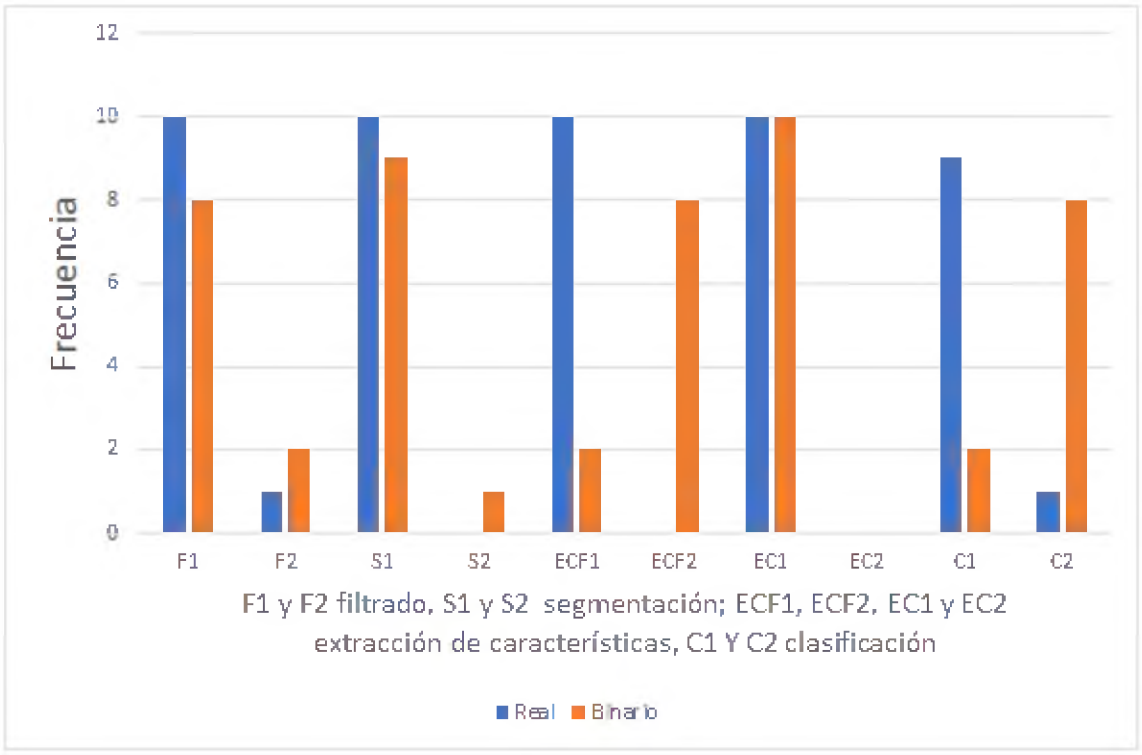


Figura. 5.4: Gráfica de frecuencia de los métodos seleccionados por cada codificación.

(DTW) y Euclidiana (ED) para la real.

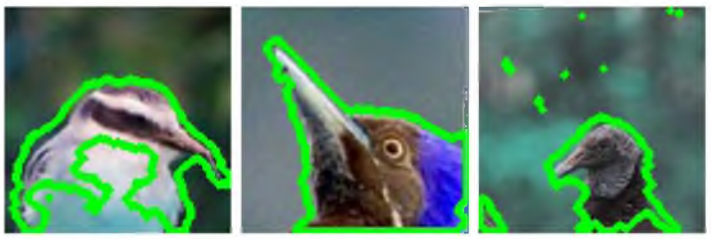
Debido a que nuestro interés es obtener la mejor representación de la cabeza del ave; se aplicaron los modelos obtenidos por la mediana y el mejor fit para visualizar su desempeño; a continuación se muestran las imágenes generadas.

((a)) Luis Bienteveo ((b)) Carpintero común ((c)) Zopilote común

((d)) Luis Bienteveo ((e)) Carpintero común ((f)) Zopilote común

((g)) Luis Bienteveo ((h)) Carpintero común ((i)) Zopilote común

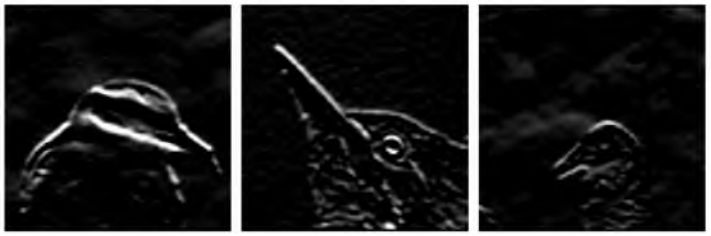
((j)) Luis Bienteveo ((k)) Carpintero común ((l)) Zopilote común



((m)) Luis Bienteveo ((n)) Carpintero común ((ñ)) Zopilote común

Figura. 5.5: Mejor modelo de la representación real: de la $a - c$: recorte; $d - f$: filtro; $g - i$: segmentada; $j - l$: eroción; $n -$: contornos.

((a)) Luis Bienteveo ((b)) Carpintero común ((c)) Zopilote común



((d)) Luis Bienteveo ((e)) Carpintero común ((f)) Zopilote común

((g)) Luis Bienteveo ((h)) Carpintero común ((i)) Zopilote común

((j)) Luis Bienteveo ((k)) Carpintero común ((l)) Zopilote común

((m)) Luis Bienteveo ((n)) Carpintero común ((ñ)) Zopilote común

Figura. 5.6: Mejor modelo de la representación binaria: de la $a - c$: recorte; $d - f$: filtro; $g - i$: segmentada; $j - l$: eroción; $n -$: contornos.

((a)) Luis Bienteveo ((b)) Carpintero común ((c)) Zopilote común

((d)) Luis Bienteveo ((e)) Carpintero común ((f)) Zopilote común

((g)) Luis Bienteveo ((h)) Carpintero común ((i)) Zopilote común

((j)) Luis Bienteveo ((k)) Carpintero común ((l)) Zopilote común

((m)) Luis Bienteveo ((n)) Carpintero común ((ñ)) Zopilote común

Figura. 5.7: Modelo apróximado a la mediana de la representación real: de la $a - c$: recorte; $d - f$: filtro; $g - i$: segmentada; $j - l$: eroción; $n -$: contornos.

((a)) Luis Bienteveo ((b)) Carpintero común ((c)) Zopilote común

((d)) Luis Bienteveo ((e)) Carpintero común ((f)) Zopilote común

((g)) Luis Bienteveo ((h)) Carpintero común ((i)) Zopilote común

((j)) Luis Bienteveo ((k)) Carpintero común ((l)) Zopilote común

((m)) Luis Bienteveo ((n)) Carpintero común ((ñ)) Zopilote común

Figura. 5.8: Modelo apróximado a la mediana de la codificación binaria: de la $a - c$: recorte; $d - f$: filtro; $g - i$: segmentada; $j - l$: eroción; $n -$: contornos.

Capítulo 6

Conclusiones y trabajos futuros

En este capítulo se expondrán las conclusiones del desarrollo de esta tesis, así también se especificarán los trabajos a futuro que puedan mejorar el ya realizado.

6.1. Conclusiones

El principal enfoque para el desarrollo de esta tesis consistió en clasificar imágenes de aves basándose en una metodología semi-automática que aplicaba técnicas de procesamiento de imágenes, teniendo en cuenta que la cabeza del ave sería la característica morfológica que más la describiría. Es importante mencionar que las imágenes de las aves se encontraban en un ambiente natural.

La metodología propuesta se basó de algoritmos metaheurísticos para buscar la combinación de métodos e hiperparámetros que mejor describieran la morfología de la cabeza del ave, y así obtener un porcentaje de clasificación competitivo. Para ello, se consultó la literatura, donde se encontró que las metaheurísticas propuestas sólo abarcaban una tarea, ya fuera el preprocesamiento, la segmentación o la clasificación. Sin embargo, se demostró la posibilidad de aplicar estos algoritmos para resolver diferentes tareas.

De las metaheurísticas presentes en la literatura, se implementó un algoritmo genético aplicando dos técnicas de codificación para su comparación. En este sentido, se puede concluir que:

- Desempeño: Bajo el análisis estadístico realizado, se encontró que la codificación binaria es la que proporciona resultados competitivos, en relación de minimizar el error de clasificación. Así también, en relación al tiempo de ejecución esta codificación es más eficiente, aun a pesar de realizar la decodificación y reparación de los valores del vector.

- Las selección de métodos: En este caso se observó a través de un gráfico de frecuencias como, en la codificación real, los modelos de las diez ejecuciones se

enfocan a un sólo método por cada tarea, así el filtro Guassiano, segmentación mediante umbralización adaptativa, modo de recuperación de bordes externos, método PAA, y la métrica ED son los más óptimos. Es evidente como a pesar de que los modelos son muy parecidos los errores estan aproximados entre ellos, esto debido a que sus hiperparámetros son distintos.
En cambio para la codificación binaria los métodos más óptimos son, filtro sobel, umbralización simple, modo de recuperación de contornos completo, método SAX y DTW.

- Calidad de métodos e hiperparámetros sobre las imágenes: Dentro del interés de desarrollar una metodología semi-automática se encontraba el obtener los mejores métodos e hiperparámetros que nos pudieran proporcionar una buena representación del contorno de la cabeza del ave a través de la imagen. Tal y como se describe en el capítulo 5 que el hecho de aplicar algoritmos metaheurísticos ha dado resultados favorables ya que se obtiene la mejor caracterización de las aves. Aunque de las dos codificaciones se observó que la codificación real es la que más se ajusta a este tipo de problemas, analizándolo desde el aspecto visual.

Finalmente cuando se trabaja en el desarrollo de sistemas basados en machine learning (ML), implementar las prácticas propuestas por la IEEE sobre ingeniería de software es complejo, debido a que no todos los sistemas ML se adaptan a las tareas plantedas por la IEEE. Algunos autores indican incluso que es un nuevo tipo de software y que, posiblemente sea necesario replantear o proponer un estándar que se enfoque sólo a estos sistemas.

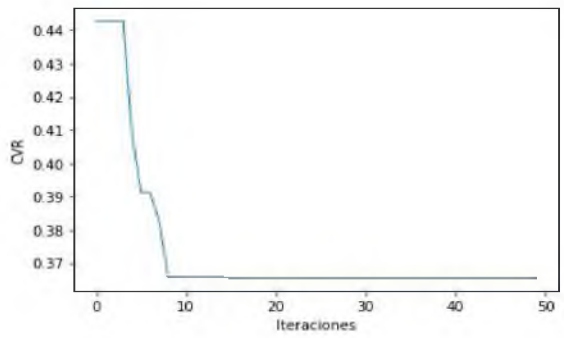
6.2. Trabajos futuros

A continuación se exponen los trabajos a futuro que se han considerado para enriquecer o mejorar este proyecto de tesis.

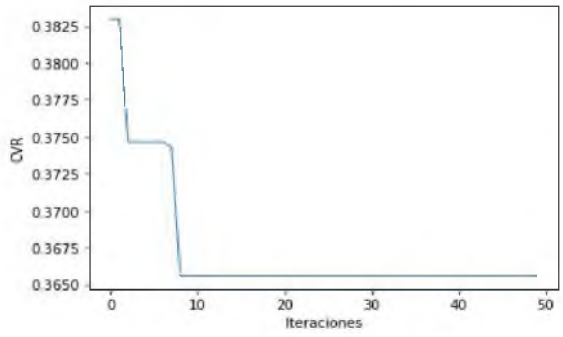
- Trabajar en mejorar la metodología de recorte de la cabeza del ave.
- Implementar un sistema basado en machine learning en una raspberry Pi como herramienta para quienes participan en actividades de ciencia ciudadana.
- Desarrollar un lineamiento de ingeniería de software para sistemas de clasificación de imágenes.
- Comparar los resultados obtenidos con los de otras técnicas de clasificación para determinar la eficacia de los algoritmos genéticos con representación real y binaria en la extracción de la forma de la cabeza de las tres especies de aves.

Apéndice A

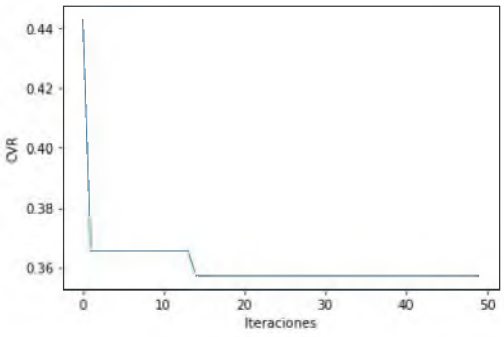
Anexo I: Gráficas de convergencia codificación real



((a))

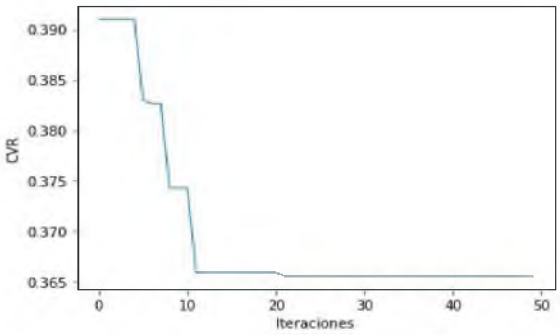


((b))

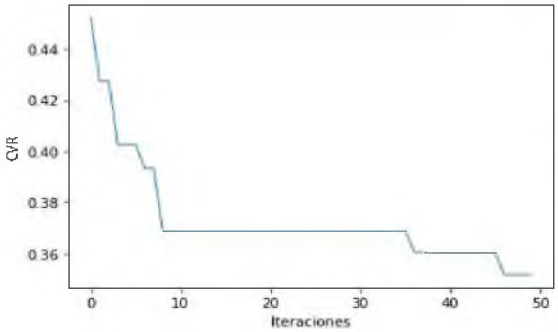


((c))

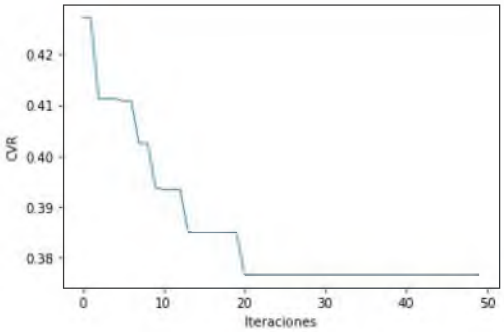
Figura. A.1: Convergencias de la ejecución uno a tres



((a))

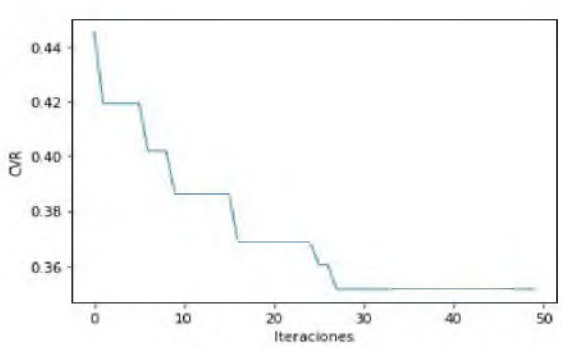


((b))

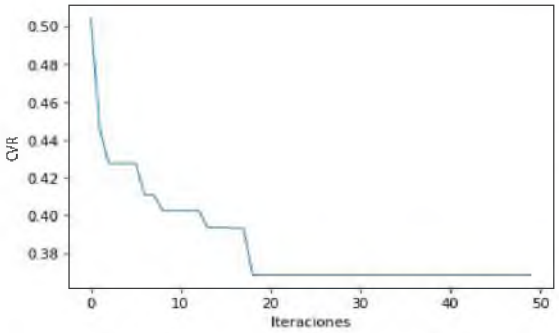


((c))

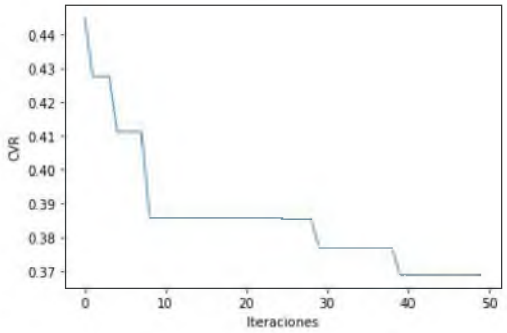
Figura. A.2: Convergencias de la ejecución cuatro a seis



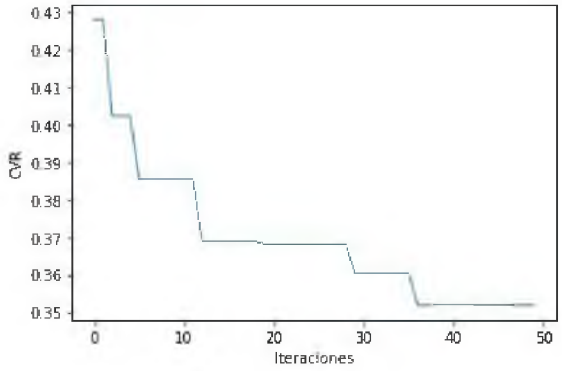
((a))



((b))



((c))

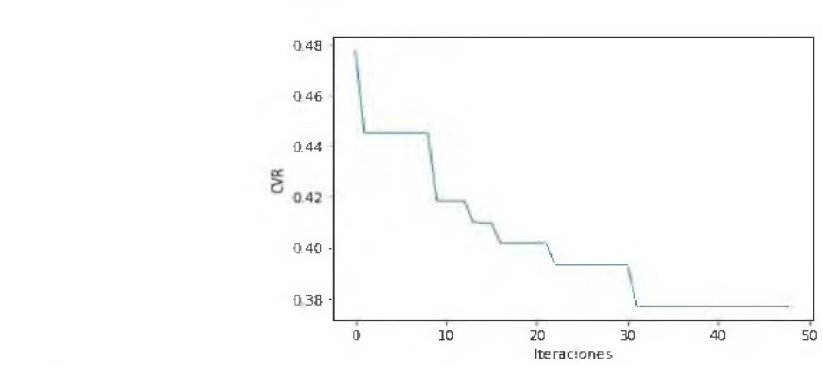


((d))

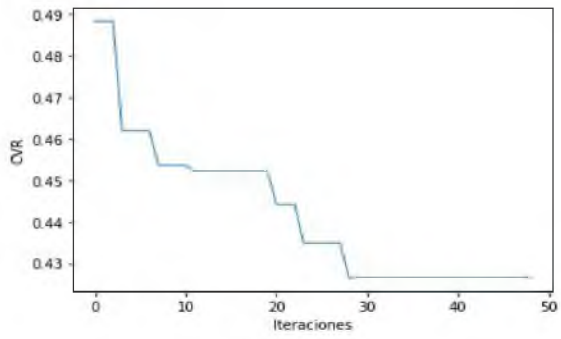
Figura. A.3: Convergencias de la ejecución siete a diez

Apéndice B

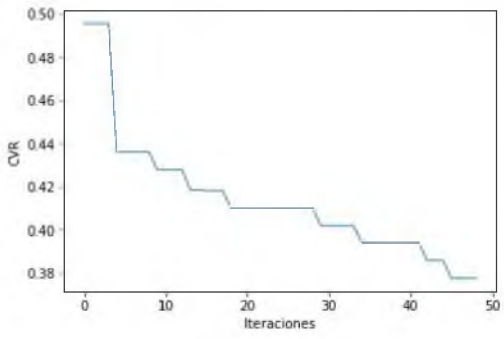
Anexo II:Gráficas de convergencia codificación binaria



((a))

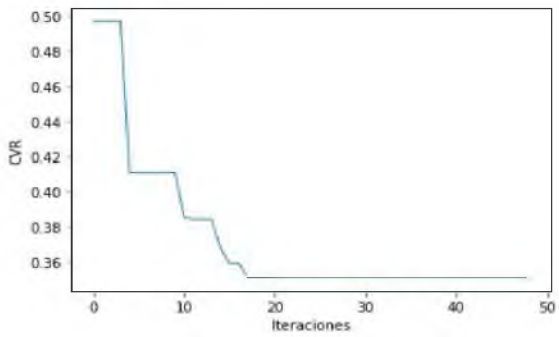


((b))

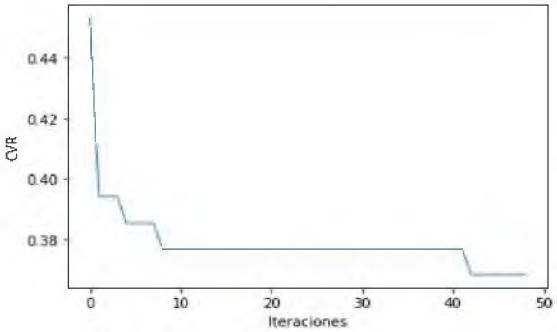


((c))

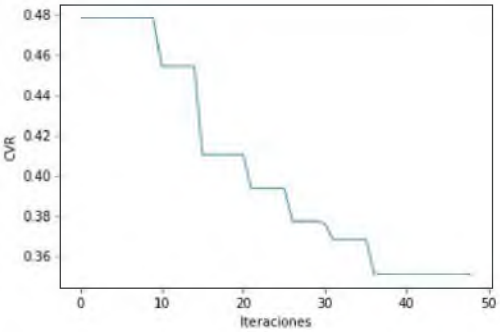
Figura. B.1: Convergencias de la ejecución uno a tres



((a))

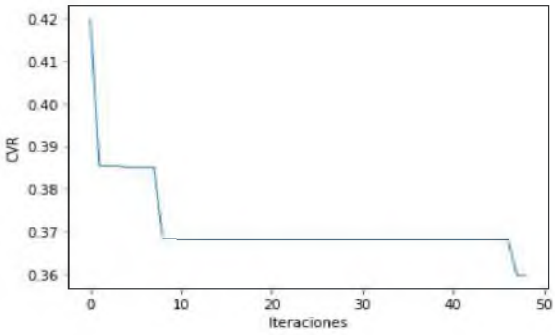


((b))

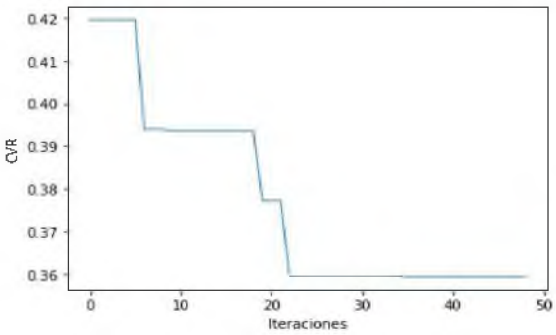


((c))

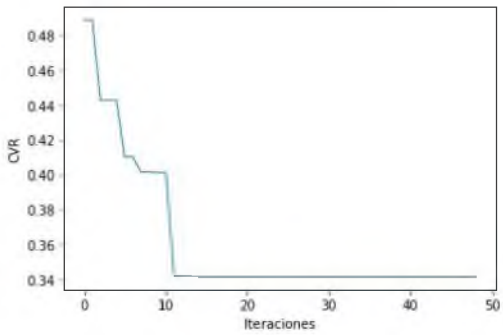
Figura. B.2: Convergencias de la ejecución cuatro a seis.



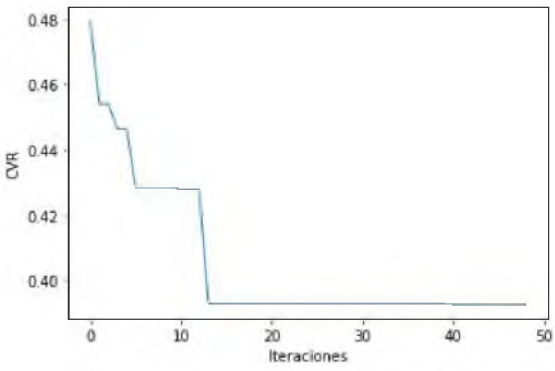
((a))



((b))



((c))



((d))

Figura. B.3: Convergencias de la ejecución siete a diez.

Bibliografia

[1] *¿Qué son los pipelines de datos? | Glosario | HPE España*. https://www.hpe.com/es/es/what-is/data-pipelines.html. (Accessed on 03/30/2023).

[2] *(99+) Tratamiento de Imagenes con MATLAB Erik Cuevas pdf | Ana Betancourt - Academia.edu*. https://www.academia.edu/38570337/Tratamiento_de_Imagenes_con_MATLAB_Erik_Cuevas_pdf. (Accessed on 03/31/2023).

[3] Elizabeth Betancur y Julio Eduardo Cañón Barriga. "La ciencia ciudadana como herramienta de aprendizaje significativo en educación para la conservación de la biodiversidad en Colombia". En: *Revista Cientıfica en Ciencias Ambientales y Sostenibilidad* 3.2 (2016).

[4] Elba M BODERO y col. "Google Colaboratory como alternativa para el procesamiento de una red neuronal convolucional". En: *Revista Espacios* 41.07 (2020).

[5] Lauraborse Borsellino. "El uso de la fotografıa y la Ciencia Ciudadana como herramientas para la conservación de la biodiversidad". En: *Revista Photo & Documento,(3)* (2017).

[6] Edson D Carvalho y col. "An approach to the classification of COVID-19 based on CT scans using convolutional features and genetic algorithms". En: *Computers in biology and medicine* 136 (2021), pág. 104744.

[7] Joel Chacón Castillo. "Mejora del Desempeño de Algoritmos Evolutivos Multiobjetivo con Esquemas de Diversidad en las Variables de Decisión". En: (2017).

[8] Jorge Castro, Roberto Vargas-Masıs y Danny Alfaro-Rojas. "Understanding Variable Performance on Deep MIL Framework for the Acoustic Detection of Tropical Birds". En: (2020).

[9] Carlos Artemio Coello Coello. "Computación Evolutiva". En: ().

[10] J Carlo Cuevas y col. "¡ Sal a pajarear! Una mirada a la observación de aves en México". En: *Órama. Revista Iberoamericana de Divulgación y Cultura Cientıfica* 2 (2018), págs. 29-33.

[11] Pádraig Cunningham. "Dimension Reduction". En: *Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval.* Ed. por Matthieu Cord y Pádraig Cunningham. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, págs. 91-112. ISBN: 978-3-540-75171-7. DOI: 10.1007/978-3-540-75171-7_4. URL: https://doi.org/10.1007/978-3-540-75171-7_4.

[12] S Delgado. "Pokémon, la biodiversidad de los estudiantes". En: *Pero... ¿ Qué pasa con nuestra fauna* 1.1 (2017), págs. 43-44.

[13] Jorge S Delgado Morales y col. "Selección óptima de parámetros para algoritmos de detección de obstáculos con visión monocular". En: *Ingeniería Electrónica, Automática y Comunicaciones* 37.1 (2016), págs. 9-19.

[14] "Dynamic Time Warping". En: *Information Retrieval for Music and Motion.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, págs. 69-84. ISBN: 978-3-540-74048-3. DOI: 10.1007/978-3-540-74048-3_4. URL: https://doi.org/10.1007/978-3-540-74048-3_4.

[15] Susana Finquelievich, Celina Fischnaller y Patricio Julián Feldman. "E-ciencia ciudadana: la "prosumición" de la ciencia en la sociedad del conocimiento". En: *XI Simposio sobre la Sociedad de la Información (SSI)-JAIIO 42 (2013).* 2013.

[16] Luis Alberto Vives Garnique y col. "Visión artificial: Aplicación de filtros y segmentación en imágenes de hojas de café". En: *Ingeniería: Ciencia, Tecnología e Innovación* 1.2 (2014), págs. 71-71.

[17] Görkem Giray. "A software engineering perspective on engineering machine learning systems: State of the art and challenges". En: *Journal of Systems and Software* 180 (2021), pág. 111031.

[18] Pilar Gomez Gil. "El reconocimiento de patrones y su aplicacin a las señales digitales". En: (2018).

[19] Ildefonso Harnisch, Raúl Sanhueza y Horacio Díaz. "Despacho económico con unidades de características no convexas empleando Algoritmos Genéticos". En: *Revista Facultad de Ingeniería* 7 (2000), págs. 13-19.

[20] Santiago Javier Riofrio Hidalgo. "Implementacin de un sistema de reconocimiento de patrones de movimiento con las extremidades superiores del cuerpo". Tesis doct. FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA, ESCUELA POLITÉCNICA NACIONAL, Agosto de 2014.

[21] Nhat-Duc Hoang y Quoc-Lam Nguyen. "Metaheuristic optimized edge detection for recognition of concrete wall cracks: a comparative study on the performances of roberts, prewitt, canny, and sobel algorithms". En: *Advances in Civil Engineering* 2018 (2018).

[22] ISO ISO y TR IEC. "19759: Guide to the Software Engineering Body of Knowledge (SWEBOK)". En: *International Organization for Standardization, Geneva, Switzerland* (2005).

[23] Raúl Moreno Jordán y David González_Solıs. "El papel de la ciencia ciudadana en el monitoreo y manejo sustentable de los recursos naturales". En: (2020).

[24] Eamonn Keogh y col. "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases". En: *Knowledge and Information Systems* 3 (ene. de 2002), págs. 263-286. DOI: 10.1007/PL00011669.

[25] Richard J Larsen y Morris L Marx. *An introduction to mathematical statistics.* Prentice Hall, 2005.

[26] Jessica Lin y col. "Experiencing SAX: a novel symbolic representation of time series". En: *Data Mining and knowledge discovery* 15.2 (2007), págs. 107-144.

[27] Luisa Banet López. "Conocimientos de futuros docentes de Educación Infantil sobre categorización animal". En: *Enseñanza de las ciencias: revista de investigación y experiencias didácticas* Extra (2017), págs. 2127-2134.

[28] Lucy Ellen Lwakatare y col. "A taxonomy of software engineering challenges for machine learning systems: An empirical investigation". En: *Agile Processes in Software Engineering and Extreme Programming: 20th International Conference, XP 2019, Montréal, QC, Canada, May 21–25, 2019, Proceedings 20.* Springer International Publishing. 2019, págs. 227-243.

[29] Silverio Martınez-Fernández y col. "Software engineering for AI-based systems: a survey". En: *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31.2 (2022), págs. 1-59.

[30] Marıa Angélica Mosquera Moreno. "Prototipo de un sistema de visión artificial basado en técnicas de aprendizaje de máquina para el reconocimiento de aves del área metropolitana de Bucaramanga". En: (2021).

[31] Enrique Cabello Pardos. "T´cnicas de reconocimiento facial mediante redes neuronales". Tesis doct. DEPARTAMENTO DE TECNOLOGÍA FOTÓNICA, FACULTAD DE INFORMÁTICA, Abril de 2004.

[32] E. Peiró A. y Uriel Jimenez. *Introducción al análisis de series temporales.* Alfa Centauro, S.A., 2000. ISBN: 9788472881341. URL: https://books.google.com.mx/books?id=SCkUNAAACAAJ.

[33] *Qué es una canalización de datos | IBM.* https://www.ibm.com/es-es/topics/data-pipeline. (Accessed on 03/30/2023).

[34] Google Research.

[35] Rene Reynaga y William Mayta. "Introducción al reconocimiento de patrones". En: *Fides et Ratio-Revista de Difusión cultural y cientifica de la Universidad La Salle en Bolivia* 3.3 (2009), págs. 41-44.

[36] Matias Gabriel Rojas y col. "Optimización de Support Vector Machine mediante metaheurısticas para clasificación de retinopatia diabética". En: (2020).

[37] Juan Carlos Sepúlveda Peña y col. "Guatinı: un proyecto para fomentar el conocimiento de la avifauna cubana en las nuevas generaciones". En: *Revista Ciencia y Agricultura; Volumen 16, número 1 (Enero-Abril 2019)* (2019).

[38] Luis Sucar y Giovani Gmez. "Visin Computacional". En: *Instituto Nacional de Astrofısica, Optica y Electrnica* (2022).

[39] Phan Thanh Noi y Martin Kappas. "Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery". En: *Sensors* 1 (2018). ISSN: 1424-8220. DOI: 10.3390/s18010018. URL: https://www.mdpi.com/1424-8220/18/1/18.

[40] P Estévez Valencia. "Optimización mediante algoritmos genéticos". En: *Anales del Instituto de Ingenieros de Chile*. Vol. 109. 2. 1997, págs. 83-92.