Universidad Autónoma de Querétaro

Facultad de Ingeniería

Maestría en Instrumentación y
Control Automático

Clasificador automático de nubes mediante un algoritmo inteligente embebido

TESIS

Que como parte de los requisitos para obtener el grado de
Maestro en Ciencias en Instrumentación y Control Automático

Presenta:

Edgar Vega Maya

Dirigido por:

Dr. Juvenal Rodríguez Reséndiz

SINODALES

Dr. Juvenal Rodríguez Reséndiz		
Presidente		Firma
Dr. José Manuel Álvarez Alvarado		
Secretario		Firma
Dr. César J. Ortiz Echeverri		
Vocal		Firma
Dra. Diana Carolina Toledo Pérez		
Sinodal		Firma
Omar Rodríguez Abreo		
Sinodal		Firma

Dr. Manuel Toledano Ayala	Dr. Juan Carlos Jáuregui Correa
Director de la Facultad	Director de Investigación y Postgrado

Centro Universitario
Querétaro, QRO
México.
Agosto 2023

A Dios, mi madre la sra. Ma. Guadalupe, mi padre el sr. José Domingo y mi hermana Selene por su amor incondicional.

El que quiera salvarse a sí mismo se perderá;
y el que pierda su vida por causa mía se slavará.
Mt 16:25

Agradecimientos

A Dios por no abandonarme nunca y seguirme mostrando lo maravillosa que puede ser la vida.

A mi familia por su constante apoyo durante este camino, especialmente a mis padres y hermana.

A mi asesor el Dr. Juvenal Rodríguez Reséndiz por resolver todas mis dudas y buscar soluciones y recursos para sacar adelante este proyecto. También al sínodo por sus observaciones y comentarios.

Al Consejo Nacional de Ciencia y Tecnología (Conacyt) por los fondos económicos para poder realizar esta investigación.

A la Universidad Autónoma de Querétaro y la Facultad de Ingeniería por la beca institucional y por las instalaciones, equipo y todo lo necesario para sacar esta tesis adelante.

A mis amigos, compañeros y demás personas que conocí durante este tiempo por los momentos agradables que compartimos.

Abstract

Clouds are among the most significant meteorological phenomena that affect our lives, as they contribute significantly to the energy balance of the earth by absorbing and distributing solar radiation. In addition, they are essential for weather analysis and forecasting. However, ground-based cloud observation mainly depends on human observers, resulting in uncertainties due to subjective judgments. Therefore, research in automatic cloud observation has increased and is still in progress. Different approaches have been published to solve this problem; these approaches range from classical machine learning techniques, such as support vector machines, to deep learning techniques, such as ensembles of convolutional neural networks. The ensemble of two networks has achieved the best performance in terms of classification accuracy. However, since this technique consists of a large number of parameters, which translates into a greater computational load, memory consumption, and processing time, it is difficult for these kinds of algorithms to be implemented in embedded systems with constrained resources. Furthermore, optimizing the hyperparameters of a convolutional neural network belongs to the class of NP-hard problems. This implies that there is no unique set of hyperparameters that performs the same classification on every dataset. As a result, existing architectures that have been proven might not be effective for our cloud dataset. The principal goal of this research is to develop an automatic cloud classifier based on convolutional neural networks while incorporating optimization for the network hyper-parameters. The optimization is intended to maintain the classification accuracy at the state-of-the-art level, while reducing the number of network parameters and, consequently, the memory and computational resources needed. To accomplish this task, a metaheuristic algorithm will be used to obtain the hyper-parameters that yield the best performance. Additionally, the metaheuristic algorithm will be fine-tuned using a method from the chess world for ranking players: the chess rating system (CRS).

Keywords: *automatic cloud classification, metaheuristic algorithms, convolutional neural network, ground-based cloud images.*

Resumen

Las nubes se encuentran entre los fenómenos meteorológicos más significativos que afectan nuestras vidas, ya que contribuyen significativamente al equilibrio energético de la tierra, al absorber y distribuir la radiación solar. Además, son esenciales para el análisis y pronóstico del tiempo. Sin embargo, la observación de nubes desde tierra depende principalmente de observadores humanos, lo que genera incertidumbre debido a los juicios subjetivos. Por lo tanto, la investigación en la observación automática de nubes ha aumentado y todavía está en progreso. Distintas propuestas han sido publicadas para resolver este problema, estas propuestas van desde técnicas de aprendizaje de máquina clásicas como máquinas de vectores de soporte hasta técnicas de aprendizaje profundo como ensambles de redes neuronales convolucionales. Estas últimas han logrado el mejor desempeño en cuanto a porcentaje de clasificación. Sin embargo, debido a la naturaleza del algoritmo este consta de grandes cantidades de parámetros lo cual se traduce en mayor demanda de cómputo, memoria y tiempo de procesamiento, haciendo difícil que estos algoritmos puedan ser implementados en sistemas embebidos con recursos más limitados. Además de que la optimización de los hiper-parámetros de una red neuronal convolucional pertenece a los problemas NP-Complejo, lo que se traduce principalmente en que no existe un conjunto de hiper-parámetros para una red que obtenga el mismo porcentaje de clasificación en cualquier base de imágenes, es decir que, las redes existentes que ya han sido probadas pueden no ser igual de efectivas para nuestra base de imágenes de nubes. El objetivo principal de esta investigación es desarrollar un clasificador automático de nubes basado en redes neuronales convolucionales la cual es optimizada (buscar un conjunto de hiper-parámetros que aumente la eficiencia de la red) de modo que se pueda lograr un porcentaje de clasificación cercano a lo reportado en el estado del arte pero reduciendo significativamente el numero de parámetros de la red y por consiguiente reducir también la memoria y recursos computacionales necesarios para ejecutarlo. Para llevar a cabo esta tarea se utilizó un algoritmo metaheurístico que lleve a cabo la tarea de encontrar los hiper-parámetros con el mejor desempeño. Por otro lado el algoritmo metaheurístico también fue ajustado utilizando un método de ranking de jugadores de ajedrez (*chess rating system*, CRS).

Palabras clave: *clasificación automática de nubes, algoritmos metaheuristicos, redes neuronales convolucionales, imágenes de cielo completo.*

Índice general

Índice de figuras

Índice de tablas

Lista de abreviaciones

CCT	Color Census Transform.
CEA	Comisión Estatal de Aguas.
CENTRIST	Census Transform Histogram.
CNN	Convolutional Neural Network / Red neuronal convolucional.
ICA	Internacional Cloud Atlas.
ICI	Infrared Cloud Imager / Generador de imágenes de nubes en infrarojo.
LOOCV	Leave One Out Cross Validation.
MGCD	MultimodalGround-based Cloud Dataset.
ReLU	Rectified Linear Unit.
RES	Renewable Energy Systems / Sistemas de energías renovables.
SIFT	Scale Inavariant Feature Transform.
SMN	Servicio Meteorológico Nacional.
SVM	Support Vector Machine / Máquina de vectores de soporte.
WMO	World Meteorological Organization / Organización meteorológica mundial.
WSI	Whole Sky Imager / Generador de imágenes de cielo completo.

CAPÍTULO 1

Introducción

Los datos relacionados con las nubes tienen aplicaciones valiosas en diversas áreas. Por ejemplo, en la predicción de la irradiancia normal directa (DNI), la cual tiene un impacto directo en la generación de energía solar. La estimación local del DNI es esencial para lograr una alta precisión en la energía solar de concentración (CSP). Por lo tanto, se han propuesto numerosos trabajos para la predicción del DNI, que incluyen la clasificación, detección y seguimiento de nubes.

Otra dimensión en la que los datos sobre nubes pueden ser cruciales es la agricultura de precisión (AP), donde las nubes influyen tanto en la disponibilidad de radiación solar como en la precipitación necesaria para los cultivos. La AP es un enfoque de gestión agrícola que utiliza tecnología de la información para optimizar la salud y el rendimiento de los cultivos y el suelo [5]. Además, la AP está avanzando hacia sistemas de gestión autónomos, que no requieren supervisión humana. Estos sistemas utilizan imágenes capturadas de manera remota, y la presencia de nubes afecta estos datos al variar la iluminación y reducir la calidad de la imagen y la precisión radiométrica [6].

Este trabajo se centra en la clasificación automática de imágenes de nubes tomadas desde tierra. Se propone un enfoque de aprendizaje profundo basado en redes neuronales convolucionales como método de clasificación de estas imágenes. A diferencia de otros enfoques que también se basan en redes neuronales convolucionales, este trabajo utiliza un algoritmo metaheurístico, específicamente un algoritmo genético, para ajustar los hiperparámetros de la arquitectura de la red neuronal, como el tamaño del filtro, el número de filtros, el número de capas, la función de pérdida, entre otros. A su vez, los hiperparámetros del algoritmo genético, como el tamaño de la población, el número de generaciones, la tasa de mutación, la tasa de supervivencia y el número de fases, también se ajustaron utilizando un método basado en el sistema de ranking de jugadores de ajedrez (*chess rating system*, CRS). El algoritmo clasificador se ha incorporado en una tarjeta de desarrollo Raspberry Pi 3, que trabaja en conjunto con un sistema de adquisición de imágenes en tiempo real basado en una cámara de lente gran angular capaz de capturar imágenes del cielo completo. Estas imágenes se clasifican automáticamente mediante la red neuronal convolucional resultante.

1.1. Justificación

Las nubes son uno de los fenómenos meteorológicos que ejercen mayor influencia en nuestra vida cotidiana. Esto se debe principalmente a dos razones:

1) El impacto de las nubes en el balance de energía de la Tierra debido a su interacción con la radiación solar y terrestre. Esta interacción engloba la reflexión y absorción de partículas en las

nubes, la cual está fuertemente condicionada por su volumen, forma, grosor y composición.

2) El ciclo hidrológico, crucial para el funcionamiento de diversos ecosistemas y actividades humanas, depende de la presencia de ciertos tipos de nubes.

De estos dos puntos surgen diversas aplicaciones, como modelos de pronóstico de lluvia, estimaciones de radiación solar, previsiones climáticas y plantas de generación de energía a través de paneles fotovoltaicos. Además, las nubes también tienen un impacto en nuestra rutina diaria al influir, por ejemplo, en la cantidad de radiación ultravioleta (UV) que alcanza la superficie terrestre [7]. Estas aplicaciones y otras que requieren información precisa acerca de las nubes son la principal motivación detrás de este trabajo. Instituciones como la Comisión Estatal de Aguas (CEA) en Querétaro se encargan de observar, describir y registrar los fenómenos climatológicos, como la nubosidad, a lo largo del tiempo. Estos registros son fundamentales para comprender y predecir las condiciones actuales y futuras.

El desarrollo de este trabajo busca proporcionar una alternativa a las estaciones meteorológicas en términos de observación y descripción de las nubes, además de ofrecer una opción para avanzar en la generación de energías limpias. Además, el hecho de tener el algoritmo clasificador integrado en un dispositivo portátil conlleva una amplia variedad de ventajas y aplicaciones. No se requiere la presencia de un observador experto para obtener información sobre la nubosidad, lo que permite llevar este dispositivo a aplicaciones como invernaderos, agricultura, estaciones meteorológicas locales y todos los contextos donde el cálculo de radiación solar o la predicción de lluvia sean de gran relevancia.

En última instancia, esta investigación podría sentar las bases para futuros trabajos relacionados con el procesamiento de imágenes y el fenómeno meteorológico de las nubes.

1.2. Descripción del Problema

Instituciones como la Comisión Estatal de Aguas (CEA) en Querétaro y el Servicio Meteorológico Nacional (SMN) en todo el país tienen la responsabilidad de vigilar continuamente la atmósfera para identificar fenómenos meteorológicos que puedan generar desastres naturales y afectar diversas actividades económicas. Entre sus actividades principales se encuentran:

- Proporcionar al público información meteorológica y climatológica.
- Realizar estudios climatológicos y meteorológicos.
- Recopilar, revisar, depurar y organizar información para crear el Banco Nacional de Datos Climatológicos, disponible para consulta pública.

Para cumplir con estos objetivos, el estado de Querétaro cuenta con varias estaciones meteorológicas y un radar meteorológico que proporciona información sobre nubosidad, altura de las nubes, densidad y desplazamiento. Sin embargo, la información obtenida vía satélite puede carecer de precisión en observaciones locales, como en localidades o municipios, debido a que las imágenes satelitales abarcan un rango muy amplio.

En la Figura 1.1 se muestra una imagen del radar que brinda información a la CEA en Querétaro, donde se aprecia su amplio rango de visualización.

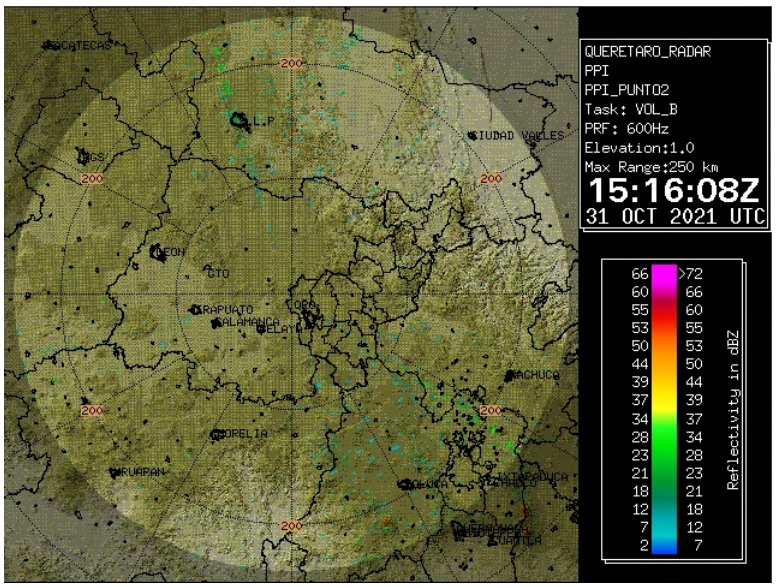


Figura 1.1: Imagen tomada del sitio web de la CEA.

Por esta razón, los sistemas de adquisición de imágenes tomadas desde tierra, como el propuesto en este trabajo, pueden ofrecer una solución al problema al proporcionar información más específica y localizada.

Las imágenes tomadas desde tierra, como el ejemplo en la Figura 1.2, brindan información más detallada sobre el área de captura. Esta información es especialmente relevante para aplicaciones como la predicción de radiación solar o la predicción de lluvia.

Figura 1.2: Imagen con nubes clasificadas como 'cumulus'.

Además, dentro de las estaciones meteorológicas, los registros acerca de las nubes, como la nubosidad y el tipo de nube, dependen de la interpretación de un observador capacitado. Esto

puede llevar a juicios subjetivos o errores por malas interpretaciones. Es por esto que ha crecido la investigación en la clasificación automática, ya que no está sujeta a juicios subjetivos ni errores humanos y puede brindar información más precisa para áreas específicas.

Es importante mencionar que esta investigación no se limita al estado de Querétaro ni a alguna institución en particular, ya que los resultados obtenidos podrían aplicarse en cualquier parte del mundo.

Tabla 1.1: Comparativa presente - futuro del problema que se ha abordado

Situación Actual (es)	**Futuro (debería ser)**
Esta tarea se lleva a cabo por personas capacitadas y los instrumentos existentes solo calculan altura y porcentaje de nubosidad.	Un instrumento de bajo costo que clasifique de forma autómatica.
RNC's robustas pero muy grandes	RNC optimizadas que puedan usarse en sistemas embebidos.
Las RNC pertenecen a los problemas NP-complejo.	

En la Tabla 1.1, se aborda el problema científico que esta investigación busca resolver. En primer lugar, se destaca que la tarea de clasificación automática de nubes actualmente recae en personas capacitadas, y no existe un instrumento que realice esta clasificación de manera automática, de la misma manera en que se mide la nubosidad y la altura de las nubes. En la sección de resultados, se presenta la primera versión o prototipo de dicho instrumento.

Además, los modelos de clasificación basados en redes neuronales en el estado del arte son robustos pero demandan una alta capacidad de cómputo, lo que los hace complicados de implementar en sistemas embebidos de bajo costo. La arquitectura desarrollada en este trabajo resulta ser significativamente más ligera que las arquitecturas más representativas del estado del arte, lo que la hace viable para su uso en un sistema embebido.

1.3. Hipótesis

Un clasificador basado en CNN pertenece a los problemas NP-Complejo y es optimizado con un algoritmo meta-heurístico logrando una precisión mayor al 90 % en clasificación de imágenes de nubes.

1.4. Objetivos

1.4.1. Objetivo general

Desarrollar un clasificador basado en CNN y optimizado con un algoritmo meta-heurístico, para clasificar imágenes de nubes de forma automática, el cual estará embebido en un dispositivo portátil.

1.4.2. Objetivos específicos

Los objetivos específicos de este proyecto se enuncian a continuación:

- Obtener un clasificador basado en CNN que funcionará como base para ser optimizado.
- Implementar un algoritmo meta-heurístico para optimizar los hiper-parámetros del clasificador.
- Verificación de la hipótesis con los resultados obtenidos mediante un reporte de clasificación.
- Construir un dispositivo portátil que incluya un sistema de adquisición y clasificación de imágenes de nubes.

1.5. Revisión de la Literatura

En los últimos años, se ha experimentado un avance significativo en el uso de tecnologías solares [8], especialmente en los Sistemas de Energías Renovables (RES) para la generación de energía eléctrica. Esto incluye tanto parques fotovoltaicos a gran escala como a baja escala [9]. Dado que el conocimiento y la capacidad de predecir la cantidad de radiación solar son esenciales para la planificación y optimización de estos sistemas [10], y considerando que el tipo de nube y el grado de nubosidad son los factores que más afectan la variabilidad de la radiación, la observación y clasificación de las nubes se convierte en una tarea crucial para la predicción de radiación, entre otras aplicaciones [11].

Hasta finales del siglo pasado, la tarea de monitorear variables como el tipo de nube, la altura de la nube y la nubosidad estaba completamente en manos de observadores entrenados, que utilizaban su criterio para medir estas variables. Sin embargo, llevar a cabo estas mediciones mediante observadores humanos implica un consumo considerable de tiempo y esfuerzo. Además, los resultados de estas observaciones podrían carecer de fiabilidad debido a la interpretación subjetiva involucrada [12]. Por lo tanto, en las últimas tres décadas, se han realizado esfuerzos para automatizar la obtención de estas variables (altura de la nube, nubosidad y tipo de nube) mediante técnicas de procesamiento de imágenes.

1.6. Antecedentes

Diversos enfoques han sido propuestos para la identificación y clasificación automática de las nubes. Actualmente, existen distintos dispositivos o sistemas para capturar imágenes de nubes desde la superficie terrestre, entre los cuales se destacan: el Whole Sky Imager (WSI)[13] y el Infrared Cloud Imager (ICI) [14].

Basándose en estos dispositivos, se han desarrollado diversas técnicas y algoritmos para la clasificación automática de nubes. Calbo *et al.* extrajeron características de textura mediante métodos estadísticos y la transformada de Fourier, empleando un algoritmo de paralelepípedo supervisado como clasificador [15]. Heinle *et al.* presentaron un clasificador *k-nearest neighbor* capaz de clasificar siete tipos de nubes, basándose en características de textura, color y tonalidad [16]. Zhuo *et al.* realizaron un pre-procesamiento de las imágenes mediante una transformación por censo de color, y combinaron las características de textura y estructura utilizando la estrategia de píramide espacial

[17]. Kazantzidis *et al.* propusieron un pre-procesamiento mediante umbral multicolor, incorporando estrategias previas de extracción de características y clasificación. Además, consideraron el ángulo cenital y la presencia de gotas de lluvia en las imágenes, ya que estas están asociadas con ciertos tipos de nubes [18]. Cheng *et al.* dividieron las imágenes en bloques debido a que una sola imagen puede contener varios tipos de nubes, lo que podría confundir al clasificador. Para solucionarlo, aplicaron extracción de características de textura a cada bloque utilizando patrones binarios locales y análisis de componentes principales para reducir la dimensión del vector de características. Finalmente, utilizaron una máquina de vectores de soporte como clasificador [19]. Xiao *et al.* extrajeron simultáneamente características de textura, estructura y color de las nubes mediante algoritmos como la Transformación de Característica en Escala Invariable (SIFT), empleando una máquina de vectores de soporte como clasificador [20].

En los últimos años, las Redes Neuronales Convolucionales (CNN) han demostrado un rendimiento notable en diversas áreas, como la clasificación visual, detección de objetos y reconocimiento de voz. Ye *et al.* extrajeron características de las nubes de las capas convolucionales de una red, luego utilizaron la codificación Fisher para vectores y una Máquina de Soporte Vectorial (SVM) para mejorar la precisión de la clasificación [21]. De manera similar, en [1], se utilizó la transferencia de aprendizaje y un ensamble de dos redes previamente probadas en otras aplicaciones (ResNet-50 e Inception-v3).

Otro enfoque estudiado consiste en incorporar variables relacionadas con el tipo de nube, como la temperatura, humedad y velocidad del viento, entre otras. Este enfoque se emplea en [12], donde dicha información se obtiene de una estación meteorológica.

No obstante, la clasificación de nubes sigue siendo una tarea desafiante que sigue evolucionando. Este desafío, junto con la falta de información precisa sobre nubes en estaciones meteorológicas locales, nos motiva a proponer un algoritmo de clasificación de nubes basado en técnicas novedosas de inteligencia artificial, con una precisión superior al 90 %. Una vez que se pruebe con conjuntos de datos relevantes, este algoritmo tendrá la versatilidad para ser implementado tanto en un sistema embebido con capacidades de adquisición de imágenes en tiempo real, como en una aplicación descargable en dispositivos móviles. La disponibilidad de información precisa en tiempo real sobre este fenómeno meteorológico tiene un valor significativo en diversas aplicaciones, como se mencionó anteriormente en esta sección.

Tabla 1.2: Antecedentes.

Año	**Título-Autor**	**Descripción**	**Exactitud**
2008	*Feature extraction from whole-sky ground based images for cloud-type recognition* -J Calbo, J Sabburg	Extrae características estadísticas y basadas en la transformada de Fourier para clasificar las nubes mediante la técnica del paralelepípedo [15].	62 % - 8 clases 76 % 5 clases
2010	*Automatic Cloud classification of whole sky images* -A. Heinle, A. Macke, A. Srivastav	Extrae características de color y textura mediante métodos estadísticos y usa *k-nearest neighbour* como clasificador para distinguir hasta siete tipos de nube [16].	74.58 % - 7 clases medido con LOOCV
2013	*Equipment and methodologies for cloud detection and classification: A review* -R. Tapakis, A. G. Charalambides	Presenta una revisión del equipo y métodos utilizados para la clasificación de nubes tanto satelitales como basados en Tierra [22].	Tabla comparativa con exactitudes subjetivas dado el método, número de clases, instrumento de medición, etc.

2014	*Cloud Classification of Ground-Based Images Using Texture–Structure Features* -W. Zhuo, Z. Cao	Realiza un preprocesamiento de las imágenes mediante CCT , hace extracción de características mediante un algortimo de subdivisión piramidal de la imagen y clasifica con SVM [17].	81.11 % - 6 clases
2015	*Block-based cloud classification with statistical features and distribution of local texture features* -H.Y. Cheng, C.C.Yu	Combina características de textura y agrega local binary pattern para extracción de características locales. Propone un enfoque a partir de dividir cada imagen en bloques y clasifica basado en las características de cada bloque, extrae características de textura locales, aplica análisis de componentes principales y luego compara clasificadores como k-nearest neighbour, clasificador bayesiano con análisis discriminante y SVM [19].	77 % - 6 clases 10-fold-cross validation
2016	*mcloud: A multiview visual feature extraction mechanism for ground-based cloud image categoriza-tion* -Y. Xiao, Z. Cao, W. Zhuo, L. Ye, L. Zhu	Extrae características de textura con SIFT, estructura con CENTRIST y de color. Clasifica con una SVM[20].	81.2 % - 9 clases 87.5 % - 6 clases
2017	*Deepcloud: Ground-based cloud image ca-tegorization using deep convolutional features* -L. Ye, Z. Cao, Y. Xiao	Aplica CNN para extracción de características y utiliza local pattern mining para discriminar entre las características más relevantes. Finalmente utiliza Fisher vector encoding para codificar las características y clasificar con SVM.[21].	86.7 % - 9 clases 89.5 % - 6 clases
2018	*Deep multimodal fusion for ground-based cloud classification in weather station networks* -S. Liu, M. Li	Se hace un ajuste fino a una CNN ya entrenada, y extrae las características de las capas convolucionales y se fusiona con información multimodal (temperatura, velocidad y dirección del viento, humedad, presión) y finalmente se aplica una SVM para clasificar. [12].	86.3 % - 7 clases 10-fold-cross-validation
2021	*A Novel Robust Classification Method for Ground-Based Clouds* -A. Yu, M. Tang, G. Li	Utiliza transferencia de aprendizaje y ensamble de redes para la extracción de características y propone *sparse coding* como método de clasificación. [23].	99.28 % - 7 clases

1.7. Estructura de la tesis

Esta tesis está organizada de la siguiente manera:

- El capítulo 2 presenta la base teórica que sustenta esta tesis, describiendo los algoritmos de clasificación y optimización utilizados, así como las herramientas y algunos conceptos fundamentales.
- El capítulo 3 contiene la metodología empleada en este trabajo, es decir, los pasos a seguir para obtener los resultados presentados en el siguiente capítulo.
- El capítulo 4 aborda la discusión y los resultados obtenidos a partir de la investigación realizada. Además, describe el impacto que esto genera en los distintos sectores y plantea posibles

líneas de trabajo futuro derivadas de esta investigación.

- Finalmente, el capítulo 5 recopila las conclusiones extraídas del trabajo presentado en esta tesis.

CAPÍTULO 2

Marco Teórico

Las imágenes se definen como una función de dos dimensiones, $f(x, y)$, donde x e y son coordenadas espaciales, y la amplitud de f en cualquier par de coordenadas (x, y) se llama intensidad o nivel de grises de la imagen en ese punto. Cuando los valores de x, y y la amplitud son discretos y finitos, nos referimos a la imagen como una imagen digital. El campo del procesamiento digital de imágenes se refiere al procesamiento de estas imágenes mediante una computadora digital. El término más comúnmente utilizado para denotar la unidad mínima en una imagen es el píxel [24].

Una estrategia común para el procesamiento de imágenes, en este caso de imágenes de nubes, involucra tres etapas: preprocesamiento de las imágenes, extracción de características y clasificación. La Figura 2.1 muestra el diagrama de flujo de un algoritmo clasificador de imágenes de nubes.

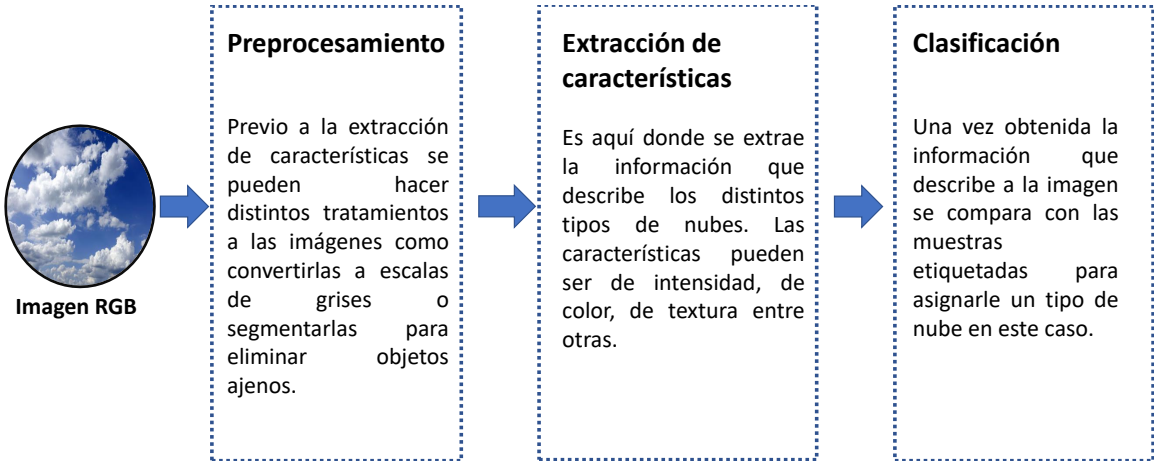


Figura 2.1: Flujo de trabajo típico para un clasificador de imágenes (nubes).

En el cuadro 1.2 de la sección de antecedentes se observan diversas técnicas y métodos que se han utilizado en cada una de las etapas de un algoritmo de clasificación de nubes, entre ellas destacan las redes neuronales convolucionales.

2.1. Redes neuronales convolucionales (CNN)

Las redes neuronales convolucionales están diseñadas para trabajar principalmente con imágenes. Este tipo de datos tienen la característica de tener dependencias espaciales y también que no se ven afectadas por traslaciones en el espacio 2D. Por ejemplo un plátano tiene la misma interpretación cuando esta ubicado al centro o en una esquina de la imagen. Las redes neuronales convolucionales tienden a crear características similares a partir de regiones locales con patrones similares [25].
Estas redes funcionan de manera muy similar a las tradicionales redes *feed-forward*, con la excepción de que las operaciones dentro de las capas están organizadas espacialmente con conexiones no densas y cuidadosamente diseñadas entre ellas [25].
Las tres principales operaciones o tipos de capa que comúnmente se encuentran en una red convolucional son *convolución*, *agrupación (pooling)* y *activación* las cuales se explicarán a continuación y se pueden observar gráficamente en la Figura 2.2.

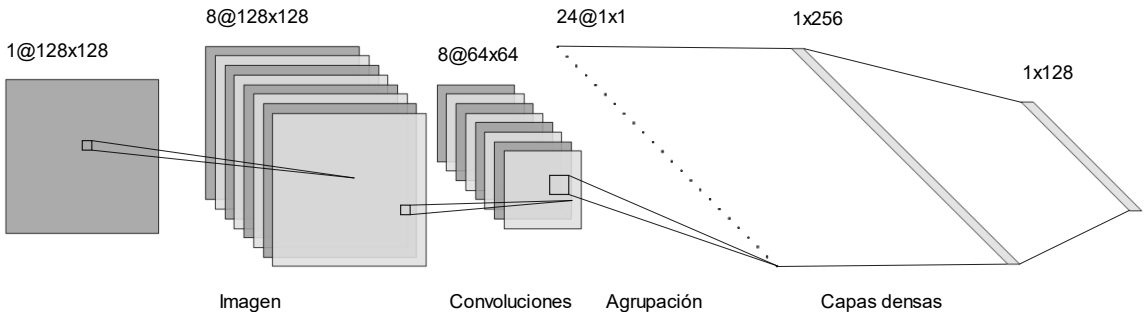


Figura 2.2: Esquema básico de una red neuronal convolucional.

2.1.1. Capas convolucionales

La principal diferencia entre una capa convolucional y una capa densa es la siguiente: las capas densas aprenden patrones globales en el espacio de características de la entrada, una imagen por ejemplo, mientras que las capas convolucionales aprenden patrones locales 2.3: para el caso de una imagen, se pueden encontrar patrones en pequeñas ventanas de la misma.

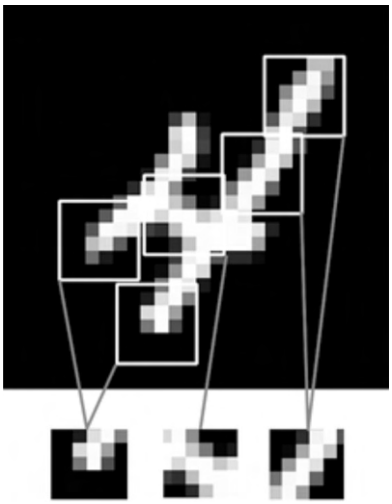


Figura 2.3: Las imágenes pueden ser divididas en patrones locales como bordes, texturas entre otros [1]

De manera general una convolución es una operación de dos funciones: la entrada y el kernel lo cual produce una salida llamada mapa de características. En el campo del aprendizaje de automático, la entrada es usualmente un arreglo multidimensional de datos y el kernel es usualmente un arreglo multidimensional de parámetros que son modificados durante el entrenamiento. Las convoluciones son usadas a menudo sobre un espacio dimensional mayor a uno, si usamos una imagen bi-dimensional I como entrada, seguramente usaremos también un kernel bi-dimensional K:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n). \tag{2.1}$$

Aunque la ecuación en (2.1) define la operación de una convolución, usualmente en una red neuronal convolucional se utiliza una función relacionada llamada correlación cruzada, la cual es igual a la convolución pero aplicando la propiedad de la conmutación como se observa en (2.2):

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n). \tag{2.2}$$

Las redes neuronales tradicionales utilizan la multiplicación de matrices por una matriz de parámetros que contiene un parámetro por separado que describe la interacción entre cada unidad de entrada y cada unidad de salida. Esto significa que cada unidad de salida interactúa con todas las unidades de entrada. Sin embargo las redes convolucionales comúnmente tienen conexiones dispersas (no densas o totalmente conectadas). Esto se logra haciendo que el kernel sea más pequeño que la entrada. Por ejemplo, cuando se procesa una imagen, ésta puede tener miles o millones de píxeles, pero se pueden detectar características significativas como bordes con kernels que abordan decenas o cientos de píxeles. Esto significa que necesitamos almacenar menos parámetros, lo cual reduce tanto los requerimientos de memoria del modelo y aumenta la eficiencia [2]. La capa de convolución superpone el kernel en cada posible posición en la imagen y realiza una convolución o correlación cruzada entre los parámetros del kernel y los datos de entrada. Las dimensiones del mapa de características resultante esta dado por:

$$(m - m_k + 1)\ x\ (n - n_k + 1)\ x\ p \tag{2.3}$$

donde m y m_k son el alto de la imagen y el kernel respectivamente, n y n_k son el ancho y p es el numero de kernels o filtros aplicados. En la Figura 2.4 se muestra un ejemplo de una convolución, donde los valores de los píxeles de la imagen y los parámetros del kernel están representados por letras.

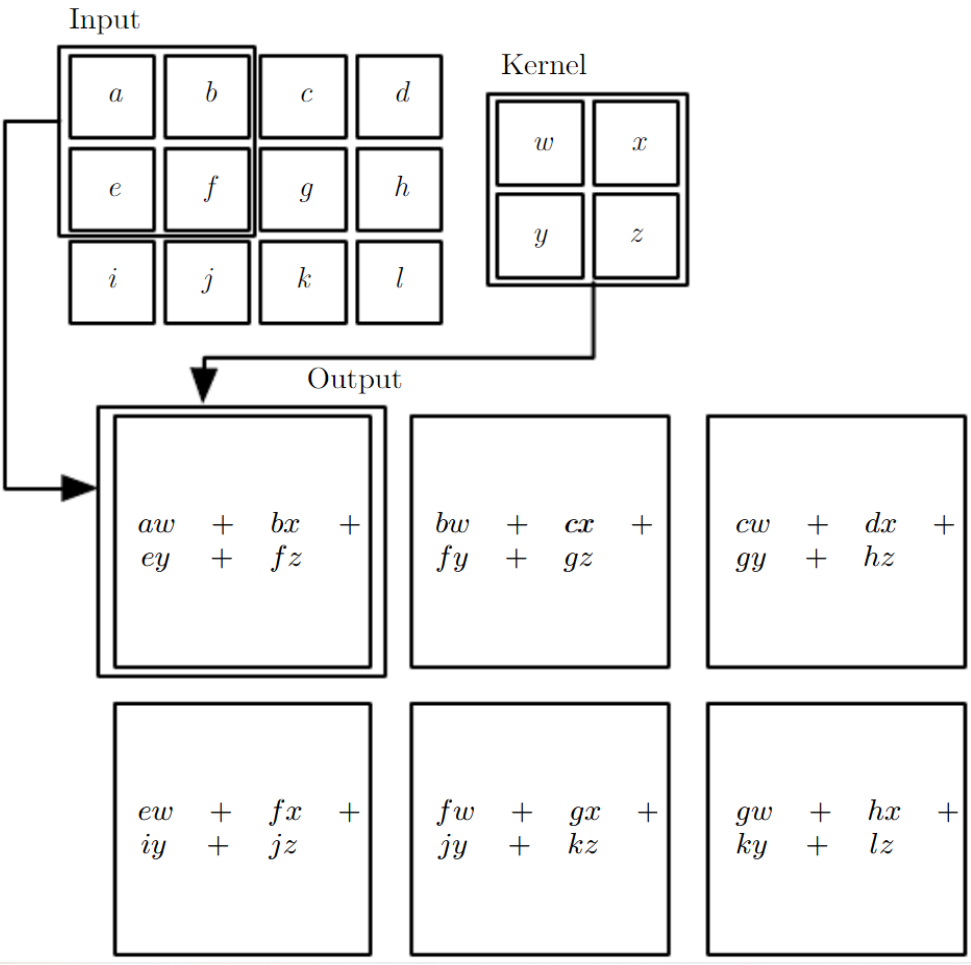


Figura 2.4: Ejemplo de una convolución con una entrada de dos dimensiones [2]

2.1.2. Capas de agrupación (pooling)

La segunda operación que típicamente conforma una red convolucional es la agrupación mejor conocida como pooling. Como se vio en la operación de convolución, la primer etapa de la red es aplicar una serie de kernels a una imagen de entrada lo cuál produce mapas de características, estos mapas pueden ser vistos como activaciones lineales que en una segunda etapa son pasados por funciones de activación no lineal tales como ReLU, de manera muy similar a las redes neuronales densas o tradicionales.
Luego de obtener el resultado de de las funciones de activación no lineal se utilizan estrategias de pooling que sintetizan las salidas mediante alguna técnica estadística.

Max pooling

Una de las técnicas más usadas de sintetizar o resumir es el max pooling [26] que obtiene el valor máximo de un vecindario rectangular de salidas como se muestra en la figura 2.5 donde se hace una operación de max pooling a un mapa de características sobre vecindades de 2x2.

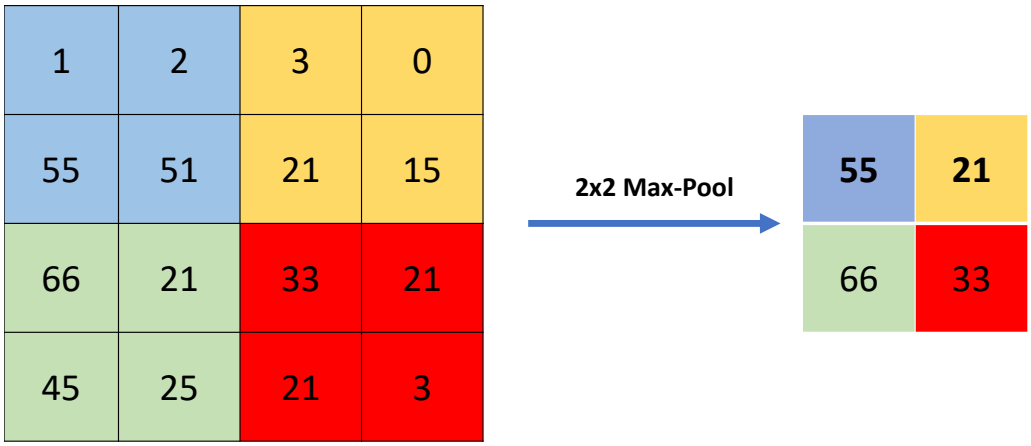


Figura 2.5: Ejemplo de una operación max pooling de 2x2.

Average pooling

Otra técnica para sintetizar o reducir el número de parámetros es el average pooling. Esta técnica es muy similar a la primera, con la diferencia de que de cada ventana o vecindario se obtiene el promedio de todos los vecinos y no el máximo valor como se muestra en la figura 2.6.

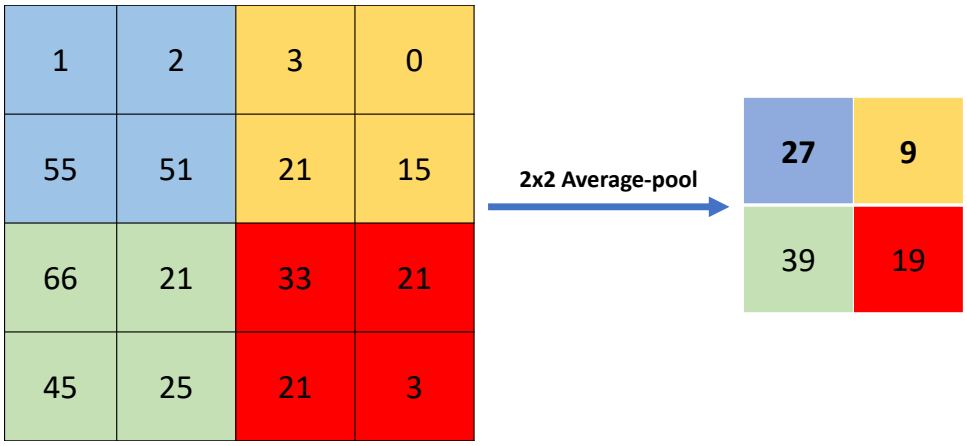


Figura 2.6: Ejemplo de una operación average pooling de 2x2.

2.1.3. Capas densas (fully connected)

La última etapa de una red neuronal convolucional corresponde a las capas densas o totalmente conectadas que a diferencia de las capas convolucionales se mencionó que éstas tenían conexiones

dispersas para obtener mapas de características. Estas capas están diseñadas para realizar la clasificación, una vez que se han extraído las características y dependiendo de el número de clases se pueden utilizar funciones de activación como softmax o sigmoid para clasificación binaria.

Sigmoid

Es una función de activación no lineal que mapea las entradas a una salida, también suelen llamarse funciones de transferencia. Esta función es una curva en forma de 's' cuyo dominio son todos los números reales y tiene un rango acotado en $[0, 1]$. Si el valor a evaluar en la función sigmoide es muy grande la salida será igual a 1, y si el valor es muy grande del lado negativo la salida será igual a 0. En (2.4) se muestra su definición y en la figura 2.7 se muestra la gráfica de la función.

$$f(z) = \frac{1}{1 + e^{-z}} \tag{2.4}$$

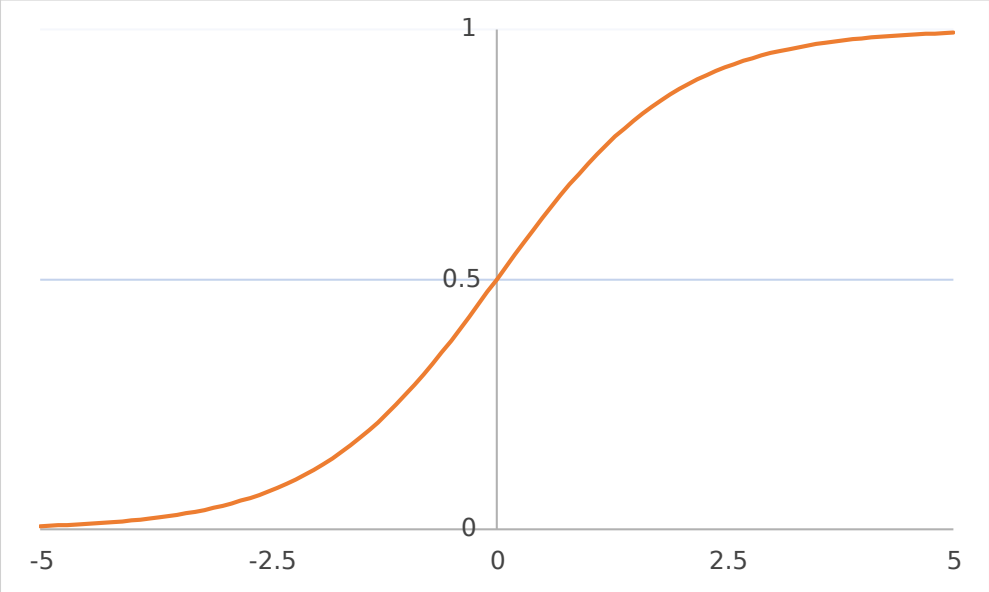


Figura 2.7: Gráfica de la función sigmoide

Finalmente la clasificación binaria se realiza de la siguiente manera: si la salida de la función sigmoide es mayor que 0.5 se clasifica como verdadero y si es menor a 0.5 como falso como se muestra en (2.5):

$$a(x) = \begin{cases} 1, & \text{si } f(z) \geq 0.5 \\ 0, & \text{si } f(z) < 0.5 \end{cases} \tag{2.5}$$

Softmax

Esta función se utiliza cuando se tienen problemas de clasificación múltiple, es decir, más de dos clases. Una capa de activación softmax tiene tantas neuronas como clases existen, y la función garantiza que la suma de todas las salidas de las nueronas sea igual a 1 ya que cada valor de las

salidas corresponde a la probabilidad de que la neurona o clase sea la correcta. En (2.6) se observa la definición de la función softmax y se puede intuir que es una generalización de la función sigmoide:

$$f(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{para } j = 1, ..., K. \tag{2.6}$$

2.1.4. Transferencia de aprendizaje (transfer learning)

El transfer learning es una técnica de aprendizaje automático donde un modelo previamente entrenado se utiliza en otra base de datos similar, es decir, si se tiene un modelo entrenado que clasifica bien entre perros y gatos, se puede utilizar ese modelo ya entrenado para clasificar entre tigres y lobos. Dependiendo de la similitud en las bases de datos se puede hacer un ajuste fino o un entrenamiento en las últimas capas del modelo, generalmente se hace un re-entrenamiento con los nuevos datos en las capas densas. Esta técnica es bien conocida por mejorar no sólo el tiempo de entrenamiento sino también el desempeño de la red en la nueva tarea [25].

El concepto de transfer learning puede ser definido como una situación donde lo que ha sido aprendido en una configuración, es aprovechado para mejorar la generalización en otra configuración. En el contexto del aprendizaje automático la generalización se refiere a la capacidad de un modelo de llevar a cabo una predicción de manera correcta en ejemplos que nunca ha visto con las mismas características que los usados para su entrenamiento [2].

2.2. Clasificación de las nubes

El Atlas Internacional de nubes ICA reconoce diez principales géneros de nubes, los cuales están definidos de acuerdo con su altura en el cielo, su forma y su apariencia. Las nubes altas típicamente tienen una altura de base por encima de 6000m; las nubes medias entre 2000m y 6000m y las nubes bajas están por debajo de los 2000m.

La mayoría de los nombres de las nubes contienen prefijos y sufijos que combinados ofrecen un indicador de las características de ese género como los siguientes:

- Stratus/strato: plano /en capas y suave
- Cumulus/cumulo: Acumulación / hinchado
- Cirrus/cirro: plumas, tenue
- Nimbus/nimbo: portador de lluvia

El cuadro 2.1 contiene los diez tipos de nubes según el ICA con abreviaciones y una descripción breve.

Tabla 2.1: Clasificación de nubes por su altura.

Nivel Alto	Nivel medio	Nivel bajo
Cirrus (Ci) Cirrocumulus (Cc) Cirrostratus (Cs)	Altocumulus (Ac) Altostratus (As) Nimbostratus (Ns)	Cumulus (Cu) Stratocumulus (Sc) Stratus (St) Cumulonimbus (Cb)
Usualmente son delgadas y blancas en apariencia, pero pueden aparecer en grandes arreglos de colores cuando el Sol es bajo en el horizonte.	Compuestas de gotitas de agua, pero también pueden componerse de cristales de hielo cuando la temperatura es lo suficientemente baja.	Nubes densas que van despegadas unas de otras. Pueden desprender lluvia, agujas de hielo o nieve.

Una descripción más detallada acerca de cada género de nube y una guía para identificar el tipo de nube se puede encontrar en [27]. Finalmente se muestra de manera gráfica los tipos de nubes ordenados por la altura donde se pueden encontrar 2.8:

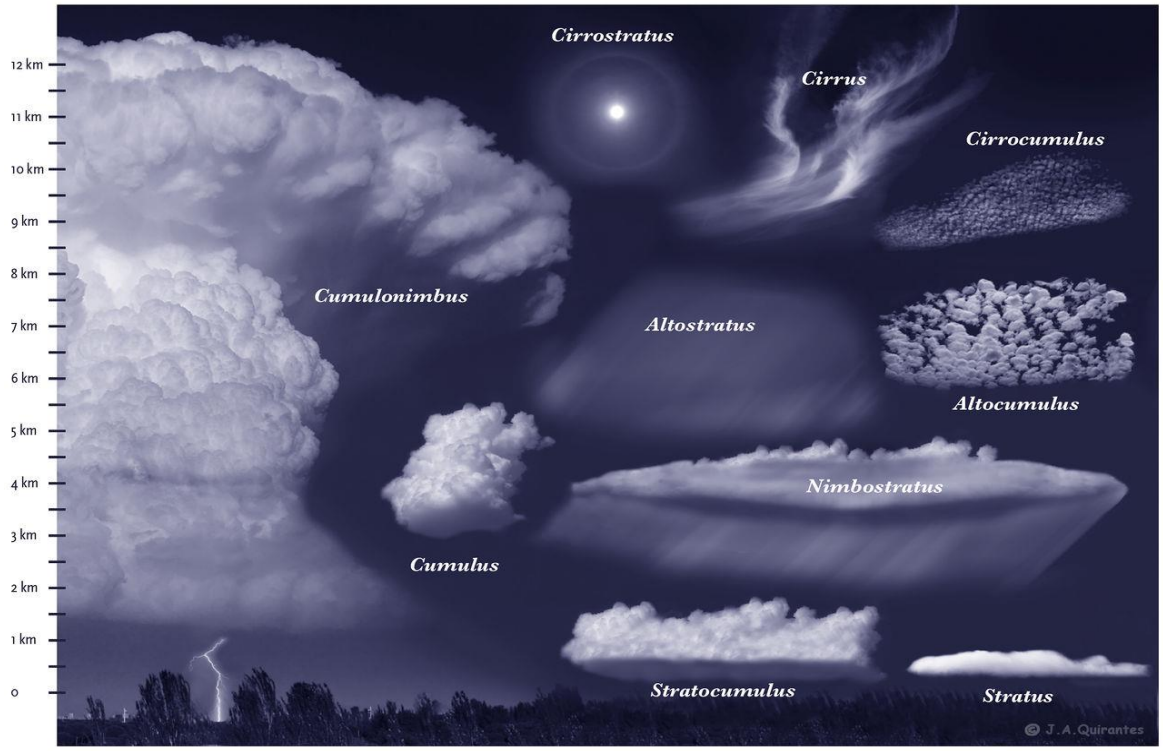


Figura 2.8: Gráfico de los diez tipos de nubes [3]

2.3. Optimización de hiper-parámetros de una CNN

Búsqueda en rejilla o búsqueda exhaustiva (*grid search*) y búsqueda aleatoria, son algunos de los métodos más populares para la optimización de hiper-parámetros. La búsqueda exhaustiva como su nombre lo indica consiste en probar todas las combinaciones posibles de hiper-parámetros para

garantizar que se seleccione la arquitectura que mejor rendimiento produce. Conforme el número de hiper-parámetros aumenta, el espacio de búsqueda crece exponencialmente y la cantidad necesaria de tiempo para explorar todas las combinaciones es tan grande que hace imposible tomar este camino.
Por otro lado la búsqueda aleatoria necesita menos tiempo que la búsqueda exhaustiva, porque a diferencia de este método este no agota todas las combinaciones posibles, sino que selecciona algunas combinaciones de hiper-parámetros de forma aleatoria. Este procedimiento no garantiza que se encuentre la combinación óptima de hiper-parámetros, aunque los experimentos reportados en la literatura han probado que bajo ciertas condiciones la búsqueda aleatoria puede alcanzar resultados similares a la búsqueda exhaustiva siendo un método más eficiente.

El problema de optimización de hiper-parámetros puede ser formulado en términos de una función de costo $f(x)$. Dicha función es usada para entrenar un modelo de red neuronal convolucional M.
Considere un espacio de búsqueda n-dimensional S_{hparam} denotado como como sigue:

$$S_{hparam} = [s_1, s_2, s_3, ..., s_p] \tag{2.7}$$

donde cada miembro del vector corresponde a un conjunto solución con n hiper-parámetros, es decir:

$$s_i = [h_1, h_2, h_3, ..., h_n] \tag{2.8}$$

donde h_i denota un conjunto de posibles valores para el i-ésimo hiper-parámetro.
Si se considera que existen n categorías de hiper-parámetros, los cuales pueden tener m posibles valores, el problema se resume a encontrar de manera rápida los valores óptimos para los hiper-parámetros de M.

2.4. Problemas NP-Complejos

Con frecuencia los problemas científicos o en ingeniería pueden ser reducidos a obtener la solución de un sistema de ecuaciones y desigualdades lineales. En otras ocasiones, la solución implica la extracción de ciertos valores como los eigenvectores o valores singulares de una matriz. Por otro lado en el campo de la visión por computadora, por ejemplo, la segmentación de una imagen digital a lo largo de su contorno puede ser encontrada calculando los primeros eigenvectores de una cierta matriz asociada a la imagen [28].
Se dice que un problema es NP-completo si puede ser resuelto por medio de busqueda exhaustiva. Desafortunadamente cuando el tamaño de las instancias crece, el tiempo de ejecución de la búsqueda exhaustiva se vuelve extremadamente grande, incluso para problemas donde las instancias son relativamente pocas pueden provocar que las combinaciones posibles sean imposibles de probar debido al tiempo que llevaría.
Según el teorema de *Cook-Levin*, si un problema particular NP-completo puede ser resuelto con un algoritmo en tiempo polinómico mediante una máquina de Turing determinista, entonces, cada problema NP puede resolverse mediante un algoritmo de tiempo polinomial deteterminista. Dicho lo anterior un problema se convierte en NP-complejo si todos los problemas NP pueden reducirse en tiempo polionomial a dicho NP-complejo, dicho de otro modo, si H es un problema NP-complejo entonces cualquier problema L que sea NP se podrá reducir a H. Entonces, un problema NP-complejo es al menos tan difícil como cualquier problema NP-completo y posiblemente mucho más difícil.

2.4.1. Optimización de hiper-parámetros como un problema NP-complejo

Las redes neuronales convolucionales han demostrado ser una de las mejores técnicas de clasificación de imágenes y en general tienen buen desempeño en el área de la visión por computadora. Como se ha mostrado en parrafos anteriores la arquitectura de una red neuronal convolucional esta formada por distintas capas entre las cuales se pueden encontrar: capas convolucionales, capas densas, *max-pooling*, *average-pooling*, *batch normalization*, entre otras capas que se han ido desarrollando a través del tiempo como las capas de salto, etc,. Todo esto implica una serie de hiper-parámetros que pueden ser modificados bajo el criterio del diseñador o en base a la aplicación que se este trabajando, estos hiper-parámetros van desde el número de capas convolucionales dentro de la red, el tamaño y número de filtros dentro de esas capas, ¿se contará con capas de normalización?, si es así, ¿cuál será el porcentaje?, ¿que función de activación se utilizara? y así sucesivamente. Esto implica que existen múltiples combinaciones para el diseño de una arquitectur, tradicionalmente, la selección de hiper-parámetros se lleva a cabo a prueba y error, es decir, manualmente probar e ir cambiando diferentes conjuntos de hiper-parámetros que se intuye (basado en la experiencia o en los reportes de publicaciones) funcionarán bien. Esto requiere cierto nivel de conocimiento y experiencia en el área del aprendizaje profundo y además el proceso puede ser tedioso y dependiendo del número de hiper-parámetros que se estén optimizando puede llevar demasiado tiempo probar distintas combinaciones.

2.5. Algoritmos Metaheurísticos

2.5.1. Algoritmo Genético

Acorde con Forrest en [29] Un algoritmo genético es un modelo computacional de la evolución biológica, los cuales son útiles como métodos de búsqueda o para resolver y modelar sistemas evolutivos. Para el caso de esta tesis, principalmente se utilizará este algoritmo como método de búsqueda, aunque no podemos dejar de lado la perspectiva de que el entrenamiento de una red neuronal convolucional también podría modelarse como un sistema evolutivo.
Los algoritmos genéticos están biológicamente inspirados. Se trata de imitar la teoría de la evolución de Darwin. En la teoría de la selección natural de Darwin, los individuos más aptos son seleccionados y estos producen descendientes los cuales heredan características de las generaciones anteriores pero pueden mejorarlas para ser aún más aptos y tener más posibilidades de sobrevivir a las próximas generaciones. Un algoritmo genético, contiene cadenas de datos binarios los cuales se almacenan en la memoria de la computadora y a través del tiempo y de ciertas operaciones bit a bit como la mutación son modificadas de manera analoga a las poblaciones de individuos que evolucionan acorde a la selección natural.
Como se muestra en el diagrama de la figura 3.3 el algoritmo comienza creando una **población de individuos de forma aleatoria**, para el caso más simple, un individuo puede ser una cadena de bits el cual representa un candidato a ser la solución para un determinado problema.

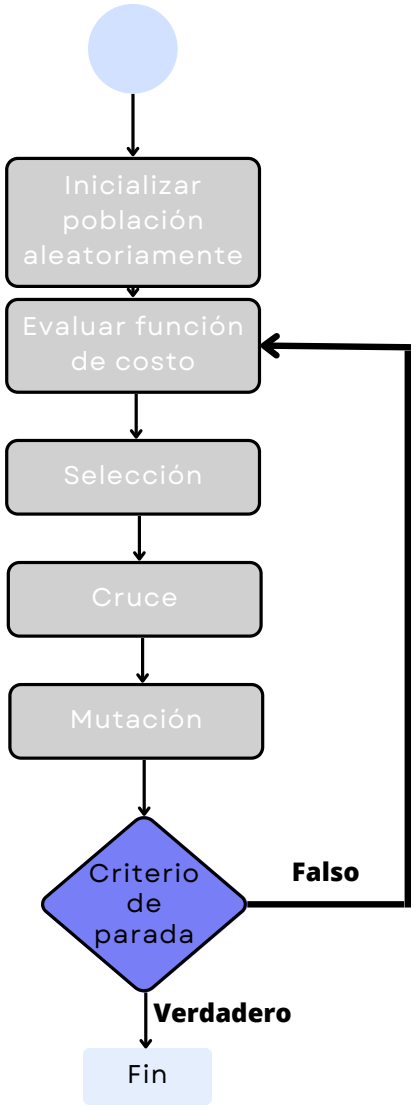


Figura 2.9: Diagrama de flujo general para un algoritmo genético

La población ha sido inicializada aleatoriamente (dentro de un espacio o rango acorde al problema de interés) para que las diferencias entre los individuos resulten en unos individuos mejores que otros, lo cual se comprueba en en el siguiente paso en la **evaluación de la función de costo** o mejor conocido dentro de la literatura *fitness*.
Una vez evaluados los individuos se realiza una **selección** de aquellos con el mejor fitness, es decir, aquellos que dentro de la población se ajustan más a una posible solución, estos sobreviven a la siguiente generación mientras que los menos exitosos se eliminan.
Para la nueva generación además de los individuos con mejor fitness de la generación pasada se sustitutyen los individuos eliminados por nuevos individuos con diferente información. Estos nue-

vos individuos no son formados aleatoriamente sino que son creados a partir de las operaciones de **cruce** (*crossover*) y de **mutación**, la primera consiste en tomar dos de los individuos con mejor

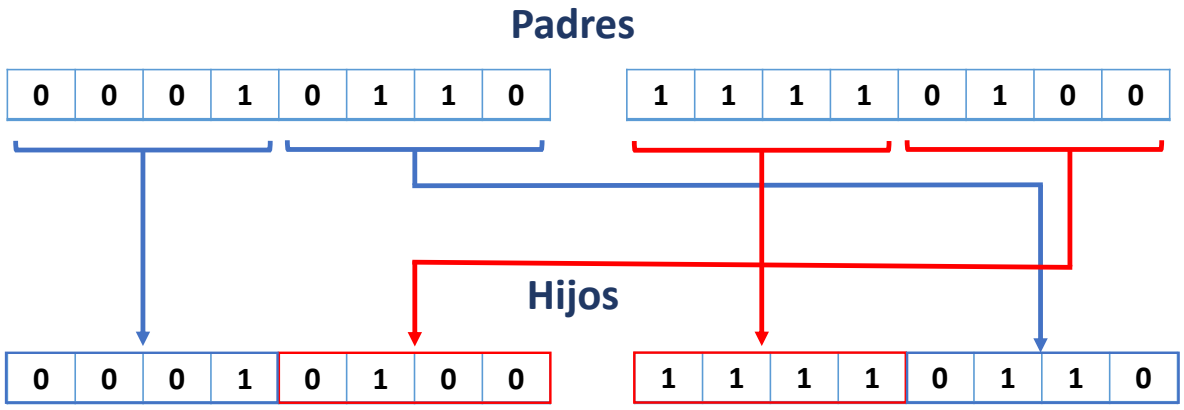


Figura 2.10: Funcionamiento de la operación genética cruce o crossover.

Mientras que la mutación consiste en alterar aleatoriamente alguno de los bits o datos del

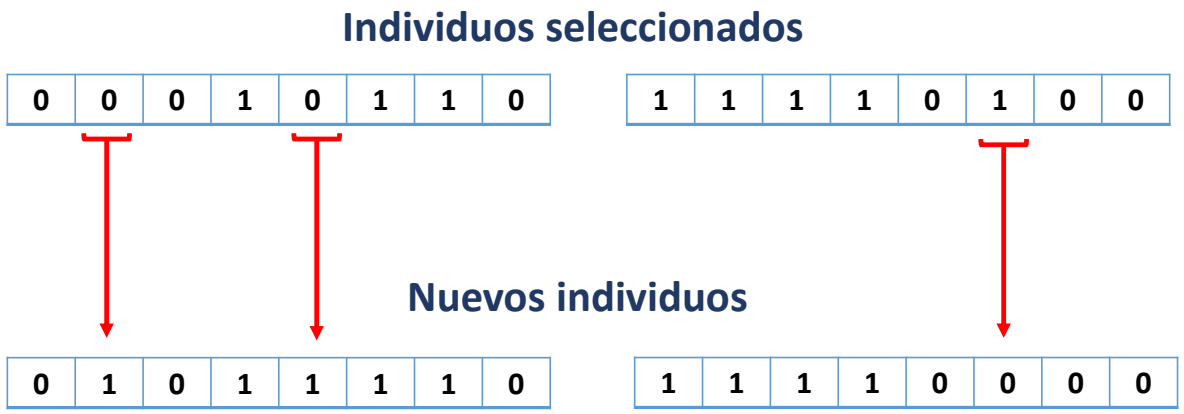


Figura 2.11: Funcionamiento de la operación genética mutación.

CAPÍTULO 3

Metodología

En esta sección se presentan los métodos, estrategias y recursos necesarios para llevar a cabo la investigación. También se mencionan de manera detallada los recursos humanos y materiales necesarios para la conclusión del trabajo.

En la Figura 3.1 se ilustra la metodología utilizada en esta investigación.

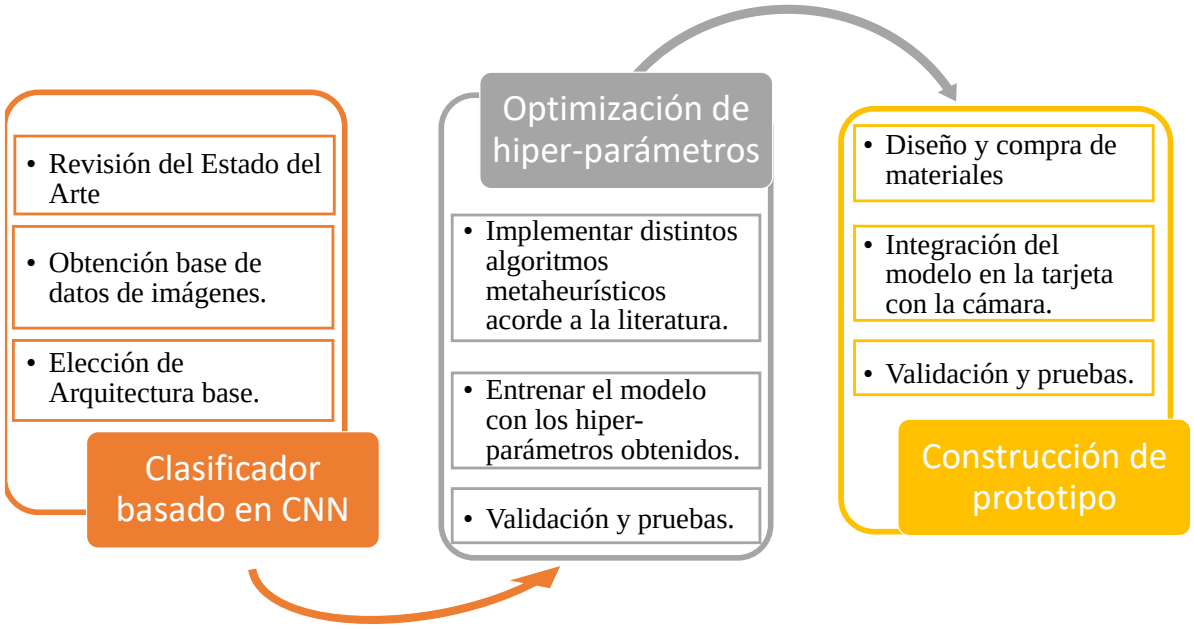


Figura 3.1: Diagrama de metodología de la investigación

3.1. Clasificador basado en CNN

El primer paso consistió en hacer una revisión de la literatura adecuada, utilizando motores de búsqueda como SCOPUS y Google Académico. Algunas palabras clave para esta búsqueda fue-

ron: *ground-based cloud*, *cloud classification*, *deep learning classification*. Debido al enfoque de este trabajo se excluye material relacionado con clasificación en imágenes de satélite. Este paso nos permitió encontrar la tendencia de las técnicas que han mostrado mejores resultados en cuanto a la clasificación de nubes y plantear una mejora a partir de lo que se ha hecho.

3.1.1. Colección de imágenes

El segundo paso fue la obtención de la base de datos necesaria para probar el algoritmo propuesto. Dentro del marco de algoritmos de inteligencia artificial como las redes neuronales, se dice que estos algoritmos son tan buenos como los datos con los que se alimenta. Bajo esta premisa y con la información obtenida en la revisión del estado del arte, se eligió la base de datos MultimodalGround-based Cloud Dataset (MGCD) [30] la cual contiene 8000 imágenes de 7 clases diferentes siguiendo la clasificación hecha por la WMO, según se muestra en la tabla 3.1. Se eligió esta base de datos ya que ha sido utilizada por varios de los artículos más relevantes de los últimos años y nos permite hacer una comparación más justa de los algoritmos utilizados con el propuesto en este trabajo. Además como se muestra en la figura 1.2 las muestras contenidas en esta base de datos son imágenes de cielo completo, es decir, fueron tomadas con un instrumento capaz de capturar el cielo de horizonte a horizonte.

Tabla 3.1: Distribución y mapeo de las categorías dentro de la base de datos MGCD

Etiqueta	**Categoría**	**Muestras**
1	Cumulus	1438
2	Altocumulus	731
3	Cirrus	1323
4	Cielo despejado	1338
5	Stratocumulus	963
6	Cumulonimbus	1187
7	Mixto	1020

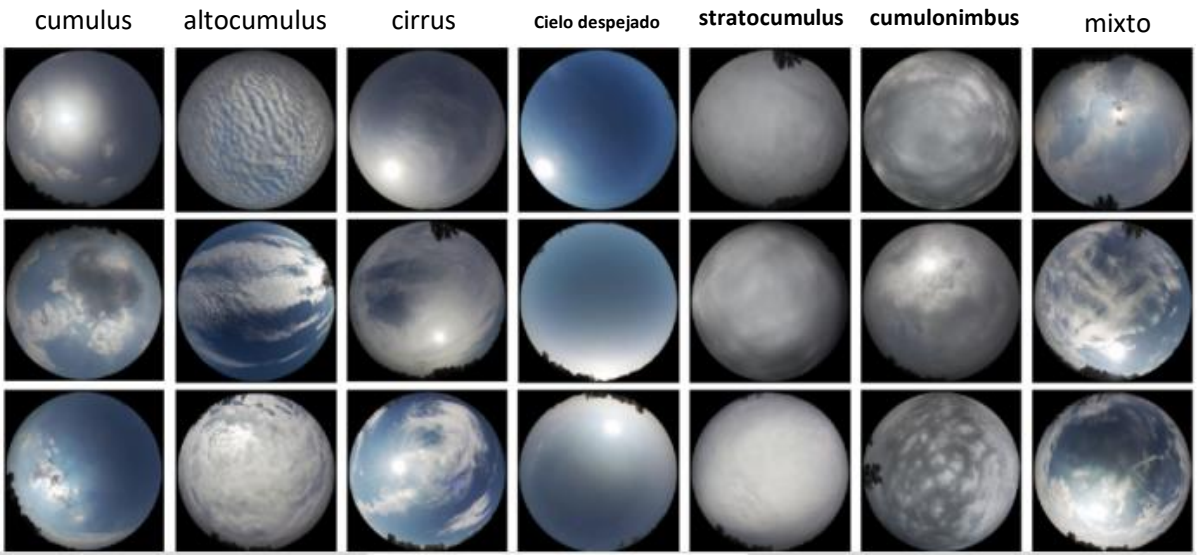


Figura 3.2: Ejemplos de las diferentes categorías en la base de datos MGCD

3.1.2. Elección de arquitectura base

La etapa del desarrollo del algoritmo es una de las más importantes y donde se llevó a cabo gran parte de la experimentación. En la sección de antecedentes se estableció que la tendencia en clasificación de imágenes de nubes tomadas desde tierra está inclinada hacia algoritmos de aprendizaje profundo como lo son las CNNs.

Lo anterior nos motivó a usar la metodología descrita en 3.1, donde se propuso comenzar con una arquitectura probada y partir de ese punto para realizar una mejora en términos de precisión, aunado a la restricción de memoria y cómputo que representa el implementarlo en un sistema embebido.

De esta manera, y similarmente a trabajos como los reportados en [23] y [12], se realizó un diseño de experimento con los siguientes pasos:

- Elegir alguna arquitectura de CNN que haya reportado buenos resultados en la literatura.
- Estudiar la arquitectura y tratar de replicarla o de ser posible conseguir el modelo entrenado para usar *transfer learning*.

Los modelos propuestos para comenzar a hacer pruebas como clasificadores fueron *Efficient Net* [31] y *Mobile Net* [32], este último ha sido diseñado para ser propuesto como un modelo que puede ser utilizado en dispositivos con mayores restricciones de memoria y poder de cómputo como lo son los móviles o sistemas embebidos. Por otra parte *EfficientNet* esta basado en la idea de ir escalando la arquitectura de la red de una manera eficiente acorde a los recursos disponibles, calculando la eficiencia a manera de lograr mayores porcentajes de precisión con la arquitectura más ligera posible.
A continuación en la tabla 3.2se presenta una comparación con algunas de las características como número de capas y parámetros y además la precisión lograda en un primer experimento entrenando

ambas con la base de datos MGDC y entrenando por el mismo número de épocas. Cabe resaltar que en este experimento no se hace enfásis en el número de épocas ni otros hiperparámetros ya que al ser los mismos y solamente se busca hacer una comparación entre ambas redes, por lo tanto es información irrelevante.

Tabla 3.2: Característcas de los modelos propuestos

	Efficient Net		**Mobile Net**	
Parámetros	Entrenables	Total	Entrenables	Total
	6,297,607	70,395,294	3,676,167	5,934,151
# de capas	813		53	
Tamaño en MB	326.31		52.39	
Precisión	85 %		82.3 %	

3.2. Optimización de hiper-parámetros

En esta etapa se implementó un algoritmo metaherístico como técnica inovadora para clasificación de imágenes de nubes, con el objetivo principal de optimizar los hiper-parámetros de una red neuronal convolucional que funciona como algoritmo de extracción de características y clasificador. El algoritmo metahehuristico es un algoritmo genético que fue propuesto en 2020 por Xiao *et al.* y en el diagrama de la figura 3.3 se muestra la manera en como funciona. A continuación se describen los pasos que explican el diagrama.

1. Inicializar el tamaño de la población p y el número de generaciones g de manera arbitraria.

2. Crear la población inicial. Cada uno de los individuos en la población contiene hiper-parámetros aleatorios. El tamaño de la población es p, según lo especifique el usuario.

3. Evaluar el fitness de cada individuo en la generación actual. El fitness corresponde a la precisión del individuo sobre el dataset de validación.

4. Ordenar cada individuo de acuerdo a su valor de precisión, del mejor al peor.

5. Seleccionar los mejores individuos. Seleccione una porción de la población que contenga los mejores fitness y dejelos sobrevivir a la siguiente generación. El porcentaje de supervivencia puede ser elegido manualmente.

6. Permita que algunos individuos con menor precisión sobrevivan también. Esta probabilidad también se configura manualmente.

7. Aleatoriamente mutar algunos de los individuos en la seguiente generación. La tasa de mutación es configurada manualmente. Si un individuo es elegido para ser mutado, alguno de sus hiper-parámetros sera cambiado.

8. Producir nuevos individuos. Aleatoriamente elija dos individuos de la siguiente generación para que funcionen como padres, y ejecute la operación de crossover para producir un hijo. Cada hiper-parámetro en el hijo es elegido aleatoriamente de los hiper-parámetros de sus padres. Repita la operación de crossover tantas veces hasta que la generación tenga p individuos.

9. Repetir los pasos 3-8 hasta que el numero de generaciones g se alcance.

10. Seleccione al mejor individuo de la población actual.

11. Producir una nueva población con un tamaño de cromosomas mayor. Para cada individuo de la población, una parte de su cromosoma sera el mejor individuo del paso 10, la otra parte se genera aleatoriamente.

12. Repetir los pasos 9-11 hasta la convergencia.

13. Seleccionar el mejor individuo y entrenar por más épocas hasta la convergencia.

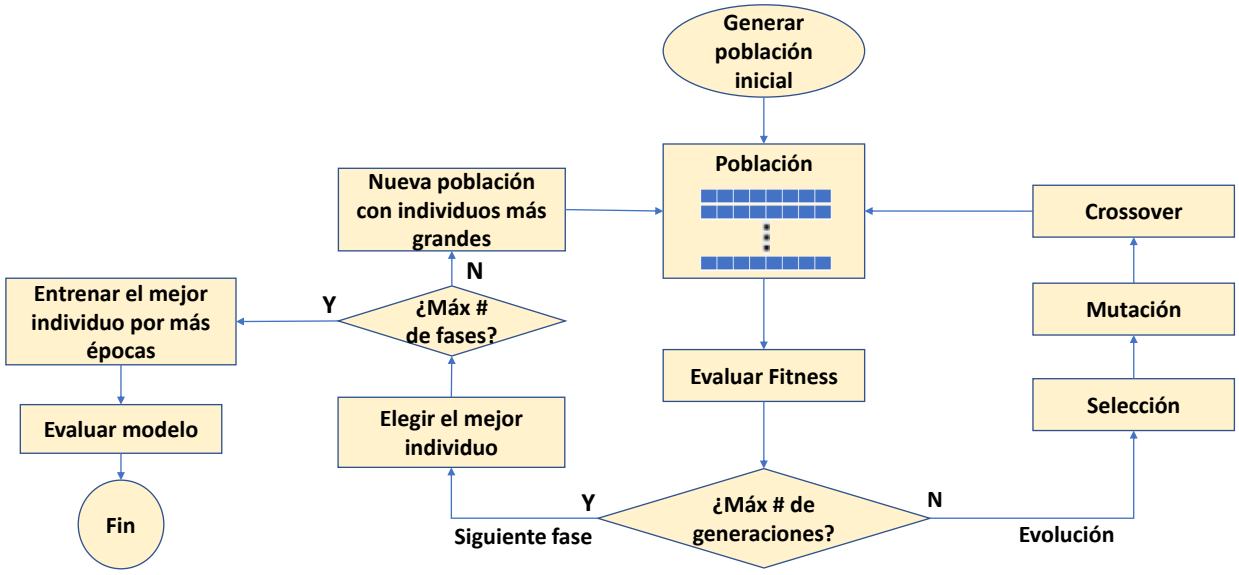


Figura 3.3: Diagrama de flujo del funcionamiento del algoritmo genético propuesto en [4]

A continuación en el pseudocódigo del algoritmo 1, se muestra el funcionamiento del algoritmo genético en términos de instrucciones de programación, pero sin ser código realmente. Se omiten detalles e instrucciones que de forma general sobran para explicar el funcionamiento del algoritmo.

Algoritmo 1 Pseudocódigo del algoritmo genético utilizado

Entrada: fases, población, generaciones, dataset
Salida: modelo a entrenar
Procedimiento:

```
Generar población inicial aleatoriamente
Fases (f) ← 0
    while (f < f_máx) do
    Generaciones (g) ← 0
        while (g < g_máx) do
            Entrenar los individuos por 5 épocas
            Calcular el valor fitness de cada individuo
            Seleccionar mejores individuos
            Aplicar crossover y mutación
        g ← g + 1
        endwhile
        f ← f + 1
    endwhile
```

Fin

3.2.1. Configuración del experimento

En la sección de marco teórico se ha explicado de forma general como funciona un algoritmo genético, en esta sección se explicará el paso a paso de como se llevó a cabo la puesta en marcha del algoritmo genético siguiendo la metodología en [4], expuesta anteriormente.

Los experimentos fueron llevados a cabo utilizando el lenguaje de programación Python, y los modelos de redes neuronales convolucionales utilizando la librería de Keras. Debido a la falta de recursos se optó por utilizar un cuaderno de *Google Colab* en su versión de paga, las características del equipo rentado así como la versión de keras y tensorflow se muestran en la tabla 3.3.

Tabla 3.3: Características del equipo de cómputo donde se llevó a cabo la experimentación

GPU	NVIDIA A100-SXM4	
	Disponible	***Utilizado***
RAM	83.5 GB	67.5 GB
RAM-GPU	40 GB	33.3 GB
Disco duro	166.8 GB	27.8 GB
	Versión	
Tensorflow	2.11.0	
Keras	2.11.0	
Python	3.8	

Partición del dataset para los experimentos

Como se muestra en la sección de los resultados no se utilizó el dataset completo a excepción de los últimos dos modelos, esto debido principlamente a las restricciones de poder de cómputo durante los experimentos, sin embargo para cada uno de los experimentos se mantuvo constante la partición para el dataset de entrenamiento y de evaluación siguiendo la norma de 80-20, es decir, 80 % del total de las imágenes para el entrenamiento y el restante 20 % para la evaluación. La principal razón de no utilizar el total de las imágenes para entrenar, es que esto nos llevaría a un sobreajuste. El sobreajuste se presenta cuando el modelo obtiene un alto porcentaje de clasificación sobre las imágenes de entrenamiento pero cuando se presentan imágenes que nunca ha visto el rendimiento está lejos del porcentaje alcanzado durante el entrenamiento. Es decir, el modelo simplemente aprendió las características de las muestras pero no hizo una abstracción de estas características, al hecho de abstraer las características y poder reconocerlas en imágenes nuevas se le conoce como generalización.
La regla de partición de 80-20 es una práctica que surgió debido al flujo de trabajo que ocurre generalmente cuando se quiere ajustar un modelo de aprendizaje de máquina o aprendizaje profundo. Dado que normalmente se hacen modificaciones de hiper-parámetros y se vuelve a probar lo que se está haciendo es ajustando conforme al dataset de entrenamiento y la partición sobrante nos da una mejor idea del funcionamiento real del modelo. De modo general la regla es obtener 3 sub conjuntos del total de los datos, el mayor para entrenamiento uno más para hacer validación de cada cambio que se este efectuando en los hiper-parámetros y finalmente el de prueba que nos da una métrica del rendimiento del modelo sin sesgo, es decir, que tan bien generalizó ya el modelo. Aunque esta última regla indica la división del dataset en un 60 % - 20 % - 20 %, en muchos trabajos se suele omitir la partición de prueba, quedando en 80 % - 20 %.

3.2.2. Espacio de búsqueda

Acorde al diagrama de la figura 3.3 el segundo paso corresponde a la creación de los individuos de manera aleatoria cuando se esta corriendo por primera vez el algoritmo. Estos individuos corresponden a un diccionario que contiene los hiper-parámetros de la red que se van a optimizar con su respectivo valor, los cuales forman un conjunto solución para obtener el mejor desempeño en el porcentaje de clasificación.

Esta asignación aleatoria se hace dentro de un espacio de búsqueda definido por el diseñador, donde a priori se eligen los hiper-parámetros que se desean optimizar, así como los posibles valores de entre los cuales se podrá escoger. A continuación en la tabla 3.4 se muestran los hiper-parámetros y los posibles valores que definen el espacio de búsqueda para este trabajo.

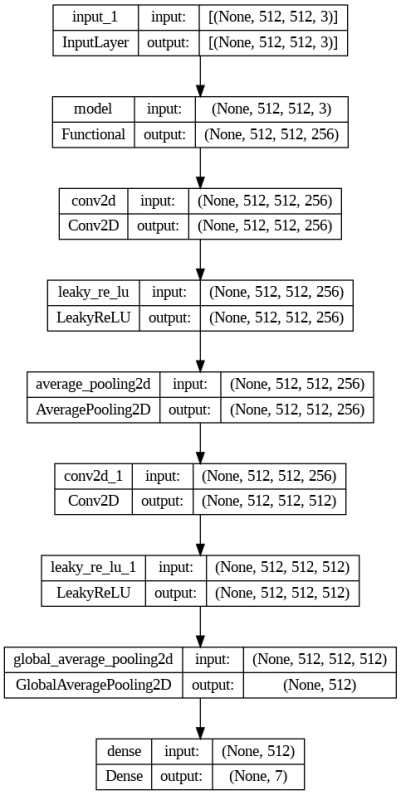
Tabla 3.4: Espacio de búsqueda utilizado por el algoritmo genético

Hiper-parámetro	**Opciones**
Tamaño del filtro	(1,1) , (3,3) , (5,5) , (7,7) , (9,9)
Batch normalization	Sí, No
Número de filtros	8, 16, 32, 64, 128, 256, 512
Función de activación	ReLU, ELU, LeakyReLU
Pooling	Sí, No
Tipo de pooling	maxpooling, averagepooling
Conexión salto	Sí, No

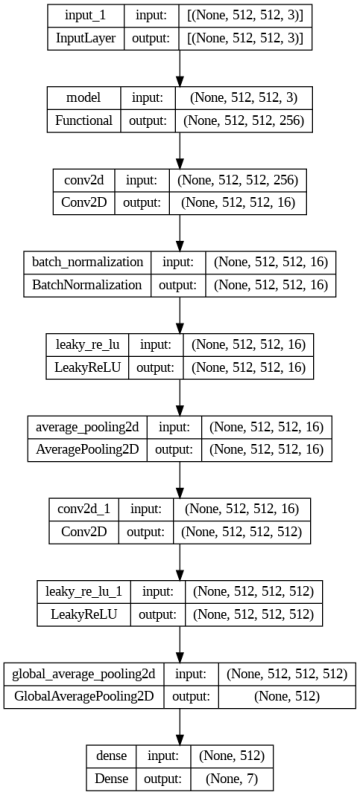
Como se observa en la tabla 3.4 algunos hiper-parámetros como el *batch normalization* y el *pooling* se mantuvieron en inglés dado que sus traducciones son poco conocidas dentro de la literatura en español y comúnmente se han adoptado los nombres en inglés, de mismo modo pasa con las funciones de activación y sus abreviaturas. Cabe mencionar que los hiper-parámetros y las opciones no son limitativas, y se puede ampliar o disminuir el espacio de búsqueda según el criterio del diseñador, mencionando solamente que un espacio muy pequeño puede restringir las posibilidades del algoritmo al tener menos opciones, aunque disminuiría el tiempo de búsqueda, y por el contrario un espacio de búsqueda muy grande ofrece mayor posibilidad de combinaciones para la solución pero exige mayor tiempo de cómputo así como recursos de memoria y procesamiento.

3.2.3. Modelos obtenidos con el algoritmo genético

En la subsección anterior se mostró el funcionamiento del algoritmo genético utilizando un diagrama de flujo y también el pseudo-código que muestra la forma de ejecutar dicho algoritmo. En esta subsección se muestran las arquitecturas (3.4 3.5) de los modelos obtenidos a fin de que se puedan repetir los modelos ya optimizados y sean utilizados para alguna aplicación fuera de esta tesis.
A pesar de que estas arquitecturas son los resultados obtenidos de la optimización se ha decidido ponerlos en esta sección de modo que la sección de resultados este completamente enfocada a los resultados de los entrenamientos de dichos modelos.

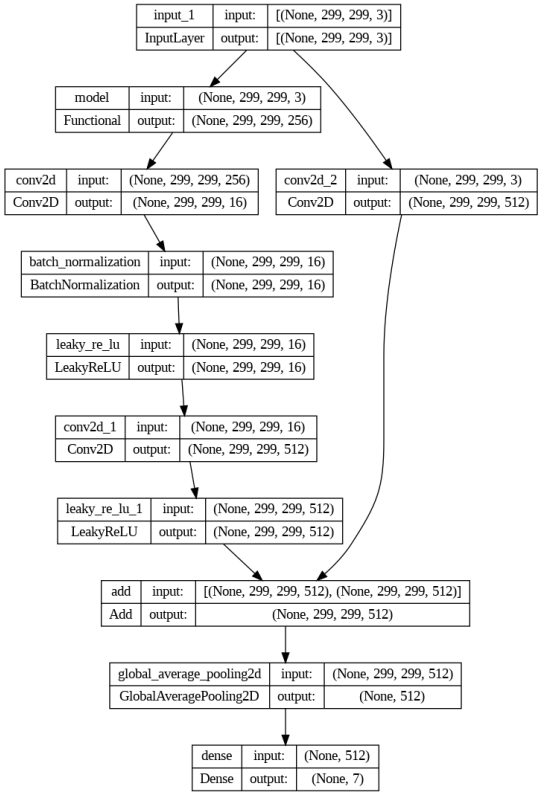


(a) Modelo 1

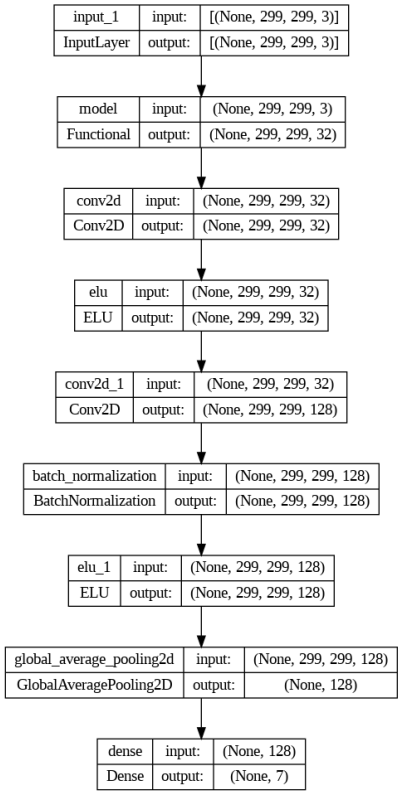


(b) Modelo 2

Figura 3.4: Arquitectura generada por el algoritmo genético para los modelos 1 y 2.



(a) Modelo 3



(b) Modelo 4

Figura 3.5: Arquitectura generada por el algoritmo genético para los modelos 3 y 4.

Arquitectura desglosada por capas de los modelos

Tabla 3.5: Desglose de capas del modelo 2.

Capa	Operación	Kernel	Padding	Salida
1	Conv2D → 32 Función A. → LeakyReLU Batch normal.	(7,7)	'same'	(512,512,32)
2	Avg. pooling	(2,2)	'same'	(512,512,32)
3	Conv2D→ 8 Función A. → LeakyReLU	(9,9)	'same'	(512,512,8)
4	Conv2D→ 256 Función A. → LeakyReLU Batch normal.	(1,1)	'same'	(512,512,256)
5	Máx. pooling	(2,2)	'same'	(512,512,256)
6	Conv2D→ 256 Función A. → LeakyReLU Batch normal.	(9,9)	'same'	(512,512,256)
7	Conv2D→ 16 Función A. → LeakyReLU Batch normal.	(1,1)	'same'	(512,512,16)
8	Avg. pooling	(2,2)	'same'	(512,512,16)
9	Conv2D→ 512 Función A. → LeakyReLU	(9,9)	'same'	(512,512,512)
10	Global average pooling Dense → 7 Función A. → Softmax			(,7)

Tabla 3.6: Desglose de capas del modelo 3.

Capa	Operación	Kernel	Padding	Salida
1	Conv2D->16 Función A. ->LeakyReLU Batch normal.	(7,7)	'same'	(512,512,16)
2	Avg. pooling	(2,2)	'same'	(512,512,32)
3	Conv2D->512 Función A. ->LeakyReLU Batch normal.	(5,5)	'same'	(512,512,512)
4	Conv2D->64 Función A. ->LeakyReLU	(1,1)	'same'	(512,512,64)
5	Máx. pooling	(2,2)	'same'	(512,512,64)
6	Conv2D->512 Función A. ->LeakyReLU Batch normal.	(3,3)	'same'	(512,512,512)
7	Conv2D->128 Función A. ->LeakyReLU Batch normal.	(5,5)	'same'	(512,512,128)
8	Máx. pooling	(2,2)	'same'	(512,512,128)
9	Conv2D->256 Función A. ->LeakyReLU	(7,7)	'same'	(512,512,256)
10	Conv2D->16 Función A. ->LeakyReLU Batch normal.	(9,9)	'same'	(512,512,16)
11	Máx. pooling	(2,2)	'same'	(512,512,16)
12	Conv2D->512 Función A. ->LeakyReLU	(3,3)	'same'	(512,512,512)
13	Global average pooling Dense ->7 Función A. ->Softmáx			(,7)

Tabla 3.7: Hiper-parámetros del algoritmo genético para los diferentes modelos.

Modelo	Tamaño de población	# de fases	# de generaciones	Tasa de supervivencia	Tasa de mutación
1	4	2	3	0.5	0.1
2	8	3	4	0.5	0.1
3	8	4	4	0.5	0.1
4	8	3	8	0.5	0.1

Acorde a la tabla 3.7 se ejecutó el algoritmo genético con los hiper-parámetros mostrados. Cabe mencionar que se eligieron el tamaño de la población, el número de fases y de generaciones de modo que se pudieran probar distintas combinaciones pero con la restricción de la memoria y la demanda de cómputo que estos podían generar debiso a los recursos limitados. Sin embargo dichos

valores pueden ser incrementados de modo que se logre una mejor optimización de la red neuronal convolucional.

3.3. Construcción del prototipo

Acorde a los objetivos planteados en esta investigación, el último de ellos consiste en construir un prototipo que contenga el clasificador obtenido en el objetivo anterior para clasificar imágenes tomadas en tiempo real. Por definición este podría no ser un objetivo de la investigación, debido a que no contribuye a demostrar o refutar la hipótesis, sin embargo se ha añadió como objetivo alcanzable una vez que se ha demostrado la hipótesis, y sería interesante probar alguno de los modelos para clasificar imágenes en tiempo real y que pueda funcionar como un instrumento que mide una de las variables que se pueden extraer de las nubes.

3.3.1. Equipo y materiales

En la tabla 3.8 se muestran los materiales utilizados para la construcción del prototipo...

Tabla 3.8: Recursos materiales para el prototipo.

Cantidad	**Equipo / Materiales**	**Costo ($ MXN)**
1	Raspeberry Pi 3 Model-B with 1GB RAM RAM	2,600
1	Cámara de alta calidad raspberry Pi high quality (con base para lente intercambiable)	1,300
1	Lente gran angular 6mm para cámara de alta calidad raspberry Pi	800
1	Domo de protección para cámara	300
1	Útiles electronicos (LDR, push botton, resistencias LEDs, etc)	200
1	Cable alimentación USB tipo C 15.3W	250
1	Memoria microSD de 64 GB	200
1	Cable de red UTP CAT 6	
	Total	6,250

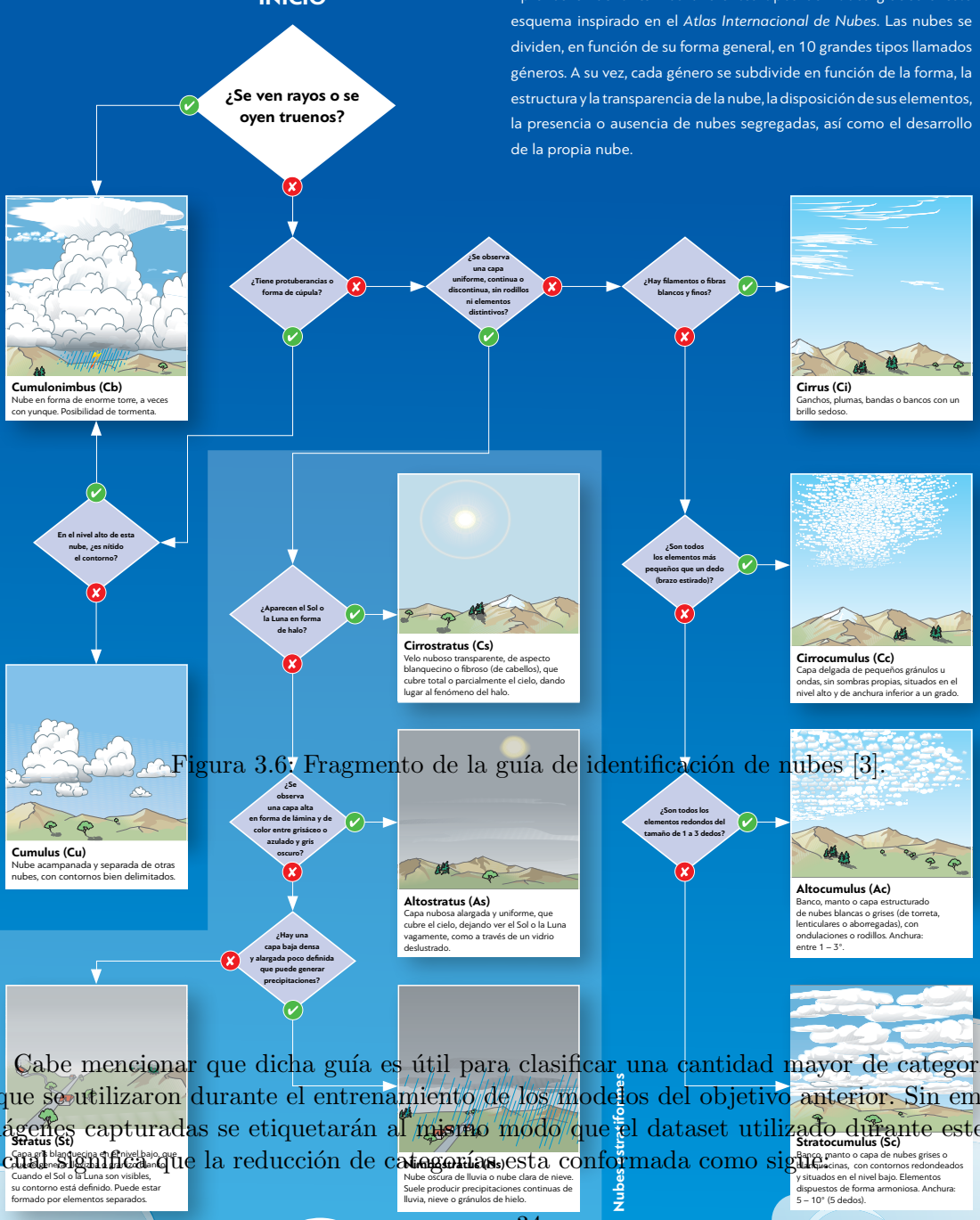
3.3.2. Etiquetado de las imágenes capturadas

Una vez que se ha reunido un conjunto de imágenes es necesario asignarles una etiqueta real para contrastar con la predicción del clasificador, para llevar a cabo esta tarea se hizo uso de la guía de identificación de nubes que ofrece el atlas internacional de nubes desde el sitio web de la WMO. A continuación en la figura 3.6 se muestra un fragmento extraído de dica guía que se puede encontrar en [3] y que indica en modo de diagrama de flujo las opciones para ir descartando y continuando de modo que se llega a una resolución.

ORGANIZACIÓN METEOROLÓGICA MUNDIAL

TIEMPO CLIMA AGUA

GUÍA DE IDENTIFICACIÓN DE NUBES



Aprenda a identificar los diferentes tipos de nubes gracias a este esquema inspirado en el *Atlas Internacional de Nubes*. Las nubes se dividen, en función de su forma general, en 10 grandes tipos llamados géneros. A su vez, cada género se subdivide en función de la forma, la estructura y la transparencia de la nube, la disposición de sus elementos, la presencia o ausencia de nubes segregadas, así como el desarrollo de la propia nube.

Figura 3.6: Fragmento de la guía de identificación de nubes [3].

Cabe mencionar que dicha guía es útil para clasificar una cantidad mayor de categorías de las 7 que se utilizaron durante el entrenamiento de los modelos del objetivo anterior. Sin embargo las imágenes capturadas se etiquetarán al mismo modo que el dataset utilizado durante este trabajo, lo cual significa que la reducción de categorías está conformada como sigue.

Categorías definidas en [33] y utilizadas en este trabajo	Categorías definidas según el Atlas internacional de las nubes
cumulus	cumulus
altocumulus	altocumulus, cirrocumulus
cirrus	cirrus, cirrostratus
cielo despejado	cielo despejado
stratocumulus	stratocumulus, stratus, altostratus
cumulonimbus	cumulonimbus, nimbostratus
mixta	mixta

3.3.3. Entrenamiento con las imágenes locales

Las imágenes que se han recopilado con el prototipo aunque son también imágenes de nubes son totalmente nuevas para el clasificador, así que no podemos esperar que tenga un buen desempeño clasificandolas sin hacer un reentrenamiento del modelo. Esto no significa que el aprendizaje que ya tiene el modelo con el dataset original no sirva de nada, sino que se pueden utilizar esos pesos obtenidos y solamente hacer un ajuste fino y utilizar la transferencia de aprendizaje al nuevo modelo que vamos a utilizar con las imágenes locales.
Cabe mencionar que al momento de hacer el nuevo entrenamiento se optó por hacer aumento de datos (*data augmentation*), esta técnica consiste en aumentar la cantidad de las imágenes de entrenamiento haciendo rotaciones y espejo a a algunas de las imágenes originales de manera aleatoria, incrementando el tamaño del dataset y ofreciendo al clasificador mayor información para aprender, esta técnica es utilizada generalmente cuando se tienen casos como este donde el conjunto de entrenamiento es relativamente chico.

3.4. Ajuste fino del algoritmo genético mediante CRS

La sección de resultados de este trabajo se divide principalmente en 4 modelos se encontraron mediante la optimización de una red neuronal convolucional a través de un algoritmo genético, pero una vez encontrados esos resultados parate del sínodo de este trabajo recomendó agregar un algoritmo de ajuste de parámetros del algoritmo genético por lo que se procedió a agregar dicho procedimiento produciendo los resultados del modelo 5 en la sección de resultados, los cuales muestran mejorías significativas en cuanto a la precisión de la red neuronal.
En esta sección se limitó a dar un poco de contexto del algortimo con el que se ajustaron los parámetros del algoritmo genético, dicho método esta basado en el trabajo de [34] publicado en 2016.

3.4.1. Descripción y funcionamiento del algoritmo (CSR).

En resumen, el algoritmo CRS-Tuning utiliza un enfoque inspirado en el sistema de calificación de ajedrez para evaluar, comparar y refinar iterativamente las configuraciones para algoritmos metaheurísticos. Mediante el uso del concepto de utilidad y un sistema de calificación, el algoritmo identifica y retiene configuraciones que muestran un mejor rendimiento en diferentes instancias de problemas.

Según el pseudocódigo proporcionado por el autor, el algoritmo funciona de la siguiente manera:

Inicialización

El algoritmo comienza inicializando un conjunto de configuraciones C con M configuraciones aleatorias. Estas configuraciones representan diferentes ajustes de parámetros para el algoritmo metaheurístico. El contador `execs` se establece en 0, que se utilizará para realizar un seguimiento del número de iteraciones a través del bucle principal.

Bucle de Ejecución

El algoritmo entra en un bucle donde itera `maxExecs` veces o hasta que se cumpla una cierta condición de detención. Para cada configuración c_j en el conjunto C, el algoritmo realiza los siguientes pasos:

- Para cada problema gl en el conjunto de problemas G, el algoritmo ejecuta la configuración c_j en el problema gl durante nG ejecuciones independientes.
- Las soluciones obtenidas de estas ejecuciones se guardan para comparaciones posteriores.

Bucle de Comparación de Juegos

Esta sección involucra comparaciones en pares entre diferentes configuraciones en cada problema. Para cada par de configuraciones c_j y c_k en C (donde $j < k$), el algoritmo realiza los siguientes pasos:

- Para cada problema gl en el conjunto de problemas G, el algoritmo realiza nG ejecuciones para comparar el rendimiento de c_j y c_k en gl.
- Basado en las soluciones obtenidas de las ejecuciones, el algoritmo determina el resultado de la comparación. Si la diferencia en el rendimiento es menor que un umbral δ, se registra un empate. De lo contrario, se considera ganadora a la configuración con mejor rendimiento.

Cálculo de Utilidad

Después de las comparaciones, el algoritmo calcula la utilidad de cada configuración c_j. La utilidad se calcula utilizando métricas relacionadas con la calificación, como la calificación (R), la desviación de la calificación (RD) y el intervalo de calificación (RI). Estas métricas se utilizan a menudo en sistemas de calificación como Glicko-2 para cuantificar la fuerza de los jugadores o configuraciones.

Ordenar y Eliminar Configuraciones

Las configuraciones en C se ordenan en orden descendente según sus calificaciones calculadas. La configuración con la calificación más alta se identifica como *cbest*. Para cada configuración c_j en C (excepto *cbest*), el algoritmo verifica si los intervalos de calificación de c_j y *cbest* se superponen. Si no se superponen o si el tamaño de C excede el tamaño máximo de padres (MPS), se elimina c_j de C. Este paso ayuda a retener solo las configuraciones más prometedoras.

Generación de Descendencia

Mientras el tamaño de C sea menor que M, el algoritmo genera nuevas configuraciones descendientes a través de dos procesos:

- Cruce Uniforme: Se generan configuraciones descendientes utilizando el cruce uniforme con una cierta probabilidad Cr. Esto implica combinar elementos de dos configuraciones padre.
- Mutación: Las configuraciones descendientes también están sujetas a mutación con una probabilidad F, lo que introduce pequeños cambios aleatorios en la configuración.

Se adaptó la metodología propuesta por el autor para hacer el ajuste de nuestro algoritmo genético (el código final se muestra en los apéndices) y a continuación se muestran cuales fueron los hiper-parámetros que fueron ajustados mediante el CRS:

Tabla 3.9: Espacio de búsqueda utilizado por el algoritmo de CRS

Hiper-parámetro	**Opciones**
Tamaño de población	4, 5, 6,7,8
Número de generaciones	2, 3, 4, 5
Tasa de supervivencia por ajuste	0.5, 0.7
Probabilidad de supervivencia por no ajuste	0.1, 0.2, 0.3
Tasa de mutación	0.3, 0.4, 0.5, 0.6, 0.7
Número de fases	2, 3

CAPÍTULO 4

Resultados

Como se observó en la metodología se consiguió entrenar 4 modelos bajo distintas condiciones principalmente diferenciándose en la cantidad y el tamaño de las imágenes con las que fue entrenado, además de los hiper-parámetros del algoritmo genético con que fue construido. El tamaño del dataset impacta directamente en la memoria RAM y el poder de cómputo que va a ser necesario, esta fue la principal restricción que fue guiando los experimentos y que al final nos permite tener una comparación más enriquecida por las disitintas condiciones de los modelos entrenados.

Los resultados se muestran principalmente en dos gráficos, el primero es un reporte de clasificación el cuál es generado mediante la biblioteca de aprendizaje automático *scikit-learn*. Este reporte contiene las métricas de precisión, *recall* y *f1-score*, además de mostrar el soporte de cada una de las categorías, es decir, la cantidad de imágenes que fueron evaluadas. A continuación se definen las métricas de evaluación que son utilizadas principalmente cuando se reporta el desempeño de un clasificador de imágenes:

Primeramente la gráfica de la figura 4.1 muestra qué es un falso negativo, falso positivo, verdadero positivo y verdadero negativo.

Predicción

Valor real		**Sí**	**No**
	Sí	Verdadero positivo	Falso negativo
	No	Falso positivo	Verdadero negativo

Figura 4.1: Casos posibles en los que puede resultar cada inferencia del clasificador.

Para las definiciones de las métricas de evaluación utilizaremos las siguientes abreviaturas por

simplicidad: **VP** = verdadero positivo, **FN** = falso negativo, **VN** = verdadero negativo, **FP** = falso positivo.

$$Precisión = \frac{VP}{VP + FP} \tag{4.1}$$

$$recall = \frac{VP}{VP + FN} \tag{4.2}$$

$$F1 - score = \frac{2\ x\ precisión\ x\ recall}{precisión + recall} \tag{4.3}$$

El segundo gráfico con el que se expresan los resultados es la matriz de confusión, esta matriz nos permite visualizar el desempeño del clasificador dado que muestra las intersecciones entre las etiquetas reales y las predicciones,es decir, se espera que la diagonal de la matriz contenga todos los casos de las predicciones hechas. Esto último significaría que el clasificador tiene un 100 % de precisión.

A continuación se muestran los resultados por separado de cada uno de los modelos, acompañados de una breve discusión de los mismos. Por simpleza y orden cronológico en el que fueron creados se han enumerado del 1 al 4 y llamados de la misma manera.

4.1. Modelo 1

El primer modelo fue entrenado utilizando un dataset de 1002 imágenes, el cual fue dividido 80 % para el entrenamiento y el 20 % para evaluarlo. En el momento en el que fue entrenado este modelo se tuvieron problemas para correr el algoritmo genético ya que la memoria RAM que se requería era mayor a la disponible, así que uno de los primeros ajustes fue reducir el tamaño del batch para los entrenamientos internos que se realizan en cada ciclo del algoritmo, se hicieron varias pruebas reduciendo el tamaño del batch hasta quedar finalmente el parámetro con el siguiente valor ***batch size = 8***. Además de reducir este parámetro tambien se hicieron varias pruebas modificando el tamaño de la población el número de generaciones y el número de fases quedando con los siguientes valores:
Población: 4
Generaciones: 3
Fases: 2

En la tabla 4.1 se muestra el reporte completo de clasificación, donde se observan distintas métricas que nos ofrecen una mejor evaluación del algoritmo clasificador.
Principalmnte nos podemos enfocar en que de manera general el clasificador tiene una precisión del **93 %**, el cuál ya es comparable con los distintos métodos que han sido publicados en la literatura. Este análisis se despliega para cada una de las categorías, lo cual nos permite observar que algunas de las categorías son más fáciles de distinguir logrando 100 % de precisión en ambas como es el caso de la categoría cielo despejado.

Tabla 4.1: Reporte de Clasificación para el modelo 1.

Reporte de Clasificación				
	precision	**recall**	**f1-score**	**support**
1_cumulus	1.00	0.92	0.96	36
2_altocumulus	1.00	0.87	0.93	23
3_cirrus	0.87	1.00	0.93	33
4_cielo despejado	1.00	1.00	1.00	31
5_stratocumulus	0.94	0.74	0.83	23
6_cumulonimbus	0.88	1.00	0.93	28
7_mixto	0.86	0.93	0.89	27
accuracy			0.93	201
macro avg	0.94	0.92	0.92	201
weighted avg	0.94	0.93	0.93	201

La figura 4.9 muestra la matriz de confusión del modelo final que fue entrenado por 321 épocas. Para este modelo se pueden observar los casos por clase donde el clasificador se equivoca, es decir, hace una predicción de una etiqueta diferente de la real, además de que en esta matriz se pueden onservar de manera gráfica los casos de falsos positivos y falsos negativos.

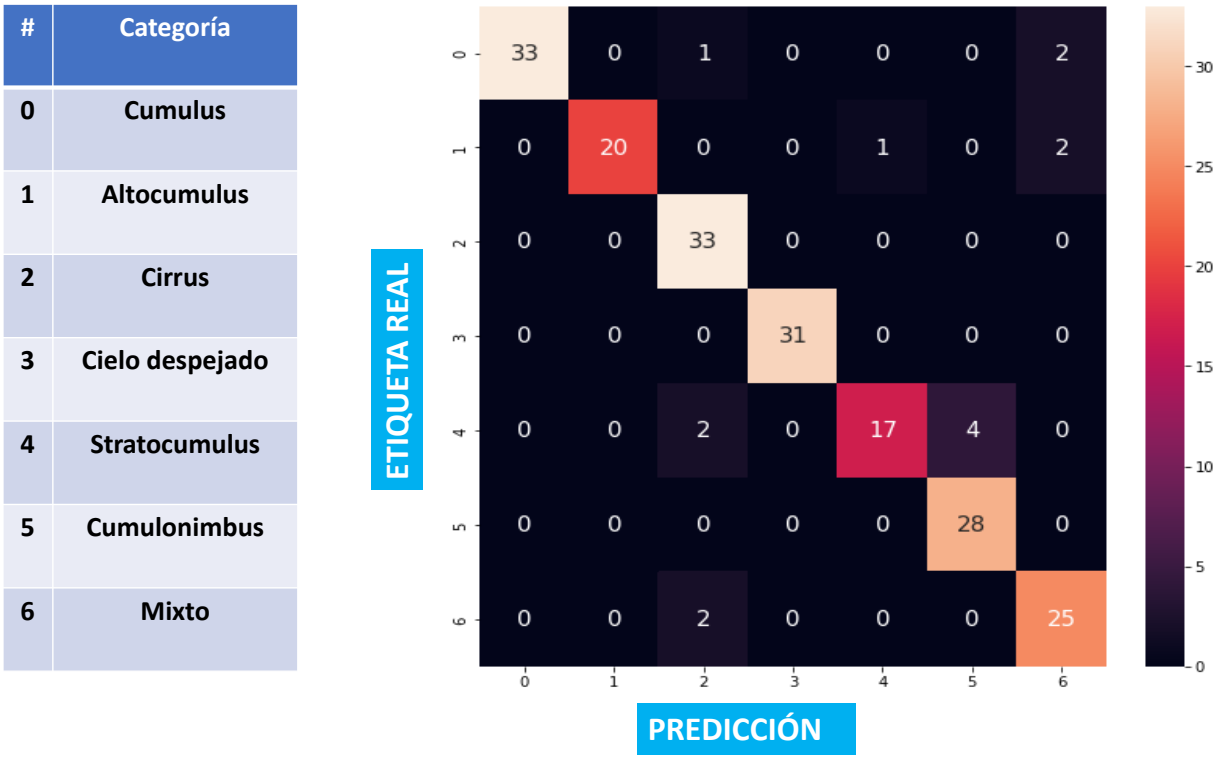


Figura 4.2: Matriz de confusión para el primer modelo

4.2. Modelo 2

Para el segundo modelo se adquirieron los recursos computacionales mencionados en la tabla 3.3, de modo que se pudo aumentar el tamaño del dataset quedando conformado por 5400 imágenes igualmente dividido en entrenamiento y evaluación tomando 80 % y 20 % respectivamente del total, y con un tamaño de 512 x 512 pixeles.
El tamaño del batch no fue modificado para este y los sigueintes experimentos, manteniendo el mismo valor del modelo 1, sin embargo los hiper-parámetros del algortimo genético se aumentaron con la intención de tener más combinaciones.
El primer hiper-parámetro que se modificó fue el tamaño de la **población** aumentándolo a **8** individuos lo cual permite que se genere mayor cantidad de posibles soluciones y se explote de mejor manera el espacio de búsqueda que se muestra en la tabla 3.4.

En la tabla 4.3 se observa que la precisión del modelo alzanzó hasta un 90 % de manera general o en promedio con un soporte de 1082 imágenes de prueba. Este segundo modelo con la arquitectura mostrada en 3.4b también comprueba la hipótesis como verdadera, mostrando la robustez del algoritmo genético ya que se ha aumentado la cantidad de imágenes con las que ha sido entrenado.
Sin embargo si observamos a detalle este reporte podemos encontrar algunas categorías donde la precisión es menor, siendo la categoría de nubes mixtas la que menor porcentaje alcanza con un 75 % en precisión lo cual indica que hay un 25 % de las imágenes pertenecientes a la categoría que han sido clasificadas con alguna otra etiqueta.
Este fenómeno se puede entender de mejor manera si observamos el ejemplo de la figura 4.3 donde tenemos una imagen que pertenece a la categoría mixta 4.14d, sin embargo observamos que las nubes que predominan son las de la categoría cirrus con algunos cumulos en un tono más oscuro y esto lleva al clasificador a hacer una clasificación incorrecta, aunque es solo parcialmente incorrecta, dado que dicha categoría errónea si existe en la imagen. Si comparamos con la imagen 4.14c podemos ver que es el mismo fondo de la imagen mixta sin los cumulos delante.
Estas observaciones pueden ser o no de importancia dependiendo de la aplicación donde se utilize el clasificador, dado que si la categoría de la predicción si existe dentro de la imagen y además predomina en ella puede aceptarse como no error.

(a) Categoría real: cirrus, Clasficación: cirrus (b) Categoría real: mixta, Clasificación: cirrus

Figura 4.3: Ejemplo de una predicción parialmente incorrecta del clasificador.

En la figura 4.9 se muestra la matriz de confusión correspondiente al segundo modelo donde se puede observar con mayor facilidad el fenómeno explicado anteriormente, ya que podemos ver que de las 150 muestras que pertenecen a la categoría mixto, 123 fueron verdaderos positivos mientras que el resto fueron falsos positivos los cuales se dividieron en distintas categorías predominando cirrus y altocumulus.

#	Categoría
0	Cumulus
1	Altocumulus
2	Cirrus
3	Cielo despejado
4	Stratocumulus
5	Cumulonimbus
6	Mixto

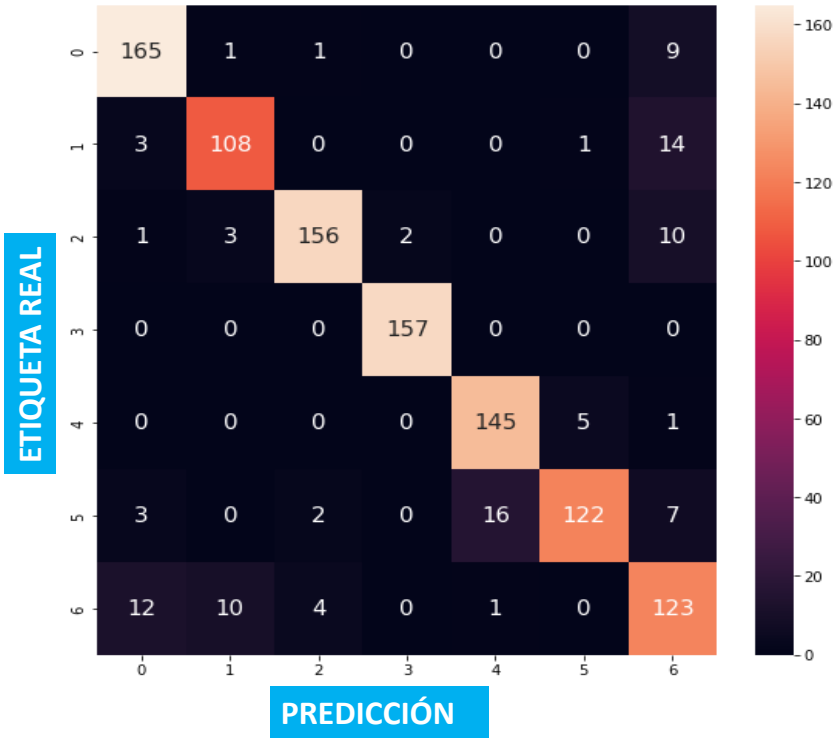


Figura 4.4: Matriz de confusión para el segundo modelo

Un punto importante del que no se ha hecho mención es la métrica del valor f (*f1-score*) el cual también se puede utilizar para comparar con otros modelos dado que resume o hace un promedio de la precisión y el recall, ademas de que es una técnica utilizada cuando las clases no están balanceadas como en este caso.
Así que de modo general si queremos saber que tan bueno es un clasificador y no nos interesa mucho enfocarnos en si comete más falsos negativos o falsos negativos, podemos fijarnos simplemente en esta métrica que nos estará dando información más completa que solo fijarnos en la precisión, en este caso de manera general obtiene un valor de 0.9 en una escala de 0 a 1, ademas de que se puede observar dicho valor para cada categoría.

Tabla 4.2: Reporte de Clasificación para el modelo 2

Reporte de Clasificación				
	precision	**recall**	**f1-score**	**support**
1_cumulus	0.90	0.94	0.92	176
2_altocumulus	0.89	0.86	0.87	126
3_cirrus	0.96	0.91	0.93	172
4_cielo despejado	0.99	1.00	0.99	157
5_stratocumulus	0.90	0.96	0.93	151
6_cumulonimbus	0.95	0.81	0.88	150
7_mixto	0.75	0.82	0.78	150
accuracy			0.90	1082
macro avg	0.90	0.90	0.90	1082
weighted avg	0.91	0.90	0.90	1082

El segundo modelo fue entrenado por 161 épocas en las gráficas 4.5 y 4.6 se muestra el comportamiento de los parámetros de evaluación *loss* y *accuracy* respectivamente. El primer parámetro de evaluación *loss* es obtenido directamente de la función de pérdida que se este utilizando (Relu, sigmoid, softmax, etc.,) en la última capa densa la cual es la que hace la clasificación, en este caso utilizamos la función softmax con el número de neuronas igual al número de categorías, este parámetro es el que toma en cuenta el algoritmo para ir modificando los pesos de la red neuronal generalmente por medio de un gradiente descendente, dicho de otro modo este parámetro representa el error que esta obteniendo la red en cada época. Esta métrica no tiene unidades y su valor es único para cada modelo por lo que no puede ser comparado con otros modelos y solo es usado para lo que ya se comentó y también para comparar con otros entrenamientos del mismo modelo.
El segundo parámetro corresponde a la precisión que tiene de manera general el clasificador, tiene un rango de 0 a 1 y nos da información de cuantos aciertos tiene el clasificador sobre todas las predicciones hechas, técnicamente esta definido en la ecuación (4.1).

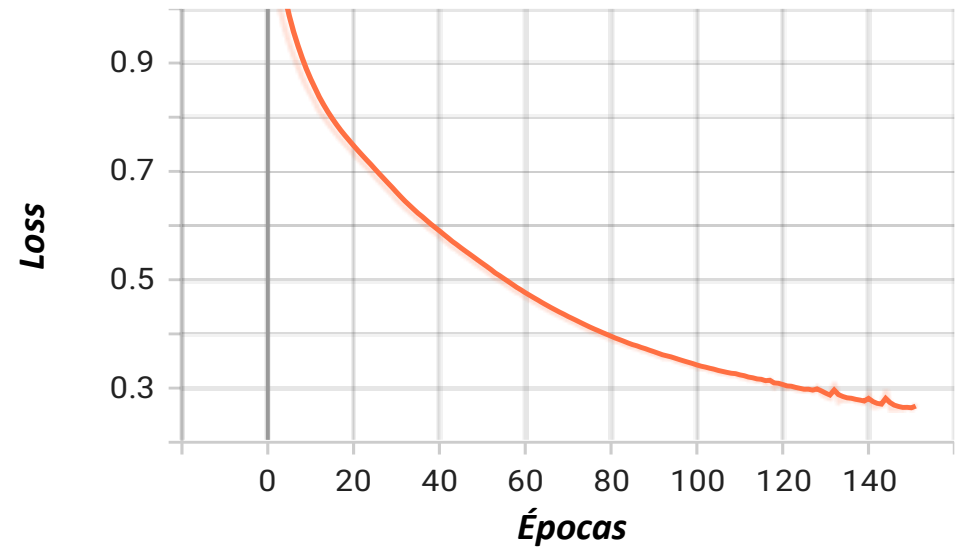


Figura 4.5: Gráfica de comportamiento de la función de pérdida para las primeras 145 épocas

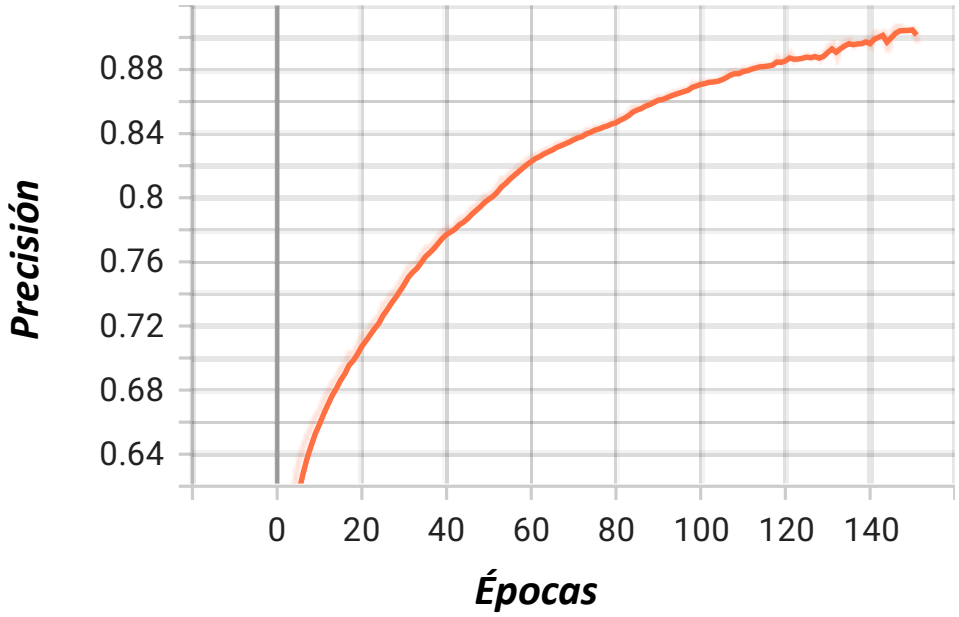


Figura 4.6: Gráfica de comportamiento de la precisión para las primeras 145 épocas

En ambas gráficas 4.5 y 4.6 se observa el comportamiento del entrenamiento como durante las primeras épocas el error va disminuyendo al mismo tiempo en que la precisión aumenta de similar forma. En la metodología se menciona que el modelo final se puede entrenar de manera exhaustiva. Sin embargo cuando la tendencia en el error parece mantenerse en el mismo error sin disminuir más o inclusive comienza a aumentar podemos llegar a un fenómeno conocido como sobreentrenamiento, lo cuál se debe evitar.

A continuación se muestran las gráficas de las últimas épocas del total (161) por las que fue entrenado este modelo.

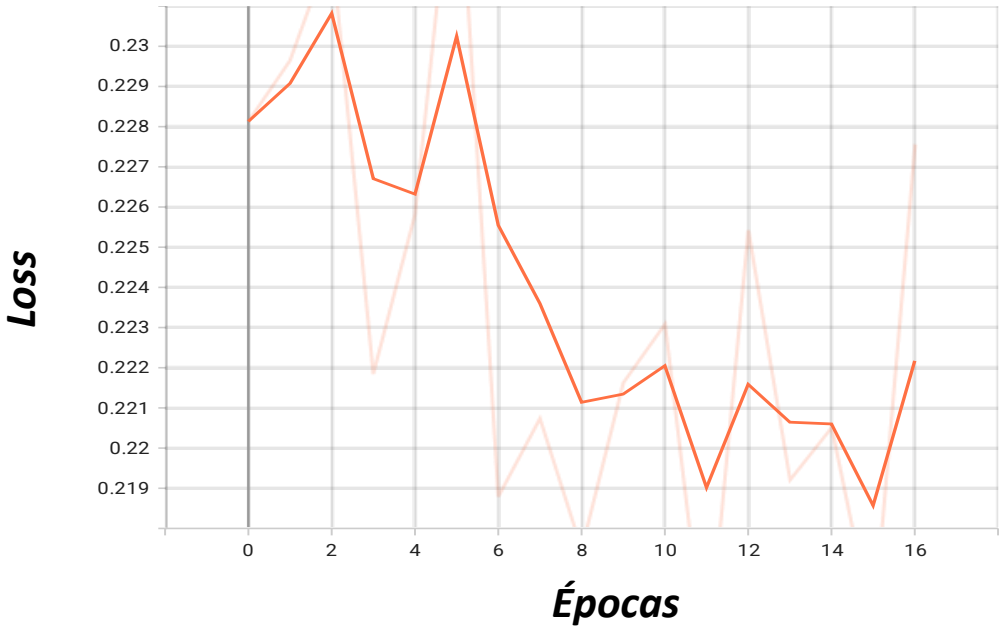


Figura 4.7: Gráfica de comportamiento de la función de pérdida para las últimas 16 épocas

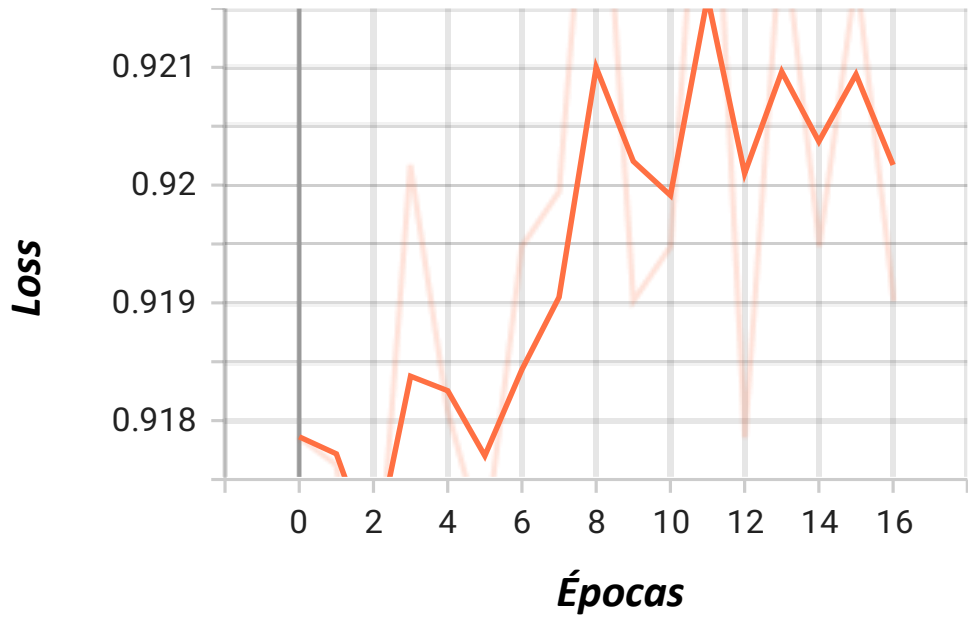


Figura 4.8: Gráfica de comportamiento de la precisión para las últimas 16 épocas

En las gráficas de las figuras 4.7 y 4.8 se puede apreciar con una mejor escala el comportamiento

final del entrenamiento, dado que se observan solo las últimas 16 épocas. Como se mencionó anteriormente se debe tener un criterio de parada para el entrenamiento y es por esto que dado que los recursos eran limitados no valía la pena continuar entrenando, puesto que en la gráfica 4.8 se puede observar que la precisión para las épocas de la 8 a la 16 en promedio es la misma, inclusive se observa como oscila al final entre los mismos valores sin dar un aumento significativo. Lo mismo pasa para la gráfica del error para las mismas épocas al final se observa que en promedio el error se mantiene en el mismo valor, sin disminuir significativamente. Esto último no significa que ya no puede mejorar más la red, aún se puede hacer alguna modificación a la arquitectura de la red o a los hiper-parámetros del entrenamiento, pero en este caso debido a la escasez de los recursos se utilizó dicho criterio para detener el entrenamiento.

4.2.1. Entrenamiento con el dataset balanceado

Como recomendación de los sinodales se hicieron pruebas entrenando el mismo modelo con un dataset balanceado en cuanto a número de imágenes por categoría. Se utilizaron 731 imágenes por categoría sumando un total de 5117 imágenes y obteniendo los siguientes resultados:

Tabla 4.3: Reporte de Clasificación para el modelo 2 con dataset balanceado

Reporte de Clasificación				
	precision	**recall**	**f1-score**	**support**
1_cumulus	0.84	0.91	0.87	168
2_altocumulus	0.80	0.81	0.80	134
3_cirrus	0.88	0.73	0.80	138
4_cielo despejado	0.96	0.99	0.98	160
5_stratocumulus	0.94	0.97	0.95	135
6_cumulonimbus	1.00	0.86	0.92	142
7_mixto	0.76	0.84	0.80	147
accuracy			0.88	1024
macro avg	0.88	0.87	0.88	1024
weighted avg	0.88	0.88	0.88	1024

Como se puede observar en el reporte de clasificación el entrenamiento con un dataset balanceado no produce algún cambio significativo dado que alcanza un porcentaje ligeramente menor al alcanzado con un dataset no balanceado pero con más muestras siendo 5400. Así que después de este experimento los siguientes ya no se hicieron con dataset balanceado.

4.3. Modelo 3

Para el tercer modelo se hicieron varias modificaciones, una de las principales fue la reducción del tamaño de las imágenes quedando en un tamaño de las muestras de **299 x 299**. Lo anterior fue motivado principalmente porque uno de los artículos más representativos publicados recientemente utilizó dicho tamaño para probar su técnica la cuál fue un ensamble de dos redes conocidas [23]. De este modo se puede hacer una comparación más justa entre las técnicas utilizadas y que nos permite sacar algunas conclusiones importantes, además debido a que se ha reducido el tamaño de

las imágenes nos permitió aumentar el número de imágenes utilizando todas las muestras disponibles en el dataset, haciendo más robusto el clasificador.
En cuanto al algoritmo genético los hiper-parámetros establecidos para generar el modelo fueron los siguientes:
Población: 8
Generaciones: 4
Fases: 4
donde se resalta que se ha dejado correr el algoritmo por cuatro fases, generando el modelo más grande en cuanto al número de capas que puede añadir.

En la tabla 4.4 se meustra el reporte de clasificación del tercer modelo cuya arquitectura se muestra en la figura 3.5a, además de reducir el tamaño de las imágenes se pudo aprovechar el dataset en su totalidad para entrenar el modelo. En total se utilizaron las 8000 imágenes para los modelos 3 y 4. Dicho reporte comprueba la hipótesis al alcanzar una precisión mayor al 90 % con un soporte de 1400 imágenes de prueba. A pesar de ser distintos modelos y que se han entrenado con diferente cantidad de imágenes y tamaño se mantiene la tendencia en las categorías con menor precisión. En este caso la categoría peor evaluada es otra vez la de nubes mixtas con un valor f de 0.82.
A pesar de que se entrenó con una cantidad mayor de imágenes obtuvo en general un mejor desempeño que el segundo modelo, lo cuál se le puede atribuir a que la arquitectura es más robusta al tener mayor "profundidad".

Tabla 4.4: Reporte de Clasificación para el modelo 3.

Reporte de Clasificación

	precision	**recall**	**f1-score**	**support**
1_cumulus	0.98	0.89	0.94	256
2_altocumulus	0.88	0.88	0.88	129
3_cirrus	0.97	0.89	0.93	237
4_cielo despejado	0.97	1.00	0.98	250
5_stratocumulus	0.81	0.99	0.89	159
6_cumulonimbus	0.99	0.83	0.90	197
7_mixto	0.75	0.89	0.82	172
accuracy			0.91	1400
macro avg	0.91	0.91	0.91	1400
weighted avg	0.92	0.91	0.91	1400

A continuación se muestra la matriz de confusión de este modelo, si continuamos analizando la categoría de nubes mixtas como lo hicimos en el modelo anterior, observamos que en este caso la tendencia es mayor en clasificar este tipo de imágenes como nubes del tipo cumulus, el cual obtuvo 22 falsos positivos seguido de las categorías cirrus y altocumulos. Después de la categoría de nubes mixtas le sigue la de stratocumulus que también obtuvo 31 falsos positivos en una sola categoría que fue la de los cumulonimbus.

#	Categoría
0	Cumulus
1	Altocumulus
2	Cirrus
3	Cielo despejado
4	Stratocumulus
5	Cumulonimbus
6	Mixto

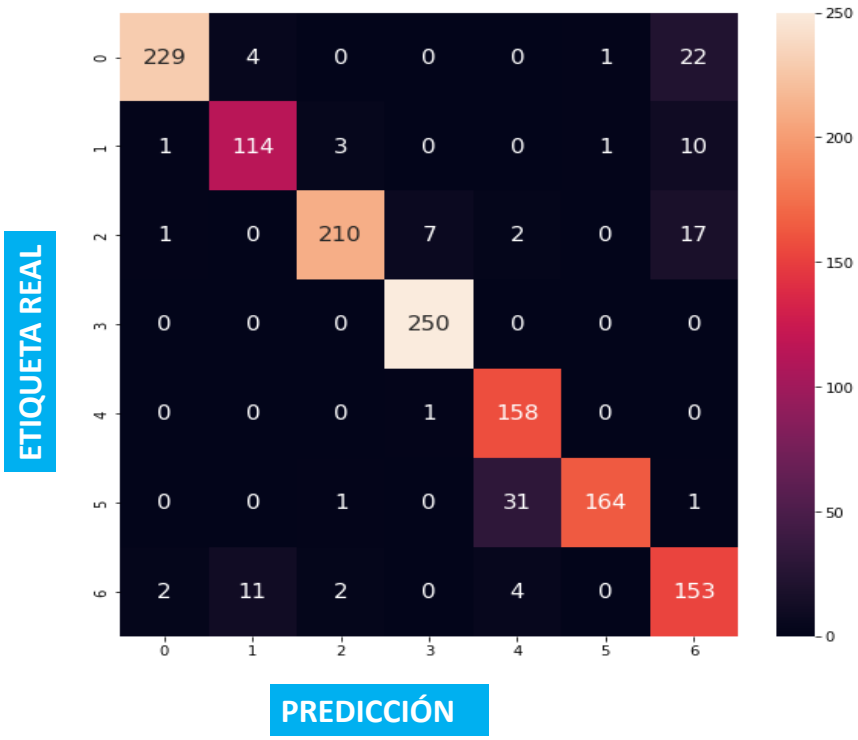


Figura 4.9: Matriz de confusión para el tercer modelo

Finalmente en las figuras 4.10 y 4.11 se muestra el comportamiento del entrenamiento a través del *loss* y la precisión, las cuales como ya se mostró son inversamente proporcionales.

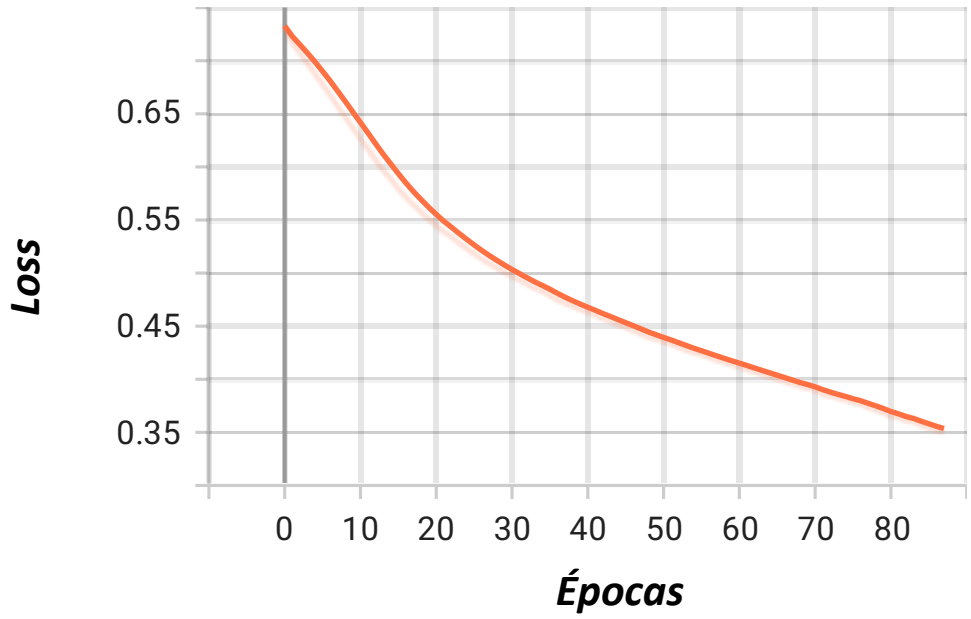


Figura 4.10: Comportamiento de la función de pérdida durante las primeras épocas del modelo 3

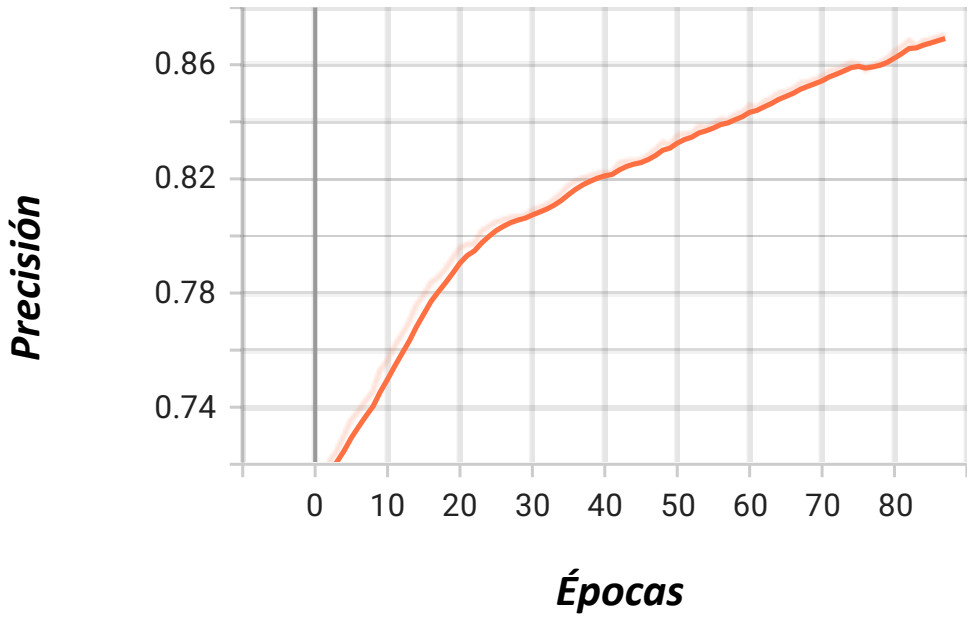


Figura 4.11: Comportamiento de la precisión durante las primeras épocas del modelo 3

Las siguientes figuras 4.12 y 4.13 pertenecen a las últimas épocas del entrenamiento como una especie de zoom a las gráficas anteriores. Es importante mencionar que la gráfica de precisión pertenece al cálculo que se hace durante el mismo entrenamiento no a la evaluación, es decir, que las gráficas son obtenidas cuando se utiliza la partición del dataset con el que se entrena cuyas imágenes son diferentes a las que se utilizan para obtener el reporte de clasificación así como la matriz de confusión. Estas imágenes deben de ser nuevas para el clasificador, es decir, durante la

etapa de aprendizaje no fueron mostradas, de este modo se puede evaluar realmente si el clasificador realmente aprendió o adquirió la capacidad de generalizar.
Debido a lo anterior las gráficas del comportamiento pueden mostrar una precisión mayor a la que se ve en el reporte. Sin embargo la diferencia entre ambas métricas no debe ser tan grande ya que se puede tratar de un caso de sobreajuste, lo cual sucedió en algunos de los experimentos, esto ocurre cuando al clasificador se aprende todos los ejemplos del dataset de entrenamiento pero no ha generalizado bien lo cual provoca que ante nuevos ejemplos tenga una precisión muy baja. El último modelo es un ejemplo de sobreajuste se ha añadido a la sección de resultados de manera que se pueda, comparar con los otros modelos exitosos y se puedan sacar conclusiones.

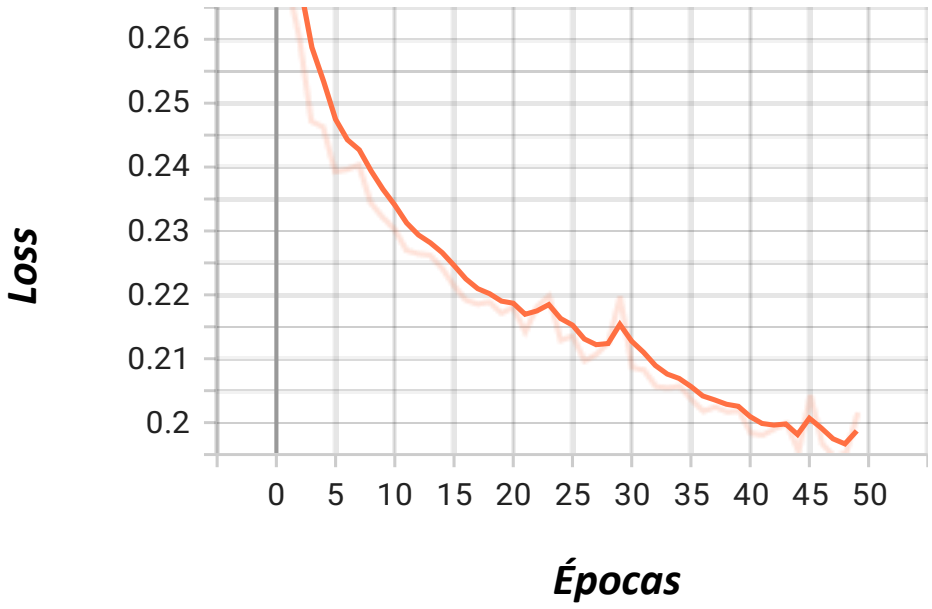


Figura 4.12: Comportamiento de la función de pérdida durante las últimas épocas del modelo 3

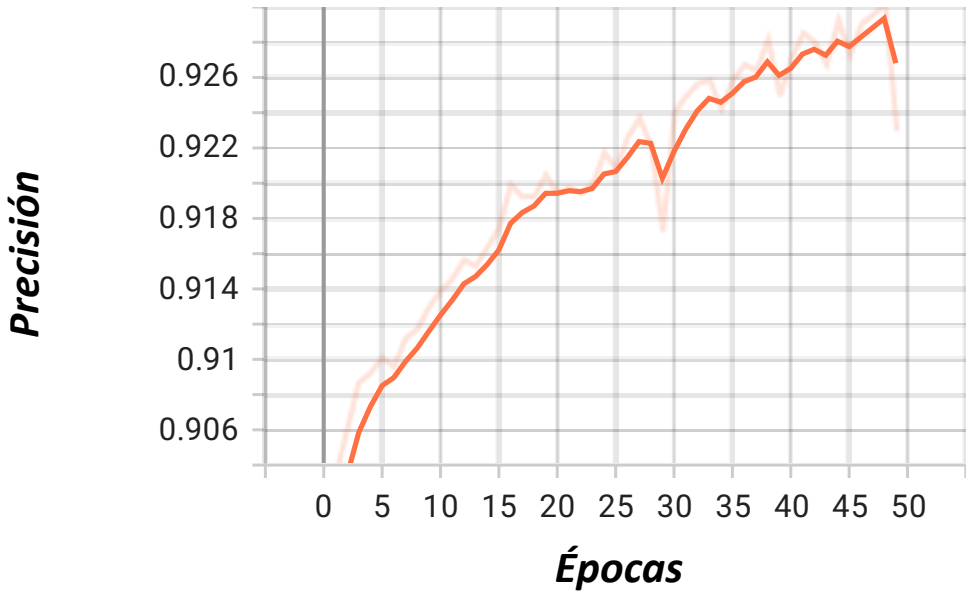
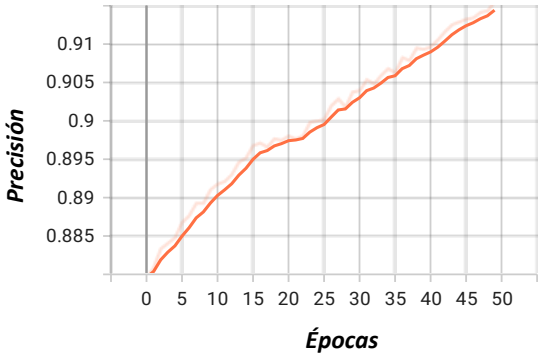


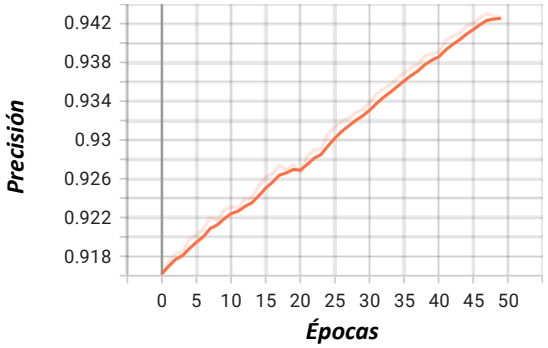
Figura 4.13: Comportamiento de la precisión durante las últimas épocas del modelo 3

4.4. Modelo 4

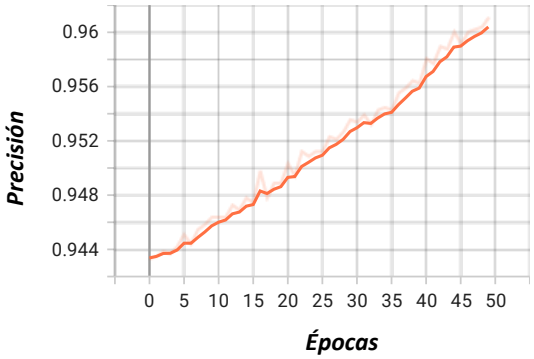
Este último modelo se ha añadido a la sección de resultados como ejemplo de un experimento fallido dado que no logró el porcentaje de precisión que se propuso en la hipótesis, puesto que antes de llegar a dicho porcentaje comenzó a sufrir de sobreajuste en el entrenamiento. Este modelo 4 cuya arquitectura se muestra en la figura 3.5b se generó de la misma manera que los anteriores con los siguiente valores de hiper-parámetros para el algoritmo genético:
Población: 8
Generaciones: 8
Fases: 3
al igual que el modelo anterior se utilizó el dataset completo de 8000 imágenes de 299 x 299 píxeles.



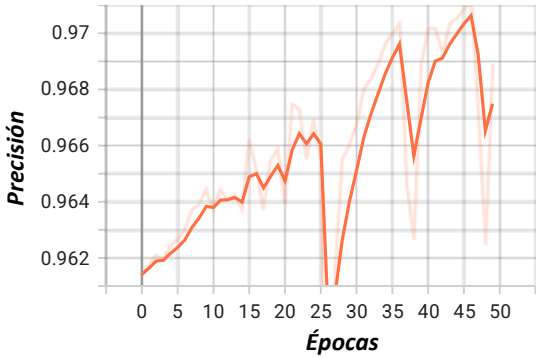
(a) épocas: 0-100



(b) épocas: 100-150



(c) épocas: 151-200



(d) épocas: 201-250

Figura 4.14: Evolución de la precisión durante el entrenamiento cada 50 épocas.

Las gráficas de la figura 4.14 pertenecen a la precisión del modelo durante el entrenamiento. Se puede observar que en las últimas épocas supera hasta el 97 % de precisión, sin embargo despúes de cada 50 épocas se realizó la evaluación del modelo con la partición de prueba como se muestra en la tabla 4.5 y se puede observar como el modelo con el paso de las épocas aumenta en precisión pero llega un punto donde esta comienza a caer drásticamente. Como ya mencionamos esto significa que tiene un sobreajuste, es decir, que la precisión en la evaluación es mucho menor que en el entrenamiento.

Tabla 4.5: Evaluación del modelo a través de las épocas

# de épocas	**Evaluación**
0-100	0.77
101-150	0.83
151-200	0.86
201-250	0.73

Finalmente se muestra el reporte completo de clasificación de este modelo en la tabla 4.6.

Tabla 4.6: Reporte de Clasificación para el modelo 4.

Reporte de Clasificación				
	precision	**recall**	**f1-score**	**support**
1_cumulus	0.46	0.99	0.63	256
2_altocumulus	1.00	0.33	0.49	129
3_cirrus	0.99	0.45	0.62	237
4_cielo despejado	1.00	1.00	1.00	250
5_stratocumulus	0.91	0.86	0.88	159
6_cumulonimbus	0.90	0.85	0.88	197
7_mixto	0.68	0.42	0.52	172
accuracy			0.74	1400
macro avg	0.85	0.70	0.72	1400
weighted avg	0.84	0.74	0.73	1400

4.5. Base de datos de imágenes locales

Una vez reunidos los materiales que se muestran en la tabla 3.8 se procedió a hacer un setup de manera local para generar una base de datos con imágenes locales, las cuales fueron capturadas en un domicilio de la colonia Adara en el municipio del Marqués en el estado de Querétaro. Las figuras en 4.15 muestran el prototipo ya capturando imágenes, es decir, la cámara con el lente gran angular conectados a la raspberry pi3, se observa también el cable de alimentación y el cable de red, para poder establcer conexión con la raspberry desde una computadora en un lugar accesible, ya que la cámara se colocó en un lugar lo más alto posible en la azotea de una casa para que existiera la minima cantidad de objetos que pudieran aparecer en las imágenes como árboles o edificios.

Figura 4.15: Setup local para capturar las imágenes de cielo completo (primera fase).

Se logró obtener un total de 618 imágenes correspondientes a distintas categorías capturadas en el periodo del 22 de febrero al 29 de marzo del 2023. Es importante mencionar que se decidió hacer un primer corte o lote de imágenes para una primera fase de pruebas y obtener algunos resultados, y decidir si hay que modificar algún aspecto importante como el tamaño o la resolución de las imágenes. En la tabla 4.7 se muestra el desglose de las imágenes ya etiquetadas acorde al procedimiento mostrado en la metodología.

Se puede resaltar que debido al poco tiempo de estar capturando la mayoria de ellas corresponden a las categorías de cirrus y cielo despejado, aunque si se lograron observar al menos algunas muestras de las demás categorías. Es importante mencionar también que de las imágenes tomadas se tuvieron que descartar varias muestras que se consideraron no aptas para el experimento, por diferentes razones entre ellas principalmente porque aparecian objetos en las imágenes, o el lente se había desenfocado. También en la figura 4.16 se observan algunas de las muestras capturadas.

Tabla 4.7: Desglose de muestras capturadas por categoría

Categorías	**Muestras obtenidas**
cumulus	39
altocumulus	13
cirrus	193
cielo despejado	188
stratocumulus	2
cumulonimbus	27
mixta	53
descartadas	103
total	**618**

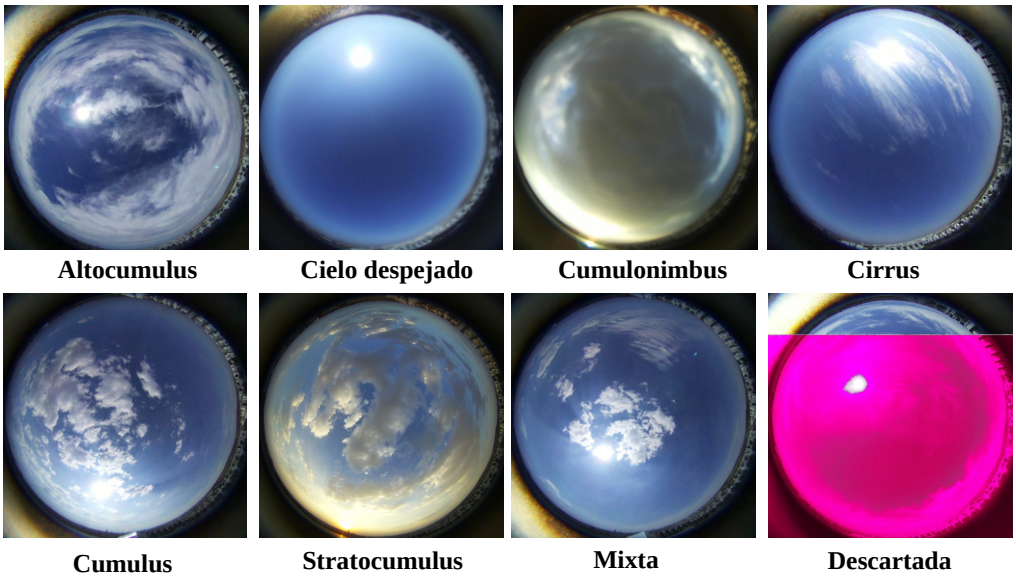


Figura 4.16: Ejemplos de las imágenes capturadas durante la primera fase

Finalmente el prototipo se colocó en un gabinete para exteriores, a prueba de agua y polvo para agregarle mayor robustez y se vea cada vez más como un prototipo de instrumento, en la figura 4.17 se observa el prototipo final listo para ser colocado en el mismo sitio o en otro y comenzar con la segunda etapa de captura y clasificación.

Figura 4.17: Imágenes del prototipo listo para colocarse en cualquier zona donde se desee clasificar nubes.

4.6. Clasificación de las imágenes locales

El prototipo de la figura 4.15 sirvió durante la primera fase para tomar algunas imágenes y comenzar a probar los modelos con imágenes locales. Para comenzar a hacer estas pruebas se realizaron las siguientes tareas:

- El primer paso fue recortar las imágenes debido a que la cámara las captura originalmente en forma de rectángulo y los modelos trabajan con imágenes cuadradas, es importante mencionar que en este paso no se hizo un ajuste al tamaño, es decir, un *resize* sino que se recortaron partes de las orillas para convertirlas en cuadrados pasandod e un tamaño de imagen de 4056 x 3040 píxeles a 3040 x 3040 píxeles como se muestra en la figura 4.18.

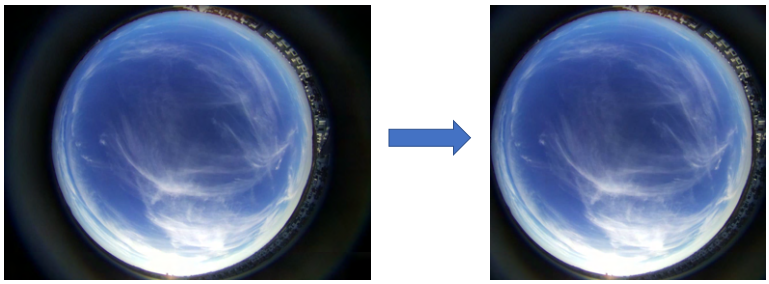


Figura 4.18: Comparación de las imágenes capturadas después del recorte

- El siguiente paso fue importar el modelo 2 generado anteriormente y volver a hacer un entrenamiento con el dataset de imágenes locales produciendo los siguientes resultados:

#	Categoría
0	Cumulus
1	Altocumulus
2	Cirrus
3	Cielo despejado
4	Cumulonimbus
5	Mixto

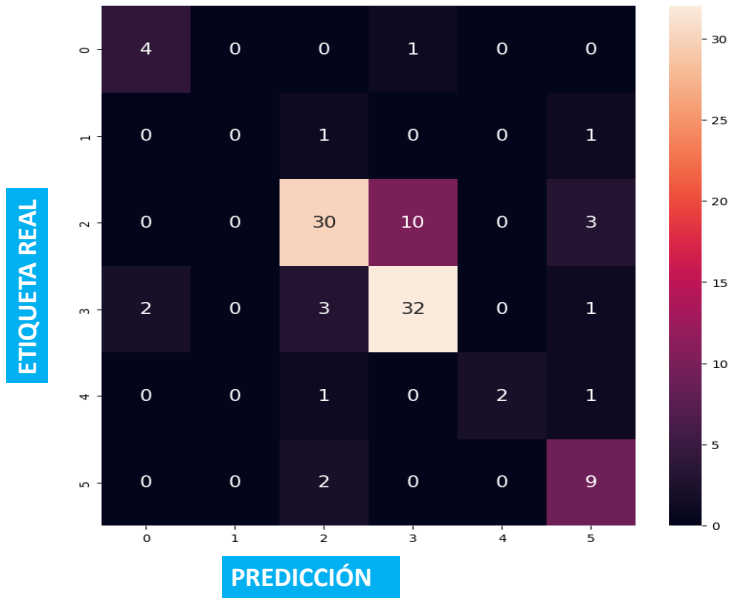


Figura 4.19: Matriz de confusión para el modelo local

A partir de este momento le llamaremos modelo local haciendo alusión a que ha sido entrenado con imágenes locales. La primer observación a resaltar en esta matriz de confusión de la figura 4.19 es que solo contiene 6 categorías de nubes y no 7 como en los modelos anteriores la razón de esto es que el dataset solamente contiene 2 muestras de la categoría 'stratocumulus' y al momento de hacer las particiones del mismo para entrenar y evaluar, las únicas dos muestras se quedaron en el conjunto de entrenamiento, y al no haber imágenes de esa categoría para evaluar no se generan datos para poder mostrar.

- A continuación se muestra el reporte de clasificación generado con esta misma evaluación:

Tabla 4.8: Reporte de Clasificación para el modelo local.

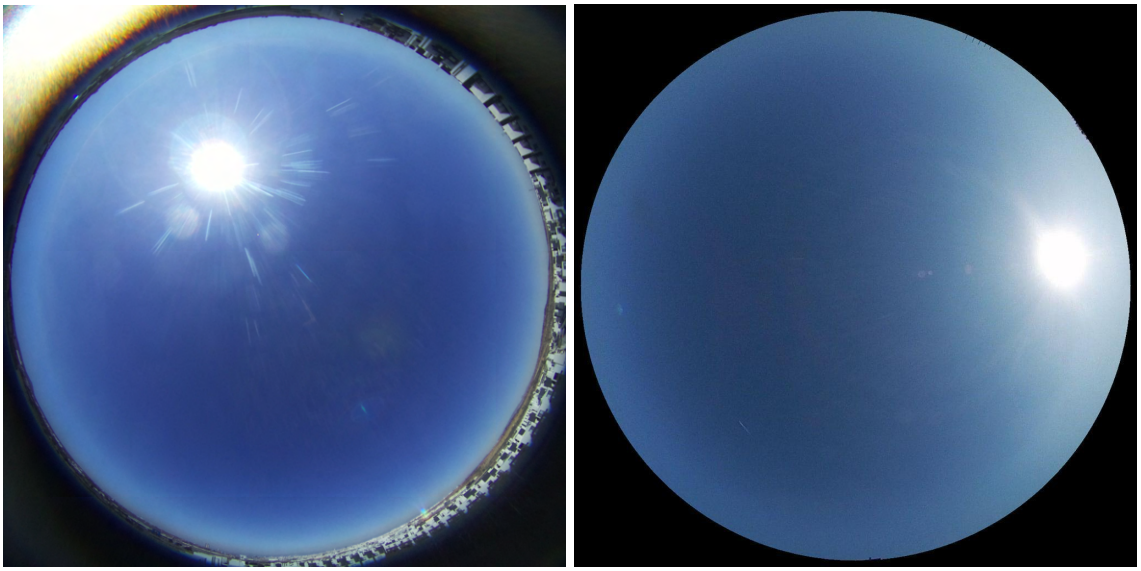
Reporte de Clasificación

	precision	**recall**	**f1-score**	**support**
1_cumulus	0.67	0.80	0.73	5
2_altocumulus	0.00	0.00	0.00	2
3_cirrus	0.81	0.70	0.75	43
4_cielo despejado	0.74	0.84	0.79	38
5_cumulonimbus	1.00	0.50	0.67	4
6_mixto	0.60	0.82	0.69	11
accuracy			0.75	103
macro avg	0.64	0.61	0.60	103
weighted avg	0.75	0.75	0.74	103

De manera general el clasificador obtiene un 75 % de precisión, lo cual, es un buen signo para ser el primer intento con imágenes propias. Sin embargo, si hay algunas areas de oprtunidad en las cuales debemos poner atención para mejorar el rendimiento.

La primera de ellas como ya se mencionó en el punto anterior, el dataset necesita enriquecerse con más muestras y no sólo más muestras sino de categorías específicas, dado que durante esta primera recopilación la mayoría fueron cirrus y cielo despejado y las menos observadas fueron stratocumulos, altocumulos y cumulonimbus.

Otra observación a resaltar es que la categoría de cielo despejado obtuvo una precisión y recall menor comparado con los modelos anteriores, esto muestra la dificultad de trabajar con imágenes que no se capturaron bajo las mismas condiciones (cámara, lente, lugar, etc.). Comparando una de las muestras que etiquetamos como cielo despejado de las imágenes locales con una del dataset MGCD 4.20 vemos que la imagen capturada en 4.22a pertenece claramente a cielo despejado pero probablemente los rayos del sol son capturados de diferente manera lo cual no aparece en las imagenes de MGCD 4.22b y como se observa en la matriz de confusión la categoría con la que más se confunde es con cirrus. Por lo tanto habría que intentar resolver la manera en la que se capturan las imágens y poder quitar la influencia del sol en el clasificador o mantenerlo así pero fortalecer el clasificador con más imágenes.



(a) Categoría cielo despejado, imágenes locales. (b) Categoría cielo despejado, dataset MGCD.

Figura 4.20: Imágenes pertenecientes a la categoría cielo despejado tomadas de diferente dataset.

Dado que es la categoría más fácil de clasificar al resolver este problema el la precisión del clasificador aumentaría considerablemente, ya que se clasifica de mejor manera esta categoría y también la de cirrus que es con la que esta teniendo interferencia o falsos positivos.

4.6.1. Preprocesamiento de las imágenes capturadas para eliminar ruido de luz natural

Observando nuevamente la figura 4.20(a) nuestras imágenes capturadas estaban influenciadas por la luz del sol (esquina superior izquierda), lo cual levantó sospechas de que podia estar teniendo efectos negativos sobre el aprendizaje del clasificador ya que el ruido no se representa de la misma manera en las diferentes imágenes, sino que a veces es más intenso o menos. Debido a esta situación se procedió a hacer un preprocesamiento a las imágenes para eliminar ese ruido, colocando una máscara de ceros con la forma del contorno circular manteniendo la información del cielo sin alterar y lo que hay fuera del circulo haciendolo completamente negro. La imagen de la máscara que se utilizó se muestra en la figura 4.21 eliminando el ruido de las imágenes de modo que el clasificador tenga menos perturbaciones al momento del aprendizaje.

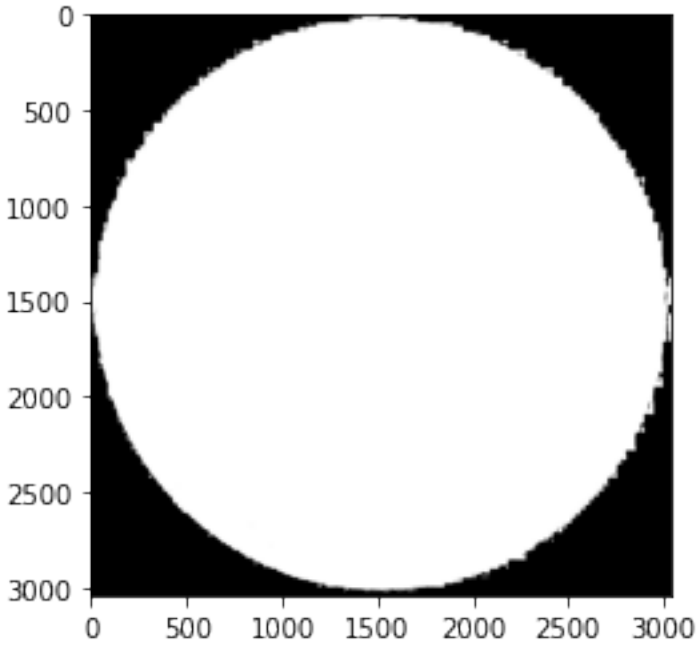
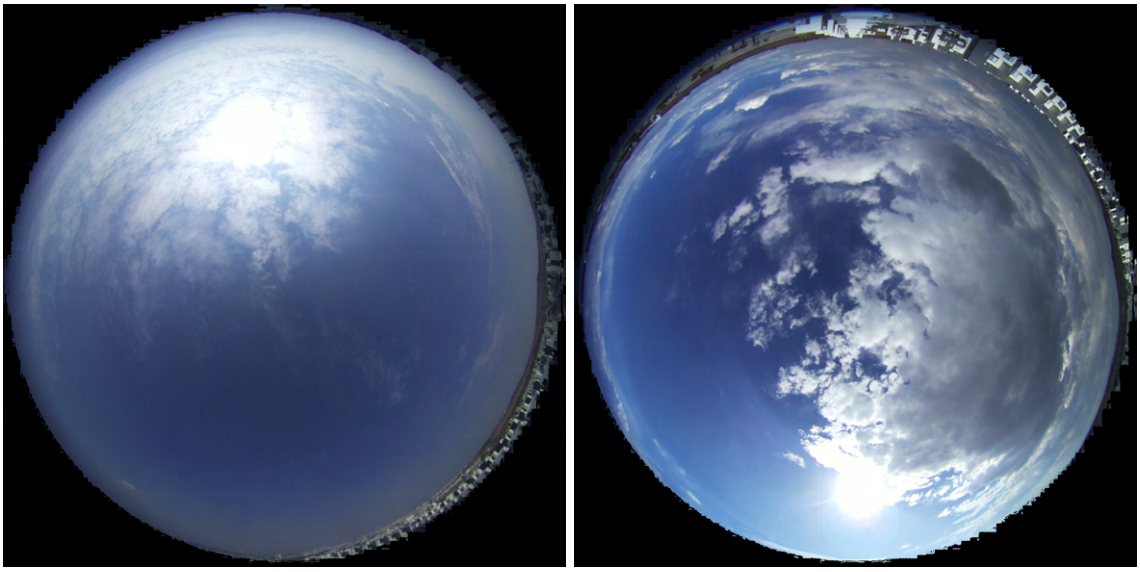


Figura 4.21: Máscara utilizada para eliminar el ruido de las imágenes locales.

Como se observa en la figura 4.22 las imágenes se diferencian únicamente por la información que continen acerca del cielo y las nubes presentes y eliminando la información de ruido fuera del contorno que se veia anteriormente.



(a) Categoría altocumulos. (b) Categoría cumulonimbus.

Figura 4.22: Imágenes pertenecientes al dataset capturado de manera local.

4.6.2. Modelo local con mayor número de imágenes

Después de los resultados presentados en el reporte de la tabla 4.6 se volvió a entrenar el mismo modelo pero esta vez con mayor número de imágenes, mismas que se estuvieron recolectando con el prototipo, estas imágenes ya fueron tratadas con el preprocesamiento mostrado en el apartado 4.6.1.

A continuación se muestra el reporte de clasificación para el segundo entrenamiento. El total de imágenes fue de 1057 y se observa una mejora significativa en los porcentajes de clasificación comparándolos con el primer reporte de la tabla 4.8 con menos imágenes. Se observó una amplia mejoría en las categorías de cielo despejado y stratocumulus con apenas el doble de imágenes aproximadamente, lo cuál indica un buen comportamiento. Por otro lado las categorías como *cumulus*, *stratocumulus* y *cumulonimbus* presentaron los números más bajos debido a que son las categorías con menos número de imágenes de soporte.

Tabla 4.9: Reporte de Clasificación con mayor número de imágenes locales.

Reporte de Clasificación				
	precision	**recall**	**f1-score**	**support**
1_cumulus	0.83	0.56	0.67	18
2_altocumulus	0.86	0.60	0.71	10
3_cirrus	0.85	84	0.84	81
4_cielo despejado	0.84	0.91	0.87	57
5_stratocumulus	0.83	87	0.85	23
6_cumulonimbus	0.70	0.78	0.74	9
7_mixto	0.65	0.79	0.71	14
accuracy			0.82	212
macro avg	0.79	0.76	0.77	212
weighted avg	0.82	0.82	0.82	212

En la imagen de la figura 4.25 se muestra la matriz de confusión del modelo la cuál mostró también un comportamiento deseable. Es importante resaltar que aunque siguen siendo pocas imágenes esta vez fueron suficientes para obtener datos de evaluación de todas las categorías a diferencia del primer entenamiento.

#	Categoría
0	Cumulus
1	Altocumulus
2	Cirrus
3	Cielo despejado
4	Stratocumulus
5	Cumulonimbus
6	Mixto

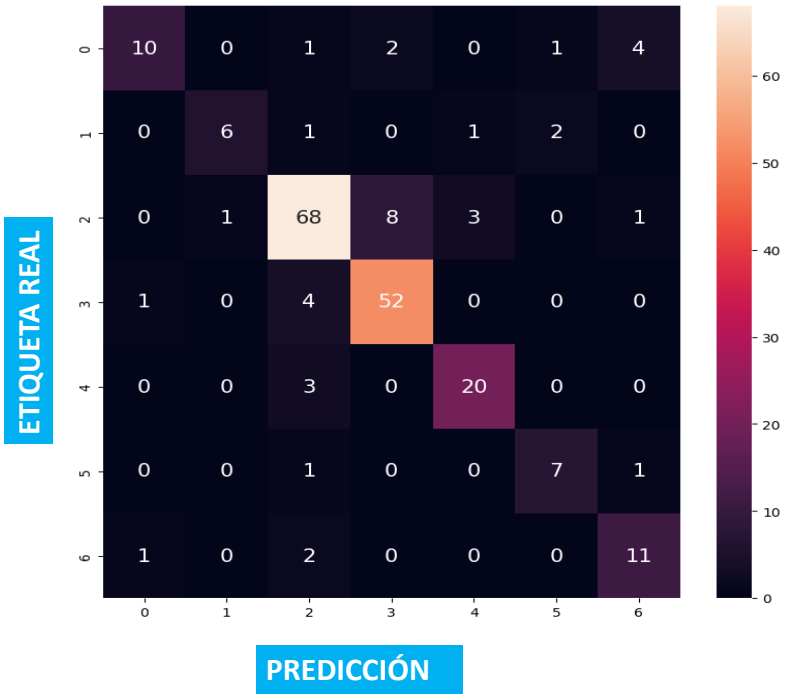


Figura 4.23: Matriz de confusión para el modelo local, segundo entrenamiento

4.7. Sistema de ajuste para el algoritmo genético (CRS)

siguiendo la metodología descrita en 3.4.1 se adaptó el CRS al código del algoritmo genético para ajustar los hiper-parámetros del algoritmo genético el cuál a su vez ajusta los hiper-parámetros de la red neuronal convolucional como se ve descrito en el diagrama de la figura 4.24.

El experimento se corrió bajo los siguientes parámetros del CRS:
maxExecutions $= 5$
populationSize $= 8$
maxParentSize $= 4$
crossoverProbability $= 4$
mutationProbability $= 0.9$

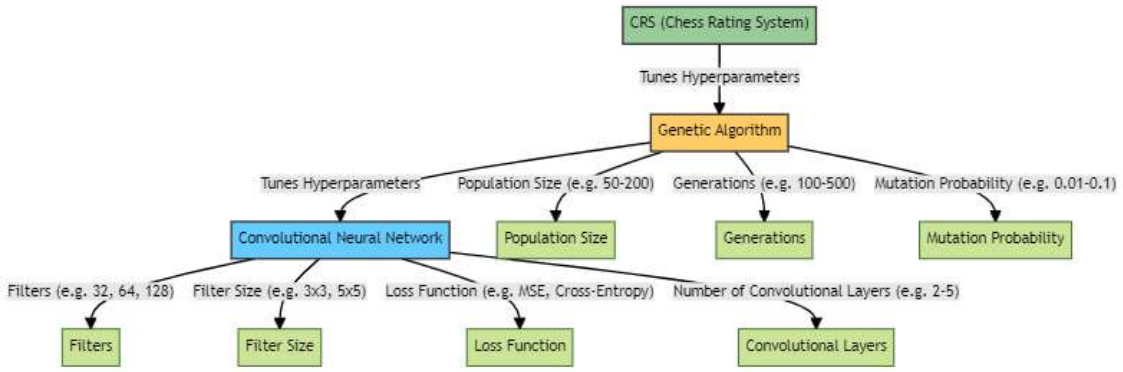


Figura 4.24: Diagrama aproximado del funcionamiento de los algoritmos utilizados.

A continuación se muestra el reporte de clasificación para el segundo entrenamiento, donde se puede apreciar la mejoría que tuvo el clasificador con respecto a los modelos anteriores cuyo mejor porcentaje (*accuracy*) fue de 0.91 y en este caso se logró hasta un 0.98 de *accuracy*:

Tabla 4.10: Reporte de Clasificación para el modelo generado por CRS.

Reporte de Clasificación

	precision	**recall**	**f1-score**	**support**
1_cumulus	0.98	0.98	0.98	264
2_altocumulus	0.95	0.92	0.94	113
3_cirrus	0.99	1.00	0.99	226
4_cielo despejado	1.00	1.00	1.00	229
5_stratocumulus	0.97	1.00	0.99	169
6_cumulonimbus	0.97	0.97	0.97	202
7_mixto	0.97	0.96	0.97	197
accuracy			0.98	1400
macro avg	0.98	0.98	0.98	1400
weighted avg	0.98	0.98	0.98	1400

Finalmente, se muestra la matriz de confusión correspondiente a este modelo. Se observa que las predicciones son casi perfectas, lo cual demuestra que el modelo es muy robusto cuando se entrena con una cantidad suficiente de imágenes.

#	Categoría
0	Cumulus
1	Altocumulus
2	Cirrus
3	Cielo despejado
4	Stratocumulus
5	Cumulonimbus
6	Mixto

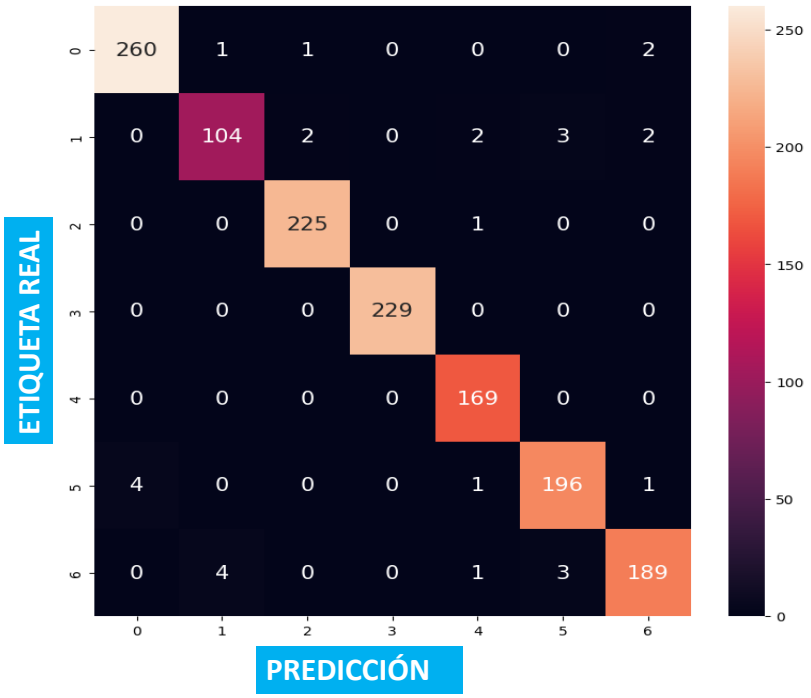


Figura 4.25: Matriz de confusión para el modelo generado con CRS.

4.8. Aporte Tecnológico de la Investigación

El algoritmo clasificador se integrará en un sistema de cómputo portátil, lo que permitirá la creación de un dispositivo portátil como producto final. Este dispositivo tendría la capacidad de generar imágenes y realizar la clasificación en tiempo real y en el mismo lugar.

4.9. Aporte Científico de la Investigación

Dado que la clasificación automática de nubes continúa siendo un tema en desarrollo en el estado actual del arte, esta investigación pretende contribuir con un algoritmo novedoso basado en técnicas de aprendizaje automático que logre una clasificación con una precisión superior al 90 %. Además, se busca que este algoritmo pueda ser implementado en sistemas embebidos o en aplicaciones móviles.

4.10. Aporte tecnológico de la investigación

El algoritmo clasificador estará embebido en un sistema cómputo portátil lo cuál otorga la posibilidad de tener un dispositivo portátil como producto final donde se generen las imágenes y se puedan clasificar en el mismo momento y lugar.

4.11. Impacto Sostenible de la Investigación

Contar con información precisa acerca del tipo de nube, su altura y cobertura genera un impacto positivo en actividades económicas como la generación de energía limpia mediante paneles solares. Una buena predicción de la nubosidad permite una gestión más eficiente y, por ende, una producción mejorada.

No solo las plantas de generación de energía pueden beneficiarse, sino también las instituciones encargadas de observar, describir y registrar los fenómenos atmosféricos a lo largo del tiempo. Contar con información más precisa en sus registros tiene un efecto positivo en actividades dependientes de estos datos. Estas actividades incluyen la explicación de los fenómenos presentes basados en registros pasados, así como la predicción de eventos futuros.

Creemos que el impacto puede ser incluso más amplio, ya que las actividades vinculadas al trabajo de las estaciones meteorológicas abarcan desde aspectos económicos, como la generación de energía y la agricultura, hasta cuestiones de seguridad ante fenómenos naturales o una precisa previsión del tiempo.

En resumen, el impacto sostenible de esta investigación abarca todas aquellas actividades donde el ciclo del agua y la radiación solar ejercen influencia sobre sus procesos.

CAPÍTULO 5

Conclusiones

La clasificación automática de nubes sigue siendo un campo de investigación en evolución debido a su importancia como una de las variables meteorológicas más influyentes en la vida cotidiana de las personas. Las nubes tienen impactos en aspectos como la radiación solar y el ciclo del agua, y también son fundamentales en estudios y pronósticos climáticos. En este trabajo, se ha presentado un clasificador automático de nubes basado principalmente en redes neuronales optimizadas mediante un algoritmo genético.

En la sección de resultados, se ha demostrado el rendimiento de este clasificador optimizado a través de informes de clasificación que incluyen métricas como precisión, recall y valor f, junto con el método de evaluación de la matriz de confusión.

5.1. Comprobación de la Hipótesis

Basado en la revisión de la literatura, se propuso una hipótesis que establece que una red neuronal convolucional puede ser optimizada mediante un algoritmo genético y alcanzar una precisión superior al 90 %. En la sección de resultados se presentan los informes de los modelos generados, los cuales alcanzan e incluso superan la precisión esperada en la hipótesis.

La clasificación automática de nubes es una tarea que pertenece al campo de la visión por computadora, la cual ha ido avanzando con técnicas cada vez más efectivas, en gran medida gracias a la actual revolución de la inteligencia artificial, específicamente el aprendizaje automático y profundo. Sin embargo, estas técnicas pueden llegar a ser muy eficaces pero no necesariamente eficientes.

Tomando como ejemplo las redes neuronales convolucionales, se han propuesto diferentes tipos de ellas con diversas operaciones en sus capas. Algunas son más efectivas que otras, y también se ha demostrado que aumentar la profundidad de las redes contribuye a mejorar su rendimiento. No obstante, en entornos como los sistemas embebidos o sistemas con recursos limitados, estas opciones pueden resultar incompatibles debido a la cantidad de memoria y poder de cómputo necesarios para realizar más operaciones. Como se muestra en la tabla 5.1, métodos como el ensamble de dos redes preentrenadas pueden ser muy efectivos. Sin embargo, al comparar la cantidad de parámetros (pesos) de la red, la diferencia es diez veces mayor en relación con los modelos presentados en este trabajo, lo que hace que estos últimos sean más eficientes.

Tabla 5.1: Comparación de distintos modelos de clasificación de nubes en cuanto a número de parámetros de la red

Modelo	Autor	Tamaño en # de parámetros		
		Entrenables	No entrenables	Total
Modelo 2	propio	671,287	5,339,712	6,010,999
Modelo 3	propio	411,191	3,787,808	4,198,999
Fusión de Resnet_50 y otra subred densa	[12]	25,583,592 +	53,120 +	25,636,712 +
Ensamble de redes Resnet-50 e InceptionV3	[23]	49,400,944	87552	49,488,496
Votación de varias redes Resnet, Alexnet, Googlenet, entre otras	[35]	110,683,376	53120	110,736,496

En el modelo donde se hace una fusión entre una Resnet-50 y otra subred de capas totalmente conectadas se agrega el signo + en la cantidad de parámetros dado que solo se tiene el dato de los parámetros de la Resnet-50.

5.2. Prototipo de Instrumento

Se han propuesto varios artículos y algunas tesis con el mismo o similar objetivo al de esta tesis, pero la mayoría de ellos se quedan en la parte de mostrar los resultados obtenidos sobre alguno de los datasets que están en la misma literatura. En este trabajo, se pretende darle una utilidad a los resultados encontrados, es decir, al clasificador, y por esto se decidió probar el funcionamiento del modelo optimizado clasificando nubes en tiempo real.

Los principales hallazgos de esta tarea, los cuales no solo arrojan buenos resultados, sino que también nos permiten vislumbrar el futuro y señalar algunas áreas de oportunidad, se enuncian a continuación:

- A pesar del poco tiempo disponible para la recopilación de imágenes, las obtenidas mostraron una buena similitud con las que se tenían del dataset MGCD. Esto significa que si el modelo es capaz de clasificar correctamente en el dataset, también lo será con las imágenes locales.
- El prototipo generado tiene un costo muy bajo en comparación con algunos modelos comerciales [36] que cumplen la función de tomar fotografías del cielo completo, pero que no etiquetan las imágenes capturadas.
- La metodología produce buenos resultados, obteniendo una arquitectura más 'ligera' que el estado del arte actual y con la precisión proyectada en la hipótesis.
- Como se mencionó en la sección de resultados, el dataset de imágenes locales necesita ser enriquecido para mejorar el rendimiento del clasificador. Viéndolo como trabajo a futuro y considerando que en México existen las cuatro estaciones durante periodos de tiempo similares y el clima varía durante estos lapsos, nos motiva a proponer como trabajo futuro el seguir capturando imágenes y obtener una base de datos con la calidad y cantidad suficientes para continuar con la investigación.

- Al concluir este trabajo, las personas involucradas pertenecen completamente al área de ingeniería, específicamente de instrumentación y control automático, así como de inteligencia artificial. Esto plantea otra dirección para el trabajo futuro, que es abrir el proyecto a otras disciplinas relacionadas con el estudio del clima donde pueda ser útil y encontrar una aplicación significativa. Por ejemplo, durante la clasificación de las imágenes obtenidas, que se llevó a cabo siguiendo una guía publicada por un organismo profesional en el estudio de las nubes, surgieron ciertas incertidumbres debido a zonas grises en la definición de ciertas categorías que comparten similitudes. Esta tarea podría ser abordada de manera más efectiva por un experto en el estudio del clima o de las nubes.

5.3. Algoritmo de Ajuste *CRS*

El algoritmo de ajuste *CRS* utilizado mostró tener excelentes resultados, los cuales ya hemos descrito en el capítulo anterior. A partir de ello, podemos obtener las siguientes conclusiones:

- Como es sabido, entre más imágenes se tengan en el dataset, el modelo tendrá mejores resultados. Por lo tanto, el reporte de clasificación mostrado tiene un buen índice de confianza, el cual está respaldado por el número de imágenes de soporte, el cual es suficientemente grande en cualquiera de las categorías.

- Una de las conclusiones más importantes podría ser la mejora obtenida en la precisión de la red neuronal al agregar *CRS* para ajustar el algoritmo genético. Aunque el costo computacional aumenta considerablemente al aumentar en un orden el número de iteraciones para obtener un modelo de red neuronal, este costo se paga solo una vez durante el entrenamiento. Al final, el resultado es una red neuronal con un tamaño en términos de número de capas y parámetros que no difiere significativamente de los modelos obtenidos anteriormente. Después de esto el modelo obtenido es ligero y puede utilizarse para hacer inferencias sin exigir demasiados recursos. Además, la precisión es incluso del 100 % en algunas categorías.

5.4. Productos obtenidos

Durante el desarrollo de esta investigación, se publicaron dos artículos relacionados con el tema de la clasificación de nubes. El primero consiste en una revisión sistemática acerca de los trabajos más emblemáticos publicados sobre las metodologías y técnicas utilizadas para la detección y clasificación de nubes. El segundo artículo aborda el problema de cuando varios tipos de nubes aparecen en una misma fotografía y solo se puede asignar una etiqueta, la cual, como ya se observó en este trabajo, es la categoría "mixta". Por lo tanto, en este artículo se propone utilizar una técnica de detección de objetos como *YoloV3* para poder asignar etiquetas a cada uno de los objetos (en este caso, nubes) que aparecen en una sola imagen.

5.4.1. Methodologies for Ground-based cloud classification: A review

Artículo publicado en el XVIII Congreso Internacional de Ingeniería (CONIIN), celebrado en la Facultad de Ingeniería de la UAQ, el 16 de Mayo de 2022.

5.4.2. Detección de Objetos para la Clasificación de Nubes en Imágenes de Múltiples Instancias

Artículo publicado en el marco del XVI Coloquio del Posgrado de la Facultad de Ingeniería de la UAQ, realizado el 16 de Noviembre de 2022.

Methodologies for ground-based cloud classification: a review

Edgar Vega Maya
División de Investigación y Posgrado Facultad de Ingeniería
Universidad Autónoma de Querétaro
Querétaro, México
evega17@alumnos.uaq.mx

Juvenal Rodríguez Résendiz
División de Investigación y Posgrado Facultad de Ingeniería
Universidad Autónoma de Querétaro
Querétaro, México
juvenal@uaq.edu.mx

***Abstract*— Clouds are among the most significant meteorological phenomena that affect our lives, as they contribute significantly to the earth's energy balance by absorbing and distributing solar radiation. In addition, they are essential for weather analysis and forecasting. However, ground-based cloud observation mainly depends on human observers, causing uncertainties since of subjective judgments. Therefore, research in automatic cloud observation has increased and is still in progress. The principal goal of this study is to explore the various techniques and methodologies that have been proposed for cloud classification focusing only on those based on ground-based cloud images and excluding research on satellite images. Articles over the last ten years were studied. This work initiates with a brief theoretical framework about cloud types, cloud classification, and imaging systems. Then a detailed description of the methodologies used for classification is presented. Convolutional neural network-based algorithms are among the most precise approaches, reaching accuracies of 99% under certain conditions.**

Keywords— automatic cloud classification; ground-based cloud images

I. INTRODUCTION

There are several areas in which cloud data is quite valuable, such as predicting direct normal irradiance (DNI). The solar irradiance that reaches the earth's surface has a direct influence on the output of solar power generation. Estimating local DNI is usually necessary to obtain a high concentrating solar power (CSP) accuracy[1]. Hence, many works have been proposed for DNI prediction, including cloud classification, cloud detection, and cloud motion [2]–[4].

Another dimension where cloud data can be very important is precision agriculture (PA) since clouds are involved in the rain and the solar radiation necessary for crops. PA is an approach to farm management that uses information technology to ensure that crops and soil receive exactly what they need for optimum health and yield [5]. Furthermore, PA is evolving towards autonomous management systems, i.e., systems that do not require interaction or supervision by humans. These autonomous systems use remotely sensed imagery, and the presence of clouds impacts this data by varying illumination and reducing image quality and radiometric accuracy [6]. Likewise, cloud base height (CBH) is another variable in meteorology that is of particular interest in many applications. Either directly or indirectly, CBH data is helpful in different areas such as: validating and improving climate models [7], aviation for air traffic controllers [8], and solar power applications [9]. The use of solar technologies, especially for electricity production, has increased during the past few years. Domestic and massive solar parks are an example of the growth mentioned above. Yet, they have failed to attain broad acceptance compared to traditional methods, primarily because of their output production variability [10]. In this case, the CBH and other cloud-related parameters are useful for predicting the solar budget for the next few seconds to a few hours (nowcasting). Such nowcasts can help to reduce supply uncertainty from photovoltaic plants and contribute to an appropriate balance of energy supply and demand [11].

Given the importance of clouds in the applications mentioned above and the difficulties in detecting their behavior and contribution, technologies for automatic and continuous cloud features description are needed. Ongoing research has been developed to improve cloud classification and CBH measurement since they are designed to provide vital information to those applications related to cloud data. Until this century, human observers were responsible for determining cloud data, such as type and coverage. Nonetheless, the growing demand for precise measurements encouraged researchers to develop computer-based techniques for identifying clouds and providing information on their classification. These algorithms range in complexity from simple threshold or rule-based ones to more sophisticated machine learning and deep learning algorithms.

This paper synthesizes the works proposed in the last ten years for these two essential duties cloud classification and CBH measurement. In this study are considered exclusively papers that use cloud images captured by ground-based camera systems, excluding those based on satellite images. More specifically, this paper aims to answer the following questions by conducting a methodological review of existing research:

- What are the main practical motivations behind cloud classification and CBH measurement?

- What are the existing techniques and methods to enable cloud classification and CBH measurement? In addition, what is the method with the best accuracy?
- What should the focus of future cloud classification research be?

The following is the structure of this research: Section II provides a theoretical framework for types of clouds and imaging systems. Then section III consists of a detailed description of the methods and technologies used for cloud classification. Finally, the paper is concluded in section IV.

II. Theoretical Framework

This section is divided into three subsections; the primary purpose is to briefly introduce the theory involved with techniques and algorithms used, types of clouds and their properties, and imaging systems.

A. Methods and Algorithms

As shown in Fig. 1, a typical cloud image processing strategy consists of three significant steps: preprocessing, feature extraction, and classification. Various techniques and methods have been proposed in each stage, from statistical to deep learning models. Some examples of these are shown below.

1) Pre-processing

Preprocessing is mainly used to obtain specific image forms for feature extraction; this stage includes converting RGB to gray-scale images, image segmentation methods, and image enhancement methods. High-pass filtering is used for image enhancement; it emphasizes fine details in the image. This process consists of a convolution (1) between the image and a kernel $\boldsymbol{\omega}$ (2).

$$g(x,y) = \omega * f(x,y) \tag{1}$$

where $g(x,y)$ is the filtered image, $f(x,y)$ is the original image and ω is the filter kernel.

$$\boldsymbol{\omega} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & -18 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{2}$$

2) Feature extraction

The ability to extract representative features of clouds is crucial for successful ground-based image cloud categorization. In the literature, a variety of feature extraction strategies have been explored, ranging from hand-crafted descriptors such as census transform histogram (CENTRIST), local binary patterns (LBP), scale-invariant feature transform (SIFT) to those based on a deep learning framework. Convolutional neural network (CNN) models have demonstrated high performance for feature extraction. CNNs are intended to function with grid-structured inputs and have significant spatial relationships in local areas. CNNs work similarly to classic feed-forward neural networks, except that the computations in their layers are spatially structured, and the connections within them are sparse[12]. The main kind of layers that generally appear in CNNs are *convolution, pooling, and fully connected layers*, as shown in Fig 2.

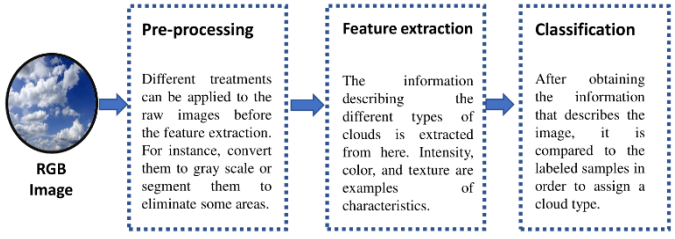


Fig. 1. Typical workflow for cloud image classification

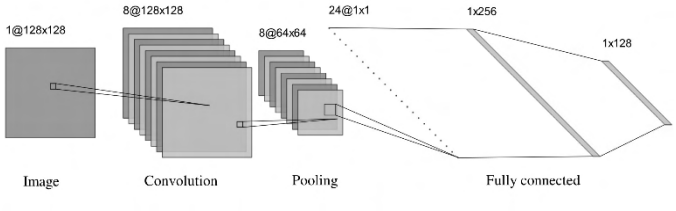


Fig. 2. Example of a CNN architecture.

In the case of convolutional layers, a filter or a series of them are utilized to recognize features in every spatial location. The filter overlaps the image in every possible position and performs a convolution between the weights in the filter and the pixel values in the image. As a result, a feature map is generated, and the equation that defines this operation is described in (3), where $s(i,j)$ is the feature map, I corresponds to the image, and K is the filter.

$$s(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m,n) \tag{3}$$

The definition presented in (3) corresponds to the cross-correlation operation. However, in the framework of neural networks, the operation carried out in these layers is known as convolution.

3) Classification

Once the characteristics of an image have been extracted, a classifier receives them and identifies the cloud type. Different classifiers have been proposed, even though this step has received less attention because most researchers have focused on feature extraction. Among the most frequently used are the k-nearest neighbor (k-NN), support vector machines (SVM), and fully connected layers of CNN. Let suppose a training set has n dimensional feature vectors $\boldsymbol{x_i} \in R^n$, and the corresponding label $y_i \in \{-1,1\}$. Then, the SVM solves the optimization problem in (4).

$$\min_{w,b,\xi} \frac{1}{2}\boldsymbol{w^T w} + C\sum_{i=1}^{l}\xi_i$$
$$\text{subject to} \quad y_i[\boldsymbol{w}^T\phi(x_i) + b] \geqq 1 - \xi_i, \qquad (4)$$
$$\xi_i \geqq 0.$$

where $\boldsymbol{x}_i$ is mapped into a higher dimensional space by means of $\phi(\boldsymbol{x}_i)$ to perform nonlinear classification, and $C > 0$ is the penalty error parameter. Moreover, $K(\boldsymbol{x}_i, \boldsymbol{x}_j) \equiv \phi(\boldsymbol{x}_i)^T\phi(\boldsymbol{x}_j)$ is the kernel function [13].

B. Types of clouds

According to the World Meteorological Organization [14], a cloud is defined as a hydrometeor consisting of tiny particles of liquid water or ice suspended in the atmosphere and usually not touching the ground. Despite studies on cloudiness, it is commonly acknowledged that there is still much to learn about this phenomenon. The combined effects of various types of cloud cover provide a high element of uncertainty, resulting in a significant factor of doubt in climate models and forecasts.

The International Cloud Atlas (ICA) is the single authoritative and most comprehensive reference for identifying clouds. Today, the ICA recognizes ten primary cloud genera classified according to their vertical location in the atmosphere and their relative morphology. The shape, structure, transparency of a cloud, and the layout of its content, are used to further subdivide the ten genera [14]. In general, clouds are divided according to their base height on the Earth´s surface, as shown in Table I, and cloud names are made up of Latin prefixes and suffixes that, when combined, reveal the nature of it.

- Stratus: flat, layered, and smooth
- Cumulus: heaped up, puffy
- Cirrus: feathery, wispy
- Nimbus: rain-bearing
- Alto: mid-level (even *alto* is Latin for high)

TABLE I. CLOUD CLASSIFICATION

	Genus	**Height (m)**
High-level	Cirrus Cirrostratus Cirrocumulus	> 5000
Middle-level	Altocumulus Altostratus Nimbostratus	[2000 ,7000]
Low-level	Stratus Stratocumulus Cumulonimbus Cumulus	< 2000

Table I presents a brief description of ten genera according to their appearance and some effects they can produce. Finally, in Fig. 3, the ten genera are illustrated concerning their height in the sky.

TABLE II. CLOUD DESCRIPTION

Genus	**Code**	**Description**
Cirrus	Ci	They are the tallest of the ten different kinds. These clouds have a fibrous appearance and are formed of tiny ice crystals.
Cirrostratus	Cs	A transparent, white cloud veil covering the sky entirely or partially, causing halo effects.
Cirrocumulus	Cc	Although they do not bring rain, but their presence may indicate stormy weather in the coming days.
Altocumulus	Ac	Groups of globular clusters or layers of white to gray tone are common. In aviation, this is a crucial class to keep an eye on.
Altostratus	As	It is formed by a layer of grayish or grayish-blue clouds, striated or uniform in appearance. Thick enough to obscure the sun, so there is no halo around it.
Nimbostratus	Ns	They have the appearance of a thick, dark gray cover with steady rain or snow falling but no significant thunderstorms.
Stratus	St	Flat, gray to white cloud, related to light rain, and it has a significant effect on solar radiation.
Stratocumulus	Sc	Big dark round masses and usually in large groups. They may be found in all kinds of weather and rarely produce rain.
Cumulonimbus	Cb	With the shape of a mountain and significant vertical growth. And almost always produce storms, in the form of rain or/and hail.
Cumulus	Cu	Puffy, cotton-like shapes often appear on sunny days and may bring showers.

C. Imaging systems

Most cloud observation applications can be divided into those that use satellite-based images and those that use ground-based images. The main distinction between ground-based over satellite-based monitoring is that the first is suitable for local

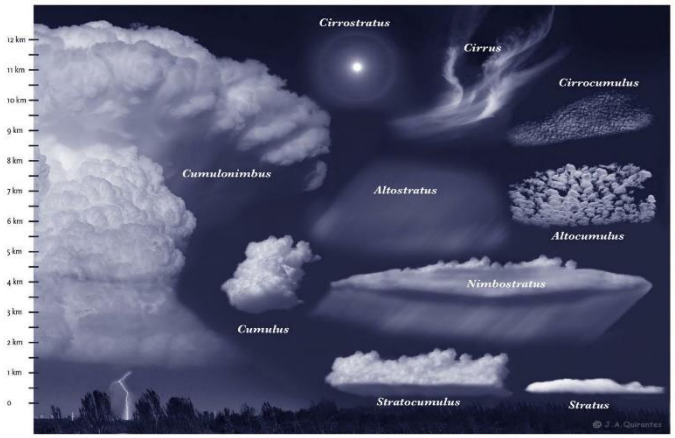


Fig. 3. The ten cloud genera in relation to their height in the sky [15].

monitoring and obtaining base-cloud height and cloud type. On the other hand, satellite-based data has a limited geographical and temporal resolution, resulting in certain ambiguities when calculating cloud properties. The applications and a more detailed review of both benefits and drawbacks of satellite and ground observation are presented in [16].

In recent years, different ground-based sky imaging technologies have been developed and deployed to acquire cloud coverage and type, principally from the visible and infrared spectrum. The Total-Sky Imager (TSI) [17] is one of the most popular, which basic design consists of a curved mirror with a camera mounted on it to achieve a total sky view, as shown in Fig. 4, and it generates 352 x 288 images with a 24-bit JPEG. The TSI captures sky images during daylight only, and the sampling rate can be set up to ten seconds. This technology has been commercialized, and one model available under the name TSI-880 can be found in [18].


Fig. 4. Total Sky Imager TSI-880.

The Whole Sky Imager (WSI) [19] is another ground-based imager; the principal difference with the TSI is that it uses a camera with a fisheye lens instead of a mirror. The WSI can collect high-resolution data in the visible and near-infrared spectrum (450nm - 800nm), and it can do so at any time of day or night. The interval acquisition is user-selected, ranging from one minute to ten minutes.

The All-Sky Imager (ASI) was presented by [20] as an instrument for cloud cover assessment; it is based on a charge-coupled device (CCD) camera, which is adapted to an environmental housing with a solar-shadow system as shown in Fig. 5. A commercial fisheye lens provides a 180 degrees field of view (FOV). As a result, the image captured includes the entire skydome. The solar-shadow system is intended to block the rays of the Sun.


Fig. 5. All-Sky Imager presented in [20].

Other instruments for cloud monitoring have been proposed in the literature, such as the Whole-Sky Camera (WSC)[21] and the Total-Sky Imager (TCI) [22]; however, the underlying idea is the same for all of them. Finally, in Fig. 6, is presented an example of an image taken by a ground-based cloud imager.

III. CLOUD CLASSIFICATION

Over the last ten years, cloud classification has been widely researched. To perform an exhaustive search and analysis of the most representative articles, it has been used academic search engines such as IEEE, Scopus, and Web of Science, as well as conventional search engines such as Google Scholar; the results were filtered according to the following criteria:

- Algorithms using ground-based images
- Articles from the last ten years
- Excluding articles whose main objective is cloud segmentation, detection, or tracking.

The work presented in [23] developed an algorithm called multiple random projections (MRP), which takes advantage of the benefits of reducing dimensionality. The images are divided into patches, and each patch corresponds to a high dimensional vector that is reduced for clustering feasibility. After MRP is performed, further steps are clustering, calculating cluster centers called *textons* (discriminative features are extracted here), histograms for training are obtained, and finally, nearest neighbor (NN) is used for classification. This article used two sets of images: the Kiel database, provided by Kiel University in Germany, and the IapCAS-E database, with 1500 and 2000 images, respectively.

According to [24], until this point, there are no reliable methods for classifying cirrus clouds due to the influence of sunlight which is nonuniform. Therefore, this paper proposes a method to solve this named background subtraction, adaptive threshold (BSAT). The background subtraction is used to

Fig. 6. Ground-based image classified as 'Altocumulus' in MGCD dataset collected in [25].

overcome the illumination problem and detect cirrus clouds in visible images. The images utilized in this work were selected from two different instruments, a TCI and an ordinary camera obtaining images under different conditions. This paper focuses mainly in performing a robust detection of thin cirrus clouds as a previous step for further classification.

In [26], the authors proposed a cloud classification approach, proposing the color census transform (CCT). Census transform is an operation for extracting texture features and local and global structure descriptors from grayscale images. Since cloud images have three color channels and some grayscale conversion methods lose information, the authors proposed using opponent color space and applying the census transform to each component. The Census transform compares the center pixel against its neighbors and sets a bit to 1 or 0 for each check, depending on whether the center pixel is greater or not. The bits are ordered to form a number representing the census transform for the center pixel. An example is given in (5). After extracting characteristics, several experiments were performed to prove the classification accuracy. Support vector machine outperformed was the method with the highest accuracy over neural network models and k-NN.

$$\begin{matrix} 30 & 90 & 90 \\ 30 & 80 & 90 \\ 30 & 80 & 80 \end{matrix} \Rightarrow \begin{matrix} 1 & 0 & 0 \\ 1 & & 0 \\ 1 & 1 & 1 \end{matrix} \Rightarrow (10010111)_2 \Rightarrow \mathrm{CT} = 151. \tag{5}$$

Until now, and as seen in the review presented in [16], which synthesizes the work proposed before 2013, most researchers only consider the classification task over one single image. In contrast, a technique for extracting temporal information from a sequence of images and treating them as dynamic texture (DT) is developed in [27]. A linear autoregressive moving average (ARMA) model is used to process the sequences. The novelty of this paper is their model named Tensor Ensemble of Ground-based Cloud Sequences (eTGCS), which can be modeled as a time function; therefore, a transfer function and an output equation can be obtained. This strategy maps GCSs into feature vectors that are directly classified with a multiclass support vector machine. The experimental results were carried out on the IapCAS-E dataset.

Ref. [28], the authors argued that critical challenges for cloud classification were yet to be addressed. For example, feature extraction may not be discriminative enough to characterize the images accurately, resulting in incorrect categorization. To overcome this drawback, they present a novel approach based on learning group patterns (LGP) for cloud representation. These learning group patterns are descriptors that cascade the salient local binary pattern (SLBP) information to consider texture resolution differences. Pyramid representation of local binary patterns has been more efficient than conventional linear binary pattern (LBP) descriptors.

In [29], a new feature extraction technique is implemented. The proposed method, known as Multiview cloud (mCLOUD), extracts raw descriptors from texture, structure, and color views in a densely sampled way, as seen in Fig. 7. More specifically, the authors utilize the scale invariant feature transform (SIFT) to extract texture characteristics. Likewise, the census transform histogram (CENTRIST) focuses more on the rough structure features, and finally, statistical techniques are used for color features. The Fisher vector encoding (FV) is used to encode the raw descriptors. The feature aggregation procedure follows FV. To determine the final cloud image categorization result, an SVM is applied. The experiments to validate mCLOUD were carried out over the HUST dataset consisting of 1231 images divided into six classes. The results show an improvement in the accuracy of classification compared to [26]. Nonetheless, the method is computationally and memory expensive.

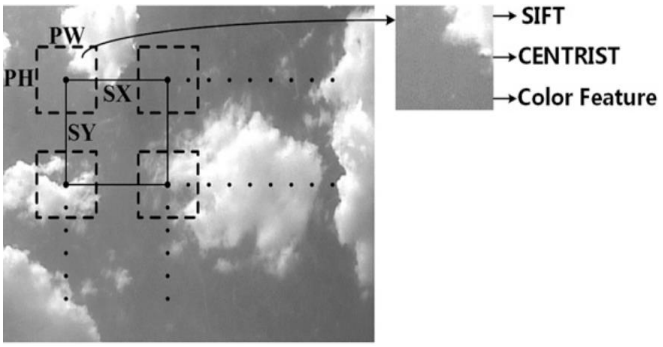


Fig. 7. The densely sampled feature extraction [26].

In [30], the authors developed a methodology where the integration of a TSI and a ceilometer in parallel is used for automatic cloud classification. The ceilometer provides heigh and thickness (Cloud penetration depth) information regarding cloud types (CBH and CPD) that can assist in distinguishing between clouds that appear to be identical. In this way, twelve features are extracted from the images and six more features from the ceilometer. The next step is to use the feature vector as input of a random forest (RF) algorithm for classification.

The dataset used for experiments consists of 717 TSI images and the corresponding CBH and CPD data from the ceilometer. It is worth noting that the images were first preprocessed with a mask applied to detect non-sky-related objects. A projection is applied to the images based on procedures in [31] to convert them from spherical to rectangular form. Results show an improvement when the ceilometer information is used together with the image information; the difference in accuracy obtained when adding the ceilometer data is around 15.72%.

Ref. [32], the authors proposed a new method based on duplex norm-bounded sparse coding (DNSC). Norm-bounded is used for encoding features, but it has also been proven to be an efficient classifier because it considers local and sparse characteristics, improving the discriminative power for classification. The general pipeline of the method is as follows: local descriptors are extracted, then a duplex norm-based sparse coding is performed in two stages: the first is feature extraction for complete representation, and the second is classification. The total-sky cloud image set (TCIS) was used to evaluate the proposed algorithm, which has five cloud classes, with 1000 images each. The algorithm was implemented in MATLAB, and the time for classification of one image (821 x 821 pixels) was approximately 0.32 seconds.

In [33], the authors first include deep convolutional neural networks (CNN) as a technique for feature extraction (DeepCloud). According to the authors, the classification must be approached as a multi-class problem. Consequently, using only CNN for feature extraction and classification will not guarantee success. Thus, the authors proposed the following procedure: images are fed into a CNN model for deep feature extraction, then the results are filtered via pattern mining, and after that, the selected descriptors are encoded using FV to generate the cloud image representation; finally, an SVM is used for classification. The dataset used for testing DeepCloud is collected in [26], consisting of nine classes, each ranging from 59 to 200. Since DeepCloud performs the cloud characterization under the learning paradigm instead of hand-crafted operations, it makes it easier to extract high discriminative power representation. However, the memory consumption is relatively massive.

Ref. [34], the authors proposed a new method named deep multimodal fusion (DMF). The classification problem is attempted not only by using images but also by adding other sensors that provide information about the cloud type. A CNN model is trained to obtain the visual features, then visual and multimodal cloud features are combined, and a classification model is trained using an SVM. The experiments were conducted on the multimodal ground-based cloud (MGC) dataset. It consists of seven cloud classes ranging from 160 to 350 images per class. A weather station was also used to capture multimodal data simultaneously as the image was taken. The method was tested and compared with other state-of-the-art methods, such as completed LBP (CLBP). DMF has a higher accuracy of at least 16 percent, according to the results.

In [35], a cloud classification method based on infrared images is developed. Infrared image feature extraction is accomplished by first using symmetric matrix manifolds, which are then projected into a feature vector. A support vector machine is used for classification with a simple linear kernel function. The proposed model was validated on 500 images (240 x 320 pixels) divided into five classes with 100 images each. The results were compared with those achieved in [36], and the proposed model shows a higher accuracy.

In [37], another multimodal approach is proposed, the joint fusion convolutional neural network (JFCNN) method, which contains the vision subnetwork and multimodal subnetwork. The joint fusion layer is a new stage introduced to the network with the goal of learning two types of cloud characteristics in the same framework. Once the network is trained, all the extracted features (multimodal and visual) are integrated using a weighted strategy. The flowchart of this method is shown in Fig. 8. The ResNet-50 model is adopted as a vision subnetwork, and it was pretrained on the ImageNet dataset for further fine-tuning on cloud images. The dataset utilized was the MGC dataset which consists of images and weather variables. The images were resized to 256 x 256, and the multimodal information (weather variables) was concatenated into a six-dimensional vector. Seven classes were considered for classification. To prove the effectiveness of the method, it was compared with other seven techniques, namely BoW model, PBoW model, LBP, CLBP, PBoW+CLBP. The results show that JFCNN outperforms all the previous methods.

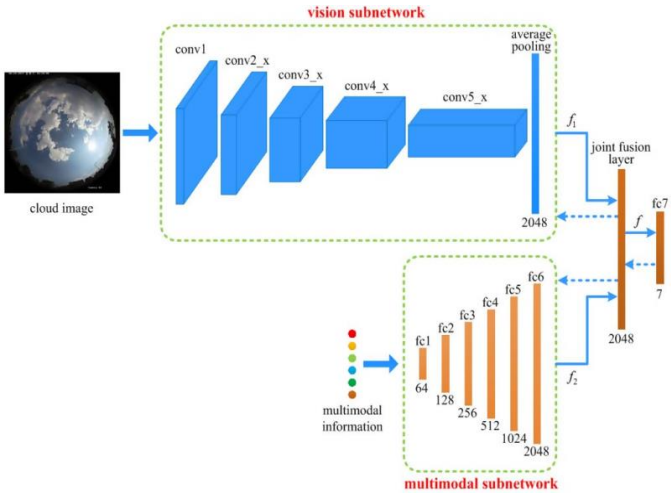


Fig. 7. The main flowchart of JFCNN [37].

In [38], the authors suggested that intrinsic correlations between multimodal information and visual information cannot be mined sufficiently. Accordingly, they proposed a new hierarchical multimodal fusion (HMF) approach for ground-based cloud classification in weather station networks. The highlight of this method is that it divides the fusion of

information into two levels: low-level fusion, on the one hand, concatenates deep visual and deep multimodal data to produce fused features that are modality-specific. Whereas high-level fusion combines the output of the low-level fusion and the heterogeneous features. Since the hierarchical fusion results in a deeper model, compared with the traditional multimodal methods, the semantic information mined from the features is more representative. The dataset used was the MGCD dataset which contains 8000 cloud images of seven classes and the multimodal information. Finally, it is worth noting that the training step used only one loss function to represent the cloud types more completely.

Ref [39], the authors considered the case when occlusion is present in the images and proposed a novel method that is more robust in this case. The structure of the process consists of convolutional neural networks for feature extraction and weighted sparse representation for classification. An ensemble of two neural networks is proposed for extracting features. The models used for feature extraction were the Res-Net50 and the Inception V3 network. The experiments were carried out on the MGCD dataset; although the multimodal information was not used, it was based only on the ground-based images. Several experiments were carried out with different percentages of occlusion in images. It was compared with other approaches using only one neural network for feature extraction and classification, adding weighted sparse representation for classification, and adding two neural networks.

Finally, Table III presents a comparison of the most significant works in cloud classification. The state-of-the-art is illustrated by extracting parameters such as accuracy, number of classes, and dataset used from the articles.

IV. Conclusions

Methodologies and techniques for cloud classification have been reviewed; convolutional neural networks, support vector machines, and hybrid models that include both, are among the most widely used. The approach with the highest accuracy is the CNN ensemble (99 %). However, it is hard to make a fair comparison of the methods because the conditions of the datasets utilized in most of the works are different. One of the most interesting details of this review is that the total of papers reviewed mention several areas where cloud classification is important. Yet, there are no data or results where the classification is used in a specific task. The main reason for this is that after training and evaluating the algorithm on the dataset, the authors do not further evaluate it on real-time images. Future work should be proposed to fill this blank. Additionally, most of the existing datasets were obtained with a commercial sky imager that is limited to taking images, and some versions can automatically provide cloud coverage and height. Still, there is no commercial instrument to obtain the cloud type. The last idea suggests that it should be of great importance to test these classification algorithms on embedded systems, which could make possible the idea of a portable instrument; one proposal could be the use of general-purpose microcontrollers for this function or small board computers such as the NVidia Jetson nano that is suitable for intelligent algorithms such as neural networks.

Reference	Algorithm/technique	Dataset	Classes	Accuracy (%)
[23]	Multiple Random Projections	Kiel IapCAS-E	7 7	93.8 88.7
[26]	CCT SVM	Images taken with a sky camera (60° FOV) and a servo motor	6	79.8 ± 1.2
[27]	eTGCS	IapCAS-E	7	92.31
[28]	LGP	Kiel IapCAS-E	7 7	83.21 78.94
[29]	mCLOUD	HUST	6	81.2
[30]	RF	Data obtained from a TSI-880 and a ceilometer	7 10	71.7 71.1
[32]	DNSC	TCIS	5	97.61 ± 0.92
[33]	DeepCloud	[26]	9	86.7
[34]	DMF	MGC	7	86.3
[35]	SVM	Images taken with a WSI	5	82.2
[37]	JFCNN	MGC	7	93.37
[38]	HMF	MGCD	7	87.9
[39]	CNN ensemble	MGCD	7	99.8

REFERENCES

[1] Y. Chu and C. F. M. Coimbra, "Short-term probabilistic forecasts for direct normal irradiance," *Renewable Energy*, vol. 101, pp. 526–536, 2017.

[2] C.-L. Fu and H.-Y. Cheng, "Predicting solar irradiance with all-sky image features via regression," *Solar Energy*, vol. 97, pp. 537–550, 2013.

[3] H.-Y. Cheng and C.-C. Yu, "Multi-model solar irradiance prediction based on automatic cloud classification," *Energy*, vol. 91, pp. 579–587, 2015.

[4] H.-Y. Cheng, C.-C. Yu, and S.-J. Lin, "Bi-model short-term solar irradiance prediction using support vector regressors," *Energy*, vol. 70, pp. 121–127, 2014.

[5] M. A. Oliver, "An overview of geostatistics and precision agriculture," *Geostatistical applications for precision agriculture*, pp. 1–34, 2010.

[6] J. M. P. Czarnecki, S. Samiappan, M. Zhou, C. D. McCraine, and L. L. Wasson, “Real-Time Automated Classification of Sky Conditions Using Deep Learning and Edge Computing,” *Remote Sensing*, vol. 13, no. 19, p. 3859, 2021.

[7] M. Costa-Surós, J. Calbó, J. A. González, and J. Martin-Vide, “Behavior of cloud base height from ceilometer measurements,” *Atmos Res*, vol. 127, pp. 64–76, 2013.

[8] K. Khlopenkov, D. Spangenberg, and W. L. Smith Jr, “Fusion of surface ceilometer data and satellite cloud retrievals in 2D mesh interpolating model with clustering,” in *Remote Sensing of Clouds and the Atmosphere XXIV*, 2019, vol. 11152, p. 111521F.

[9] M. Sawant, M. K. Shende, A. E. Feijóo-Lorenzo, and N. D. Bokde, “The State-of-the-Art Progress in Cloud Detection, Identification, and Tracking Approaches: A Systematic Review,” *Energies (Basel)*, vol. 14, no. 23, p. 8119, 2021.

[10] R. Tapakis and A. G. Charalambides, “Equipment and methodologies for cloud detection and classification: A review,” *Solar Energy*, vol. 95, pp. 392–430, 2013.

[11] A. Kaur, L. Nonnenmacher, H. T. C. Pedro, and C. F. M. Coimbra, “Benefits of solar forecasting for energy imbalance markets,” *Renew Energy*, vol. 86, pp. 819–830, 2016.

[12] C. C. Aggarwal, “Neural networks and deep learning,” *Springer*, vol. 10, pp. 973–978, 2018.

[13] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995, doi: 10.1007/BF00994018.

[14] “World Meteorological Organization,” *https://public.wmo.int/en/WorldMetDay2017/resources*, 2017.

[15] “Meteoglosario Visual,” *https://meteoglosario.aemet.es/es/termino/464_genero-nuboso*, 2018.

[16] R. Tapakis and A. G. Charalambides, “Equipment and methodologies for cloud detection and classification: A review,” *Solar Energy*, vol. 95, pp. 392–430, 2013.

[17] C. N. Long and J. J. DeLuisi, “Development of an automated hemispheric sky imager for cloud fraction retrievals,” in *Proc. 10th Symp. on meteorological observations and instrumentation*, 1998, pp. 171–174.

[18] “Automatic Total Sky Imager TSI-880 | Biotechnology | Geneq,” *https://geneq.com/environment/en/product/yankee-environmental-systems/automatic-total-sky-imager-11922*, 2022.

[19] J. E. Shields, *Automated Whole Sky Imagers for Day and Night Cloud Field Assessment*, no. 234. University of California, San Diego, Scripps Institution of Oceanography …, 1993.

[20] A. Cazorla, F. J. Olmo, and L. Alados-Arboledas, “Development of a sky imager for cloud cover assessment,” *JOSA A*, vol. 25, no. 1, pp. 29–39, 2008.

[21] C. Long, J. Sabburg, J. Calbó, and D. Pagès, “Retrieving cloud characteristics from ground-based daytime color all-sky images.,” vol. 23. pp. 633–652, 2006. doi: 10.1175/JTECH1875.1.

[22] J. Yang, W. Lu, Y. Ma, and W. Yao, “An automated cirrus cloud detection method for a ground-based cloud image,” *Journal of Atmospheric and Oceanic Technology*, vol. 29, no. 4, pp. 527–537, 2012.

[23] S. Liu, C. Wang, B. Xiao, Z. Zhang, and Y. Shao, “Ground-based cloud classification using multiple random projections,” in *2012 International Conference on Computer Vision in Remote Sensing*, 2012, pp. 7–12.

[24] J. Yang, W. Lu, Y. Ma, and W. Yao, “An automated cirrus cloud detection method for a ground-based cloud image,” *Journal of Atmospheric and Oceanic Technology*, vol. 29, no. 4, pp. 527–537, 2012.

[25] S. Liu, M. Li, Z. Zhang, X. Cao, and T. S. Durrani, “Ground-based cloud classification using task-based graph convolutional network,” *Geophysical Research Letters*, vol. 47, no. 5, p. e2020GL087338, 2020.

[26] W. Zhuo, Z. Cao, and Y. Xiao, “Cloud classification of ground-based images using texture–structure features,” *Journal of Atmospheric and Oceanic Technology*, vol. 31, no. 1, pp. 79–92, 2014.

[27] S. Liu, C. Wang, B. Xiao, Z. Zhang, and X. Cao, “Tensor ensemble of ground-based cloud sequences: its modeling, classification, and synthesis,” *IEEE Geoscience and Remote Sensing Letters*, vol. 10, no. 5, pp. 1190–1194, 2013.

[28] J. Yang *et al.*, “An automated cloud detection method based on the green channel of total-sky visible images,” *Atmospheric Measurement Techniques*, vol. 8, no. 11, pp. 4671–4679, 2015.

[29] Y. Xiao, Z. Cao, W. Zhuo, L. Ye, and L. Zhu, “mCLOUD: A multiview visual feature extraction mechanism for ground-based cloud image categorization,” *Journal of Atmospheric and Oceanic Technology*, vol. 33, no. 4, pp. 789–801, 2016.

[30] J. Huertas-Tato, F. J. Rodríguez-Benítez, C. Arbizu-Barrena, R. Aler-Mur, I. Galvan-Leon, and D. Pozo-Vázquez, “Automatic cloud-type classification based on the combined use of a sky camera and a ceilometer,” *Journal of Geophysical Research: Atmospheres*, vol. 122, no. 20, pp. 11–45, 2017.

[31] R. Marquez and C. F. M. Coimbra, “Intra-hour DNI forecasting based on cloud tracking image analysis,” *Solar Energy*, vol. 91, pp. 327–336, 2013.

[32] J. Gan *et al.*, “Cloud type classification of total-sky images using duplex norm-bounded sparse coding,” *IEEE Journal of Selected Topics in Applied Earth*

Observations and Remote Sensing, vol. 10, no. 7, pp. 3360–3372, 2017.

[33] L. Ye, Z. Cao, and Y. Xiao, "DeepCloud: Ground-based cloud image categorization using deep convolutional features," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 10, pp. 5729–5740, 2017.

[34] S. Liu and M. Li, "Deep multimodal fusion for ground-based cloud classification in weather station networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, pp. 1–8, 2018.

[35] Q. Luo, Y. Meng, and Z. Zhou, "Cloud classification of ground-based infrared images based on Log-Euclidean distance," in *AIP Conference Proceedings*, 2018, vol. 1982, no. 1, p. 020032.

[36] J. Calbó and J. Sabburg, "Feature Extraction from Whole-Sky Ground-Based Images for Cloud-Type Recognition," *Journal of Atmospheric and Oceanic Technology*, vol. 25, no. 1, pp. 3–14, 2008, doi: 10.1175/2007JTECHA959.1.

[37] S. Liu, M. Li, Z. Zhang, B. Xiao, and X. Cao, "Multimodal ground-based cloud classification using joint fusion convolutional neural network," *Remote Sensing*, vol. 10, no. 6, p. 822, 2018.

[38] S. Liu, L. Duan, Z. Zhang, and X. Cao, "Hierarchical multimodal fusion for ground-based cloud classification in weather station networks," *IEEE Access*, vol. 7, pp. 85688–85695, 2019.

[39] A. Yu *et al.*, "A Novel Robust Classification Method for Ground-Based Clouds," *Atmosphere (Basel)*, vol. 12, no. 8, p. 999, 2021.

Detección de objetos para clasificación en imágenes de nubes multi-instancia.
Object detection for multi-instance cloud images classification.

Vega Maya, Edgar
División de Investigación y Posgrado, Facultad de Ingeniería
Universidad Autónoma de Querétaro
Querétaro, México
evega17@alumnos.uaq.mx

Resumen- Las nubes se encuentran entre los fenómenos meteorológicos que más efecto tienen sobre nuestra vida, ya que contribuyen significativamente al equilibrio energético de la tierra al absorber y distribuir radiación solar. Sin embargo, los registros y mediciones de las nubes que se hacen desde tierra dependen principalmente de observadores humanos, causando incertidumbres debido a la subjetividad que pueden tener sus juicios o mediciones. Por lo tanto, la investigación sobre la observación automática de nubes ha aumentado y sigue en progreso. Diversos trabajos en el área de clasificación de imágenes de nubes tomadas desde tierra se han presentado, sin embargo, cuando se encuentran distintos tipos de nube en una sola imagen (multi-instancia) el enfoque o estrategia que se ha tomado es el de asignarle la etiqueta 'mixta'. El objetivo principal de este trabajo es explorar las capacidades de la detección de objetos aplicado a la observación automática de las nubes. Se propone el uso del algoritmo para detección YOLOv3, con el fin de diferenciar entre distintos tipos de nubes contenidos en una imagen. En los resultados se demuestra la capacidad del algoritmo de detectar hasta con un índice de confiabilidad del 99% únicamente el tipo de nubes con el que fue entrenado (cúmulos).

Palabras clave: IAR, Clasificación automática de nubes; YOLOv3; detección de objetos

Abstract- Clouds are among the meteorological phenomena that have the most effect on our lives since they contribute significantly to the earth's energy balance by absorbing and distributing solar radiation. However, the records and ground-based measurements of clouds depend mainly on human observers, causing uncertainties due to the subjectivity that their judgments or measurements may have. Therefore, research on automatic cloud observation has increased and is still in progress. Continuous research in the area of ground-based cloud classification has been presented, however, when different types of cloud are found in a single image (multi-instance), the approach or strategy that has been taken is to assign the label 'mixed'. The main objective of this work is to explore the capabilities of object detection applied to automatic cloud observation. The use of the YOLOv3 detection algorithm is proposed, to differentiate between different types of clouds

contained in an image. The results show the ability of the algorithm to detect up to a reliability index of 99%, only the type of clouds with which it was trained (cumulus).

Keywords: IAR, Automatic cloud classification, YOLOv3, object detection.

1. Introducción

Existen distintas áreas donde la información de las nubes es bastante valiosa, una de ellas es la predicción de la irradiancia normal directa (DNI, por sus siglas en inglés). La irradiancia solar que llega a la superficie de la tierra tiene una influencia directa en la generación de energía por medio de parques fotovoltaicos. Por lo tanto, se han propuesto muchos trabajos para la predicción de DNI, donde se incluyen clasificación de nubes, detección de nubes y el movimiento de nubes[1]–[3].

Otra dimensión donde los datos de nubes pueden ser muy importantes es la agricultura de precisión (PA, por sus siglas en inglés) ya que las nubes están involucradas en la lluvia y la radiación solar necesaria para los cultivos. PA es un enfoque para el manejo de granjas que utiliza tecnología de la información para garantizar que los cultivos y el suelo reciban exactamente lo que necesitan para una salud y un rendimiento óptimos [4].

Dada la importancia de las nubes en distintas aplicaciones como las mencionadas anteriormente y las dificultades para detectar su comportamiento y contribución, se necesitan tecnologías para la descripción automática y continua de las características de las nubes. Se han propuesto distintas técnicas y métodos para la clasificación automática de las nubes, (ya sea con imágenes satelitales o tomadas desde tierra, este trabajo se enfoca especialmente en imágenes desde tierra) desde algoritmos de clasificación clásicos como el vecino más cercano (k-NN), el cual se utiliza después de dividir las imágenes en pequeños parches [5]. También se han utilizado máquinas de vectores de soporte como clasificador y diferentes técnicas para extracción de características como transformación de censo de color [6], transformación de características invariantes a escala (SIFT) [7], y transformación de censo de histograma (CENTRIST) [8]. Algoritmos más sofisticados como las redes neuronales convolucionales (CNN, por sus siglas en inglés) han sido propuestos inicialmente utilizándolos para extraer características y también clasificar [9], más adelante se propusieron diferentes enfoques utilizando CNN como un híbrido entre CNN y SVM, donde el primero se utiliza en la extracción de características y el segundo como clasificador [10]. Otra técnica propuesta para llevar a cabo la tarea de clasificar nubes fue la de utilizar información de una estación meteorológica como complemento a las características de las nubes y así ayudar al clasificador basado en CNN [11],[12]. Las últimas propuestas han incluido el efecto de la oclusión en las imágenes y así probar

la robustez de un algoritmo basado en un ensamble de dos redes CNN logrando altos niveles de precisión [13].

Sin embargo, uno de los problemas que aún no han sido abordados correctamente es el de clasificación de imágenes multi-instancia, es decir, cuando en la misma imagen aparecen distintos tipos de nubes. Tradicionalmente este problema se ha abordado definiendo una clase "nubes mixtas" para clasificar todas las imágenes con más de un tipo de nube presente. Esto puede ser de poca utilidad donde se requiere saber con exactitud el tipo de nubes que están apareciendo en el momento, además de que frecuentemente aparece más de un tipo de nubes en el cielo.

Para solventar este problema de clasificación se propone en este trabajo, utilizar detección de objetos para clasificar adecuadamente imágenes de nubes multi-instancia. La detección de objetos ha probado ser exitosa para detectar distintas categorías de objetos en una sola imagen [14], y hasta el momento no ha sido incluida en clasificación de nubes, así que este trabajo pretende llenar ese hueco y presentar los resultados obtenidos, de modo que se demuestre si es un método viable para clasificación de imágenes multi-instancia. De modo especifico este trabajo pretende obtener resultados que ayuden a responder las siguientes preguntas de investigación mediante la implementación del algoritmo YOLOv3 y el dataset MGCD propuesto en [15]:

• ¿Clasificar imágenes multi-instancia de nubes mediante algoritmos de detección de objetos permite distinguir y clasificar correctamente los distintos tipos de nube presentes?

• ¿Utilizar algoritmos de detección de objetos para clasificación de imágenes de nubes es eficiente para las aplicaciones relacionadas, frente a los métodos tradicionales de clasificación?

Dado que nuestro objetivo es aplicar detección de objetos para clasificar los distintos tipos de nubes en una imagen, el alcance de este trabajo será solamente entrenar el algoritmo para detectar la clase 'cúmulos' en imágenes donde haya más tipos de nubes, esto nos ayudará a sacar conclusiones sobre si el camino que se está tomando es correcto y se podría continuar aplicándolo a las demás clases. La estructura de este trabajo es como sigue: La sección de introducción continúa con un marco teórico acerca de los métodos utilizados y una descripción breve acerca de los tipos de nubes. Luego en la sección 2 se presenta la metodología seguida para el desarrollo de este trabajo. Finalmente, la sección 3 proporciona los resultados obtenidos y las conclusiones pertinentes al trabajo realizado.

Marco teórico

Esta sección se divide en dos partes la primera para el marco teórico de la detección de objetos y la segunda acerca de los distintos tipos de nubes.

A. Detección de objetos con YOLOv3

YOLOv3 (por sus siglas en inglés, You Only Look Once) es un algoritmo propuesto para detección de objetos por Redmon et al. [14]. La función principal del algoritmo

de detección es clasificar los objetos presentes en una imagen y además señalar la posición de cada objeto por medio de coordenadas que definen un recuadro que contiene al objeto. YOLOv3 utiliza una sola red neuronal convolucional para realizar la detección, la cual, de manera general se divide en los siguientes pasos:

1) Predicción de cajas delimitadoras (Bounding Box)

La predicción de los recuadros se hace utilizando clústeres, es decir, divide la imagen en regiones y luego utiliza el centroide de los clústeres como anclas de los recuadros y finalmente predice el ancho y el alto de la caja. La red neuronal predice 4 coordenadas para cada caja y además calcula una puntuación de confiabilidad utilizando regresión logística. La confiabilidad está en un rango de [0,1] siendo 1 la máxima puntuación, la cual se obtendría cuando la predicción de la caja delimita o se superpone exactamente sobre el objeto real.

Si la predicción se superpone al objeto real pero no lo delimita en su totalidad, se puede ignorar la predicción acorde a algún umbral que en este caso es 0.5. Esta métrica es conocida como IoU (Intersection over unión, por sus siglas en inglés), la cual nos ayuda a medir que tan buena fue la predicción de la caja delimitadora. En Fig. 1 se puede observar un ejemplo de las cajas delimitadoras, así como el puntaje de confiabilidad.

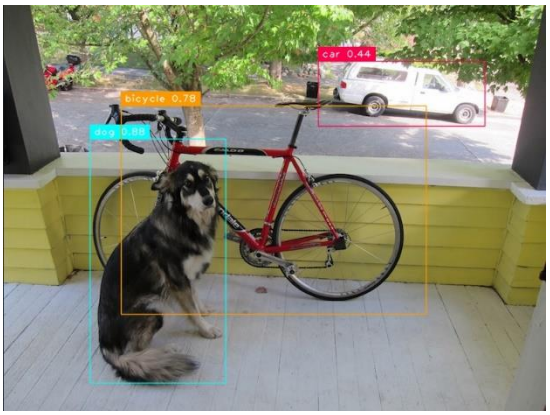


Figura 1. Ejemplo de detección de objetos con YOLOv3 [12]

2) Predicción de la clase

Además de hacer la predicción de donde se ubica un objeto, también se predice la clase a la que pertenece el objeto, usando clasificadores logísticos y entropía cruzada binaria como función de pérdida para el entrenamiento. Esto facilita la clasificación cuando un objeto pertenece a más de una clase por ejemplo persona y mujer lo cual sería imposible con la función softmax por ejemplo.

3) Extractor de características

La parte que realiza la extracción de características es una red con 53 capas llamada Darknet-53 la cual es entrenada con las imágenes y anotaciones de la aplicación en curso, además de las imágenes proporcionadas el algoritmo realiza aumento de datos, entrenamiento multiescala, normalización por lotes ente otras técnicas. Cabe mencionar que este extractor de características puede cambiarse por algún otro modelo de red que se ajuste mejor a nuestras necesidades.

4) Métricas de evaluación

- *Intersección sobre unión (IOU)*

Es una métrica basada en el índice Jaccard, que evalúa la superposición entre dos cuadros delimitadores, se define como el cociente del área de superposición entre la caja delimitadora predicha y la caja delimitadora verdadera:

$$IOU = \frac{área\ de\ superposición}{área\ de\ unión}$$

- Curva *AP* (precisión vs recall)

Si la precisión de una categoría del detector permanece alta a medida que aumenta la recuperación (recall), se considera buena, lo que significa que si se cambia el umbral de confianza, la precisión y la tasa de recuperación siguen siendo altas.

B. Tipos de nubes

Según la Organización Meteorológica Mundial [16], una nube se define como un hidrometeoro que consiste en pequeñas partículas de agua líquida o hielo, o ambas, suspendidas en la atmósfera y que generalmente no tocan el suelo. A pesar de los estudios realizados sobre la nubosidad, se reconoce comúnmente que todavía hay mucho que aprender sobre este fenómeno. Los efectos combinados de varios tipos de cobertura de nubes proporcionan un alto elemento de incertidumbre, lo que resulta en un gran factor de duda en los modelos y pronósticos climáticos.

El Atlas Internacional de Nubes (ICA) es la única referencia autorizada y más completa para identificar nubes. El ICA reconoce hoy 10 géneros de nubes primarias, que se clasifican en función principalmente de su ubicación vertical en la atmósfera y su morfología relativa. En general, las nubes se dividen de acuerdo con su altura de base sobre la superficie de la Tierra como se muestra en la Tabla 1, y los nombres de las nubes se componen de prefijos y sufijos latinos que, cuando se combinan, revelan la naturaleza de esta.

TABLA 1 CLASIFICACIÓN DE LAS NUBES POR ALTURA

	Género	**Altura (m)**
Nivel Alto	Cirrus Cirrostratus Cirrocúmulos	> 5000
Nivel medio	Altocúmulos Altostratus Nimbostratus	[2000 ,7000]
Nivel bajo	Stratus Stratocúmulos Cumulonimbus Cúmulos	< 2000

En Fig. 2 se muestra un gráfico de las nubes acomodadas según la altura en la que se pueden formar.

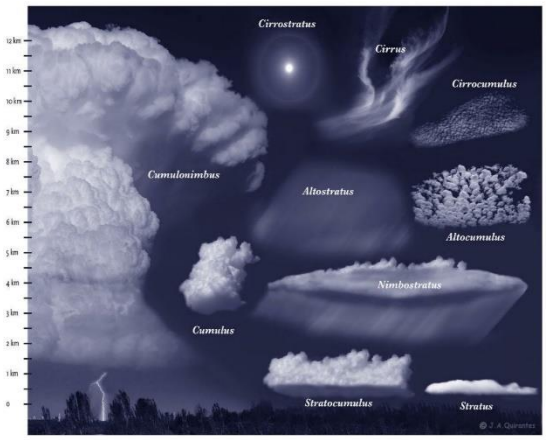


Figura 2. Relación entre los diez tipos de nube y su altura [14].

2. Metodología

El desarrollo de este trabajo consta de varias etapas que se muestran a continuación a modo de subsecciones:

1) Conjunto de imágenes con anotaciones

La base de datos utilizada fue la siguiente: *Multimodal Ground-based Cloud Dataset (MGCD)* [15], el cual contiene imágenes de nubes del tamaño 1024 x 1024 pixeles capturadas con un lente ojo de pescado el cual proporciona un amplio rango de observación de 180 grados. Para generar las anotaciones se utilizó una herramienta llamada labelImg la cual está desarrollada en Python, es de libre uso y se puede encontrar en https://github.com/tzutalin/labelImg.

Con esta herramienta podemos ir anotando las cajas delimitadoras una a una que servirán como datos de entrenamiento para YOLOv3, en Fig. 3 se observa el entorno con un ejemplo de las cajas realizadas para esa imagen, dado que nuestro objetivo es reconocer cúmulos se crearon cajas solamente en las nubes más representativas de este tipo.

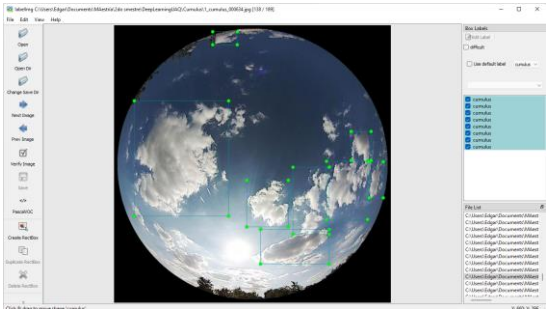


Figura 3. Ejemplo de anotaciones hechas con la herramienta labelImg.

Luego de haber marcado todas las cajas de cada una de las imágenes, automáticamente se crea un archivo con la información de las clases y de las coordenadas por cada una de las imágenes. Este archivo puede tener distintos formatos el cual se puede elegir en la misma herramienta y para este caso utilizamos PASCAL VOC, el cual nos genera archivos .xml, dado que para entrenar YOLOv3 necesitamos otro formato podemos cambiarlo directamente desde la herramienta a formato YOLO o mediante un script podemos hacer la conversión, el repositorio que estaremos utilizando contiene un script para convertirlos al formato que acepta YOLOv3.

2) Conversión de PASCAL VOC a YOLO

Como mencionamos antes, ya existen herramientas que te permiten hacer anotaciones en formato YOLO directamente, en esta ocasión decidimos mantener nuestras anotaciones en formato VOC para los casos donde ya tengamos las anotaciones en este formato y necesitemos convertirlas.

El primer paso es que necesitamos tener un archivo txt con las rutas de cada una de las imágenes del dataset a excepción de la exensión, dado que la imagen (.jpg) y la anotación(.xml) tienen el mismo nombre este archivo funciona para mandar llamar tanto imágenes como anotaciones. El repositorio que estaremos utilizando es el siguiente https://gitee.com/kaanwang/keras-YOLOv3-model-set el cual contiene un

script para convertir de PASCAL VOC a YOLO y se llama voc_annotation.py en Fig. 4 se encuentra el archivo una vez que hemos clonado el repositorio en nuestro ambiente de programación que en este caso es un notebook de Google Colab.

3) Convertir modelo a keras

Antes de entrenar debemos convertir nuestro modelo en este caso YOLO3 a un modelo de keras, el repositorio contiene las configuraciones de la red, pero hay que descargar los pesos de la página del desarrollador https://pjreddie.com/media/files/yolov3.weights y pasarlos como argumento al script que convierte a un modelo en formato h5. Aquí se descargan los pesos de la página del desarrollador y se guardan en la ruta indicada como primer argumento, el siguiente paso es generar el modelo de keras con extensión .h5.

4) Entrenamiento

Una vez que tenemos nuestros datos para entrenar en formato YOLO utilizamos el archivo train.py que contiene el mismo repositorio con los siguientes argumentos:

model_type=yolo3_mobilenet_lite anchors_path=configs/yolo3_anchors.txt annotation_file=Annotation_cúmulos/2007_train.txt val_annotation_file=Annotation_cúmulos/2007_val.txt classes_path=predefined_classes.txt eval_online save_eval_checkpoint

El primer argumento corresponde al modelo a usar para el extractor de características (backbone) en este caso se utilizó yolo3_mobilenet_lite estos modelos se pueden encontrar en la siguiente ruta en el repositorio /content/keras-YOLOv3-model-set/yolo3/models/ además de los modelos existentes se pueden descargar algunos otros si es necesario.

El siguiente argumento es anchors_path que como se explicó en el marco teórico son anclas que usa el detector para la predicción de los bounding box, en esta ocasión se utilizan los que vienen por default en el repositorio para YOLO3.

Luego se pasan como argumento los archivos generados en el paso anterior tanto los datos de entrenamiento como los de validación, y finalmente el archivo donde vienen las clases de objetos, también se le dice en los argumentos que evalue mientras entrena (eval_online) y que guarde cada checkpoint de evaluación. De estos checkpoints elegiremos el que mejor haya dado resultados de acuerdo con las métricas mAP y lo utilizaremos para generar el modelo entrenado con nuestros datos como backbone de YOLOv3.

5) Evaluar y crear modelo

Finalmente evaluamos el modelo entrenado con los datos de 2007_test.txt y este nos generara algunas métricas para evaluar el desempeño del modelo entrenado. También es necesario generar el modelo en formato h5 que será el que nos permitirá hacer detección de nubes tipo cúmulos.

Para evaluar y exportar el modelo obtenido se necesitan algunos argumentos obligatorios, el primer argumento es el modelo de la red a utilizar, en este caso es una mobilenet_lite,

el siguiente argumento son los pesos que fueron calculados durante el entrenamiento, recordemos que en cada evaluación durante el entrenamiento se guardaba un log de los pesos y los resultados que obtuvo. Se pasan también los anchors de YOLO3 y las clases. El modelo final model.h5 será el que usemos para hacer detecciones en algunas imágenes de prueba.

3. Resultados y discusión

Se entrenó el algoritmo de YOLOv3 con 189 imágenes con sus respectivas anotaciones de nubes del tipo cúmulos durante 250 épocas. Acorde a la evaluación se obtuvieron las siguientes métricas que se muestran en la tabla 2.

TABLA 2 MÉTRICAS OBTENIDAS PARA EL MODELO ENTRENADO

Evaluación Average precision Pascal voc			
cúmulos	AP: 0.7578	Precision: 0.1365	Recall: 0.9290
mAP@IoU = 0.50	Result: 75.77729		
mPrec@IoU = 0.50	Result: 13.65461		
mRec@IoU = 0.50	Result: 92.89617		
Evaluation time cost:	7.645564s		

En Fig. 5 observamos la curva AP la cual está formada por el comportamiento de la Precisión vs el Recall respecto a nuestro modelo, se observa que a medida que aumenta el recall la precisión trata de mantenerse en un nivel alto, sin embargo, al final cae drásticamente lo cual no es deseable. Como observamos en la Tabla 2 se obtuvo un AP de 0.757, el cual sabemos que es una métrica donde el ideal sería el 100% pero dada la aplicación que tenemos podemos preguntarnos qué tan bueno es nuestro modelo. Para el caso de este trabajo donde solo tenemos una clase las métricas obtenidas para la única clase son las mismas del modelo, como podría suponerse.

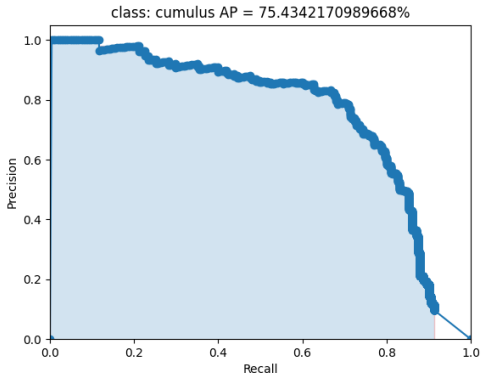


Figura 4. Curva de la métrica AP para la clase cúmulos.

Otro resultado que nos arroja la evaluación de nuestro modelo se presenta en la Fig. 5, donde se presenta el número total de predicciones divididas en falsas predicciones y predicciones verdaderas (False positives & True positives), se evaluó sobre 38 imágenes que es el tamaño de nuestro Test_set, y como solo tenemos una clase el total de objetos detectados pertenecen a la clase cúmulos. Se observa que el modelo tiene una mayoría de predicciones erróneas, es decir, que creó bounding boxes donde no había nubes de tipo cúmulos.

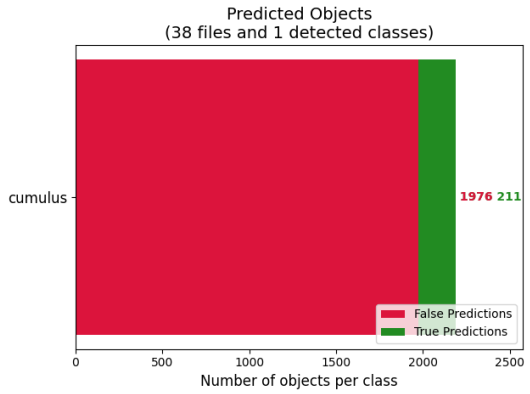


Figura 5. Predicción de objetos durante la evaluación.

Finalmente se hicieron algunas pruebas con imágenes externas, simulando algunos casos donde se podrían encontrar distintos tipos de nubes, entre ellas cúmulos. En Fig. 6 se observa una imagen que contiene principalmente dos tipos de nubes cúmulos y cirrocúmulos, la principal diferencia entre estas dos clases es que la primera está formada casi en su totalidad de una sola pieza con textura como de algodón mientras que los cirrocúmulos están formados por conjuntos de nubes más pequeñas con similar textura.

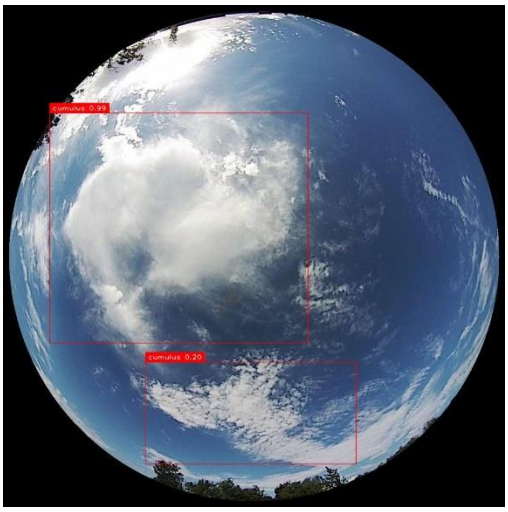


Figura 6. Detección de cúmulos en una imagen multiclase.

Se observa que se detectaron dos bounding box uno con un índice de confiabilidad del 99% el cual a la vista humana es una clasificación acertada, por otro lado, el otro bounding box detectado tiene una confiabilidad del 20% y en efecto es una nube muy parecida pero no pertenece a la clase cúmulos sino a los cirrocúmulos.

En Fig. 8 se observa otra imagen donde se probó el detector, en su mayoría se observan nubes de la clase cúmulos, pero también hay algunas nubes en forma de velos o con textura como de plumas que corresponden a la clase Altoestratos. Es interesante mencionar que el detector logró identificar la mayoría de las nubes cúmulos pero ignoro completamente la nube correspondiente a Altoestratos indicada con una flecha amarilla.

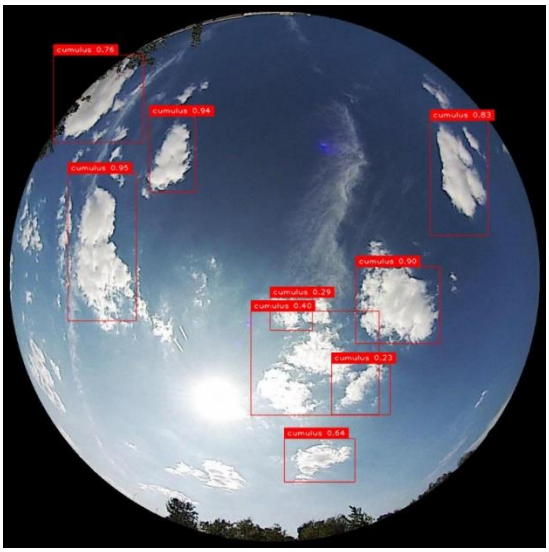


Figura 7. Detección de cúmulos en una imagen multiclase (cúmulos y altoestratos).

Además de los hallazgos mencionados anteriormente también se encontraron casos donde el comportamiento del

detector fue errático. Por ejemplo, detectar gotas de agua como si fueran instancias de cúmulos, estos y otros problemas se proponen atacar en trabajo futuro.

4. Conclusiones

Luego de haber implementado el algoritmo de detección de objetos YOLOv3, con el objetivo de clasificar imágenes multiclase o multi-instancia se obtuvieron las siguientes conclusiones.

Las métricas obtenidas para este trabajo en la sección de resultados deben ser evaluadas acorde a la aplicación donde se está utilizando el detector, en este caso, no nos interesa mucho si el bounding box delimito correctamente todos los pixeles del objeto, inclusive podría no importarnos el lugar donde se encuentra el objeto simplemente saber que está presente. Pensemos en una estación meteorológica donde se hacen registros de este fenómeno, es posible que se importante saber todos los tipos de nubes que aparecieron durante cierto lapso de días o meses, pero podría no importarnos en qué lugar exactamente se formaron.

Como se observó en algunas de las Figuras los errores donde detectaba nubes que no eran las correctas lo hacía con un índice de confiabilidad bajo, lo cual daría lugar a utilizar un umbral de clasificación para descartar predicciones con base a su confiabilidad.

Finalmente es importante mencionar que el desarrollo de este trabajo fue implementado en su totalidad en un notebook de Google Colab, lo cual impidió por ejemplo entrenar con un backbone más robusto debido a la cantidad de recursos que demandaba. Esto se traduce a que se puede mejorar el desempeño del detector de manera significativa, sin mencionar las versiones más potentes que ahora existen de YOLO.

Sin embargo, se pueden enlistar los siguientes logros de esta investigación, así como el trabajo futuro propuesto.

- Los resultados indican que esta técnica podría funcionar exitosamente como clasificador para este tipo de imágenes, aunque debido a las limitantes de tiempo y de recursos solo se pudo probar con una sola clase de nubes, los resultados motivan a intentarlo con todas las clases necesarias para evaluar los resultados.
- Se propone también utilizar nuevas metodologías de detección de objetos, puesto que la literatura en esta área continúa avanzando constantemente.
- Se propone una mejora al problema de clasificación de imágenes de nubes, cuando estas contienen más de un tipo de nubes en ellas.

Agradecimientos

El autor agradece al Consejo Nacional de Ciencia y tecnología por la beca durante el tiempo de maestría en esta universidad, así como al Dr. Juvenal Rodríguez

Reséndiz por el asesoramiento y apoyo durante la investigación realizada.

Referencias

[1] C.-L. Fu and H.-Y. Cheng, "Predicting solar irradiance with all-sky image features via regression," Solar Energy, vol. 97, pp. 537–550, 2013.

[2] H.-Y. Cheng and C.-C. Yu, "Multi-model solar irradiance prediction based on automatic cloud classification," Energy, vol. 91, pp. 579–587, 2015.

[3] H.-Y. Cheng, C.-C. Yu, and S.-J. Lin, "Bi-model short-term solar irradiance prediction using support vector regressors," Energy, vol. 70, pp. 121–127, 2014.

[4] M. A. Oliver, "An overview of geostatistics and precision agriculture," Geostatistical applications for precision agriculture, pp. 1–34, 2010.

[5] S. Liu, C. Wang, B. Xiao, Z. Zhang, and Y. Shao, "Ground-based cloud classification using multiple random projections," in 2012 International Conference on Computer Vision in Remote Sensing, 2012, pp. 7–12.

[6] W. Zhuo, Z. Cao, and Y. Xiao, "Cloud classification of ground-based images using texture–structure features," Journal of Atmospheric and Oceanic Technology, vol. 31, no. 1, pp. 79–92, 2014.

[7] S. Liu, C. Wang, B. Xiao, Z. Zhang, and X. Cao, "Tensor ensemble of ground-based cloud sequences: its modeling, classification, and synthesis," IEEE Geoscience and Remote Sensing Letters, vol. 10, no. 5, pp. 1190–1194, 2013.

[8] Y. Xiao, Z. Cao, W. Zhuo, L. Ye, and L. Zhu, "mCLOUD: A multiview visual feature extraction mechanism for ground-based cloud image categorization," Journal of Atmospheric and Oceanic Technology, vol. 33, no. 4, pp. 789–801, 2016.

[9] L. Ye, Z. Cao, and Y. Xiao, "DeepCloud: Ground-based cloud image categorization using deep convolutional features," IEEE Transactions on Geoscience and Remote Sensing, vol. 55, no. 10, pp. 5729–5740, 2017.

[10] S. Liu and M. Li, "Deep multimodal fusion for ground-based cloud classification in weather station networks," EURASIP Journal on Wireless Communications and Networking, vol. 2018, no. 1, pp. 1–8, 2018.

[11] S. Liu, M. Li, Z. Zhang, B. Xiao, and X. Cao, "Multimodal ground-based cloud classification using joint fusion convolutional neural network," Remote Sensing, vol. 10, no. 6, p. 822, 2018.

[12] S. Liu, L. Duan, Z. Zhang, and X. Cao, "Hierarchical multimodal fusion for ground-based cloud classification in weather station networks," IEEE Access, vol. 7, pp. 85688–85695, 2019.

[13] A. Yu et al., "A Novel Robust Classification Method for Ground-Based Clouds," Atmosphere (Basel), vol. 12, no. 8, p. 999, 2021.

[14] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[15] S. Liu, M. Li, Z. Zhang, X. Cao, and T. S. Durrani, "Ground-based cloud classification using task-based graph convolutional network," Geophysical Research Letters, vol. 47, no. 5, p. e2020GL087338, 2020.

[16] "keras-YOLOv3-model-set," https://gitee.com/kaanwang/keras-YOLOv3-model-set, 2018.

[17] "World Meteorological Organization," https://public.wmo.int/en/WorldMetDay2017/resources, 2017.

[18] "Meteoglosario Visual," https://meteoglosario.aemet.es/es/termino/464_genero-nuboso, 2018.

Bibliografía

[1] F. Chollet, *Deep learning with Python*. Simon and Schuster, 2021.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] A. Meteorología, "Género nuboso," 2018. `https://meteoglosario.aemet.es/es/termino/464_genero-nuboso`.

[4] X. Xiao, M. Yan, S. Basodi, C. Ji, and Y. Pan, "Efficient hyperparameter optimization in deep learning using a variable length genetic algorithm," *arXiv preprint arXiv:2006.12703*, 2020.

[5] M. A. Oliver, "An overview of geostatistics and precision agriculture," *Geostatistical applications for precision agriculture*, pp. 1–34, 2010.

[6] J. M. P. Czarnecki, S. Samiappan, M. Zhou, C. D. McCraine, and L. L. Wasson, "Real-time automated classification of sky conditions using deep learning and edge computing," *Remote Sensing*, vol. 13, no. 19, p. 3859, 2021.

[7] J. Calbó, D. Pages, and J.-A. González, "Empirical studies of cloud effects on uv radiation: A review," *Reviews of Geophysics*, vol. 43, no. 2, 2005.

[8] M. Thirugnanasambandam, S. Iniyan, and R. Goic, "A review of solar thermal technologies," *Renewable and sustainable energy reviews*, vol. 14, no. 1, pp. 312–322, 2010.

[9] L. El Chaar, N. El Zein, *et al.*, "Review of photovoltaic technologies," *Renewable and sustainable energy reviews*, vol. 15, no. 5, pp. 2165–2175, 2011.

[10] M. Medrano, A. Gil, I. Martorell, X. Potau, and L. F. Cabeza, "State of the art on high-temperature thermal energy storage for power generation. part 2—case studies," *Renewable and Sustainable Energy Reviews*, vol. 14, no. 1, pp. 56–72, 2010.

[11] J. Gan, W. Lu, Q. Li, Z. Zhang, J. Yang, Y. Ma, and W. Yao, "Cloud type classification of total-sky images using duplex norm-bounded sparse coding," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 7, pp. 3360–3372, 2017.

[12] S. Liu and M. Li, "Deep multimodal fusion for ground-based cloud classification in weather station networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, pp. 1–8, 2018.

[13] J. Shields, M. Karr, T. Tooman, D. Sowle, and S. Moore, "The whole sky imager–a year of progress," in *Eighth Atmospheric Radiation Measurement (ARM) Science Team Meeting, Tucson, Arizona*, pp. 23–27, Citeseer, 1998.

[14] B. Thurairajah and J. A. Shaw, "Cloud statistics measured with the infrared cloud imager (ici)," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 9, pp. 2000–2007, 2005.

[15] J. Calbo and J. Sabburg, "Feature extraction from whole-sky ground-based images for cloud-type recognition," *Journal of Atmospheric and Oceanic Technology*, vol. 25, no. 1, pp. 3–14, 2008.

[16] A. Heinle, A. Macke, and A. Srivastav, "Automatic cloud classification of whole sky images," *Atmospheric Measurement Techniques*, vol. 3, no. 3, pp. 557–567, 2010.

[17] W. Zhuo, Z. Cao, and Y. Xiao, "Cloud classification of ground-based images using texture–structure features," *Journal of Atmospheric and Oceanic Technology*, vol. 31, no. 1, pp. 79–92, 2014.

[18] A. Kazantzidis, P. Tzoumanikas, A. F. Bais, S. Fotopoulos, and G. Economou, "Cloud detection and classification with the use of whole-sky ground-based images," *Atmospheric Research*, vol. 113, pp. 80–88, 2012.

[19] H.-Y. Cheng and C.-C. Yu, "Block-based cloud classification with statistical features and distribution of local texture features," *Atmospheric Measurement Techniques*, vol. 8, no. 3, pp. 1173–1182, 2015.

[20] Y. Xiao, Z. Cao, W. Zhuo, L. Ye, and L. Zhu, "mcloud: A multiview visual feature extraction mechanism for ground-based cloud image categorization," *Journal of Atmospheric and Oceanic Technology*, vol. 33, no. 4, pp. 789–801, 2016.

[21] L. Ye, Z. Cao, and Y. Xiao, "Deepcloud: Ground-based cloud image categorization using deep convolutional features," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 10, pp. 5729–5740, 2017.

[22] R. Tapakis and A. Charalambides, "Equipment and methodologies for cloud detection and classification: A review," *Solar Energy*, vol. 95, pp. 392–430, 2013.

[23] A. Yu, M. Tang, G. Li, B. Hou, Z. Xuan, B. Zhu, and T. Chen, "A novel robust classification method for ground-based clouds," *Atmosphere*, vol. 12, no. 8, 2021.

[24] R. C. Gonzalez and R. E. Woods, "Processamento digital de imagem," *Pearson, ISBN-10: 8576054019,*, vol. 10, pp. 11–27, 2010.

[25] C. C. Aggarwal *et al.*, "Neural networks and deep learning," *Springer*, vol. 10, pp. 978–3, 2018.

[26] Y.-T. Zhou and R. Chellappa, "Computation of optical flow using a neural network.," in *ICNN*, pp. 71–78, 1988.

[27] 2017.

[28] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[29] S. Forrest, "Genetic algorithms," *ACM computing surveys (CSUR)*, vol. 28, no. 1, pp. 77–80, 1996.

[30] S. Liu, M. Li, Z. Zhang, B. Xiao, and T. S. Durrani, "Multi-evidence and multi-modal fusion network for ground-based cloud recognition," *Remote Sensing*, vol. 12, no. 3, p. 464, 2020.

[31] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, pp. 6105–6114, PMLR, 2019.

[32] M. Sandler, "col. mobilenetv2: Inverted residuals and linear bottlenecks," *arXiv preprint arXiv:1801.04381*, 2019.

[33] S. Liu, M. Li, Z. Zhang, X. Cao, and T. S. Durrani, "Ground-based cloud classification using task-based graph convolutional network," *Geophysical Research Letters*, vol. 47, no. 5, p. e2020GL087338, 2020.

[34] N. Veček, M. Mernik, B. Filipič, and M. Črepinšek, "Parameter tuning with chess rating system (crs-tuning) for meta-heuristic algorithms," *Information Sciences*, vol. 372, pp. 446–469, 2016.

[35] M. Manzo and S. Pellino, "Voting in transfer learning system for ground-based cloud classification," *Machine Learning and Knowledge Extraction*, vol. 3, no. 3, pp. 542–553, 2021.

[36] "Total sky imager, tsi880," 2023. `https://geneq.com/environment/en/product/yankee-environmental-systems/automatic-total-sky-imager-11922`.