COLEGIO DE POSTGRADUADOS

INSTITUCIÓN DE ENSEÑANZA E INVESTIGACIÓN EN CIENCIAS AGRÍCOLAS

CAMPUS MONTECILLO

POSTGRADO DE HIDROCIENCIAS

CLASIFICACIÓN DE USO DEL SUELO Y VEGETACIÓN CON REDES NEURONALES CONVOLUCIONALES: CASO DE ESTUDIO CUENCA DEL RÍO ATOYAC-SALADO

RODOLFO MONTIEL GONZÁLEZ

T E S I S
PRESENTADA COMO REQUISITO PARCIAL
PARA OBTENER EL GRADO DE:

MAESTRO EN CIENCIAS

MONTECILLO, TEXCOCO, ESTADO DE MÉXICO

2022

COLEGIO DE POSTGRADUADOS

INSTITUCIÓN DE ENSEÑANZA E INVESTIGACIÓN EN CIENCIAS AGRÍCOLAS

La presente tesis titulada: **Clasificación de uso del suelo y vegetación con redes Neuronales Convolucionales: Caso de estudio Cuenca del Río Atoyac-Salado,** realizada por el estudiante: **Rodolfo Montiel González,** bajo la dirección del Consejo Particular indicado, ha sido aprobada por el mismo y aceptada como requisito parcial para obtener el grado de:

MAESTRO EN CIENCIAS
HIDROCIENCIAS

CONSEJO PARTICULAR

CONSEJERO

DR. MARTIN ALEJANDRO BOLAÑOS GONZÁLEZ

ASESORA

DRA. ANTONIA MACEDO CRUZ

ASESOR

DR. AGUSTÍN RODRÍGUEZ GONZÁLEZ

Montecillo, Texcoco, Estado de México, mayo de 2022

CLASIFICACIÓN DE USO DEL SUELO Y VEGETACIÓN CON REDES NEURONALES CONVOLUCIONALES: CASO DE ESTUDIO CUENCA DEL RÍO ATOYAC-SALADO

Rodolfo Montiel González, MC
Colegio de Postgraduados, 2022

RESUMEN

La clasificación de uso del suelo y vegetación es un ejercicio complejo y difícil de realizar con métodos tradicionales, ya que son costosos, consumen mucho tiempo y requieren la intervención manual de expertos. Los modelos de aprendizaje profundo son altamente capaces de aprender esta semántica compleja, dando resultados similares o superiores, lo que hace plausible su aplicación en la identificación automática de usos del suelo y vegetación a partir de patrones espacio-temporales extraídos de su apariencia, de estudios previos de clasificación e identificación de clases. El objetivo fue construir y probar un modelo de red neuronal convolucional de aprendizaje profundo en la clasificación de 22 clases distintas de cobertura y uso del suelo dentro de la cuenca río Atoyac-Salado, utilizando información obtenida con el Instrumento Multiespectral Sentinel-2. El modelo propuesto se entrenó con una combinación diferente de hiperparámetros donde la precisión depende del optimizador, la función de activación, el tamaño del filtro, la tasa de aprendizaje y el tamaño del lote. El aprendizaje profundo de las redes neuronales convolucionales utilizando una combinación optimizada de parámetros proporcionó una precisión del 84.57% para el conjunto de datos. Para reducir el sobreajuste se empleó un método de regularización llamado dropout que resultó ser muy eficaz.

Palabras clave: *aprendizaje de máquina, aprendizaje profundo, clasificación automática, entrenamiento, imágenes Sentinel-2, río Atoyac-Salado*.

LAND USE AND VEGETATION CLASSIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS: CASE STUDY OF THE ATOYAC-SALADO RIVER BASIN

Rodolfo Montiel González, MC
Colegio de Postgraduados, 2022

ABSTRACT

Land use and vegetation classification is a complex and difficult exercise to perform with traditional methods, as they are costly, time-consuming and require manual intervention by experts. Deep learning models are highly capable of learning this complex semantics, giving similar or superior results, which makes plausible their application in the automatic identification of land use and vegetation from spatio-temporal patterns extracted from their appearance, from previous classification studies and class identification. The objective was to build and test a deep learning convolutional neural network model in the classification of 22 different land cover and land use classes within the Atoyac-Salado river basin, using information obtained with the Sentinel-2 Multispectral Instrument. The proposed model was trained with a different combination of hyperparameters where the accuracy depends on the optimizer, activation function, filter size, learning rate and lot size. Deep learning of convolutional neural networks using an optimized combination of parameters provided an accuracy of 84.57% for the data set. To reduce overfitting, a regularization method called dropout was employed and proved to be very effective.

Key words: *automatic classification, Atoyac-Salado River, deep learning, machine learning, Sentinel-2 images, training.*

AGRADECIMIENTOS

Al Colegio de Postgraduados campus Montecillo, por haberme brindado la oportunidad de continuar mi formación académica y personal, además de pertenecer al prestigiado posgrado de Hidrociencias.

Al Consejo Nacional de Ciencia y Tecnología, CONACYT, por la beca otorgada para el desarrollo de los estudios e investigación de maestría.

Al Dr. Martín Alejandro Bolaños González por su valiosa guía a lo largo de la investigación, asesorando y apoyando en todo momento para la culminación de este trabajo y a su familia.

A la Dra. Antonia Macedo Cruz por la confianza depositada en mí y la atención recibida para revisar este trabajo.

Al Dr. Agustín Rodríguez González por su amistad y apoyo durante muchos años.

A ti mami que con tu entrañable amor me criaste, a Conchita por todo su cariño, a ti padre por ser mi superhéroe.

A Lila por su valioso apoyo, cariño y confianza en mí.

A mis colegas Vicente, Álvaro, Daniel, Enrique, Abraham y Antonio por compartir este camino.

Me gustaría agradecer a todas aquellas personas que me han formado desde niño, en todos los niveles de educación que sin ellos no estaría ahora escribiendo esto.

Finalmente, a todas esas personas que han compartido su vida conmigo.

DEDICATORIA

Una de las bendiciones más grandes de mi vida ha sido conocer a mi hija

Con amor para Sophie Nayhan

CONTENIDO

LISTA DE CUADROS

LISTA DE FIGURAS

I. INTRODUCCIÓN

Hoy en día gran parte de nuestra vida cotidiana es acompañada con aplicaciones que ofrece el mundo moderno en el desarrollo de herramientas digitales como la inteligencia artificial, que basa gran parte de su ocupación en la capacidad de aplicar algoritmos de aprendizaje automático a diversas necesidades. Esta disciplina ha registrado los avances más significativos en tecnología y se ha convertido en uno de los campos que promete grandes avances a corto plazo; bien puede ser considerada como el mayor avance tecnológico de los últimos tiempos. La inteligencia artificial es un campo multidisciplinario que se centra en la investigación de múltiples conceptos enfocados a imitar las funciones que el humano desarrolla, como el procesamiento del lenguaje, el razonamiento, la planificación, la representación del conocimiento, el aprendizaje, entre otros; una de las ramas de las ciencias de la computación que más interés ha despertado en la actualidad, debido a su enorme campo de aplicación, (Ponce *et al*., 2014). Dentro de este campo destaca el aprendizaje automático o aprendizaje máquina (*machine learning* en inglés), cuyo nombre sugiere el uso de una máquina/computadora para aprender de forma análoga a cómo el cerebro aprende y predice con el fin de automatizar operaciones para reducir la necesidad de la intervención humana en la detección automática de patrones significativos en los datos (Theodoridis, 2015). En las últimas dos décadas el aprendizaje automático se ha convertido en una herramienta común en casi cualquier tarea que requiera la extracción de información de grandes conjuntos de datos (Shalev & Ben, 2014).

Dentro de la rama del Aprendizaje Automático una categoría que también está siendo muy popular es el aprendizaje profundo (*Deep Learning* en inglés). El aprendizaje

profundo es una de las técnicas modernas más versátiles para la extracción y clasificación de características (Bhosle & Musande, 2019); puede analizar de forma inteligente los datos a gran escala (Sarker, 2021). Los algoritmos de aprendizaje profundo extraen abstracciones complejas de alto nivel como representaciones de datos a través de un proceso de aprendizaje jerárquico (Najafabadi *et al*., 2015), son programas computacionales que buscan optimizar los parámetros de un modelo usando datos previos o datos de entrenamiento. Los modelos pueden ser inductivos, cuando permiten hacer predicciones sobre el futuro o bien descriptivos cuando permiten generar conocimiento a partir de los datos (Ponce *et al*., 2014). La red neuronal es la arquitectura más empleada en *deep learning*. Está formada por múltiples capas, donde se conectan las neuronas de una capa con las neuronas de la capa adyacente por medio de conexiones, que tienen asociado un valor variable el cual es llamado peso sinóptico. Se conoce con el nombre de arquitectura la forma en la que se unen los diferentes elementos, neuronas, mediante una serie de conexiones, pesos sinápticos (Soria y Blanco, 2001).

En el campo del aprendizaje automático se encuentran las redes neuronales convolucionales (CNN), que son una clase de red de aprendizaje profundo, aplicada en el análisis visual de imágenes y ofrecen un sistema que trabaja en paralelo, donde al mismo tiempo se pueden procesar varias conexiones de neuronas, reduciendo el tiempo de cómputo. Esto ha logrado mejoras considerables más allá de los registros de última generación en el análisis de imágenes y ha atraído un gran interés tanto en las comunidades académicas como industriales (Krizhevsky *et al*., 2012). Estas redes utilizan conexiones locales para extraer eficazmente la información espacial y pesos

compartidos para reducir significativamente el número de parámetros (Chen *et al*., 2016) y aunque estas requieran de un elevado número de parámetros para ser entrenadas, una vez que esos parámetros están ajustados ofrecen grandes resultados.

Por otro lado, la clasificación digital de imágenes de satélite se ha convertido en una herramienta indispensable para monitorear la cobertura terrestre, la percepción remota y los sistemas de información geográfica, al ser capaces de obtener estos datos de manera ágil y a bajo costo, se han convertido en tecnologías fundamentales en el desarrollo de la geografía moderna (García & Mas, 2008).

Entre las aplicaciones más frecuentes del procesamiento digital de imágenes multiespectrales de satélite se encuentra la generación de cartografía relacionada con la identificación y estado de los diferentes tipos de cubierta del terreno (Palacio & Luna, 1994). La teledetección de los cambios en la cobertura y el uso del suelo es un área de estudio y aplicación muy diversa, con diferentes significados para los distintos usuarios y profesionales (Treitz & Rogan, 2004). A lo largo de los últimos años ha aumentado el interés y la necesidad de disponer de una información de usos y coberturas del territorio fiable y actualizada, siendo numerosos los proyectos de carácter local, nacional e internacional cuyo objetivo es la creación y actualización de bases de datos de usos y ocupación del suelo (Borràs *et al*., 2017).

Los mapas de uso de suelo y vegetación son necesarios para conocer la distribución de los ecosistemas en nuestro territorio y constituyen una herramienta imprescindible en los procesos de planeación. A partir de éstos, se deducen escenarios sobre la pérdida del capital natural o biodiversidad, se generan modelos sobre los posibles efectos del cambio

global y se fundamentan las estrategias de planificación de uso del suelo (Mas *et al*., 2009).

La cobertura terrestre es un rasgo geográfico que expresa las actividades humanas y puede tomarse como referencia para algunas aplicaciones que van desde el monitoreo ambiental, la producción de estadísticas, planeación, inversión, biodiversidad, cambio climático hasta el control de la desertificación (INEGI, 2009), siendo un insumo importante para los investigadores y tomadores de decisiones en materia de recursos naturales y de los servicios que prestan estos a la sociedad.

El cambio de uso y cobertura del suelo, se refiere a la alteración de la superficie del suelo y de su cobertura biótica, el cual contribuye al cambio global: afectando sistemas globales (la atmósfera, el clima mundial, el nivel del mar) o al producirse de forma localizada en lugares suficientes como para sumar un total significativo a nivel mundial, como la pérdida de biodiversidad, la degradación del suelo y el cambio hidrológico (Meyer & Turner II, 1992).

Los procesos de cambio de cobertura y uso de suelo se han convertido en un tema central dentro de la investigación ambiental actual debido a su importancia en los ámbitos gubernamental, académico y social (García & Mas, 2008); sin embargo, los métodos tradicionales usados comúnmente se basan en gran medida en la elaboración manual de descriptores y características, se realizan mediante la interpretación visual de imágenes satelitales, lo cual es costoso, dispendioso e impreciso. Implementar métodos computacionales permite generar la clasificación de coberturas con imágenes satelitales de manera automática, rápida, precisa y económica (Suárez *et al*., 2017).

Esta complejidad conlleva a clasificar el suelo y vegetación con métodos alternativos como el uso de Redes Neuronales Artificiales, que reúnen una serie de técnicas informáticas para realizar funciones de aprendizaje automático mediante códigos de programación computacional o algoritmos para resolver diversos problemas, de forma similar a como lo haría el ser humano.

La precisión de las técnicas manuales convencionales depende de las habilidades de observación y del esfuerzo de los observadores humanos, donde el nivel de incertidumbre y la fiabilidad de los datos obtenidos depende de la calidad de su sistema de clasificación y su confiabilidad (Mas *et al*., 2003). Además, hay diversos métodos para clasificar imágenes, pero no todos son aplicables a la clasificación de la cubierta terrestre (Macedo *et al*., 2010), por eso, en este trabajo se presenta la aplicación y evaluación de un método computacional basado en redes neuronales convolucionales para la clasificación supervisada de uso de suelo y vegetación, con clases definidas de acuerdo al sistema de clasificación de INEGI en la Serie VI (INEGI, 2017).

La presente investigación se realizó para proponer un método alternativo de inteligencia artificial en la clasificación de uso de suelo y vegetación de acuerdo con el sistema de clasificación de INEGI, debido a que a nivel nacional no hay estudios relacionados con este tema, de modo que se propone usar aprendizaje automático con redes neuronales convolucionales para predecir el USV. Las redes neuronales convolucionales tienen suficiente nivel de confianza para ser utilizadas como método alternativo al normalizado para la obtención de clasificación de uso de suelo y vegetación. Por lo que se desarrollará un algoritmo para aprender a reconocer patrones en las imágenes de entrada a la red para la clasificación de cobertura vegetal y uso de suelo; así mismo, se mostrará el

proceso de entrenamiento y validación de la red neuronal.

El objetivo de este trabajo es ambicioso en comparación con otros estudios de teledetección, ya que se utilizaron 22 clases distintas, lo que añade un importante grado de dificultad, cuando habitualmente se utilizan en torno a diez; por ejemplo: Suárez *et al*., (2017) utilizaron 4 clases con 91.02% de exactitud, Hu *et al*. (2018) clasificaron 7 clases y 82% en precisión, Bhosle & Musande (2019) clasifican 16 y 4 clases con precisiones de 97.58% y 79.43%, respectivamente.

El modelo se entrenó para la clasificación de más de 20 clases de USV, permitió establecer patrones sobre el entrenamiento de redes neuronales de convolución, y proveer información para asignar los parámetros de entrenamiento. Este trabajo proveyó información que será de utilidad como guía para el entrenamiento de herramientas de clasificación supervisada con aprendizaje automático y constituye una base para explorar el mundo de la inteligencia artificial en aplicaciones orientadas a la evaluación de escenarios, tomando en cuenta otras variables como cambio climático, biodiversidad, seguimiento de cultivos; de la misma manera, estudios enfocados a modelos de gestión integral de cuencas, estimación de almacenes de carbono o la valoración de servicios ecosistémicos.

La realización de este trabajo también se justifica como herramienta para acotar el tiempo de análisis en la identificación de cambios en la cobertura vegetal y uso de suelo, y poder clasificar en periodos cortos los nuevos límites, a fin de estimar la dinámica del cambio en los procesos naturales y producidos por el hombre.

El alcance de esta investigación implica diseñar un algoritmo de clasificación supervisada

en redes neuronales convolucionales para el entrenamiento de un modelo integrado para la identificación automática de la cobertura vegetal en la cuenca río Atoyac-Salado en el estado de Oaxaca. Este estudio ofrece mostrar una alternativa para la extracción de características de imágenes de satélite con aprendizaje automático.

Los límites de esta investigación se sujetan a las condiciones de diversidad de ecosistemas y desarrollo urbano que presenta la cuenca del río Atoyac-Salado, siendo este un factor determinante por el tipo de orografía y, en particular, la gran diversidad de clases de USV que convergen en ésta. Uno de los problemas comunes del procesamiento de imágenes multiespectrales con fines de clasificación de tipos de cobertura, es la confusión de respuestas espectrales y de atributos (Palacio & Luna, 1994), donde la diferencia entre diversos ecosistemas a simple visto no es tan diferente y se ha clasificado así.

Se delimita esta investigación de acuerdo con la capa de uso de suelo y vegetación de la Serie VI, donde la escala que se utiliza es 1:250000, por lo tanto, al momento de realizar la extracción de muestras, se estima que puede acarrearse un error por el origen de los datos. Además de que los datos de las capas de información digital corresponden a imágenes 2014, y en la investigación actual se usaron imágenes de 2021, donde las condiciones pudieron haber cambiado.

II. OBJETIVOS E HIPÓTESIS

Objetivo general

Diseñar un modelo de inteligencia artificial con redes neuronales convolucionales para clasificación automática con aprendizaje supervisado de los tipos de uso de suelo y vegetación en la cuenca río Atoyac-Salado en el estado de Oaxaca.

Objetivos específicos

1. Entrenar un modelo de clasificación automático de uso de suelo y vegetación con aprendizaje profundo.

2. Estimar la precisión del modelo de clasificación automático con datos de campo y la serie VI de uso del suelo y vegetación de INEGI.

Hipótesis

El aprendizaje profundo con redes neuronales convolucionales tiene la precisión suficiente para identificar patrones en los datos de la reflectancia captada por los sensores remotos a bordo de plataformas satelitales, clasificando cada factor según su grado de influencia, aprendiendo y mejorando el proceso al clasificar el uso de suelo y vegetación.

III. REVISIÓN DE LITERATURA

Uso de suelo y vegetación

El uso de suelo y vegetación abarca la ocupación que se le da a la capa superficial de la Tierra, indicando las actividades que ahí se realizan o la vegetación que predomina en ese sitio en particular. La cubierta vegetal hace referencia a la información del material físico en la superficie de la tierra y el uso del suelo se asocia a las modificaciones hechas sobre esta cobertura por el hombre (Suárez, 2017). El uso de suelo, en su sentido más amplio, son las diferentes formas en que se emplea un terreno y su cubierta vegetal (Trucíos *et al*., 2013).

En México, el Instituto Nacional de Estadística y Geografía (INEGI) se encarga de la generación y actualización de información geográfica sobre recursos naturales, incluyendo Uso del Suelo y Vegetación (USV), el cual se inició hace más de 40 años (INEGI, 2005); y se ha convertido en un insumo importante para apoyar estudios espacio-temporales del comportamiento de las comunidades vegetales presentes en el país, contribuyendo al conocimiento del estado que guarda la cobertura de la tierra en los últimos años (INEGI, 2017).

La dinámica de crecimiento de la población y el desarrollo del país han causado cambios significativos en la distribución y estado de la vegetación y de las áreas agrícolas y urbanas (INEGI, 2017). Desde el año 1978 se han generado mapas de vegetación y de uso del suelo en México en seis series, mostrando la distribución y estado de la vegetación natural, además de que permite disponer de información geoestadística sobre el monitoreo de la cubierta vegetal y los principales usos del suelo, también identificar la

ubicación de las diferentes zonas agrícolas y los cultivos que en ellas se desarrollan, los tipos de ganadería, las actividades forestales, etcétera. El sistema de clasificación de INEGI se enfoca en representar la cartografía de los diferentes tipos de vegetación y el uso del suelo existentes en México (INEGI, 2017). El sistema de información está concebido como un sistema de clasificación *a priori*, estandarizado y jerárquico que combina las características del sistema de clasificación de la vegetación con las del *Land Cover Classification System* de la FAO (INEGI, 2009).

Las series permiten seguir la cobertura variable de 12 ecosistemas vegetales, 58 tipos de vegetación e información sobre su estado sucesional, dando cerca de 200 diferentes categorías. Se considera de interés nacional la Información de Uso del Suelo y Vegetación, según el Sistema Nacional de Información Estadística y Geográfica (INEGI, 2017).

Aunque el cambio de uso de suelo pone en entredicho la permanencia de la vegetación natural, esta constituye un capital natural vital asociado a un conjunto de bienes y servicios estratégicos, valorados positivamente como condición para el desarrollo económico de las sociedades. En el futuro el bienestar de la sociedad dependerá no solo del uso de suelo, sino también del aprovechamiento de la vegetación y sus componentes suelo, agua y biodiversidad (Galicia Sarmiento, 2016).

Clasificación de imágenes de satélite

En las últimas décadas, el crecimiento constante de imágenes digitales como fuente principal de representación de la información para aplicaciones científicas ha hecho de la clasificación de imágenes una tarea desafiante (Fidel y Cort, 2019). La clasificación de

imágenes de satélite reside en identificar regiones del medio físico de la tierra captados en imágenes de satélite para estimar que hay en un área determinada ya sea usando observación visual o métodos computacionales. Consiste en separar elementos de la imagen considerados como individuos dentro de un conjunto de clases dadas, en función de una serie de variables cuantitativas, que midan la semejanza o diferencia entre individuos y las clases. La clasificación de escenas de alta resolución, tiene como objetivo clasificar las subregiones extraídas de imágenes que cubren múltiples tipos de cobertura terrestre u objetos terrestres en diferentes categorías semánticas (Hu *et al*., 2015).

De acuerdo con Pascual Ramírez *et al*., (2010) el estudio de su firma espectral es uno de los métodos eficaces para distinguir diferentes tipos de objetos presentes en una imagen obtenida por un sensor a bordo de alguna plataforma satelital. Sin importar el procedimiento de clasificación utilizado, algunos individuos resultan inclasificables, ya sea por su baja probabilidad de pertenecer a cualquier clase o porque dos o más clases con muy alta probabilidad se disputan el individuo. Uno de los problemas comunes del procesamiento de imágenes multiespectrales con fines de clasificación de tipos de cobertura, es la confusión de respuestas espectrales y de atributos (Palacio & Luna, 1994).

Las imágenes de satélite se estudian como la representación visual de la reflectancia de la tierra adquirida por el sensor particular, en donde se han desarrollado procedimientos de representación de esta información, como las firmas espectrales y los índices de vegetación (Suárez *et al*., 2017). La extracción de información útil de imágenes satelitales, mediante clasificación, es uno de los problemas técnicos más importantes de

la teledetección, en la determinación de las características y fenómenos que tienen lugar en la superficie de la tierra; sin embargo, es un proceso complejo debido al procesamiento simultáneo de datos históricos y actuales de forma masiva, la interacción en tiempo real de escenarios y datos ambientales espacialmente diversos (Wang *et al*, 2022).

Inteligencia artificial

La Inteligencia Artificial (IA) es una rama de investigación dentro del campo de la informática que nace en el siglo pasado. Desde al menos la década de 1950, se han realizado intentos para modelar artificialmente los mecanismos de procesamiento de información de las neuronas, principalmente con el desarrollo de perceptrones; sin embargo, a fines de la década de 1980, el desarrollo de redes neuronales multicapa y la popularización de la retropropagación resolvieron muchas de las limitaciones de los primeros perceptrones (Macpherson *et al*., 2021).

La investigación sobre IA se ha centrado principalmente en la creación de máquinas que puedan percibir, aprender y razonar, con el objetivo general de crear un sistema de inteligencia general artificial que pueda emular la inteligencia humana (Macpherson *et al*., 2021). Las capacidades emergentes de inteligencia artificial probablemente abordarán todos los entornos y actividades organizacionales, basando sus herramientas en varios enfoques para simular la inteligencia humana, incluido el aprendizaje automático, las redes neuronales y el aprendizaje profundo (Jarrahi *et al*., 2022).

Aprendizaje automático

A nivel fundamental, el aprendizaje automático (*Machine learning* en inglés, ML) es una

categoría de inteligencia artificial que permite a los ordenadores pensar y aprender por sí mismos (Alzubi *et al*., 2018). Las máquinas son entrenadas con altas dosis de datos para aprender a ejecutar diferentes tareas de forma autónoma (Porcelli, 2020). Se trata de hacer que las computadoras modifiquen sus acciones con el fin de mejorarlas para conseguir más precisión, donde la precisión se mide en términos del número de veces que las acciones elegidas son correctas (Alzubi *et al*., 2018). Lo anterior implica la capacidad del *software* o una máquina para mejorar el rendimiento de las tareas a través de la exposición a datos y experiencia, aprendiendo primero del conocimiento de los datos a los que está expuesto y luego aplica este conocimiento para proporcionar predicciones sobre datos futuros (Luxton, 2016). En lugar de programar reglas en una computadora y esperar el resultado, con *machine learning*, la máquina aprenderá esas reglas por cuenta propia (Porcelli, 2020).

El aprendizaje automático permite que las computadoras imiten y adapten el comportamiento humano. Usando el aprendizaje automático, cada interacción, cada acción realizada, se convierte en algo que el sistema puede aprender y usar como experiencia para la próxima vez (Alzubi *et al*., 2018). Dentro de los algoritmos del aprendizaje automático, destacan dos tipos de aprendizaje que se distinguen por el método de entrada de datos en: supervisado y no supervisado, siendo el primero cuando el programa se "entrena" en un conjunto predefinido de "ejemplos o conjuntos de entrenamiento"; en tanto el segundo es cuando el programa recibe datos, pero debe descubrir patrones y relaciones en esos datos (Luxton, 2016).

En el aprendizaje supervisado, el objetivo es modelizar una función que permita generar los resultados esperados, el modelo de entrenamiento se realiza con datos conocidos de

la clase a identificar (Suárez *et al*., 2017), el cual se caracteriza por disponer de un conjunto de patrones sobre los que se puede definir un criterio de evaluación, donde existen valores de entrada asociados a salidas deseadas. El aprendizaje no supervisado no requieren información de etiqueta de clases (Verona *et al*., 2016) y se realiza agrupando datos con características o patrones similares en un número de clases dadas.

La entrada de un algoritmo de aprendizaje son los datos de entrenamiento, que representan la experiencia, y la salida es algún tipo de experiencia, que suele adoptar la forma de otro programa informático que puede realizar alguna tarea (Shalev & Ben, 2014). El aprendizaje desarrollado por el entrenamiento de los modelos supervisados ofrece un conjunto de experiencias que permiten realizar predicciones apropiadas del comportamiento de nuevos datos que aún no han pasado por el modelo.

Durante la fase de aprendizaje, se aplica una función de transferencia a través de una serie de iteraciones, de manera que los valores predichos se comparan con los valores observados (Bocco *et al*., 2007). Es decir, a partir de unos datos y unas respuestas tomadas como experiencia, el modelo crea reglas para esas circunstancias; por lo que, usa esas reglas que no han sido explícitamente reguladas por programadores para aumentar sus probabilidades de éxito en su cometido, a medida que se ejecuta y va recibiendo más datos de entrada experimenta con ello el aprendizaje. Los modelos generados pueden ser usados para realizar clasificación y predicción, uno de los modelos más populares dentro del aprendizaje supervisado son las redes de neuronas artificiales.

Aprendizaje Profundo

El Aprendizaje Profundo es un conjunto de algoritmos que se ha convertido gradualmente en el enfoque computacional más utilizado en el campo del Aprendizaje Automático (Alzubaidi *et al*., 2021). Está basado en redes neuronales artificiales con un número elevado de capas que pueden modelar abstracciones de alto nivel en los datos, utilizando arquitecturas computacionales que admiten operaciones no lineales.

El aprendizaje profundo ha experimentado un crecimiento exponencial en los últimos años debido a su versatilidad, alto rendimiento, alta capacidad de generalización y usos multidisciplinares, entre muchas otras cualidades (Anaya *et al*., 2021), estando presentes en tareas tan diversas como reconocimiento facial, reconocimiento de voz, traducción de textos o la conducción autónoma. Otra de sus capacidades más relevantes es que ha eliminado la necesidad de diseñar y calcular características complejas de manera manual en la extracción de características automatizada a partir de los patrones disponibles, lo que reduce los sesgos introducidos al diseñar los algoritmos clásicos de Aprendizaje Automático, donde se tiene que analizar y decidir sobre las características más relevantes para el problema a resolver. Gran parte del crecimiento del aprendizaje profundo se debe principalmente a los avances realizados en la visión artificial, siendo uno de los algoritmos más utilizados en este campo el de convolución, del que se deriva la red neuronal convolucional, un sistema inspirado en la corteza visual primaria (Alzubaidi *et al*., 2021).

Redes neuronales artificiales

Las redes neuronales artificiales (RNA) son algoritmos de modelos matemáticos que

forman parte del *deep learning,* donde una red neuronal es un modelo computacional, paralelo, compuesto de unidades procesadoras adaptativas (llamadas neuronas) con una alta interconexión entre ellas (Soria & Blanco, 2001). La red neuronal artificial es un arreglo que se puede aplicar para simular el comportamiento de un objeto sin una solución algorítmica simplemente utilizando los datos experimentales disponibles (Mursina, 2010). Estas son una extensión de los métodos estadísticos si se ven como un proceso de reconocimiento de patrones, el cual es usado para dos posibles objetivos, clasificación y predicción. En la clasificación, una red neuronal puede ser usada para diferenciar unos elementos de otros, puede ir desde la solución de funciones lógicas, a la clasificación de imágenes.

Las redes neuronales artificiales son algoritmos computacionales de procesamiento de la información que imitan el funcionamiento del cerebro humano y al mismo tiempo intenta lograr un mejor rendimiento en la resolución de problemas difíciles, siendo una abstracción y una simplificación de las redes neuronales biológicas, siendo su deficiencia más importante la incapacidad de explicar sus decisiones (Mursina, 2010). Están formadas por neuronas que se conectan a otras neuronas mediante valores variables llamados pesos sinópticos, los cuales están asociados a cada conexión, son números reales que determinan la fuerza de la conectividad entre las dos neuronas conectadas, y son fundamentales ya que, junto con el patrón de conexión, determinan la propagación de la señal a través de la red (Vandeginste, 1998).

Redes neuronales convolucionales

Las redes neuronales convolucionales (*Convolutional Neural Networks* o CNN, en inglés)

son un tipo concreto de redes neuronales que típicamente se utiliza para el procesamiento de imágenes. Las redes neuronales convolucionales son el tipo de red de Aprendizaje profundo más utilizado (Yao *et al*., 2019). Las CNN combinan las características de bajo nivel dentro de características abstractas de alto nivel con transformaciones no lineales, lo que le permite tener la capacidad de aprender la representación semántica de las imágenes (Fidel y Cort, 2019).

A pesar de que el concepto teórico de las CNN se desarrolló en la década de 1980 y 1990, no fue hasta el año 2012 cuando comenzaron a popularizarse con la publicación de AlexNet (Krizhevsky *et al*., 2012), una CNN diseñada para realizar clasificación de objetos en imágenes y que en su momento superó los métodos que definían el estado del arte de ese campo. La principal ventaja de las CNN en comparación con sus predecesores es que detecta automáticamente las características significativas sin supervisión humana, lo que la convirtió en la más utilizada (Alzubaidi *et al*., 2021).

La arquitectura más típica de red CNN (Figura 1) consiste del empleo de capas convolucionales y capas de submuestreo alternadas seguidas de una capa completamente conectada (López Pacheco, 2021).

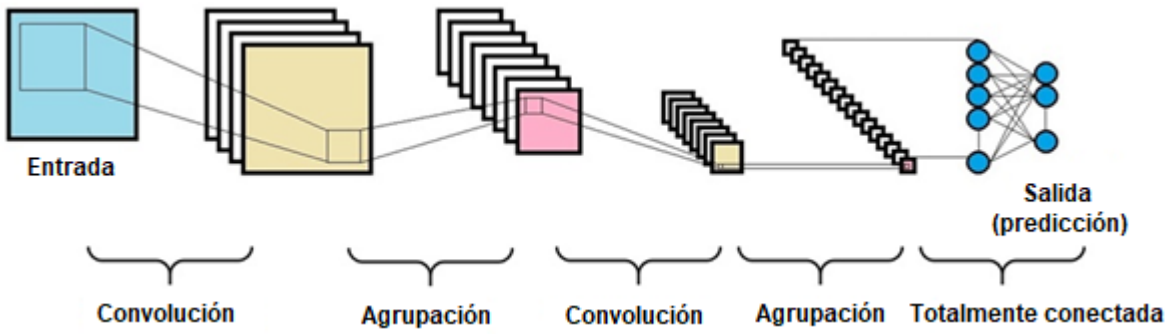

Figura 1.- Arquitectura de una red neuronal convolucional.

Fuente: Adaptado de (López Pacheco, 2021).

Python

Python es un lenguaje de programación potente y fácil de aprender creado por Guido van Rossum a principios de los años 90 (González Duque, 2000). Se trata de un lenguaje interpretado o de *script*, con escritura (*typing*) dinámica, multiplataforma y un simple pero efectivo sistema de programación orientado a objetos. Python es actualmente el lenguaje más popular, Python fue el que más creció en los últimos 5 años (11%) (https://pypl.github.io/PYPL.html, consultado 04/05/2022). Además, es uno de los lenguajes de programación más usado dentro del campo de la inteligencia artificial y la ciencia de datos, debido a su elegante sintaxis y su escritura dinámica, junto a su naturaleza interpretada lo convierten en un lenguaje ideal para la secuencia de comandos (*scripting*) y desarrollo rápido de aplicaciones en muchas áreas, para la mayoría de plataformas, con gran cantidad de librerías auxiliares de alto nivel que permiten un manejo sencillo de grandes volúmenes de datos, creación de redes neuronales, etc. (https://docs.python.org/3/tutorial/index.html, consultado 07/05/2022).

La gran cantidad de librerías más importantes para el desarrollo de aplicaciones de inteligencia artificial (Scikit-learn, Tensorflow, PyTorch, Keras o Theano entre otras) se encuentran disponibles para este lenguaje, lo que hace que desarrollar una aplicación en Python sea sencillo y muy rápido. La versión actual de este lenguaje con una gran popularidad es Python 3.X, motivo por el cual este trabajo se realiza con la versión 3.7, con soporte actualizado por parte de las librerías necesarias para el desarrollo https://pythoninsider.blogspot.com/2020/04/python-2718-last-release-of-python-2.html.

TensorFlow

TensorFlow es un sistema de aprendizaje automático que opera a gran escala y en entornos heterogéneos, utiliza gráficos de flujo de datos para representar el cálculo, el estado compartido y las operaciones que mutan ese estado, que permiten a los usuarios programar y entrenar eficientemente redes neuronales .(Abadi *et al*., 2016).

TensorFlow, es creado originalmente por investigadores de Google, es el más popular entre la plétora de bibliotecas de aprendizaje profundo (Pang *et al*., 2020). Es una de las librerías más populares dentro de su área de manejo de tensores, usada principalmente dentro del campo del Aprendizaje Automático.

TensorFlow es una biblioteca de software flexible y escalable para realizar cálculos numéricos utilizando gráficos de flujo de datos. Esta biblioteca y las herramientas relacionadas permiten a los usuarios programar y entrenar eficientemente programar y entrenar redes neuronales y otros modelos de aprendizaje automático (https://www.tensorflow.org/?hl=es-419, consultado 05/05/2022).

Los motivos de la elección de Tensorflow frente a los demás *backends* soportados por Keras son: su mayor popularidad y que dispone de implementaciones que permiten su ejecución en una GPU con una potencia de cálculo y memoria dedicada suficiente, pudiendo alcanzar velocidades más de 10 veces mayores con respecto a una CPU. TensorFlow facilita y acelera en gran medida la investigación y la aplicación de los modelos de redes neuronales (Pang *et al*., 2020).

Keras

Keras es una API diseñada para seres humanos, no para máquinas. Keras sigue las mejores prácticas para reducir la carga cognitiva: ofrece APIs consistentes y simples, minimiza el número de acciones del usuario requeridas para los casos de uso comunes, y proporciona mensajes de error claros y procesables. También cuenta con una amplia documentación y guías para desarrolladores https://keras.io/, consultado 09/05/2022). Keras es un marco de aprendizaje profundo para Python que proporciona una manera conveniente de definir y entrenar casi cualquier tipo de modelo de aprendizaje profundo. Keras se desarrolló inicialmente para investigadores, con el objetivo de permitir una experimentación rápida (Chollet, 2015).

Keras es una API de aprendizaje profundo escrita en Python, que se ejecuta sobre la plataforma de aprendizaje automático TensorFlow. Fue desarrollado con un enfoque en permitir la experimentación rápida. Ser capaz de pasar de la idea al resultado lo más rápido posible es clave para hacer una buena investigación, reduciendo la carga cognitiva del desarrollador para que pueda concentrarse en las partes del problema que realmente importan (Chollet, 2015). Keras está bien estructurado y documentado, con principios de diseño que lo hacen modular y extensible, siendo fácil construir modelos complejos (Peris y Casacuberta, 2018).

NumPy

NumPy se creó para llevar la funcionalidad de computación numérica a Python, una iniciativa que data de mediados de la década de los 1990s, pero se liberó oficialmente en 2006 (McCaffrey, 2020). NumPy es una biblioteca que proporciona un medio flexible

medios para definir y manipular vectores y matrices de mayor dimensión. Esta es una extensión fundamental para el uso de Python en entornos científicos, que permite cálculos rápidos utilizando matrices, vectores y contenedores similares (McClarren, 2018). Los motivos para su uso en la presente investigación fueron su capacidad para trabajar con vectores o arrays y matrices, permitiendo realizar operaciones con ellos de manera eficiente y sencilla.

Matplotlib

Matplotlib es uno de los paquetes de trazado 2D de Python más populares, que admite una amplia gama de tareas de trazado, desde simples gráficos exploratorios hasta cifras de calidad de publicación con diseños, esquemas de color y anotaciones altamente personalizados (McClarren, 2018). Matplotlib se utiliza para el desarrollo de aplicaciones, secuencias de comandos interactivas y generación de imágenes en interfaces de usuario y sistemas operativos, orientada a la generación de gráficos a partir de datos contenidos en listas o arrays de Numpy, y permite la exportación de los gráficos generados en múltiples resoluciones y formatos (Hunter, 2007).

En este trabajo se usó para para generar las imágenes de los resultados debido a que permite la visualización de resultados numéricos con sólo unas pocas líneas de código.

Entorno de desarrollo

Para gestionar los distintos paquetes de manera eficiente se usó la herramienta Anaconda, la cual permite crear distintos entornos de desarrollo (IDE) de Python, así como sincronizar programas auxiliares como IDEs con esos entornos. El IDE utilizado para este modelo ha sido Jupyter. Esta aplicación puede ser alcanzada a través de un

navegador web como la dirección IP y el puerto del host, normalmente 127.0.0.1 y 8888, respectivamente (McCaffrey, 2020). El cual es un proyecto de código abierto Jupyter pretende generar una plataforma computacional que permita utilizar diferentes lenguajes, su nombre proviene de unir los tres lenguajes quizás más populares en cálculo científico: Julia, Python y R (https://eprints.ucm.es/id/eprint/48304/1/ManualJupyter.pdf, consultado 08/05/2022).

Arquitecturas cnn

Las redes neuronales convolucionales presentan una arquitectura multicapa, donde cada capa está constituida por un número determinado de convoluciones con su respectiva función de activación no lineal (López Pacheco, 2021).La profundidad de la red está determinada por el número de capas, donde un mayor número de capas permite explorar más opciones de identificación de patrones de alto nivel, añadiendo en cada capa un incremento significativo del coste del proceso de aprendizaje.

La arquitectura CNN consta de una serie de capas. Las capas convolucionales y las capas de agrupación construyen las primeras etapas. Las capas convolucionales generan mapas de entidades, cada elemento del cual se obtiene calculando un producto de punto entre la región local (campo receptivo) a la que está conectado en los mapas de entidades de entrada y un conjunto de pesos (también llamados filtros o núcleos). En general, se aplica una función de activación no lineal por elementos a estos mapas de entidades, siendo las dos principales funciones utilizadas en este tipo de red la ReLU o tanh (Szegedy et al., 2015).

Las capas de agrupación realizan una operación de reducción de muestreo a lo largo de

las dimensiones espaciales de los mapas de entidades mediante el cálculo del máximo en una región local. Las capas totalmente conectadas (FC) finalmente siguen varias capas convolucionales y de agrupación apiladas, y la última capa totalmente conectada es una capa *Softmax* que calcula las puntuaciones de cada clase definida. Las CNN transforman la imagen de entrada de los valores de píxeles originales a las puntuaciones de clase finales a través de la red de una manera *feedforward*. Los parámetros de las CNN (es decir, los pesos en las capas convolucional y FC) se entrenan con el descenso clásico del gradiente estocástico basado en el algoritmo de retropropagación.

Capas de convolución

En la arquitectura CNN una capa convolucional es el bloque de construcción principal, contiene un conjunto de filtros (núcleos o kernels), cuyos parámetros deben aprenderse a lo largo del entrenamiento, donde el tamaño de los filtros suele ser más pequeño que la imagen real (Mostafa & Wu, 2021). La imagen de entrada, expresada como métricas N-dimensionales, se convoluciona con estos filtros para generar el mapa de características de salida (Alzubaidi *et al*., 2021).

Los filtros realizan la operación de convolución sobre la imagen para crear un mapa de activación que se genera repitiendo este proceso para cada elemento de la imagen de entrada. El volumen de salida de la capa convolucional se genera apilando los mapas de activación de cada filtro a lo largo de la dimensión de profundidad (Mostafa & Wu, 2021). La operación de convolución consiste en calcular la suma del producto elemento a elemento entre ambas estructuras, como se muestra en la Figura 2.

La convolución consiste en tomar grupos de píxeles cercanos de la imagen de entrada e

ir haciendo el producto escalar contra una pequeña matriz que se llama filtro o *kernel* con el que realiza la operación de convolución sobre la imagen. La entrada de la capa convolucional se modifica matemáticamente para detectar algunos tipos específicos de características de la imagen; sin embargo, tienen la facilidad de devolver la imagen modificada, enfocar la imagen, desenfocar la imagen y detectar los bordes (Arul, 2021). En la operación de convolución este filtro recorre todas las neuronas de entrada (de izquierda a derecha y de arriba a abajo), y consiste en calcular la suma del producto elemento a elemento entre ambas estructuras. El resultado es un valor numérico que indica la probabilidad de que la característica que representa dicho filtro esté presente en la región de la imagen que está siendo analizada, generando una nueva matriz de salida, la cual es la nueva capa de neuronas ocultas.

Una convolución está en función del tamaño del *kernel* y del tamaño salto (*stride*) que haga con la convolución, dando como resultado una matriz de un tamaño que depende de valores. El desplazamiento del filtro sobre la imagen se realiza ordenadamente, en ambos ejes, con un parámetro de paso o *stride*. Normalmente el paso suele ser de una posición, por lo que el filtro operaría con la región determinada por cada elemento. Para que una operación de convolución con saltos de un píxel resulte una imagen de la misma dimensión que la original, haría falta poner un relleno que permita controlar los valores a insertar en las zonas de la imagen de entrada para las cuales la superposición del filtro no sería posible.

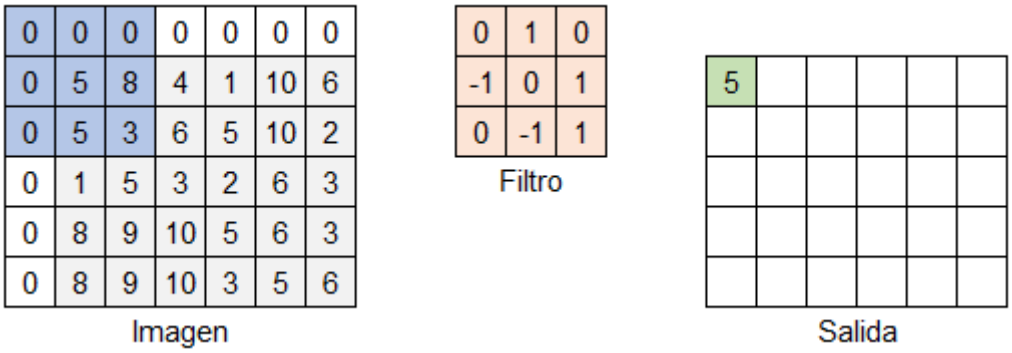

Figura 2.- Convolución con filtro de 3 x 3.

Fuente: elaboración propia.

Al principio los valores del filtro serán aleatorios, que mediante la retropropagación se ajustan los valores como en toda red neuronal. Las capas convolucionales disponen de un parámetro adicional que es el número de filtros a calcular en esa capa de la red. El objetivo de las capas convolucionales es por tanto encontrar de manera automatizada los filtros que extraigan las mejores características para los datos de entrada disponibles. El mapeo de características surge al convolucionar los *kernels* con la imagen, donde se obtiene una nueva imagen con ciertas características de la imagen original que ha sido filtrada. Para obtener el valor de esa neurona, la suma ponderada de píxeles se pasa a una función de activación que determina los valores que serán propagados hacia la siguiente capa de la red. La más usada es la función ReLU, ya que es la que mejores resultados experimentales ha dado.

En este punto, la dimensión de las características aún no se ha reducido en las nuevas matrices, que muestran características destacadas, se conserva la imagen original con la misma dimensión. Sin embargo, es conveniente reducir la dimensión para no aumentar significativamente el número de parámetros de la red. Para dejar las características más importantes que detectó cada filtro. Se usa una operación llamada submuestreo

(*Subsampling*). El tamaño de la matriz que se obtiene depende de la dimensión de la reducción.

Los parámetros que definen la convolución mediante filtros son:

- Tamaño de la imagen de entrada.
- Tamaño del filtro convolucional.
- Número de canales de entrada y salida (deben ser los mismos).
- Paso o *stride*, que constituyen los píxeles de desplazamiento entre cada aplicación del filtro sobre la entrada.
- Relleno o *padding*, que permite controlar los valores a insertar en las zonas de la imagen de entrada para las cuales la superposición del filtro no sería posible.

Pooling

Para acelerar el proceso de aprendizaje, reducir el consumo de memoria, e incrementar la capacidad de generalización del modelo, es común utilizar un operador de submuestreo. La capa de submuestreo se utiliza para reducir la resolución de las características. Esta capa reduce el número de conexiones entre las capas convolucionales, por lo que también reducirá el tiempo de cálculo. En Hay tres tipos de agrupación: agrupación máxima, agrupación mínima y agrupación media. En cada caso, la imagen de entrada de entrada se divide en espacios bidimensionales no superpuestos (Deepan y Sudha, 2020). siendo la más común la del valor máximo, o *max pooling*, aunque en este trabajo de eligió el valor de la media *Average pooling* (Figura 3). En esta técnica, se recorren regiones de la imagen de acuerdo con un paso establecido y, en cada paso, el valor promedio dentro de la ventana se agrupa en una matriz de salida.

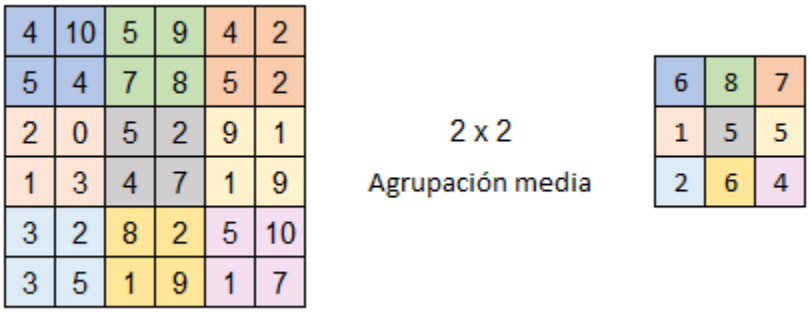

Figura 3.- Submuestreo.

Fuente: elaboración propia.

El objetivo principal de esta etapa es la reducción de la dimensión del resultado de la etapa anterior, de forma que, al reducir el número de parámetros, es posible ampliar la profundidad de las capas y obtener características más complejas. Este tipo de capa toma una ventana de la entrada y la reduce a un sólo valor, tomando la media de los datos de entrada. Esta capa se utiliza para submuestrear la entrada, eliminando parte de la información espacial.

Las capas de *pooling* tienen como objetivo la simplificación de los datos mediante una reducción en la dimensión de estos. Esta reducción consigue mejorar el costo computacional de la red, al reducir el número de parámetros que deben ser aprendidos, lo que también ayuda a controlar el sobreentrenamiento. En suma, las capas de *pooling* son muy comunes en las redes CNN y son colocadas a continuación de las capas convolucionales, reduciendo el tamaño de las características convolucionadas para reducir la potencia requerida para la reducción de la dimensionalidad, y extrae características interesantes que no varían con la rotación (Mitta *et al*., 2021).

Los parámetros de este tipo de capa son:

- Dimensión del elemento de pooling.
- Paso o stride.
- Relleno o padding.

Dropout

El abandono es una técnica que aborda estos dos problemas. Evita el sobreajuste y proporciona una forma de combinar de forma eficiente un número exponencial de arquitecturas de redes neuronales diferentes. El término "*dropout*" se refiere a la eliminación de unidades (ocultas y visibles) en una red neuronal. Con el término "dropout" nos referimos a la eliminación temporal de una unidad de la red, junto con todas sus conexiones entrantes y salientes (Srivastava *et al*., 2014).

Es una técnica de regularización que permite reducir el sobreentrenamiento y consiste en desactivar neuronas de manera aleatoria con probabilidad $1 - p$ o dejarlas activadas con probabilidad p (Sanjurjo, 2020). En la última década, Dropout ha surgido como un método poderoso y simple para entrenar redes neuronales que previenen la coadaptación al omitir neuronas de manera estocástica (Herlau *et al*., 2022).

La reducción del número de neuronas reduce la complejidad computacional de la red, pero implica una ralentización en el proceso de convergencia. Sus efectos son mayores en redes con muchos parámetros y con conjuntos de entrenamiento pequeños, ya que son algunas de las situaciones que más favorecen el sobreentrenamiento de las redes (Sanjurjo, 2020).

Capas flatten

El proceso de *Flattening* solo modifica la estructura de los datos, no introduce modificaciones en sus valores, consiste en convertir las salidas de las capas convolucionales, con forma de tensores, a una representación en una dimensión. El *Flattening* es necesario para poder trabajar con capas que reciben vectores como entrada, como pueden ser las capas totalmente conectadas (Sanjurjo, 2020).

La capa totalmente conectada

Al final de las combinaciones de las capas convolucionales y de submuestreo, se utilizan capas completamente conectadas donde cada elemento corresponde a una neurona y el número total de neuronas en estas capas corresponderá al número de clases que se desean predecir (López Pacheco, 2021). La capa totalmente conectada o *fully-connected* es la red más básica, la cual conecta a una o más capas completamente conectadas, también conocidas como capas densas, en las que cada entrada está conectada a cada salida por un peso que se puede aprender (Yamashita *et al*., 2018). En este tipo de capas, todas las entradas se conectan a todas las salidas, de forma que la operación utilizada en este caso es una multiplicación matricial. La capa totalmente conectada generalmente aprende características no lineales de alto nivel de convolución y capa de agrupación (Mitta *et al*., 2021).

Función de activación

Durante el proceso de aprendizaje de una red neuronal se deben aplicar otros factores como las reglas de activación y desactivación de las neuronas, se trata de una regla que da el efecto de las entradas en la activación de la unidad. El propósito de la función de

activación es introducir no linealidad en el funcionamiento de la red, ampliando su capacidad para resolver problemas.

Las funciones de activación son funciones matemáticas, utilizadas por las capas convolucionales y totalmente conectadas para introducir la no linealidad en los datos, permitiendo modelizar funciones no triviales. La unidad lineal rectificada (ReLU, por sus siglas en inglés) es una de las funciones de activación estándar y populares en los últimos años (Deepan y Sudha, 2020). Esta función (Figura 4) se ve menos afectada por el problema del desvanecimiento del gradiente al definirse como una rampa de pendiente 1, lo cual permite pasar el error entre las capas.

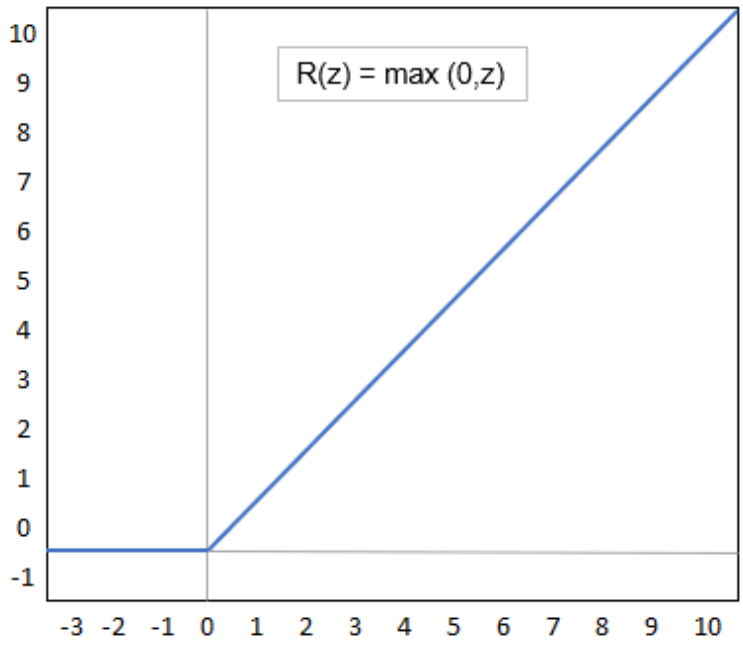

Figura 4.- Función de activación Unidad lineal rectificada.

Fuente: adaptado de (Chollet, 2015).

Otra de las ventajas que presenta, es la simplicidad del cálculo, lo cual hace que su evaluación sea más rápida y se define como:

$$R(z) = max\,(0, z)$$

Significa que algún valor menor que 0 en la matriz z será reemplazado por 0 para tener una matriz cuyo valor mínimo sea 0.

Entrenamiento

Cuando se crea la Red Neuronal, los pesos de las conexiones son aleatorios y se irán ajustando durante el entrenamiento hasta llegar a los valores que minimicen la función de error de dicha red.

Las redes neuronales cuentan con dos fases, entrenamiento y predicción. Para el ajuste de los pesos de las conexiones (fase de entrenamiento), se necesitan dos conjuntos de datos, uno de entrenamiento y otro de validación. Al comienzo la red asigna pesos puestos al azar. Mediante el conjunto de datos de entrenamiento se procede al ajuste de los datos minimizando el error producido por estos con una función de coste que cuantifica el error cometido a la salida de la red y un optimizador que reduce lo más posible el valor de esta función. El optimizador calcula el gradiente de la función de coste, para modificar ligeramente los pesos. Con los pesos actualizados, se computa nuevamente la salida de la red y se repite el proceso descrito. Los datos de entrenamiento se pueden introducir varias veces llamando a cada reiteración época. Una vez realizas todas las épocas deseadas, habiendo ajustado los pesos de forma que los datos de entrenamiento son predichos de forma correcta, se introducen los datos de validación. Si la predicción de los datos de validación es correcta, se da por finalizada la fase de entrenamiento. (García Sánchez et al., 2019).

Además, existe un parámetro que afecta tanto a la rapidez con que avanza el

entrenamiento como a la convergencia de los pesos de la red, este parámetro es la tasa de aprendizaje o *learning rate*. Si la *learning rate* es muy pequeña, el entrenamiento evolucionará de forma muy lenta, los pesos apenas cambiarán y es posible que la red se quede en un mínimo local y no aprenda más https://www.cienciadedatos.net/documentos/py35-redes-neuronales-python.html, consultado 02/05/2022).

Por el contrario, si la tasa es demasiado elevada, es posible que las modificaciones de los pesos sean demasiado grandes, haciendo que estos oscilen a lo largo del entrenamiento sin llegar a converger. Al terminar un entrenamiento, es necesario comprobar que la red que se ha alcanzado se comporta de la forma esperada al recibir unos datos que no sean los de entrenamiento, para ello, es necesario utilizar un conjunto distinto con el que se probará el funcionamiento de la red. Al hacer esta comprobación hay tres posibles resultados.

1. Red sobreajustada: El sobreajuste o *underfitting* se tiene cuando el porcentaje de acierto es bajo tanto para los datos de entrenamiento como para los datos de prueba. La red no ha sido bien entrenada y muchas veces se solucionará aumentando el número de épocas del entrenamiento.
2. Red correctamente entrenada: el porcentaje de acierto será alto tanto para la fase de entrenamiento como para la de prueba. "Esto quiere decir que la generalización de la función objetivo se ha conseguido con éxito, y el problema queda resuelto".
3. Red sobreentrenada: El *overfitting* o sobreentrenamiento se identifica con un porcentaje de acierto muy alto en la fase de entrenamiento, pero sensiblemente más bajo en la de prueba. En este los pesos se han ajustado demasiado a los datos de

entrenamiento y no funcionan correctamente al utilizarlos para unos datos nuevos.

Para evitar el *overfitting*, hay una gran cantidad de técnicas denominadas de regularización que buscan soluciones a este problema. Una de las más utilizadas es *Dropout.*

Optimizador Adam

El optimizador se encarga de realizar el cálculo del error final y de los errores intermedios en el proceso de entrenamiento. A partir de estos errores, el optimizador determina como modificar los pesos de la red para mejorar el modelo. El optimizador Adam (Kingma & Ba, 2014) es uno de los más avanzados actualmente, incorporando ventajas de otros optimizadores, entre sus características están la existencia de diferentes tasas de aprendizaje para diferentes parámetros de la red, la adaptación de la tasa de aprendizaje a partir del cálculo de dos estimaciones de momentos de donde recibe su nombre, un número reducido de parámetros a ajustar y unos valores por defecto que suelen dar buen resultado en la mayoría de problemas. En estos momentos, no hay consenso sobre cuál de los métodos vistos es más efectivo a la hora de enfrentarse a un problema. La elección suele basarse en los conocimientos del usuario sobre los métodos y con cuál de ellos se siente más cómodo.

Función de pérdida

La función de pérdida o coste define la función a minimizar durante el entrenamiento de la red neuronal. Su valor indica el nivel de error entre la salida de la red y el valor real. En el caso de la clasificación, las salidas son de tipo categórico. La función de pérdida más usada para problemas de clasificación multiclase es la entropía cruzada categórica

(*categorical cross-entropy*, en inglés). El valor de esta función disminuye cuando la clase predicha con mayor probabilidad es la de la clase esperada.

Medidas de rendimiento

Para el desempeño de un modelo se usan las métricas de evaluación, de manera común se utilizan seis métricas de evaluación: exactitud (*accuracy*), precisión (*precision*), recuperación (*recall*), puntaje F1 (F1 *score*), matriz de confusión y curva de características operativas del receptor (ROC) (Deepan & Sudha, 2020). Debido a que existen múltiples términos en español para los mismos conceptos y que su uso más frecuente es en inglés, se usará este último idioma para referirse a estas métricas y evitar confusiones.

Exactitud (Accuracy)

Se define como la proporción de casos acertados sobre el total de predicciones.

$$accuracy = \frac{VP + VN}{VP + FP + VN + FN} \quad \cdots\cdots\cdots\cdots\cdots\cdots (1)$$

Siendo VP la cantidad de positivos clasificados como tal, VN la cantidad de negativos clasificados como tal, FP la cantidad de negativos clasificados como positivos y FN la cantidad de positivos clasificados como negativos.

Esta medida da la proporción de clases bien clasificadas y el total de datos, dando una idea sobre el rendimiento general del sistema evaluado. El rango de valores para esta medida estará entre 0.5 y 1.

Precisión (precision)

La métrica de *precision* se define como la proporción de elementos positivos correctamente clasificados para una determinada clase.

$$precision = \frac{V\ P}{V\ P\ +\ F\ P} \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (2)$$

Recall (tasa de verdaderos positivos) o sensibilidad

La métrica de *recall* puede definirse como la proporción de elementos positivos detectados para una determinada clase.

$$recall = \frac{V\ P}{V\ P\ +\ F\ N} \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (3)$$

F1-Score

La métrica *F1-Score* (F1) es la media armónica de *precision* y *recall*.

$$F1 = \frac{2\ *\ precision\ *\ recall}{precision\ +\ recall} \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (4)$$

IV. MATERIALES Y MÉTODOS

Localización de la zona de estudio

La cuenca río Atoyac-Salado, se localiza en el sureste de la República Mexicana, en la parte central del estado de Oaxaca (Figura 5), entre los paralelos 16°49'25.86" y 17°11'34.09" de latitud norte y entre 96°17'23.60" y 96°43'41.66" de longitud oeste, comprende desde el nacimiento del río Salado hasta la estación hidrométrica Oaxaca, drena una superficie de 1,192.98 kilómetros cuadrados y se encuentra delimitada al Norte por la Región Hidrológica Número 28 Papaloapan (DOF, 2017).

El río Atoyac tiene sus orígenes al sur del Municipio de San Francisco Telixtlahuaca, aproximadamente a 8 kilómetros de la cabecera municipal donde lleva el nombre de río Nariz, a una altitud aproximada de 2,418 m, su dirección es hacia el sur recibiendo varios nombres, al cruzar el poblado de Santiago Suchuilquitongo, al sur del Municipio de San Pablo Huitzo, toma el nombre de río Atoyac, continúa con ese nombre y cruza la ciudad de Oaxaca de Juárez hasta la estación hidrométrica Oaxaca a una altitud aproximada de 1,500 m, con una longitud aproximada de 43.4 kilómetros. A 1.5 kilómetros aguas abajo de la estación hidrométrica Oaxaca, se le incorpora otro río que lleva el mismo nombre y que nace al sur del Municipio de Villa Díaz Ordaz, estado de Oaxaca, donde toma el nombre de Arroyo Grande, a altitud aproximada de 2,600 m; en su trayecto también recibe varios nombres hasta su confluencia con el río del mismo nombre, en general mantiene una dirección hacia el Oeste, con una longitud acumulada de 66 kilómetros. El río Atoyac continúa su dirección hacia el sur del estado cruzando Zimatlán, Tlapacoya, Amatengo y Coatlán hasta la estación hidrométrica Paso Ancho (DOF, 2017).

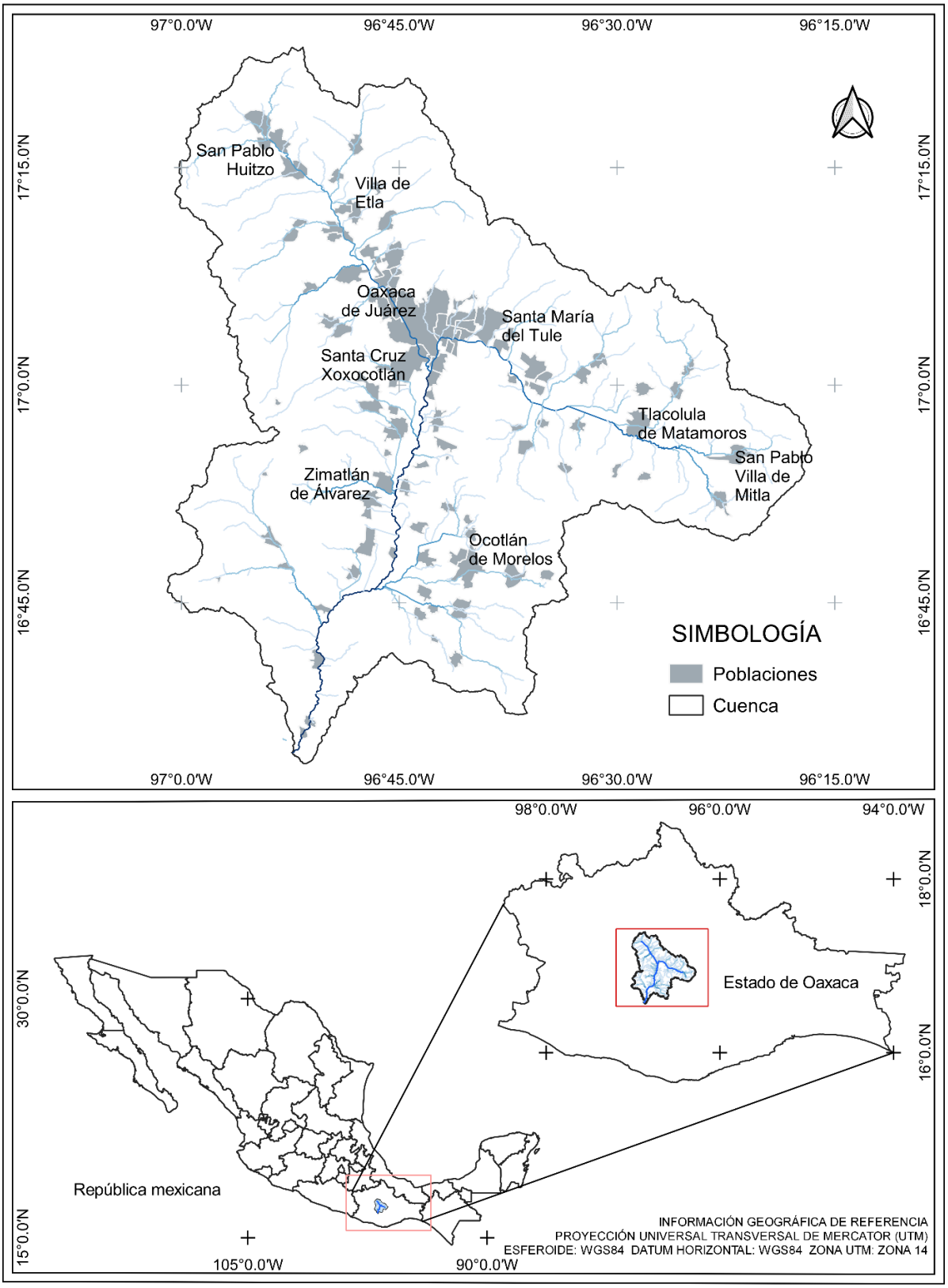

Figura 5.- Localización de la zona de estudio.

Fuente: elaboración propia.

La delimitación de la cuenca del río Atoyac-Salado (Figura 6), se realizó en ArcSWAT como una extensión del *software* ArcGIS a partir del modelo digital de elevación de alta resolución LiDAR de INEGI, resolución de 15 m, proyección Universal Transversa de Mercator (UTM) zona 14. La salida de la cuenca se ubicó en la estación hidrométrica Paso Ancho.

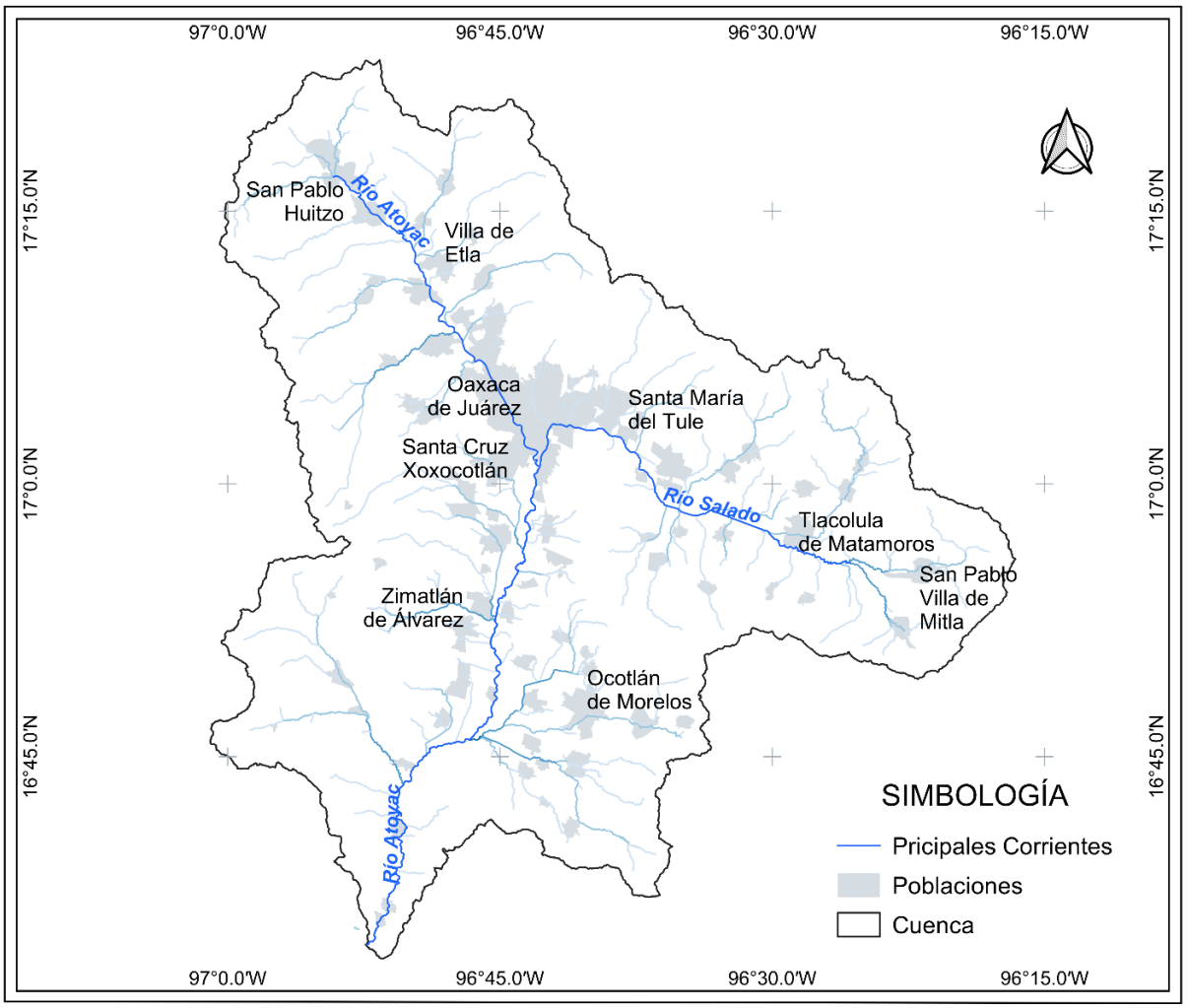

Figura 6.- Principales corrientes de la cuenca.

Fuente: elaboración propia.

Estudio de caso

La Serie VI es un conjunto de datos vectoriales de uso del suelo y vegetación, en escala 1:250 000 que presenta información de la ubicación, distribución y extensión de diferentes comunidades vegetales y usos agrícolas con sus respectivas variantes en tipos de vegetación e información ecológica relevante. Entre las 22 clases de uso de suelo y vegetación (Cuadro 1) presentes en la cuenca de estudio (Figura 7), pueden distinguirse dos tipos de cobertura vegetal con mayor extensión dentro de ésta. El de mayor superficie corresponde a agricultura de temporal anual, que representa el 21.41 % de la superficie total (INEGI, 2017). El segundo corresponde a vegetación secundaria arbustiva de bosque de encino, que ocupa un 17.78 % de la superficie total. Este último es un tipo de vegetación alterado, en el que surge una comunidad vegetal significativamente diferente a la original con estructura y composición florística heterogénea, que está formado por comunidades arbóreas, subarbóreas u ocasionalmente arbustivas integradas por múltiples especies del género *Quercus* (INEGI, 2017).

Se encontraron diferentes tipos de agricultura de acuerdo con el tipo de agroecosistema, que se dividen conforme al suministro de agua a los cultivos en temporal, riego y humedad, y por su duración en anuales, semipermanentes y permanentes. Teniendo en cuenta esta variabilidad de usos de suelo en el área de estudio, la cuenca del río Atoyac-Salado resultó adecuada para aplicar métodos de clasificación supervisada con inteligencia artificial.

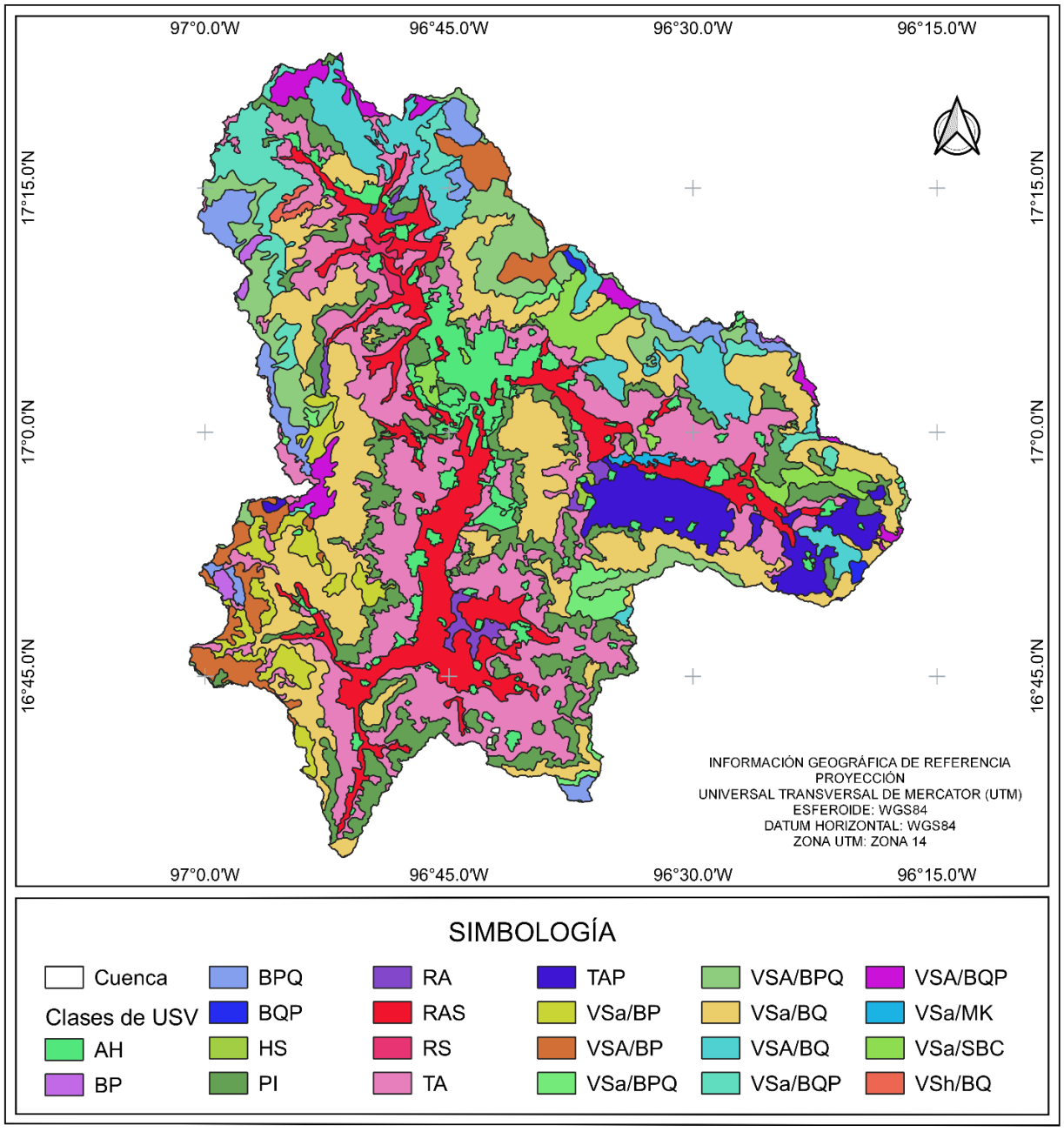

Figura 7.- Distribución de tipos de uso de suelo y vegetación en la zona de estudio.

Fuente: elaboración propia.

Cuadro 1.- Tipos de uso de suelo y vegetación presentes en la cuenca del río Atoyac-Salado.

Clase	Clave	Tipo de uso de suelo y vegetación	Superficie (ha)
0	AH	Urbano construido	21690.4
1	BP	Bosque de pino	1384.1
2	BPQ	Bosque de pino-encino	10157.7
3	BQP	Bosque de encino-pino	630.6
4	HS	Agricultura de humedad semipermanente	358.6
5	PI	Pastizal inducido	41935.9
6	RA	Agricultura de riego annual	2915.4
7	RAS	Agricultura de riego anual y semipermanente	38362
8	RS	Agricultura de riego semipermanente	1032.2
9	TA	Agricultura de temporal annual	79647.6
10	TAP	Agricultura de temporal anual y permanente	15552.3
11	VSa/BP	Vegetación secundaria arbustiva de bosque de pino	9473.3
12	VSa/BPQ	Vegetación secundaria arbustiva de bosque de pino-encino	4378.7
13	VSa/BQ	Vegetación secundaria arbustiva de bosque de encino	66145.4
14	VSa/BQP	Vegetación secundaria arbustiva de bosque de encino-pino	11682.1
15	VSa/MK	Vegetación secundaria arbustiva de bosque de mezquite	772.8
16	VSa/SBC	Vegetación secundaria arbustiva de selva baja caducifolia	8138.4
17	VSA/BP	Vegetación secundaria arbórea de bosque de pino	10569.2
18	VSA/BPQ	Vegetación secundaria arbórea de bosque de pino-encino	20728.7
19	VSA/BQ	Vegetación secundaria arbórea de bosque de encino	19473.6
20	VSA/BQP	Vegetación secundaria arbórea de bosque de encino-pino	6210.4
21	VSh/BQ	Vegetación secundaria herbácea de bosque de encino	726.5
		Total	372068.3

Fuente: Serie VI (INEGI, 2017). Asignación de clase y clave por tipo de USV para el modelo de clasificación.

Imágenes de satélite

Los sensores remotos, especialmente las imágenes satelitales, constituyen una importante fuente de información para mapear y caracterizar el uso del suelo y la estructura del paisaje a escala regional (Borges *et al*., 2002). Con el rápido aumento de las técnicas de imagen de teledetección en la última década, ahora se dispone de una cantidad considerable de imágenes de teledetección de alta resolución, lo que nos permite estudiar la superficie del suelo con mayor detalle (F. Hu *et al*., 2015). La misión Copernicus Sentinel-2 se basa en una constelación de dos satélites idénticos (Sentinel-2A y Sentinel-2B) en la misma órbita, desarrollados por la Agencia Espacial Europea (ESA, https://www.esa.int/Applications/Observing_the_Earth/Copernicus/Sentinel-2).

Equipados con un sensor óptico, el instrumento multiespectral tiene una resolución espacial que varía de 10 m a 60 m dependiendo de la banda espectral (Drusch *et al*., 2012) con 13 bandas en los rangos visible, infrarrojo cercano e infrarrojo de onda corta del espectro electromagnético, y con un tiempo de revisita de 5 días en el ecuador (Gascon *et al*., 2017).

Muestra

La primera fase de esta evaluación fue seleccionar los sitios de entrenamiento y verificación en campo. Para ello, se visitaron sitios accesibles y que fuera posible distinguir con facilidad en la imagen, para cada uno de los tipos de vegetación que el INEGI clasifica para la Cuenca. Es necesario señalar que esta fase de campo se realizó para la caracterización de las zonas de entrenamiento; además se llevó a cabo visitas de verificación en octubre y noviembre de 2020. Con ese conocimiento previo del área

de estudio, se seleccionaron los sitios de entrenamiento generales para las imágenes por analizar. Como éstos se fueron adaptando en específico sobre cada imagen, usando la capa vectorial de la Serie VI se seleccionaron los sitios de entrenamiento correspondientes en las imágenes de satélite, principalmente porque de ésta partió el que se asignaran las clases de USV.

Muestras de entrenamiento

Las CNN requieren de conjuntos de datos de entrenamiento grandes, con el fin de ser más robustas a la hora de clasificar automáticamente. También es necesario que los datos de las clases estén balanceados si es un problema de múltiples clases *(Suárez et al.*, 2017). La CNN puede extraer características más eficaces con la ayuda de información específica de la clase, que puede ser proporcionada por las muestras de entrenamiento (Chen *et al*., 2016).

Las muestras de entrenamiento utilizadas en el estudio se obtuvieron de imágenes de satélite que corresponden a los *Tiles* T14QQD y T14QQE y a las bandas RGB y NIR de escenas multitemporales con resolución espacial de 10 m, adquiridos el 13 de abril de 2021 y el 3 de mayo de 2021, respectivamente. Ambas imágenes fueron capturadas por Sentinel 2A, con un nivel de procesamiento 2-A. La fecha más conveniente para seleccionar la imagen está en estrecha relación con el tipo de fenómeno en estudio (Chuvieco, 1995). Se seleccionaron escenas con poca o ninguna nube y/o neblina, y se descargaron del portal Copernicus Open Access Hub (https://scihub.copernicus.eu/). Los productos de reflectancia de superficie de nivel 2A se generan con el procesador Sen2Cor, cuyo objetivo principal es corregir los productos Sentinel-2 de nivel 1C de los

efectos de la atmósfera (Gascon *et al*., 2017), siendo este el único tratamiento de preprocesamiento que se le dio a la imagen antes de obtener las muestras.

La unidad de muestreo se basó en recortes de 20 x 20 píxeles. El método de muestreo utilizado fue aleatorio estratificado (R. Congalton & Green, 2009)**,** utilizando conocimiento previo del área de estudio adquirido mediante recorridos de campo para dividir el área en grupos o estratos, y luego cada estrato se muestreó aleatoriamente**.**

Las unidades de análisis de este trabajo corresponden a las distintas coberturas y usos del suelo de la Serie VI que se encuentran distribuidas espacialmente en la cuenca del río Atoyac-Salado. Así, con la finalidad de obtener resultados representativos, y hacer una valoración del método de clasificación, se eligió una superficie de extensión y complejidad considerable, que planteó un desafío de clasificación, en particular para la diferenciación entre tipos de vegetación.

Para llevar a cabo la clasificación supervisada en el modelo CNN se empleó un conjunto de entrenamiento balanceado para evitar que se incline la clasificación a la clase con mayor número de muestras; es decir, para evitar realizar la clasificación con un grupo de datos desequilibrados (Gnip *et al*., 2021). También fue necesario considerar las clases con menor superficie, de la cuales no se puede extraer la misma cantidad de muestras de entrenamiento. Para realizar la extracción de muestras de entrenamiento se usó QGIS, donde se proyectaron las imágenes de satélite, el límite de la cuenca y el conjunto de datos vectoriales de la Serie VI; usando el mismo sistema de referencia de coordenadas. Se delimitó el área de estudio y para cada clase de USV se obtuvieron muestras de entrenamiento de 20x20x4 píxeles (alto, ancho y número de bandas) en

formato *.tiff*, considerando que al menos 80% del recorte pertenezca a una sola clase, con la que se clasifica la muestra para la generación del conjunto de entrenamiento y validación.

De cada clase de USV se generaron 6,000 muestras de entrenamiento, con excepción de las clases Agricultura de humedad de ciclo semipermanente con 2280, y Agricultura de riego de ciclo semipermanente con 4356, que por tener menor superficie se evitó el sobremuestreo. En total se generaron 126,636 muestras de entrenamiento. Las muestras generadas de cada clase constaron de cuatro bandas de la imagen original, tres del espectro visible (azul, verde, rojo) y una en el infrarrojo cercano.

Las CNN tienen una fase de extracción de características y de clasificación *(Suárez et al.*, 2017). En la extracción de características, la red utilizó una capa de convolución, una técnica matemática que asigna importancia (pesos) a ciertos elementos de la imagen y extrae los píxeles más relevantes de la imagen de entrada; lo cual divide la imagen en partes pequeñas para aprender la mayoría de los elementos esenciales y entidades cada vez más complejas en cada capa. En la capa convolucional se aplican filtros sistemáticamente a una entrada y crea mapas de características de salida y posteriormente se le atribuye una etiqueta.

Modelo de CNN

La programación del algoritmo del modelo de CNN se realizó en lenguaje *Python*, en entorno de desarrollo *Jupyter notebook*, usando bibliotecas de código abierto como *Tensorflow* y *Keras* para el aprendizaje automático. El modelo usado fue de tipo Secuencial, las capas de la red están ordenadas y apiladas linealmente (Xie *et al*., 2020),

el cual permite construir un modelo con estructuras sencillas e intuitivas, donde todas las neuronas de una capa conecten con todas las neuronas de la capa siguiente, basado en secuencias de tres tipos de capas, convolucionales, de agrupación y totalmente conectadas; la convolución y las capas totalmente conectadas suelen ir seguidas de una función de activación no lineal (Rousset *et al*., 2021).

Arquitectura de la red neuronal convolucional

La arquitectura del modelo consta de tres capas convolucionales y tres capas de agrupación siguiendo las sugerencias de Chen para equilibrar la complejidad y la solidez de la red (Chen *et al*., 2016); usando en cada capa 128 neuronas, un tamaño de kernel de 3 por 3, con mismo relleno (*Padding*), agregando ceros alrededor de las imágenes de entrada, las salidas de la capa tendrán las mismas dimensiones espaciales que sus entrad*as*; una función de activación denominada como Unidad Rectificada Lineal (*ReLu*) que devuelve 0 por cada valor negativo en la imagen de entrada mientras que devuelve el mismo valor por cada valor positivo, seguido de un filtro de submuestreo de agrupación promedio *(Average-Pooling)* que considera la media de los valores de activación de una ventana, y una capa de regulación (*Dropout)* con un 20% de posibilidades de establecer las entradas en cero. A continuación, se tiene una capa (*Flatten)*, que aplana las salidas multidimensionales de la última capa de convolución en un formato unidimensional, y dos capas densas, una de 512 neuronas ocultas con una activación ReLu y capa *Dropout* a un 20% y la última con 22 neuronas de salida que corresponden al número de clases a identificar, con función de activación *softmax* para predecir la probabilidad de cada clase.

Método de entrenamiento

El entrenamiento es la selección de hiperparámetros y el ajuste de parámetros para aumentar la precisión en la identificación de clases de las muestras de entrenamiento. Una forma de llevar esto a cabo es a partir del establecimiento de pesos conocidos con anterioridad (Ponce *et al*., 2014). Para el ajuste de los pesos de las conexiones, se dividió el conjunto de datos, en dos grupos: entrenamiento (80%) y prueba (20%). Los datos de entrenamiento a su vez se dividen en entrenamiento (80%) y evaluación (20%), los cuales se pueden introducir varias veces en la red llamando época a cada reiteración, el modelo se entrenó con 100 épocas. El conjunto de pruebas se usó para proporcionar una evaluación imparcial del modelo final. Durante el entrenamiento, el conjunto de pruebas no es visto por el modelo y se utiliza sólo después de que el modelo esté completamente entrenado tras el ajuste de los hiperparámetros.

Al entrenar la red se usó la función de pérdida de entropía cruzada categórica que, para estimar el error del estado actual del modelo, calculó la diferencia promedio entre las distribuciones de probabilidad real y pronosticada para todas las clases de modo que los pesos se puedan actualizar para reducir la pérdida en la siguiente evaluación conforme transcurre cada época. Al realizar las épocas y haber ajustado los pesos de modo que los datos de entrenamiento son predichos de forma correcta, se ingresan los datos de validación. El entrenamiento finaliza cuando se alcanza un error aceptablemente bajo para todos los patrones de aprendizaje (Bocco *et al*., 2007), al no encontrar mejora en los resultados y cuando la predicción de los datos es aceptable. Sin embargo, es importante saber si el modelo se está sobreajustando, y se hace inspeccionando el valor de la pérdida en cada época.

Compilando el modelo

La compilación del modelo toma tres parámetros: optimizador, pérdida y métricas. Se usó Adam como algoritmo optimizador que calcula el gradiente de la función de coste, para modificar ligeramente los pesos ,donde se puede ajustar la tasa de aprendizaje durante el entrenamiento y esta determina qué tan rápido se calculan los pesos óptimos para el modelo; ya que es computacionalmente eficiente, tiene pocos requisitos de memoria, es invariable al cambio de escala diagonal de los gradientes y es adecuado para problemas que son grandes en términos de datos y/o parámetros (Kingma & Ba, 2014). Se compiló con la función *categorical cross entropy* que se utiliza en tareas de clasificación de clases múltiples y la métrica de rendimiento de interés es la exactitud (*accuracy).* La exactitud de la clasificación relaciona la observación correctamente predicha y el total de observaciones.

Evaluación

La clasificación no está completa hasta que se haya evaluado la cantidad de predicciones positivas que fueron correctas (R. G. Congalton, 1991), ya que evalúa qué tan buenas son sus predicciones. Se usó la matriz de confusión, una herramienta del aprendizaje automático que permite visualizar el desempeño de un algoritmo de aprendizaje supervisado y consta de una tabla de doble entrada que confronta los valores reales con los resultados de la clasificación, haciendo fácil detectar dónde el sistema está confundiendo dos clases; siendo que los elementos en la diagonal corresponden a la predicción correcta y los elementos fuera de ésta corresponden a predicciones incorrectas, tanto en forma horizontal como vertical (Yeturu, 2020). La proporción de

puntos correctamente asignados (diagonal) expresa la confiabilidad (Mas *et al.*, 2003). Además de la exactitud, se analizaron más métricas de evaluación: la precisión, que se refiere a la dispersión del conjunto de valores obtenidos a partir de mediciones repetidas de una magnitud, la sensibilidad, que son valores que indican la capacidad del estimador para discriminar los casos positivos, y la puntuación F1 que resume la precisión y sensibilidad en una sola métrica.

V. RESULTADOS Y DISCUSIÓN

La precisión general indica la proporción de detecciones positivas del clasificador que fueron realmente correctas. La relación entre el número total de entradas identificadas correctamente y el número total de entradas da la precisión global de la clasificación. La exactitud de clasificación general (Figura 8) alcanzó un máximo de 89.44% sobre los datos de entrenamiento y un 84.57% en validación durante 100 épocas.

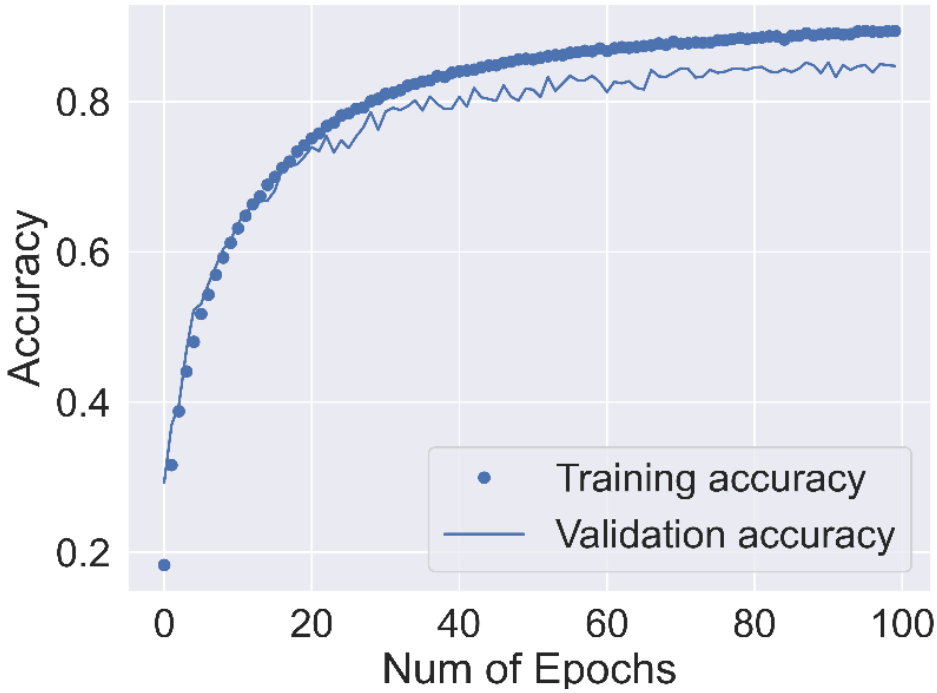

Figura 8.- Resultados de la precisión del modelo con el conjunto de datos de entrenamiento en la fase de entrenamiento y validación, durante cada época de entrenamiento.

Los resultados de la clasificación de imágenes se evaluaron utilizando la matriz de confusión la cual se puede representar gráficamente como en la Figura 9. En la matriz de confusión se muestra en color más claro las clases con mayor precisión en la clasificación y también muestra con que clase se confundió la clase ingresada.

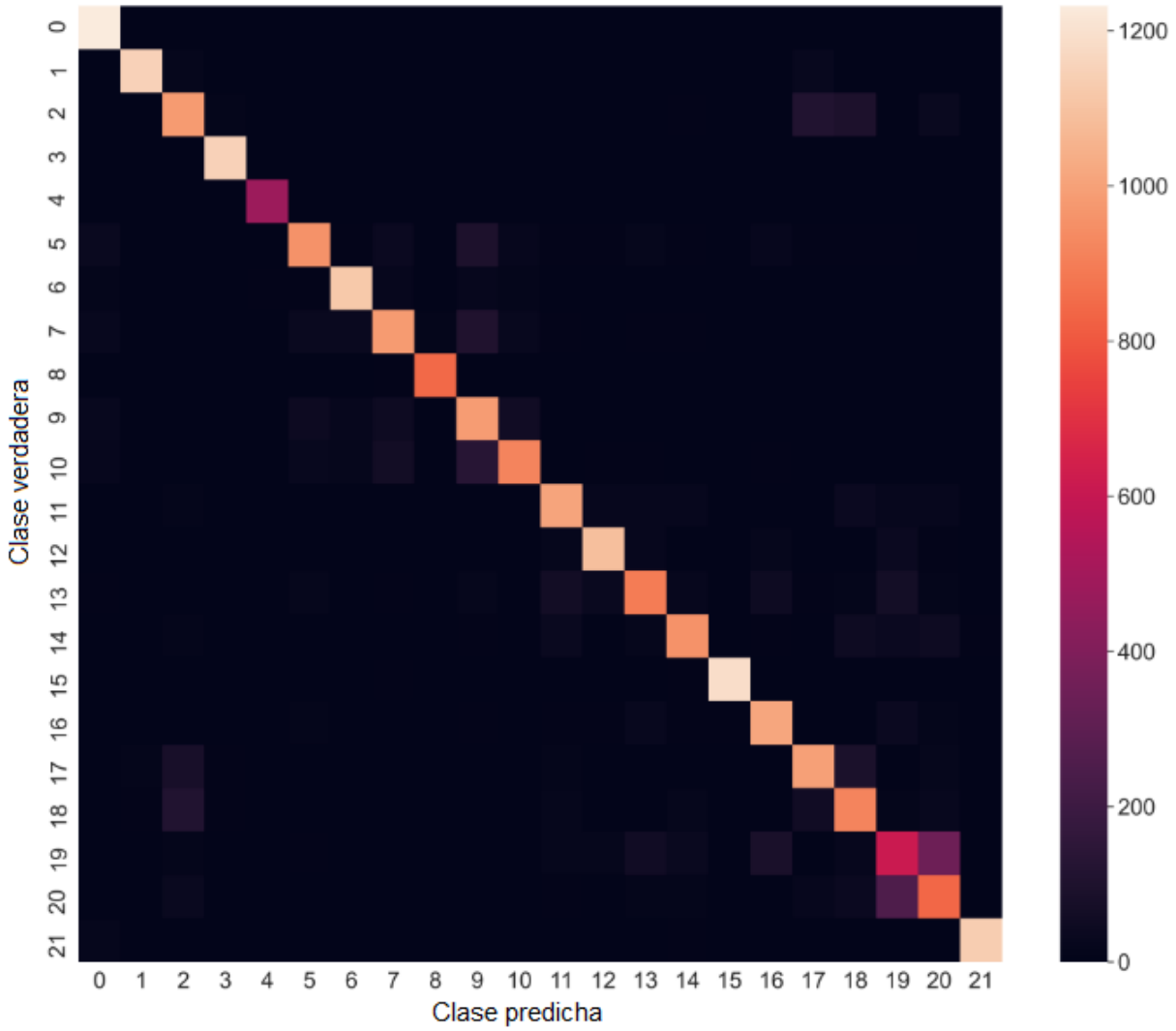

Figura 9.- Diagrama de Matriz de confusión. Los valores sobre la diagonal se refieren al número de entradas correctamente identificados por clase ingresada.

Nota: Los números de clase corresponden a los mencionados en el Cuadro 1.

Los elementos diagonales representan el número de puntos para los cuales la clase predicha es igual a la clase verdadera, mientras que cualquier valor fuera de la diagonal muestra de forma explícita cuándo una clase es confundida con otra. Por lo tanto, cuanto más altos son los valores diagonales de la matriz de confusión, mejor rendimiento, indicando muchas predicciones correctas. En este caso la red confunde más las clases etiquetadas como 19 y 20, que corresponden a vegetación secundaria arbórea de

bosque de encino y vegetación secundaria arbórea de bosque de encino-pino. Ambos tipos de cobertura vegetal comparten el mismo ecosistema de vegetación seminatural y predominio de árboles, donde cambia únicamente el componente florístico, eso podría explicar la confusión entre clases.

Métricas

De acuerdo con los resultados obtenidos para las métricas de evaluación (Cuadro 2), se aprecian un conjunto de puntuaciones medias (macro y ponderada) y exactitud con rendimiento general estimado de 85% para todas las métricas, por lo que se considera que el modelo de clasificación de uso de suelo y vegetación es robusto. Estos resultados indican que el modelo tiene una baja dispersión del conjunto de valores obtenidos, con el 85% de casos positivos que fueron identificados correctamente por el algoritmo.

Adicionalmente, el rendimiento del modelo se analizó con las variaciones de su sensibilidad y especificidad utilizando la curva de características operativas del receptor (ROC), un parámetro para evaluar la bondad de la prueba (Figura 10). Esta curva se considera una métrica eficaz para evaluar el rendimiento de los modelos predictivos, especialmente desarrollados para clasificación, los colores de las curvas representan cada uno de las clases identificadas y el área bajo la curva representa la exactitud de cada clase; la cual aumenta a medida que la curva se desplaza desde la diagonal hacia el vértice superior izquierdo. Un valor mayor indica generalmente que el modelo correspondiente puede lograr un mejor rendimiento (Liu *et al*., 2022).

Cuadro 2.- Métricas de evaluación del modelo por clase.

Clase	Precisión	Sensibilidad	Puntuación F1
0	0.89	1	0.94
1	0.97	0.96	0.97
2	0.78	0.79	0.79
3	0.97	0.99	0.98
4	0.98	0.98	0.98
5	0.86	0.77	0.82
6	0.93	0.93	0.93
7	0.83	0.79	0.81
8	0.98	0.99	0.99
9	0.72	0.82	0.77
10	0.88	0.77	0.82
11	0.84	0.85	0.85
12	0.91	0.9	0.91
13	0.82	0.74	0.78
14	0.86	0.81	0.83
15	0.99	0.98	0.99
16	0.84	0.89	0.86
17	0.81	0.82	0.82
18	0.72	0.78	0.75
19	0.56	0.51	0.53
20	0.62	0.69	0.65
21	0.99	0.98	0.98
Exactitud			0.85
Media macro	0.85	0.85	0.85
Media ponderada	0.85	0.85	0.85

Entrenamiento

Los conjuntos de datos de entrenamiento y validación se utilizaron para proporcionar una evaluación insesgada del modelo entrenado, ajustando los hiperparámetros en el entrenamiento para obtener el mejor rendimiento del modelo de red neuronal desarrollado.

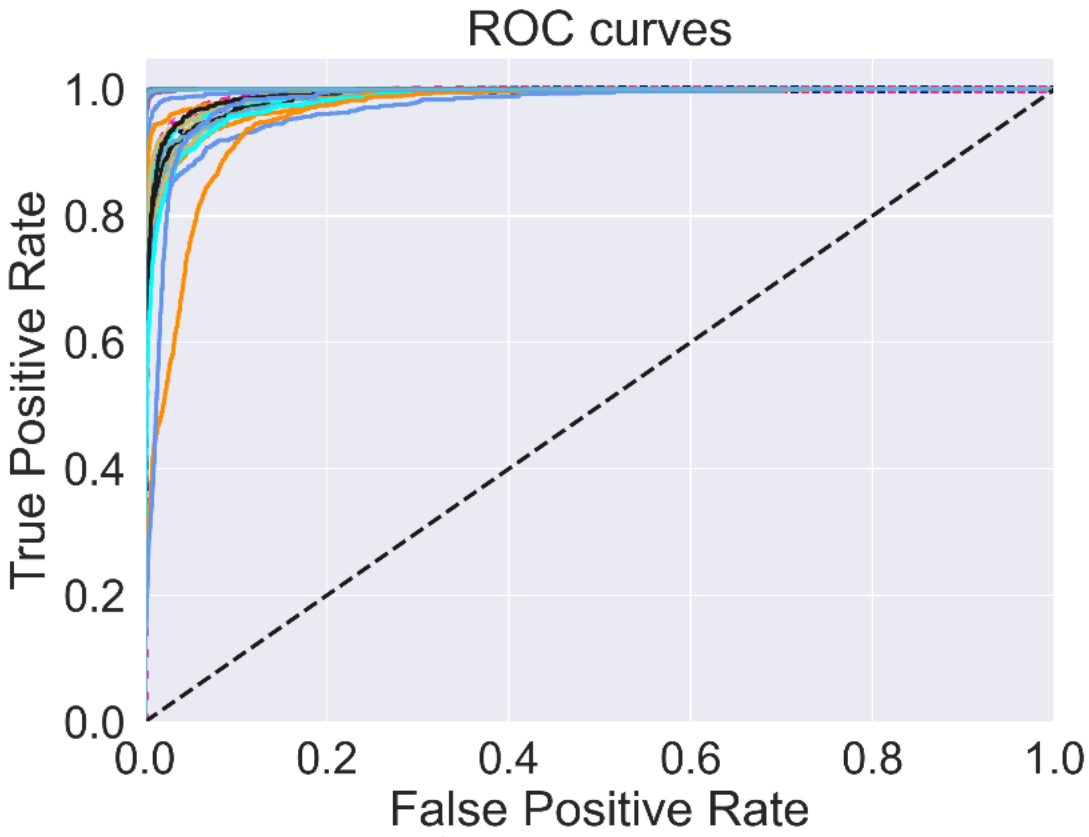

Figura 10.- Curva de características operativas del receptor por clase identificada.

El diseño del modelo de clasificación se prueba con los resultados del entrenamiento del modelo, y a su vez, se van ajustando los parámetros y la configuración de la arquitectura de la red, de tal manera que se elijan los hiperparámetros que arrojan mejores resultados de clasificación. Tanto en los conjuntos de entrenamiento, evaluación y prueba se utilizaron hiperparámetros como tamaño de *kernel*, la tasa de abandono, las capas ocultas, la profundidad de las capas, numero de épocas, cantidad de neuronas, las funciones de activación, entre otros.

Durante el entrenamiento la red experimentó un cambio positivo al usar una capa de regulación (*Dropout)* con un 20% de posibilidades de establecer las entradas en cero, lo que permite al modelo el ajuste de los datos minimizando el error producido por estos en

cada época. Mientras que, al no usarse la capa de regulación, llega a un punto en que el error aumenta (Figura 11). En el primer gráfico se presenta la exactitud (*accuracy*) obtenida en cada época, tanto para los datos de entrenamiento (*training*) como para los de validación (*validación*). En el segundo gráfico muestra la evolución en cada época de la función de perdida para los dos conjuntos de datos.

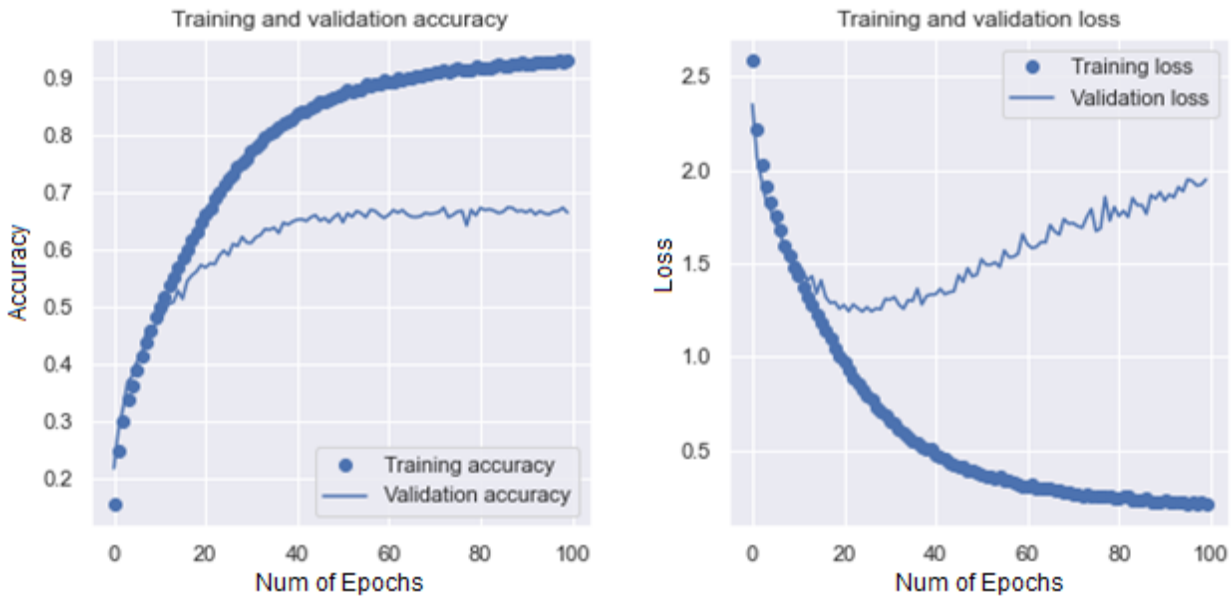

Figura 11.-. Monitoreo de parámetros en entrenamiento.

El comportamiento general de las gráficas mostradas en la Figura 11 es el característico de un modelo cuando presenta sobreentrenamiento (*Overfitting*). Por un lado, la exactitud de los datos de entrenamiento aumenta hasta alcanzar un valor superior a 90%, mientras que la exactitud de los datos de validación se detiene en alrededor del 67% y se mantiene constante. Por otro lado, la pérdida de entrenamiento va disminuyendo poco a poco; es decir, el modelo se va ajustando a los datos de entrenamiento, mientras la pérdida de validación aumenta.

Discusión

Los resultados obtenidos en rendimiento fueron altos, presentan respuestas acertadas y muestran avances para el procedimiento realizado con CNN en la clasificación de uso de suelo y vegetación con 22 clases, a pesar de que la escala 1:250 000 usada para crear las series tiene el problema de generar grandes polígonos de clases de USV no representativas de la escala local (Paz *et al*., 2019).

Estos resultados son mejores comparados con trabajos previos de clasificación con CNN donde se logró exactitud del 83.27% en entrenamiento y 91.02% en validación para identificar 4 clases (Suárez *et al*., 2017), y con 12 clases donde la media de las precisiones de la clasificación fueron del 90.18% para cobertura vegetal y del 87.92% para uso de suelo, utilizando un modelo de aprendizaje profundo, conjunto de dos redes de aprendizaje automático, un perceptrón multicapa y CNN (Zhang *et al*., 2019).

El modelo propuesto, mediante la implementación de una red neuronal convolucional profunda, mostró resultados satisfactorios en un conjunto de datos altamente desafiante utilizando puramente el aprendizaje supervisado. Habiendo entrenado en el conjunto de datos, la red experimenta un sobreajuste sustancial cuando se omite *Dropout*, mientras que no se informa sobreajuste cuando se agrega esta función (Srivastava *et al*., 2014). Esta función le permite al modelo reducir el sobreentrenamiento y consiste en desactivar neuronas de manera aleatoria. Cabe destacar que al eliminar cualquiera de las capas intermedias el rendimiento de la red se degrada (Krizhevsky *et al*., 2012) y supone una pérdida de alrededor del 5% si se elimina una sola capa convolucional. La configuración de profundidad de la red CNN es esencial en la precisión de la clasificación ya que la

calidad de las características aprendidas está influenciada por los niveles de representaciones y abstracciones (Zhang *et al*., 2019).

VI. CONCLUSIONES

En este trabajo se logró la clasificación de uso del suelo y vegetación mediante el uso de un algoritmo de aprendizaje profundo, en específico de red neuronal convolucional, para extraer la información profunda de la red multicapa en el proceso de clasificación supervisada. El proceso de modelización con redes neuronales convolucionales resultó eficaz para estimar el uso del suelo y vegetación en la cuenca del río Atoyac-Salado con imágenes del satélite Sentinel 2, a partir de datos de entrenamiento de acuerdo con la Serie VI del INEGI. Se comprobó con suficiente precisión que el aprendizaje profundo con redes neuronales convolucionales puede identificar patrones en los datos de la reflectancia captada por las imágenes del satélite Sentinel 2 para la clasificación el uso de suelo y vegetación en áreas con una dificultad intrínseca.

El modelo implementado detectó correctamente las clases más separadas espectralmente y que poseen características diferenciales, aunque algunas clases como VSA/BQ y VSA/BQP tuvieron índices de reconocimiento muy bajos. No se afectaron las clases con menor número de datos de entrenamiento, ya que éstas no tuvieron precisiones bajas. Los resultados experimentales mostraron mejoría al aumentar el tamaño de la red en número de capas de convolución hasta llegar a tres, donde lo permite el tamaño de la imagen y tiempo de entrenamiento, logrando buenos resultados, pero aún quedan órdenes de magnitud por superar para mejorar la precisión en la clasificación.

El aprendizaje profundo supone la mejor alternativa para la clasificación de USV. Sin embargo, el tiempo tanto para el entrenamiento como para la predicción aumentan con

la precisión, aun así, este aumento de tiempo no perjudica a la solución del problema real, ya que no se necesita una velocidad de reconocimiento instantánea. Por lo que la red neuronal convolucional es una opción idónea para la clasificación de uso del suelo y vegetación. Además, durante el entrenamiento, al aumentar la cantidad de neuronas hasta 128 y observar el comportamiento de los parámetros ayudó a mejorar aún más la precisión de detección, ya que esto favorecía la capacidad de abstracción de la red neuronal, aunque ésta no llegó a ser más profunda porque la dimensión de entrada de las imágenes fue muy pequeña para tantas capas, dejando solo tres capas.

Para la identificación de USV se probó el algoritmo de aprendizaje profundo en diferentes tamaños del conjunto de datos, desde mil muestras por clase hasta llegar a un límite de seis mil, en donde solo dos clases no alcanzaran el mismo tamaño de muestras para el conjunto de datos de entrenamiento. Se apreció que tuvo gran influencia en el rendimiento: la variedad de muestras de entrenamiento distribuidas por toda la zona de estudio, la calidad de las imágenes de pertenecer en al menos el 80 % del parche de entrenamiento a una sola clase, con la que se identificó, así como, la cantidad y formas de las clases a predecir. Aunque se tenga un conjunto de datos muy grande y variado, al querer identificar en la imagen características parecidas entre ellas, no se consigue que la red funcione de una manera consistente, por lo que se considera que la red deja de aprender, creando un sobreajuste y bajando el rendimiento en la clasificación debido a que los patrones en los datos son parecidos entre ellos, por lo que es necesaria la función de abandono. De esta manera, al adaptar el conjunto de muestras de entrenamiento, según su forma de la clase de USV a identificar, se consigue mejorar notablemente la identificación de las clases en el resultado. En suma, al aumentar la

cantidad de datos de entrada a la red, durante el entrenamiento y la predicción, los resultados en la clasificación mejoran de manera significativa. Esto debido a que tiene más variedad de datos de donde puede identificar los patrones, pero si llega a haber imágenes que puedan ser similares en los patrones de reconocimiento, la red deja de aprender.

VII. RECOMENDACIONES

Actualmente, la oferta de herramientas que soportan el uso de métodos de aprendizaje profundo es muy alta debido al auge de estas técnicas; sin embargo, entender todos los parámetros que intervienen en el entrenamiento de una red neuronal convolucional se dificulta. La selección de hiperparámetros y la asignación de valores óptimos a los parámetros que convergen al diseñar una red no es un proceso trivial, de ello dependen tanto el correcto funcionamiento del entrenamiento, el costo de entrenamiento, así como la obtención de resultados satisfactorios. Por lo tanto, para cualquier proyecto basado en aprendizaje profundo, cualquier cambio pequeño en los valores afectara el rendimiento general, siendo fundamental la comprensión de la repercusión de estos parámetros sobre la evolución de una red.

Las imágenes de satélite ofrecen tantos retos como oportunidades para explotar las ventajas potenciales en el campo de clasificación de cobertura y uso del suelo con aprendizaje automático. Con el aumento de estos retos en las tareas de visión artificial y aprendizaje automático, los modelos de las redes neuronales profundas son cada vez más complejos y potentes, los cuales requieren más datos para su entrenamiento con el fin de evitar el sobreajuste y, al mismo tiempo, los datos de entrenamiento también plantean nuevos retos de como entrenar las redes en un tiempo factible, por lo que requieren recursos de *hardware*.

Los resultados obtenidos en este trabajo son de enorme ayuda para abordar una tarea de clasificación mediante *Deep learning*; sin embargo, no es posible abarcar las vertientes de un campo tan vasto en un trabajo de estas características. A continuación,

se enuncian algunas propuestas que podrían abordarse en un trabajo futuro:

- En este proyecto se utilizó una red convolucional básica con el fin de entender su arquitectura y su funcionamiento. Sin embargo, en la actualidad existen muchas otras redes con arquitecturas más complejas que se podrían estudiar/utilizar, es deseable en un trabajo futuro utilizar una red de mayor profundidad.
- Respecto a las muestras de entrenamiento, en este trabajo se utilizaron recortes de 20x20 pixeles lo que se puede considerar como un recorte pequeño debido a que se pueden hacer menos operaciones de convolución, por lo que se recomienda experimentar con recortes mayores que permitan hacer más operaciones para la extracción de patrones en el entrenamiento de la red.
- Además de usar solo las reflectancias de las bandas espectrales, también se podría experimentar añadiendo capas de índices de vegetación.

Conocimientos adquiridos

El desarrollo de este trabajo se basó en los contenidos temáticos específicos de la Maestría en Hidrociencias. Resultó un desafío importante debido a que se requirió aplicar y expandir los conceptos y uso de herramientas tecnológicas aplicados al manejo integral de cuencas y el efecto del cambio climático, estudiados en asignaturas como visión artificial, impacto del cambio climático en los recursos hídricos, deterioro-preservación ambiental, sistemas de información geográfica y conceptos básicos de matemáticas, estadística e hidráulica, a un problema particular de gran interés y aplicación para un amplio sector de investigadores y usuarios.

VIII. LITERATURA CITADA

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y, and Zheng, X. (2016). TensorFlow : A System for Large-Scale Machine Learning. *USENIX Association*. https://dl.acm.org/doi/10.5555/3026877.3026899.

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. In Journal of Big Data (Vol. 8, Issue 1). Springer International Publishing. https://doi.org/10.1186/s40537-021-00444-8.

Alzubi, J., Nayyar, A., & Kumar, A. (2018). Machine Learning from Theory to Algorithms: An Overview. *Journal of Physics: Conference Series*, *1142*(1). https://doi.org/10.1088/1742-6596/1142/1/012012.

Anaya, A., Mera, L., & Zequera, M. (2021). An overview of deep learning in medical imaging. *Informatics in Medicine Unlocked*, *26*. https://doi.org/10.1016/j.imu.2021.100723.

Bhosle, K., & Musande, V. (2019). Evaluation of Deep Learning CNN Model for Land Use Land Cover Classification and Crop Identification Using Hyperspectral Remote Sensing Images. *Journal of the Indian Society of Remote Sensing*, *47*(11), 1949–1958. https://doi.org/10.1007/s12524-019-01041-2.

Bocco, M., Ovando, G., Sayago, S., & Willington, E. (2007). Neural Network Model for Land Cover Classification from Satellite Images. *Agricultura Técnica*, *67*(4), 414–421. https://doi.org/10.4067/S0365-28072007000400009.

Borges, S., Sarandón, S., & Alperín, M. (2002). Caracterización Espacial de los Tipos de Cobertura de Suelo usando Técnicas Geoestadísticas a partir de Información Satelital. *Revista de La Facultad de Agronomía, La Plata*, *105*(1), 40–51.

Borràs, J., Delegido, J., Pezzola, A., Pereira, M., Morassi, G. & Camps-Valls, G. (2017). Clasificación de usos del suelo a partir de imágenes Sentinel-2. *Revista de Teledetección*, *2017*(48), 55. https://doi.org/10.4995/raet.2017.7133.

Chen, Y., Jiang, H., Li, C., Jia, X., & Ghamisi, P. (2016). Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks. *IEEE Transactions on Geoscience and Remote Sensing*, *54*(10), 6232–6251. https://doi.org/10.1109/TGRS.2016.2584107.

Chollet, F. (2015). *Deep Learning with Python*. Manning Publications Co. https://unica.it/static/resources/cms/documents/DeepLearningWithPython.pdf.

Chuvieco, E. (1995). *Fundamentos de teledetección espacial* (S. A. EDICIONES RIALP

(ed.)).

Congalton, R. G. (1991). A review of assessing the accuracy of classifications of remotely sensed data. *Remote Sensing of Environment*, *37*(1), 35–46. https://doi.org/10.1016/0034-4257(91)90048-B.

Congalton, R., & Green, K. (2009). Sample Design Considerations. In C. Press (Ed.), *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices, Second Edition.* (pp. 63–83). https://doi.org/10.1201/9781420055139.ch5.

Deepan, P., & Sudha, L. R. (2020). Chapter 8. Object Classification of Remote Sensing Image Using Deep Convolutional Neural Network. In *The Cognitive Approach in Cloud Computing and Internet of Things Technologies for Surveillance Tracking Systems*. INC. https://doi.org/10.1016/B978-0-12-816385-6.00008-8.

DOF. (2017). Acuerdo por el que se dan a conocer los resultados del estudio técnico de las aguas nacionales superficiales en las cuencas hidrológicas Río Papagayo 1, Río Petaquillas, Río Omitlán, Río Papagayo 2, Río Papagayo 3, Río Papagayo 4, Río Nexpa 1, Río Nexpa 2. *Diario Oficial de La Federación*. https://dof.gob.mx/nota_detalle.php?codigo=5496053&fecha=04/09/2017.

Drusch, M., Del Bello, U., Carlier, S., Colin, O., Fernandez, V., Gascon, F., Hoersch, B., Isola, C., Laberinti, P., Martimort, P., Meygret, A., Spoto, F., Sy, O., Marchese, F., & Bargellini, P. (2012). Sentinel-2: ESA's Optical High-Resolution Mission for GMES Operational Services. *Remote Sensing of Environment*, *120*, 25–36. https://doi.org/10.1016/j.rse.2011.11.026.

Fidel, L., & Cort, J. V. (2019). Red neuronal convolucional con extracción de características multi-columna para clasificación de imágenes Convolutional Neural Network with Extraction. *Research in Computing Science*, *148*(7), 391–404.

Galicia Sarmiento, L. (2016). Dinámica de cambio del uso de suelo y vegetación en México: patrones de cambio, causas directas e indirectas y prioridades futuras. In *Geografía de México: una reflexión espacial contemporánea* (Issue November 2016, pp. 235–249). https://www.researchgate.net/publication/310234086_Dinamica_de_cambio_del_uso_de_suelo_y_vegetacion_en_Mexico_patrones_de_cambio_causas_directas_e_indirectas_y_prioridades_futuras.

García Sánchez, E., Alcalá Nalvaiz, J. T., & Orellana Lozano, L. (2019). *Introducción a las redes neuronales de convolución. Aplicación a la visión por ordenador* (Universidad de Zaragoza (ed.)). https://zaguan.unizar.es/record/87398/files/TAZ-TFG-2019-3085.pdf.

García, T. J., & Mas, J. F. (2008). Comparación de metodologías para el mapeo de la cobertura y uso del suelo en el sureste de méxico. *Investigaciones Geográficas*, *67*(8701), 7–9. https://www.redalyc.org/articulo.oa?id=56911125002.

Gascon, F., Bouzinac, C., Thépaut, O., Jung, M., Francesconi, B., Louis, J., Lonjou, V.,

Lafrance, B., Massera, S., Gaudel-Vacaresse, A., Languille, F., Alhammoud, B., Viallefont, F., Pflug, B., Bieniarz, J., Clerc, S., Pessiot, L., Trémas, T., Cadau, E., Fernandez, V. (2017). Copernicus Sentinel-2A Calibration and Products Validation Status. *Remote Sensing*, *9*(6), 584. https://doi.org/10.3390/rs9060584.

Gnip, P., Vokorokos, L., & Drotár, P. (2021). Selective oversampling approach for strongly imbalanced data. *PeerJ Computer Science*, *7*, e604. https://doi.org/10.7717/peerj-cs.604.

González Duque, R. (2000). Python para todos. *Web Book*, *6*.

Herlau, T., Schmidt, M. N., & Mørup, M. (2022). Bayesian dropout. *Procedia Computer Science*, *201*(2019), 771–776. https://doi.org/10.1016/j.procs.2022.03.105.

Hu, F., Xia, G. S., Hu, J., & Zhang, L. (2015). Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. *Remote Sensing*, *7*(11), 14680–14707. https://doi.org/10.3390/rs71114680.

Hu, Y., Zhang, Q., Zhang, Y., & Yan, H. (2018). A deep convolution neural network method for land cover mapping: A case study of Qinhuangdao, China. *Remote Sensing*, *10*(12), 1–17. https://doi.org/10.3390/rs10122053.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science and Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55.

INEGI. (2005). *Guía para la Interpretación de la Cartografía Uso del Suelo y Vegetación* (Primera ed). Instituto Nacional de Estadística y Geografía.

INEGI. (2009). Sistema de información de la cobertura de la tierra SICT. In *Instituto Nacional de Estadística y Geografía*. http://www.jstor.org/stable/2097281.

INEGI. (2017). Guía para la interpretación de cartografía: uso del suelo y vegetación. *Intituto Nacional de Estadística y Geografía*, *México*, 204.

Jarrahi, M. H., Askay, D., Eshraghi, A., & Smith, P. (2022). Artificial intelligence and knowledge management: A partnership between human and AI. *Business Horizons*. https://doi.org/10.1016/j.bushor.2022.03.002.

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations, ICLR*, 1–15. http://arxiv.org/abs/1412.6980.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, *25*. http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf.

Liu, R., Yang, X., Xu, C., Wei, L., & Zeng, X. (2022). Comparative Study of Convolutional Neural Network and Conventional Machine Learning Methods for Landslide Susceptibility Mapping. *Remote Sensing*, *14*(2), 321. https://doi.org/10.3390/rs14020321.

López Pacheco, M. A. (2021). *Redes neuronales convolucionales para el modelado de sistemas no lineales con aplicación al monitoreo de daño estructural*.

Luxton, D. D. (2016). An Introduction to Artificial Intelligence in Behavioral and Mental Health Care. In *Artificial Intelligence in Behavioral and Mental Health Care*. Elsevier Inc. https://doi.org/10.1016/B978-0-12-420248-1.00001-5.

Macedo, A., Pajares, G., & Santos, M. (2010). Clasificación no supervisada con imágenes a color de cobertura terrestre. *Agrociencia*, *44*(6), 711–722. https://www.redalyc.org/articulo.oa?id=30215554009.

Macpherson, T., Churchland, A., Sejnowski, T., DiCarlo, J., Kamitani, Y., Takahashi, H., & Hikida, T. (2021). Natural and Artificial Intelligence: A brief introduction to the interplay between AI and neuroscience research. *Neural Networks*, *144*, 603–613. https://doi.org/10.1016/j.neunet.2021.09.018.

Mas, J.-F., Reyes Díaz Gallegos, J., & Pérez Vega, A. (2003). Evaluación de la confiabilidad temática de mapas o de imágenes clasificadas: una revisión. *Investigaciones Geográficas*, *51*, 53–72. https://doi.org/10.14350/rig.30414.

Mas, J.-F., Velázquez, A., & Couturier, S. (2009). La evaluación de los cambios de cobertura / uso del suelo en la República Mexicana. *Investigación Ambiental*, *1*(1), 23–39.

McCaffrey, P. (2020). Packages , interactive computing , and analytical documents. In *An Introduction to Healthcare Informatics* (pp. 159–174). Elsevier Inc. https://doi.org/10.1016/B978-0-12-814915-7.00012-0.

McClarren, R. G. (2018). NumPy and Matplotlib. In *Ingeniería nuclear computacional y ciencia radiológica utilizando Python* (pp. 53–74). https://doi.org/10.1016/B978-0-12-812253-2.00005-4.

Meyer, W., & Turner II, B. (1992). Human Population Growth and Global Land-Use/Cover Change. *Annual Review of Ecology and Systematics*, *23*, 39–61.

Mitta, M., Shah, R. R., & Roy, S. (2021). *Recognition of trivial humanoid group event using clustering and higher order local auto-correlation techniques* (Issue January). https://doi.org/10.1016/B978-0-323-85769-7.00001-X.

Mostafa, S., & Wu, F. (2021). Diagnosis of Autism Spectrum Disorder with Convolutional Autoencoder and Structural MRI Images. In *Neural Engineering Techniques for Autism Spectrum Disorder* (Issue January, pp. 23–38). https://doi.org/10.1016/B978-0-12-822822-7.00003-X.

Mursina, L. (2010). Artificial neural networks. *Medizintechnik*, *130*(3), 96–103. https://doi.org/10.1533/9780857099440.275.

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, *2*(1), 1–21. https://doi.org/10.1186/s40537-014-0007-

7.

Palacio, J. L., & Luna, L. (1994). Clasificación espectral automática vs clasificación visual: un ejemplo al sur de la Ciudad de México. *Investigaciones Geográficas*, *29*, 25–40.

Pang, B., Nijkamp, E., & Wu, Y. N. (2020). Deep Learning With TensorFlow: A Review. *Journal of Educational and Behavioral Statistics*, *45*(2), 227–248. https://doi.org/10.3102/1076998619872761.

Pascual Ramírez, F., Paz Pellat, F., Martínez Menes, M., Palacios Vélez, E., Mejía Sáenz, E., & Rubio Granados, E. (2010). Clasificador genérico de objetos en imágenes AVHRR. *Terra Latinoamericana*, 1–13. https://www.researchgate.net/publication/296707260_Clasificador_generico_de_objetos_en_imagenes_AVHRR.

Paz, F., Romero, V. M., Argumedo-Espinoza, J. A., Bolaños, M., de Jong, B., de la Cruz, J. C., & Velázquez, A. (2019). Dinámica Del Uso Del Suelo Y Vegetación. In *Estado del Ciclo del Carbono en México, Agenda Azul y Verde* (Vol. 23, Issue June, pp. 529–572).

Peris, Á., & Casacuberta, F. (2018). *NMT-Keras: a Very Flexible Toolkit with a Focus on Interactive NMT and Online Learning Álvaro Peris, Francisco Casacuberta*. 1–12.

Ponce, J. C., Torres, A., Quezada, F. S., Silva, A., Martínez, E. U., Casali, F. A., Scheihing, E., Túpac, Y. J., Torres, M. D., Ornelas, F. J., Hernández, J. A., Zavala, C., Vakhni, N., & Pedreño, O. (2014). Inteligencia Artificial. In *Iniciativa Latinoamericana de Libros de Texto Abiertos (LATIn) (Vol. 1).*

Porcelli, A. M. (2020). Inteligencia Artificial y la Robótica: sus dilemas sociales, éticos y jurídicos. *Derecho Global. Estudios Sobre Derecho y Justicia*, *6*(16), 49–105. https://doi.org/10.32870/dgedj.v6i16.286.

Rousset, G., Despinoy, M., Schindler, K., & Mangeas, M. (2021). Assessment of deep learning techniques for land use land cover classification in southern new caledonia. *Remote Sensing*, *13*(12), 1–22. https://doi.org/10.3390/rs13122257.

Sanjurjo, A. (2020). *Aplicación del Aprendizaje Profundo a la clasificación de estados de sueño*. http://hdl.handle.net/2183/25163.

Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, *2*(3), 160. https://doi.org/10.1007/s42979-021-00592-x.

Shalev, S., & Ben, S. (2014). Understanding machine learning: From theory to algorithms. In *Understanding Machine Learning: From Theory to Algorithms*. https://doi.org/10.1017/CBO9781107298019.

Soria, E., & Blanco, A. (2001). Redes neuronales artificiales. *Autores Científico-Técnicos y Académicos*, 25–33. https://www.acta.es/medios/articulos/informatica_y_computacion/019023.pdf.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, *15*, 1929–1958.

Suárez, A. S., Jiménez, A. F., Castro, M., & Cruz, A. A. (2017). Clasificación y mapeo automático de coberturas del suelo en imágenes satelitales utilizando Redes Neuronales Convolucionales. *Orinoquia*, *21*(1 Sup), 64–75. https://doi.org/10.22579/20112629.432.

Szegedy, C., Reed, S., Sermanet, P., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *In Proceedings of CVPR*, 1–12.

Theodoridis, S. (2015). Introduction. In *Machine Learning* (pp. 1–8). Elsevier. https://doi.org/10.1016/B978-0-12-801522-3.00001-X.

Treitz, P., & Rogan, J. (2004). Remote sensing for mapping and monitoring land-cover and land-use change—an introduction. *Progress in Planning*, *61*(4), 269–279. https://doi.org/10.1016/S0305-9006(03)00064-3.

Trucíos, R., Rivera, M., Delgado, G., Estrada, J., & Cerano, J. (2013). Análisis sobre cambio de uso de suelo en dos escalas de trabajo. *Terra Latinoamericana*, *31*, 339–346.

Vandeginste, B. (1998). Artificial neural networks. *Data Handling in Science and Technology*, *20*(PART 2), 649–699. https://doi.org/10.1016/S0922-3487(98)80054-3.

Verona, P., Cristina, I., & García, A. (2016). Una revisión sobre aprendizaje no supervisado de métricas de distancia. *Revista Cubana de Ciencias Informáticas*, *10*(4), 43–67.

Xie, G., Shangguan, A., Fei, R., Ji, W., Ma, W., & Hei, X. (2020). Motion trajectory prediction based on a CNN-LSTM sequential model. *Science China Information Sciences*, *63*, 1–21. https://doi.org/10.1007/s11432-019-2761-y

Yao, G., Lei, T., & Zhong, J. (2019). A review of Convolutional-Neural-Network-based action recognition. *Pattern Recognition Letters*, *118*, 14–22. https://doi.org/10.1016/j.patrec.2018.05.018.

Yeturu, K. (2020). Machine learning algorithms, applications, and practices in data science. In *Handbook of Statistics* (1st ed., Vol. 43, pp. 81–206). Elsevier B.V. https://doi.org/10.1016/bs.host.2020.01.002.

Zhang, C., Sargent, I., Pan, X., Li, H., Gardiner, A., Hare, J., & Atkinson, P. M. (2019). Joint Deep Learning for land cover and land use classification. *Remote Sensing of Environment*, *221*, 173–187. https://doi.org/10.1016/j.rse.2018.11.014.

ANEXO

Código Fuente

```
Importar librerías para crear la red
import tensorflow as tf
import numpy as np
import pandas as pd
import seaborn as sn
import os
import re
import matplotlib.pyplot as plt
import tifffile as tiff
%matplotlib inline
from scipy import interp
from itertools import cycle
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_curve, auc
from sklearn.metrics import roc_auc_score
import keras
from keras.utils import to_categorical
from keras.optimizers import Adam
from keras.models import Sequential,Input,Model
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, AveragePooling2D, MaxPooling2D
from keras.layers.normalization import BatchNormalization
from keras.layers.advanced_activations import LeakyReLU
from keras.metrics import categorical_crossentropy
from keras.constraints import max_norm
```

```
Importar imágenes
dirname = os.path.join(os.getcwd(), 'usv')
imgpath = dirname + os.sep
images = []
directories = []
dircount = []
prevRoot=''
cant=0
print("leyendo imagenes de ",imgpath)
      for root, dirnames, filenames in os.walk(imgpath):
            for filename in filenames:
                  if re.search("\.(tiff)$", filename):
                        cant=cant+1
                        filepath = os.path.join(root, filename)
                        image = tiff.imread(filepath)
                        images.append(image)
                        b = "Leyendo..." + str(cant)
                        print (b, end="\r")
                        if prevRoot !=root:
                              print(root, cant)
                              prevRoot=root
                              directories.append(root)
                              dircount.append(cant)
                              cant=0
dircount.append(cant)
dircount = dircount[1:]
dircount[0] = dircount[0] + 1
print('Directorios leidos:',len(directories))
print("Imagenes en cada directorio", dircount)
print('Suma total de imagenes:',sum(dircount))
Crear etiquetas
labels=[]
```

```
indice=0
for cantidad in dircount:
for i in range(cantidad):
labels.append(indice)
indice=indice+1
print("Cantidad de etiquetas creadas: ",len(labels)
coberturas=[]
indice=0
for directorio in directories:
name = directorio.split(os.sep)
print(indice , name[len(name)-1])
coberturas.append(name[len(name)-1])
indice=indice+1
y = np.array(labels)
X = np.array(images, dtype=np.uint8) #list to numpy
classes = np.unique(y)
nClasses = len(classes)
print('Total number of outputs : ', nClasses)
print('Output classes : ', classes)
Crear Sets de Entrenamiento y Test
train_X,test_X,train_Y,test_Y = train_test_split(X,y,
      random_state=13, test_size=0.2)
print('Training data shape : ', train_X.shape, train_Y.shape)
print('Testing data shape : ', test_X.shape, test_Y.shape)
plt.figure(figsize=[6,6])
plt.subplot(121)
plt.imshow(train_X[0,:,:], cmap='gray')
plt.title("First im train: {}".format(train_Y[0]))
plt.subplot(122)
plt.imshow(test_X[0,:,:], cmap='gray')
plt.title("First im test: {}".format(test_Y[0]))
Preprocesamiento de las imágenes
```

```
train_X = train_X.astype('float32')
test_X = test_X.astype('float32')
train_X = train_X / 255.
test_X = test_X / 255.
```

Vector One-hot Encoding para la red

La clase de cada imagen se convierte a un vector en el que todos los números son cero excepto el que se corresponde con la clase a la que pertenece la imagen, que toma el valor uno.

```
train_Y_one_hot = to_categorical(train_Y)
test_Y_one_hot = to_categorical(test_Y)
print('Original label:', train_Y[0])
print('After conversion to one-hot:', train_Y_one_hot[0])
```

Set de Entrenamiento y Validación (80-20)

```
train_X,valid_X,train_label,valid_label =
train_test_split(train_X,
train_Y_one_hot, test_size=0.2, shuffle=True)
print(train_X.shape,valid_X.shape,train_label.shape,
valid_label.shape)
```

Crear el modelo de CNN

```
epochs =100
INIT_LR = 1e-3
batch_size = 64
usv_model = Sequential()
usv_model.add(Conv2D(filters=128, kernel_size=(3,3),
padding='same', activation
usv_model.add(Conv2D(filters=128, kernel_size=(3,3),
padding='same', activation
usv_model.add(AveragePooling2D())
usv_model.add(Dropout(0.2))
usv_model.add(Conv2D(filters=128, kernel_size=(3,3),
padding='same', activation
usv_model.add(AveragePooling2D())
```

```
usv_model.add(Dropout(0.2))
usv_model.add(Conv2D(filters=128, kernel_size=(3,3),
padding='same', activation
usv_model.add(AveragePooling2D())
usv_model.add(Dropout(0.2))
usv_model.add(Flatten())
usv_model.add(Dense(512, activation='relu'))
usv_model.add(Dropout(0.2))
usv_model.add(Dense(len(classes), activation='softmax'))
usv_model.build()
usv_model.summary()
Compilación
usv_model.compile(loss='categorical_crossentropy',optimizer='Ada
m',metrics=['accuracy'])
metrics=['accuracy'])
Entrenar el modelo: Aprende a clasificar imágenes
usv_train = usv_model.fit(train_X, train_label, batch_size=
batch_size,epochs= epochs
Guardar la red
usv_model.save("Modelo_usv.h5py")
Evaluar la red (resultados)
test_eval = usv_model.evaluate(test_X, test_Y_one_hot,
verbose=1)
print('Función de pérdida:', test_eval[0])
print('Precisión:', test_eval[1])
Y_pred=usv_model.predict(test_X)
usv_pred = usv_model.predict(test_X, batch_size=32, verbose=1)
usv_predicted = np.argmax(usv_pred, axis=1)
La matriz de confusión
usv_cm = confusion_matrix(np.argmax(test_Y_one_hot, axis=1),
usv_predicted)
```

```
usv_df_cm = pd.DataFrame(usv_cm, range(len(classes)),
range(len(classes)))
plt.figure(figsize = (20,17))
sn.set(font_scale=1)
sn.heatmap(usv_df_cm, annot=False, annot_kws={"size": 12})
plt.show()
Gráficas
accuracy = usv_train.history['accuracy']
val_accuracy = usv_train.history['val_accuracy']
loss = usv_train.history['loss']
val_loss = usv_train.history['val_loss']
epochs = range(len(accuracy))
plt.figure(figsize=(14, 12))
plt.subplot(2, 2, 1)
plt.plot(epochs, accuracy, 'bo', label='Training accuracy')
plt.plot(epochs, val_accuracy, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.legend()
plt.subplot(2, 2, 2)
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.show()
Evaluación
predicted_classes2 = usv_model.predict(test_X)

predicted_classes=[]
for predicted_usv in predicted_classes2:
      predicted_classes.append(predicted_usv.tolist().index(max(
predicted_usv)     ))
```

```
predicted_classes=np.array(predicted_classes)
predicted_classes.shape, test_Y.shape
target_names = ["Class {}".format(i) for i in range(nClasses)]
print(classification_report(test_Y, predicted_classes,
target_names=target_names

Curva ROC
fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(nClasses):
      fpr[i], tpr[i], _ =
roc_curve(test_Y_one_hot[:,i],Y_pred[:,i])
      roc_auc[i] = auc(fpr[i], tpr[i])
fpr["micro"], tpr["micro"], _ =
roc_curve(test_Y_one_hot.ravel(), Y_pred.ravel
roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])
all_fpr = np.unique(np.concatenate([fpr[i] for i in
range(nClasses)]))
mean_tpr = np.zeros_like(all_fpr)
for i in range(nClasses):
      mean_tpr += interp(all_fpr, fpr[i], tpr[i])
mean_tpr /= nClasses
fpr["macro"] = all_fpr
tpr["macro"] = mean_tpr
roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])
plt .figure()
lw=2
plt. plot(fpr[ "micro"], tpr[ "micro" ],
      label='micro-average ROC curve (area= {0:0.2f})'
            '' .format(roc_auc[ "micro" ]),
            color='deeppink' , linestyle=':' , linewidth=4)
```

```
plt.plot(fpr[ "macro"], tpr[ "macro" ],
            label='macro-average ROC curve (area= {0:0.2f})'
            '' .format(roc_auc[ "macro" ]),
            color='navy' , linestyle=':' , linewidth=4)
colors = cycle([ 'c', 'm', 'y', 'k', 'aqua' , 'darkorange' ,
'cornflowerblue'])
for i, color in zip( range (nClasses), colors):
      plt.plot(fpr[i], tpr[i], color=color, lw=lw,
            label='ROC curve of class {0} (area = {1:0.2f})'
            '' .format(i, roc_auc[i]))
plt.plot([0, 1], [0, 1], 'k--' , lw=lw)
plt.xlim([ 0.0, 1.0])
plt.ylim([ 0.0, 1.05])
plt.xlabel( 'False Positive Rate' )
plt.ylabel( 'True Positive Rate' )
plt.title( 'ROC curves' )
plt.legend(loc="lower right")
plt. show()
```