DESARROLLO DE MODELO DE CLASIFICACIÓN DE SUELO URBANO RECREATIVO BASADO EN DEEP LEARNING USANDO IMÁGENES SATELITALES

Jairo David Chía Bejarano
Nicolás Castaño Cardona

Proyecto Aplicado para optar al título de
Magister en Ciencia de Datos

Director(a)
Omar Andrés Castaño Idárraga

FACULTAD DE INGENIERÍA Y CIENCIAS
MAESTRÍA EN CIENCIA DE DATOS
SANTIAGO DE CALI, DICIEMBRE 5 DE 2024

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

INTRODUCCIÓN

El acelerado crecimiento urbano experimentado en América Latina en las últimas décadas ha ejercido gran presión sobre los espacios públicos destinados a áreas verdes y recreación como parques, plazas y canchas deportivas. La creciente densificación de las ciudades no se ha visto acompañada de una adecuada ampliación de estas zonas para dar cobertura a la mayor población, resultando frecuentemente insuficientes [1].

Esta situación se ve agravada por la falta de información precisa y actualizada, la cual debería estar soportada en imágenes que evidencien la localización y extensión territorial de los existentes espacios recreativos urbanos para así permitir una adecuada planificación. Frente a esto, los métodos tradicionales de machine learning en teledetección, como uso de índices de vegetación o clasificación digital mediante algoritmos como Random Forest, presentan limitaciones para una identificación detallada en imágenes satelitales de alta resolución debido a la confusión espectral con otros objetos [2].

Ante esta problemática, el presente proyecto tiene como propósito de implementar un modelo automático para la detección y clasificación de parques y canchas deportivas en imágenes satelitales de zonas urbanas mediante la aplicación de redes neuronales convolucionales y técnicas de aprendizaje profundo.

Específicamente, se ha implementado una Red Neuronal Convolucional (CNN) capaz de extraer características distintivas de los objetos de interés, buscando superar los métodos convencionales antes mencionados, evaluando su efectividad en términos de precisión y conteo. Esta CNN optimizada ha sido entrenada con un riguroso conjunto de datos etiquetados y que han sido dispuestos para el uso público.

Por medio de este proyecto aplicamos un modelo que permite clasificar y estimar la cantidad de estas áreas recreativas en diferentes zonas de la ciudad, el cual pude servir como base para generar posteriormente datos actualizados de cobertura espacial y así detectar posibles déficits en algunas regiones, información que es valiosa para sustentar políticas públicas de expansión o construcción ante el crecimiento poblacional. Además, este proyecto puede orientar a posteriores usuarios de este en la aplicación de esta propuesta de modelo de manera diferente en otros aspectos urbanos de las ciudades.

Los scripts y la estructura del modelo están publicados en repositorios públicos para uso de la comunidad con su respectiva documentación, además de una guía detallada que permite orientar a posteriores usuarios del modelo en su aplicación. Así se espera proveer una solución complementaria para automatizar el monitoreo de zonas recreativas urbanas, fundamentales para la habitabilidad y salud pública en contextos de alta urbanización

1. DEFINICIÓN DEL PROBLEMA

PLANTEAMIENTO DEL PROBLEMA

El espacio público urbano, que abarca áreas como parques, plazas y escenarios culturales al aire libre, desempeña un papel crucial en la calidad y planificación urbana al facilitar el intercambio social y promover actividades recreativas. Aunque la evaluación tradicional se basa en indicadores cuantitativos y cualitativos, como los déficits de espacio público, estas métricas revelan limitaciones para comprender problemas urbanos complejos [3]. Investigaciones recientes destacan la necesidad de enfoques más avanzados para una evaluación precisa, resaltando la importancia de considerar detalladamente el espacio público en el contexto de la planificación urbana para lograr ciudades más habitables y funcionales.

El mapeo preciso de usos de suelo urbano mediante la identificación de áreas verdes y recreativas a partir de imágenes satelitales constituye un desafío crítico en la planificación y gestión urbanas. Los métodos tradicionales de clasificación automática, como Random Forest y SVM, han demostrado limitaciones en la detección precisa de ciertas clases de objetos y espacios [4]. La incapacidad de estos enfoques para lograr una clasificación de imágenes detallada impide la generación eficiente de inventarios georreferenciados de tales áreas urbanas.

La situación actual se ve agravada por la creciente necesidad de datos precisos para respaldar decisiones de planificación y políticas urbanas, subrayando la urgencia de avanzar en la capacidad de clasificación y segmentación de imágenes satelitales. Este problema, en el ámbito de la ciencia de datos, requiere una solución que vaya más allá de los métodos convencionales de machine learning. La relevancia de esta investigación radica en su potencial para mejorar significativamente la precisión del mapeo de usos de suelo urbano, especialmente en la identificación de espacios públicos, facilitando así una planificación urbana más informada y eficaz.

FORMULACIÓN DEL PROBLEMA

¿Cómo implementar un modelo de clasificación de imágenes a partir de Redes Neuronales Convolucionales (CNN) que permita clasificar, identificar y contar espacios públicos abiertos, como parques y canchas deportivas, a partir de imágenes satelitales?

En el contexto de la planificación urbana, la identificación precisa de usos de suelo y, en particular, la clasificación detallada de espacios públicos abiertos, como parques y canchas deportivas, es

esencial para una gestión efectiva de las ciudades. Los métodos convencionales de clasificación automática de imágenes satelitales han demostrado limitaciones en la detección de estas áreas específicas, lo que destaca la necesidad de enfoques más avanzados.

Teniendo en cuenta esto, surgen las siguientes incógnitas: ¿Cómo podemos diseñar y entrenar un modelo de redes neuronales convolucionales que obtenga mejores resultados que los obtenidos por los métodos tradicionales de machine learning para la clasificación de estas áreas urbanas? ¿Cuál es la efectividad comparativa de este modelo frente a métodos tradicionales de machine learning, en términos de precisión y detección de espacios públicos? ¿Cómo se desempeña el modelo de Redes Neuronales Convolucionales (CNN) cuando se aplica en un entorno práctico utilizando imágenes satelitales actuales para la identificación y clasificación de espacios públicos recreativos en un área urbana específica?

Resolver estas preguntas orientaron el proyecto hacia el desarrollo de un enfoque innovador en ciencia de datos para el mapeo detallado de usos de suelo urbano, particularmente en la identificación precisa de espacios públicos recreativos, contribuyendo así a una planificación urbana más informada y eficaz.

2. OBJETIVOS DEL PROYECTO

2.1. OBJETIVO GENERAL

Implementar un modelo de clasificación de imágenes a partir de Redes Neuronales Convolucionales (CNN) que identifique, clasifique y cuente espacios públicos recreativos en suelo urbano haciendo uso de imágenes satelitales.

2.2. OBJETIVOS ESPECÍFICOS

- Desarrollar el modelo de redes neuronales convolucionales incorporando técnicas avanzadas de Deep Learning para lograr una clasificación precisa de áreas recreativas en imágenes satelitales urbanas.

- Aplicar el modelo de Redes Neuronales Convolucionales (CNN) desarrollado en un entorno práctico, utilizando imágenes satelitales actuales, y evaluar su desempeño en la identificación y clasificación de espacios públicos recreativos en un área urbana específica.

- Evaluar la efectividad del modelo propuesto mediante comparaciones contra métodos tradicionales de clasificación automática, como por ejemplo Random Forest, en términos de identificación, precisión y conteo de espacios públicos recreativos urbanos.

3. MARCO TEÓRICO Y ANTECEDENTES

3.1. MARCO TEÓRICO

En este capítulo se establece un fundamento teórico relacionado con la investigación sobre el mapeo detallado de usos de suelo urbano, enfocado en la identificación de espacios públicos abiertos. Se abordarán definiciones clave y desafíos en la planificación urbana, seguido de un análisis de métodos tradicionales y las limitaciones en su precisión. Se explorarán algunos casos en la aplicación exitosa de Deep Learning y Redes Neuronales Convolucionales en la clasificación de objetos en imágenes. La evaluación comparativa entre modelos convencionales y basados en Deep Learning, junto con estrategias para mejorar la robustez, completará esta sección.

3.1.1. Introducción al Mapeo de Usos de Suelo Urbano

La complejidad de las áreas urbanas contemporáneas demanda una comprensión minuciosa y actualizada de los usos de suelo que las componen. El mapeo de usos de suelo urbano emerge como un componente crucial en la planificación y gestión efectiva de ciudades en constante cambio y crecimiento. Este proceso no solo se limita a una mera catalogación de las diferentes funciones y actividades presentes en el entorno urbano, sino que representa una herramienta poderosa para abordar desafíos inherentes a la expansión urbana, la optimización de recursos y la creación de entornos urbanos habitables y sostenibles [5].

En el contexto de este proyecto, se da especial énfasis a la definición de usos de suelo urbano como un factor determinante para la toma de decisiones en el desarrollo urbano sostenible. La precisión en la delimitación de estos usos no solo es esencial para la adecuada planificación de infraestructuras y servicios, sino que también impacta directamente en la calidad de vida de los habitantes. En este sentido, uno de los elementos cruciales dentro del amplio espectro de usos de suelo es la identificación de espacios públicos abiertos [6].

La importancia de los espacios públicos abiertos en la configuración de entornos urbanos saludables y accesibles no puede ser subestimada. Más allá de su función estética, estos espacios sirven como puntos de encuentro, recreación y actividad física, contribuyendo a la cohesión social y al bienestar general de la población urbana. Los gobiernos, por medio de sus entidades públicas, establecen parámetros y herramientas que le permite a las ciudades poder desarrollarse conforme a estructuras ordenadas que permiten dar cabida a espacios que contribuyen a la realización de actividades de bienestar social, tal es el caso de los Planes de Ordenamiento Territorial en Colombia, cuyos principios establecen una directriz para la adecuada distribución del uso del suelo en el espacio dentro del ordenamiento territorial de municipios y distritos.

No obstante, los desafíos asociados con la identificación precisa y mapeo detallado de estos espacios en el contexto urbano son significativos. La necesidad de superar estas dificultades se

convierte en un catalizador para la investigación propuesta, donde se busca avanzar en la eficacia de la identificación de espacios públicos abiertos mediante enfoques innovadores basados en técnicas de aprendizaje profundo y Redes Neuronales Convolucionales.

3.1.2. Clasificación Automática de objetos y espacios en Imágenes Satelitales

La clasificación automática de objetos y espacios en imágenes satelitales es un tema de gran interés en la ciencia de datos. Los métodos tradicionales de machine learning, como Random Forest y SVM, han sido ampliamente utilizados para este propósito. Sin embargo, estos métodos tienen limitaciones en la precisión de la clasificación de objetos y espacios en imágenes satelitales. Algunas de las limitaciones y desafíos que se presentan en estos métodos tradicionales son la falta de capacidad para manejar grandes cantidades de datos, la necesidad de una gran cantidad de recursos computacionales y la dificultad para manejar datos no estructurados [7]. En los últimos años, se han desarrollado avances en técnicas de machine learning para la clasificación de objetos y espacios en imágenes satelitales. Estos avances incluyen el uso de Redes Neuronales Convolucionales (CNN), que han demostrado ser muy efectivas en la clasificación de objetos y espacios en imágenes satelitales. Las CNN son capaces de aprender características de las imágenes satelitales a través de múltiples capas, lo que les permite identificar patrones complejos en los datos [8].

La clasificación detallada de objetos complejos en escenas urbanas es un desafío importante en la ciencia de datos. Aunque se han desarrollado técnicas avanzadas de machine learning para la clasificación de objetos y espacios en imágenes satelitales, la clasificación detallada de objetos complejos en escenas urbanas sigue siendo un problema difícil de resolver. Algunas de las limitaciones que se presentan en la clasificación detallada de objetos complejos en escenas urbanas son la falta de datos etiquetados, la variabilidad en la apariencia de los objetos y la complejidad de las escenas urbanas [9].

Para abordar estas limitaciones, se han propuesto diversas técnicas de machine learning, como el aprendizaje por transferencia. Sin embargo, estas técnicas aún no son capaces de clasificar detalladamente objetos complejos en escenas urbanas con la misma precisión que los humanos [10].

Algunos estudios han propuesto el uso de técnicas de aprendizaje profundo para la clasificación detallada de objetos complejos en escenas urbanas. Por ejemplo, en el artículo desarrollado por Rodríguez [11], donde los resultados experimentales muestran que el enfoque propuesto supera a los métodos tradicionales de machine learning en términos de precisión.

3.1.3. Aplicación de Deep Learning y Redes Neuronales Convolucionales (CNN) en Visión por Computadora aplicada en imágenes satelitales.

La visión por computadora es un campo de la inteligencia artificial que se enfoca en permitir a las computadoras interpretar y comprender el mundo visual. Consiste en la extracción automatizada

de información de las imágenes, abarcando desde modelos 3D hasta reconocimiento de objetos y búsqueda de contenido [12]. El Deep Learning es una rama del Machine Learning que se centra en el aprendizaje de representaciones de datos a través de múltiples capas de procesamiento. Dentro de este ámbito, las Redes Neuronales Convolucionales (CNN) son un tipo de red neuronal profunda comúnmente utilizadas en aplicaciones de visión por computadora. Las CNN buscan imitar el funcionamiento y la estructura de las redes neuronales cerebrales del ser humano, estando formadas por nodos interconectados que reciben información, la procesan y la transmiten a través de otras neuronas artificiales [13].

En el contexto de visión por computadora, las CNN han demostrado ser muy efectivas en la clasificación de objetos en imágenes. Pueden detectar características y patrones de una imagen, lo que les permite reconocer objetos, clases y categorías. Se han utilizado en una amplia variedad de aplicaciones, incluyendo la detección de objetos, la segmentación de imágenes, la clasificación de imágenes y la generación de imágenes [14].

Comparadas con métodos clásicos de machine learning, las CNNs presentan dos grandes diferencias: primero, aprenden estas representaciones directamente desde los datos sin requerir diseño manual de descriptores. Segundo, incorporan invariancia espacial al compartir pesos en las convoluciones, reduciendo el overfitting o sobre-ajuste [15].

Estas propiedades hacen que las CNNs sean particularmente útiles para clasificar escenas complejas en imágenes de satélite. Trabajos como el de Karypidis [16] demuestran desempeños muy superiores de CNNs frente a técnicas tradicionales de machine learning en tareas de clasificación de imágenes. El modelo basado en CNN demuestra mejores porcentajes de exactitud en cuanto a los resultados en varios casos de prueba.

Mediante operaciones de convolución, las CNNs incorporan nociones de jerarquía, compartiendo parámetros a través de las distintas capas para identificar patrones cada vez más abstractos y poder reducir el sobreajuste, logrando así una buena generalización a nuevos conjuntos de datos. Además, por su naturaleza diferenciable, las CNNs pueden entrenarse de manera supervisada a gran escala mediante el algoritmo de backpropagation y gradiente descendente para optimizar su rendimiento en la tarea objetivo. Estas bondades han impulsado la aplicación de CNNs en diversos estudios de percepción remota [17].

3.1.4. Evaluación de Modelos de Clasificación de Objetos en Imágenes Satelitales.

La clasificación de objetos en imágenes satelitales es una tarea importante en el procesamiento de imágenes y la teledetección. La evaluación de estos modelos es crucial para determinar la precisión y la eficacia de estos. En este sentido, se han propuesto varios métodos de evaluación de modelos de clasificación de objetos en imágenes satelitales, como la matriz de confusión, el Accuracy, Recall, entre otros [18].

Tradicionalmente, los clasificadores como Random Forests o SVM eran validados mediante particionamiento aleatorio en conjuntos de entrenamiento y test, y calculando métricas agregadas sobre dichas particiones como la exactitud (accuracy) global, matrices de confusión o curvas ROC. No obstante, en escenarios complejos con múltiples clases no balanceadas, se requieren métricas más granulares como la precisión y exhaustividad por clase [19].

La evaluación rigurosa de los modelos de clasificación de imágenes constituye una parte fundamental en su desarrollo, pues permite cuantificar y comparar su capacidad predictiva antes de ser implementados en aplicaciones del mundo real. En el contexto de la percepción remota y la clasificación de objetos urbanos en imágenes satelitales, han surgido nuevos desafíos evaluativos con la introducción de métodos de deep learning [20].

Con la adopción de técnicas más complejas como las CNNs que pueden sobre-ajustarse fácilmente, cobra importancia el uso de conjuntos independientes de validación, y métricas como el error en ese conjunto. La generalización del modelo a nuevos lugares resulta un factor de evaluación crítico, por lo que se analiza su desempeño frente a imágenes de áreas geográficas distintas [21].

En cuanto a la comparación entre modelos convencionales y deep learning para la clasificación de objetos en imágenes satelitales, se ha demostrado que los modelos de deep learning superan a los modelos convencionales en términos de precisión y eficacia. Los modelos de deep learning, como las redes neuronales convolucionales (CNN), son capaces de aprender características complejas de las imágenes satelitales y, por lo tanto, son más precisos en la clasificación de objetos en imágenes satelitales [22].

Los factores clave en la evaluación de modelos para la clasificación de objetos en imágenes satelitales incluyen la selección de características, la selección de algoritmos de clasificación, la selección de conjuntos de datos de entrenamiento y prueba, la selección de parámetros de modelos, la validación cruzada, entre otros. Estos factores permiten realizar comparativas sistemáticas entre distintos algoritmos bajo métricas diversas, utilizando particionamientos adecuados de los datos y evaluando la robustez del modelo, lo cual resulta ser indispensable para garantizar su capacidad predictiva antes de implementar clasificadores de objetos en imágenes satelitales.

3.1.5. Random Forest

El algoritmo Random Forest es un método de aprendizaje automático ampliamente utilizado tanto para problemas de clasificación como de regresión. Su estructura se basa en la construcción de múltiples árboles de decisión, formando un "bosque" de modelos, donde cada árbol aporta una predicción independiente. Luego, el resultado final del modelo se obtiene a través de la votación mayoritaria (para clasificación) o el promedio (para regresión) de las predicciones de estos árboles. Este enfoque es una forma de ensemble learning, lo que significa que se combinan

múltiples modelos individuales para mejorar la precisión general.

Una de las principales ventajas de Random Forest es su capacidad para manejar grandes conjuntos de datos sin caer en el overfitting. Mientras que un árbol de decisión único tiende a sobre ajustarse a los datos de entrenamiento, al agregar más árboles y promediando sus resultados, Random Forest reduce el riesgo de sesgo y varianza en las predicciones. La clave está en que cada árbol se entrena en un subconjunto aleatorio de datos y características, un método conocido como bagging (bootstrap aggregation) [23].

El proceso de construcción de un Random Forest involucra varios pasos. Primero, se toman muestras aleatorias de los datos de entrenamiento con reemplazo (bootstrap sampling), lo que significa que algunas observaciones pueden repetirse en diferentes árboles. Luego, para cada árbol, en lugar de considerar todas las características para dividir los nodos, se selecciona un subconjunto aleatorio de características, lo que se conoce como feature bagging. Esta estrategia introduce una mayor diversidad entre los árboles, lo que contribuye a un mejor desempeño del modelo en comparación con un solo árbol de decisión que consideraría todas las características en cada nodo.

Además, el conjunto de datos no utilizado para entrenar cada árbol se conoce como out-of-bag (OOB) sample. Este conjunto se utiliza para validar el modelo de manera interna y obtener estimaciones de precisión sin necesidad de separar explícitamente datos de prueba, lo que hace que Random Forest sea eficiente en la validación cruzada.

Random Forest presenta numerosas ventajas, entre las cuales destacan:

- **Reducción del riesgo de sobreajuste**: Al combinar múltiples árboles, se disminuye el riesgo de que el modelo se ajuste excesivamente a los detalles del conjunto de datos de entrenamiento.
- **Manejo de datos faltantes**: Debido a su capacidad para trabajar con subconjuntos aleatorios de características, Random Forest puede manejar conjuntos de datos con valores faltantes de manera más robusta.
- **Evaluación de la importancia de las variables**: Random Forest facilita la interpretación de la relevancia de las características del modelo, lo que permite identificar cuáles son las más importantes para predecir el resultado.

Sin embargo, Random Forest también presenta algunos desafíos. Entre ellos, su complejidad computacional es notable, ya que para obtener una predicción se deben evaluar varios árboles, lo que puede ser costoso en términos de tiempo y recursos. Además, aunque se puede interpretar la importancia de las características, la interpretación del modelo completo es menos directa que con un solo árbol de decisión.

El algoritmo Random Forest se utiliza en diversas áreas. En el ámbito financiero, por ejemplo, es valioso para la detección de fraudes y la predicción de riesgos crediticios. En el campo de la salud, se aplica para analizar grandes volúmenes de datos y ayudar en diagnósticos o predicciones de resultados clínicos. También es común en problemas de marketing, donde se predicen las

probabilidades de respuesta de los clientes a campañas específicas.

3.1.6. Supported Vector Machine

Support Vector Machines (SVM) son un algoritmo de aprendizaje automático poderoso y ampliamente utilizado, especialmente eficaz para tareas de clasificación. La idea principal detrás de las SVM es encontrar el hiperplano óptimo que separe los puntos de datos de diferentes clases en un espacio de características. El objetivo es maximizar el margen entre el hiperplano y los puntos de datos más cercanos de cada clase, denominados vectores de soporte. Este margen óptimo ayuda a que las SVM generalicen bien en datos no vistos [24].

Las SVM funcionan mapeando los datos de entrada a un espacio de características de alta dimensión donde es más fácil encontrar una separación lineal. Si los datos no son separables linealmente en el espacio de características original, las SVM utilizan una técnica llamada truco del núcleo para transformar los datos a una dimensión más alta, donde puede encontrarse un separador lineal. Algunos núcleos comunes utilizados para esta transformación incluyen el lineal, el polinomial y el de función de base radial (RBF).

En su forma más simple, las SVM se utilizan para datos linealmente separables, donde un hiperplano lineal puede distinguir entre clases. El concepto clave es maximizar la distancia entre el hiperplano y los puntos de datos más cercanos de cada clase, lo que asegura una separación robusta. La ecuación de un modelo SVM lineal puede expresarse como: $f(x) = w \cdot x + b$ donde *w* representa el vector de pesos y *b* es el término de sesgo.

Por ejemplo, en un escenario de clasificación binaria, el algoritmo crea una frontera de decisión que separa dos clases. Los puntos en un lado de la frontera pertenecen a una clase, mientras que los puntos en el otro lado pertenecen a la otra clase. La distancia entre el hiperplano y los vectores de soporte se llama *margen*, y las SVM buscan maximizar este margen para mejorar el rendimiento de la clasificación.

En cuanto a la SVM no lineal, se puede decir que, con frecuencia, los datos del mundo real no son linealmente separables, y aquí es donde entra en juego el truco del núcleo. Las SVM pueden clasificar eficientemente datos no lineales separables transformando el espacio de características original mediante núcleos no lineales como RBF o polinomial. El núcleo de función de base radial es uno de los más utilizados por su capacidad para manejar relaciones más complejas entre las características.

En las SVM no lineales, el algoritmo esencialmente crea fronteras de decisión que son curvas en el espacio de características original, pero que se vuelven lineales en el espacio transformado de mayor dimensión. Esta flexibilidad hace que las SVM sean una herramienta muy poderosa para datos tanto linealmente como no linealmente separables.

Varios parámetros importantes se utilizan en las SVM:

- **Parámetro C**: Este es un parámetro de regularización que controla el equilibrio entre maximizar el margen y minimizar los errores de clasificación. Un valor pequeño de C

permite más errores de clasificación, mientras que un valor grande de C intenta clasificar correctamente todos los ejemplos de entrenamiento.

- **Función de núcleo**: Esta función define la transformación de los datos de entrada en un espacio de características de mayor dimensión. Algunos núcleos comunes incluyen:
    - Núcleo lineal: Adecuado para datos linealmente separables.
    - Núcleo polinomial: Útil para problemas con fronteras de decisión más complejas.
    - Núcleo RBF: Ampliamente utilizado para la clasificación de datos no lineales.

En cuanto a las ventajas de SVM:

1. Eficaces en espacios de alta dimensionalidad: Las SVM son muy eficientes cuando se trata de datos con muchas características.
2. Eficientes en memoria: Dado que utilizan un subconjunto de puntos de entrenamiento (los vectores de soporte), las SVM requieren menos memoria en comparación con otros algoritmos como los k-vecinos más cercanos (KNN).
3. Versátiles con diferentes núcleos: La flexibilidad del truco del núcleo permite a las SVM manejar una variedad de estructuras de datos, tanto lineales como no lineales.

Por otro lado, en sus desventajas encontramos:

1. Tiempo de entrenamiento: Las SVM pueden ser computacionalmente costosas y lentas, especialmente cuando el conjunto de datos es grande.
2. Elección del núcleo: Seleccionar la función de núcleo correcta y ajustar los parámetros puede ser complejo y requiere experimentación.
3. Difíciles de interpretar: Los modelos SVM, especialmente con núcleos no lineales, a menudo no son tan interpretables como los modelos lineales.

Las SVM se utilizan con frecuencia en aplicaciones como la clasificación de textos, el reconocimiento de imágenes y la bioinformática. Por ejemplo, en la clasificación de imágenes, las SVM pueden utilizarse para distinguir entre diferentes tipos de objetos basados en intensidades de píxeles u otras características extraídas de la imagen.

3.1.7. K-nearest neighbors (KNN)

El algoritmo de *K-nearest neighbors* (KNN) es uno de los métodos más simples y efectivos en el ámbito de la clasificación y la regresión en aprendizaje automático. Se basa en el principio de que objetos similares tienden a estar cerca unos de otros. Por ello, KNN busca los “k” vecinos más cercanos de un punto de interés y, según los datos de esos vecinos, asigna una etiqueta a dicho punto o realiza una predicción.

KNN es un algoritmo no paramétrico, lo que significa que no asume ninguna suposición particular

sobre la distribución de los datos. A diferencia de otros algoritmos de clasificación, como las máquinas de soporte vectorial (SVM), KNN no construye un modelo durante la fase de entrenamiento. En su lugar, almacena todos los ejemplos de entrenamiento y realiza las predicciones basándose en los datos de entrada. Esta característica convierte a KNN en un algoritmo basado en la instancia, donde las predicciones se hacen directamente al consultar los datos almacenados [25].

Para cada nuevo dato que se quiere clasificar o predecir, el algoritmo busca los “k” puntos de datos más cercanos en el conjunto de entrenamiento y realiza la clasificación basándose en la mayoría de las etiquetas de estos vecinos. En el caso de la regresión, la predicción es el promedio de los valores de los vecinos más cercanos.

Una de las principales ventajas de KNN es su simplicidad, lo que lo convierte en una excelente opción para problemas donde la relación entre las variables de entrada y la salida es compleja y no lineal. Además, no necesita una fase de entrenamiento prolongada como algunos otros modelos, ya que simplemente almacena los ejemplos de entrenamiento.

Sin embargo, esta simplicidad también conlleva ciertos inconvenientes. El costo computacional de KNN puede ser alto, especialmente en conjuntos de datos grandes, ya que se requiere calcular la distancia entre el nuevo punto de datos y todos los puntos de entrenamiento. Esto puede ralentizar significativamente el proceso cuando se trabaja con grandes volúmenes de datos o datos de alta dimensionalidad. Otro desafío importante es la selección del valor de “k”, ya que un valor inapropiado puede llevar a predicciones inexactas.

KNN se utiliza en una amplia variedad de aplicaciones, incluyendo la clasificación de imágenes, el análisis de datos biométricos y la predicción de enfermedades. Dado que es un algoritmo sencillo, a menudo se utiliza como punto de referencia para medir el rendimiento de otros algoritmos más complejos.

Debido a los problemas que enfrenta en espacios de alta dimensionalidad (el llamado “problema de la maldición de la dimensionalidad”), se han desarrollado diversas optimizaciones. Estas incluyen métodos de reducción de dimensionalidad como el análisis de componentes principales (PCA), que reduce el número de características al seleccionar las más relevantes. También se han implementado estructuras de datos especializadas como árboles KD y Ball Trees para mejorar la eficiencia en la búsqueda de los vecinos más cercanos.

En situaciones donde los conjuntos de datos son muy grandes, es común usar paralelización, con técnicas como el uso de GPUs para dividir la carga de trabajo y acelerar los cálculos. Además, se han desarrollado versiones aproximadas de KNN que reducen el costo computacional sin sacrificar demasiado la precisión [25].

3.1.8. Métricas de desempeño.

En el ámbito del Machine Learning y la clasificación, las métricas de desempeño como la accuracy, la precisión (precision), el recall, y el F1-score son fundamentales para evaluar la efectividad de un modelo predictivo. Cada una de estas métricas se calcula a partir de la matriz de confusión,

una tabla que resume los resultados de clasificación y que contiene los valores de verdaderos positivos (TP), falsos positivos (FP), verdaderos negativos (TN), y falsos negativos (FN).

- **Accuracy (Exactitud)**

La accuracy mide la proporción de predicciones correctas con respecto al total de predicciones realizadas. Es una métrica muy utilizada cuando las clases están equilibradas, es decir, cuando los datos contienen un número similar de ejemplos de cada clase [26].

La fórmula para calcular la accuracy es:

$$Accuracy = (TP + TN)/(TP + TN + FP + FN)$$

Donde:

- **TP** (True Positives): Casos correctos donde la clase real y la clase predicha son positivas.
- **TN** (True Negatives): Casos correctos donde la clase real y la clase predicha son negativas.
- **FP** (False Positives): Casos donde se predice incorrectamente la clase positiva.
- **FN** (False Negatives): Casos donde se predice incorrectamente la clase negativa.

La *accuracy* es útil cuando las clases están balanceadas, pero puede ser engañosa si hay un desbalance de clases (por ejemplo, en un conjunto de datos donde el 95% son negativos y el 5% son positivos, un modelo que prediga todo como negativo tendrá una alta *accuracy* pero será ineficaz).

- **Precision (Precisión)**

La *precisión* mide cuán preciso es el modelo al predecir una clase positiva. Es decir, indica qué proporción de las predicciones positivas realizadas por el modelo son correctas. Es especialmente útil en contextos donde los falsos positivos son costosos (como en diagnósticos médicos o detección de fraude) [26].

La fórmula para la *precisión* es:

$$Precision = TP/(TP + FP)$$

Una alta *precisión* indica que un modelo tiene pocos falsos positivos.

- **Recall (Sensibilidad o Exhaustividad)**

El recall mide la capacidad del modelo para identificar correctamente todas las instancias positivas. Es especialmente importante en escenarios donde los falsos negativos son más costosos (como en la detección de enfermedades graves) [26].

La fórmula para el recall es:

$$Recall = TP/(TP + FN)$$

Un *recall* alto indica que el modelo está capturando la mayoría de las instancias positivas, pero esto puede ocurrir a expensas de tener más falsos positivos.

- **F1-score**

El *F1-score* es una métrica que combina la *precisión* y el *recall* en una única medida. Es útil cuando se quiere equilibrar ambas métricas y es especialmente valioso cuando las clases están desbalanceadas o cuando hay una necesidad de equilibrar entre falsos positivos y falsos negativos [26].

La fórmula del *F1-score* es:

$$F1 = 2\ x\ (Precision\ x\ Recall)/(Precision + Recall)$$

Un valor alto de *F1-score* indica un buen balance entre precisión y *recall*. Sin embargo, en algunos casos específicos, se puede preferir optimizar más una métrica sobre la otra.

3.1.9. Redes Neuronales

Las redes neuronales están inspiradas en las conexiones neuronales del cerebro humano y se utilizan principalmente en el aprendizaje profundo (deep learning). El componente básico es la neurona artificial o perceptrón, el cual procesa una entrada mediante una combinación lineal de pesos, aplicando una función de activación no lineal como la función sigmoide, tangente hiperbólica o ReLU (Rectified Linear Unit). Estas neuronas están organizadas en capas, que suelen clasificarse en:

1. **Capa de entrada**: Recibe los datos brutos.
2. **Capas ocultas**: Procesan las entradas aplicando transformaciones y activaciones.
3. **Capa de salida**: Produce el resultado final.

El aprendizaje de la red se realiza ajustando los pesos mediante retropropagación (backpropagation), minimizando el error a través de un algoritmo de optimización como el descenso de gradiente [27].

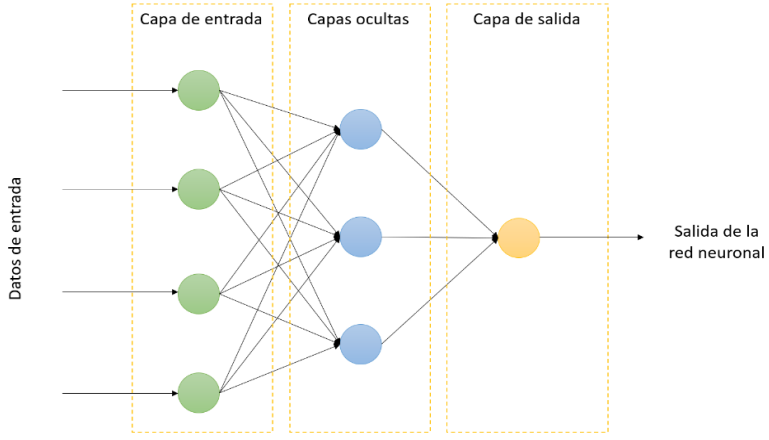


Figura 1. *Estructura básica de una Red Neuronal. Fuente: [28]*

Existen diferentes tipos de redes neuronales, cada una adecuada para distintas tareas y proyectos. Por ejemplo, las Redes Neuronales Artificiales (ANNs) son la forma más básica y están compuestas por capas completamente conectadas, utilizadas comúnmente para tareas de clasificación y regresión en datos tabulares. Las Redes Neuronales Convolucionales (CNNs), por otro lado, están diseñadas para procesar datos espaciales, como imágenes, y son la elección predominante en problemas de visión por computadora, como la clasificación de imágenes y la detección de objetos [28].

Las Redes Neuronales Recurrentes (RNNs) están optimizadas para procesar datos secuenciales, siendo útiles en tareas como el procesamiento del lenguaje natural y la predicción de series temporales. Un tipo avanzado de RNN es la LSTM (Long Short-Term Memory), que resuelve el problema del desvanecimiento de gradientes y permite manejar secuencias largas, siendo clave en proyectos de traducción automática y generación de texto. Finalmente, las Redes Generativas Antagónicas (GANs) han ganado popularidad en la generación de datos sintéticos, incluidas imágenes y videos, y son ampliamente utilizadas en el campo del arte digital y la creación de contenido [29].

3.1.10. Redes Neuronales Convolucionales (CNN)

Las redes neuronales convolucionales (CNN) son un tipo especial de red neuronal profunda diseñada específicamente para procesar y analizar datos estructurados en una matriz, como imágenes. A diferencia de las redes neuronales tradicionales, las CNN aprovechan la estructura espacial de los datos para reducir la cantidad de parámetros y mejorar la eficiencia en tareas como clasificación de imágenes, segmentación y detección de objetos. Las CNN logran esto mediante el uso de capas convolucionales, que aplican filtros o kernels sobre las entradas para extraer características relevantes [30].

El proceso general en una CNN incluye capas convolucionales, capas de pooling, capas de activación y capas completamente conectadas. Además, se utilizan optimizadores, funciones de pérdida, y técnicas de regularización para ajustar el modelo durante el entrenamiento.

Proceso de Entrenamiento de Redes Neuronales Convolucionales

El entrenamiento de una CNN sigue el ciclo de entrenamiento estándar de las redes neuronales profundas, que consiste en los siguientes pasos:

Propagación hacia adelante (Forward Pass): Los datos de entrada (una imagen, por ejemplo) son pasados a través de las capas convolucionales, que aplican filtros de tamaño fijo (como 3x3) para generar mapas de características que resaltan patrones importantes en los datos, como bordes, texturas y formas.

Funciones de Activación: Después de aplicar cada filtro, las CNN emplean funciones de activación para introducir no linealidad en el modelo. Las funciones de activación más comunes son:

1. **ReLU (Rectified Linear Unit):** Convierte todos los valores negativos a cero, manteniendo los positivos. Es la activación más usada en las capas convolucionales.

2. **Sigmoid:** Escala los valores a un rango entre 0 y 1, útil para tareas de clasificación binaria.

3. **Tanh:** Similar a la sigmoid, pero escala los valores entre -1 y 1, mejorando la centración de los datos en torno a cero [27].

Capas de Pooling: Estas capas reducen la dimensionalidad de los mapas de características generados, manteniendo la información más relevante. El tipo más común es el Max-Pooling, que selecciona el valor máximo de una región (por ejemplo, 2x2) de los mapas de características, reduciendo el tamaño, pero reteniendo la información importante.

Flatten: Luego de varias capas convolucionales y de pooling, se aplica una capa flatten, que convierte el mapa de características en un vector unidimensional. Este vector se utiliza como entrada para las capas totalmente conectadas o densas.

Capas Completamente Conectadas (Fully Connected Layers): Estas capas toman el vector "aplanado" y lo pasan a través de una red neuronal tradicional, donde cada neurona está conectada con todas las neuronas de la capa anterior.

Dropout: Una técnica común de regularización es dropout, que apaga (es decir, ignora) aleatoriamente un porcentaje de las neuronas durante cada iteración del entrenamiento. Esto evita que el modelo se sobreajuste y mejora su capacidad de generalización.

Optimización del Modelo

Durante el entrenamiento, el objetivo es minimizar una función de pérdida para mejorar el rendimiento del modelo. Esto se logra ajustando los pesos de la red mediante algoritmos de optimización.

Optimización: Los optimizadores actualizan los pesos de la red neuronal en función del gradiente de la función de pérdida:

1. **Gradiente Descendente Estocástico (SGD):** Actualiza los pesos usando un solo ejemplo de entrenamiento en cada iteración. Es rápido, pero puede ser inestable.

2. **Adam (Adaptive Moment Estimation):** Es una versión mejorada de SGD que adapta la tasa de aprendizaje para cada parámetro usando los momentos del gradiente. Es uno de los optimizadores más populares por su eficiencia [31].

3. **RMSProp:** Modifica la tasa de aprendizaje dividiendo el gradiente por una media móvil de gradientes recientes. Es útil en problemas donde los gradientes varían mucho [32].

Funciones de Pérdida: Las funciones de pérdida miden qué tan lejos están las predicciones del modelo de los valores reales. Algunas de las más comunes son:

1. **Cross-Entropy (Entropía Cruzada):** Utilizada principalmente en clasificación, mide la distancia entre dos distribuciones de probabilidad.

2. **Mean Squared Error (MSE):** Común en tareas de regresión, calcula el promedio de los errores al cuadrado entre los valores reales y los predichos [30].

3.1.11. Arquitectura de redes neuronales convolucionales

Las redes neuronales convolucionales (CNN) han demostrado ser una herramienta poderosa en el procesamiento y análisis de imágenes, especialmente con el advenimiento de arquitecturas preentrenadas. Una de las bases fundamentales de las CNN modernas es la disponibilidad de conjuntos de datos como ImageNet, que contiene millones de imágenes etiquetadas en miles de clases. Este conjunto de datos ha permitido entrenar arquitecturas profundas que pueden generalizar de manera eficiente a nuevas tareas visuales con un proceso conocido como transferencia de aprendizaje. En este proceso, las capas profundas de una red se congelan, es decir, no se actualizan sus pesos, y solo se ajustan las capas superficiales a la tarea específica que se quiere resolver. Este enfoque es común en la práctica, ya que permite reutilizar representaciones complejas aprendidas por las capas profundas mientras se adapta el modelo a un nuevo conjunto de clases o tareas específicas con menos datos etiquetados [33].

Entre las arquitecturas más populares que han sido preentrenadas en ImageNet se encuentran VGG16, Xception, ResNet18 y MobileNet. A continuación, se describen sus características más relevantes.

Arquitectura VGG16

La arquitectura VGG16, desarrollada por el Visual Geometry Group (VGG), se caracteriza por su simplicidad y profundidad. Se estructura en capas convolucionales pequeñas (3x3) que permiten captar detalles finos de las imágenes sin incrementar excesivamente la cantidad de parámetros. Las capas convolucionales de VGG16 están seguidas por capas de max-pooling, que reducen la dimensionalidad manteniendo la información más relevante.

Esta arquitectura tiene las siguientes características:

- 13 capas convolucionales organizadas en bloques de 2 o 3 capas consecutivas, todas con filtros de tamaño 3x3 y con pasos de convolución de 1.

- 5 capas de max-pooling, que aplican ventanas de tamaño 2x2 con un stride de 2, reduciendo la resolución de las características a la mitad en cada aplicación.

- 3 capas completamente conectadas (fully connected), donde las dos primeras tienen 4096 neuronas y la última está conectada a la capa de salida.

- Utiliza una capa Softmax como activador en la última capa para realizar la clasificación final entre las 1000 clases del conjunto de datos ImageNet.

VGG16 es utilizada principalmente en tareas de clasificación de imágenes, donde destaca por su precisión y simplicidad, a pesar de tener un número elevado de parámetros en comparación con arquitecturas más recientes [33].

Arquitectura Xception

La arquitectura Xception fue propuesta como una mejora de la arquitectura Inception, utilizando convoluciones separables en profundidad (depthwise separable convolutions). Este tipo de convolución permite descomponer el proceso en dos pasos: primero, aplicar convoluciones espaciales independientes en cada canal, y luego, realizar una convolución puntual (1x1) para combinar los resultados. Esto reduce drásticamente la cantidad de parámetros sin sacrificar el rendimiento, lo que permite mejorar la eficiencia computacional.

Xception cuenta con:

- 36 capas convolucionales, organizadas en bloques modulares de convoluciones separables en profundidad.

- No utiliza capas completamente conectadas. En su lugar, aplica una global average pooling antes de la capa de salida, reduciendo el riesgo de sobreajuste y mejorando la eficiencia.

- Convoluciones de tamaño 3x3 en las capas espaciales y convoluciones 1x1 en las capas puntuales.

Esta arquitectura ha sido preentrenada en ImageNet y es particularmente útil para tareas de clasificación y detección de objetos, mostrando un rendimiento comparable a arquitecturas más complejas, pero con menor costo computacional [34].

Arquitectura ResNet18

La arquitectura ResNet18 es parte de la familia de redes residuales (ResNet), conocidas por su capacidad para entrenar redes muy profundas mediante el uso de bloques residuales. Estos bloques introducen conexiones de identidad que permiten que las entradas de una capa se salten varias capas y se agreguen directamente a la salida de un bloque posterior. Este enfoque resuelve el problema del desvanecimiento del gradiente que suele afectar a redes muy profundas, lo que facilita su entrenamiento.

ResNet18 tiene:

- 18 capas en total, distribuidas en bloques residuales que permiten el paso de la información a través de varias capas.
- Convoluciones de 3x3 y filtros en cada bloque que empiezan en 64 y se duplican a medida que la red profundiza.
- 4 capas de max-pooling en diferentes etapas para reducir las dimensiones de las características.
- Utiliza un global average pooling antes de la capa de salida.

ResNet18 es una de las versiones más simples de la familia ResNet, pero su eficiencia y robustez la hacen ideal para tareas de clasificación con arquitecturas menos complejas [31].

Arquitectura MobileNet

MobileNet es una arquitectura diseñada para ser eficiente en términos computacionales y de memoria, lo que la hace ideal para dispositivos móviles y aplicaciones embebidas. Al igual que Xception, MobileNet utiliza convoluciones separables en profundidad, lo que permite una reducción significativa en la cantidad de operaciones necesarias para procesar imágenes.

Características principales de MobileNet:

- Convoluciones separables en profundidad en todas sus capas, lo que reduce la carga computacional en un factor de hasta 9 veces en comparación con las convoluciones estándar.

- Parámetros ajustables como width multiplier y resolution multiplier, que permiten ajustar el tamaño de la red según las necesidades de precisión y eficiencia.

- Preentrenada en ImageNet, MobileNet ofrece un balance entre precisión y eficiencia en tareas de clasificación, detección de objetos y segmentación.

MobileNet es ideal para aplicaciones en dispositivos con recursos limitados, como teléfonos inteligentes, donde se requiere un rendimiento en tiempo real sin sacrificar mucha precisión [35].

3.1.12. Librerías utilizadas durante el desarrollo del proyecto

El desarrollo de proyectos en aprendizaje automático y profundo implica el uso de diversas bibliotecas que facilitan la implementación de modelos complejos y permiten su integración en aplicaciones del mundo real. Las bibliotecas de aprendizaje profundo, como TensorFlow, Keras y PyTorch, son herramientas fundamentales para diseñar, entrenar y desplegar redes neuronales, mientras que frameworks ligeros como Flask permiten la implementación de estos modelos en entornos web. Estas herramientas no solo aceleran el proceso de desarrollo, sino que también optimizan la ejecución en plataformas con recursos limitados o en aplicaciones industriales.

Durante este proyecto, se han utilizado varias librerías clave, cada una con sus características distintivas que abordan diferentes necesidades, desde la creación de redes neuronales hasta el despliegue de aplicaciones basadas en web. A continuación, se detallan las principales bibliotecas empleadas y su rol en el desarrollo del proyecto.

TensorFlow: Es una biblioteca de aprendizaje automático de código abierto desarrollada por Google, que facilita el diseño, implementación y despliegue de modelos de aprendizaje profundo. Una de sus características más potentes es su capacidad para manejar grafos computacionales (Graphs), donde los nodos representan operaciones matemáticas y las aristas representan tensores de datos (arrays multidimensionales).

TensorFlow permite la ejecución eficiente en diferentes plataformas, como CPUs, GPUs y TPUs, optimizando el rendimiento. Otra ventaja importante es que TensorFlow soporta tanto el entrenamiento en lotes grandes como la inferencia en tiempo real, lo que lo hace ideal para aplicaciones industriales y móviles [36].

Keras: Es una API de alto nivel construida sobre TensorFlow que proporciona una interfaz simplificada para el desarrollo de redes neuronales. Está diseñada para hacer que el trabajo con

deep learning sea accesible y rápido. Keras permite construir modelos de forma secuencial o utilizando un grafo computacional mediante su API funcional, lo que facilita la creación de arquitecturas complejas como redes recurrentes y convolucionales.

Su principal ventaja es la facilidad de uso, ya que abstrae muchos de los detalles técnicos de TensorFlow, lo que la hace ideal para prototipos rápidos. Además, Keras tiene una gran cantidad de módulos predefinidos, como capas convolucionales, de activación, de pooling, y otras esenciales para el diseño de redes neuronales profundas [37].

PyTorch: Es una biblioteca de aprendizaje automático desarrollada por Facebook, especialmente popular en la comunidad académica. PyTorch utiliza grafos computacionales dinámicos (Dynamic Computational Graphs), lo que significa que las redes pueden modificarse durante la ejecución, lo que lo hace muy flexible para investigación y desarrollo de nuevos algoritmos. Además, su integración nativa con Python permite una fácil depuración, y su compatibilidad con GPUs facilita el entrenamiento en grandes volúmenes de datos.

Una característica clave es la capacidad de trabajar directamente con tensores y operaciones matemáticas, lo que proporciona a los desarrolladores un control más granular sobre los modelos [38].

Flask: Es un microframework web ligero para Python que permite crear aplicaciones web con mínima configuración. Es muy utilizado para crear APIs que interactúan con modelos de aprendizaje automático, facilitando la implementación de modelos en producción. Flask es ideal para aplicaciones pequeñas y medianas que no necesitan las características avanzadas de frameworks más grandes como Django.

Es muy sencillo integrar Flask con bibliotecas de aprendizaje profundo como TensorFlow o PyTorch, lo que lo convierte en una herramienta popular para desarrollar aplicaciones web para machine learning y deep learning [39].

3.2. ANTECEDENTES

El mapeo y clasificación de áreas urbanas a partir de imágenes de satélite es un campo de investigación activo. Los métodos tradicionales de aprendizaje automático como Random Forest, SVM se han utilizado comúnmente para la clasificación de usos del suelo, pero presentan limitaciones en la identificación precisa de ciertos tipos de objetos [40].

Los métodos de aprendizaje profundo (deep learning) han emergido recientemente como potentes herramientas para el análisis y clasificación de imágenes de percepción remota [41]. Comparados con enfoques estándar de aprendizaje automático, los modelos de deep learning como las redes neuronales convolucionales (CNNs) han demostrado obtener mayor precisión en tareas complejas de segmentación semántica y detección de objetos.

Hemos revisado varios artículos e investigaciones relacionadas con la implementación de Deep Learning para la clasificación de imágenes a partir de imágenes satélites y consideramos que los ejemplos descritos a continuación constituyeron una base sobre la cual logramos desarrollar nuestro proyecto.

3.2.1. Revisión sistemática de técnicas de deep learning para la clasificación de imágenes de percepción remota.

Sobre esto, Ying Li y otros presentan, en su artículo titulado “Deep learning for remote sensing image classification: A survey” [42], una revisión sistemática de técnicas de deep learning para la clasificación de imágenes de percepción remota. Inicia con una descripción de modelos como redes neuronales convolucionales (CNN), autoencoders apilados (SAE) y redes de creencia profunda (DBN), que permiten extraer representaciones abstractas directamente desde los datos.

Luego, el artículo se enfoca en dos ejes: clasificación a nivel de píxel de imágenes hiperespectrales, y clasificación de escenas en imágenes aéreas/satelitales ópticas. En relación con el primer eje, se revisan investigaciones recientes sobre uso de CNNs, SAEs y DBNs para extraer características espectrales, espaciales y espectral-espaciales en forma supervisada y no supervisada. Se discuten experimentos donde los métodos basados en información espectral-espacial conjunta suelen tener mejor precisión. Respecto a clasificación de escenas, el artículo describe trabajos que usan CNNs pre-entrenadas en otros conjuntos de datos, CNNs que se entrenan desde cero en los nuevos datos, y CNNs donde se hace fine-tuning. También cubre métodos no supervisados como autoencoders apilados. Los experimentos revelan que las CNN pre-entrenadas superan ampliamente a técnicas tradicionales con características manuales.

En términos de aportes a nuestra propuesta, este artículo nos permitió contextualizar el problema de clasificación de imágenes dentro del campo de deep learning aplicado a percepción remota. Revisa avances recientes en detección de áreas verdes que pueden servir como punto de partida. Además, provee conjuntos de datos y métricas de evaluación de modelo que le servirán como referencia.

Sin embargo, el proyecto que estamos desarrollando tiene un enfoque más aplicado y específico en detección de canchas deportivas y parques en entornos urbanos usando imágenes satelitales. La investigación de Ying Li y su equipo, tiene un alcance más amplio, analizando también sensores hiperespectrales y radar. Además, nuestra investigación buscará desarrollar un modelo de CNN optimizado para esta tarea, mientras la investigación de Li sólo revisa aplicaciones previas de modelos genéricos. Nos ayudó a enmarcar nuestra investigación, pero con un objetivo más ampliado.

3.2.2. Detección basada en aprendizaje profundo de cambios en la cubierta forestal urbana junto con cambios urbanos generales utilizando imágenes satelitales de muy alta resolución.

En nuestra búsqueda, encontramos otro artículo llamado “Deep Learning-Based Detection of Urban Forest Cover Change along with Overall Urban Changes Using Very-High-Resolution Satellite Images” [43], el cual propone una técnica semántica de detección de cambios enfocada en los cambios en la cubierta forestal urbana, junto con otros cambios urbanos. Utiliza dos redes neuronales: Deeplabv3+ para la generación de máscaras binarias de bosques urbanos, y Deeply Supervised Image Fusion Network (DSIFN) para la detección binaria de cambios. Ambas redes se entrenan de forma independiente en conjuntos de datos públicos. Luego se realiza transfer learning con un conjunto de datos generado a partir de imágenes satelitales bitemporales de muy alta resolución (VHR) de tres ciudades urbanas, adquiridas con diferentes sensores satelitales.

Se plantea una metodología divida en tres partes: generación de máscaras binarias de bosques urbanos mediante Deeplabv3+, generación de máscara binaria de cambios mediante DSIFN y monitoreo de cambios en cubierta forestal. En el primer paso, Esta red neuronal convolucional permite realizar una segmentación semántica precisa a nivel de píxel. Utiliza un codificador basado en ResNet-50 que extrae características de alto nivel, y un decodificador que recupera la resolución espacial mediante upsampling. Además, emplea un módulo de Atrous Spatial Pyramid Pooling (ASPP) para incorporar contexto multi-escala. Las imágenes pre y post cambio se procesan de forma independiente en Deeplabv3+ para obtener máscaras binarias de las áreas forestales urbanas. Luego se aplica un umbral para binarizar los resultados.

En el segundo paso, esta red neuronal implementa un esquema de fusión profunda de imágenes bitemporales para detección de cambios. Consta de dos flujos que extraen características profundas de las imágenes pre y post cambio de forma independiente. Luego, estas características se concatenan para alimentar una red discriminadora de diferencias. Adicionalmente, se incorporan características basadas en diferencia de imágenes. También se aplican módulos de atención espacial y canal para refinar las características. Finalmente, se genera una máscara binaria de cambios mediante upsampling.

Por último, en el monitoreo de cambios en cubierta forestal, Las máscaras binarias generadas previamente se combinan para crear un resultado de cambio semántico enfocado en las áreas forestales. Esto se logra extrayendo las regiones de cambio forestal de las máscaras pre y post cambio usando la máscara de detección de cambios. Luego se concatenan con esta última para el resultado final. De esta forma se pueden detectar disminuciones y aumentos en las áreas forestales urbanas con respecto a las demás transformaciones urbanas. Así se logra un monitoreo conjunto de la cubierta forestal y los cambios globales en las escenas analizadas.

Esta metodología presentada en el artículo fue utilizada como una base valiosa para desarrollar el modelo de red neuronal convolucional (CNN) que buscamos implementar para la identificación de espacios públicos urbanos en imágenes satelitales. Específicamente, la aproximación en tres pasos facilita incorporar tanto la extracción de las áreas de interés a través de segmentación semántica, como su análisis multitemporal para evaluar cambios mediante detección de diferencias en las características extraídas. La arquitectura Deeplab v3+ demostró precisión en la detección de cubiertas forestales urbanas, por lo que una adaptación supervisada de este modelo mediante transfer learning podría ser efectiva para extraer parques y canchas deportivas.

Más allá de Deeplab v3+ y DSIFN, existen arquitecturas recientes de redes neuronales convolucionales que podrían aportar más valor para nuestra propuesta. Por ejemplo, redes como Mask R-CNN han demostrado altas precisiones en tareas combinadas de segmentación semántica y detección de objetos. Otra alternativa es emplear técnicas de Vision Transformers, las cuales aplican mecanismos de atención para modelar relaciones en toda la imagen, logrando una visión más holística de la escena. Esto puede facilitar la identificación conjunta de diferentes elementos urbanos.

Este artículo nos enseñó el potencial de combinar segmentación semántica y detección de cambios mediante deep learning para mapeo urbano. Sobre esa base, investigamos arquitecturas avanzadas de CNN orientadas a optimizar el desempeño en nuestra aplicación específica de clasificación y monitoreo de parques y canchas.

Sin embargo, el artículo tiene un enfoque más amplio en distintos tipos de cambios urbanos, mientras que lo que buscábamos se centraba específicamente en canchas deportivas y parques. Además, nuestro objetivo es desarrollar un modelo CNN optimizado para esta tarea, mientras que el artículo utiliza modelos genéricos. Finalmente, el artículo trabaja con distintos sensores, y nosotros queremos enfocarnos en imágenes ópticas para un análisis más detallado.

3.2.3. Deep Learning aplicado a imágenes satelitales como herramienta de detección de viviendas sin servicio de energía en el caserío Media Luna – Uribia – Guajira

Finalmente, encontramos un trabajo de investigación titulado “Deep Learning aplicado a imágenes satelitales como herramienta de detección de viviendas sin servicio de energía en el caserío Media Luna – Uribia – Guajira” [44]. Esta investigación aborda la problemática de estimar la cantidad de viviendas sin servicio en zonas rurales a partir de imágenes satelitales, debido a que el acceso y censo en estas áreas resulta complejo. Para automatizar esta tarea, se propone crear un modelo de red neuronal convolucional (CNN) capaz de detectar techos en dichas imágenes.

El objetivo de la investigación es desarrollar un modelo de red neuronal convolucional (CNN) capaz de detectar automáticamente techos de viviendas sin servicio en imágenes satelitales de zonas rurales, para estimar la cantidad de hogares en esas áreas. Se comparan dos arquitecturas: una CNN personalizada de 6 capas, y la red VGG16 pre-entrenada en millones de imágenes. El conjunto de datos contiene 2000 imágenes de 60x60 píxeles con techos y lotes vacíos. Se dividió en 75% entrenamiento y 25% validación para optimizar los modelos.

Se realizó una etapa de preprocesamiento, donde se reescalaron los píxeles de 0-255 a 0-1 para acotar la dispersión. También se aplicó zoom, rotación y volteos aleatorios para hacer al modelo más robusto frente a variaciones. Las imágenes se agruparon en batches de 32 muestras para entrenar por lotes. La CNN aplica dos capas convolucionales apiladas extrayendo 32 y 64 filtros respectivamente, los cuales detectan bordes y texturas. Se intercalan capas de Max Pooling que reducen la dimensión manteniendo características distintivas. Luego se aplana la salida y se

conecta a dos capas totalmente conectadas de 256 y 2 neuronas, esta última con función softmax para estimar probabilidades de cada clase. En total hay 60 mil parámetros entrenables en esta CNN compacta.

Se entrenó durante 8 épocas con 100 pasos por época, probando optimizadores Adam, Adamax y Nadam. Adamax logró la mayor precisión de validación: 99.79%, indicando un modelo bien entrenado. VGG16 consiste en 5 bloques apilados de CNN con 16 capas, aplicando múltiples convoluciones seguidas de Max Pooling para comprimir la representación en cada etapa. Pero dado el pequeño tamaño de las imágenes de entrada, la dimensionalidad se reduce muy rápido, por lo que luego de 4 épocas su precisión sólo alcanzó 60%. Para la detección en nuevas imágenes, éstas se dividieron en parches de 60x60 píxeles que se clasificaron por separado. Desplazando la cuadrícula de parches se hicieron 25 pasadas en total. El modelo CNN detectó entre 37 y 41 viviendas, muy cercano a los 42 hogares contados manualmente.

En términos de aportes, la investigación evidencia la efectividad de CNNs poco profundas para extraer características distintivas de objetos simples como techos de vivienda en imágenes de baja resolución. Además, destaca la importancia de tunear apropiadamente los hiperparámetros según la complejidad de las imágenes.

Sin embargo, esta investigación tiene un enfoque más específico en la detección de techos. Como ya lo hemos mencionado antes, nuestra investigación apunta a identificar parques y canchas deportivas, para lo cual se requirió plantear una CNN optimizada para estas nuevas clases. Aun así, consideramos que este trabajo se acerca bastante a lo que queremos implementar y fue utilizada a modo de guía útil para nuestra investigación aplicada a la detección de áreas recreativas. Podríamos considerarlo como punto de partida para aplicar transfer learning y desarrollar un modelo especializado en detectar canchas deportivas y parques a partir de imágenes satelitales para mapeo urbano.

4. IMPLEMENTACIÓN DE REDES NEURONALES PARA LA CLASIFICACIÓN DE ESPACIOS RECREATIVOS

En este capítulo se profundiza en el primer objetivo específico del proyecto, que consiste en el desarrollo de un modelo de redes neuronales convolucionales (CNN) utilizando técnicas avanzadas de deep learning para la clasificación precisa de áreas recreativas en imágenes satelitales urbanas. Este capítulo no solo describe la implementación de estas arquitecturas, sino que también explora los diferentes conjuntos de datos empleados, el preprocesamiento necesario para mejorar el rendimiento del modelo, y los resultados obtenidos en términos de precisión y generalización.

4.1. Descripción Detallada de las Bases de Datos Utilizadas

En este proyecto, se utilizaron múltiples conjuntos de datos de imágenes satelitales y áreas recreativas, orientados a la clasificación de imágenes mediante redes neuronales convolucionales (CNNs). Entre estos conjuntos, se emplearon bases de datos ampliamente reconocidas en la comunidad de investigación como WHU-RS19, UCMerced LandUse, PatternNET, Optimal_31, y MLRSNet. Además, se generó un conjunto de datos propio y, como nuevo desarrollo, se creó una base de datos Mixta que combina imágenes seleccionadas aleatoriamente de diversas fuentes. Los conjuntos de datos WHU-RS19 y UCMerced LandUse fueron fusionados para la etapa de predicción. A continuación, se presenta una descripción detallada de cada conjunto de datos.

1. WHU-RS19

El conjunto de datos WHU-RS19 es una colección de imágenes satelitales de alta resolución, destinada a la clasificación de diferentes tipos de instalaciones y áreas recreativas. Las imágenes fueron capturadas desde una vista aérea, lo que lo convierte en un recurso valioso para tareas de clasificación geográfica y análisis del uso del suelo. El dataset fue creado por el grupo de investigación de la Universidad de Wuhan y se ha utilizado en varios estudios de reconocimiento de patrones geoespaciales.

- **Clases**: Football Field (50 imágenes), Park (50 imágenes), Pond (54 imágenes)
- **Número de imágenes**: 154
- **Dimensión**: Todas las imágenes tienen una dimensión de 600x600 píxeles.
- **Enlace**: WHU-RS19 Dataset
- **Uso en la literatura**: WHU-RS19 ha sido ampliamente utilizado en investigaciones relacionadas con la clasificación de imágenes aéreas, como se demuestra en estudios donde se evalúa la efectividad de diferentes algoritmos de aprendizaje profundo para el reconocimiento de patrones espaciales [45].

***Figura 2**. Ejemplo imágenes satelitales WHU-RS19. Fuente: [45]*

2. UCMerced LandUse

El conjunto de datos UCMerced LandUse es uno de los más conocidos para el análisis de uso del suelo en imágenes satelitales de alta resolución. Fue desarrollado por la Universidad de California, Merced, y está compuesto por imágenes de diferentes tipos de paisajes, lo que lo convierte en un recurso versátil para la clasificación de imágenes en contextos urbanos y rurales.

- **Clases**: Baseball Diamond (100 imágenes), Golf Course (92 imágenes), Tennis Court (100 imágenes)
- **Número de imágenes**: 292
- **Dimensión**: Las imágenes presentan ligeras variaciones en sus dimensiones, siendo aproximadamente de 256x256 píxeles.
- **Enlace**: UCMerced LandUse Dataset
- **Uso en la literatura**: Este dataset ha sido utilizado en una amplia gama de estudios que abordan problemas de clasificación de imágenes satelitales, particularmente en áreas urbanas [46]. Su aplicación en la clasificación de terrenos deportivos ha sido prominente en el desarrollo de arquitecturas de redes neuronales convolucionales.

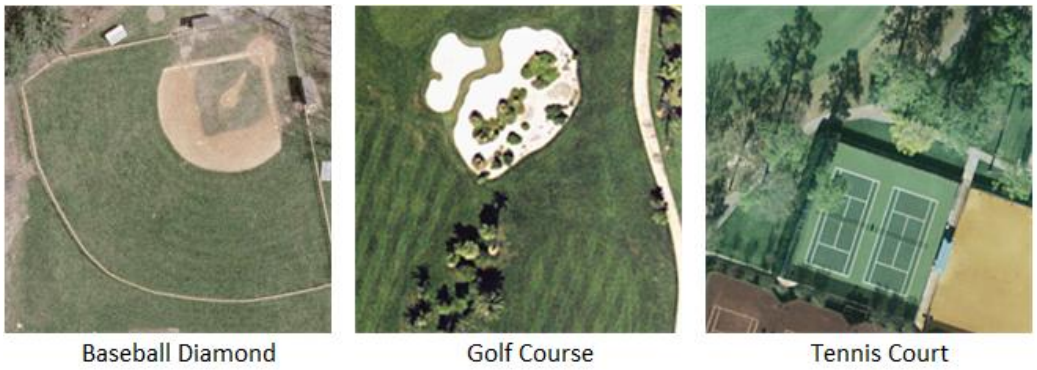


***Figura 3**. Ejemplo imágenes satelitales UCMerced LandUse . Fuente: [46]*

3. PatternNET

PatternNET es un extenso conjunto de datos que se centra en la clasificación de instalaciones deportivas desde imágenes satelitales. Este dataset es ampliamente utilizado en tareas de reconocimiento de patrones, siendo uno de los más grandes en su categoría.

- **Clases**: Baseball Field (800 imágenes), Basketball Court (800 imágenes), Football Field (800 imágenes), Golf Course (800 imágenes), Tennis Court (800 imágenes)
- **Número de imágenes**: 4000
- **Dimensión**: Todas las imágenes tienen una dimensión de 256x256 píxeles.
- **Enlace**: PatternNET Dataset
- **Uso en la literatura**: PatternNET ha sido utilizado en múltiples investigaciones enfocadas en la clasificación automática de instalaciones deportivas mediante CNNs, incluyendo estudios sobre la mejora en la eficiencia de clasificación usando aprendizaje profundo [47].

***Figura 4**. Ejemplo Imágenes Satelitales PatternNET. Fuente: [47]*

4. Optimal_31

El conjunto de datos Optimal_31 se enfoca en la clasificación de diferentes tipos de instalaciones deportivas y recreativas. Está compuesto por un número relativamente pequeño de imágenes, lo que lo hace adecuado para experimentos que buscan entrenar modelos en conjuntos de datos limitados.

- **Clases**: Baseball Diamond (60 imágenes), Basketball Court (60 imágenes), Golf Course (60 imágenes), Track Field (60 imágenes)
- **Número de imágenes**: 240
- **Dimensión**: Todas las imágenes tienen una dimensión de 256x256 píxeles.
- **Enlace**: Optimal_31 Dataset
- **Uso en la literatura**: Este dataset ha sido empleado en investigaciones centradas en la clasificación de instalaciones deportivas, especialmente en estudios donde se evalúa el rendimiento de redes neuronales con datos limitados [48].

***Figura 5**. Ejemplo Imágenes satelitales Optimal_31. Fuente: [48]*

5. MLRSNet

MLRSNet es uno de los conjuntos de datos más grandes utilizados en este proyecto. Contiene imágenes de múltiples clases de instalaciones recreativas, capturadas desde una perspectiva aérea. Es importante destacar que al momento de realizar el entrenamiento se utilizaron únicamente 800 imágenes de cada clase, esto debido a que para procesar tantas imágenes se requería un costo computacional representativo.

- **Clases**: Baseball Field (2002 imágenes), Basketball Court (2895 imágenes), Park (1682 imágenes), Stadium (2462 imágenes), Golf Course (2515 imágenes), Track Field (2500 imágenes), Tennis Court (2500 imágenes)
- **Número de imágenes**: 16,556
- **Dimensión**: Todas las imágenes tienen una dimensión de 256x256 píxeles.
- **Enlace**: MLRSNet Dataset
- **Uso en la literatura**: MLRSNet ha sido utilizado en una variedad de investigaciones centradas en la clasificación de grandes conjuntos de datos de imágenes aéreas. Su uso ha sido reportado en estudios de reconocimiento geoespacial y clasificación de áreas deportivas en imágenes de alta resolución [49].

***Figura 6**. Ejemplo Imágenes satelitales MLRSNet. Fuente: [49]*

6. Base de datos Mixta

El conjunto de datos Mixto fue diseñado específicamente para mejorar la capacidad de generalización del modelo, combinando imágenes de distintas bases con diferentes niveles de resolución y calidad visual. La diversidad en las fuentes de datos permite que el modelo enfrente un entorno de entrenamiento más variado y desarrolle una mayor robustez en la clasificación.

- **Clases**: Baseball Diamond, Basket, Campo de Fútbol, Cancha Múltiple, Golf Course, Parque, Tenis
- **Número de imágenes**:
    - Baseball Diamond: 300 imágenes (MLRSNet)
    - Basket: 150 imágenes (base propia) + 150 imágenes (PatternNet)
    - Campo de Fútbol: 150 imágenes (base propia) + 150 imágenes (PatternNet)
    - Cancha Múltiple: 328 imágenes (base propia)
    - Golf Course: 300 imágenes (MLRSNet)
    - Parque: 150 imágenes (base propia) + 150 imágenes (MLRSNet)
    - Tenis: 150 imágenes (base propia) + 150 imágenes (PatternNet)
- **Enlace**: Mixta_Propia
- **Dimensión**: Varía según la fuente, con imágenes de dimensiones que van desde 256x256 píxeles hasta 4080x4080 píxeles, lo que añade un desafío adicional al proceso de clasificación.

Este conjunto de datos se creó con el fin de mezclar imágenes que provienen de bases de datos con diferentes resoluciones, niveles de ruido, y estilos visuales. Esto permite que el modelo entrenado con la base Mixta desarrolle una mayor capacidad de generalización, es decir, que sea capaz de identificar y clasificar correctamente las diferentes clases en escenarios más variados.

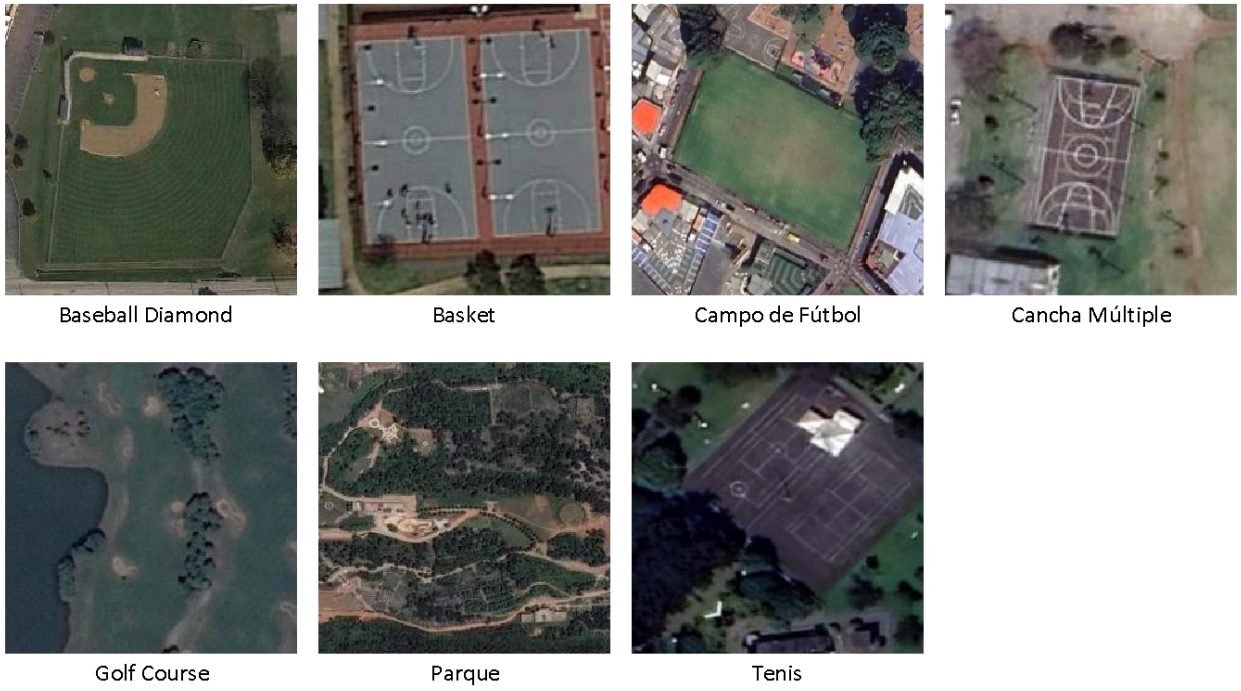


Figura 7. *Ejemplo Imágenes satelitales Base de datos Mixta. Fuente: Elaboración propia*

7. Dataset Propio

Además de los datasets disponibles públicamente, se generó un dataset propio como parte del proyecto. Este conjunto de datos contiene imágenes de diversas instalaciones deportivas en varias resoluciones. Sobre esta base de datos se llevó a cabo una investigación con el objetivo de desarrollar un método de etiquetado de imágenes que contribuyera al proyecto en curso. No obstante, se identificó la ausencia de conjuntos de datos abiertos que contuvieran imágenes de espacios urbanos recreativos en el territorio nacional colombiano. Ante esta limitación, se optó por explorar herramientas que facilitaran la construcción de un conjunto de datos propio, el cual no era un objetivo específico dentro de nuestro proyecto, pero surgió la necesidad de crearlo por las características particulares de los escenarios deportivos urbanos en Colombia.

Inicialmente, se evaluó el uso del proyecto Iris, que emplea técnicas de aprendizaje profundo para el etiquetado de imágenes. Sin embargo, su enfoque basado en la segmentación resultó inadecuado para nuestros fines. Posteriormente, se exploraron otros proyectos, algunos de los cuales no estaban disponibles de manera pública o no cumplían con los requisitos del estudio.

Como alternativa, se adoptó un enfoque operativo mediante la obtención de imágenes ráster utilizando el software QGIS. Este programa permitió realizar recortes precisos y ajustados a las necesidades del proyecto, a partir de imágenes capturadas por diversas tecnologías satelitales, como Google Maps o ESRI. Se dedicó una parte significativa del proyecto a esta tarea con el fin de generar un conjunto de datos más representativo y alineado con los requerimientos del estudio, centrado en imágenes de escenarios deportivos y recreativos implementados en ciudades colombianas.

De una zona ubicada en alguna ciudad del territorio colombiano, se seleccionaban los espacios deportivos disponibles y se capturaban las imágenes para conformar el dataset, cada una de las imágenes representa una capa del proyecto en QGIS y así se podía exportar cada una de ellas a un archivo en formato '.tif'. Para obtener estas imágenes directamente de QGIS, se realizó un proceso de ajuste en este software.

En la configuración de QGIS para la obtención de imágenes satelitales, se instaló el plugin 'Tile+', el cual provee acceso rápido a basemaps populares y accesibles al público, tales como Google y Bing. Para el desarrollo del dataset propio, se configuro el plugin para utilizar las imágenes de mapa de 'Google Satellite', estos son los datos técnicos:

- Name: Google Satellite
- URL: https://mt1.google.com/vt/lyrs=s&x={x}&y={y}&z={z}
- Source: url=https://mt1.google.com/vt/lyrs%3Ds%26x%3D%7Bx%7D%26y%3D%7By%7D%26z%3D%7Bz%7D&zmax=19&zmin=0&type=xyz
- Provider: wms
- Coordinate Reference System (CRS): EPSG:3857 - WGS 84 / Pseudo-Mercator

Se utilizó la herramienta 'Raster Tools' de QGIS para convertir la zona del mapa seleccionada que contiene el espacio deportivo a una imagen de tipo raster, con un tamaño de cuadro de 255 y 0,015 de unidades por píxel para garantizar una baja perdida en la resolución de la imagen. Una vez generada la imagen como un nuevo layer sobre el mapa, esta se exportaba a un archivo de tipo GeoTIFF con una dimensión de 255 x 255 pixeles. Estas imágenes GeoTIFF exportadas de QGIS son las que conformaron el conjunto de datos propio.

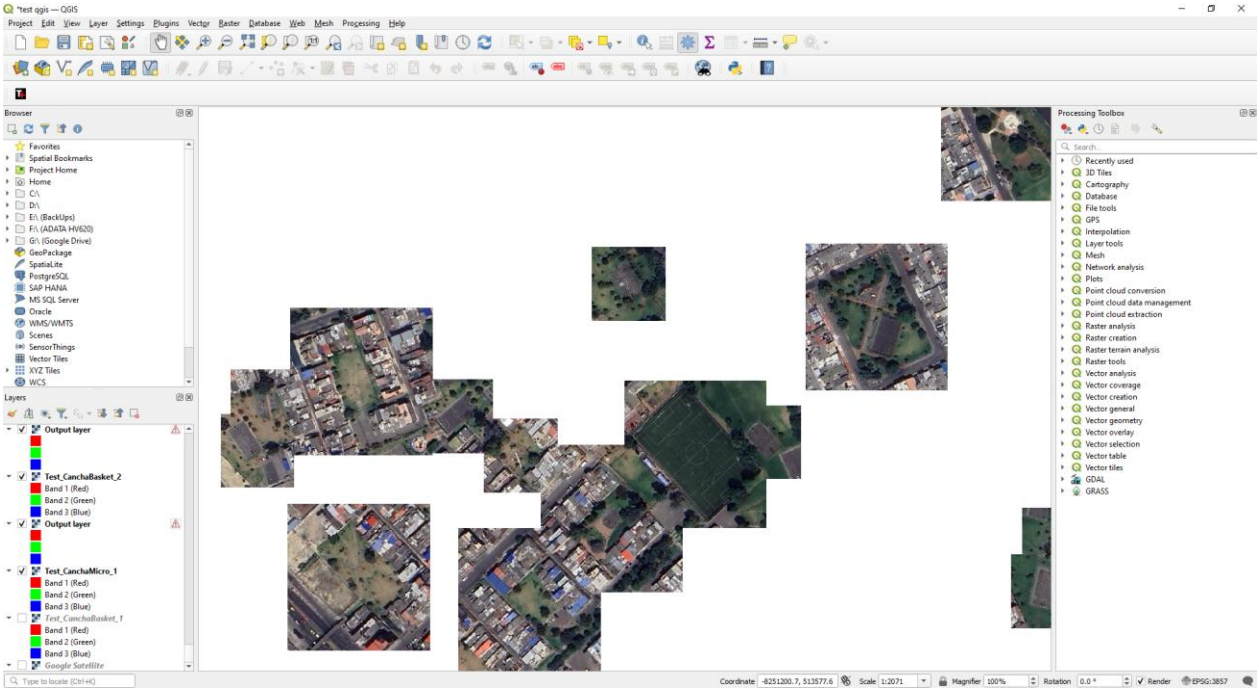


Figura 8. *Captura de pantalla de imágenes satelitales recortadas en QGIS que contienen espacios recreativos urbanos en la ciudad de Bogotá. Fuente: Elaboración propia*

Se decidió construir un conjunto de datos compuesto por cinco clases de imágenes satelitales obtenidas desde QGIS siguiendo el proceso descrito anteriormente, correspondientes a los siguientes tipos de escenarios deportivos comúnmente presentes en el territorio nacional:

- Cancha de baloncesto
- Campo de fútbol
- Cancha múltiple
- Parque barrial o zonal
- Cancha de Tenis

Cada una de estas clases fue conformada con al menos 50 imágenes originales obtenidas a través del software QGIS. Además, se empleó la herramienta Image Data Generator de Keras para aumentar el número de imágenes por clase, con el fin de disponer de un volumen adecuado para los procesos de entrenamiento y validación del modelo.

Es importante señalar que la calidad de las imágenes satelitales del territorio colombiano es considerablemente inferior en comparación con los conjuntos de datos públicos encontrados durante la fase de diseño del modelo, los cuales estaban basados en imágenes satelitales del territorio europeo. No obstante, se consideró pertinente emplear este tipo de imágenes de baja resolución, dado que el objetivo es que el modelo sea aplicable a imágenes obtenidas a través de plataformas accesibles como Google Maps. Este enfoque asegura que cualquier entidad, mediante capturas de pantalla u otros métodos sencillos, pueda utilizar el modelo para realizar

sus respectivos análisis de espacios urbanos recreativos.

Resumen del dataset Propio

- **Clases**: Basket (358 imágenes), Campo de Fútbol (369 imágenes), Cancha Múltiple (328 imágenes), Parque (328 imágenes), Tenis (322 imágenes)
- **Número de imágenes**: 1705
- **Dimensiones:** Las imágenes de generadas mediante QGIS y generadas mediante Keras Image Generator cuentan con un tamaño de 256 x 256 pixeles.
- **Enlace**: Dataset Propio
- **Uso**: Este conjunto de datos se utilizó para entrenar modelos específicos del proyecto, abordando la clasificación de instalaciones deportivas desde una perspectiva aérea y terrestre.

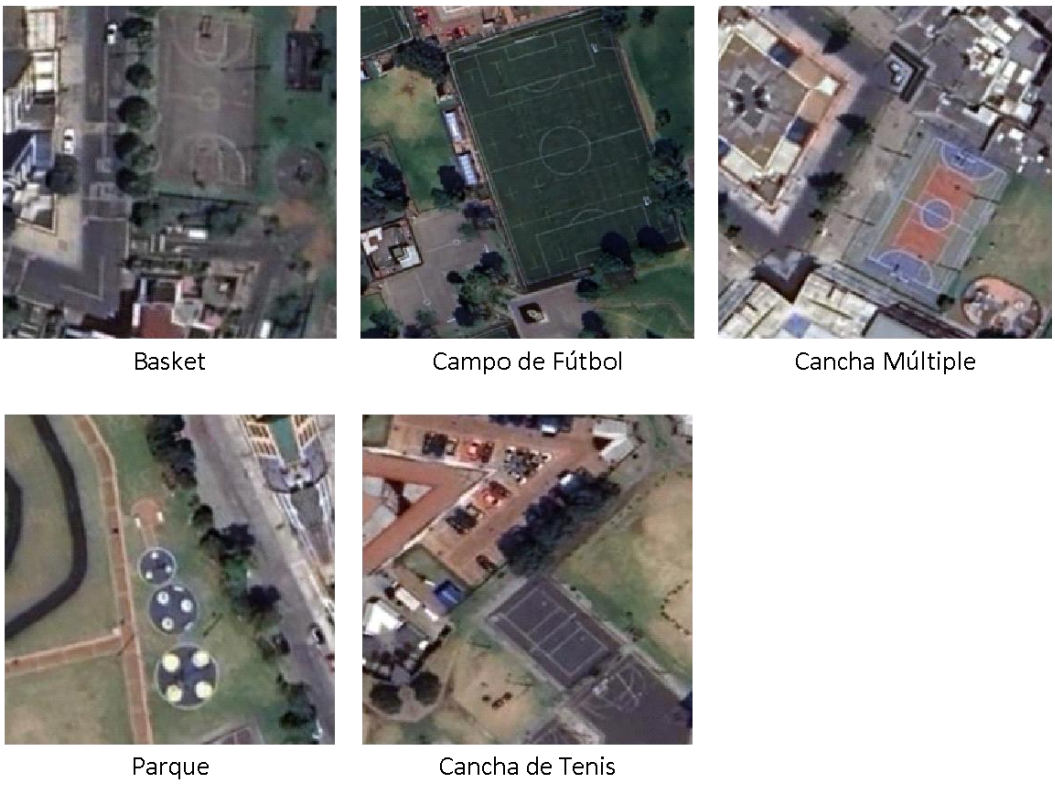


***Figura 9**. Ejemplo Imágenes base de datos Propia. Fuente: Elaboración propia*

Estos conjuntos de datos, especialmente los fusionados WHU-RS19 y UCMerced LandUse, junto con la incorporación de la base Mixta, proporcionan una base sólida para el entrenamiento y evaluación de redes neuronales convolucionales en tareas de clasificación geográfica y de instalaciones deportivas. La diversidad en resolución, clases y número de imágenes permitió desarrollar y probar modelos robustos, capaces de generalizar eficientemente en diversos

escenarios. La inclusión de la base de datos Mixta es un elemento clave que mejora la capacidad de los modelos para adaptarse a imágenes de diferentes calidades, permitiendo que el sistema de clasificación sea más eficaz y preciso cuando se enfrenta a imágenes del mundo real con variaciones significativas en sus características visuales.

4.2. Descripción del Proceso de Entrenamiento de los Modelos de Redes Neuronales Convolucionales

Durante el desarrollo del proyecto, se implementaron cuatro modelos de redes neuronales convolucionales basados en arquitecturas populares, además de un modelo customizado diseñado específicamente para este trabajo. Las arquitecturas empleadas fueron: Custom, VGG16, ResNet18, MobileNet, y Xception. Todas las arquitecturas, excepto el modelo customizado, fueron inicializadas con pesos preentrenados en el dataset ImageNet, permitiendo aprovechar las características aprendidas en este conjunto de datos. A continuación, se detallan los principales pasos de preprocesamiento y los ajustes realizados para el entrenamiento de cada modelo.

1. Preprocesamiento de Datos

El preprocesamiento de los datos fue clave para garantizar una correcta entrada a los modelos. Se implementó un conjunto de transformaciones comunes a todas las arquitecturas, con el objetivo de estandarizar las imágenes para el entrenamiento. Los pasos seguidos fueron los siguientes:

- **Cargar Imágenes**: Todas las imágenes fueron leídas en formato RGB para asegurar la consistencia en los modelos preentrenados.

- **Redimensionamiento**: Se redimensionaron todas las imágenes a 256x256 píxeles, lo que permitió una entrada consistente en los modelos preentrenados con ImageNet, ya que este tamaño es estándar en muchas arquitecturas.

- **Normalización**: Las imágenes fueron normalizadas dividiendo los valores de los píxeles por 255, para que todos los valores estuvieran entre 0 y 1. Además, se aplicó una normalización adicional restando la media, basada en los valores estándar de ImageNet, lo que ayudó a mejorar la estabilidad durante el entrenamiento.

- **División del Dataset**: Los datos fueron divididos en conjuntos de entrenamiento y prueba en una proporción del 80% y 20%, respectivamente.

- **Balanceo de Clases**: En aquellos casos donde existía desbalance de clases, se aplicó un sobremuestreo con RandomOverSampler para garantizar una representación equitativa de cada clase.

2. Modelo Customizado

El modelo Custom fue diseñado específicamente para este proyecto. Se trata de una arquitectura secuencial que sigue una estructura tradicional de capas convolucionales y densas:

- **Capas Convolucionales**: Se añadieron dos capas convolucionales con 32 y 64 filtros, respectivamente, y un tamaño de kernel de 3x3. Se utilizó la activación ReLU en ambas capas para asegurar la no linealidad.

- **Capas de Max-Pooling**: Después de cada capa convolucional, se incluyeron capas de Max-Pooling con un tamaño de 2x2 para reducir la dimensionalidad.

- **Capa de Aplanamiento**: Las características extraídas fueron aplanadas en un vector unidimensional utilizando una capa de Flatten.

- **Capas Densas**: Se añadieron capas densas con 64 neuronas y activación ReLU, finalizando con una capa de salida con activación softmax para realizar la clasificación del número de clases presentes en los dataset.

El modelo fue entrenado desde cero utilizando la función de pérdida de entropía cruzada categórica y el optimizador Adam, durante un total de 15 épocas.

3. VGG16 Preentrenado en ImageNet

El modelo VGG16 fue utilizado con los pesos preentrenados en ImageNet y modificado de la siguiente manera:

- **Congelación de Capas Preentrenadas**: Las capas convolucionales del modelo base fueron congeladas para evitar que sus pesos se actualizarán durante el entrenamiento.

- **Capas Superiores Personalizadas**: Se añadieron capas adicionales para adaptar el modelo a la tarea de clasificación. Estas capas incluyen una capa Flatten, una capa densa con 128 neuronas y activación ReLU, una capa de Dropout con una tasa de 0.5, y una capa de salida con activación softmax.

El modelo fue entrenado utilizando el optimizador Adam y la función de pérdida entropía cruzada categórica, con una técnica de early stopping y un programador de tasa de aprendizaje para optimizar el proceso de entrenamiento.

4. ResNet18 Preentrenado en ImageNet

El modelo ResNet18 también fue preentrenado en ImageNet y se implementó de la siguiente manera:

- **Capas Descongeladas**: A diferencia de los otros modelos, las capas base de ResNet18 fueron descongeladas, permitiendo que los pesos se ajustaran durante el entrenamiento. Esta estrategia mejoró la capacidad del modelo para adaptarse a las características específicas de las imágenes satelitales.

- **Modificación de la Capa de Salida**: Se reemplazó la última capa densa del modelo con una nueva capa totalmente conectada con en número de neuronas de cada base de datos y activación softmax.

Al igual que en VGG16, se aplicó early stopping y se utilizó un programador de tasa de aprendizaje para ajustar dinámicamente el entrenamiento.

5. MobileNet Preentrenado en ImageNet

El modelo MobileNet, conocido por su eficiencia en dispositivos con recursos limitados, fue ajustado de la siguiente manera:

- **Capas Descongeladas**: Siguiendo el mismo enfoque que en ResNet18, las capas preentrenadas fueron descongeladas para permitir el ajuste de los pesos en función de las características del nuevo conjunto de datos.

- **Capas Superiores Personalizadas**: Se añadieron capas densas similares a las de VGG16 y ResNet18, incluyendo una capa Flatten, una capa densa con activación ReLU, y una capa de salida con activación softmax.

El modelo fue optimizado para ser eficiente tanto en precisión como en recursos computacionales, haciéndolo adecuado para dispositivos de baja potencia.

6. Xception Preentrenado en ImageNet

El modelo Xception se utilizó con pesos preentrenados en ImageNet y se ajustó de manera similar

a los otros modelos preentrenados:

- **Capas Descongeladas**: Al igual que en ResNet18 y MobileNet, las capas preentrenadas de Xception fueron descongeladas para mejorar la capacidad del modelo de adaptarse a las nuevas imágenes.

- **Capas Superiores Personalizadas**: Se añadieron capas superiores, incluidas una capa Flatten, una capa densa con 128 neuronas y activación ReLU, y una capa de salida con activación softmax para clasificar las diferentes clases.

Este modelo también fue entrenado utilizando early stopping y un programador de tasa de aprendizaje para asegurar la optimización del proceso de entrenamiento.

Teniendo en cuenta la metodología presentada, es importante mencionar que, durante el preprocesamiento de las imágenes en todos los modelos, se optó por un escalado fijo a 256x256 píxeles para garantizar la compatibilidad con los modelos preentrenados utilizados (MobileNet, ResNet18, Xception, entre otros). Sin embargo, esta decisión presentó una limitación significativa, ya que penalizó los datasets con menor resolución espacial, reduciendo su capacidad de detección y la calidad de las características extraídas por los modelos. Este efecto fue particularmente evidente en el dataset propio, donde las imágenes presentaban una resolución considerablemente inferior comparada con otras bases de datos como PatternNET.

4.3. Análisis de Resultados Obtenidos en las Arquitecturas y Datasets

En esta sección se presentan los resultados del rendimiento de las distintas arquitecturas de redes neuronales convolucionales utilizadas para la clasificación de áreas recreativas. Las métricas que se analizan, como accuracy, precision, recall y F1-Score, fueron obtenidas utilizando exclusivamente el conjunto de datos de test, es decir, imágenes que no fueron vistas durante el entrenamiento de los modelos. Esto asegura una evaluación más justa y objetiva del rendimiento de cada arquitectura en términos de su capacidad para generalizar a nuevos datos. Por esta razón, el objetivo es obtener valores de estas métricas lo más cercanos posible al 100%, lo que indicaría un modelo altamente efectivo en la tarea de clasificación. A continuación, se detallan las observaciones y conclusiones derivadas de los resultados obtenidos en las diversas arquitecturas y datasets empleados.

Tabla 1. *Resultados de métricas de evaluación para las arquitecturas de CNN empleadas. Fuente: Elaboración propia*

Dataset	Arquitectura	Pre - entrenamiento	Precision	Recall	F1-Score	Accuracy
WHU-RS19	Custom	FALSO	0.84381	0.83516	0.83675	0.83516
	VGG16	VERDADERO	0.94045	0.93407	0.93467	0.95607

	Resnet18	VERDADERO	1.00000	1.00000	1.00000	1.00000
	MobileNet	VERDADERO	0.99023	0.98901	0.98907	0.98901
	Xception	VERDADERO	0.93588	0.93407	0.93403	0.93407
Optimal_31	Custom	FALSO	0.76724	0.58333	0.55655	0.58333
	VGG16	VERDADERO	0.89554	0.89583	0.89489	0.89583
	Resnet18	VERDADERO	0.89730	0.89583	0.89583	0.89583
	MobileNet	VERDADERO	0.96167	0.95833	0.95833	0.95833
	Xception	VERDADERO	0.90041	0.89583	0.89551	0.89583
PatternNET	Custom	FALSO	0.90298	0.90250	0.90266	0.90250
	VGG16	VERDADERO	0.96901	0.96875	0.96874	0.96875
	Resnet18	VERDADERO	0.99260	0.99250	0.99249	0.99250
	MobileNet	VERDADERO	0.99503	0.99500	0.99500	0.99500
	Xception	VERDADERO	0.98383	0.98375	0.98377	0.98375
Propio	Custom	FALSO	0.68089	0.65982	0.66297	0.65982
	VGG16	VERDADERO	0.79597	0.77419	0.76666	0.77419
	Resnet18	VERDADERO	0.96037	0.95894	0.95889	0.95894
	MobileNet	VERDADERO	0.96091	0.95894	0.95895	0.95894
	Xception	VERDADERO	0.73410	0.72141	0.72537	0.72141
MLRSNet	Custom	FALSO	0.84553	0.84458	0.84405	0.84458
	VGG16	VERDADERO	0.89016	0.88721	0.88784	0.88721
	Resnet18	VERDADERO	0.97875	0.97869	0.97865	0.97869
	MobileNet	VERDADERO	0.98760	0.98757	0.98755	0.98757
	Xception	VERDADERO	0.89904	0.89609	0.89586	0.89609
Mixto	Custom	FALSO	0.77422	0.76956	0.76991	0.76956
	VGG16	VERDADERO	0.85307	0.84507	0.84442	0.84507
	Resnet18	VERDADERO	0.97521	0.97418	0.97411	0.97418
	MobileNet	VERDADERO	0.98864	0.98826	0.98827	0.98826
	Xception	VERDADERO	0.85264	0.84742	0.84896	0.84742

En la Tabla 1, se muestran las métricas de evaluación Accuracy, Precision, Recall y F1-Score para las diferentes arquitecturas de redes neuronales convolucionales: Custom, VGG16, ResNet18, MobileNet y Xception. Estas arquitecturas fueron evaluadas en seis datasets: WHU-RS19, Optimal_31, PatternNET, Propio, MLRSNet y la base Mixta, que combina imágenes de diferentes fuentes. A continuación, se detallan las observaciones y conclusiones derivadas de los resultados obtenidos:

1. Análisis del Dataset WHU-RS19

- ResNet18 preentrenado en ImageNet fue el modelo con mejor desempeño, obteniendo un Accuracy perfecto de 100%, lo que demuestra su capacidad para extraer características complejas y generalizar eficientemente en la clasificación de este dataset, con valores máximos en todas las métricas (Precision, Recall y F1-Score).

- MobileNet también mostró un desempeño sobresaliente con un Accuracy del 98.9%, destacando su eficiencia computacional y capacidad para mantener altos niveles de precisión y recall.

- Xception y VGG16 mostraron un desempeño robusto, con Accuracy de 93.41% y 95.60% respectivamente.

- En contraste, el modelo Custom tuvo un Accuracy significativamente menor (83.52%), lo que resalta la ventaja de los modelos preentrenados sobre los entrenados desde cero en grandes datasets como ImageNet.

2. Análisis del Dataset Optimal_31

- MobileNet fue el modelo con mejores resultados, obteniendo un Accuracy de 95.83%, lo que demuestra su capacidad de generalización en datasets más pequeños como Optimal_31, gracias a su arquitectura ligera y eficiente.

- ResNet18 y VGG16 obtuvieron valores de Accuracy de 89.58%, mostrando efectividad en este dataset.

- El modelo Xception alcanzó un Accuracy de 89.58%, lo que muestra que sigue siendo competitivo, aunque ligeramente inferior a MobileNet.

- El modelo Custom presentó un desempeño significativamente más bajo (58.33% de Accuracy), lo que refuerza la ventaja de los modelos preentrenados para este tipo de tareas en datasets con pocas imágenes.

3. Análisis del Dataset PatternNET

- MobileNet fue nuevamente el mejor modelo con un Accuracy de 99.50%, manejando la complejidad de las imágenes de PatternNET de manera eficiente.

- Xception y ResNet18 también obtuvieron resultados excelentes, con Accuracy de 98.37% y 99.25% respectivamente.

- VGG16 se mantuvo sólido con un Accuracy de 96.87%.

- El modelo Custom mostró una mejora en este dataset, obteniendo un Accuracy de 90.25%, aunque sigue siendo superado por las arquitecturas preentrenadas.

4. Análisis del Dataset Propio

- MobileNet y ResNet18 fueron los modelos con mejor desempeño, obteniendo un Accuracy de 95.89%. Esto indica su capacidad de adaptarse a conjuntos de datos con imágenes de diversas resoluciones.

- VGG16 y Xception obtuvieron Accuracy de 77.41% y 72.14% respectivamente, con una ligera caída en su rendimiento en este conjunto de datos.

- El modelo Custom tuvo el desempeño más bajo, con un Accuracy de 65.98%, lo que confirma la superioridad de los modelos preentrenados.

5. Análisis del Dataset MLRSNet

- En el dataset MLRSNet, MobileNet lideró con un Accuracy de 98.76%, mostrando consistencia en su rendimiento en diferentes datasets.

- VGG16 y ResNet18 también obtuvieron buenos resultados, con Accuracy de 88.72% y 97.87% respectivamente.

- Xception mostró un rendimiento competitivo, con un Accuracy de 89.60%.

- El modelo Custom, como en otros datasets, tuvo un Accuracy de 84.45%, aunque su rendimiento fue más competitivo en este dataset.

6. Análisis del Dataset Mixto

- MobileNet volvió a destacar en la base de datos Mixta, obteniendo un Accuracy de 98.83%, mostrando su capacidad para manejar imágenes de diferentes fuentes y resoluciones con alta generalización.

- ResNet18 y VGG16 obtuvieron Accuracy de 97.41% y 84.50% respectivamente, con un excelente desempeño en este dataset heterogéneo.

- Xception también tuvo un rendimiento aceptable, con un Accuracy de 84.74%, mientras que el modelo Custom tuvo el desempeño más bajo en esta base con 76.95% de Accuracy.

Conclusiones Generales de los Resultados

- **Arquitecturas Preentrenadas vs Custom**: Los resultados muestran consistentemente que las arquitecturas preentrenadas en ImageNet (como VGG16, ResNet18, MobileNet y Xception) superan al modelo Custom en todas las métricas evaluadas, debido a que

los modelos preentrenados ya han capturado características profundas útiles para la clasificación de imágenes.

- **Rendimiento Destacado de MobileNet**: MobileNet se destacó como la arquitectura más consistente y efectiva, obteniendo los mejores resultados en varios datasets, incluida la base de datos Mixta. Su capacidad para lograr altos niveles de precisión y recall, con un costo computacional bajo, lo convierte en una opción ideal para aplicaciones en entornos con recursos limitados.

- **Importancia del Preentrenamiento**: Los resultados destacan la importancia del preentrenamiento en grandes datasets como ImageNet. Arquitecturas como ResNet18, MobileNet y Xception obtuvieron mejores resultados en términos de precisión, recall y F1-Score, demostrando que estas técnicas son más efectivas para tareas complejas de clasificación de imágenes satelitales.

- **Explicación de Métricas Similares en un Modelo y Dataset**: En algunos casos, las métricas de precisión, recall, F1-score y accuracy son iguales o muy cercanas debido a la excelente capacidad de generalización del modelo y a la similitud en la distribución de clases en el dataset. Esto puede indicar un buen balance entre clases y una baja cantidad de errores de clasificación.

4.4. Pruebas de Clasificación con Diferentes Tipos de Imágenes

En esta sección se analiza el desempeño de los cinco modelos evaluados en las pruebas de clasificación, haciendo referencia a los anexos que corresponden a cada uno de los datasets empleados en el proyecto. Los anexos presentan los resultados cualitativos de las pruebas, donde cada imagen se clasifica en función de si el modelo predijo correctamente (Positivo), incorrectamente (Negativo), o si la clase no estaba definida en ese modelo (No Definido). Es importante destacar que las imágenes seleccionadas fueron elegidas estratégicamente para encontrar el modelo que mejor permite generalizar, incluso frente a datos no vistos previamente. Además, los datasets tienen diferentes calidades de imagen, lo que añade un reto adicional, ya que cada uno sirve para clasificar diferentes tipos de imágenes.

Las pruebas se dividieron en cuatro tipos:

- **Prueba 1**: Clasificación de imágenes pertenecientes los mismos datasets con el que se entrenaron los modelos.
- **Prueba 2**: Clasificación de imágenes obtenidas de Google Maps, con el fin de medir la capacidad de generalización.
- **Prueba 3**: Clasificación de imágenes con características atípicas (variaciones en color, textura, entre otros).

- **Prueba 4**: Clasificación de imágenes que contienen múltiples instancias de una misma clase en un mismo escenario.

Los resultados detallados se encuentran en los Anexos A, B, C, D, E y F, los cuales muestran el desempeño de cada modelo en cada una de estas pruebas. A continuación, se presentan las conclusiones obtenidas a partir de los resultados:

Análisis de Resultados por Dataset

1. Dataset Propio

- MobileNet y Xception fueron los modelos con mejor rendimiento en el dataset propio, alcanzando un 73% de aciertos en todas las pruebas. En las pruebas con imágenes del mismo dataset de entrenamiento (Prueba 1), ambos modelos mostraron una mayor capacidad para clasificar correctamente las clases como Campo de Baloncesto y Campo de Fútbol, lo que evidencia su capacidad de generalización en las imágenes conocidas.

- ResNet18, VGG16, y Custom tuvieron un rendimiento más bajo alcanzando un 64% de aciertos cada uno. El modelo Custom mostró el peor desempeño, con una tasa de aciertos de solo 64%, especialmente en clases como Parque y Campo de Fútbol, donde se registraron múltiples clasificaciones incorrectas. Este modelo parece ser menos eficaz cuando se enfrenta a variaciones en las imágenes o escenarios más complejos.

- En general, las pruebas de clasificación en el dataset propio muestran que los modelos preentrenados (MobileNet y Xception) son más robustos y capaces de adaptarse mejor tanto a las imágenes de entrenamiento como a aquellas con características más atípicas (Pruebas 3 y 4).

2. Dataset PatternNET

- MobileNet, ResNet18 fueron los modelos que mostraron los mejores resultados en el dataset PatternNET, alcanzando un 85% de aciertos cada uno. Estos modelos lograron clasificar correctamente casi todas las clases en las pruebas, lo que indica que son muy efectivos para imágenes de alta calidad y resolución estándar, como las presentes en PatternNET.

- Xception y Custom presentaron un rendimiento ligeramente inferior, con un 69% de aciertos cada uno. En particular, Xception mostró un buen desempeño en la mayoría de las clases, pero falló en algunas imágenes con características atípicas, como las variaciones en Campo de Tenis y Campo de Baseball.

- En este dataset, las pruebas con imágenes con características atípicas (Prueba 3) mostraron que MobileNet y VGG16 mantuvieron un rendimiento sobresaliente, mientras que los demás modelos tuvieron dificultades para generalizar en imágenes no vistas.

3. Dataset Optimal_31

- ResNet18 y Xception lideraron con un 69% de aciertos cada uno en el dataset Optimal_31, demostrando su capacidad de generalización en un dataset más pequeño y uniforme. Ambos modelos lograron clasificar correctamente la mayoría de las clases, a pesar de las variaciones en las pruebas.

- VGG16 presentó un desempeño intermedio con un 54% de aciertos, mostrando algunos errores en la clasificación de clases como Campo de Baloncesto y Campo de Fútbol, en particular en imágenes con características atípicas (Pruebas 3 y 4).

- MobileNet tuvo un rendimiento relativamente bajo en este dataset, con un 38% de aciertos, lo que contrasta con su buen desempeño en otros datasets. Esto sugiere que este modelo podría tener dificultades para adaptarse a datasets más pequeños o con menos variabilidad en las imágenes.

- El modelo Custom fue el que presentó el peor rendimiento, con solo un 23% de aciertos, mostrando muchas dificultades para clasificar correctamente las clases, lo que refuerza su ineficiencia frente a los modelos preentrenados.

4. Dataset Mixto

- MobileNet y Xception fueron los modelos con mayor éxito en el dataset Mixto, alcanzando un 100% de aciertos cada uno. Esto demuestra la gran capacidad de estos modelos para manejar la variabilidad en las imágenes de diferentes fuentes y calidades, y para generalizar incluso frente a datos no previamente vistos.

- ResNet18 y VGG16 también tuvieron un buen rendimiento, con 86% y 93% de aciertos, respectivamente. Ambos modelos mostraron un buen desempeño en la mayoría de las clases, aunque presentaron algunos fallos en las pruebas con imágenes más complejas o atípicas.

- El modelo Custom fue nuevamente el de peor desempeño, con un 50% de aciertos, lo que indica su incapacidad para generalizar en escenarios donde las imágenes provienen de distintas fuentes.

5. Dataset WHU-RS19

- VGG16 mostró el mejor rendimiento en el dataset WHU-RS19, alcanzando un 100% de aciertos. Este modelo fue capaz de clasificar correctamente todas las clases, incluso frente a imágenes complejas y de alta resolución.

- MobileNet, ResNet18 y Xception también mostraron un buen rendimiento, con 80% de aciertos cada uno. Estos modelos lograron clasificar correctamente la mayoría de las clases, pero presentaron fallos en algunas imágenes más difíciles, como en la clase Parque.

- El modelo Custom tuvo el peor rendimiento, con 70% de aciertos, mostrando dificultades para generalizar y clasificar correctamente las imágenes más complejas del dataset.

6. Dataset MLRSNet

- Xception fue el modelo con mejor desempeño en el dataset MLRSNet, alcanzando un 79% de aciertos. Este modelo mostró una buena capacidad para manejar imágenes satelitales de alta resolución y clasificar correctamente la mayoría de las clases.

- MobileNet y ResNet18 también tuvieron un buen rendimiento, con 71% de aciertos cada uno, aunque presentaron algunos fallos en clases como Campo de Baloncesto y Campo de Fútbol.

- VGG16 tuvo un rendimiento algo inferior, con 57% de aciertos, mostrando dificultades para generalizar en las imágenes más complejas y con características atípicas.

- El modelo Custom, con solo un 21% de aciertos, fue el de peor desempeño en este dataset, lo que refuerza su incapacidad para adaptarse a imágenes satelitales de alta resolución y clasificar correctamente las distintas clases.

Teniendo en cuenta lo anterior, se puede concluir que los modelos MobileNet y Xception se destacan como los más robustos y consistentes en la mayoría de los datasets, alcanzando tasas de acierto superiores al 70% en casi todos los casos. Estos modelos preentrenados en ImageNet demostraron una gran capacidad para manejar la variabilidad de las imágenes, independientemente de la calidad o la fuente.

ResNet18 y VGG16 también tuvieron un buen desempeño en general, aunque con algunas fallas notables en los datasets más complejos o con imágenes atípicas, lo que redujo ligeramente sus tasas de aciertos.

El modelo Custom fue el menos efectivo en todos los datasets, con tasas de aciertos considerablemente más bajas en comparación con los modelos preentrenados. Esto subraya la

importancia del preentrenamiento en datasets grandes como ImageNet para mejorar el rendimiento en tareas de clasificación de imágenes.

4.5. Tiempos de cómputo de los modelos de CNN

El desempeño computacional de los modelos de clasificación de imágenes está altamente influenciado por la arquitectura del modelo, el tamaño del dataset y las capacidades del hardware utilizado. En este apartado, se analizan los tiempos de entrenamiento obtenidos para diferentes arquitecturas de redes neuronales convolucionales (CNN) aplicadas a distintos conjuntos de datos, considerando las especificaciones del equipo utilizado para la ejecución de los experimentos.

Especificaciones de Hardware

Los experimentos se realizaron en un equipo con las siguientes especificaciones técnicas:

Tabla 2. *Especificación del equipo usado para ejecutar los modelos de CNN.*

Componente	**Especificación**
Fabricante	Acer
Modelo	Nitro AN515-57
Sistema Operativo	Windows 11 Home Single Language (64 bits)
Procesador (CPU)	Intel Core i5-11400H @ 2.70 GHz
Memoria RAM	24 GB (24576 MB)
Versión de DirectX	DirectX 12
Tarjeta Gráfica (GPU)	NVIDIA GeForce GTX 1650 (4GB VRAM)
Memoria total GPU	16 GB (3,937 MB VRAM + 12,177 MB compartidos)

El uso de una GPU NVIDIA GTX 1650 con 4GB de VRAM fue determinante en la aceleración del entrenamiento de los modelos de deep learning. Sin embargo, la limitada capacidad de VRAM representó un desafío en modelos de mayor complejidad, como MobileNET y VGG16, donde los tiempos de entrenamiento fueron significativamente mayores.

Tiempo de cómputo de los modelos de CNN

En el Anexo G se presenta una tabla detallada con los tiempos de entrenamiento obtenidos para cada modelo en los distintos datasets, incluyendo el tiempo promedio por época y el tiempo total. A partir de esta información, se identificaron tendencias en el rendimiento computacional de cada arquitectura, evidenciando que modelos más complejos, como VGG16, MobileNet y Xception, requieren tiempos de entrenamiento significativamente mayores, mientras que arquitecturas más ligeras, como Custom y ResNet18, presentan tiempos más eficientes. Con base en estos resultados, se obtuvieron las siguientes conclusiones:

- MobileNet fue el modelo con mayor tiempo de entrenamiento en todos los datasets, debido a su arquitectura basada en separable convolutions, que incrementa la carga computacional.

- VGG16 y Xception también presentaron tiempos elevados, aunque en algunos casos fueron más eficientes que MobileNet, especialmente en datasets más pequeños.

- Custom y ResNet18 fueron los modelos más rápidos, por lo que resultan opciones más eficientes para entornos con limitaciones de hardware, esto teniendo en cuenta que el modelo Custom es el modelo con menor desempeño.

- Los datasets más grandes, como MLRSNet y PatternNET, generaron tiempos de entrenamiento más largos, con algunos modelos superando las 2 horas.

- Los datasets pequeños, como Optimal_31 y WHU-RS19, redujeron significativamente los tiempos de cómputo, lo que sugiere que disminuir el tamaño del dataset puede ser una estrategia viable para acelerar el entrenamiento en modelos más complejos.

5. APLICACIÓN DE MODELOS TRADICIONALES EN LA CLASIFICACIÓN DE ESPACIOS RECREATIVOS

5.1. Desarrollo de modelo de Machine Learning para evaluación de efectividad

Se desarrolló un modelo en el que se hace uso de bibliotecas populares de aprendizaje automático, procesamiento de imágenes y visualización. Está diseñado para realizar un flujo completo de trabajo de clasificación de imágenes, desde la carga y preprocesamiento de imágenes, hasta la construcción y evaluación de modelos de clasificación como Random Forest, SVM y K-Vecinos. Además, combina métodos tradicionales de machine learning con técnicas de visión por computadora para segmentar imágenes y hacer predicciones a nivel de ventanas de imagen. Este desarrollo responde al segundo objetivo específico del proyecto, que consiste en evaluar la efectividad en modelos tradicionales de Machine Learning para la clasificación de este tipo de espacios.

Se utilizaron 6 Datasets con el fin de contemplar escenarios en donde los resultados sean afectados por la calidad o cantidad de imágenes utilizadas en el modelo. Uno de estos modelos (llamado “Propio”) corresponde a un conjunto de imágenes creadas por nosotros a partir de imágenes de Google Maps ubicadas en ciudades de nuestro país. Otro dataset llamado “Mixto” se construyó tomando imágenes del dataset “Propio” y de otros datasets con el fin de probar rendimientos con una mezcla de imágenes de diferentes calidades:

Tabla 3. *Composición de los dataset utilizados para entrenamiento y pruebas en los modelos.*

	Composición	
Dataset	**Cantidad de Imágenes**	**Número de clases**
MLRSNet	7100	7
Optimal_31	240	4
PatternNet	4000	5
WHU	454	6
Propio	1705	5
Mixto	2128	7

El modelo desarrollado realiza varias tareas relacionadas con la clasificación de imágenes utilizando tres algoritmos de aprendizaje automático diferentes y la manipulación de imágenes:

- Cargar imágenes del dataset: contempla dos funciones, una que recorre las carpetas dentro de un directorio base para cargar imágenes y sus etiquetas correspondientes. Las

imágenes se redimensionan a un tamaño uniforme (256x256) y se convierten en arrays numéricos. La segunda función verifica que todas las imágenes tengan el mismo tamaño para garantizar la coherencia de los datos

- Preprocesamiento de los datos: Se asignan números a las etiquetas de las imágenes (por ejemplo, 'baseballdiamond' se codifica como 0, ‘tenniscourt’ se codifica como 1, etc.). Luego, se divide el conjunto de datos en dos partes: entrenamiento (80%) y prueba (20%).

- Entrenamiento de modelos: Se realiza un aplanado de imágenes: Las imágenes se convierten en un formato 1D (a través de reshape) para facilitar su procesamiento por los modelos de clasificación:
    - Random Forest
    - SVM (Support Vector Machine)
    - K-Vecinos

  Cada modelo predice las clases de las imágenes del conjunto de prueba y se calculan métricas como accuracy, precisión, recall, y F1-score.

Este modelo fue ejecutado en un computador con las siguientes características de hardware:

***Tabla 4**: Especificación del equipo usado para ejecutar los modelos de Machine Learning.*

Componente	**Especificación**
Sistema Operativo	Windows 10 pro
Procesador (CPU)	AMD Ryzen 5 5600x 6-core
Memoria RAM	32 GB DDR4
Versión de DirectX	DirectX 12
Tarjeta Gráfica (GPU)	NVIDIA GeForce RTX 3060Ti
Memoria total GPU	8 GB (8,192 MB VRAM + 16,345 MB compartidos)

Resultados de métricas del modelo

Se procesaron los datasets con cada uno de los modelos para lograr medir resultados en cuanto las métricas de calidad del modelo de clasificación y así determinar cómo se desempeña bajo diferentes condiciones. Estos fueron los resultados obtenidos:

Tabla 5. Resultados de las métricas de desempeño en cada Dataset para cada algoritmo.

Dataset	Algoritmo	Accuracy	Precision	Recall	F1-Score	Tiempo Entrenamiento
Optimal_31	Random Forest	0.5833	0.5891	0.5833	0.5688	2 min 3 seg
Optimal_31	SVM	0.5208	0.5275	0.5208	0.5119	5 min 7 seg
Optimal_31	KNN	0.3542	0.1829	0.3542	0.2321	2 min 2 seg
MLRS	Random Forest	0.6606	0.6668	0.6606	0.661	7 min 39 seg
MLRS	SVM	0.5408	0.5676	0.5408	0.5365	88 min 59 seg
MLRS	KNN	0.3599	0.5069	0.3599	0.3099	5 min 26 seg
PatternNET	Random Forest	0.7462	0.7425	0.7462	0.7398	3 min 12 seg
PatternNET	SVM	0.7238	0.7251	0.7238	0.7233	15 min 58 seg
PatternNET	KNN	0.5337	0.5672	0.5337	0.4989	2 min 49 seg
WHU	Random Forest	0.5934	0.6275	0.5934	0.5841	2 min 5 seg
WHU	SVM	0.6374	0.6918	0.6374	0.6259	5 min 21 seg
WHU	KNN	0.4396	0.4942	0.4396	0.3463	2 min 4 seg
Propio	Random Forest	0.4861	0.4775	0.4861	0.4728	2 min 20 seg
Propio	SVM	0.3375	0.3309	0.3375	0.3317	7 min 49 seg
Propio	KNN	0.2469	0.2255	0.2469	0.1901	2 min 14 seg
Mixto	Random Forest	0.7397	0.7535	0.7397	0.7245	10 min 35 seg
Mixto	SVM	0.6985	0.6876	0.6985	0.6851	20 min 20 seg
Mixto	KNN	0.3427	0.3892	0.3427	0.2664	8 min 22 seg

Estos resultados a nivel general representan uno de los escenarios que contemplábamos, relacionado con la calidad/cantidad de imágenes en los conjuntos de datos utilizado para los entrenamientos. Los mejores resultados fueron obtenidos utilizando un dataset con una cantidad representativa de imágenes (4.000 imágenes) y al mismo tiempo con una buena muy buena calidad (Conjunto de Datos de PatternNET). Los resultados más desfavorables están relacionados con uno de los conjuntos con peor calidad de resolución (conjunto de datos 'Propio').

Podemos llegar a algunas conclusiones respecto al desempeño o rendimiento:

- Random Forest es, en términos generales, el algoritmo que mejor se desempeña en la mayoría de los datasets. En especial, se destaca en el dataset PatternNET (Accuracy = 0.7462, F1-Score = 0.7398) y MLRS (Accuracy = 0.6606, F1-Score = 0.661). Para el dataset Mixto, obtenemos también buenos resultados (Accuracy = 0.7397, F1-Score = 0.7245). El tiempo de cómputo es bueno comparado con los otros tres algorimos, tomando como base el total de tiempos de entrenamiento de todos los datasets y algoritmos.

- SVM muestra un rendimiento también sólido, aunque en algunos datasets es inferior a Random Forest. Destaca en PatternNET (Accuracy = 0.7238) y en WHU (Accuracy = 0.6374).

Sin embargo, este algoritmo es el que requirió más tiempo de cómputo para poder entregar resultados (por ejemplo, 88 minutos en el dataset MLRS o 15 minutos en PatternNET).

- KNN tiene el rendimiento más bajo en casi todos los datasets. Por ejemplo, en Propio tiene un accuracy de 0.2469, lo que muestra un desempeño significativamente pobre en comparación con los otros algoritmos. En tiempos de cómputo, es el algoritmo que menos tiempo de cómputo utilizó para entregar resultados.

Respecto al rendimiento en los conjuntos de datos podríamos resaltar lo siguiente:

- Optimal31: En este conjunto obtuvimos resultados regulares, por lo cual consideramos que quizá no es el óptimo para nuestro propósito.

- PatternNET: Este dataset parece ser el más adecuado para los algoritmos, con Random Forest y SVM obteniendo las mejores puntuaciones en todas las métricas. Ambos algoritmos tienen altas precisiones, recalls y F1-Scores, lo que indica que ambos modelos identifican bien las instancias positivas y logran un equilibrio en la clasificación.

- MLRS: Aquí también Random Forest tiene el mejor desempeño en general. El SVM sigue con un desempeño decente, aunque notablemente más bajo que Random Forest, mientras que KNN tiene problemas significativos en términos de todas las métricas. Por supuesto, este dataset tomo mas tiempo en ser entrenado ya que tiene el mayor número de imágenes por dataset.

- WHU: En este dataset, SVM se desempeña mejor que Random Forest en precisión, pero no por una diferencia muy grande. KNN nuevamente tiene el peor desempeño.

- Propio: Todos los algoritmos tienen bajos rendimientos en este dataset, lo que sugiere que es un conjunto de datos más difícil de clasificar. Random Forest tiene un rendimiento algo mejor que SVM y KNN, pero sigue siendo bajo en comparación con los otros datasets.

- Mixto: Como esperábamos, se mejoran los resultados de las métricas de desempeño al combinar imágenes de diferente calidad; sin embargo, estos no superan los resultados obtenidos por el dataset de mayor calidad (PatternNet), lo cual nos indica que la calidad de definición en las imágenes juega un papel importante al mejorar los resultados de las métricas.

En cuanto a la comparación entre métricas, podemos resaltar que es importante observar la relación entre Precisión y Recall. Por ejemplo, en el dataset MLRS, la precisión de KNN es alta (0.5069), pero el recall es bajo (0.3599), lo que indica que KNN identifica bien los positivos, pero no detecta una cantidad suficiente de ellos. El F1-Score, que muestra el balance en la precisión y el recall (lo que lo hace útil para evaluar el rendimiento general) muestra, por ejemplo, que en el dataset MLRS el uso de Random Forest tiene un F1-Score de 0.661, lo que muestra que mantiene un buen balance entre precisión y recall.

KNN tiene el peor desempeño general en términos de todas las métricas. Esto es notable en datasets como Optimal_31 y Propio, donde tanto la precisión como el recall son extremadamente bajos. Esto podría indicar que el algoritmo KNN no es adecuado para estos conjuntos de datos, probablemente debido a la alta dimensionalidad o a la falta de patrones claros en los datos.

Una vez los modelos están entrenados, se desarrolla un ejercicio dirigido a la realización de las pruebas de predicción/clasificación.

5.2. Prueba predicción de nuevas imágenes

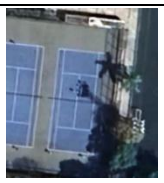
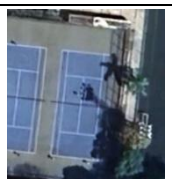
En este ejercicio de pruebas, por medio del modelo, se carga una imagen externa, se redimensiona y se convierte en un array numérico para hacer predicciones. Luego, utiliza los tres modelos entrenados para predecir la clase de una nueva imagen y devuelve los resultados para cada modelo.

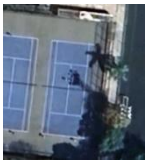
Se establecieron como marca de respuesta valores binarios en donde un resultado 'Positivo' evidencia una clasificación de la imagen efectiva en alguna de las clases de imágenes del modelo entrenado y 'Negativo' si la clasificación no era correcta. A continuación, se identifica la tabla de resultados y, posteriormente, el análisis de cada prueba y nuestros hallazgos frente al uso de estos algoritmos de Machine Learning:

Tabla 6. Resultados de las pruebas de clasificación en los modelos de ML.

Dataset	Tipo de Prueba	Image	Random Forest	SVM	K-Nearest
MLRSNet	Prueba 1		Positive	Negative	Negative

MLRSNet	Prueba 2		Negative	Positive	Positive
MLRSNet	Prueba 3		Negative	Negative	Negative
MLRSNet	Prueba 4		Negative	Negative	Negative
Optimal_31	Prueba 1		Positive	Positive	Negative
Optimal_31	Prueba 2		Positive	Negative	Negative
Optimal_31	Prueba 3		Positive	Positive	Positive
Optimal_31	Prueba 4		Positive	Negative	Negative
PatternNet	Prueba 1		Positive	Positive	Positive

PatternNet	Prueba 2		Negative	Positive	Negative
PatternNet	Prueba 3		Positive	Positive	Negative
PatternNet	Prueba 4		Positive	Positive	Negative
WHU	Prueba 1		Positive	Positive	Positive
WHU	Prueba 2		Positive	Positive	Negative
WHU	Prueba 3		Positive	Positive	Negative
WHU	Prueba 4		Positive	Negative	Negative
Propio	**Prueba 1**		**Positive**	**Negative**	**Positive**
Propio	**Prueba 2**		**Positive**	**Negative**	**Positive**

Propio	**Prueba 3**		**Positive**	**Negative**	**Positive**
Propio	**Prueba 4**		**Positive**	**Negative**	**Positive**
Mixto	Prueba 1		Positive	Positive	Positive
Mixto	Prueba 2		Positive	Negative	Negative
Mixto	Prueba 3		Positive	Negative	Negative
Mixto	Prueba 4		Positive	Negative	Positive

Conforme a estos resultados, podemos realizar algunos tipos de análisis:

- **Análisis por tipo de prueba:**

    a. **Prueba 1:** Clasificación de imágenes del mismo dataset utilizado en el entrenamiento:

        - En esta prueba, los algoritmos Random Forest y SVM generalmente muestran buenos resultados, obteniendo mayor cantidad de clasificaciones "Positivas" en la mayoría de los datasets. Esto es esperable, ya que las

imágenes que se prueban provienen del mismo conjunto utilizado en el entrenamiento, lo que facilita la predicción.

- KNN, sin embargo, muestra una tendencia a fallar más en esta prueba, obteniendo clasificaciones "Negativas" en varios datasets como Optimal 31 y MLRSNet. Esto podría indicar que KNN tiene dificultades para generalizar incluso dentro del mismo conjunto de datos, tal vez debido a la sensibilidad del algoritmo a la selección de características locales.

b. **Prueba 2:** Clasificación de una imagen obtenida de Google Maps (fuente externa)

- Esta prueba muestra variabilidad significativa. Random Forest, que anteriormente era muy efectivo, tiene problemas al clasificar correctamente imágenes externas en la mayoría de los datasets (clasificaciones mayormente "Negativas"), lo que sugiere que este algoritmo puede no generalizar bien cuando se enfrenta a datos ligeramente diferentes de los vistos en el entrenamiento.

- SVM tiene un desempeño mixto. Si bien logra clasificaciones "Positivas" en algunos datasets, como PatternNet y MLRSNet, también muestra fallas en otros conjuntos de datos. Esto puede deberse a que SVM tiende a encontrar límites de decisión que separan las clases basados en características específicas del dataset, y puede fallar cuando la imagen externa no se ajusta a esos límites.

- KNN, en cambio, tiene un mejor desempeño en esta prueba comparado con la primera, logrando "Positivo" en varios casos (como MLRSNet), lo que podría sugerir que, aunque KNN no es tan robusto dentro del conjunto de entrenamiento, puede ser útil para clasificaciones externas si los vecinos más cercanos son representativos.

c. **Prueba 3:** Clasificación de una imagen con características atípicas (color, textura)
- En esta prueba, Random Forest y SVM mantienen un rendimiento superior a KNN, con más clasificaciones "Positivas" en los datasets evaluados. Esto sugiere que ambos algoritmos tienen más capacidad de tolerar variabilidad

en las características visuales de las imágenes (como cambios en color o textura).

- KNN, por el contrario, parece ser el algoritmo más sensible a estos cambios atípicos, presentando muchas clasificaciones "Negativas", como en PatternNet. Esto podría deberse a que KNN depende en gran medida de las características exactas y locales de las imágenes para encontrar los vecinos más cercanos.

d. **Prueba 4:** Clasificación de una imagen con escenarios múltiples (por ejemplo, dos canchas de tenis)

- Random Forest y SVM nuevamente tienen el mejor desempeño, con clasificaciones mayoritariamente "Positivas". Esto indica que estos algoritmos pueden manejar bien escenarios donde las imágenes contienen más de un objeto de interés.

- KNN nuevamente muestra un bajo rendimiento en esta prueba, indicando que tiene dificultades para identificar correctamente imágenes con múltiples escenarios. Este resultado refuerza la idea de que KNN puede tener problemas para manejar complejidad en las imágenes.

- **Análisis por algoritmo:**

a. Random Forest:

- Random Forest muestra el mejor rendimiento general. Logra clasificaciones mayormente "Positivas" en todas las pruebas y datasets, excepto en la Prueba 2 (fuente externa). Esto lo posiciona como el algoritmo más robusto en términos de tolerancia a cambios atípicos en color, textura y múltiples objetos dentro de una imagen.

- Sin embargo, su rendimiento en la clasificación de imágenes externas (Prueba 2) es notablemente peor. Esto podría deberse a que Random Forest depende de características específicas del dataset de entrenamiento, y cuando se enfrenta a imágenes externas, el modelo no generaliza bien.

b. SVM:

- SVM tiene un rendimiento sólido, pero algo inferior al de Random Forest en general. SVM sobresale particularmente en la Prueba 2, donde clasifica algunas imágenes externas correctamente, lo que sugiere una capacidad de generalización superior a Random Forest en ciertos casos. Sin embargo, también presenta clasificaciones "Negativas" en datasets como WHU y MLRSNet.

- Este comportamiento mixto podría deberse a la sensibilidad de SVM a los datos de entrenamiento. Si el modelo encuentra buenos límites de decisión durante el entrenamiento, generaliza bien, pero si esos límites están sobreajustados al conjunto de entrenamiento, puede fallar con datos externos o más complejos.

c. K-**Nearest Neighbors (KNN):**

- KNN es el algoritmo con peor rendimiento en general. Su desempeño es más débil tanto en la clasificación de imágenes del conjunto de entrenamiento (Prueba 1) como en las imágenes atípicas o con escenarios múltiples.

- Sin embargo, en la Prueba 2 (fuente externa), KNN tiene un desempeño sorpresivamente mejor en algunos datasets, como MLRSNet y Propio, donde clasifica correctamente algunas imágenes externas. Esto podría ser porque KNN, al basarse en la distancia a los vecinos más cercanos, podría encontrar imágenes similares en el conjunto de entrenamiento.

5.3. Prueba de conteo de espacios urbanos deportivos en una imagen

Adicionalmente a las pruebas descritas anteriormente, se diseñó una prueba adicional a la que llamamos ‘Prueba de clasificación múltiple´, el cual consta de dos etapas:

1. **División de imágenes y detección de objetos:** se crea una función que divide una imagen en ventanas más pequeñas, dependiendo de la configuración y el tamaño de las imágenes cargada. Se muestra gráficamente cómo se ha dividido la imagen en varias ventanas para

corroborar la segmentación y luego se clasifica cada ventana obtenida de la imagen utilizando el modelo Random Forest y los otros modelos.

2. **Contar objetos:** Se creó una función que agrupa y cuenta cuántas veces aparecen diferentes tipos de objetos (como canchas de tenis o campos de fútbol) dentro de las ventanas generadas y se imprime el resultado.

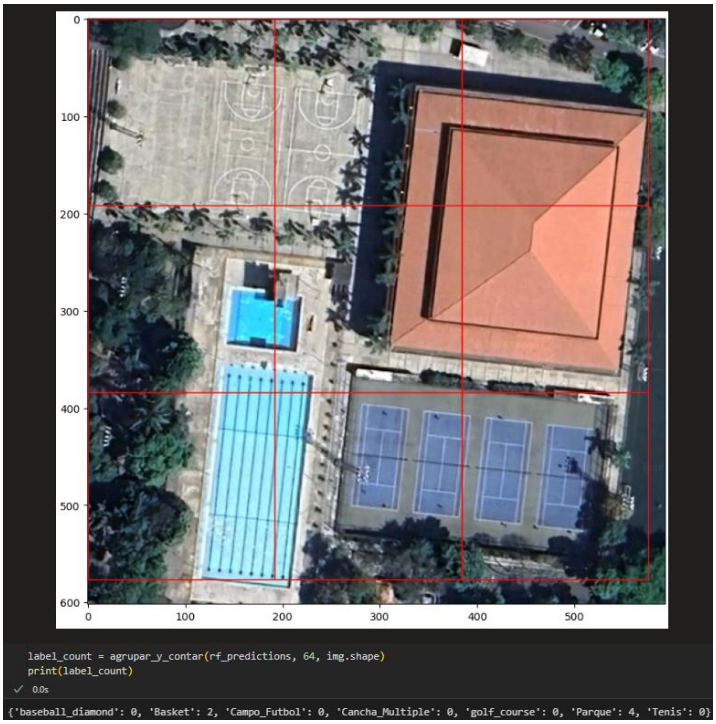
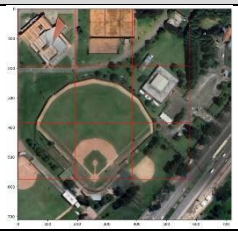
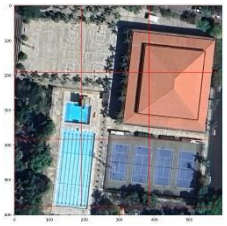
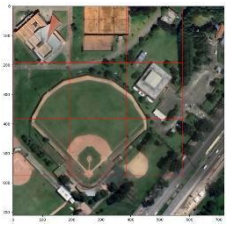
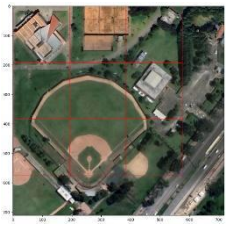
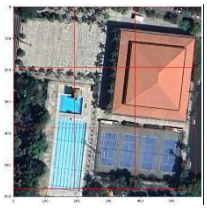


Figura 10. *Ejemplo de la división de una imagen para el conteo de espacios recreativos.*
Fuente: Elaboración propia

En esta prueba, la imagen cargada será revisada visualmente para establecer el número verdadero de cantidad de tipo de espacio que realmente existe en dicha imagen cargada. Luego, se registraban el número de espacios detectados por cada algoritmo en cada uno de los dataset. A continuación, registramos los resultados y también nuestros análisis respecto a ellos:

Tabla 7. Resultados pruebas de conteo por cada modelo y algoritmo

			Conteo de tipo de espacios bajo cada algoritmo			
Dataset	**Clases**	**Imagen**	**Prueba: Número real por tipo de espacios existentes en la imagen cargada**	**Random Forest**	**SVM**	**K-Nearest**
Optimal_31	baseball diamond		3	3	3	3
Optimal_31	basketball_court		0	1	3	2
Optimal_31	golf couse		0	2	2	4
Optimal_31	ground_track_field		0	3	1	0
MLRS	baseball_diamond		0	0	0	0
MLRS	basketball_court		2	2	3	7
MLRS	golf_course		0	0	1	0
MLRS	ground_track_field		0	0	0	1
MLRS	park		0	1	2	0
MLRS	stadium		0	6	3	1
MLRS	tennis_court		4	0	0	0
PatternNET	baseball_field		3	3	4	6
PatternNET	basketball_court		0	1	1	2
PatternNET	football_field		0	5	2	0
PatternNET	golf_course		0	0	2	0
PatternNET	tennis_court		3	0	0	1
PatternNET_2	baseball_field		0	4	1	3
PatternNET_2	basketball_court		3	3	5	3
PatternNET_2	football_field		0	2	1	3
PatternNET_2	golf_course		0	0	0	0
PatternNET_2	tennis_court		2	0	2	0
WHU	baseballdiamond		3	4	4	5
WHU	footballField		0	0	0	0
WHU	golfcourse		0	0	4	1
WHU	Park		0	2	0	0
WHU	Pond		0	2	1	3
WHU	tenniscourt		3	1	0	0
Propio	Basket		2	1	2	4
Propio	Campo_Futbol		0	0	0	2
Propio	Cancha_Multiple		0	0	3	0
Propio	Parque		0	3	1	0

Propio	Tenis		4	4	2	2
Mixto	baseball_diamond		0	0	1	0
Mixto	Basket		2	2	2	2
Mixto	Campo_Futbol		0	0	0	0
Mixto	Cancha_Multiple		2	0	2	0
Mixto	golf_course		0	0	2	7
Mixto	Parque		0	4	1	0
Mixto	Tenis		4	0	0	0

Los resultados de esta prueba de clasificación múltiple ofrecen una perspectiva sobre cómo diferentes algoritmos de métodos tradicionales de Machine Learning (Random Forest, SVM y K-Nearest Neighbors) manejan la segmentación y conteo de objetos dentro de imágenes más complejas. Podemos describir algunos hallazgos clave al respecto:

1. Desempeño de los Algoritmos por Dataset y Clases

- **Optimal_31:**
  - El conteo de "baseball diamond" es exacto en los tres algoritmos, lo que indica que este tipo de objeto es fácilmente reconocible para los tres modelos en este dataset.
  - Sin embargo, para "basketball court", la imagen real no contiene canchas de baloncesto (valor real = 0), pero SVM y KNN clasifican varios objetos como tales, mientras que Random Forest solo detecta uno incorrectamente. Esto indica que Random Forest es más preciso para evitar falsos positivos en esta clase.
  - En cuanto a "golf course", nuevamente, el valor real es 0, pero todos los algoritmos detectan algunos campos de golf, lo que sugiere que hay un patrón común de confusión para esta clase, determinado por las zonas verdes en la imagen.
- **MLRS:**
  - En esta categoría, Random Forest tiene un desempeño generalmente más

ajustado al número real de objetos en las clases, especialmente en objetos más específicos como "tennis court" (donde otros algoritmos no detectan nada), pero presenta errores en "park" y "stadium", donde sobredetecta.

- KNN comete varios errores importantes, como en el caso de la cancha de baloncesto, donde detecta hasta 7 elementos en comparación con el valor real de 2, y también presenta confusión en "stadium". Este sobredimensionamiento sugiere una sensibilidad mayor de KNN a características visuales no relevantes.

- **PatternNET:**

  - En "baseball field", Random Forest y SVM tienen buenos resultados, mientras que KNN tiende a sobre-detectar (6 canchas de béisbol en lugar de 3). Esto sugiere que KNN puede ser más propenso a generar falsos positivos en imágenes con características complejas.

  - En "football field", Random Forest detecta varios campos (5), aunque no existen en la imagen real, lo que indica una tendencia al sobredimensionamiento en esta clase.

  - Para las "tennis courts", ninguno de los algoritmos parece ser capaz de detectar correctamente los objetos, lo que sugiere que esta clase es especialmente difícil de identificar en este dataset.

- **WHU:**

  - "baseball diamond" presenta un sobre-dimensionamiento con todos los algoritmos, similar a otros datasets.

  - Curiosamente, para "tennis courts", Random Forest detecta solo 1 cancha cuando existen 3 reales, lo que sugiere que, en este caso, el modelo tiene una alta tasa de falsos negativos, posiblemente debido a problemas de segmentación o diferencias visuales dentro de las imágenes.

- **Propio:**

  - En general, todos los algoritmos tienen problemas para identificar correctamente los objetos, aunque KNN tiene algunos resultados más cercanos a la realidad en el caso de "Campo de Fútbol" y "Basket". Esto podría indicar que KNN tiene cierta ventaja en este dataset específico,

posiblemente debido a la naturaleza de las imágenes.

- **Mixto:**

    - Para la clase “basket”, es exacto en los tres algoritmos. Tiene que ver mucho el que el espacio detectado se asimila bastante a las imágenes utilizadas para el entrenamiento. Para la cancha multiple que está en la imagen de prueba no hay resultados positivos puesto que el espacio real está muy deteriorado y el delineamiento interno no está presente. Sin embargo, donde vemos un fallo es en la detección de canchas de tenis, la cual si debería haber funcionado por las imágenes utilizadas en el dataset de entrenamiento.

2. Tendencias y Patrones de los Algoritmos

- **Random Forest:**

    - En la mayoría de los casos, Random Forest tiene un desempeño relativamente consistente, con menos falsos positivos en algunas clases como "tennis court", pero tiende a sobredetectar en otras clases como "stadium" y "park". En términos generales, parece ser el algoritmo más equilibrado para el conteo y clasificación en la mayoría de los datasets.

- **SVM:**

    - SVM parece tener una tendencia más fuerte a sobredetectar objetos en varias clases, como "golf course" y "stadium", lo que puede deberse a la manera en que el modelo encuentra límites entre clases. Sin embargo, también tiene algunos buenos desempeños, como en "basketball court" y "tennis court" en ciertos datasets.

- **K-Nearest Neighbors (KNN):**

    - KNN parece tener el desempeño más inconsistente, con múltiples casos de sobre-detección (como en "basketball court" y "baseball field") y también fallos para detectar objetos que están presentes, como en "tennis court" en varios datasets. Este algoritmo parece ser el más afectado por la complejidad de las imágenes y la segmentación, probablemente debido a su dependencia en las características locales y su alta sensibilidad a la cantidad de datos cercanos.

6. APLICACIÓN DEL MODELO CNN EN LA IDENTIFICACIÓN DE ESPACIOS PÚBLICOS URBANOS

En esta sección se expone la aplicación del modelo de redes neuronales convolucionales (CNN), basado en la arquitectura MobileNet, para identificar y clasificar espacios recreativos en áreas urbanas a partir de imágenes satelitales. El modelo fue entrenado con un dataset propio que incluye diversos tipos de instalaciones recreativas, cumpliendo con el tercer objetivo específico del proyecto.

Se detallan los resultados obtenidos, incluyendo métricas como precisión, recall y F1-Score, que reflejan el desempeño del modelo en la clasificación de diferentes clases. Además, se presenta la implementación del modelo en una aplicación web, lo que facilita su uso práctico en la planificación y gestión de áreas urbanas.

6.1. Aplicación Modelo de Red Neuronal Convolucional sobre un entorno práctico

En esta sección, se describe la aplicación del modelo de redes neuronales convolucionales (CNN) desarrollado utilizando la arquitectura MobileNet, con el objetivo de identificar y clasificar espacios públicos recreativos en un área urbana específica, utilizando imágenes satelitales actuales y del entorno colombiano. Este proceso responde al tercer objetivo específico del proyecto, que consiste en aplicar el modelo entrenado en un entorno práctico y evaluar su desempeño en un escenario real.

El modelo fue entrenado con un dataset propio que contiene imágenes de diversas instalaciones recreativas como canchas de baloncesto, campos de fútbol, canchas múltiples, parques y canchas de tenis. Para esta aplicación práctica, el mismo conjunto de imágenes con las que se entrenó el modelo fue utilizado en la evaluación, con el fin de validar la precisión del modelo en un entorno controlado antes de ser probado en un entorno más complejo.

El proceso de predicción realizado se estructuró de la siguiente manera:

1. **Carga y Preprocesamiento de las Imágenes**: Las imágenes satelitales del conjunto de datos propio fueron procesadas de manera uniforme. Utilizando la librería Torchvision, las imágenes se redimensionaron a 256x256 píxeles y se normalizaron de acuerdo con los parámetros estándar utilizados en MobileNet, que fueron entrenados previamente con ImageNet.

2. **Evaluación del Modelo**: Se cargó el modelo previamente entrenado en MobileNet y se aplicaron las imágenes preprocesadas. A lo largo del proceso de inferencia, el modelo

generó predicciones que fueron comparadas con las etiquetas verdaderas de cada imagen.

3. **Generación de Métricas de Desempeño**: Se calcularon las métricas de desempeño (precisión, recall, F1-Score) para cada clase utilizando la biblioteca scikit-learn, y los resultados obtenidos se detallan a continuación.

Resultados Obtenidos

El modelo de MobileNet entrenado con el dataset propio arrojó resultados sobresalientes en términos de las métricas de desempeño:

Tabla 8. Métricas de desempeño del modelo MobileNET sobre la base de datos propia.

Clase	Precision	Recall	F1 Score
Basket	0.97	1.00	0.98
Campo_Futbol	1.00	0.98	0.99
Cancha_Multiple	0.99	0.98	0.98
Parque	1.00	1.00	1.00
Tenis	1.00	1.00	1.00

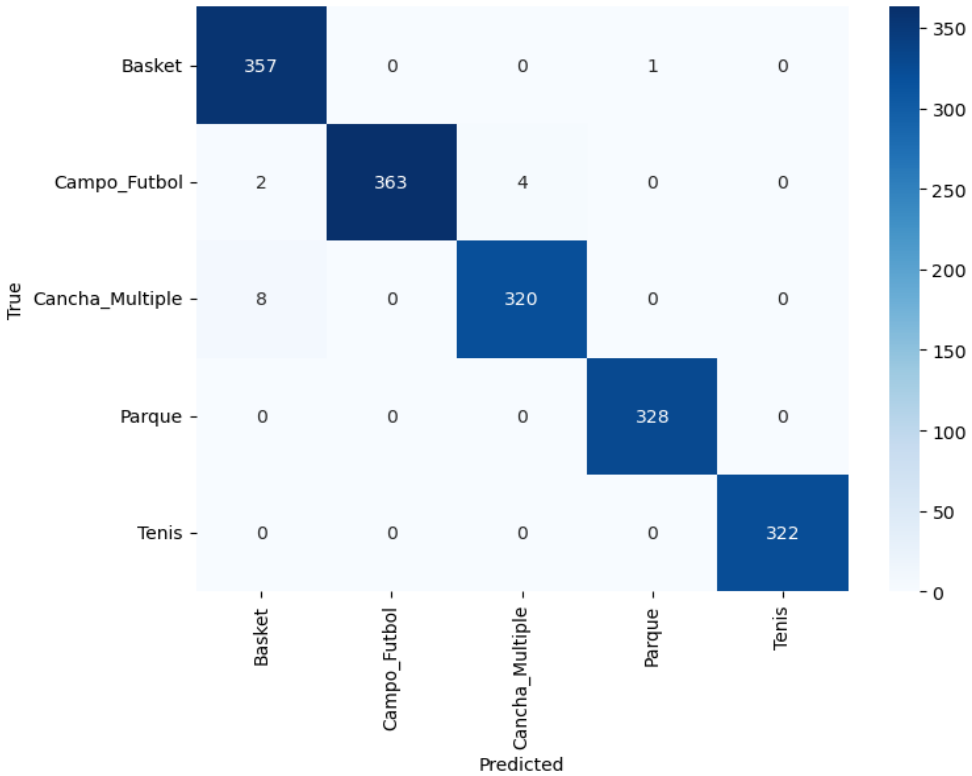


***Figura 11**. Matriz de confusión relacionada con la clasificación del modelo MobileNET sobre la base de datos Propia. Fuente: Elaboración propia*

Además, la Figura 11 (matriz de confusión adjunta) proporciona una visión detallada del desempeño del modelo al comparar las predicciones realizadas con las etiquetas verdaderas de cada clase. Como se observa, el modelo predijo correctamente la mayoría de las clases, con errores mínimos, como en el caso de la clase "Campo de Fútbol" y “Basket”, donde el modelo confundió algunas instancias con la clase "Cancha Múltiple".

Análisis de Resultados

Los resultados muestran un rendimiento excepcional del modelo con una precisión general del 99% y un F1-Score también de 99%. Estas métricas tan elevadas pueden atribuirse en gran parte al hecho de que el modelo fue evaluado utilizando el mismo conjunto de datos con el que fue entrenado. Esto implica que el modelo ya estaba familiarizado con las características de las imágenes, lo que reduce la probabilidad de error.

1. **Precisión y Recall por Clase**: Se observan valores de precisión y recall superiores al 98% en todas las clases, lo que indica que el modelo es capaz de identificar correctamente los verdaderos positivos (imágenes clasificadas correctamente) y que además no genera falsos negativos (imágenes mal clasificadas como otras clases). Este alto nivel de desempeño es consistente en todas las categorías: Basket, Campo de Fútbol, Cancha Múltiple, Parque y Tenis.

2. **Matriz de Confusión**: En la Figura 11, se aprecia que el modelo clasificó correctamente la mayoría de las imágenes. El pequeño número de errores se presentó principalmente en la clase Cancha Múltiple, que fue confundida en algunos casos con Campo de Fútbol. Esto puede deberse a las similitudes en las estructuras visuales de estas clases cuando se observan desde una perspectiva satelital.

3. **Razón de las Métricas Altas**: Es importante destacar que los resultados obtenidos presentan métricas muy elevadas debido al uso del mismo conjunto de datos con el que se entrenó el modelo. Si bien esto valida que el modelo aprendió correctamente las características de las clases, es probable que al evaluar el modelo con imágenes completamente nuevas o con variaciones más significativas, los resultados sean menos óptimos. Por esta razón, es necesario complementar estas pruebas con imágenes satelitales provenientes de otras fuentes para asegurar una adecuada generalización del modelo.

En conclusión, el modelo de MobileNet constituye una herramienta eficaz para la clasificación de espacios recreativos urbanos, cumpliendo con el tercer objetivo del proyecto al aplicar con éxito un modelo CNN en un entorno práctico. Su alta precisión lo convierte en una solución prometedora para la planificación y gestión de áreas recreativas, aunque será fundamental evaluar su rendimiento en escenarios más desafiantes para asegurar su efectividad en aplicaciones del mundo real.

6.2. Despliegue del Modelo de Red Neuronal Convolucional en una Aplicación Web

En esta fase del proyecto, se ha desarrollado una aplicación web utilizando Python y la librería de Flask para la integración y despliegue de una red neuronal convolucional (CNN) previamente entrenada. Esta aplicación web permite a los usuarios cargar imágenes, procesarlas mediante el modelo de red neuronal, y visualizar los resultados de las predicciones directamente en la interfaz del navegador. A continuación, se detallan los componentes y procesos involucrados:

1. Integración del Modelo de Red Neuronal: Para el despliegue del modelo, se ha utilizado Flask, un framework web ligero en Python. El modelo de red neuronal convolucional, que ha sido entrenado previamente y guardado en un archivo .h5, es cargado al iniciar la aplicación. Este modelo es responsable de clasificar imágenes en diferentes categorías de espacios recreativos, como campos de baloncesto, fútbol, golf, etc. La carga del modelo se realiza una sola vez cuando la aplicación se inicia, lo que permite realizar predicciones eficientes en tiempo real.

2. Desarrollo de la Interfaz de Usuario: La aplicación web cuenta con una interfaz intuitiva diseñada con HTML para facilitar la interacción del usuario:

- **Formulario de Carga:** Los usuarios pueden subir imágenes desde sus dispositivos a través de un formulario de carga. También pueden especificar el tamaño de la cuadrícula en la que desean dividir la imagen para el análisis.

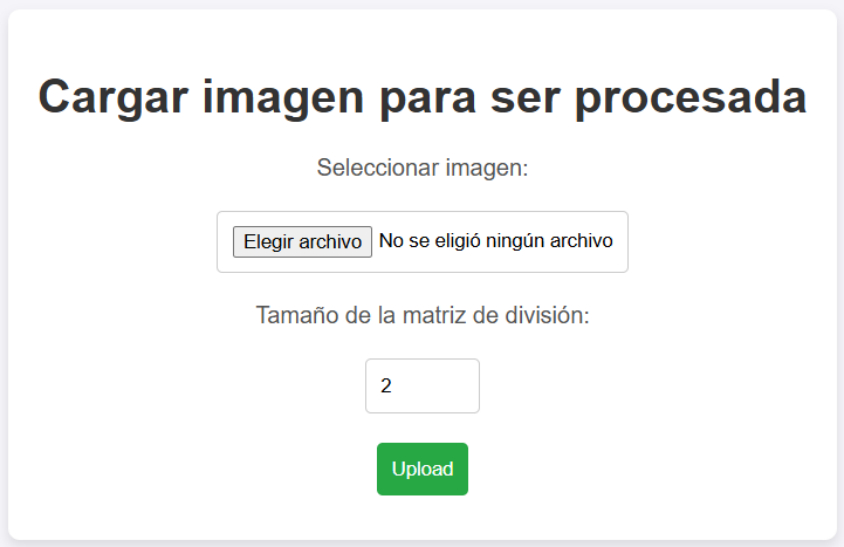


***Figura 12**. Pantalla de inicio de la interfaz con el usuario. Fuente: Elaboración propia*

- **Página de Resultados:** Una vez procesada la imagen, se muestra una página de resultados que presenta la imagen procesada con las predicciones visualizadas en ella.

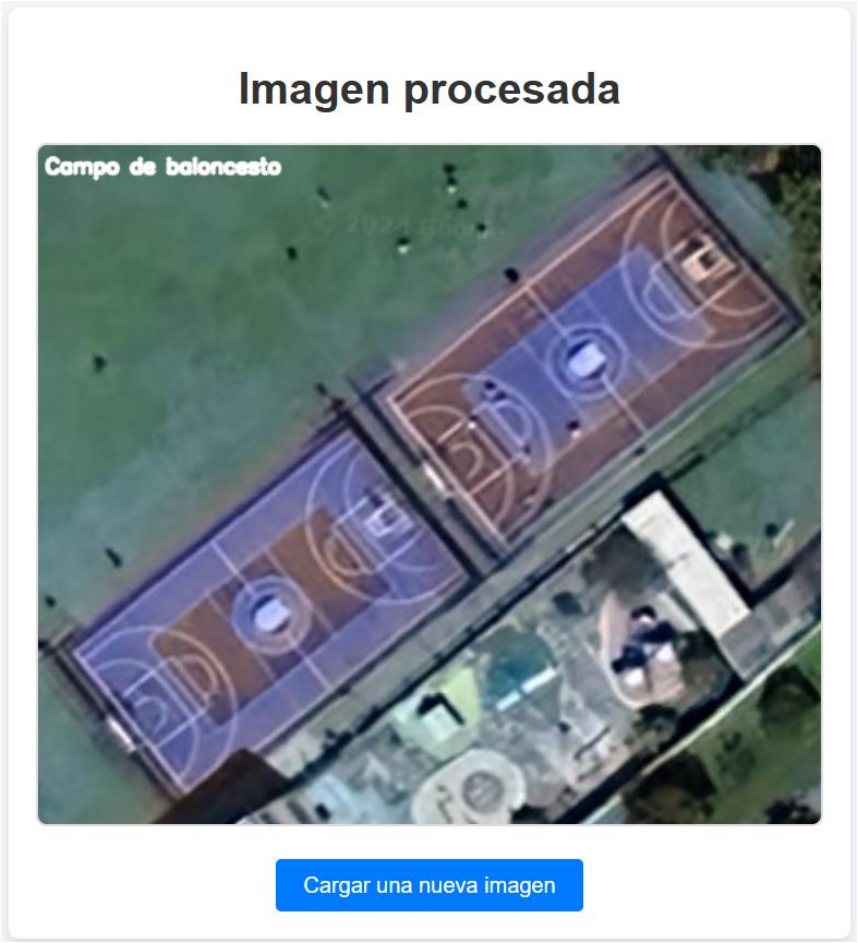


***Figura 13**. Pantalla de carga de imagen para análisis. Elaboración propia*

3. Proceso de Preprocesamiento y Análisis de Imágenes: El procesamiento de imágenes se realiza en varias etapas para asegurar que sean adecuadas para el modelo de red neuronal:

- **Conversión a RGB:** Las imágenes subidas se convierten a formato RGB si no están ya en este formato, utilizando OpenCV. Esto es necesario porque el modelo espera imágenes en formato RGB.

- **Redimensionamiento:** Cada sección de la imagen se redimensiona a un tamaño de 256x256 píxeles, que es el tamaño de entrada requerido por el modelo para realizar las predicciones.

- **Normalización:** Las imágenes se normalizan dividiendo los valores de los píxeles por 255 y ajustando la media, lo que mejora la precisión del modelo.

- **División en Secciones:** La imagen se divide en una cuadrícula de tamaño especificado por

el usuario. Cada sección se procesa de manera independiente para obtener una predicción.

4. Generación de Resultados Visuales: Una vez que cada sección de la imagen ha sido clasificada, los resultados se visualizan en la imagen original:

- **Etiquetado de Secciones:** Las predicciones se etiquetan en cada sección de la imagen con el nombre de la clase correspondiente. Si la probabilidad de la predicción es baja, se etiqueta como "Nada".

- **Dibujo de Cuadrícula:** Se dibujan líneas en la imagen para representar las secciones analizadas, facilitando la visualización de las predicciones en el contexto de la imagen completa.

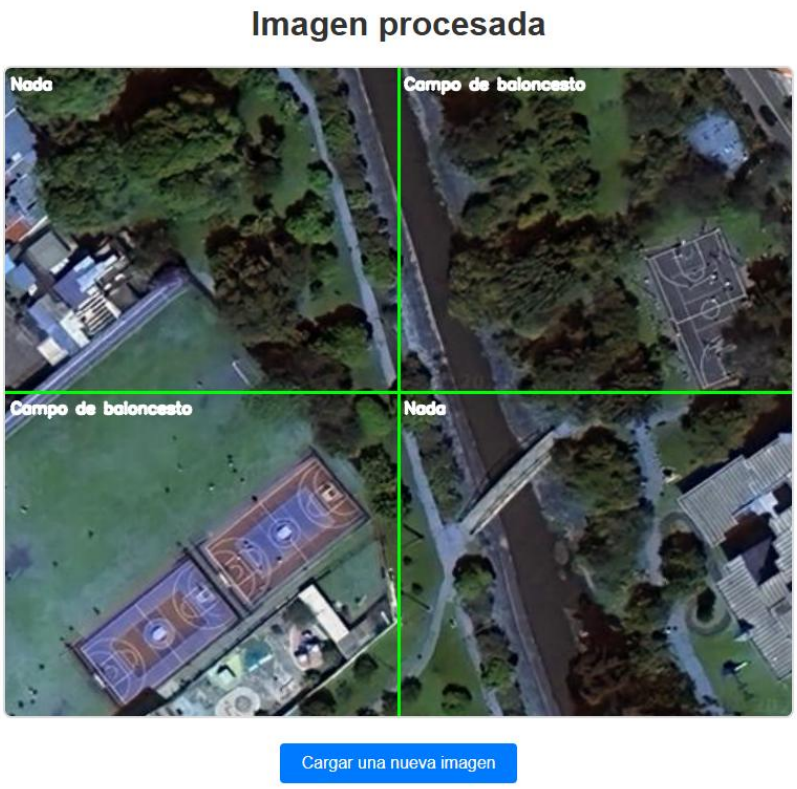


Figura 14. *Pantalla de división de imagen y resultado. Elaboración propia*

5. Conversión y Presentación de Resultados: La imagen procesada se convierte a formato base64, que es adecuado para su inclusión en una página web:

- **Conversión a Base64:** La imagen procesada se convierte a una cadena de texto en formato base64. Esto permite incrustar la imagen directamente en el HTML sin necesidad de

archivos adicionales.

- **Renderización en HTML:** La cadena base64 se inserta en una plantilla HTML y se muestra en la página de resultados, permitiendo a los usuarios ver la imagen con las predicciones sobreimpresas.

6. Manejo de Errores y Robustez: Se han implementado mecanismos para manejar errores comunes durante el procesamiento, como problemas con el formato de la imagen o errores de redimensionamiento. Estos mecanismos aseguran que la aplicación sea robusta y ofrezca retroalimentación útil en caso de fallos, mejorando la experiencia del usuario.

7. CONCLUSIONES Y TRABAJOS FUTUROS

7.1. Conclusiones

La aplicación de redes neuronales convolucionales (CNN) como MobileNet, ResNet18 y Xception ha demostrado ser altamente efectiva para la clasificación de imágenes satelitales en este proyecto. MobileNet alcanzó una muy buena precisión del 96.90% en la base de datos propia, mostrando una gran capacidad de generalización en escenarios que involucran imágenes variadas. De igual manera, la alta puntuación F1 de 97.88% lograda por Xception en PatternNET refuerza su eficiencia para clasificar imágenes aéreas con gran precisión. Estos resultados destacan a las CNN como herramientas poderosas para el análisis geoespacial y la clasificación de espacios recreativos públicos.

Los resultados del proyecto revelaron una brecha significativa en el desempeño entre las CNN y los métodos tradicionales de aprendizaje automático (ML) como Random Forest, Máquinas de Soporte Vectorial (SVM) y K-Nearest Neighbors (KNN). Las CNN mostraron una mayor capacidad para aprender características complejas y jerárquicas en las imágenes satelitales, logrando una clasificación más precisa en clases desafiantes, como las canchas múltiples. Los modelos tradicionales de ML, en contraste, a menudo enfrentan problemas con la sobre-detección y la confusión de clases, mientras que las CNN, gracias a su profundidad y habilidad para procesar grandes volúmenes de datos, capturaron patrones abstractos y redujeron los errores. Las CNN superaron a los modelos de ML por más de un 10% en precisión y recall, confirmando su superioridad para manejar imágenes complejas y variadas.

Aunque los modelos tradicionales de ML como Random Forest y SVM son más eficaces con conjuntos de datos más pequeños, este proyecto demostró que las CNN producen resultados más robustos y eficientes cuando se dispone de grandes volúmenes de imágenes satelitales. Por ejemplo, las CNN alcanzaron una precisión del 96.90% en la base de datos propia, mientras que los modelos de ML mostraron limitaciones al extraer características relevantes. Sin embargo, los modelos de ML siguen siendo útiles cuando el volumen de datos es bajo o la calidad de las imágenes es deficiente, ya que su menor costo computacional y capacidad de generalización con menos datos pueden ser ventajosos.

Los modelos preentrenados como Xception y MobileNet consistentemente superaron al modelo personalizado en todas las métricas. Por ejemplo, en el conjunto de datos MLRSNet, MobileNet alcanzó una precisión del 89.86%, en comparación con solo un 77.42% del modelo personalizado,

lo que subraya la importancia de aprovechar modelos preentrenados en grandes conjuntos de datos como ImageNet. Los modelos preentrenados extraen características más profundas y complejas que los modelos creados desde cero, lo que resulta en una mejor generalización en diversos conjuntos de datos.

A pesar del éxito de las CNN en la clasificación de imágenes satelitales, una de las limitaciones identificadas en este proyecto fue el conteo preciso de las clases, especialmente cuando varias clases coexisten en una misma imagen. En algunos casos, la capacidad del modelo para diferenciar entre clases cercanas, como canchas múltiples y campos de fútbol, se vio afectada por la superposición de características en las imágenes. Esto sugiere que es necesario mejorar los algoritmos de conteo, posiblemente integrando técnicas adicionales como segmentación más precisa o la inclusión de modelos específicos de conteo para cada clase, con el fin de mejorar la identificación y conteo de áreas recreativas individuales dentro de imágenes más complejas.

Otro logro importante de este proyecto fue la implementación de un aplicativo web que permite la clasificación y conteo de espacios recreativos en imágenes satelitales de manera interactiva. Este desarrollo no solo facilita el uso del modelo por parte de usuarios sin conocimientos técnicos avanzados, sino que también abre la puerta para futuras mejoras en la accesibilidad y escalabilidad del sistema. La integración del modelo de CNN en un entorno web proporciona una plataforma flexible para la visualización de los resultados y el análisis de datos en tiempo real, permitiendo a los usuarios cargar sus propias imágenes y obtener predicciones instantáneas. Además, la posibilidad de ajustar parámetros como el número de divisiones en las imágenes para obtener resultados más detallados demuestra el potencial del aplicativo para aplicaciones prácticas en diferentes entornos urbanos. Este avance representa un paso importante hacia la democratización de tecnologías avanzadas de clasificación de imágenes y su implementación en soluciones urbanas reales.

Una de las contribuciones más significativas de este proyecto es su enfoque en la clasificación del suelo urbano recreativo. Las CNN permitieron identificar de manera efectiva diferentes tipos de espacios recreativos urbanos, como canchas deportivas y parques, lo que facilita un análisis más detallado y preciso de la distribución de estas áreas en zonas urbanas. Esta capacidad es clave para respaldar decisiones de planificación urbana y desarrollo sostenible, ofreciendo información valiosa para la creación de políticas públicas dirigidas a mejorar la calidad de vida en las ciudades. El enfoque propuesto ofrece una solución robusta para mapear usos de suelo recreativo en zonas urbanas, superando las limitaciones de los métodos tradicionales de clasificación.

7.2. Trabajos Futuros

Este proyecto tiene un gran potencial de desarrollo en diversas áreas, las cuales podrían mejorar considerablemente la calidad y utilidad de los resultados obtenidos:

- **Generación de un Dataset con Imágenes Locales del Territorio Colombiano:** Una mejora clave sería la creación de un conjunto de datos específico para el territorio colombiano, utilizando imágenes satelitales locales. Esto permitiría que los modelos capturen mejor las características únicas de las áreas urbanas de Colombia, mejorando la precisión en la clasificación de espacios recreativos. Adicionalmente, se sugiere la exploración de otras fuentes de imágenes satelitales, como Esri, ArcGIS o plataformas gubernamentales, para obtener imágenes de mayor calidad que las ofrecidas por Google Maps. Esto permitiría que los modelos trabajen con datos más representativos y detallados, lo que potencialmente mejoraría la efectividad en las predicciones.

- **Implementación de un Validador de Espacios:** Un validadora visual que permita a los usuarios revisar los resultados de clasificación sería una adición valiosa. Esto permitiría a los usuarios confirmar si las imágenes identificadas corresponden correctamente a las clases previstas por el modelo, generando un ciclo de retroalimentación para mejorar la precisión del modelo a largo plazo. Este enfoque, similar al utilizado en aplicaciones como Google Fotos para la validación de rostros, podría permitir que los usuarios marquen imágenes como correctas o incorrectas, ayudando al entrenamiento dinámico y continuo del modelo.

- **Segmentación y Clasificación de Espacios:** Ampliar la funcionalidad del modelo para incluir la segmentación de los espacios recreativos identificados permitiría no solo contar los espacios, sino también determinar su tamaño o la superficie ocupada. Esta funcionalidad podría ser particularmente útil en estudios de planificación urbana, ya que proporcionaría información sobre el déficit de espacio recreativo en áreas densamente pobladas. La combinación de esta información con datos oficiales sobre el uso del suelo y la densidad demográfica permitiría un análisis más detallado y equitativo de la disponibilidad de espacios recreativos en diferentes regiones.

- **Técnicas de redimensionamiento variable:** Una mejora clave para futuros desarrollos sería implementar métodos de redimensionamiento variable que adapten dinámicamente el tamaño de las imágenes al modelo, preservando mejor la calidad de los datos originales. Este enfoque podría utilizar algoritmos que ajusten el tamaño de las imágenes según su resolución inicial, permitiendo que aquellas con mayor resolución mantengan detalles

críticos, mientras que las de baja resolución se optimicen sin pérdida significativa de información.

8. REFERENCIAS BIBLIOGRÁFICAS

[1] Banco Interamericano de Desarrollo, "Áreas Verdes Urbanas en Latinoamérica y el Caribe," 2005. Disponible en: https://publications.iadb.org/publications /spanish/ document/%C3%81reas-verdes-urbanas-en-Latinoam%C3%A9rica-y-el-Caribe.pdf.

[2] M. A. Gilabert, J. González-Piqueras y J. García-Haro, "Acerca de los índices de vegetación," Revista de Teledetección, núm. 8, 1997. Disponible en: https: //www.researchgate.net/publication/39195330_Acerca_de_los_indices_de_vegetacion.

[3] J. E. Jiménez Caldera and G. Y. Durango Severiche, "Diagnóstico y planificación del espacio público urbano. La participación de los ciudadanos usuario," Universidad Nacional de Colombia, 2020.

[4] J. Ding, N. Xue, G.-S. Xia, X. Bai, W. Yang, M. Y. Yang, S. Belongie, J. Luo, M. Datcu, M. Pelillo, and L. Zhang, "Object Detection in Aerial Images: A Large-Scale Benchmark and Challenges," 2021.

[5] Y. T. Hernández Peña, "El ordenamiento territorial y su construcción social en Colombia: ¿un instrumento para el desarrollo sustentable?," Cuad. Geogr. - Rev. Colomb. Geogr., núm. 19, pp. 97-109, 2010.

[6] Ministerio de Vivienda, Ciudad y Territorio, "ABC de los POT - Plan de Ordenamiento Territorial," 2012. Disponible en: https://www.minvivienda.gov.co/sites/default /files/documentos/ABC%20de%20los%20POT%20%20Plan%20de%20Ordenamiento%20Territorial.pdf.

[7] J. A. Valero Medina and B. E. Alzate Atehortúa, "Comparison of maximum likelihood, support vector machines, and random forest techniques in satellite images classification," Tecnura, vol. 23, no. 59, pp. 13–26, Jan. 2019.

[8] Ingeoexpert, "Clasificaciones de imágenes de satélite", Nov 13, 2020, https://ingeoexpert.com/articulo/clasificaciones-de-imagenes-de-satelite

[9] M. Á. López Reyes, "Métodos de aprendizaje profundo para la identificación de objetos en imágenes satelitales," Universidad de Jaén, 2023. Disponible en: https://crea.ujaen.es/bitstreams/df6bbadc-4450-4daa-a7d8-538ed39f2ecd/download

[10] IBM, "¿Qué es el aprendizaje por transferencia?,". Disponible en: https://www.ibm.com/mx-es/topics/transfer-learning.

[11] M. Rodríguez and M. Moctezuma, "Análisis Bayesiano y Fusión de Datos Para La Clasificación de Escenas Urbanas del Distrito Federal," Ingeniería, Investigación y Tecnología VII, México D.F., vol. 1, pp. 17-28, 2006.

[12] "Visión por Computadora," Libro online de IAAR. Disponible en: https://iaarbook.github.io/vision-por-computadora/.

[13] "Redes Neuronales Convolucionales: qué son, tipos y aplicaciones," Telefónica. Disponible en: https://www.telefonica.com/es/sala-comunicacion/blog/redes-neuronales-convolucionales-que-son-tipos-aplicaciones/.

[14] "Redes neuronales convolucionales (CNN): características y usos," ISDI. Disponible en: https://www.isdi.education/es/?p=26288.

[15] X. X. Zhu, D. Tuia, L. Mou, G. S. Xia, L. Zhang, F. Xu, & F. Fraundorfer, "Deep learning in remote sensing: A comprehensive review and list of resources," IEEE Geoscience and Remote Sensing Magazine, vol. 5, no. 4, pp. 8-36, 2017. https://ieeexplore.ieee.org/document/8113128

[16] E. Karypidis, S. G. Mouslech, K. Skoulariki, and A. Gazis, "Comparison Analysis of Traditional Machine Learning and Deep Learning Techniques for Data and Image Classification," 2022. https://doi.org/10.48550/arXiv.2204.05983

[17] L. L. Ankile, M. F. Heggland y K. Krange, "Deep Convolutional Neural Networks: A survey of the foundations, selected improvements, and some current applications," arXiv preprint arXiv:2011.12960, 2020. Disponible en: https://arxiv.org/abs/2011.12960.

[18] S. K. Agrawal, "Metrics to Evaluate your Classification Model to take the right decisions," Sept 29, 2023. https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/

[19] J. Garzón Barrero y N. E. Sánchez Pineda, "Evaluación comparativa de los algoritmos de aprendizaje automático Support Vector Machine y Random Forest: efectos del tamaño del conjunto de entrenamiento," Ciencia e Ingeniería Neogranadina, vol. 33, núm. 2, pp. 131-148,

2023. Disponible en: https://www.redalyc.org/journal/911/91178599010/.

[20] N. Chucos Baquerizo y E. J. Vega Ventocilla, "Evaluación de algoritmos de machine learning en la clasificación de imágenes satelitales multiespectrales, caso: Amazonia Peruana," Ciencia Latina Revista Científica Multidisciplinar, vol. 6, núm. 1, pp. 1234-1250, 2022. Disponible en: https://ciencialatina.org/index.php/cienciala/article/view/1843.

[21] M. Pérez Ortiz de Landaluce, "Clasificación de imágenes mediante algoritmos de Deep Learning," Universidad de Sevilla, 2020. Disponible en: https://biblus.us.es/bibing/proyectos/abreproy/93772/fichero/TFG3772%2BP%C3%89REZ%2BORTIZ%2BDE%2BLANDALUCE%2C%2BMARTA.pdf.

[22] O. Ghorbanzadeh, T. Blaschke, K. Gholamnia, S. R. Meena, D. Tiede, & J. Aryal, "Evaluation of different machine learning methods and deep-learning convolutional neural networks for landslide detection," Remote Sensing, vol. 11, no. 2, p. 196, 2019. https://www.mdpi.com/2072-4292/11/2/196

[23] IBM, "What is random forest?", https://www.ibm.com/topics/random-forest.

[24] IBM, "What are SVMs?", https://www.ibm.com/topics/support-vector-machine?.

[25] Syriopoulos, P.K., Kalampalikis, N.G., Kotsiantis, S.B. et al. kNN Classification: a review. Ann Math Artif Intell (2023). https://doi.org/10.1007/s10472-023-09882-x

[26] Z. Somogyi, “The Application of Artificial Intelligence: Step-by-Step Guide from Beginner to Expert”. Cham, Switzerland: Springer, 2021. Disponible en: https://link.springer.com/book/10.1007/978-3-030-60032-7

[27] I. Goodfellow, Y. Bengio, y A. Courville, *Deep Learning*. MIT Press, 2016.

[28] Interactive Chaos, "Estructura de una Red Neuronal," Tutorial de Machine Learning, 2025. [En línea]. Disponible en: https://interactivechaos.com/es/manual/tutorial-de-machine-learning/estructura-de-una-red-neuronal. [Accedido: 29-ene-2025].

[29] S. Hochreiter y J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735-1780, 1997.

[30] A. Krizhevsky, I. Sutskever, y G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[31] K. He, X. Zhang, S. Ren, y J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

[32] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, y R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[33] K. Simonyan y A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[34] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251-1258.

[35] A. G. Howard et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[36] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265-283.

[37] F. Chollet, "Keras: The Python Deep Learning library," 2015. https://keras.io.

[38] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, 2019, pp. 8024-8035.

[39] Flask, "Flask web framework," 2010. https://flask.palletsprojects.com/.

[40] A. Asokan and J. Anitha, "Machine Learning based Image Processing Techniques for Satellite Image Analysis -A Survey," 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 2019, pp. 119-124.

[41] L. Ma et al., "Deep learning in remote sensing applications: A meta-analysis and review," ISPRS Journal of Photogrammetry and Remote Sensing, 2019. https://www.sciencedirect.com/science/article/pii/S0924271619301108?via%3Dihub#ab005

[42] L. Zhang et al., "Deep Learning for Remote Sensing Image Classification: A Survey," WIREs Data Mining and Knowledge Discovery, vol. 8, no. 6, e1264, 2018. https://wires.onlinelibrary.wiley.com/doi/10.1002/widm.1264

[43] A. Javed, T. Kim, C. Lee, J. Oh, and Y. Han, "Deep Learning-Based Detection of Urban Forest Cover Change along with Overall Urban Changes Using Very-High-Resolution Satellite Images," Remote Sens., vol. 15, p. 4285, 2023. https://doi.org/10.3390/rs15174285

[44] L. Valdés Ávila and J. Baquero Vanegas, "Deep Learning aplicado a imágenes satelitales como herramienta de detección de viviendas sin servicio de energía en el caserío Media Luna – Uribia – Guajira." Jul 16, 2019.

[45] J. Shao et al., "Benchmarking the WHU-RS19 dataset for remote sensing image classification," *Remote Sensing Letters*, vol. 9, no. 4, pp. 356-365, 2018.

[46] Y. Yang y S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," *Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2010, pp. 270-279.

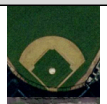
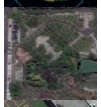
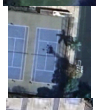
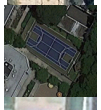
[47] W. Zhou, S. Newsam, C. Li, y Z. Shao, "PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 145, pp. 197-209, 2018.

[48] F. Hu et al., "Optimal-31: A novel dataset for deep learning research on aerial image classification," *Remote Sensing*, vol. 10, no. 7, p. 1024, 2018.

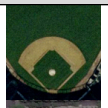
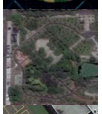
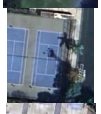
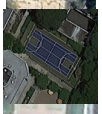
[49] J. Xu, L. Zhang, Y. Tong, y Q. Huang, "MLRSNet: A multi-label benchmark dataset for remote sensing scene classification," *Remote Sensing Letters*, vol. 11, no. 5, pp. 420-429, 2020.

ANEXOS

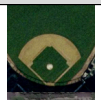
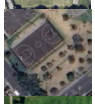
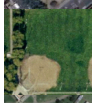
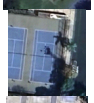
Anexo A. Resultados pruebas de predicción sobre diferentes clases usando modelos entrenados con la base de datos Propia.

Tipo de prueba	Clase	Imagen	Dataset propio				
			Custom	MobileNet	ResNet18	VGG16	Xception
Prueba 1	Campo de Baseball		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 1	Parque		Negative	Negative	Positive	Negative	Negative
Prueba 1	Campo de Baloncesto		Positive	Positive	Negative	Negative	Positive
Prueba 1	Campo de Baloncesto		Positive	Negative	Negative	Negative	Negative
Prueba 1	Campo de Fútboll		Negative	Positive	Positive	Positive	Negative
Prueba 1	Campo de Baloncesto		Positive	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baseball		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 2	Campo de Baseball		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 2	Campo de Baloncesto		Negative	Positive	Positive	Positive	Positive
Prueba 3	Campo de Tenis		Positive	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Positive	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Positive	Negative	Negative	Negative	Positive
Prueba 4	Campo de Tenis multiple		Positive	Positive	Positive	Positive	Positive
Prueba 4	Campo de Tenis multiple		Negative	Positive	Negative	Positive	Positive
			64%	73%	64%	64%	73%

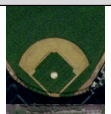
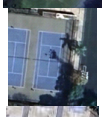
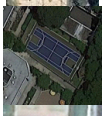
Anexo B. Resultados pruebas de predicción sobre diferentes clases usando modelos entrenados con la base de datos PatternNET.

Tipo de prueba	Clase	Imagen	PatternNet				
			Custom	MobileNet	ResNet18	VGG16	Xception
Prueba 1	Campo de Baseball		Positive	Positive	Positive	Positive	Positive
Prueba 1	Parque		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 1	Campo de Baloncesto		Positive	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baloncesto		Positive	Positive	Positive	Positive	Positive
Prueba 1	Campo de Fútboll		Negative	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baloncesto		Positive	Positive	Positive	Negative	Negative
Prueba 1	Campo de Baseball		Positive	Positive	Positive	Negative	Negative
Prueba 2	Campo de Baseball		Positive	Negative	Negative	Positive	Negative
Prueba 2	Campo de Baloncesto		Positive	Positive	Positive	Positive	Positive
Prueba 3	Campo de Tenis		Negative	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Positive	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Positive	Positive	Positive	Positive	Positive
Prueba 4	Campo de Tenis multiple		Negative	Negative	Negative	Negative	Negative
Prueba 4	Campo de Tenis multiple		Negative	Positive	Positive	Positive	Positive
			69%	85%	85%	77%	69%

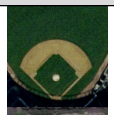
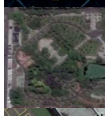
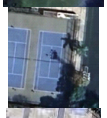
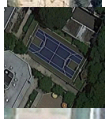
Anexo C. Resultados pruebas de predicción sobre diferentes clases usando modelos entrenados con la base de datos Optimal 31.

Tipo de prueba	Clase	Imagen	Optimal 31				
			Custom	MobileNet	ResNet18	VGG16	Xception
Prueba 1	Campo de Baseball		Positive	Negative	Positive	Positive	Positive
Prueba 1	Parque		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 1	Campo de Baloncesto		Negative	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baloncesto		Negative	Positive	Positive	Negative	Positive
Prueba 1	Campo de Fútboll		Negative	Negative	Negative	Negative	Negative
Prueba 1	Campo de Baloncesto		Negative	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baseball		Positive	Negative	Positive	Negative	Positive
Prueba 2	Campo de Baseball		Positive	Negative	Positive	Positive	Positive
Prueba 2	Campo de Baloncesto		Negative	Positive	Positive	Positive	Positive
Prueba 3	Campo de Tenis		Negative	Negative	Negative	Negative	Negative
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Negative	Negative	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Negative	Positive	Positive	Positive	Positive
Prueba 4	Campo de Tenis multiple		Negative	Negative	Negative	Negative	Negative
Prueba 4	Campo de Tenis multiple		Negative	Negative	Negative	Negative	Negative
			23%	38%	69%	54%	69%

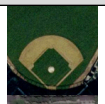
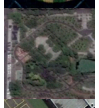
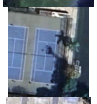
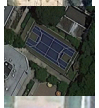
Anexo D. Resultados pruebas de predicción sobre diferentes clases usando modelos entrenados con la base de datos Mixta.

Tipo de prueba	Clase	Imagen	Mixto				
			Custom	MobileNet	ResNet18	VGG16	Xception
Prueba 1	Campo de Baseball		Positive	Positive	Positive	Positive	Positive
Prueba 1	Parque		Negative	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baloncesto		Positive	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baloncesto		Positive	Positive	Positive	Positive	Positive
Prueba 1	Campo de Fútboll		Negative	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baloncesto		Positive	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baseball		Positive	Positive	Negative	Negative	Positive
Prueba 2	Campo de Baseball		Negative	Positive	Negative	Positive	Positive
Prueba 2	Campo de Baloncesto		Positive	Positive	Positive	Positive	Positive
Prueba 3	Campo de Tenis		Negative	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Positive	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Negative	Positive	Positive	Positive	Positive
Prueba 4	Campo de Tenis multiple		Negative	Positive	Positive	Positive	Positive
Prueba 4	Campo de Tenis multiple		Negative	Positive	Positive	Positive	Positive
			50%	100%	86%	93%	100%

Anexo E. Resultados pruebas de predicción sobre diferentes clases usando modelos entrenados con la base de datos WHU.

Tipo de prueba	Clase	Imagen	WHU				
			Custom	MobileNet	ResNet18	VGG16	Xception
Prueba 1	Campo de Baseball		Positive	Positive	Positive	Positive	Positive
Prueba 1	Parque		Negative	Positive	Positive	Positive	Negative
Prueba 1	Campo de Baloncesto		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 1	Campo de Baloncesto		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 1	Campo de Fútboll		Negative	Positive	Positive	Positive	Negative
Prueba 1	Campo de Baloncesto		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 1	Campo de Baseball		Negative	Positive	Positive	Positive	Positive
Prueba 2	Campo de Baseball		Positive	Positive	Positive	Positive	Positive
Prueba 2	Campo de Baloncesto		No Definido	No Definido	No Definido	No Definido	No Definido
Prueba 3	Campo de Tenis		Positive	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Positive	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Positive	Negative	Negative	Positive	Positive
Prueba 4	Campo de Tenis multiple		Positive	Negative	Negative	Positive	Positive
Prueba 4	Campo de Tenis multiple		Positive	Positive	Positive	Positive	Positive
			70%	80%	80%	100%	80%

Anexo F. Resultados pruebas de predicción sobre diferentes clases usando modelos entrenados con la base de datos MLRSNet.

Tipo de prueba	Clase	Imagen	MLRSNet				
			Custom	MobileNet	ResNet18	VGG16	Xception
Prueba 1	Campo de Baseball		Negative	Positive	Positive	Positive	Positive
Prueba 1	Parque		Negative	Positive	Positive	Positive	Positive
Prueba 1	Campo de Baloncesto		Positive	Negative	Negative	Negative	Negative
Prueba 1	Campo de Baloncesto		Negative	Negative	Negative	Negative	Negative
Prueba 1	Campo de Fútboll		Negative	Positive	Positive	Negative	Positive
Prueba 1	Campo de Baloncesto		Negative	Negative	Negative	Negative	Positive
Prueba 1	Campo de Baseball		Negative	Positive	Positive	Negative	Positive
Prueba 2	Campo de Baseball		Positive	Positive	Positive	Positive	Positive
Prueba 2	Campo de Baloncesto		Negative	Negative	Negative	Negative	Negative
Prueba 3	Campo de Tenis		Negative	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Negative	Positive	Positive	Positive	Positive
Prueba 3	Cancha Multiple - Campo de Balncesto - Campo de Tenis		Positive	Positive	Positive	Positive	Positive
Prueba 4	Campo de Tenis multiple		Negative	Positive	Positive	Positive	Positive
Prueba 4	Campo de Tenis multiple		Negative	Positive	Positive	Positive	Positive
			21%	71%	71%	57%	79%

Anexo G. Tabla de tiempos de ejecución de cada uno de los modelos de CNN y su respectiva base de datos.

Dataset	Arquitectura	Tiempo promedio por época (Seg)	Tiempo total
WHU-RS19	Custom	17	4 min
	VGG16	56	14 min
	Resnet18	78	20 min
	MobileNet	95	24 min
	Xception	57	14 min
Optimal_31	Custom	7	2 min
	VGG16	19	5 min
	Resnet18	17	4 min
	MobileNet	36	9 min
	Xception	22	6 min
PatternNET	Custom	91	23 min
	VGG16	362	1 hrs 30 min
	Resnet18	320	1 hrs 20 min
	MobileNet	487	2 hrs 1 min
	Xception	314	1 hrs 18 min
Propio	Custom	43	11 min
	VGG16	165	41 min
	Resnet18	201	50 min
	MobileNet	137	34 min
	Xception	112	28 min
MLRSNet	Custom	201	50 min
	VGG16	540	2 hrs 15 min
	Resnet18	368	1 hrs 32 min
	MobileNet	635	2 hrs 38 min
	Xception	479	1 hrs 59 min
Mixto	Custom	45	11 min
	VGG16	184	46 min
	Resnet18	255	1 hrs 3 min
	MobileNet	172	43 min
	Xception	162	40 min

Anexo H. Mapa mental que consolida los conceptos investigados para conformar el Marco Teórico.

Anexo I. Mapa mental que consolida los antecedentes utilizados dentro de la investigación para el proyecto.