CON PATAS Y COLAS: IMPLEMENTANDO CNNs PARA CLASIFICAR IMÁGENES DE GATOS Y PERROS. (abril, 2024)

Daza Pereira Juan Pablo y Camargo Sanchez Juan Sebastian

Aprendizaje Profundo.

Escuela Colombiana de Ingeniería Julio Garavito

Resumen - En este estudio, exploramos la clasificación de imágenes utilizando redes neuronales convolucionales (CNNs) en el conjunto de datos "cats_vs_dogs". Dividimos nuestro trabajo en tres etapas clave: entrenamiento, desarrollo y pruebas.

Comenzamos descargando el conjunto de datos y preparándolo para el entrenamiento al ajustar el tamaño de las imágenes. Luego diseñamos dos planes de experimentación, cada uno con enfoques ligeramente diferentes, y seleccionamos modelos para experimentar: Modelo Denso, Modelo Convolucional y Modelo Convolucional Plus.

Durante la fase de entrenamiento, alimentamos nuestros modelos con los datos de entrenamiento y los ajustamos para optimizar su rendimiento. Luego, evaluamos la eficacia de cada modelo en el conjunto de datos de desarrollo, utilizando métricas de evaluación estándar.

Con base en los resultados de la fase de desarrollo, seleccionamos el mejor modelo y lo sometimos a una validación final utilizando el conjunto de datos de pruebas. Este paso nos permitió evaluar la capacidad de nuestro modelo para generalizar a datos no vistos.

Nuestro estudio proporciona una guía completa para el desarrollo y la evaluación de modelos de clasificación de imágenes en el conjunto de datos "cats_vs_dogs", con el objetivo de identificar el modelo más eficaz y generalizable.

Palabras clave – Experimentación, Clasificación de imágenes, CNNs, Evaluación de modelos, Precisión, Conjunto de datos, Cats_vs_dogs, Modelos, Redes neuronales convolucionales.

I. INTRODUCCIÓN

En el inmenso campo de la inteligencia artificial, las máquinas han obtenido la capacidad de comprender, entender y clasificar imágenes en los últimos años, esto gracias al desarrollo de técnicas de aprendizaje profundo, en particular las redes neuronales convolucionales (CNNs). Estas redes, inspiradas en el funcionamiento del cerebro humano, han demostrado una gran habilidad para el trabajo que vamos a realizar, la clasificación de imágenes.

Entre la cantidad de colecciones de datos que existen para entrenar y evaluar algoritmos de clasificación de imágenes, nos encontramos con el dataset “cats_vs_dogs” que nos servirá como piedra angular para el desarrollo y evaluación del proyecto.

En este artículo, nos embarcamos en el mundo de la experimentación con el conjunto de datos que disponemos. Nuestro objetivo es explorar varios modelos y determinar cuál de ellos ofrece mejores términos de precisión y generalización.

Nuestra experimentación se divide en tres etapas importantes: entrenamiento, desarrollo (Dev) y pruebas (Test). Durante la fase de entrenamiento, como es costumbre, contaremos con cierta cantidad de datos para entrenar la variedad de modelos propuestos.

Posteriormente, en la fase de desarrollo, evaluaremos el rendimiento de cada modelo en un conjunto de datos separados, ajustando parámetros y arquitecturas según sea necesario para mejorar la precisión y evitar sobreajustes.

Finalmente, una vez identificados el mejor modelo en la fase de desarrollo, pasaremos a la fase de pruebas, donde lo evaluaremos utilizando el conjunto de

datos de prueba. Este paso final nos permitirá evaluar y validar la capacidad de nuestro modelo para generalizar datos no vistos y proporcionar una evaluación de su rendimiento.

A lo largo de este artículo, les daremos a conocer la experimentación realizada, los resultados y conclusiones, brindando una visión de los desafíos y oportunidades que vimos en el campo de la clasificación de imágenes.

II. MARCO TEÓRICO

Este estudio se basa en los conceptos fundamentales del aprendizaje automático y las redes neuronales convolucionales (CNNs), por ello es importante proporcionar varios conceptos para poder comprender el proceso que se va a realizar para la clasificación de imágenes y la aplicación de modelos en el conjunto de datos “cats_vs_dogs”.

Aprendizaje automático:

El aprendizaje automático es un subconjunto de inteligencia artificial que permite que un sistema aprenda y mejore de forma autónoma mediante redes neuronales y aprendizaje profundo, sin tener que ser programado explícitamente, a través de la ingesta de grandes cantidades de datos. (Google Cloud, s.f.)

Clasificación de imágenes:

La clasificación de imágenes es una rama de la inteligencia artificial con numerosas aplicaciones, para la que se utilizan redes neuronales capaces de detectar patrones en las imágenes. (BEATANCUR, 2024)

Redes neuronales convolucionales (CNNs):

Las redes neuronales convolucionales son un tipo de redes neuronales artificiales que imitan la estructura y el funcionamiento de las redes neuronales cerebrales

humanas. Con capas convolucionales que extraen características, capas de agrupación que reducen la dimensionalidad y capas totalmente conectadas para la clasificación final. (MathWorks, s.f.)

Evaluación de modelos:

La evaluación de modelos se realiza mediante métricas como precisión, F1- score y matriz de confusión. Estas métricas permiten medir el rendimiento y la capacidad de generalización de los modelos en conjuntos de datos de prueba no vistos. (Gamco, 2021)

Trabajos relacionados:

- **"Building powerful image classification models using very little data" por François Chollet:** Este artículo presenta un enfoque para construir modelos de clasificación de imágenes potentes utilizando pequeñas cantidades de datos, utilizando el conjunto de datos "cats_vs_dogs" como ejemplo. (Chollet, 2016)
- **GitHub Repositorios:** Proyectos de código abierto en GitHub que utilizan este conjunto de datos para la experimentación y la implementación de modelos de clasificación de imágenes. (ki-ljl, 2022)

Este marco teórico proporciona una comprensión a los conceptos y técnicas fundamentales en la clasificación de imágenes con redes neuronales convolucionales.

III. METODOLOGÍA

1. **Obtención y preprocesamiento de datos:**
    - Descargar el conjunto de datos “cats_vs_dogs” de la página oficial de TensorFlow Datasets.
    - Conocimiento de los datos con lo que se cuentan (número de datos de perros, número de datos de gatos, etc).
    - Dividir el conjunto de datos de entrenamiento, desarrollo (dev) y prueba (test) en una proporción adecuada, 80%, 10%, 10% respectivamente.
    - Ajustar el tamaño de las imágenes y sus características según las necesidades, esto con el fin de facilitar el entrenamiento.
2. **Diseño de planes de experimentación:**

    Para este punto, se diseñaron dos estrategias para la experimentación de los modelos. Las estrategias van a ser nombradas como “Plan A” y “Plan B”.
    - **Plan A:** El plan consiste en los siguientes pasos:
        - ✓ **Manipulación de datos:** En esta parte, se manipularán las imágenes a nuestras necesidades, manipulando su tamaño y características.
        - ✓ **Preparación de datos:** Con los datos ya manipulados, se pasa a la parte de prepararlos para los modelos con los que se van a experimentar.
        - ✓ **Entrenamiento de los modelos:** Ya por último se entrenarán los modelos escogidos con los datos divididos para esta parte, y ya con los resultados arrojados, se evaluará cuál fue el mejor modelo.
    - **Plan B:** El plan B cuenta con los mismos pasos del plan A, el único paso que cambia es el primero (Manipulación de datos) donde se le añade un extra que es hacer aumento de datos. Este aumento de

datos se piensa realizar en caso de que los modelos de la experimentación no se ajusten a lo proyectado.

Cabe aclarar que, para poder realizar el Plan B, necesitaremos contar con capacidad de GPU y RAM. Sino contamos con esa capacidad, el plan B no será realizado.

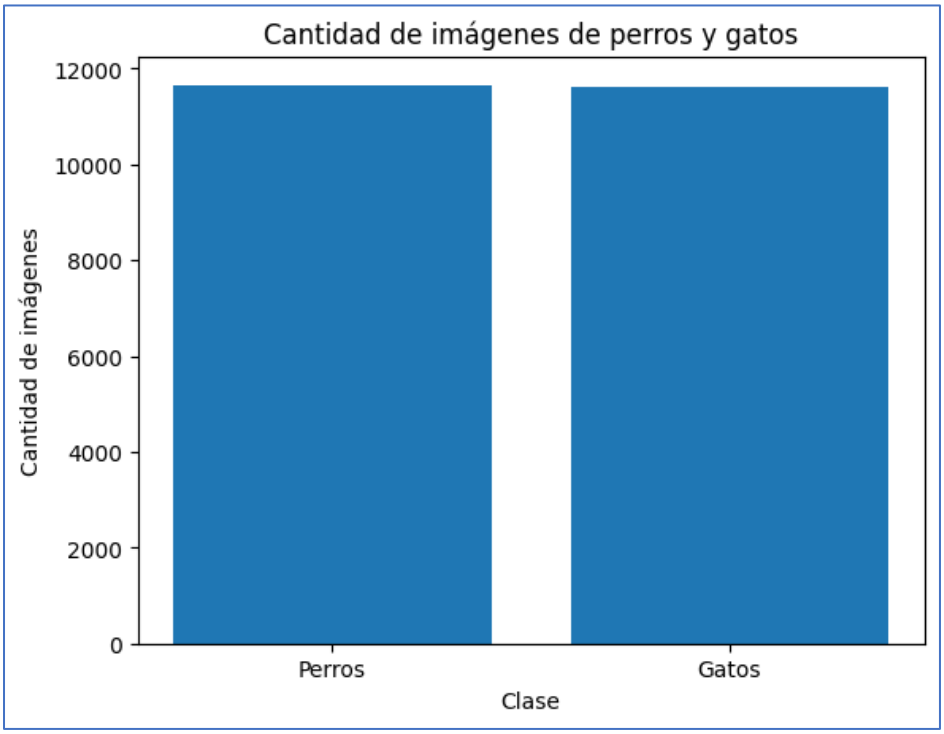
3. **Diseño de los modelos a experimentar:**
    - Diseño de las arquitecturas a experimentar:
        - ✓ Modelo Denso.
        - ✓ Modelo Convolucional.
        - ✓ Modelo Convolucional Plus (con un dropout).
    - Definir las funciones de activación y los hiperparámetros de cada modelo de experimentación.
4. **Entrenamiento de los modelos:**
    - Utilizar el conjunto de datos de entrenamiento para entrenar el modelo.
5. **Evaluación de los modelos:**
    - Evaluar el rendimiento de los modelos en el conjunto de datos de desarrollo (dev) utilizando métricas.
6. **Elección del modelo final:**
    - Luego de comparar el rendimiento de los modelos después de la fase de desarrollo y dependiendo de los resultados, se escogerá el mejor modelo para pasarlo por la fase de validación y ver si se cumple con lo proyectado (80% de precisión).
7. **Validación del modelo:**
    - Finalmente, se evaluará el modelo escogido con el conjunto de datos de pruebas para obtener una evaluación final de su rendimiento y generalización.

IV. RESULTADOS

Descubrimiento y tratamiento de datos

Decidimos analizar el data set de perro y gatos para saber sus características, columnas, sus datos útiles e inútiles. El data set consiste en tres columnas: imagen, imagen/filename, etiqueta. La etiqueta es la que clasifica a ambos animales. Uno para gato y cero para perro. Contamos con un data set balanceado por lo que no hay necesidad de tratar la cantidad de datos entre las imágenes de perros y de gatos (Figura A). Lo que si hicimos fue limitar el conjunto de datos a 20000 para hacer más fácil la división de las etapas de desarrollo de la red, que serían de 1600 para entrenamiento, 2000 para desarrollo y 2000 para pruebas (80/10/10). También por razones tecnológicas tuvimos que llevar las imágenes a blanco y negro y a un tamaño fijo de 150 x 150 x 1 (Figura B). Ya que en el mismo descubrimiento y procesamiento de datos la RAM de nuestras maquinas llegaban a su límite.

Figura A



Grafica del dataset con un numero de 11658 de perro y un numero de 11604 de gatos.

Figura B

Ejemplar de 36 imágenes de perros y gatos llevados a blanco y negro con dimensiones de 150 x 150 x1.

Preparamos un generador de datos a partir de nuestros datos, si el desarrollo de nuestra red falla en la etapa de prueba (Figura C). No se tuvo la necesidad de agregar estos nuevos datos por el desarrollo de nuestra red fue exitoso.

Figura C

Nuevas imágenes generadas a partir del dataset. Se rotaron y se cambiaron de altura y ancho.

Creación de modelos

En nuestro plan esta construir modelos para elegir el mejor y desarrollar el trabajo sobre este. Para eso tuvimos tres candidatos: red densa, red convolucional, red convolucional aplicando una regularización con dropout. Usamos en las redes convolucionales la iniciación de 32 filtros con un kernel de 3x3 por estándar (Figura D).

Figura D

```
[ ]  modeloDenso = tf.keras.models.Sequential([SSS
        tf.keras.layers.Flatten(input_shape=(100, 100, 1)),
        tf.keras.layers.Dense(150, activation = 'relu'),
        tf.keras.layers.Dense(150, activation = 'relu'),
        tf.keras.layers.Dense(1, activation = 'sigmoid')
    ])

    modeloCNN = tf.keras.models.Sequential([
        tf.keras.layers.Conv2D(32,(3,3), activation ='relu', input_shape=(100, 100, 1)),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Conv2D(64,(3,3), activation ='relu'),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Conv2D(128,(3,3), activation ='relu'),
        tf.keras.layers.MaxPooling2D(2,2),

        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(100, activation = 'relu'),
        tf.keras.layers.Dense(1, activation = 'sigmoid')
    ])

    modeloCNN2 = tf.keras.models.Sequential([
        tf.keras.layers.Conv2D(32,(3,3), activation ='relu', input_shape=(100, 100, 1)),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Conv2D(64,(3,3), activation ='relu'),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Conv2D(128,(3,3), activation ='relu'),
        tf.keras.layers.MaxPooling2D(2,2),

        tf.keras.layers.Dropout(0.5),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(250, activation = 'relu'),
        tf.keras.layers.Dense(1, activation = 'sigmoid')
    ])
```

Código de la arquitectura con que llevaremos a experimentación.

Cada uno de estos modelos se ejecutarán con optimización Adam por su desempeño con imágenes y su regularización aplicada. Entropía binaria cruzada que sea función de perdida y con métrica de exactitud.

Experimentación en etapa de entrenamiento

Decidimos que vamos a empezar a experimentar con la arquitectura, hiperparametros de batch y el número de épocas. En los primeros resultados de nos dimos cuenta de que una red densa era imprecisa y lenta para el procesamiento de los datos comparada con los resultados que nos daban ambas redes convolucionales. Comparando el 0.76 de exactitud y 0.5036 de perdida, contra, 0.99 de precisión y 0.0037 de perdida. Aunque sabíamos que estas primeras pruebas de experimentación había sobre entrenamientos en ambas, el comportamiento de las CNN era mucho mejor y manejable, si hablamos de tiempos de ejecución y manejo de los datos. Por lo que decidimos tomar la red neuronal sola, sin ninguna

regularización, como es el modelo sobrante. Aunque aprendimos de ese modelo que en algún punto de nuestra experimentación iba a ser necesario usarla.

Desarrollo

Ahora ya con nuestra modelo empezaremos a optimizar y regular para obtener la precisión deseada. Por lo que esta etapa es la de entrenamiento y desarrollo en conjunto.

Primero empezamos reduciendo el batch y el número de épocas. También reducimos el número de capas convolucionales para ver el comportamiento y a partir de eso desarrollar el modelo óptimo para nuestro objetivo. Así que cada vez que se hacía una experimentación sobre un modelo íbamos cambio primero su número de batch y después su arquitectura (número de capas convolucionales y capas densas con un numero de neuronas). En la tabla 1 se puede ver el proceso de la experimentación en el desarrollo de nuestra red. El ultimo ejemplar es la red que se eligió y que tiene la arquitectura de la figura E.

Tabla 1

Experimentación	loss	accuracy	val_loss	val_accuracy
1 Capas Conv - 20 neuronas - batch 2	0.6932	0.504	0.6931	0.5045
1 Capas Conv - 28 neuronas - batch 2	0.0334	0.9912	2.3979	0.67
1 Capas Conv - 32 neuronas - batch 2	0.0235	0.9936	2.7135	0.6945
1 Capas Conv - 100 neuronas - batch 2	0.0158	0.9961	2.6839	0.678
2 Capas Conv - 100 neuronas - batch 2	0.0158	0.9956	2.199	0.712
2 Capas Conv - 64 neuronas - Dropout - batch 2	0.1641	0.9401	0.9536	0.7515
2 Capas Conv - 100 neuronas - Dropout - batch 2	0.1934	0.9236	0.8171	0.731
3 Capas Conv - 250 neuronas - Dropout - batch 4	0.0468	0.9859	1.701	0.7465
3 Capas Conv - 250 neuronas - batch 8	0.0302	0.9901	1.2957	0.794
4 Capas Conv - 300 neuronas - Dropout - batch 22	0.0817	0.9681	0.5573	0.859
4 Capas Conv - 400 neuronas - 100 neuronas - Dropout - batch 22	0.0623	0.9779	0.5898	0.8495

Proceso de experimentación de diferentes modelos con época de 10 todos. Estos datos son los que se dieron en la última época.

Figura E

```
modeloCNN = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(tamañoimg, tamañoimg, 1)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(128, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(256, (3, 3), activation='relu'),  # Nueva capa de convolución
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Dropout(0.1),  # Añadimos una capa de dropout para regularización
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(400, activation='relu'),
    tf.keras.layers.Dense(100, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Arquitectura electa para la etapa de prueba

Prueba y resultado

Haremos prueba si nuestro modelo está bien entrenado, y en caso de que no se tomen medidas como la propuesta en generación de datos. Tomamos las cuatro métricas para la clasificación, aunque las más relevantes para nuestro trabajo son la de Exactitud y F1. En la Tabla 2 se ve los resultados con sus métricas.

Tabla 2

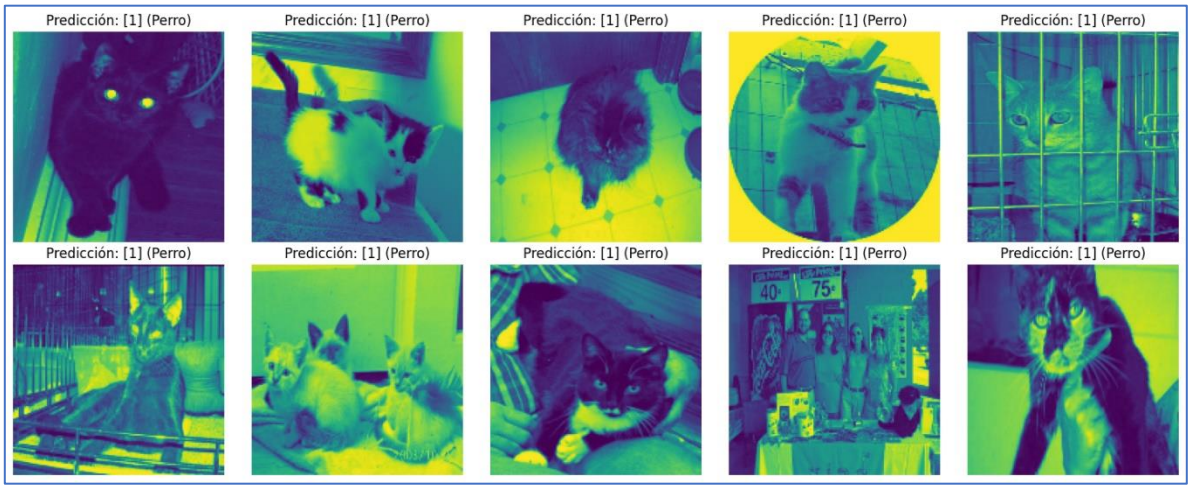
Accuracy	**0.8565**
Precision	0.8449
Recall	0.8849
F1 Score	**0.8644**

Tabla con métricas de clasificación binaria. En negrilla las métricas elegidas para nuestro trabajo.

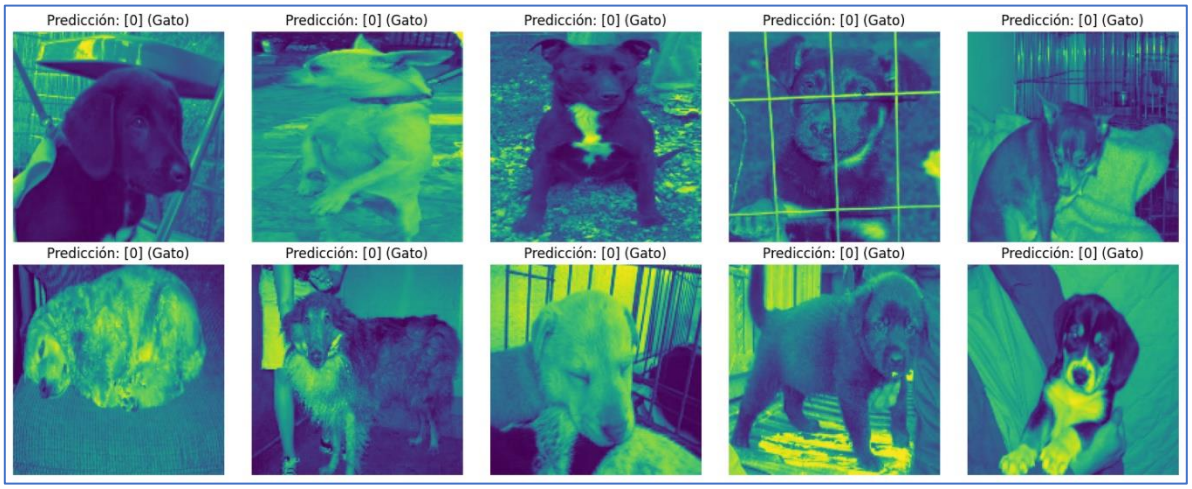
Queríamos ver el comportamiento de nuestro modelo para ver el comportamiento de la, así que aquí dejamos en la Figura F, una recolección de algunas predicciones fallidas que se clasificaron con el animal que no eran.

Figura F

- A Ejemplares que su predicción dio como perros cuando eran Gatos



- B Ejemplares que su predicción dio como gatos cuando eran perros.



Analizando estos ejemplares podemos ver el comportamiento de nuestra red resultante que está clasificando las características de un perro a veces siendo muy erróneo y otras con dificultad para un ser humano. Uno de los ejemplares un perro se acuesta como un gato y en otro ejemplar un gato esta tan lejos que es difícil de predecir correctamente.

V. CONCLUSIONES

- Dado que las métricas de exactitud y F1 son buenas y mejores a nuestro objetivo. No vimos necesario utilizar la generación de datos y el reentrenamiento de la red. Por lo que damo por exitoso este modelo.
- La capacidad del modelo supero el desafío de distinguir entre dos clases visualmente similares.
- La importancia del preprocesamiento de datos ayudó a mejorar la eficiencia del entrenamiento y la inferencia del modelo.
- Como área de trabajo futuro, se puede explorar mejores técnicas de aprendizaje profundo, como lo es el uso del aumento de datos para poder mejorar la precisión del modelo.

VI. REFERENCIAS

BEATANCUR, D. (2024). *Instituto de ingeniería del conocimiento*. Obtenido de https://www.iic.uam.es/noticias/pixeles-a-predicciones-clasificacion-de-imagenes-con-aisee/#:~:text=La%20clasificaci%C3%B3n%20de%20im%C3%A1genes%20es,entrenar%20los%20modelos%20con%20%C3%A9xito

Chollet, F. (2016). *The Keras Blog*. Obtenido de https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html

Gamco. (2021). *Gamco*. Obtenido de https://gamco.es/glosario/evaluacion-de-modelos/#:~:text=La%20evaluaci%C3%B3n%20de%20modelos%20es,determinar%20su%20precisi%C3%B3n%20y%20eficacia

Google Cloud. (s.f.). *Google Cloud*. Obtenido de https://cloud.google.com/learn/what-is-machine-learning?hl=es-419#:~:text=Descargar%20la%20gu%C3%ADa-,Definici%C3%B3n%20de%20aprendizaje%20autom%C3%A1tico,de%20grandes%20cantidades%20de%20datos

ki-ljl. (2022). *GitHub*. Obtenido de https://github.com/ki-ljl/cnn-dogs-vs-cats.git

MathWorks. (s.f.). *MathWorks*. Obtenido de https://es.mathworks.com/discovery/convolutional-neural-network.html#:~:text=Una%20red%20neuronal%20convolucional%20

TensorFlow. (Octubre de 2007). *TensorFlow Datasets*. Obtenido de https://www.tensorflow.org/datasets/catalog/cats_vs_dogs