

TP3 - SIA: Clasificación binaria entre gatos y perros

Mediante Redes Neuronales
Convolucionales (CNN)

Comisión B - Grupo 3

Integrantes:

Julia Sexe (65669)

Sofia Alfie (64244)

Isidro Perasso (65595)

Tobias Tardá (65730)

Dataset y preprocesamiento



Se usó un subconjunto de 7000 imágenes:

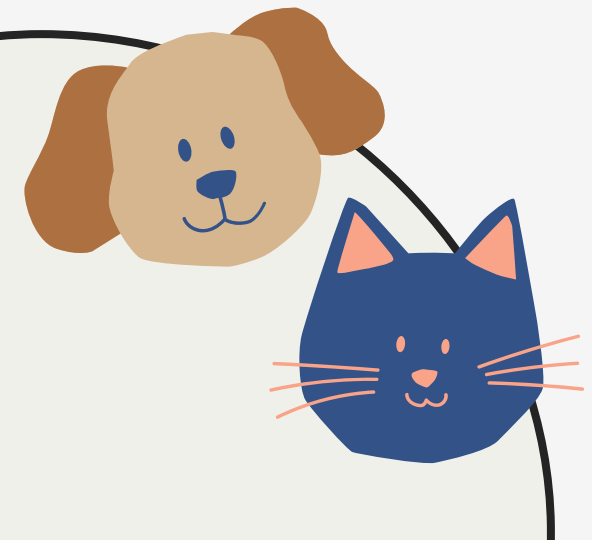
- Train: 4000
- Validation: 1000
- Test: 2000

Todas las imágenes se redimensionaron a **128×128 píxeles**, 3 canales (RGB).

Etiquetas: 0 = gato, 1 = perro.

No se aplicó one-hot encoding porque la salida del modelo es una sola neurona sigmoidea.

Se **normalizaron** los valores de píxeles de 0–255 a 0–1.



Arquitectura general de las CNN

Cada red se compone de:

Capas convolucionales (Conv2D) → detectan patrones locales (bordes, texturas).

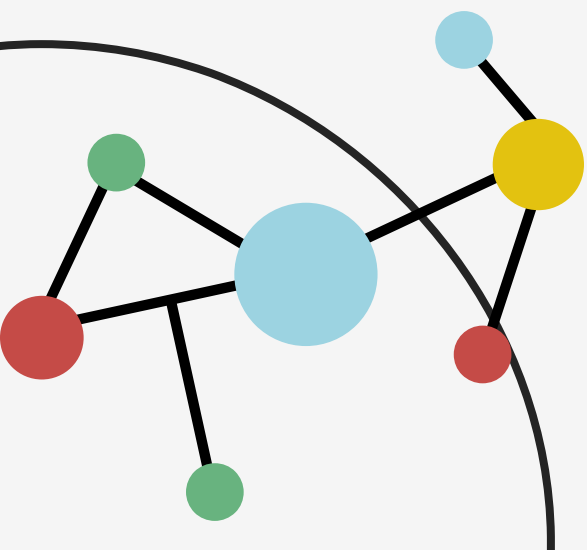
MaxPooling2D → reduce dimensiones, mantiene características importantes.

Dropout → apaga aleatoriamente neuronas durante el entrenamiento (evita overfitting).

Flatten → transforma los mapas 2D en un vector.

Capas densas (fully connected) → combinan las features extraídas.

Capa de salida → una neurona con activación sigmoid, que devuelve probabilidad de “perro”.

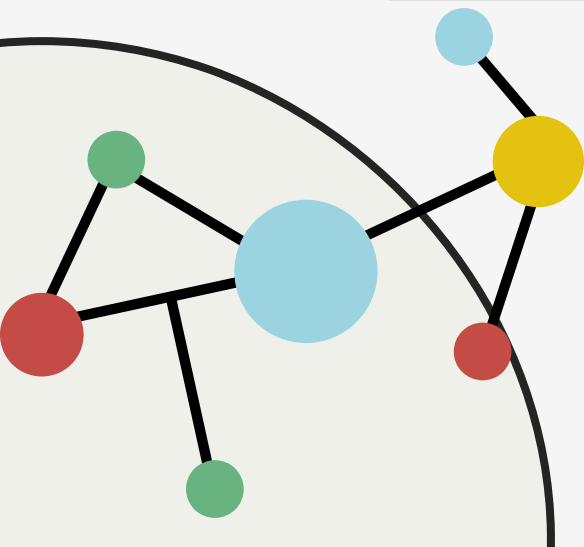


| Funciones usadas | |
|----------------------|---------------------|
| Activación interna | ReLU |
| Activación de salida | Sigmoid |
| Función de costo | Binary crossentropy |
| Optimizador | Adam |

Modelo sin optimización

Se probaron 4 redes “manuales” (Red 1 a 4) con diferentes profundidades y learning rates.

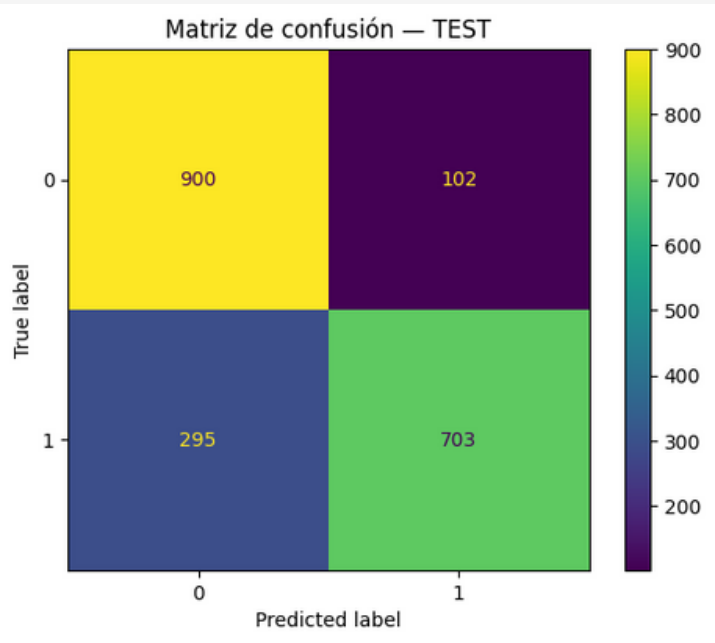
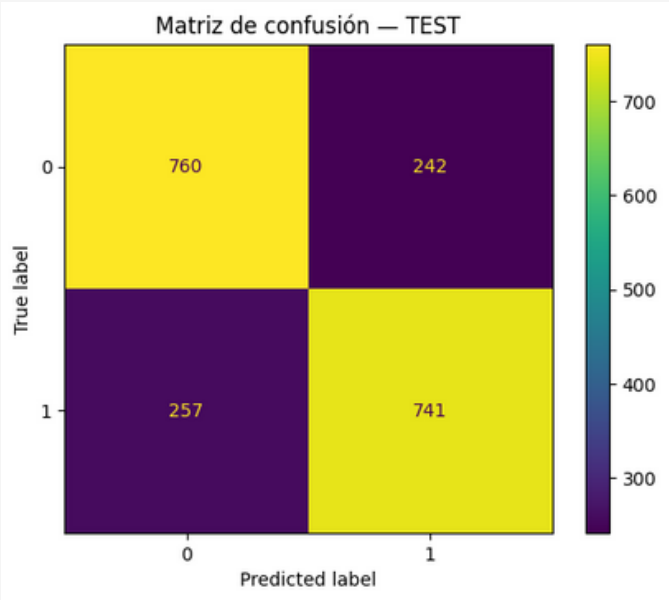
| Red | Capas conv/pool | Learning rate | Test Accuracy | Parámetros | Observaciones |
|-----|-----------------|---------------|---------------|------------|-------------------------|
| 1 | 3 | 0.001 | 72.85% | 2.19 M | Overfitting alto |
| 2 | 5 | 0.001 | 77.15% | 0.91 M | Mejor generalización |
| 3 | 5 | 0.0001 | 77.15% | 0.91 M | Aprendizaje más estable |
| 4 | 5 | 0.0005 | 81.70% | 0.91 M | Mejor desempeño final |



Modelo con optimización

Para mejorar el modelo, se usó Keras Tuner con Bayesian Optimization, que busca automáticamente los mejores hiperparámetros:

- Cantidad de filtros y tamaño de kernel.
- Tasa de dropout.
- Cantidad y tamaño de capas densas.
- Learning rate.



| Red | Estructura base | Learning rate | Val Accuracy | Test Accuracy | Parámetros | Observaciones |
|---------|-----------------|---------------|--------------|---------------|------------|---------------------|
| Opt I | 3 conv | 0.001 | 75.90% | 75.00% | 2.18 M | Mejóro poco |
| Opt II | 5 conv | 0.001 | 77.90% | 77.20% | 3.95 M | Más profunda |
| Opt III | 5 conv | 0.0005 | 79.90% | 78.40% | 0.72 M | Más liviana |
| Opt IV | 5 conv | 0.0005 | 81.00% | 80.15% | 2.20 M | Mejor balance final |

Análisis final

Las CNN logran extraer features jerárquicos: bordes → formas → objetos.

Más **capas convolucionales + regularización** (dropout) mejoran la **generalización**.

El **learning rate medio** (0.0005) fue el más estable.

Los modelos muy grandes no siempre rinden mejor; el **equilibrio entre complejidad** y datos es clave.

| Característica | Red Opt II (más compleja) | Red Opt III (más simple) |
|-----------------------|---------------------------|--------------------------|
| Capas convolucionales | 5 | 5 |
| Capas densas | 4 | 3 |
| Parámetros totales | 3.956.737 | 719.649 |
| Learning rate | 0.001 | 0.0005 |
| Val Accuracy | 77.90% | 79.90% |
| Test Accuracy | 77.20% | 78.40% |

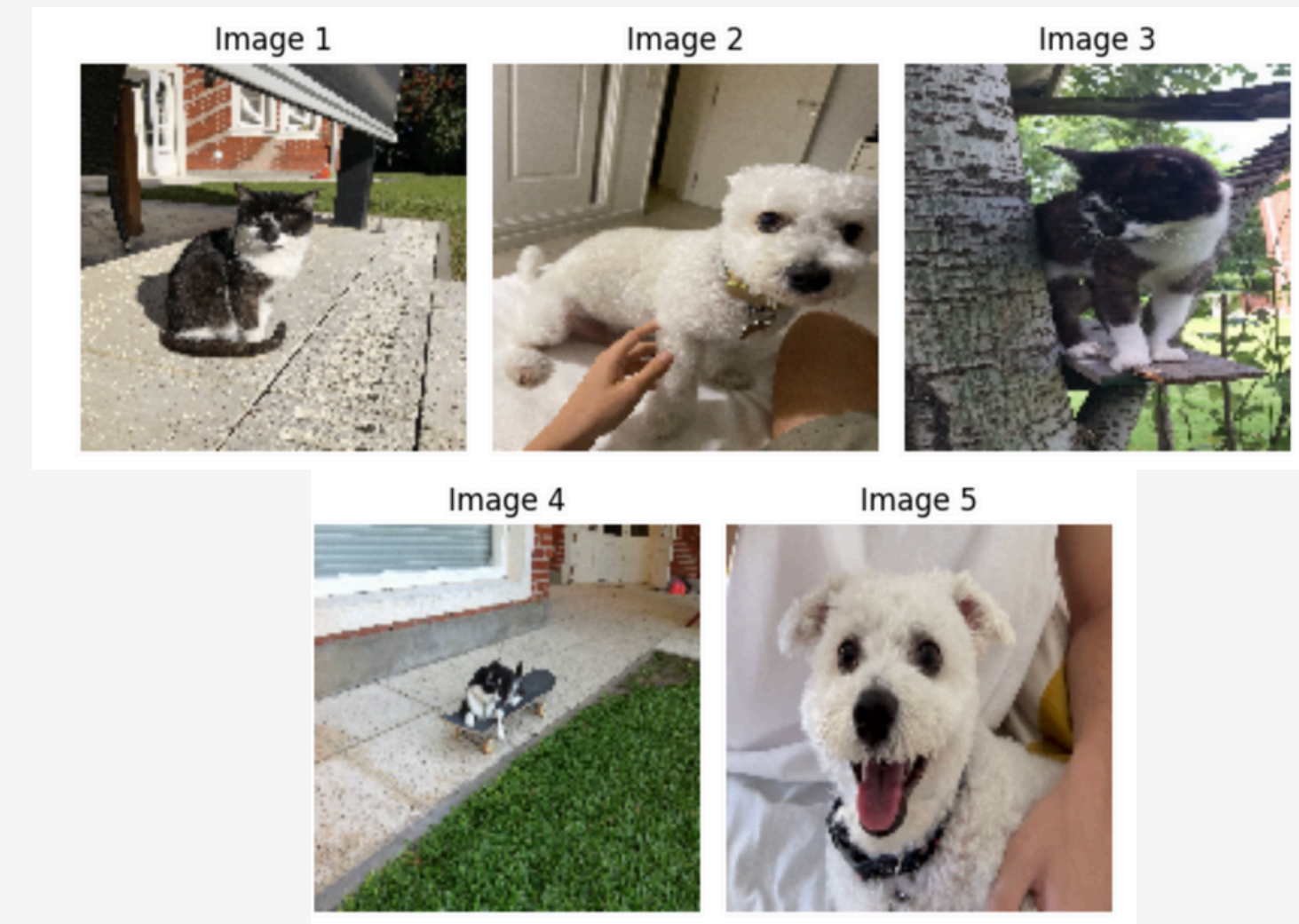
Conclusión final

Se logró una CNN con **~80% de precisión** en la clasificación binaria gatos/perros.

La optimización bayesiana permitió ajustar hiperparámetros y reducir overfitting.

El modelo final presenta buena relación entre complejidad y desempeño.

Resultados comparables con modelos de Kaggle entrenados en datasets más grandes.



Tags originales: [0, 1, 0, 0, 1]

Probabilidad de que sea un **perro** [1] : [0.35504004, 0.12580062, 0.19509245, 0.760127 , 0.96053183]

Target predicho = [0, 0, 0, 1, 1]

