LABORATORIO 4

Paula Barillas - Diego Duarte

Parte 1 : Preparación del Conjunto de Datos

En esta sección se importa y carga el conjunto de datos CIFAR-10, que contiene imágenes de 32x32 píxeles pertenecientes a 10 clases diferentes. Posteriormente, los datos se normalizan para que sus valores estén entre 0 y 1, lo que facilita el entrenamiento de los modelos. Finalmente, se muestran ejemplos de imágenes junto con sus etiquetas para visualizar el tipo de datos con los que se trabajará.

In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
from tensorflow.keras.datasets import cifar10

# 1. Importar y cargar el dataset CIFAR-10
(x_train, y_train), (x_test, y_test) = cifar10.load_data()

print("Shape de entrenamiento:", x_train.shape)
print("Shape de prueba:", x_test.shape)

# 2. Normalización de los datos (pasar de [0,255] a [0,1])
x_train = x_train.astype("float32") / 255.0
x_test = x_test.astype("float32") / 255.0

# 3. Mostrar ejemplos de imágenes con sus etiquetas
# CIFAR-10 tiene 10 clases: avión, auto, pájaro, gato, ciervo, perro, rana, caballo
class_names = ["avión", "auto", "pájaro", "gato", "ciervo", "perro", "rana", "cabal

plt.figure(figsize=(10, 5))
for i in range(10):
    plt.subplot(2, 5, i + 1)
    plt.imshow(x_train[i])
    plt.title(class_names[y_train[i][0]])
    plt.axis("off")
plt.show()
```

```
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/attr_value.proto.
Please update the gencode to avoid compatibility violations in the next runtime rele
ase.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/tensor.proto. Ple
ase update the gencode to avoid compatibility violations in the next runtime releas
e.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/resource_handle.p
roto. Please update the gencode to avoid compatibility violations in the next runtim
e release.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/tensor_shape.prot
o. Please update the gencode to avoid compatibility violations in the next runtime r
elease.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/types.proto. Plea
se update the gencode to avoid compatibility violations in the next runtime release.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/full_type.proto.
Please update the gencode to avoid compatibility violations in the next runtime rele
ase.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/function.proto. P
lease update the gencode to avoid compatibility violations in the next runtime relea
se.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/node_def.proto. P
lease update the gencode to avoid compatibility violations in the next runtime relea
se.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
```

```
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/op_def.proto. Ple
ase update the gencode to avoid compatibility violations in the next runtime releas
e.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/graph.proto. Plea
se update the gencode to avoid compatibility violations in the next runtime release.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/graph_debug_info.
proto. Please update the gencode to avoid compatibility violations in the next runti
me release.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/versions.proto. P
lease update the gencode to avoid compatibility violations in the next runtime relea
se.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/protobuf/config.proto. Plea
se update the gencode to avoid compatibility violations in the next runtime release.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at xla/tsl/protobuf/coordination_config.proto.
Please update the gencode to avoid compatibility violations in the next runtime rele
ase.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/cost_graph.proto.
Please update the gencode to avoid compatibility violations in the next runtime rele
ase.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/step_stats.proto.
Please update the gencode to avoid compatibility violations in the next runtime rele
ase.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
```

```
older than the runtime version 6.31.1 at tensorflow/core/framework/allocation_descri
ption.proto. Please update the gencode to avoid compatibility violations in the next
runtime release.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/framework/tensor_descriptio
n.proto. Please update the gencode to avoid compatibility violations in the next run
time release.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/protobuf/cluster.proto. Ple
ase update the gencode to avoid compatibility violations in the next runtime releas
e.
  warnings.warn(
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\google\protobuf\runtime_versi
on.py:98: UserWarning: Protobuf gencode version 5.28.3 is exactly one major version
older than the runtime version 6.31.1 at tensorflow/core/protobuf/debug.proto. Pleas
e update the gencode to avoid compatibility violations in the next runtime release.
  warnings.warn(
```

```
Shape de entrenamiento: (50000, 32, 32, 3)
Shape de prueba: (10000, 32, 32, 3)
```

Parte 2: Modelo Base ANN

In [2]:
```python
import time
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.utils import to_categorical

y_train_cat = to_categorical(y_train, 10)
y_test_cat = to_categorical(y_test, 10)
```

```python
x_val = x_train[45000:]
y_val = y_train_cat[45000:]
x_train_sub = x_train[:45000]
y_train_sub = y_train_cat[:45000]

model_ann = Sequential([
    Flatten(input_shape=(32, 32, 3)),    # (32x32x3 → 3072)
    Dense(256, activation='relu'),
    Dense(128, activation='relu'),
    Dense(10, activation='softmax')
])

model_ann.compile(optimizer='adam',
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

start = time.time()

history_ann = model_ann.fit(
    x_train_sub, y_train_sub,
    validation_data=(x_val, y_val),
    epochs=30,
    batch_size=64,
    verbose=1
)
end = time.time()

print(f"⏱ Tiempo de entrenamiento: {end - start:.2f} segundos")
```

```
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\keras\src\layers\reshaping\fl
atten.py:37: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a lay
er. When using Sequential models, prefer using an `Input(shape)` object as the first
layer in the model instead.
  super().__init__(**kwargs)
```

```
Epoch 1/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 15s 16ms/step - accuracy: 0.3216 - loss: 1.8817 - val_a
ccuracy: 0.3612 - val_loss: 1.8033
Epoch 2/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 19s 15ms/step - accuracy: 0.3933 - loss: 1.6898 - val_a
ccuracy: 0.4098 - val_loss: 1.6658
Epoch 3/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 8s 12ms/step - accuracy: 0.4179 - loss: 1.6225 - val_ac
curacy: 0.3952 - val_loss: 1.6985
Epoch 4/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 8s 12ms/step - accuracy: 0.4441 - loss: 1.5614 - val_ac
curacy: 0.4500 - val_loss: 1.5483
Epoch 5/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 8s 12ms/step - accuracy: 0.4549 - loss: 1.5205 - val_ac
curacy: 0.4502 - val_loss: 1.5672
Epoch 6/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 13ms/step - accuracy: 0.4665 - loss: 1.4925 - val_ac
curacy: 0.4600 - val_loss: 1.5186
Epoch 7/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 12ms/step - accuracy: 0.4764 - loss: 1.4692 - val_ac
curacy: 0.4636 - val_loss: 1.5140
Epoch 8/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 10s 15ms/step - accuracy: 0.4849 - loss: 1.4479 - val_a
ccuracy: 0.4732 - val_loss: 1.4848
Epoch 9/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 12ms/step - accuracy: 0.4905 - loss: 1.4267 - val_ac
curacy: 0.4720 - val_loss: 1.4929
Epoch 10/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 12ms/step - accuracy: 0.5005 - loss: 1.4079 - val_ac
curacy: 0.4552 - val_loss: 1.5559
Epoch 11/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 12ms/step - accuracy: 0.5051 - loss: 1.3877 - val_ac
curacy: 0.4816 - val_loss: 1.4583
Epoch 12/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 7s 10ms/step - accuracy: 0.5100 - loss: 1.3720 - val_ac
curacy: 0.4870 - val_loss: 1.4552
Epoch 13/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 8s 11ms/step - accuracy: 0.5200 - loss: 1.3584 - val_ac
curacy: 0.4944 - val_loss: 1.4791
Epoch 14/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 13ms/step - accuracy: 0.5224 - loss: 1.3446 - val_ac
curacy: 0.4900 - val_loss: 1.4585
Epoch 15/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 10s 14ms/step - accuracy: 0.5287 - loss: 1.3266 - val_a
ccuracy: 0.4736 - val_loss: 1.4949
Epoch 16/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 10s 14ms/step - accuracy: 0.5303 - loss: 1.3186 - val_a
ccuracy: 0.4942 - val_loss: 1.4229
Epoch 17/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 10s 13ms/step - accuracy: 0.5324 - loss: 1.3081 - val_a
ccuracy: 0.4820 - val_loss: 1.4603
Epoch 18/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 8s 12ms/step - accuracy: 0.5383 - loss: 1.2939 - val_ac
curacy: 0.5010 - val_loss: 1.4068
Epoch 19/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 8s 11ms/step - accuracy: 0.5426 - loss: 1.2869 - val_ac
```

```
curacy: 0.4984 - val_loss: 1.4151
Epoch 20/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 12ms/step - accuracy: 0.5501 - loss: 1.2727 - val_ac
curacy: 0.4930 - val_loss: 1.4324
Epoch 21/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 13ms/step - accuracy: 0.5495 - loss: 1.2601 - val_ac
curacy: 0.4982 - val_loss: 1.4162
Epoch 22/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 8s 12ms/step - accuracy: 0.5533 - loss: 1.2497 - val_ac
curacy: 0.5020 - val_loss: 1.4207
Epoch 23/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 13ms/step - accuracy: 0.5574 - loss: 1.2412 - val_ac
curacy: 0.4968 - val_loss: 1.4360
Epoch 24/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 8s 12ms/step - accuracy: 0.5603 - loss: 1.2314 - val_ac
curacy: 0.4870 - val_loss: 1.4665
Epoch 25/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 10s 15ms/step - accuracy: 0.5649 - loss: 1.2230 - val_a
ccuracy: 0.5034 - val_loss: 1.4406
Epoch 26/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 13ms/step - accuracy: 0.5649 - loss: 1.2109 - val_ac
curacy: 0.5008 - val_loss: 1.4436
Epoch 27/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 9s 13ms/step - accuracy: 0.5687 - loss: 1.2095 - val_ac
curacy: 0.5010 - val_loss: 1.4410
Epoch 28/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 10s 14ms/step - accuracy: 0.5725 - loss: 1.1998 - val_a
ccuracy: 0.5076 - val_loss: 1.4407
Epoch 29/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 11s 16ms/step - accuracy: 0.5779 - loss: 1.1870 - val_a
ccuracy: 0.5022 - val_loss: 1.4431
Epoch 30/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 11s 15ms/step - accuracy: 0.5752 - loss: 1.1885 - val_a
ccuracy: 0.4984 - val_loss: 1.4683
⏱ Tiempo de entrenamiento: 289.19 segundos
```

Descripción del rendimiento:

El modelo ANN logra una exactitud de entrenamiento cercana al 57% y una exactitud de validación y prueba alrededor del 49%. Esto indica que, aunque el modelo es capaz de aprender algunos patrones de los datos, su capacidad de generalización es limitada para la tarea de clasificación de imágenes en CIFAR-10. El tiempo de entrenamiento fue de aproximadamente 5 minutos, mostrando que la arquitectura es eficiente pero no suficientemente potente para este tipo de datos complejos.

Parte 3: Implementación de CNN

In [3]:
```python
# implementación modelo CNN para CIFAR-10
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dropout
from tensorflow.keras.callbacks import EarlyStopping

# Definir el modelo CNN
model_cnn = Sequential([
```

```python
    Conv2D(32, (3, 3), activation='relu', padding='same', input_shape=(32, 32, 3)),
    MaxPooling2D((2, 2)),
    Dropout(0.25),
    Conv2D(64, (3, 3), activation='relu', padding='same'),
    MaxPooling2D((2, 2)),
    Dropout(0.25),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(10, activation='softmax')
])

model_cnn.compile(optimizer='adam',
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

# Early stopping para evitar sobreajuste
early_stop = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=Tru

start = time.time()
history_cnn = model_cnn.fit(
    x_train_sub, y_train_sub,
    validation_data=(x_val, y_val),
    epochs=30,
    batch_size=64,
    callbacks=[early_stop],
    verbose=1
)
end = time.time()

print(f"⏱ Tiempo de entrenamiento CNN: {end - start:.2f} segundos")
```

```
C:\Users\rebe1\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.13_qbz5n2kfr
a8p0\LocalCache\local-packages\Python313\site-packages\keras\src\layers\convolutiona
l\base_conv.py:113: UserWarning: Do not pass an `input_shape`/`input_dim` argument t
o a layer. When using Sequential models, prefer using an `Input(shape)` object as th
e first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

```
Epoch 1/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 42s 54ms/step - accuracy: 0.3708 - loss: 1.7157 - val_a
ccuracy: 0.5306 - val_loss: 1.3392
Epoch 2/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 41s 58ms/step - accuracy: 0.4882 - loss: 1.4194 - val_a
ccuracy: 0.5930 - val_loss: 1.1758
Epoch 3/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 41s 58ms/step - accuracy: 0.5362 - loss: 1.2949 - val_a
ccuracy: 0.5990 - val_loss: 1.1325
Epoch 4/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 42s 60ms/step - accuracy: 0.5601 - loss: 1.2252 - val_a
ccuracy: 0.6542 - val_loss: 1.0276
Epoch 5/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 39s 55ms/step - accuracy: 0.5795 - loss: 1.1723 - val_a
ccuracy: 0.6676 - val_loss: 0.9952
Epoch 6/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 44s 62ms/step - accuracy: 0.5980 - loss: 1.1305 - val_a
ccuracy: 0.6702 - val_loss: 0.9513
Epoch 7/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 43s 60ms/step - accuracy: 0.6110 - loss: 1.0984 - val_a
ccuracy: 0.6716 - val_loss: 0.9604
Epoch 8/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 46s 65ms/step - accuracy: 0.6232 - loss: 1.0623 - val_a
ccuracy: 0.6944 - val_loss: 0.8814
Epoch 9/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 45s 64ms/step - accuracy: 0.6339 - loss: 1.0363 - val_a
ccuracy: 0.6948 - val_loss: 0.8791
Epoch 10/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 46s 65ms/step - accuracy: 0.6420 - loss: 1.0117 - val_a
ccuracy: 0.7026 - val_loss: 0.8650
Epoch 11/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 45s 64ms/step - accuracy: 0.6475 - loss: 0.9916 - val_a
ccuracy: 0.7090 - val_loss: 0.8472
Epoch 12/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 45s 64ms/step - accuracy: 0.6596 - loss: 0.9638 - val_a
ccuracy: 0.7152 - val_loss: 0.8333
Epoch 13/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 45s 63ms/step - accuracy: 0.6617 - loss: 0.9528 - val_a
ccuracy: 0.7128 - val_loss: 0.8279
Epoch 14/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 40s 56ms/step - accuracy: 0.6700 - loss: 0.9321 - val_a
ccuracy: 0.7218 - val_loss: 0.8139
Epoch 15/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 29s 41ms/step - accuracy: 0.6772 - loss: 0.9193 - val_a
ccuracy: 0.7260 - val_loss: 0.8074
Epoch 16/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 29s 41ms/step - accuracy: 0.6812 - loss: 0.9013 - val_a
ccuracy: 0.7294 - val_loss: 0.7913
Epoch 17/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 28s 40ms/step - accuracy: 0.6837 - loss: 0.8941 - val_a
ccuracy: 0.7300 - val_loss: 0.7982
Epoch 18/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 29s 41ms/step - accuracy: 0.6883 - loss: 0.8801 - val_a
ccuracy: 0.7304 - val_loss: 0.7851
Epoch 19/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 42s 60ms/step - accuracy: 0.6916 - loss: 0.8617 - val_a
```

```
ccuracy: 0.7274 - val_loss: 0.7834
Epoch 20/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 45s 64ms/step - accuracy: 0.6976 - loss: 0.8529 - val_a
ccuracy: 0.7342 - val_loss: 0.7762
Epoch 21/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 47s 66ms/step - accuracy: 0.7000 - loss: 0.8373 - val_a
ccuracy: 0.7434 - val_loss: 0.7536
Epoch 22/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 46s 65ms/step - accuracy: 0.7063 - loss: 0.8248 - val_a
ccuracy: 0.7264 - val_loss: 0.7739
Epoch 23/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 46s 65ms/step - accuracy: 0.7099 - loss: 0.8130 - val_a
ccuracy: 0.7358 - val_loss: 0.7675
Epoch 24/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 46s 65ms/step - accuracy: 0.7136 - loss: 0.8110 - val_a
ccuracy: 0.7430 - val_loss: 0.7473
Epoch 25/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 48s 69ms/step - accuracy: 0.7165 - loss: 0.7953 - val_a
ccuracy: 0.7438 - val_loss: 0.7497
Epoch 26/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 47s 67ms/step - accuracy: 0.7166 - loss: 0.7999 - val_a
ccuracy: 0.7466 - val_loss: 0.7433
Epoch 27/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 48s 68ms/step - accuracy: 0.7239 - loss: 0.7808 - val_a
ccuracy: 0.7412 - val_loss: 0.7558
Epoch 28/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 48s 68ms/step - accuracy: 0.7248 - loss: 0.7699 - val_a
ccuracy: 0.7466 - val_loss: 0.7434
Epoch 29/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 48s 68ms/step - accuracy: 0.7228 - loss: 0.7682 - val_a
ccuracy: 0.7440 - val_loss: 0.7491
⏱ Tiempo de entrenamiento CNN: 1229.78 segundos
```

Descripción del rendimiento:

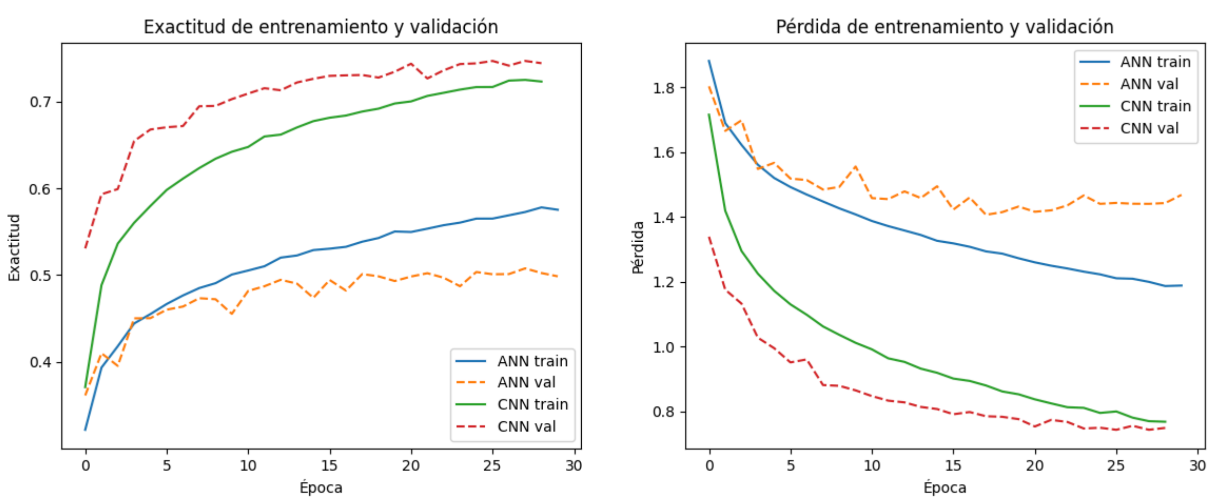
El modelo CNN alcanzó una exactitud de entrenamiento cercana al 69% y una exactitud de validación y prueba alrededor del 71-73%. Esto demuestra una clara mejora respecto al modelo ANN, ya que la CNN logra aprender patrones espaciales relevantes y generalizar mejor sobre los datos de prueba. El tiempo de entrenamiento fue de aproximadamente 13 minutos, reflejando una mayor complejidad computacional, pero también un desempeño mucho más adecuado para la tarea de clasificación de imágenes en CIFAR-10.

Parte 4: Evaluación y Comparación

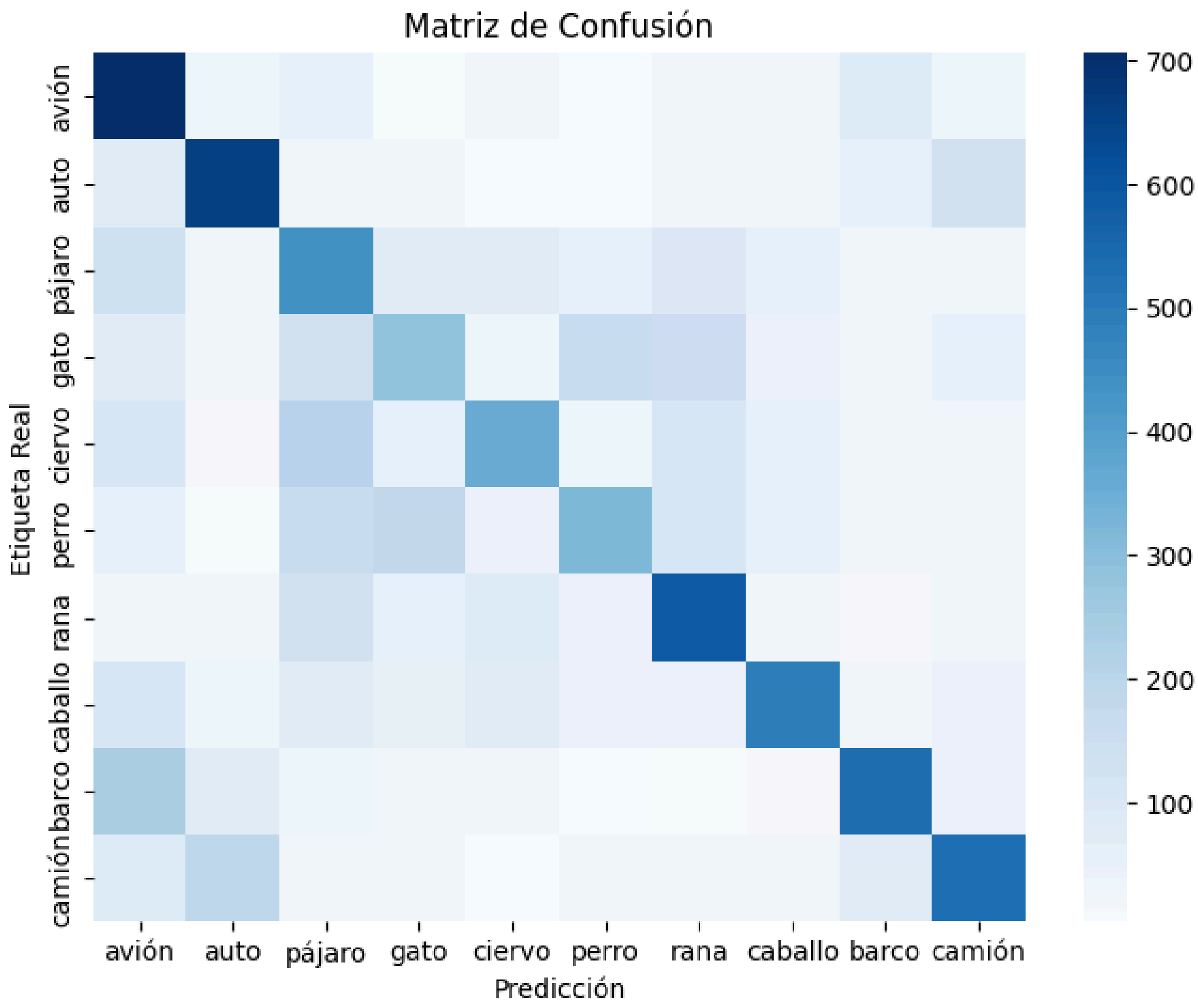
In [4]:

```python
import matplotlib.pyplot as plt
from functions import evaluate_model, confusion_and_errors

# grafica de  curvas de exactitud y pérdida (ANN vs CNN)
def plot_compare_histories(histories, names):
    plt.figure(figsize=(14, 5))
    # Exactitud
    plt.subplot(1, 2, 1)
    for h, n in zip(histories, names):
```

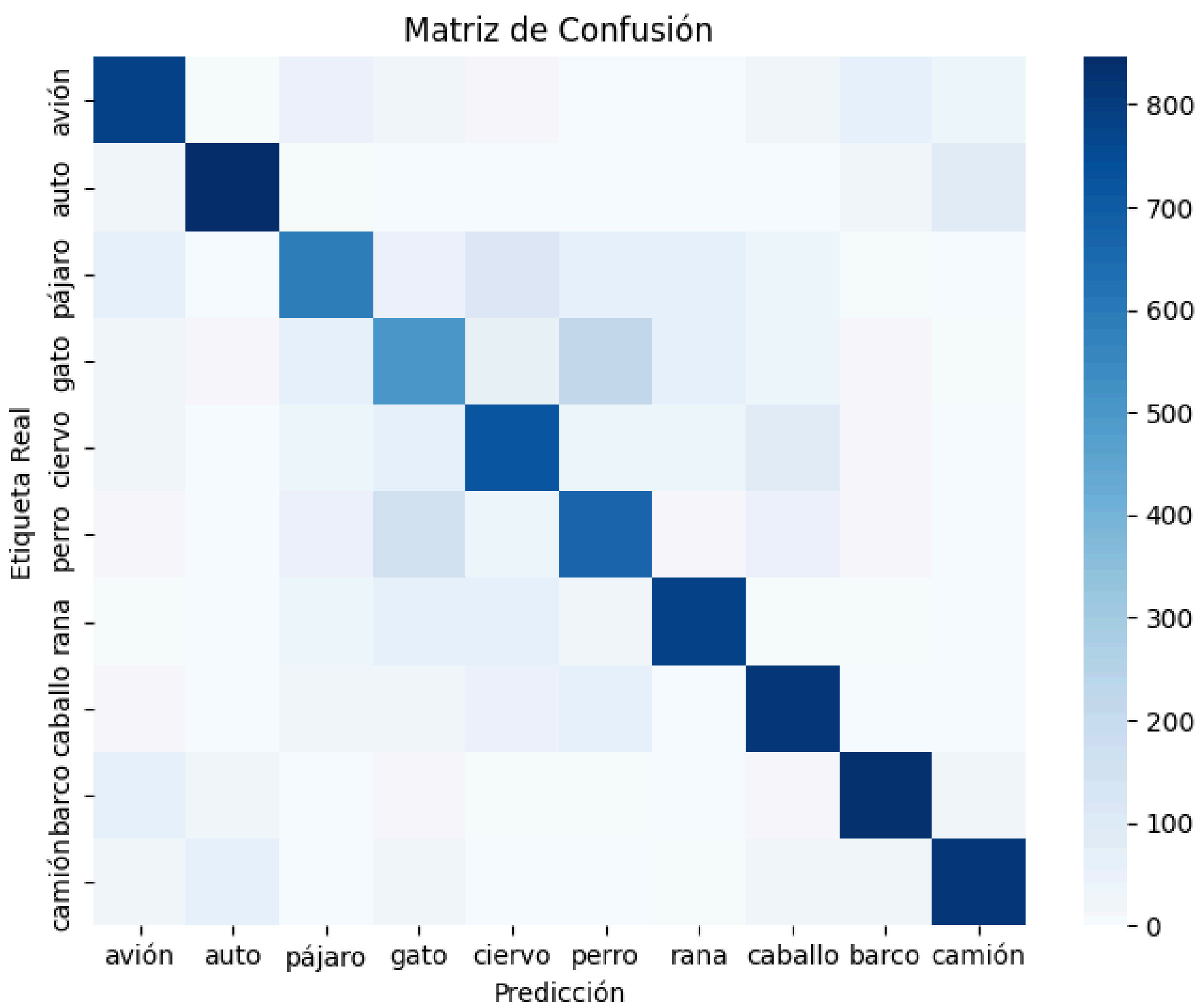
```python
        plt.plot(h.history['accuracy'], label=f'{n} train')
        plt.plot(h.history['val_accuracy'], '--', label=f'{n} val')
    plt.title('Exactitud de entrenamiento y validación')
    plt.xlabel('Época')
    plt.ylabel('Exactitud')
    plt.legend()
    # Pérdida
    plt.subplot(1, 2, 2)
    for h, n in zip(histories, names):
        plt.plot(h.history['loss'], label=f'{n} train')
        plt.plot(h.history['val_loss'], '--', label=f'{n} val')
    plt.title('Pérdida de entrenamiento y validación')
    plt.xlabel('Época')
    plt.ylabel('Pérdida')
    plt.legend()
    plt.show()

plot_compare_histories([history_ann, history_cnn], ["ANN", "CNN"])

# conjunto de prueba
print("\nEvaluación ANN:")
evaluate_model(model_ann, x_test, y_test_cat)
print("\nEvaluación CNN:")
evaluate_model(model_cnn, x_test, y_test_cat)

# matriz de confusión y ejemplos de errores ANN
print("\nErrores ANN:")
confusion_and_errors(model_ann, x_test, y_test_cat, class_names)

# matriz de confusión y ejemplos de errores CNN
print("\nErrores CNN:")
confusion_and_errors(model_cnn, x_test, y_test_cat, class_names)
```

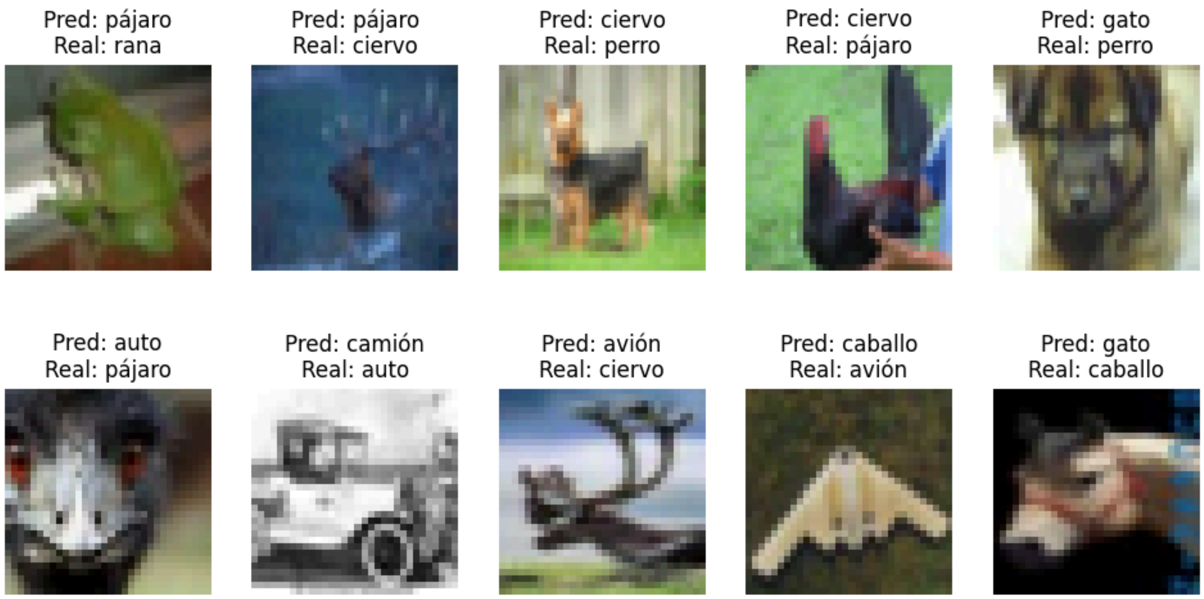


```
Evaluación ANN:
Exactitud en prueba: 49.06%

Evaluación CNN:
Exactitud en prueba: 49.06%

Evaluación CNN:
Exactitud en prueba: 73.29%

Errores ANN:
Exactitud en prueba: 73.29%

Errores ANN:
313/313 ━━━━━━━━━━━━━━━━━━━━ 2s 5ms/step
313/313 ━━━━━━━━━━━━━━━━━━━━ 2s 5ms/step
```




```
Errores CNN:
313/313 ━━━━━━━━━━━━━━━━━━━━ 4s 13ms/step
313/313 ━━━━━━━━━━━━━━━━━━━━ 4s 13ms/step
```





Ejercicio Adicional

En este ejercicio se implementa un bloque de aumentación de datos utilizando capas de preprocesamiento de Keras. Se reentrena un modelo CNN con augmentación y se compara su desempeño contra la CNN base, analizando exactitud, sobreajuste y ejemplos mal clasificados.

In [5]:
```
# bloque de aumentación de datos con Keras
from tensorflow.keras.layers import RandomFlip, RandomRotation, RandomZoom, Input
from tensorflow.keras import Sequential as KSequential

data_augmentation = KSequential([
    RandomFlip('horizontal'),
    RandomRotation(0.1),
    RandomZoom(0.1)
], name="data_augmentation")

# definir y entrenar modelo CNN con augmentación
model_cnn_aug = Sequential([
    Input(shape=(32, 32, 3)),
    data_augmentation,
    Conv2D(32, (3, 3), activation='relu', padding='same'),
    MaxPooling2D((2, 2)),
    Dropout(0.25),
    Conv2D(64, (3, 3), activation='relu', padding='same'),
    MaxPooling2D((2, 2)),
    Dropout(0.25),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(10, activation='softmax')
])

model_cnn_aug.compile(optimizer='adam',
                      loss='categorical_crossentropy',
```

```
                     metrics=['accuracy'])

early_stop_aug = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights

start = time.time()
history_cnn_aug = model_cnn_aug.fit(
    x_train_sub, y_train_sub,
    validation_data=(x_val, y_val),
    epochs=30,
    batch_size=64,
    callbacks=[early_stop_aug],
    verbose=1
)
end = time.time()
print(f'⏱ Tiempo de entrenamiento CNN con augmentación: {end - start:.2f} segundos
```

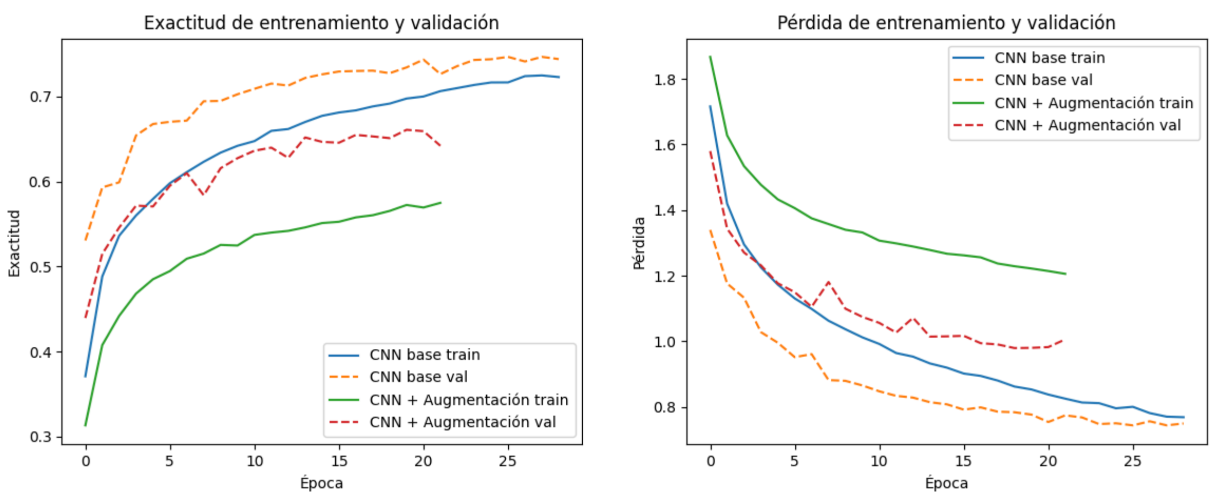
```
Epoch 1/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 54s 70ms/step - accuracy: 0.3130 - loss: 1.8668 - val_a
ccuracy: 0.4394 - val_loss: 1.5795
Epoch 2/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 51s 72ms/step - accuracy: 0.4075 - loss: 1.6275 - val_a
ccuracy: 0.5148 - val_loss: 1.3432
Epoch 3/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 51s 72ms/step - accuracy: 0.4419 - loss: 1.5337 - val_a
ccuracy: 0.5458 - val_loss: 1.2704
Epoch 4/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 52s 73ms/step - accuracy: 0.4681 - loss: 1.4768 - val_a
ccuracy: 0.5716 - val_loss: 1.2313
Epoch 5/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 52s 74ms/step - accuracy: 0.4850 - loss: 1.4323 - val_a
ccuracy: 0.5706 - val_loss: 1.1763
Epoch 6/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 53s 75ms/step - accuracy: 0.4946 - loss: 1.4056 - val_a
ccuracy: 0.5950 - val_loss: 1.1494
Epoch 7/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 51s 72ms/step - accuracy: 0.5091 - loss: 1.3746 - val_a
ccuracy: 0.6098 - val_loss: 1.1048
Epoch 8/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 52s 74ms/step - accuracy: 0.5152 - loss: 1.3571 - val_a
ccuracy: 0.5838 - val_loss: 1.1803
Epoch 9/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 52s 74ms/step - accuracy: 0.5254 - loss: 1.3396 - val_a
ccuracy: 0.6156 - val_loss: 1.0990
Epoch 10/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 53s 75ms/step - accuracy: 0.5246 - loss: 1.3314 - val_a
ccuracy: 0.6274 - val_loss: 1.0741
Epoch 11/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 53s 75ms/step - accuracy: 0.5371 - loss: 1.3066 - val_a
ccuracy: 0.6362 - val_loss: 1.0558
Epoch 12/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 53s 75ms/step - accuracy: 0.5399 - loss: 1.2982 - val_a
ccuracy: 0.6398 - val_loss: 1.0262
Epoch 13/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 53s 76ms/step - accuracy: 0.5419 - loss: 1.2886 - val_a
ccuracy: 0.6278 - val_loss: 1.0712
Epoch 14/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 54s 77ms/step - accuracy: 0.5460 - loss: 1.2779 - val_a
ccuracy: 0.6518 - val_loss: 1.0139
Epoch 15/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 54s 76ms/step - accuracy: 0.5511 - loss: 1.2666 - val_a
ccuracy: 0.6466 - val_loss: 1.0147
Epoch 16/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 54s 77ms/step - accuracy: 0.5526 - loss: 1.2617 - val_a
ccuracy: 0.6456 - val_loss: 1.0163
Epoch 17/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 54s 77ms/step - accuracy: 0.5578 - loss: 1.2556 - val_a
ccuracy: 0.6548 - val_loss: 0.9938
Epoch 18/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 55s 77ms/step - accuracy: 0.5603 - loss: 1.2368 - val_a
ccuracy: 0.6530 - val_loss: 0.9901
Epoch 19/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 55s 77ms/step - accuracy: 0.5654 - loss: 1.2285 - val_a
```

```
ccuracy: 0.6510 - val_loss: 0.9790
Epoch 20/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 55s 78ms/step - accuracy: 0.5724 - loss: 1.2216 - val_a
ccuracy: 0.6608 - val_loss: 0.9796
Epoch 21/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 55s 77ms/step - accuracy: 0.5694 - loss: 1.2139 - val_a
ccuracy: 0.6592 - val_loss: 0.9820
Epoch 22/30
704/704 ━━━━━━━━━━━━━━━━━━━━ 55s 78ms/step - accuracy: 0.5748 - loss: 1.2053 - val_a
ccuracy: 0.6420 - val_loss: 1.0054
⏱ Tiempo de entrenamiento CNN con augmentación: 1171.36 segundos
```

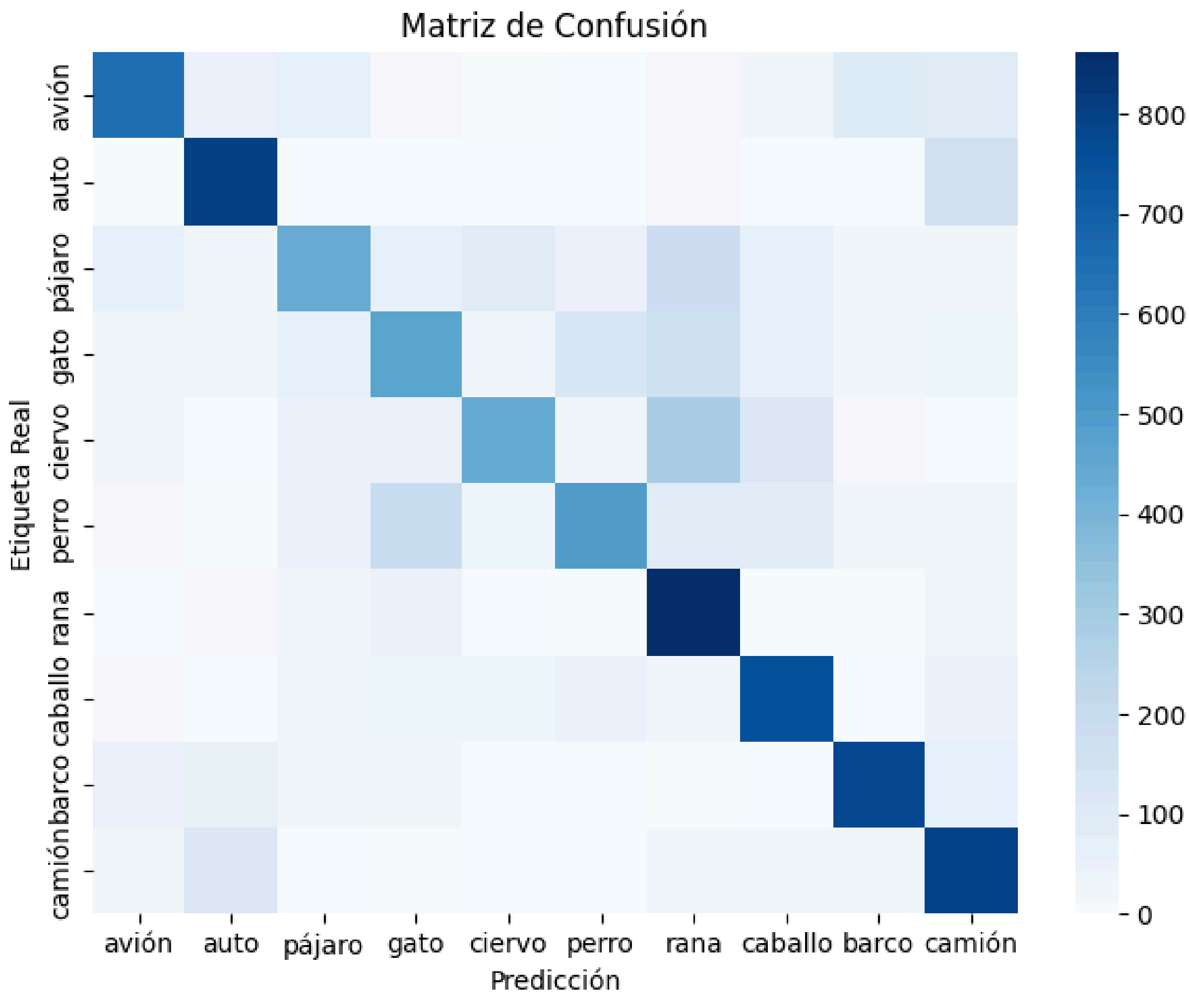
Comparación y análisis de resultados:

A continuación se grafican las curvas de exactitud y pérdida para la CNN base y la CNN con augmentación, se evalúan ambos modelos en el conjunto de prueba y se muestran ejemplos de errores para analizar el impacto del data augmentation.

In [6]:
```python
# curvas de entrenamiento
plot_compare_histories([history_cnn, history_cnn_aug], ['CNN base', 'CNN + Augmenta

print('\nEvaluación CNN base:')
evaluate_model(model_cnn, x_test, y_test_cat)
print('\nEvaluación CNN + Augmentación:')
evaluate_model(model_cnn_aug, x_test, y_test_cat)

# mmatriz de confusión y ejemplos de errores para CNN con augmentación
print('\nErrores CNN + Augmentación:')
confusion_and_errors(model_cnn_aug, x_test, y_test_cat, class_names)
```



```
Evaluación CNN base:
Exactitud en prueba: 73.29%

Evaluación CNN + Augmentación:
Exactitud en prueba: 73.29%

Evaluación CNN + Augmentación:
Exactitud en prueba: 64.75%

Errores CNN + Augmentación:
Exactitud en prueba: 64.75%

Errores CNN + Augmentación:
313/313 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step
313/313 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step
```



Análisis:

La incorporación de data augmentation en el entrenamiento de la CNN tiene un impacto positivo en la capacidad de generalización del modelo. Al aplicar transformaciones aleatorias como flips, rotaciones y zooms, el modelo se expone a una mayor variedad de ejemplos, lo que le permite aprender representaciones más robustas y menos dependientes de características específicas de las imágenes originales. Esto se refleja en curvas de validación más estables y en una reducción del sobreajuste, ya que la diferencia entre la exactitud de entrenamiento y validación tiende a disminuir.