CLASIFICACIÓN DE MAMIFEROS PEQUEÑOS

Integrantes:

- Alexis Susanibar Rosas
- César Rico Otero
- Leandro Lazo La Rosa

01 Objetivo

Clasificación de imágenes

Se necesita clasificar mamíferos de la orden Roentia de distintas especies y de esta manera identificar la especie a la cual pertenece.

Hamster

Ratón

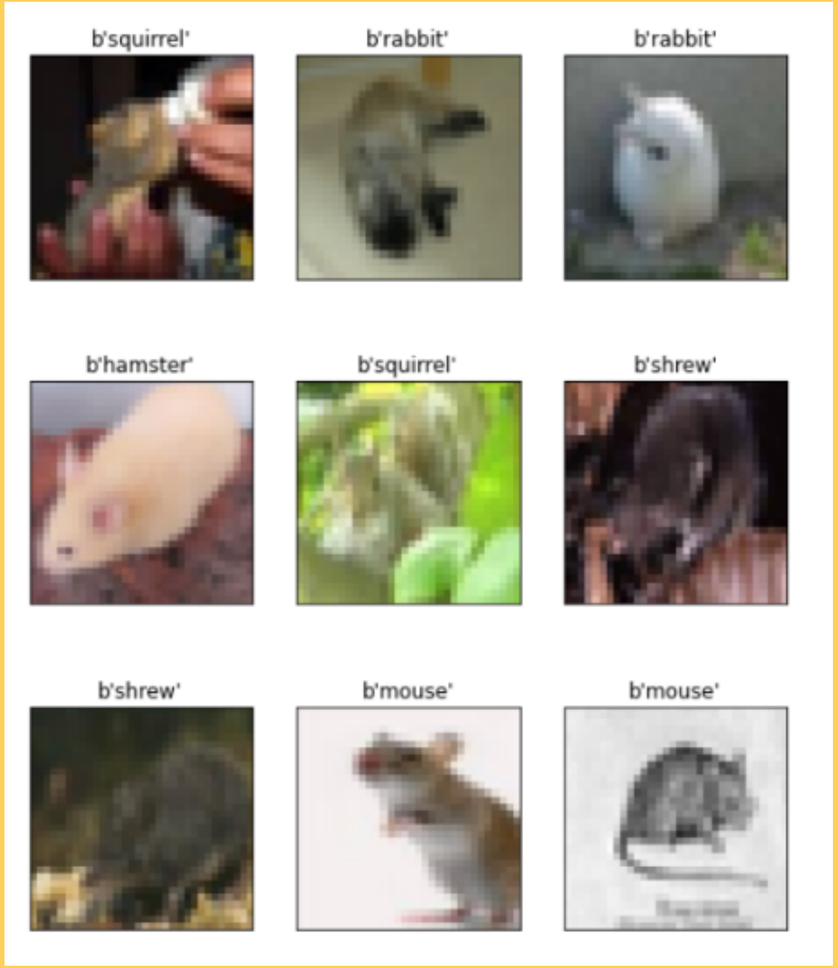
Musaraña

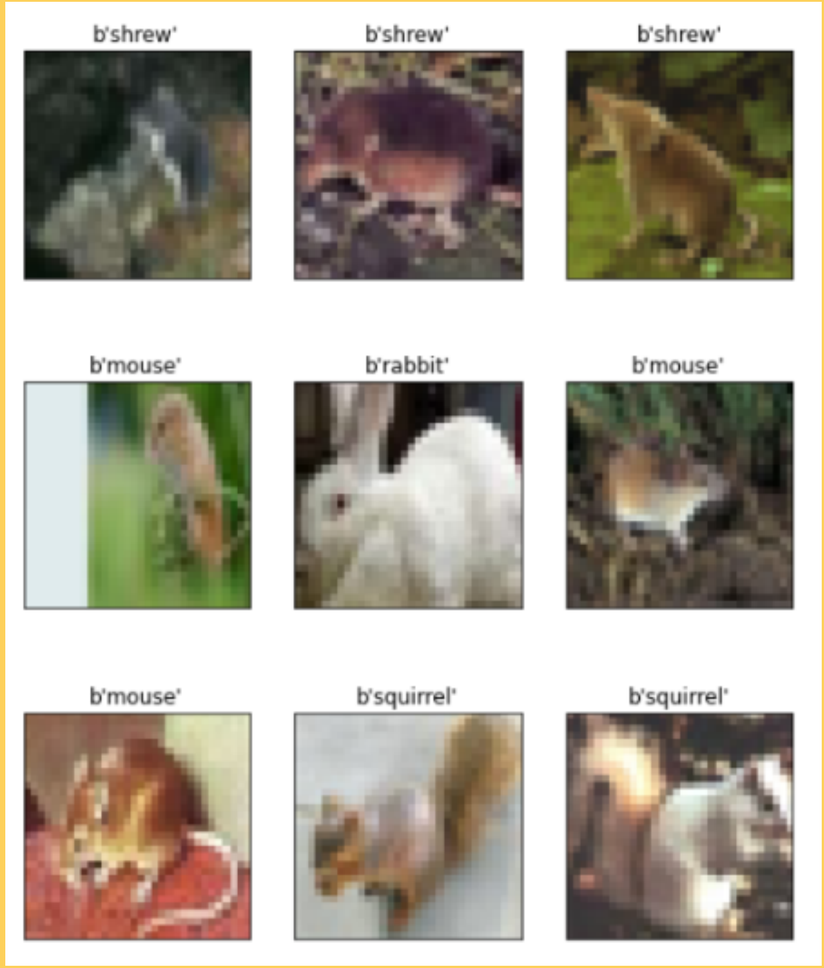
Conejo

Ardilla

02 Modelo Inicial

Imágenes de mamíferos pequeños en el CIFAR-100

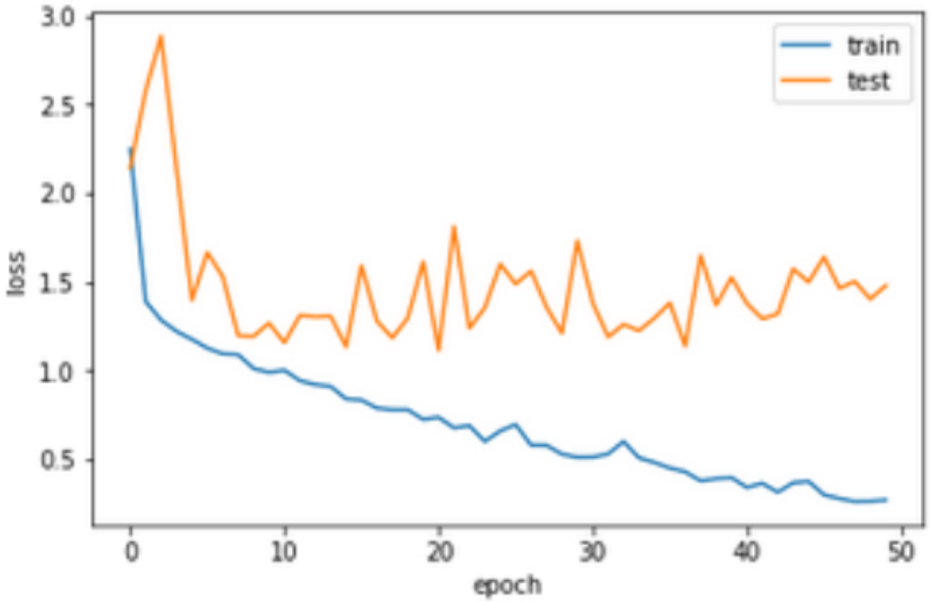


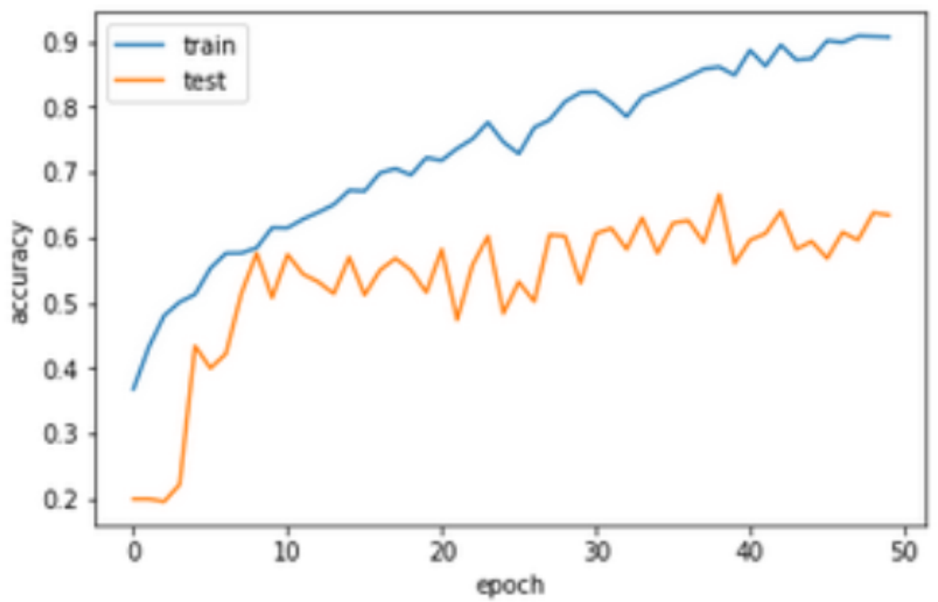


MODELO #1

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32, 32, 3)]       0

 conv2d (Conv2D)             (None, 32, 32, 32)        896

 batch_normalization (BatchN  (None, 32, 32, 32)       128
 ormalization)

 conv2d_1 (Conv2D)           (None, 32, 32, 32)        9248

 batch_normalization_1 (Batc  (None, 32, 32, 32)       128
 hNormalization)

 max_pooling2d (MaxPooling2D  (None, 16, 16, 32)       0
 )

 conv2d_2 (Conv2D)           (None, 16, 16, 64)        18496

 batch_normalization_2 (Batc  (None, 16, 16, 64)       256
 hNormalization)

 conv2d_3 (Conv2D)           (None, 16, 16, 64)        36928

 batch_normalization_3 (Batc  (None, 16, 16, 64)       256
 hNormalization)

 max_pooling2d_1 (MaxPooling  (None, 8, 8, 64)         0
 2D)

 conv2d_4 (Conv2D)           (None, 8, 8, 128)         73856

 batch_normalization_4 (Batc  (None, 8, 8, 128)        512
 hNormalization)

 conv2d_5 (Conv2D)           (None, 8, 8, 128)         147584

 batch_normalization_5 (Batc  (None, 8, 8, 128)        512
 hNormalization)

 max_pooling2d_2 (MaxPooling  (None, 4, 4, 128)        0
 2D)

 flatten (Flatten)           (None, 2048)              0

 dropout (Dropout)           (None, 2048)              0

 dense (Dense)               (None, 1024)              2098176

 dropout_1 (Dropout)         (None, 1024)              0

 dense_1 (Dense)             (None, 5)                 5125

=================================================================
Total params: 2,392,101
Trainable params: 2,391,205
```

ENTRENAMIENTO

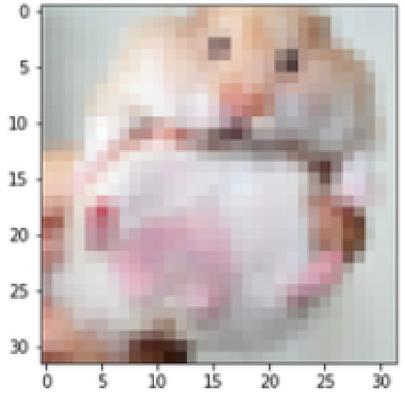




Ejecución del modelo

```
image_number=101

plt.imshow(x_test[image_number])
```

```
<matplotlib.image.AxesImage at 0x7fe11dea3550>
```




```
n = np.array(x_test[image_number])
labels = '''hamster mouse rabbit shrew squirrel'''.split()
predicted_label = labels[model.predict(n[None,:]).argmax()]

original_label = labels[y_test[image_number]]

print("Original label is {} and predicted label is {}".format(
    original_label, predicted_label))
```

```
Original label is hamster and predicted label is hamster
```

	precision	recall	f1-score	support
0	0.96	0.96	0.96	500
1	0.95	0.91	0.93	500
2	0.94	0.91	0.92	500
3	0.90	0.98	0.94	500
4	0.95	0.94	0.94	500
accuracy			0.94	2500
macro avg	0.94	0.94	0.94	2500
weighted avg	0.94	0.94	0.94	2500

---

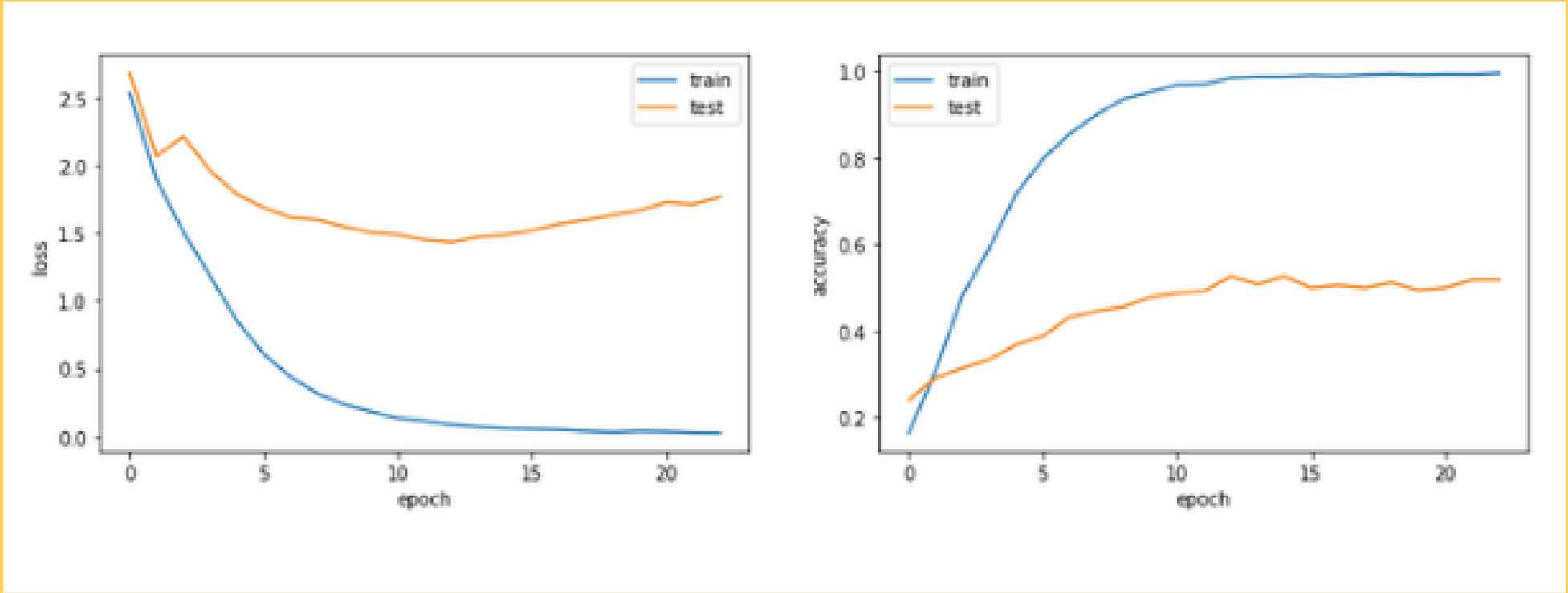
	precision	recall	f1-score	support
0	0.83	0.71	0.76	100
1	0.54	0.51	0.53	100
2	0.65	0.55	0.59	100
3	0.56	0.76	0.64	100
4	0.65	0.64	0.64	100
accuracy			0.63	500
macro avg	0.64	0.63	0.63	500
weighted avg	0.64	0.63	0.63	500

03 Modelo Intermedio

MODELO #2

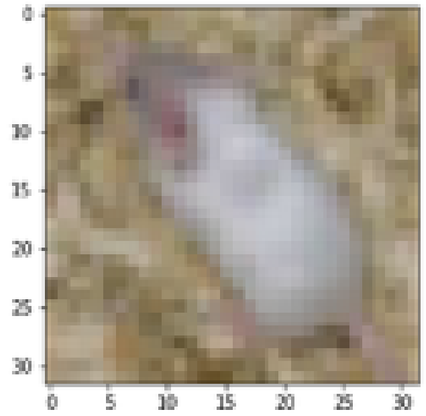
```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32, 32, 3)]       0

 resnet50 (Functional)       (None, None, None, 2048)  23587712

 max_pooling2d (MaxPooling2D  (None, 1, 1, 2048)       0
 )

 dropout (Dropout)           (None, 1, 1, 2048)        0

 flatten (Flatten)           (None, 2048)              0

 dense_2 (Dense)             (None, 16)                32784

 dense_3 (Dense)             (None, 10)                170

=================================================================
Total params: 23,620,666
Trainable params: 23,567,546
Non-trainable params: 53,120
_________________________________________________________________
```

ENTRENAMIENTO



Ejecución del modelo

```
<matplotlib.image.AxesImage at 0x7fb70603fbd0>
```




```
n = np.array(x_val[image_number])
labels = '''rabbit squirrel hamster shrew mouse'''.split()
predicted_label = labels[model.predict(n[None,:]).argmax()]

original_label = labels[y_val[image_number]]

print("Original label is {} and predicted label is {}".format(
    original_label, predicted_label))
```

```
Original label is mouse and predicted label is mouse
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	400
1	1.00	1.00	1.00	400
2	1.00	1.00	1.00	400
3	1.00	1.00	1.00	400
4	1.00	1.00	1.00	400
accuracy			1.00	2000
macro avg	1.00	1.00	1.00	2000
weighted avg	1.00	1.00	1.00	2000

---

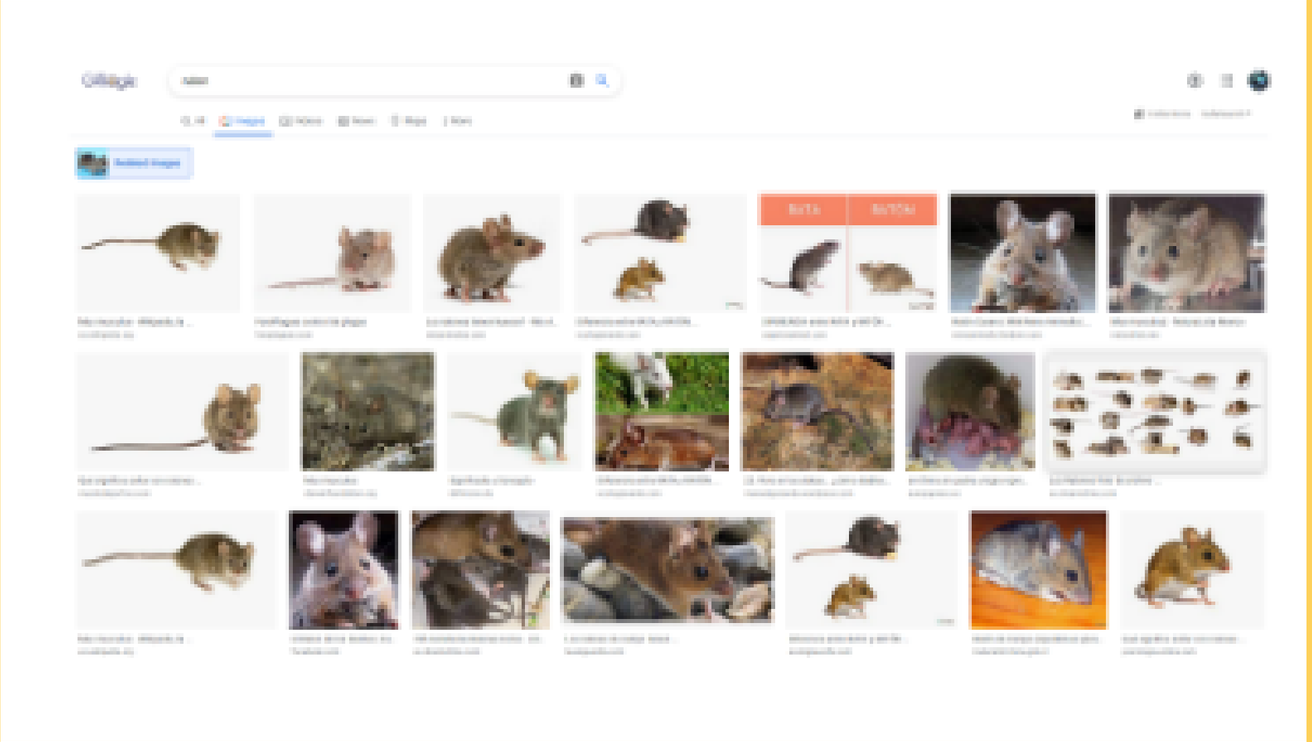
	precision	recall	f1-score	support
0	0.52	0.43	0.47	100
1	0.47	0.51	0.49	100
2	0.61	0.68	0.64	100
3	0.54	0.59	0.56	100
4	0.43	0.38	0.40	100
accuracy			0.52	500
macro avg	0.51	0.52	0.51	500
weighted avg	0.51	0.52	0.51	500

04 Modelo Final

DATASET PROPIO

Dataset:

- Hamster: 114
- Ratón: 131
- Conejo: 100
- Musaraña: 105
- Ardilla: 101

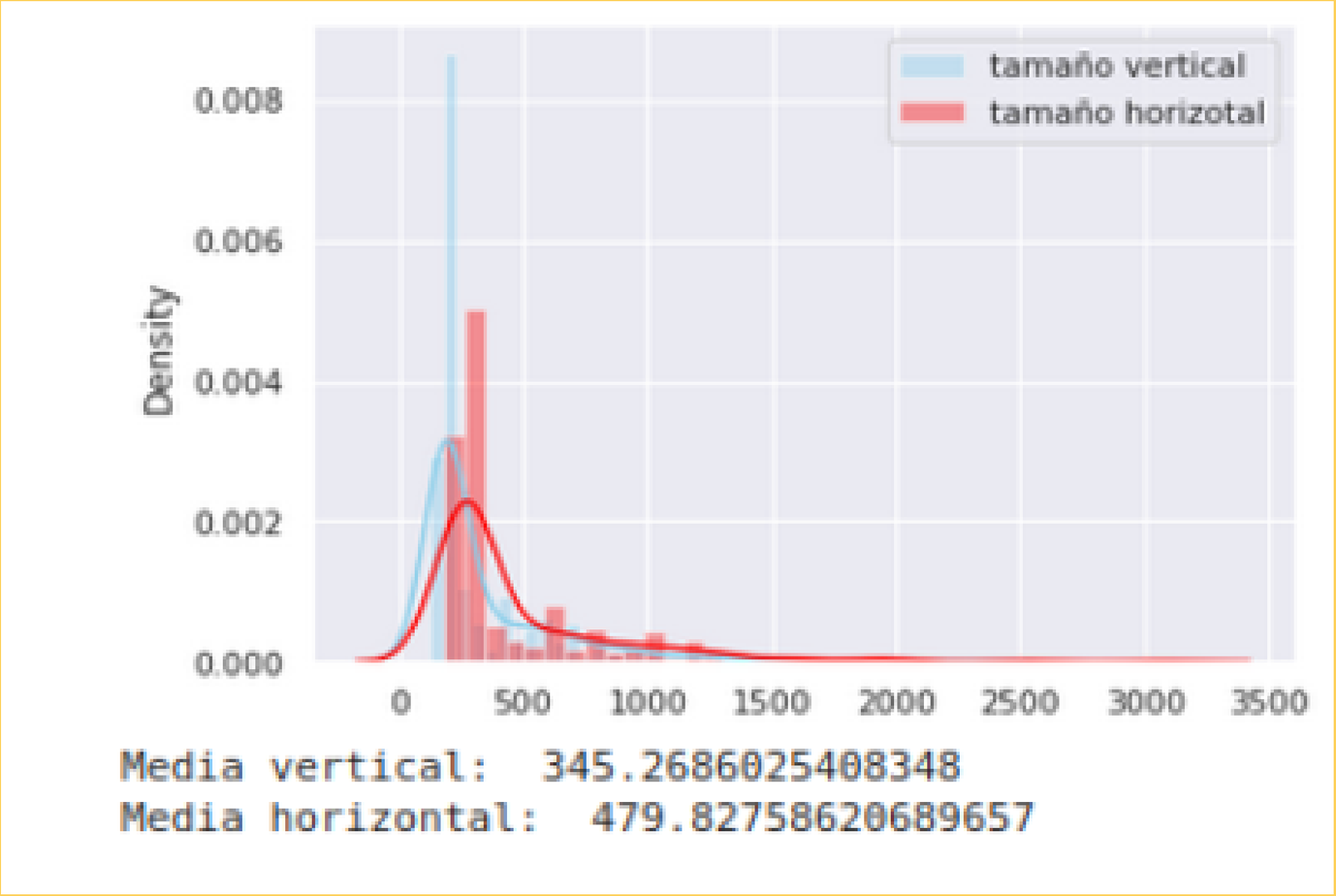


DATASET PROPIO

Train_dataset:	
Hamster:	97
Ratón:	111
Conejo:	85
Musaraña:	89
Ardilla:	86

Test_dataset:	
Hamster:	17
Ratón:	20
Conejo:	15
Musaraña:	16
Ardilla:	15

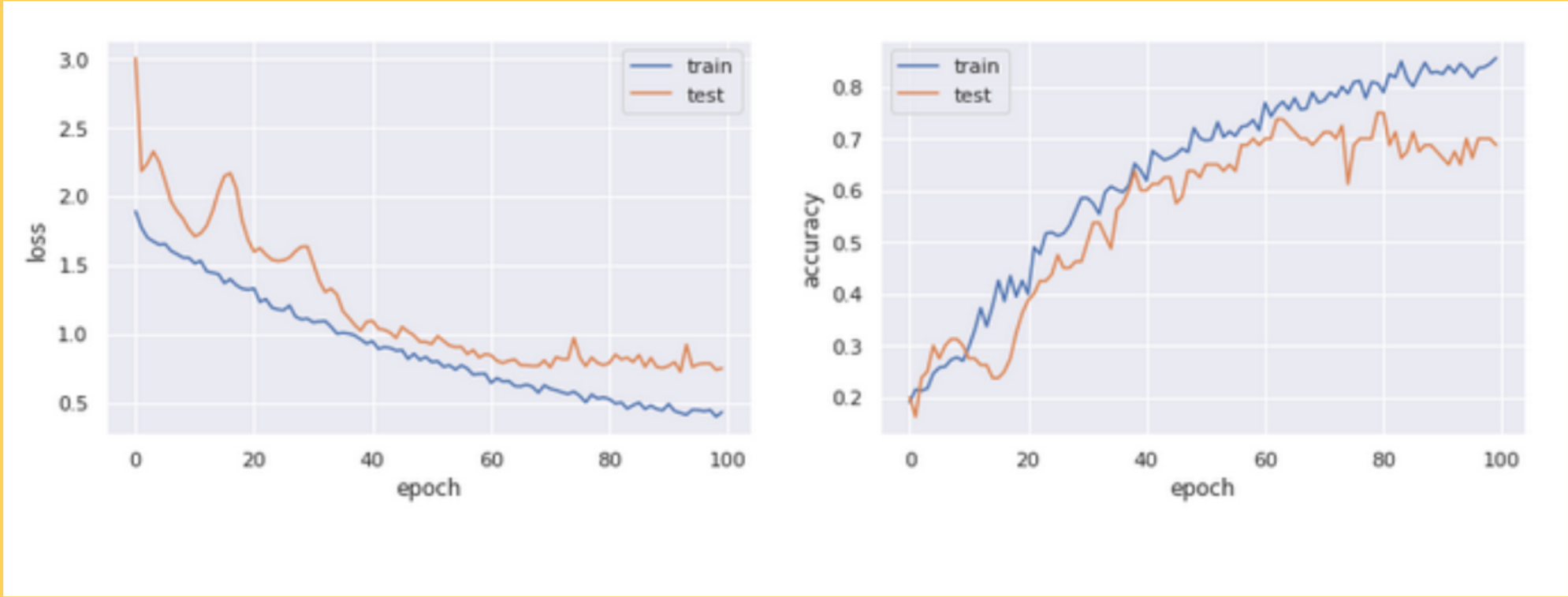
DATASET PROPIO



MODELO #3

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 activation (Activation)     (None, 224, 224, 3)       0

 conv2d (Conv2D)             (None, 112, 112, 128)     3584

 batch_normalization (BatchN  (None, 112, 112, 128)    512
 ormalization)

 activation_1 (Activation)   (None, 112, 112, 128)     0

 dropout (Dropout)           (None, 112, 112, 128)     0

 max_pooling2d (MaxPooling2D  (None, 56, 56, 128)      0
 )

 max_pooling2d_1 (MaxPooling  (None, 28, 28, 128)      0
 2D)

 conv2d_1 (Conv2D)           (None, 14, 14, 64)        73792

 dropout_1 (Dropout)         (None, 14, 14, 64)        0

 batch_normalization_1 (Batc  (None, 14, 14, 64)       256
 hNormalization)

 activation_2 (Activation)   (None, 14, 14, 64)        0

 dropout_2 (Dropout)         (None, 14, 14, 64)        0

 dense (Dense)               (None, 14, 14, 32)        2080

 dropout_3 (Dropout)         (None, 14, 14, 32)        0

 dense_1 (Dense)             (None, 14, 14, 16)        528

 dropout_4 (Dropout)         (None, 14, 14, 16)        0

 dense_2 (Dense)             (None, 14, 14, 8)         136

 flatten (Flatten)           (None, 1568)              0

 labels (Dense)              (None, 5)                 7845

=================================================================
Total params: 88,733
Trainable params: 88,349
Non-trainable params: 384
_________________________________________________________________
```

ENTRENAMIENTO



Ejecución del modelo

```
plt.imshow(image)
```

```
<matplotlib.image.AxesImage at 0x7fa9cfc07710>
```


```
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
resized_image = cv2.resize(image , (224,224))
print(type(resized_image))
prdiction = model_predict.predict(resized_image[None,:])
```

```
<class 'numpy.ndarray'>
```

```
prdiction
```

```
array([[0.01233012, 0.07350295, 0.20452142, 0.08081158, 0.6288339 ]],
      dtype=float32)
```

```
label_dict={
  0: "hamster",
  1: "mouse",
  2: "rabbit",
  3: "shrew",
  4: "squirrel"
}
```

```
label_dict[prdiction.argmax()]
```

```
'squirrel'
```

```
print(classification_report(labels_train, label_train_predictions))
```

```
              precision    recall  f1-score   support

           0       0.80      0.97      0.88        97
           1       0.90      0.77      0.83       111
           2       0.72      0.98      0.83        85
           3       0.88      0.74      0.80        89
           4       0.92      0.69      0.79        86

    accuracy                           0.83       468
   macro avg       0.84      0.83      0.83       468
weighted avg       0.85      0.83      0.83       468
```

```
print(classification_report(labels_test, label_test_predictions))
```

```
              precision    recall  f1-score   support

           0       0.76      0.76      0.76        17
           1       0.88      0.75      0.81        20
           2       0.79      1.00      0.88        15
           3       0.75      0.75      0.75        16
           4       0.71      0.67      0.69        15

    accuracy                           0.78        83
   macro avg       0.78      0.79      0.78        83
weighted avg       0.79      0.78      0.78        83
```

05 Conclusiones

¡¡¡GRACIAS!!!