DOG BREEDS IDENTIFIER

GUILLERMO BLANCO NÚÑEZ

Universidade Da Coruña - International Summer School

Data Mining and Neural Networks Course

CONTENTS

OBJECTIVES

General Objective

Develop and evaluate an automatic dog-breed classification system

Specific Objectives

1. Find and prepare the dataset
2. Implement a pre-trained CNN
3. Apply regularization techniques
4. Implement a k-NN classifier
5. Measure performance
6. Analyze errors and compare results

PROJECT SCOPE

Tools

- DogBreedsDataRepo (~10k images / 120 classes) stratified 70/15/15 split
- EfficientNetB0 (pretrained) with tuned LR/epochs/batch size
- Early stopping, dropout, L2(ridge) normalization

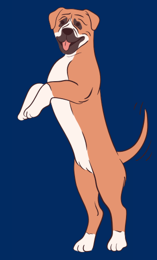
Assumptions

- All images contain at least one dog
- There are no dog breeds beyond the 120 classes
- Labels are correct

Limitations

- Limited computational capacity
- Variant image quality
- Class imbalance

METHODOLOGY - PREDICTIVE ANALYSIS



Data preparation Map image_id → breed, stratified split 70/15/15.

Pipeline Data streaming → decode, resize (224), normalize.

CNN Dropout, EarlyStopping, L2 normalization, Hyperparams

k-NN Extract embeddings, test k ∈ {3,5,7,11,21,47}.

Evaluation Accuracy, loss, F1-weighted, confusion matrixes and graphics on test

ALGORITHMS

Convolutional Neural Network

- Hyperparams:
  - Learning Rate
  - Epochs
  - Batch size
- 0.5 dropout rate
- Early stopping:
  - patience = 2
  - min_delta = 1e-3
- L2 (ridge) normalization:
  - $\lambda = 3e\text{-}4$

k - Nearest Neighbours

- Hyperparams:
  - k [4, 6, 7, 8, 9, 10, 12, 15]
- Euclidean distance
- EfficientNetB0 → 1280-D image embedding

EVALUATION - METRICS

Accuracy

- Proportion of correct predictions

$$\mathrm{Acc} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}[\hat{y}_i = y_i]$$

Loss

Sparse Categorical Cross Entropy

- Average negative log-probability of true class
- “How wrong and how confident the model is”

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} \log \hat{p}_{i,y_i}$$

F1-weighted

- Class-weighted harmonic mean of precision and recall
- Combines both errors: missing and false positives, fairly

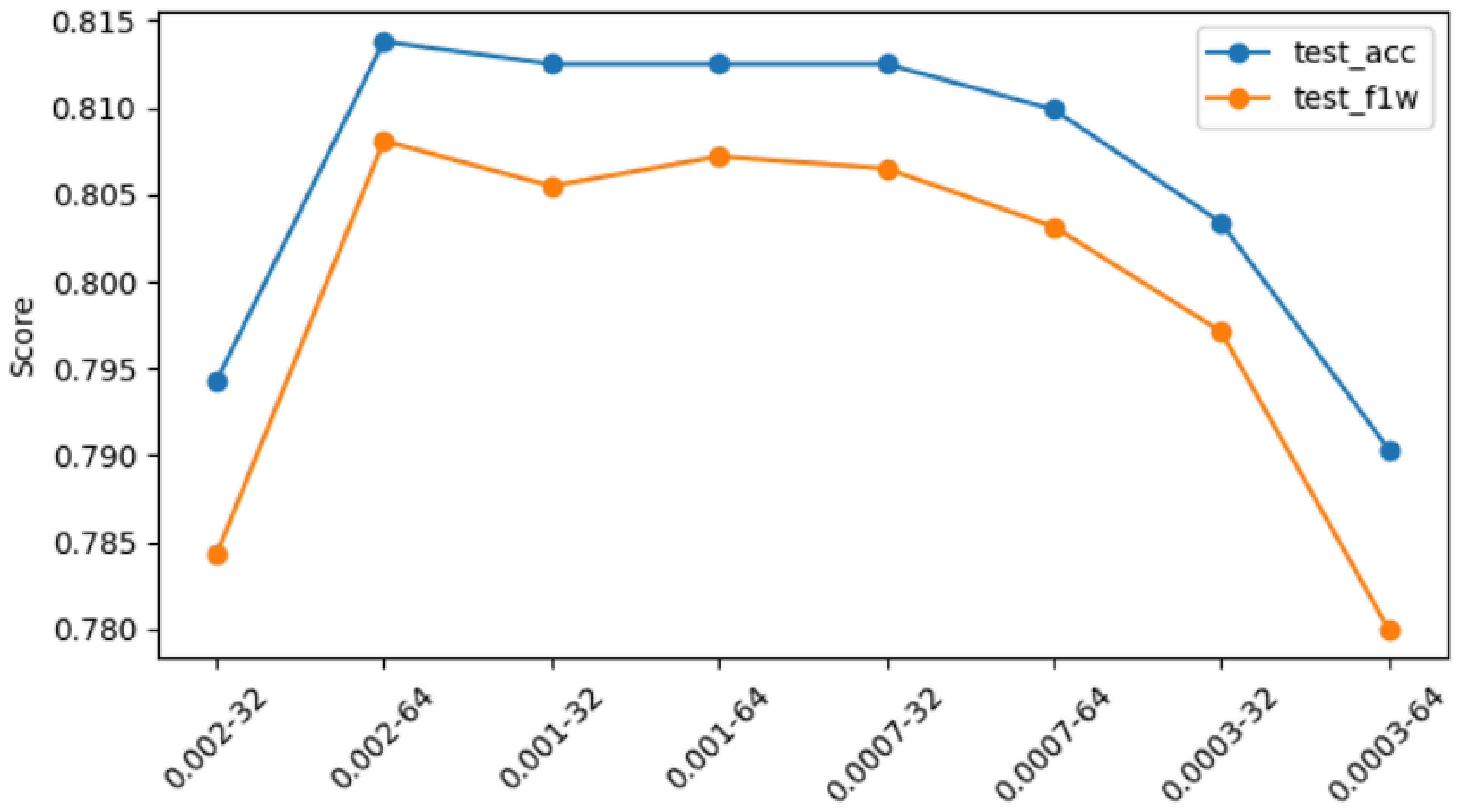
$$F1_w = \sum_c \frac{n_c}{N} \frac{2P_cR_c}{P_c + R_c}$$

RESULTS: CNN

Cuadro 1: Results for CNN classification with various hyperparameter

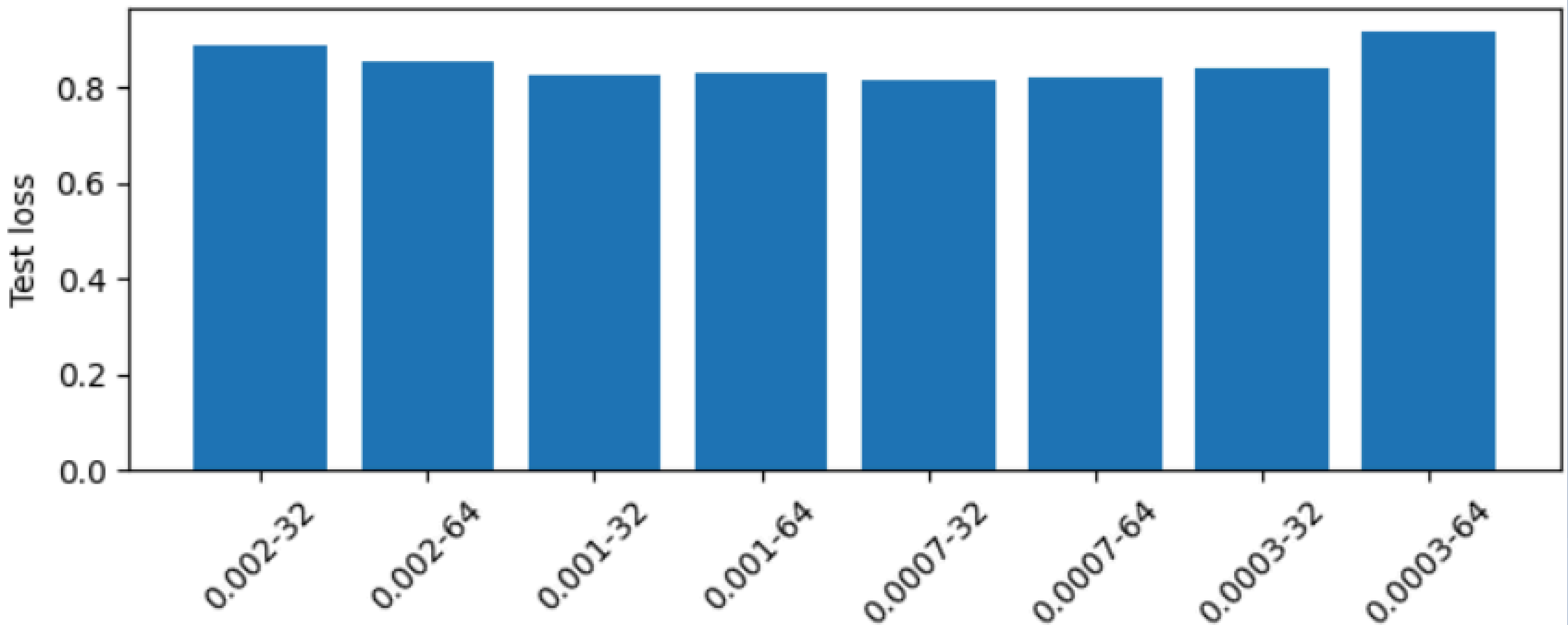
LR	epochs	batch_size	test_loss	test_acc	test_F1_weighted
0.0020	5	32	0.890785	0.794271	0.784284
0.0020	5	64	0.854059	0.813802	0.808083
0.0010	10	32	0.828719	0.812500	0.805491
0.0010	10	64	0.829831	0.812500	0.807193
0.0007	15	32	0.814895	0.812500	0.806502
0.0007	15	64	0.820036	0.808996	0.803154
0.0003	20	32	0.839535	0.803385	0.797112
0.0003	20	64	0.920145	0.790365	0.780002

Total runtime: ≈6500s

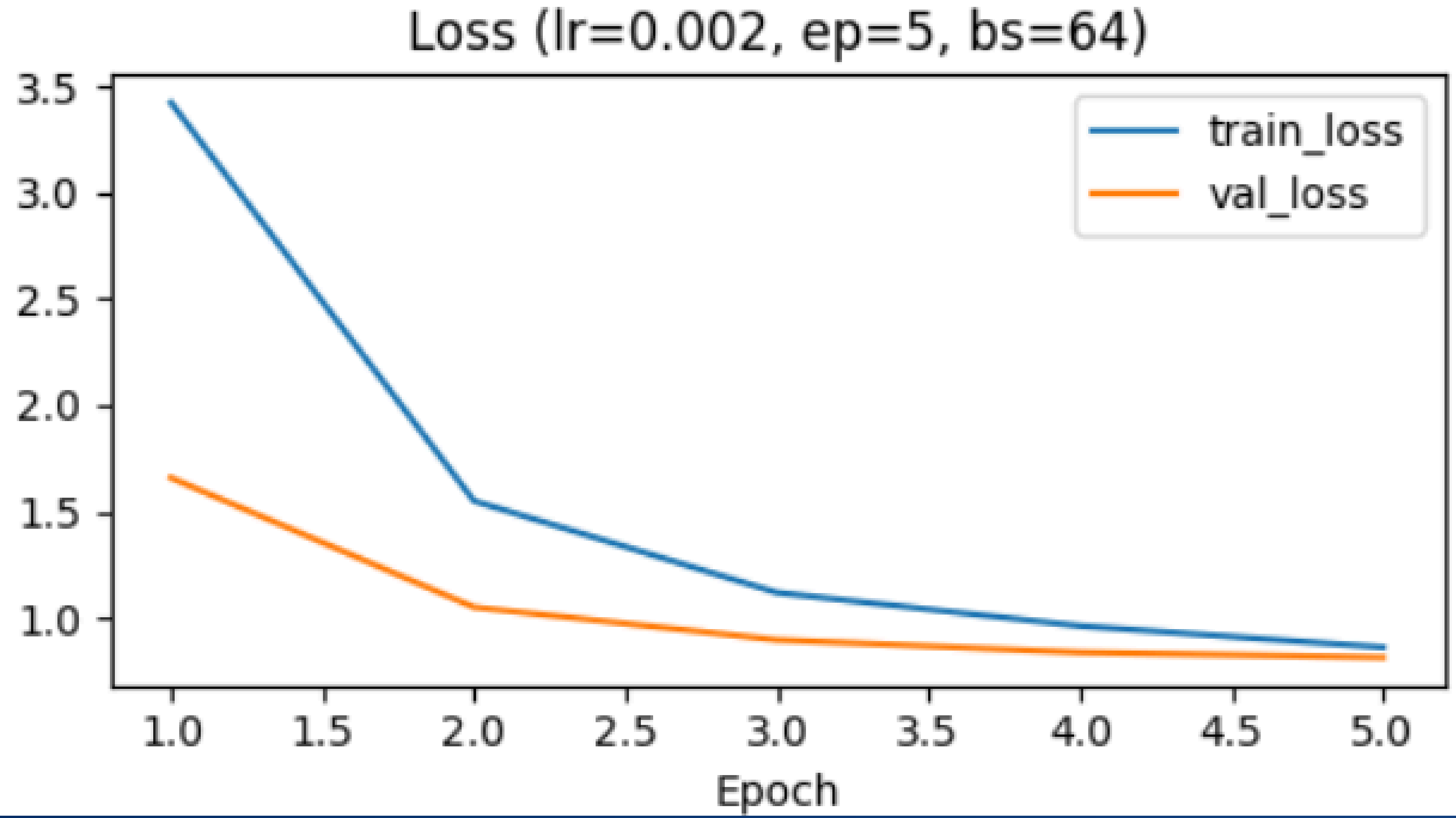


test_acc
test_f1w
Score
0.815
0.810
0.805
0.800
0.795
0.790
0.785
0.780
0.002-32
0.002-64
0.001-32
0.001-64
0.0007-32
0.0007-64
0.0003-32
0.0003-64

RESULTS: CNN



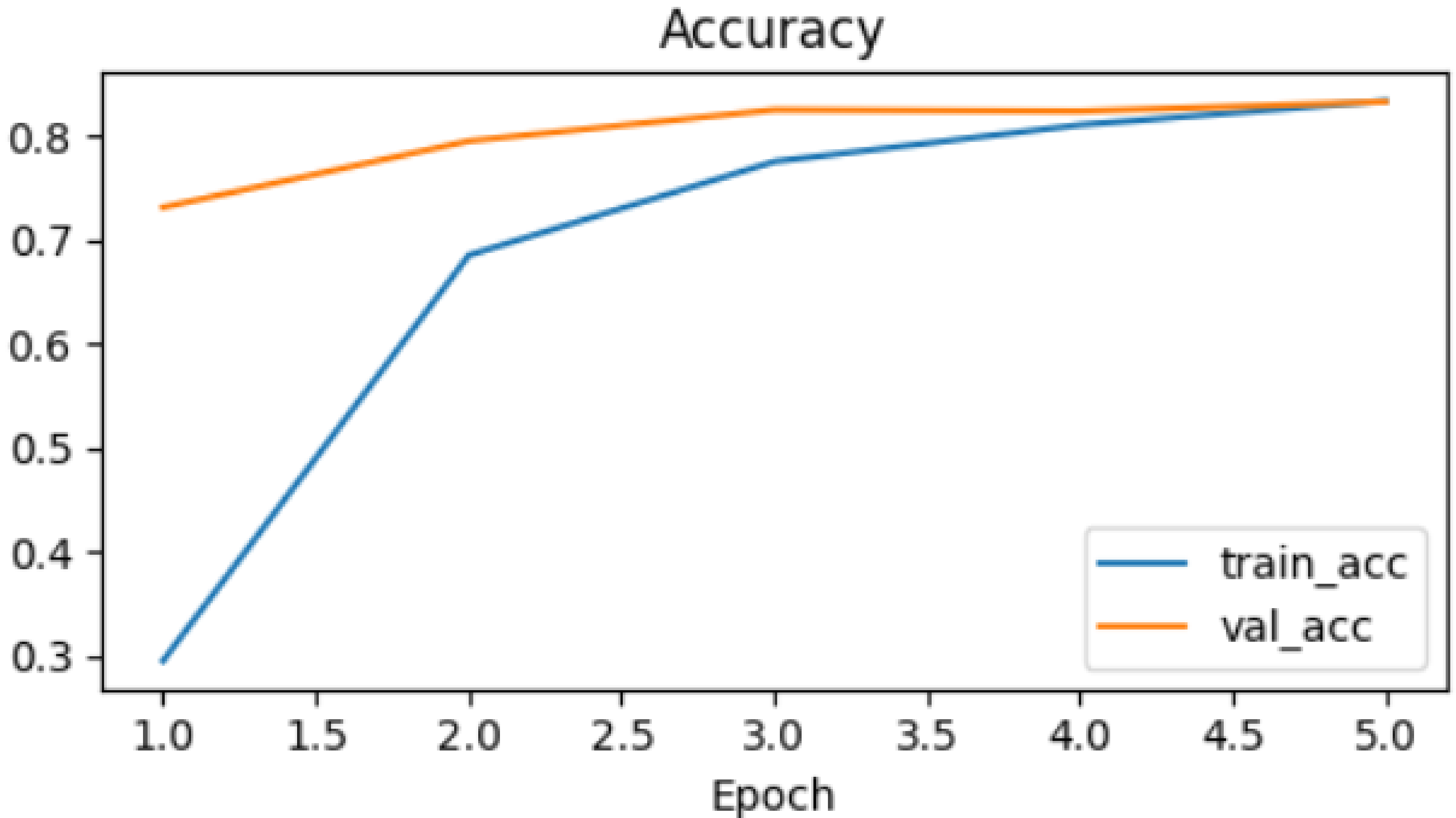
Test loss
0.8
0.6
0.4
0.2
0.0
0.002-32
0.002-64
0.001-32
0.001-64
0.0007-32
0.0007-64
0.0003-32
0.0003-64



- Hyperparameters:
    - Learning Rate: 0.002
    - Epochs: 5
    - Batch size: 64
- Train:
    - accuracy: 0.8378
    - loss: 0.8697
- Validate:
    - accuracy: 0.8333
    - loss: 0.8141
- Relative generalization gap:
    - accuracy: 0.005371
    - loss: 0.06830

RESULTS: CNN

Runtime: 342s

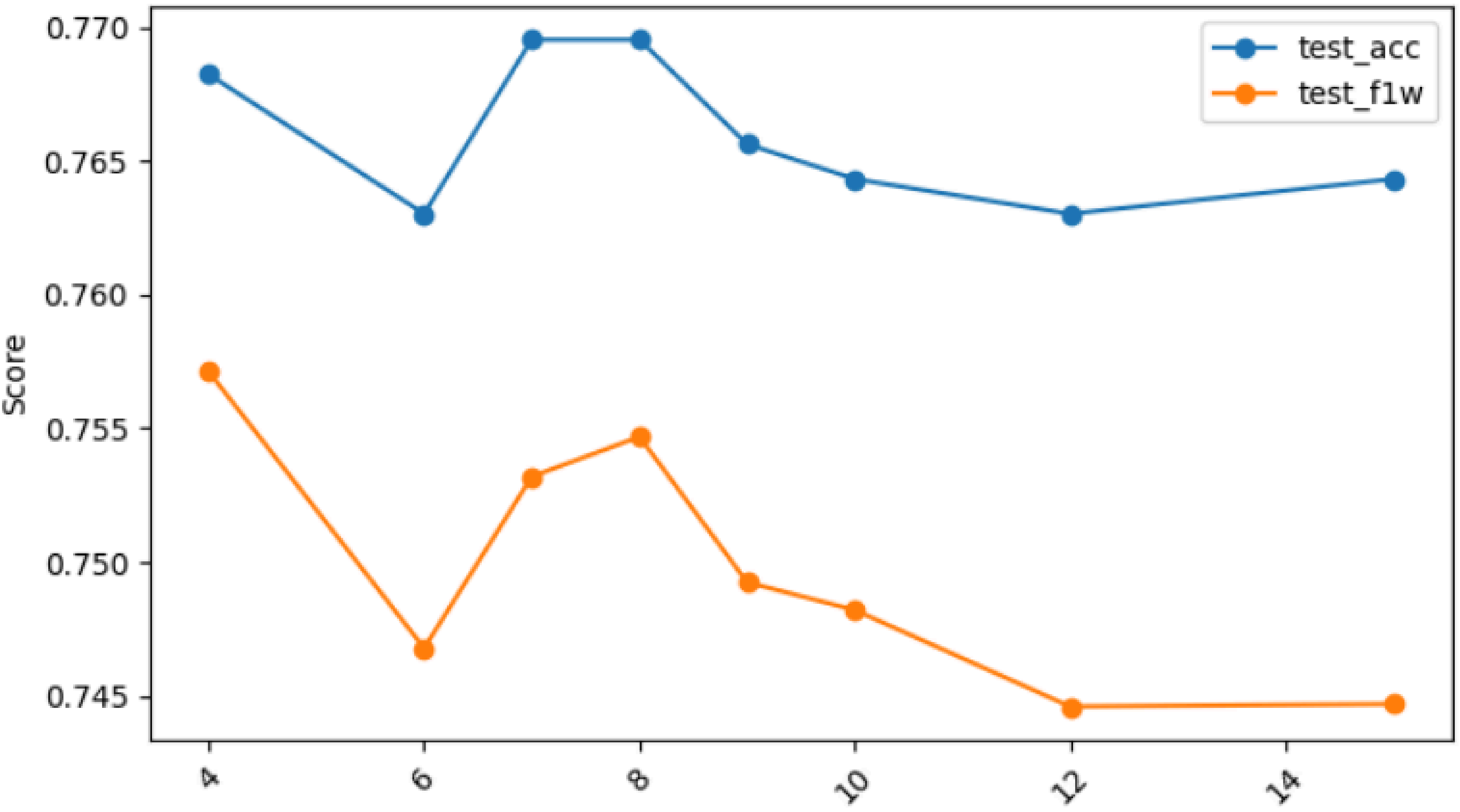


RESULTS : K-NN

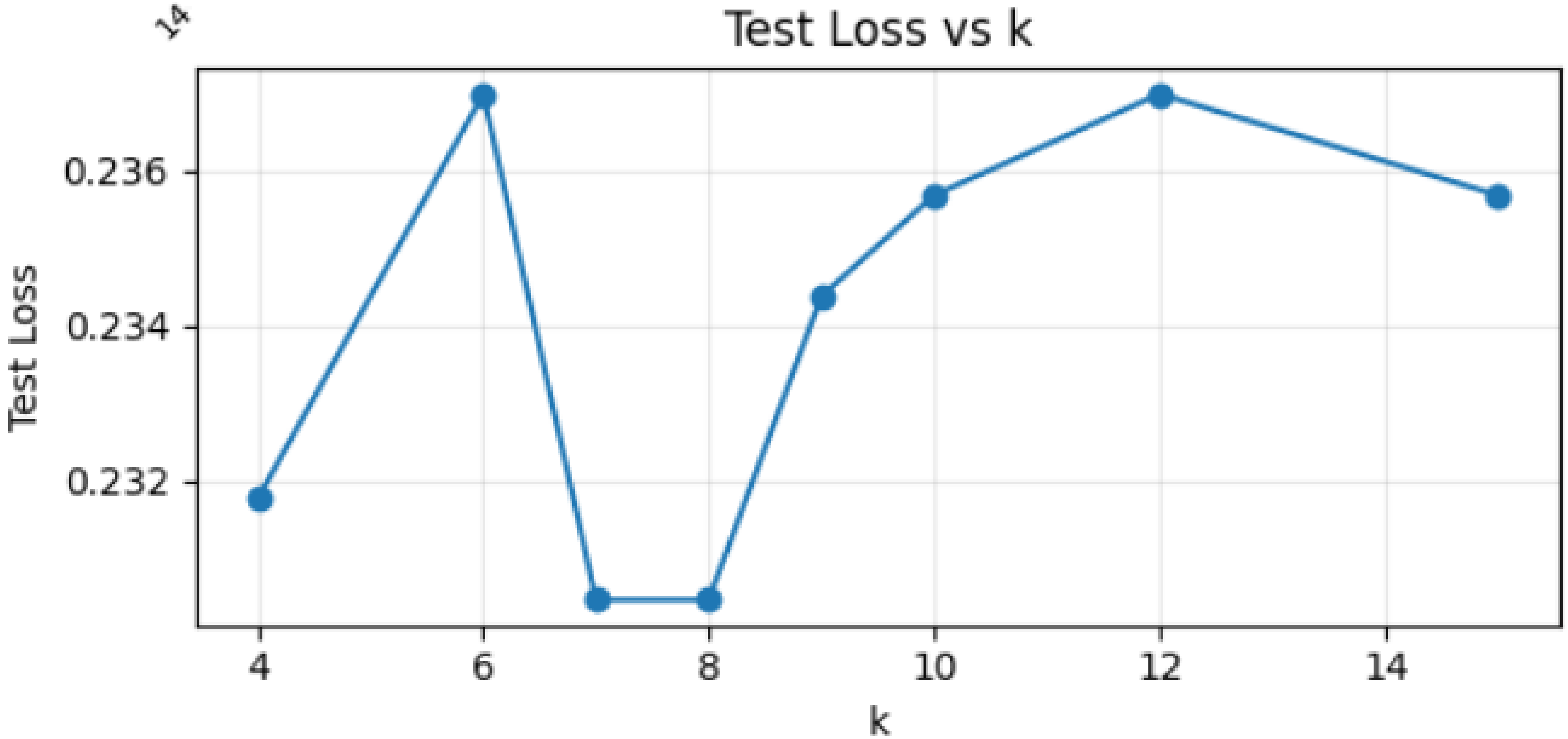
Cuadro 1: Results for k-NN classification with various k values

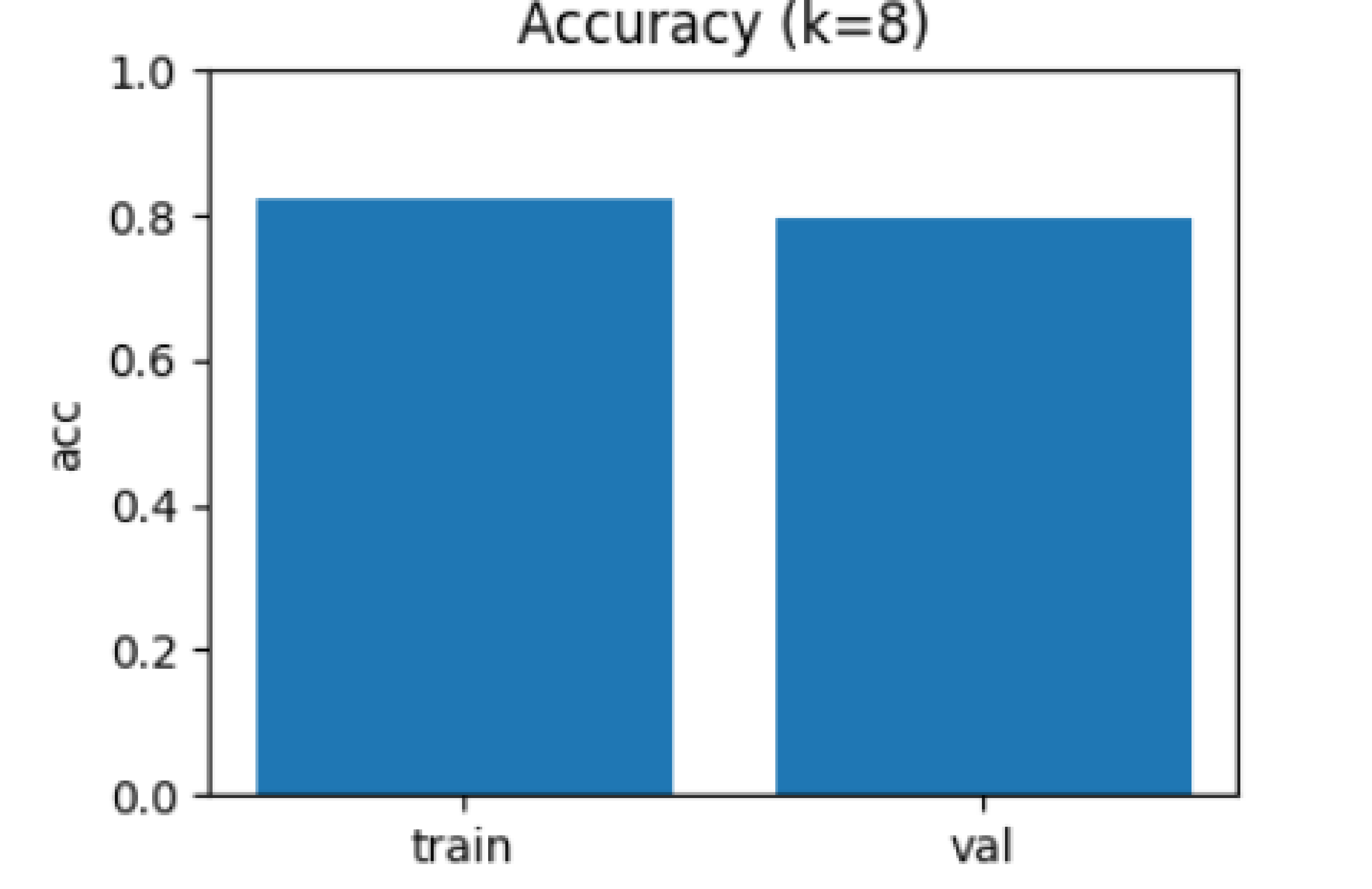
k	test_loss	test_acc	test_F1_weighted
4	0.231771	0.768229	0.757117
6	0.236979	0.763021	0.746782
7	0.230469	0.769531	0.753195
8	0.230469	0.769531	0.754686
9	0.234375	0.765625	0.749245
10	0.235677	0.764323	0.748207
12	0.236979	0.763021	0.744599
15	0.235677	0.764323	0.744694

Total runtime: 53s


test_acc
test_f1w
Score
0.770
0.765
0.760
0.755
0.750
0.745
4
6
8
10
12
14

RESULTS: K-NN

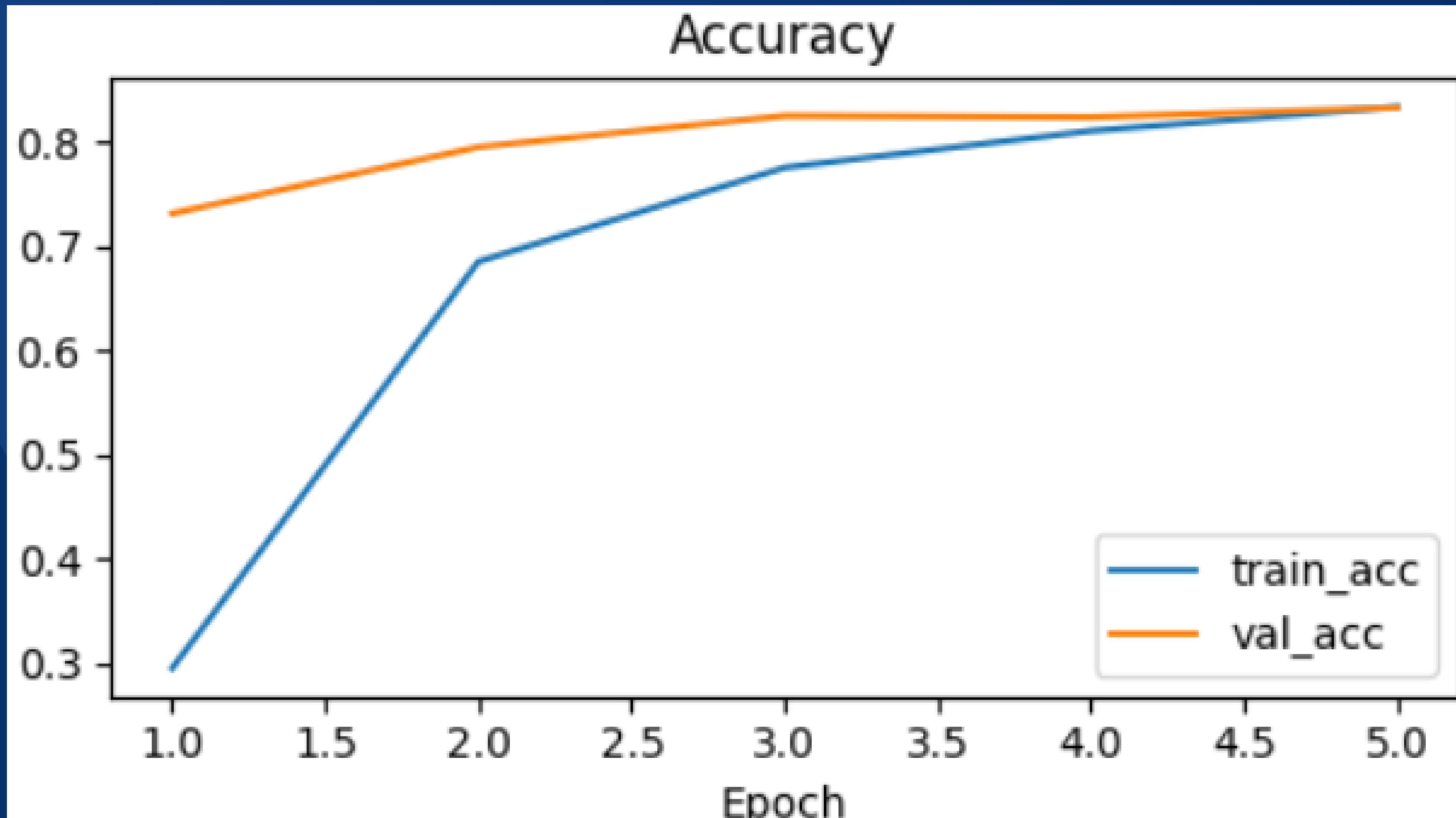

Test Loss vs k
Test Loss
0.236
0.234
0.232
4
6
8
10
12
14
k



- Hyperparameters:
  - k: 8
- Train:
  - accuracy: 0.821608
  - loss: 0.178392
- Validate:
  - accuracy: 0.795573
  - loss: 0.204427
- Relative generalization gap:
  - accuracy: 0.031688
  - loss: 0.145943

RESULTS: K-NN

Runtime: 1.55s



CONCLUSION

Validity & Reliability

- CNN is more valid given its slighter higher accuracy values
- k-NN proved to be more reliable as it has a much lower loss value

Computational Requirements

- CNN´s runtime was 10,000 times larger than k-NN´s, clearly computationally more demanding

Final Decision

Reliability between the two models is close and the difference in accuracy isn´t large enough to rule out CNN´s far larger computational demands. k-NN is preferrable.

Universidade Da Coruña - International Summer School
Data Mining and Neural Networks Course

THANK YOU

FOR ATTENTIONS

github.com/GuillermoBlancoNunez/DogBreedsClassification