Informe Final – Lab 1 – Rodolfo Varela

# 1. Objetivo

Comparar distintos enfoques de clasificación sobre el dataset CIFAR-10, desde modelos entrenados desde cero (CNN y MLP), pasando por modelos clásicos sobre características planas, hasta transfer learning con una red profunda preentrenada (ResNet-50). Evaluar su desempeño en validación y test, y resaltar beneficios de transferencia de aprendizaje.

# 2. Datos y preprocesado

• CIFAR-10: 60,000 imágenes RGB 32×32 en 10 clases. Se usó un split interno de 20% para train y 80% para validación, y el conjunto de test estándar de 10,000 imágenes para evaluación final.

• EDA: Conteo por clase confirmó equilibrio (5,000 imágenes por clase antes del split) y visualización de un ejemplo por clase para verificar integridad.

• Transforms:

**- Modelos propios:** ToTensor + normalización a [-1,1].

**- ResNet-50:** Resize 128×128 y normalización de ImageNet (mean=[0.485,0.456,0.406], std=[0.229,0.224,0.225]).

# 3. Modelos entrenados

Se entrenaron los siguientes modelos:  
1. SimpleCNN: red convolucional entrenada desde cero con tres capas convolucionales y dos fully connected. Adam lr=0.001, 5 épocas, checkpoint según validación.  
2. SimpleMLP: perceptrón sobre vectores planos de dimensión 3·32·32 con una capa oculta de 512 unidades.  
3. Modelos clásicos: Random Forest, SVM (RBF), KNN (k=5) y XGBoost entrenados sobre imágenes aplanadas.  
4. Transfer Learning con ResNet-50: se cargó la red preentrenada, se congelaron sus pesos excepto la última capa adaptada a 10 clases, y se entrenó durante 3 épocas sobre imágenes 128×128.

# 4. Métricas y resultados

|  |  |
| --- | --- |
| Modelo | Test Accuracy |
| SimpleCNN | 0.6051 |
| SimpleMLP | 0.4475 |
| Random Forest | 0.4211 |
| SVM | 0.4695 |
| KNN | 0.2954 |
| XGBoost | 0.4755 |
| ResNet-50 (TL) | 0.7520 |

Observaciones: SimpleCNN supera a los modelos clásicos y MLP, mientras que ResNet-50 con transfer learning obtiene la mejor precisión (~75.2%). SVM y XGBoost fueron los mejores entre los modelos clásicos. KNN tuvo bajo desempeño por la alta dimensionalidad sin reducción.

# 5. Análisis por clase

En la SimpleCNN, las clases con mayor recall fueron automobile, frog y ship. Las más difíciles fueron cat y bird, con confusiones entre especies similares. Transfer learning con ResNet-50 mejora especialmente cat y bird, reduciendo el error en esas categorías.

# 6. Configuración de cómputo y tiempos

Todas las redes fueron entrenadas utilizando GPU (NVIDIA GeForce GTX 1650) con PyTorch + CUDA. Se activó cudnn.benchmark, pin\_memory y transferencias non\_blocking para eficiencia. SimpleCNN tomó aproximadamente 1 minuto por época; ResNet-50 (128×128) tardó ~165 segundos por época en fine-tuning ligero.

# 7. Conclusiones

Transfer learning proporcionó la mejora más sustancial, superando por más de 15 puntos porcentuales al mejor modelo entrenado desde cero. Las arquitecturas simples funcionaron como buenos baselines. La correcta separación de datos, normalización y checkpointing fueron claves para comparabilidad.

# 8. Recomendaciones

1. Incorporar data augmentation (random crop, flip, mixup, cutmix) para mejorar generalización.  
2. Descongelar más capas de ResNet y entrenar con scheduler (p.ej. cosine annealing) para aumentar aún más la accuracy.  
3. Añadir early stopping y regularización para redes entrenadas desde cero.  
4. Incluir curvas de aprendizaje y matrices de confusión comparativas en el informe final.  
5. Explorar ensembling entre los mejores modelos para robustez.

# 9. Github