3.- Entrenamiento y Optimización de Redes Neuronales

1. Introducción

El entrenamiento de una red neuronal es el proceso mediante el cual el modelo aprende a ajustar sus pesos internos para minimizar los errores en sus predicciones. Durante este proceso, la red compara sus salidas con las salidas reales (etiquetas) y modifica los pesos mediante retropropagación y optimización.

2. Función de pérdida (Loss Function)

La función de pérdida mide el grado de error entre la salida predicha y la salida real.
Durante el entrenamiento, la red intenta minimizar esta función.

Tipo de problema	Función de pérdida típica	Descripción
Regresión	MSE (Error cuadrático medio)	Penaliza grandes desviaciones.
Clasificación binaria	Entropía cruzada binaria	Compara probabilidades entre dos clases.
Clasificación multiclase	Entropía cruzada categórica	Mide la diferencia entre distribuciones de probabilidad.
Autoencoders	MSE o Binary Crossentropy	Evalúa la fidelidad de la reconstrucción.
GAN	Binary Crossentropy	Competencia entre generador y discriminador.

3. Optimización y algoritmos de entrenamiento

Los optimizadores son algoritmos que actualizan los pesos de la red para reducir la pérdida, utilizando los gradientes calculados por retropropagación.

Optimizador	Características principales
SGD	Actualiza pesos tras cada mini-lote. Simple pero lento.
Momentum	Añade inercia para suavizar oscilaciones.
RMSProp	Ajusta tasa de aprendizaje según la variabilidad.
Adam	Combina Momentum y RMSProp. Muy usado por su estabilidad.
Adagrad / Adadelta	Adaptativos, útiles en problemas dispersos.

4. Aprendizaje por lotes (Batch, Mini-Batch, Epoch)

Durante el entrenamiento, los datos se dividen en lotes. Un 'batch' es un grupo de muestras usado en una actualización de pesos. Un 'epoch' representa una pasada completa por el conjunto de entrenamiento.

5. Overfitting, Underfitting y Regularización

El underfitting ocurre cuando el modelo no aprende lo suficiente; el overfitting, cuando memoriza los datos de entrenamiento. Existen varias técnicas de regularización para mejorar la generalización.

Técnica	Descripción
Dropout	Desactiva aleatoriamente neuronas durante el entrenamiento.
Early Stopping	Detiene el entrenamiento al detectar empeoramiento en validación.
Regularización L1/L2	Penaliza pesos grandes en la función de pérdida.
Data Augmentation	Genera variaciones del conjunto de datos para evitar sobreajuste.
Batch Normalization	Normaliza activaciones internas, mejorando la estabilidad.

6. Parámetros e Hiperparámetros

Los parámetros son los pesos aprendidos automáticamente. Los hiperparámetros son configuraciones definidas por el usuario que afectan al entrenamiento (learning rate, número de capas, batch size, etc.).

7. Evaluación del modelo

Se utilizan conjuntos de entrenamiento, validación y test. Las métricas varían según el tipo de problema.

Tipo de problema	Métricas recomendadas
Clasificación	Accuracy, Precision, Recall, F1, ROC–AUC
Regresión	MSE, MAE, R^2
Autoencoder / GAN	Pérdida de reconstrucción, calidad visual

8. Buenas prácticas y técnicas de mejora

1. Normalizar y limpiar los datos.
2. Dividir correctamente train/validation/test.
3. Empezar con modelos simples.
4. Monitorear curvas de pérdida y precisión.
5. Usar callbacks de Keras (EarlyStopping, ReduceLROnPlateau, ModelCheckpoint).
6. Guardar el modelo entrenado (.keras o .h5).
7. Evaluar la generalización en datos nuevos.
8. Cambiar un hiperparámetro por vez para entender su impacto.

Conclusión

El éxito de una red neuronal no depende solo de su arquitectura, sino del proceso de entrenamiento y optimización. Comprender las funciones de pérdida, los optimizadores y las técnicas de regularización es esencial para crear modelos precisos y estables.