Deep Learning para Business Analytics

Día 03 · Tema 1 — Buenas prácticas

Cómo entrenar redes sin que se mientan a sí mismas.

Eduardo C. Garrido-Merchán

Profesor Propio Adjunto, Métodos Cuantitativos, ICADE

ecgarrido@comillas.edu

Madrid, Día 03

BBVA · Telefónica · NETFLIX · amazon · INDITEX · Iberdrola

ejemplos del día

COMILLAS UNIVERSIDAD PONTIFICIA ICAI ICADE CIHS

Lo que vamos a cubrir hoy.

1. El bucle de entrenamiento

2. Regularización

3. Learning rate schedules

4. Dashboards y trackers de experimentos

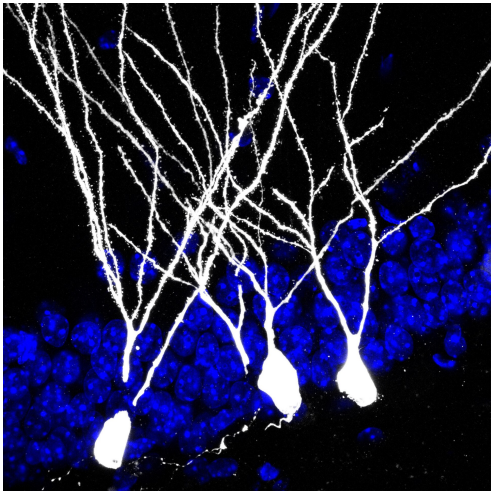
5. Caso completo: receta exprés

Lo que vas a saber hacer al final:

1. Contar iteraciones y épocas, y elegir el tamaño de batch con criterio.
2. Diagnosticar overfitting y aplicar dropout, BN y weight decay.
3. Programar un cosine annealing del LR en PyTorch en cinco líneas.
4. Defender la elección de tracker (W&B vs MLflow) ante un equipo de datos.

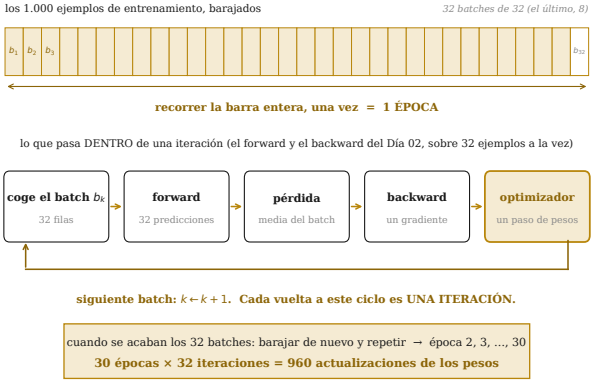
Parte 1 / 5

El bucle de entrenamiento



Entrenar es repetir el paso del Día 02 muchas veces: batch, iteración y época.

El dataset se trocea en batches y el bucle recorre los trozos, uno a uno.



Las tres palabras. 📚 **Batch** (lote): el trozo de filas que entra a la vez. **Iteración**: procesar un batch y dar *un* paso del optimizador. **Época**: una pasada completa por todo el dataset.

Las cuentas. ⚙️ Con 1.000 ejemplos y batch $B = 32$ salen $\lceil 1000/32 \rceil = 32$ iteraciones por época (la última, con 8 filas). En 30 épocas, $30 \times 32 = 960$ actualizaciones de los pesos.

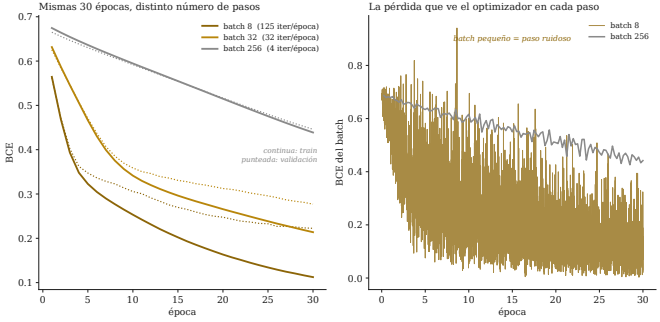
Qué añade sobre ayer. 🚀 Ayer calculamos el gradiente de un ejemplo. Aquí el gradiente es el promedio sobre las 32 filas del batch, y el bucle lo repite 960 veces.

Por qué se baraja. 🎲 Sin barajar entre épocas, los batches son siempre los mismos 32 grupos y su orden se convierte en una señal espuria.

Idea clave. En el Caso 1, fijar `epochs` y `batch_size` es decidir cuántos pasos da el modelo.

El tamaño del batch decide cuánto ruido lleva cada paso y cuántos pasos das.

Misma red, mismos datos, mismo learning rate: sólo cambia el tamaño del batch.



Montaje. ⚙️ La red del Día 02 ($20 \to 32 \to 1$, ReLU, Adam con lr $= 10^{-3}$, BCE), 1.000 filas tabulares, 30 épocas, misma semilla e igual inicialización. Sólo cambia B.

Resultado. 📊 $B = 8$: 3.750 pasos, BCE de train **0.112** (validación 0.223). $B = 32$: 960 pasos, **0.214** (0.277). $B = 256$: 120 pasos, **0.439** (0.446).

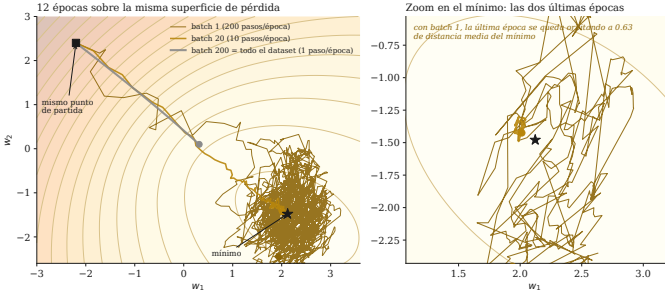
Lectura. ⏱️ A igualdad de épocas, el batch grande actualiza menos veces y avanza menos; necesita más épocas o un lr mayor. El panel derecho enseña el precio del batch pequeño: cada paso se estima con 8 filas y sale ruidoso.

Y hoy. 📚 Dropout resortea máscara en cada batch, BatchNorm usa la media del batch (se rompe con $B \leq 4$), y early stopping y los schedules cuentan épocas.

Idea clave. Empieza en $B = 32$–256 en tabular; sube B hasta llenar la GPU y sube el lr con él.

El batch pequeño zigzaguea; el dataset entero va derecho y despacio.

Mismo arranque, mismo learning rate, mismas épocas: el tamaño del batch dibuja el camino.



Qué se ve. 👁️ La superficie de pérdida de una regresión logística de dos pesos, y encima tres descensos de verdad: mismo arranque, mismo lr, las mismas 12 épocas. Lo único distinto es *B*.

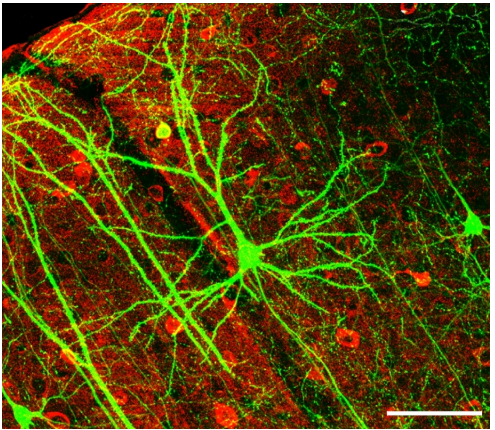
Batch 1. 🎲 200 pasos por época, y cada uno mira un solo ejemplo: apunta casi a cualquier sitio. En el zoom, la última época orbita a 0.63 del mínimo sin llegar a posarse.

Los otros dos. ⏱️ El dataset entero da un paso por época en la dirección exacta: doce pasos limpios que se quedan a medio camino. Batch 20 llega al mínimo y se queda razonablemente quieto.

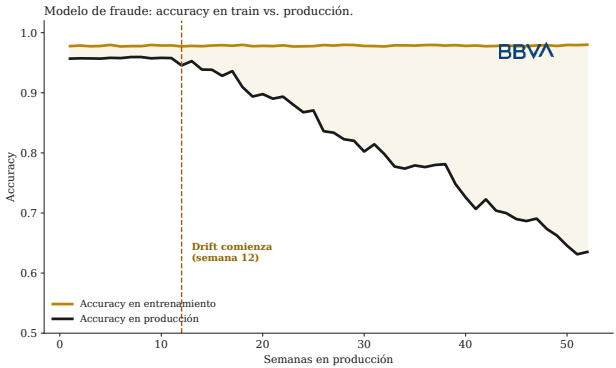
Idea clave. El ruido explora y también impide asentarse. Por eso el schedule del lr baja al final: para apagar el temblor.

Parte 2 / 5

Regularización



El modelo de fraude de BBVA dejó de funcionar a los tres meses.



Qué pasó. ⚠️ El modelo memorizó patrones que eran artefactos de la ventana de entrenamiento: un pico estacional de compras navideñas, un partner temporal, una campaña puntual.

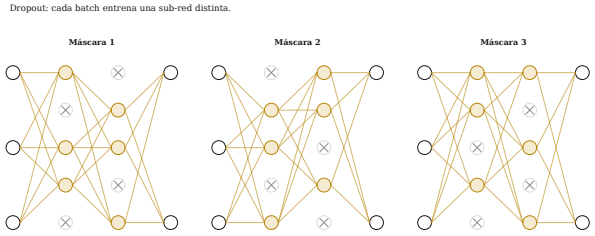
Resultado. 📊 Accuracy en train: 98 %. Accuracy en producción a las 12 semanas: 72 % y cayendo.

BBVA Caso BBVA (estilizado)

Lección. 🛡️ Regularizar = comprar seguros contra la memorización. Un modelo que generaliza es un modelo que sobrevive en producción.

Idea clave. El overfitting no es solo un problema académico: cuesta dinero real.

Dropout es estudiar para el examen tapándose los apuntes al azar.



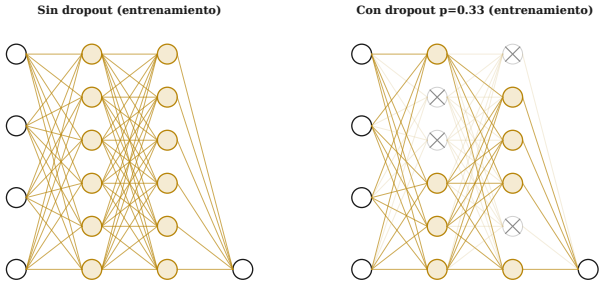
Analogía. 🎲 Si aprendes a resolver problemas sin depender de ningún compañero concreto, eres más robusto ante ausencias. Dropout hace exactamente eso con las neuronas.

Mecanismo. ⚙️ En cada minibatch, cada neurona oculta se apaga con probabilidad p. La red que queda es una sub-red distinta. El entrenamiento promedia implícitamente sobre 2^H sub-redes (H = neuronas ocultas).

Efecto. 🛡️ Fuerza representaciones redundantes: ninguna neurona individual puede "cargar" con toda la señal. Reduce co-adaptación.

Idea clave. Dropout es el ensemble más barato que existe: 2^H modelos por el precio de uno.

Dropout: apagar neuronas al azar en cada batch.

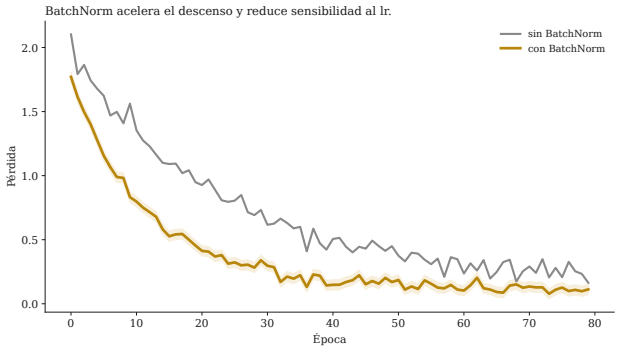


Idea. 🎲 Cada neurona se desactiva con probabilidad p (0,1 a 0,5) en entrenamiento. En inferencia se conservan todas.

Efecto. 🛡️ La red no puede depender de neuronas específicas; aprende representaciones redundantes.

Idea clave. Dropout es el regularizador más simple y más efectivo en MLP y CNN.

BatchNorm: estabilizar la distribución entre capas.



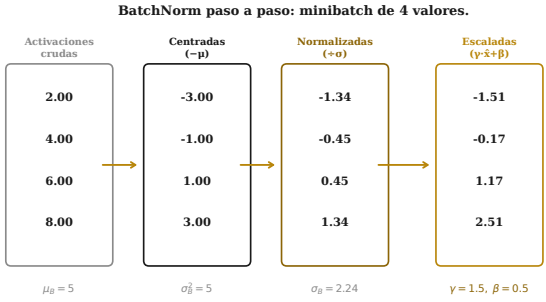
Idea. 📚 Normaliza las activaciones por minibatch (media 0, desv. 1) y aprende dos parámetros (γ, β) por canal.

Efecto. 🚀 Permite lr más alto, acelera el entrenamiento, suaviza la pérdida y reduce la sensibilidad a la inicialización.

Cuándo no. ⚠️ Si el batch es muy pequeño (≤ 4), usa LayerNorm o GroupNorm.

Idea clave. Por defecto: BatchNorm en CNN, LayerNorm en Transformer.

BatchNorm paso a paso sobre un minibatch de 4 valores.



Datos.⚙️

Minibatch $a = (2,\ 4,\ 6,\ 8)$.

Paso 1 — media y varianza.🧠

$\mu_B = \frac{2+4+6+8}{4} = 5$

$\sigma_B^2 = \frac{(2-5)^2+(4-5)^2+(6-5)^2+(8-5)^2}{4} = 5$

Paso 2 — normalizar.🎯

$\hat{a}_i = \frac{a_i-\mu_B}{\sqrt{\sigma_B^2+\epsilon}} \approx \frac{a_i-5}{2{,}236}$

$\hat{a} = (\mathbf{-1.34},\ \mathbf{-0.45},\ \mathbf{0.45},\ \mathbf{1.34})$

Paso 3 — escalar y desplazar.🚀

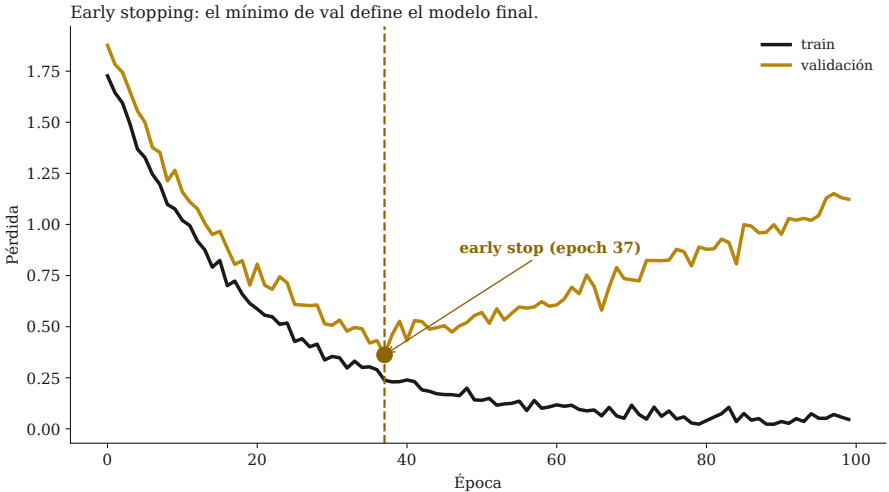
Con $\gamma = 1{,}5$, $\beta = 0{,}5$:

$y_i = \gamma\,\hat{a}_i + \beta$

$y = (\mathbf{-1.51},\ \mathbf{-0.17},\ \mathbf{1.17},\ \mathbf{2.51})$

Idea clave. γ y β se aprenden: la red puede "deshacer" la normalización si lo necesita.

Early stopping: parar antes de empezar a memorizar.



Ejemplo a mano: trazado paso a paso de early stopping.

Trazo de un early stopping con paciencia 12 sobre 28 épocas.

Época	Train loss	Val loss	best?	since_best	Acción
1	0.86	0.71	**sí**	0	guardar checkpoint
5	0.52	0.45	**sí**	0	guardar
10	0.34	0.34	**sí**	0	guardar
15	0.20	0.35	no	1	seguir entrenando
18	0.14	0.41	no	4	seguir
22	0.10	0.48	no	8	seguir, atención
28	**0.07**	**0.62**	**no**	**14**	**¡STOP! (patience=12)**

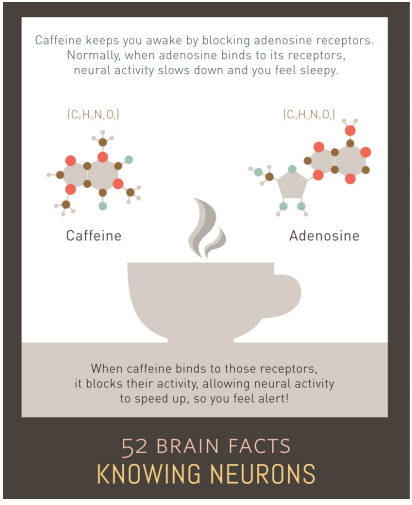
Concepto → aplicación: qué regularizador uso y para qué.

Técnica	Para qué problema	Entrada / Salida	Cuándo evitar
Dropout	MLP, CNN clásicas, modelos tabulares medianos.	Activaciones → activaciones con ruido.	Modelos muy pequeños, BN fuerte.
BatchNorm	CNN sobre imágenes, MLP densos.	Activaciones → activaciones normalizadas.	Batch $\leq$ 4, modelos preentrenados.
Weight decay	Cualquier red, casi siempre.	Pesos → término extra en la pérdida.	Modelos sub-ajustados (λ alto).
Early stop	Cualquier red, casi siempre.	Curva val → checkpoint final.	Dataset trivial, sin curva clara.
LR cosine	Red moderna con AdamW.	Iteración → lr instantáneo.	Datasets muy pequeños.

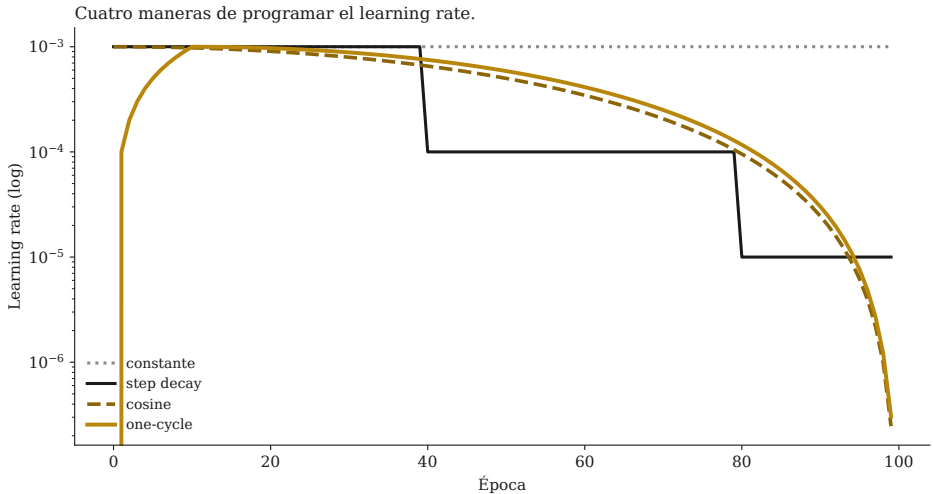
Idea clave. Combinar los cinco es el primer prototipo correcto del Bloque I.

Parte 3 / 5

Learning rate schedules



Cuatro maneras de programar el learning rate.



Cuándo usar cada schedule.

Schedule	Idea	Cuándo
Constante	lr fijo durante todo el entrenamiento.	Prototipos, datasets pequeños, primer experimento.
Step decay	Divide lr por 10 cada N épocas.	Histórico (CNN sobre ImageNet con SGD).
Cosine	Decae suavemente como medio coseno.	Defecto moderno con AdamW.
One-cycle	Warmup + cosine: sube y luego baja.	Para acelerar entrenamiento en pocas épocas.

Idea clave. Si te bloqueas con el lr, empieza con cosine. Es el menos malo en casi todos los problemas.

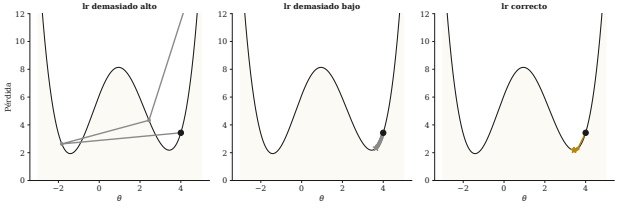
Programar cosine en PyTorch en cinco líneas.

```
from torch.optim.lr_scheduler import CosineAnnealingLR
opt = torch.optim.AdamW(m.parameters(), lr=1e-3,
weight_decay=1e-4)
sched = CosineAnnealingLR(opt, T_max=epochs)
for ep in range(epochs):
    for xb, yb in train_loader:
        opt.zero_grad()
        loss_fn(m(xb), yb).backward()
        opt.step()
    sched.step()
```

Idea clave. Esta línea sola sube la AUC de validación 2–5 % sin tocar el modelo.

El learning rate controla el balance exploración-explotación.

El learning rate controla la trayectoria de optimización.



lr alto = varianza alta. ⚠️ Los parámetros saltan entre regiones de la superficie. Puede escapar de mínimos locales, pero también divergir.

lr bajo = sesgo alto. 🎯 Los parámetros apenas se mueven desde la inicialización. Riesgo de quedarse en un mínimo local cercano al punto de partida.

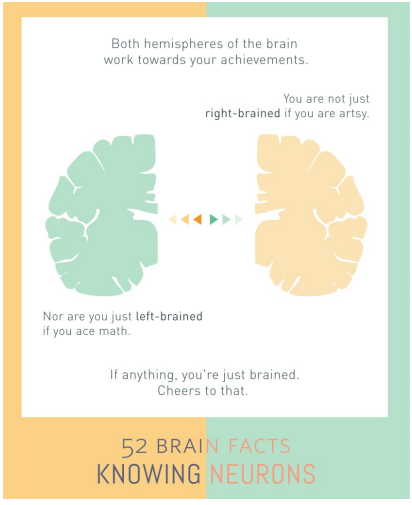
lr correcto. 🚀 Paso suficiente para explorar al inicio, cada vez más fino para explotar. Esto es lo que hace un **schedule cosine**.

Warmup. 🔥 Al inicio, las estadísticas de BatchNorm no han estabilizado. Subir el lr gradualmente evita explosiones tempranas.

Idea clave. Exploración temprana (lr alto) y explotación tardía (lr bajo).

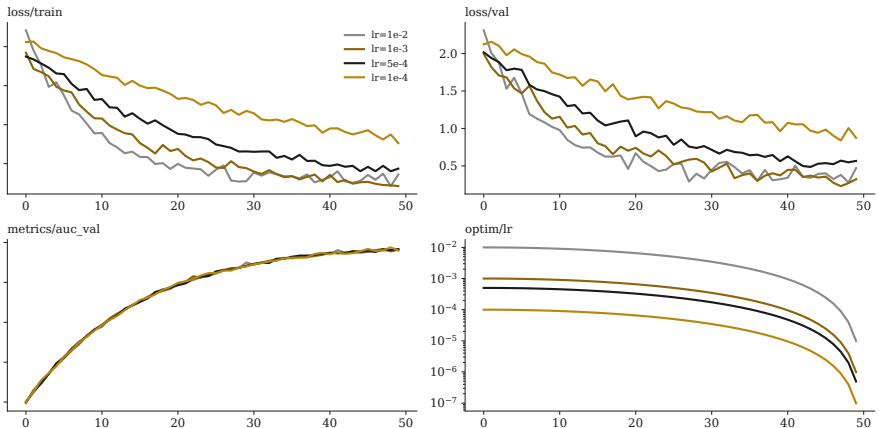
Parte 4 / 5

Dashboards y trackers de experimentos



Cuatro paneles que debes ver siempre.

Tablero de experimentos: 4 corridas, 4 paneles.



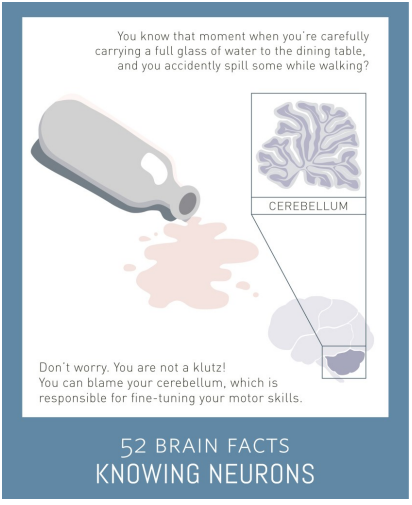
Qué herramienta para qué aplicación.

Herramienta	Para qué	Coste	Cuándo
TensorBoard	Logging local básico (loss, scalars, imágenes).	Gratis	Sesiones individuales, casi cualquier proyecto PyTorch.
Weights & Biases	Comparar experimentos en cloud con dashboard rico.	Gratis < 100GB	Trabajo en equipo, búsquedas de hiperparámetros.
MLflow	Tracking + model registry + serving open-source.	Gratis (self-host)	Producción empresarial con varios usuarios.
Comet	Alternativa cloud a W&B con énfasis en NLP.	Gratis < 100GB	Equipos NLP / Hugging Face.

Idea clave. En el proyecto AI-first todo equipo elige uno. Sin tracking no hay reproducibilidad ni defensa oral.

Parte 5 / 5

Caso completo: receta exprés

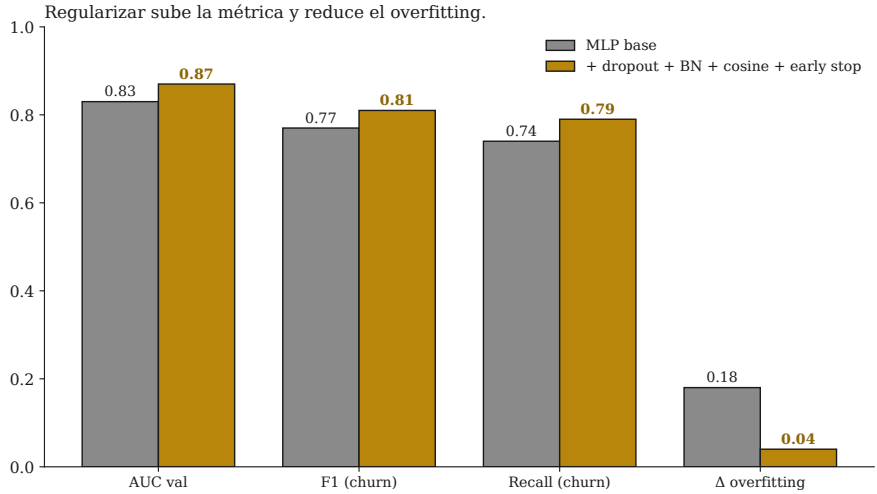


Receta exprés del "primer prototipo correcto".

1. **Arquitectura.** 🧠 2–3 capas ocultas, ancho 32–128, ReLU. Empieza pequeño.
2. **Optimizador.** 🚀 **AdamW** con lr = 10^{-3} y weight_decay = 10^{-4}.
3. **Schedule.** ⏱️ **CosineAnnealingLR** con `T_max=epochs`.
4. **Regularización.** 🛡️ **Dropout** $p \in \{0{,}1, 0{,}3\}$, **BatchNorm** tras cada Linear (excepto última capa).
5. **Stopping.** ✅ **Early stopping** con `patience=10` sobre val.
6. **Tracking.** 📊 W&B (o MLflow). Loguea `loss/train`, `loss/val`, `metric/auc`, `optim/lr`.
7. **Validación.** 🎯 5-fold antes de declarar nada.

Idea clave. Esta receta es el "primer prototipo correcto"del Bloque I. Aprended bien hoy y copiad cuatro veces en el semestre.

Mismo dataset, mismo modelo, regularización añadida.



Lo que el alumno se lleva a casa de este día.

Concepto 1. 🎲 Dropout y 📚 BatchNorm son los regularizadores por defecto.

Concepto 2. ⏱️ Cosine annealing del lr mejora casi todo entrenamiento sin coste adicional.

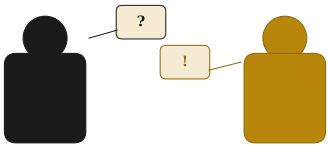
Concepto 3. ✅ Early stopping cuesta cero y previene la mitad del overfitting.

Concepto 4. 👁️ Sin dashboard no hay reproducibilidad ni decisiones defendibles.

Para la práctica. Tomar el MLP del Día 02, aplicar la receta y comparar.

Próximo día. Día 04 (Tema 2): CNN para visión y retail.

Pausa para debatir: ¿Mata la receta exprés la creatividad técnica?



La pregunta. 💡 Cosine + Dropout + BatchNorm + EarlyStopping es .[el] primer prototipo correcto". ¿Es educativo o cierra la exploración?

Posición A. ⚠️ Mata: sin probar SGD puro no se entiende por qué Adam funciona.

Posición B. ✅ No mata: la receta es el punto de partida, no la meta. Lo que pierde es tiempo no aprendizaje.

Reglas. 5 minutos, dos turnos de palabra por bando, móviles boca abajo. Yo modero, no participo.

Cierre del Tema 1.

Próxima clase: convolución, pooling, transfer learning.

Eduardo C. Garrido-Merchán

ecgarrido@comillas.edu