Clase 113 — Optimizadores: Momentum, Nesterov, AdaGrad, RMSProp, Adam, AdamW (+ Lion, Sophia)

Parte: 2 — Deep Learning · Fuente: Géron, cap. 11 § Faster Optimizers + papers Lion (Chen et al. 2023), Sophia (Liu et al. 2023). Duración estimada: 80 min.

Clase 113 — Optimizadores: Momentum, Nesterov, AdaGrad, RMSProp, Adam, AdamW (+ Lion, Sophia)

> *Parte: 2 — Deep Learning · Fuente: Géron, cap. 11 § Faster Optimizers + papers Lion (Chen et al. 2023), Sophia (Liu et al. 2023). Duración estimada: 80 min.*

Objetivo

Conocer la evolución de los optimizadores —SGD → Momentum → Nesterov → AdaGrad → RMSProp → Adam → AdamW— entendiendo qué problema resuelve cada uno. Aplicar los optimizadores 2023+ (Lion, Sophia) que están reemplazando a Adam en LLMs grandes por mejor performance y memoria. Saber elegir según contexto (Adam para casi todo, SGD+momentum para visión clásica, Lion para LLMs).

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Explicar la fórmula de cada uno: SGD (w ← w - η·g), Momentum (acumulación), Nesterov (lookahead), Adam (mom 1er + 2do orden).
- Diferenciar Adam vs AdamW — la corrección de weight decay que Loshchilov & Hutter (2019) demostraron esencial.
- Usar Lion (tf.keras.optimizers.Lion en Keras 3+) con LR 3-10× más bajo que Adam.
- Reconocer cuándo SGD+Momentum supera a Adam: visión clásica con datasets grandes (ImageNet), donde el modelo final generaliza mejor.
- Inspeccionar y entender los hiperparámetros beta_1, beta_2, epsilon, weight_decay.

Temas

- SGD vanilla y por qué es lento en cañones (zigzaguea).
- Momentum (Polyak 1964): acelera en direcciones consistentes.
- Nesterov (1983): "miro hacia adelante" antes de calcular gradiente.
- AdaGrad: tasa adaptativa por parámetro; bueno para sparse data, malo para LR que decae a 0.
- RMSProp (Hinton, sin publicar): suaviza AdaGrad con EMA.
- Adam (Kingma & Ba 2014): Momentum + RMSProp = caballito industrial.
- AdamW (Loshchilov & Hutter 2019): weight decay separado del gradiente.
- Complemento moderno: Lion (Chen et al. 2023, signo en lugar de gradient adaptive), Sophia (Liu et al. 2023, Hessian aproximada).

Versión profundizada — 2026

El tema moderno que vivía como complemento dentro de esta clase ahora tiene clase propia dedicada con patrón completo, ejercicios y homework:

- Clase 102b — Optimizadores modernos: Lion, Sophia, Schedule-Free

Definiciones y características

- SGD: w ← w - η · g. La base.
- Momentum: v ← β·v + g; w ← w - η·v. β típicamente 0.9.
- Nesterov: como momentum, pero calcula el gradiente en el punto adelantado.
- AdaGrad: s += g²; w ← w - η·g/√s. LR efectivo decrece para parámetros frecuentes.
- RMSProp: como AdaGrad pero con EMA: s ← β·s + (1-β)·g².
- Adam: combina momentum (1er momento) + RMSProp (2do momento) + bias correction.
- AdamW: aplica weight decay como una operación separada del gradiente (decoupled), no como L2.
- Lion: gradient sign + momentum. Menos memoria.
- beta_1: decay del primer momento (default 0.9).
- beta_2: decay del segundo momento (default 0.999 Adam / 0.99 Lion).
- epsilon: estabilidad numérica del √. Default 1e-7. En LLMs a veces 1e-8 / 1e-5.
- weight_decay: en AdamW, el coeficiente de "tirar pesos hacia 0" — regularización L2 implícita.

Dataset / recursos

- Fashion-MNIST + un modelo razonable.
- Librerías: tensorflow, keras (Lion incluido en Keras 3+).

Ejercicios

1. Comparar 5 optimizadores: SGD(0.01), SGD+Momentum(0.9), Adam(1e-3), AdamW(1e-3, wd=1e-2), Lion(1e-4, wd=0.1). Mismo modelo, mismo dataset, 20 épocas. Graficar val_loss.
2. Tuning del LR: para Adam y Lion, hacer un sweep de LR [1e-5, 1e-2] log. Encontrar el LR óptimo de cada uno. Verificar que el de Lion es ~5× más chico.
3. AdamW vs Adam con L2: comparar Adam + keras.regularizers.L2(1e-2) en cada capa vs AdamW con weight_decay=1e-2. AdamW gana en val_loss.
4. Inspección de buffer: imprimir optimizer.variables. Adam tiene m y v por parámetro; Lion solo m. Verificar memoria total.
5. LR alto + Momentum: SGD con LR=0.1 explota; SGD+Momentum(0.9) con LR=0.1 puede funcionar. Probar.

Homework verificable

Sobre Fashion-MNIST + un MLP [300, 100, 10]:

1. Encontrar el LR óptimo para 3 optimizadores: SGD, AdamW, Lion (sweep de 5 valores cada uno).
2. Con el LR óptimo, entrenar 30 épocas y reportar val_accuracy.
3. Comparar wall time y memoria.

Criterio de aceptación: AdamW debe igualar o superar a Adam por ≥ 0.3 pp en val_accuracy; Lion con buen LR debe ser competitivo y consumir menos memoria del optimizer.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
Cambiar de Adam a Lion sin cambiar LR	Lion necesita LR mucho más chico. Fix: div
Adam con weight_decay en Keras viejo	Lo aplica como L2 (mal). Fix: usar AdamW.
SGD plano sin schedule en redes profundas	Convergencia lentísima. Fix: SGD + Momentu

epsilon=1e-7 produce inestabilidad en bf16	Para mixed precision en LLMs, epsilon=1e-5
Asumir que Adam es siempre mejor	En visión clásica con suficiente data, SGD

Preguntas frecuentes

¿Adam o AdamW por default?

AdamW siempre que uses weight decay. Adam con weight_decay > 0 está mal implementado en muchos frameworks antiguos.

¿Lion en producción ya?

Sí, desde 2023. Google lo usa en internamente. Estable y bien probado.

¿Cuándo SGD gana a Adam?

Cuando podés permitirte LR + momentum + cosine schedule bien tuneados, sobre datasets grandes (ImageNet). El modelo final generaliza ~0.5-1 pp mejor. Pero requiere más tuning.

¿epsilon cuándo lo toco?

Casi nunca con fp32. En mixed precision (bf16/fp16), subir a 1e-4 para estabilidad.

¿beta_2 por qué 0.999 en Adam y 0.99 en Lion?

Adam el 2do momento debe ser estable (decay muy lento). Lion no tiene 2do momento — beta_2 define cómo se mezcla m_{t-1} con g_t en el lookahead, similar a Nesterov.

Referencias

- Géron, cap. 11 — Faster Optimizers.
- Kingma & Ba (2014), Adam, ICLR.
- Loshchilov & Hutter (2019), Decoupled Weight Decay Regularization (AdamW), ICLR.
- Chen et al. (2023), Symbolic Discovery of Optimization Algorithms (Lion), NeurIPS.
- Liu et al. (2023), Sophia: A Scalable Stochastic Second-order Optimizer.
- Keras optimizers.

Siguiente clase

Clase 114 — Optimizadores modernos: Lion, Sophia, Schedule-Free

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb