Optimizadores y Dinámica del Aprendizaje

Cómo y por qué cambian los pasos del descenso de gradiente

Cesar Garcia

2025

Introducción

- Entender **qué hace realmente un optimizador**
- Ver por qué SGD puro puede ser lento o inestable
- Introducir Momentum, RMSProp y Adam
- Comparar **dinámica de aprendizaje**, no solo resultados finales
- Preparar experimentos controlados en el notebook

Si dos modelos tienen la misma arquitectura, ¿por qué pueden aprender de forma tan distinta?

Recordatorio: SGD

El punto de partida

SGD actualiza los parámetros como:

$$\theta \leftarrow \theta - \eta, \nabla_\theta L$$

Propiedades:

- simple
- transparente
- sensible al learning rate

SGD define la **referencia base**.

¿Qué problemas aparecen cuando el gradiente es muy ruidoso o cambia de dirección?

El problema de los valles

Geometría de la pérdida

Las superficies de pérdida reales:

- no son esféricas
- tienen valles alargados
- presentan direcciones con distinta curvatura

SGD puede:

- oscilar
- avanzar lentamente
- desperdiciar iteraciones

¿Qué ocurre si avanzamos demasiado en una dirección y muy poco en otra?

Idea central de los optimizadores

Más que un paso fijo

Un optimizador decide:

- **dirección** del paso
- **tamaño** del paso
- cómo usar información del pasado

Todos parten del mismo gradiente, pero **lo interpretan distinto**.

¿Por qué podría ser útil "recordar" gradientes anteriores?

Momentum

Acumular dirección

Momentum introduce una variable de velocidad:

$$v_t = \beta v_{t-1} + \nabla_\theta L$$

$$\theta \leftarrow \theta - \eta v_t$$

Efecto:

- suaviza oscilaciones
- acelera en direcciones consistentes

¿Por qué esto ayuda especialmente en valles alargados?

Intuición de Momentum

Analogía física

Momentum se parece a:

- una bola rodando cuesta abajo
- que acumula inercia

Gradientes ruidosos:

- se cancelan Gradientes consistentes:
- se refuerzan

¿Qué pasaría si el momentum fuera demasiado grande?

RMSProp

Escalas adaptativas

RMSProp mantiene un promedio de gradientes al cuadrado:

$$s_t = \beta s_{t-1} + (1-\beta)(\nabla_\theta L)^2$$

Actualización:

$$\theta \leftarrow \theta - \frac{\eta}{\sqrt{s_t} + \epsilon} \nabla_\theta L$$

¿Por qué tendría sentido reducir el paso en direcciones con gradientes grandes?

Intuición de RMSProp

Normalizar el aprendizaje

RMSProp:

- reduce pasos donde el gradiente es grande
- aumenta pasos donde el gradiente es pequeño

Resultado:

- entrenamiento más estable
- menos sensibilidad a la escala

¿Qué tipo de problemas se benefician de este comportamiento?

Adam

Combinando ideas

Adam combina:

- Momentum (promedio del gradiente)
- RMSProp (promedio del gradiente al cuadrado)

Mantiene:

- primer momento (media)
- segundo momento (varianza)

¿Por qué combinar velocidad y escala puede ser ventajoso?

Adam Optimizer (Update Rule)

Given gradient $g_t = \nabla_\theta L(\theta_t)$:

First moment (momentum): $m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$

Second moment (RMS): $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$

Bias correction: $\hat{m}_t = \dfrac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \dfrac{v_t}{1-\beta_2^t}$

Parameter update: $\theta_{t+1} = \theta_t - \alpha \dfrac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}$

Por qué Adam funciona tan bien

Práctica moderna

Adam suele:

- converger rápido
- requerir menos ajuste fino
- funcionar bien "out of the box"

Pero:

- no siempre generaliza mejor
- puede ocultar problemas de entrenamiento

¿Es siempre buena idea usar el optimizador "más avanzado"?

Comparación conceptual

Qué cambia realmente

Optimizador	Idea clave
SGD	Paso fijo con gradiente actual
Momentum	Acumula dirección
RMSProp	Ajusta escala
Adam	Dirección + escala

¿Cuál de estas ideas te parece más importante para aprender mejor?

Optimización vs generalización

No son lo mismo

Un optimizador puede:

- minimizar la pérdida rápido
- pero no generalizar mejor

La velocidad de convergencia $\neq$ calidad de la solución.

¿Por qué un entrenamiento "más rápido" no garantiza mejor modelo?

Qué no cambia

Disciplina constante

Independientemente del optimizador:

- el bucle de entrenamiento es el mismo
- la validación sigue siendo diagnóstico
- las curvas siguen siendo la herramienta

Cambiar optimizador **no cambia el método**.

¿Qué errores del entrenamiento no se arreglan cambiando de optimizador?

Puente a la práctica

Qué haremos en el notebook

En el notebook:

- usaremos el **mismo modelo**
- el **mismo dataset**
- el **mismo training loop**

Solo cambiaremos el optimizador para observar la dinámica.

¿Qué diferencias esperas ver en las curvas?

Idea clave de la sesión

Elegir con criterio

Los optimizadores:

- no son magia
- no corrigen errores conceptuales
- modifican **cómo** se aprende

Primero entiende el proceso, luego optimiza la dinámica.

¿En qué situación preferirías SGD sobre Adam?