Modelos de Deep Learning

Regularización

Universidad ORT Uruguay

29 de Setiembre, 2025

Regularización: penalizando weights

- Idea: penalizar soluciones con weights extremadamente grandes
- $\mathbf{W}$ = tensor de weights del modelo (o de un layer del modelo)
- $J_{\mathbf{T}}(\mathbf{W})$ costo empírico (p. ej., entropía cruzada)
- Utilizamos un término de regularización $R(\mathbf{W})$ y penalizamos el costo sumando dicho término al costo original:

$$J_{\text{reg}}(\mathbf{W}) = J_{\mathbf{T}}(\mathbf{W}) + \lambda\, R(\mathbf{W}).$$

- Objetivo: $R(\mathbf{W})$ grande corresponde a una peor solución
- $\lambda \geq 0$ es un escalar que pondera ambos términos.
- $\lambda = 0$ la regularización no tiene efecto; $\lambda \to \infty$ *conocimiento a priori*.

Regularización como inferencia MAP

- Equivalente a realizar inferencia de máximo a posteriori (MAP) en lugar de *máxima verosimilitud*.

- Combinamos una distribución a priori sobre los pesos $p(\mathbf{W})$ con una función de verosimilitud estándar sobre los datos $\mathbf{T}$:

$$\begin{aligned}\widehat{\mathbf{W}} &= \arg\min_{\mathbf{W}} \left\{ -\ln \left[p\left(\mathbf{T} \mid \mathbf{W}\right) p(\mathbf{W}) \right] \right\} \\ &= \arg\min_{\mathbf{W}} \left\{ -\ln V_{\mathbf{T}}\left(\mathbf{W}\right) - \ln p(\mathbf{W}) \right\}\end{aligned}$$

- La regularización equivale a asumir una distribución a priori $p(\mathbf{W})$.

Regularización Ridge como a priori Gaussiana

- Recordar la regularización Ridge (o ℓ_2) de los pesos:

$$R(\mathbf{W}) = \|\mathbf{W}\|^2 = \sum (\text{entradas de } \mathbf{W})^2$$

- La distribución Gaussiana isotrópica sobre los pesos es:

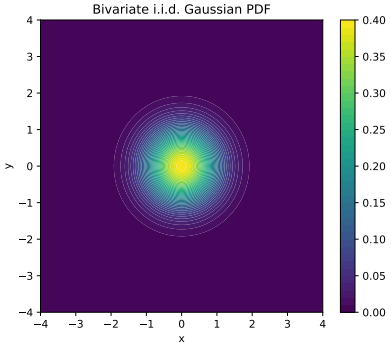
$$\mathbf{W} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}) \qquad p(\mathbf{W}) = (\text{cte}) \cdot \exp -\tfrac{1}{2\sigma^2}\|W\|^2$$

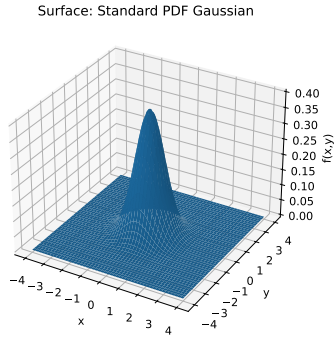
- El estimador MAP minimiza

$$-\left[\ln V_{\mathbf{T}}(\mathbf{W}) + \ln p(\mathbf{W})\right] = J_{\mathbf{T}}(\mathbf{W}) + \tfrac{1}{2\sigma^2}\|\mathbf{w}\|^2 + \text{cte.}$$

- Por lo tanto, minimizar $J_{\mathbf{T}}(\mathbf{W}) + \lambda\|\mathbf{w}\|^2$ es equivalente a MAP con

$$\lambda = \tfrac{1}{2\sigma^2} \quad \text{y } J = -\log \text{ verosimilitud.}$$


Bivariate i.i.d. Gaussian PDF
x
y


Surface: Standard PDF Gaussian
x
y
f(x,y)

Regularización LASSO como a priori Laplaciana

- Recordar la regularización LASSO (o ℓ_1) de los pesos:

$$R(\mathbf{W}) \;=\; \|\mathbf{W}\|_1 \;=\; \sum \big|\text{entradas de } \mathbf{W}\big|.$$

- Un prior Laplace (iid) sobre los pesos es:

$$p(\mathbf{W}) = (\text{cte}) \exp\left(-\tfrac{1}{b}\|\mathbf{W}\|_1\right) \Longrightarrow \ln p(\mathbf{W}) = \text{cte} - \tfrac{1}{b}\,\|\mathbf{W}\|_1.$$

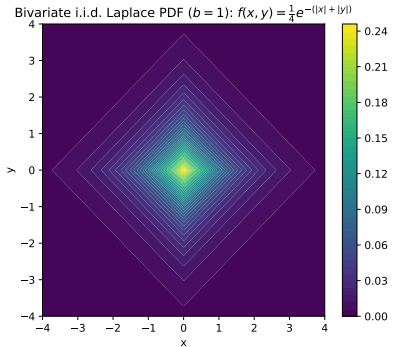
- El estimador MAP minimiza

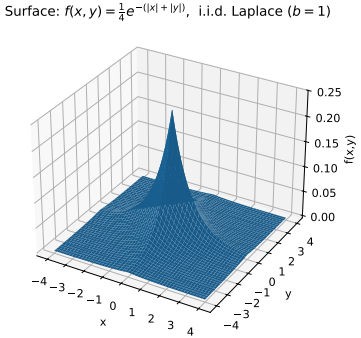
$$-[\ln V_{\mathbf{T}}(\mathbf{W}) + \ln p(\mathbf{W})] \;=\; J_{\mathbf{T}}(\mathbf{W}) \;+\; \frac{1}{b}\,\|\mathbf{W}\|_1 \;+\; \text{cte}.$$

- Por lo tanto, minimizar $J_{\mathbf{T}}(\mathbf{W}) + \lambda\|\mathbf{W}\|_1$ es equivalente a MAP con

$$\lambda \;=\; \tfrac{1}{b} \quad \text{y} \quad J_{\mathbf{T}} \;=\; -\log \text{ verosimilitud}.$$

- Efecto inductivo: LASSO induce esparsidad en los pesos.


Bivariate i.i.d. Laplace PDF ($b = 1$): $f(x, y) = \frac{1}{4}e^{-(|x| + |y|)}$
x
y
0.24
0.21
0.18
0.15
0.12
0.09
0.06
0.03
0.00


Surface: $f(x, y) = \frac{1}{4}e^{-(|x| + |y|)}$, i.i.d. Laplace ($b = 1$)
x
y
f(x, y)
0.25
0.20
0.15
0.10
0.05
0.00

Intuición geométrica de la regularización

- En muchos casos el problema regularizado puede reescribirse como el original con restricción:

$$\begin{array}{ll} \arg\min_{\mathbf{W}} & J_{\mathbf{T}}(\mathbf{W}) \\ \text{sujeto a} & R(\mathbf{W}) \leq \mu \end{array}$$

  donde μ es inversamente proporcional a λ.

- **Regularización** ℓ_2: la solución queda dentro de una bola centrada en el origen (en 2D, *círculo*).

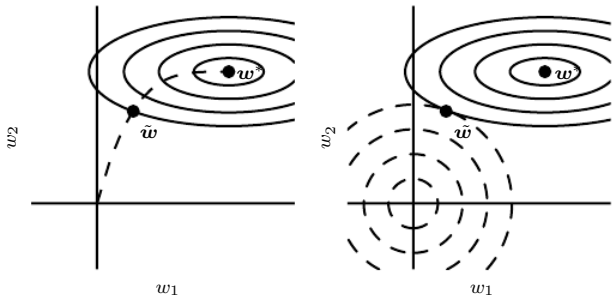
- **Regularización** ℓ_1: la solución queda dentro (o en los vértices) de un *politopo* centrado en el origen (en 2D, *rombo*); las soluciones *esparsas* aparecen en los vértices que intersectan los ejes.

Early Stopping

Early Stopping

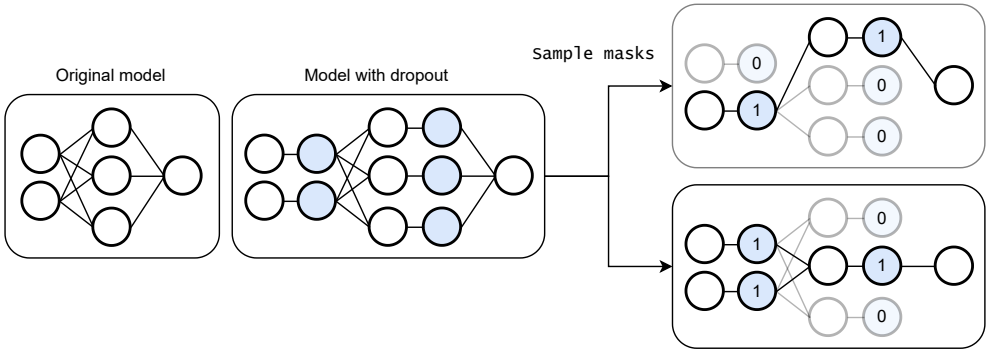
01: Inicializar $\mathbf{W}_{\text{best}} \leftarrow$ random
02: Inicializar $J_{\text{best}} \leftarrow +\infty$
03: Inicializar p paciencia
04: Inicializar $c \leftarrow 0$ contador paciencia
05:
06: Para cada epoch en epochs:
07: Entrenar red con datos de entrenamiento
08: $J \leftarrow$ evaluar red con datos de validación
09: Si $J < J_{\text{best}}$:
10: $J_{\text{best}} \leftarrow J$
11: $\mathbf{W}_{\text{best}} \leftarrow$ pesos actuales de la red
12: $c \leftarrow 0$
13: Sino:
14: $c \leftarrow c + 1$
15: si $c \geq p$: break
16: Establecer pesos de la red $\leftarrow \mathbf{W}_{\text{best}}$

Early stopping como mecanismo de regularización



Early stopping tiene un efecto regularizador similar a ℓ_2, pero es más efectivo porque determina la cantidad correcta de regularización, sin intervenir directamente sobre $\mathbf{W}$, mientras que ℓ_2 suele requerir muchos experimentos de entrenamiento con diferentes valores de λ.

Dropout: idea



Original: $y = (\text{FC} \circ \text{FC})(\mathbf{x})$

Con Dropout: $y = (\text{FC} \circ \text{Dropout} \circ \text{FC} \circ \text{Dropout})(\mathbf{x})$

Dropout: definición

- $\mathbf{X} \sim (n, c)$ *mini-batch* de activaciones internas del modelo
- Con n elementos en el mini-batch y c características.
- Muestreamos una matriz binaria $\mathbf{M} \sim \text{Binary}(n, c)$

$$M_{ij} \sim \text{Bern}(p).$$

- La salida de la capa se obtiene enmascarando la entrada:

$$\text{Dropout}(\mathbf{X}) = \mathbf{M} \odot \mathbf{X}.$$

- El único hiperparámetro es $r \in [0, 1]$ y no tiene parámetros entrenables.

Dropout: entrenamiento vs. inferencia

- m capas *dropout*. Sea $\mathbf{M}_i$ la máscara en la i-ésima capa y

$$p(\mathbf{M}_1, \ldots, \mathbf{M}_m) = \prod_{i=1}^{m} p(\mathbf{M}_i)$$

  la distribución (independiente) sobre el conjunto de máscaras.

- Sea $f(\mathbf{X}; \mathbf{M})$ la salida determinista del modelo para máscaras $\mathbf{M} \sim p(\mathbf{M})$ fijas.

- En inferencia se reemplaza el efecto de dropout por su valor esperado:

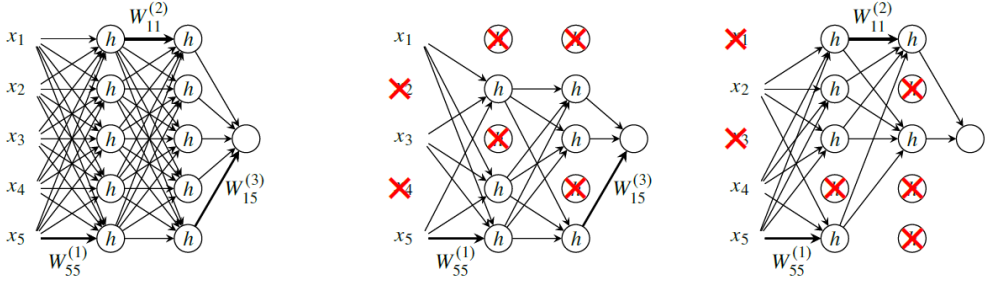
$$f(\mathbf{X}) = \begin{cases} f(\mathbf{X}; \mathbf{M}), & \mathbf{M} \sim p(\mathbf{M}) \quad \texttt{[training]} \\ \mathbb{E}_{p(\mathbf{M})}[f(\mathbf{X}; \mathbf{M})], & \texttt{[inference]} \end{cases}$$

- Aproximación: $\mathbb{E}_{p(\mathbf{M})}\left[\text{Dropout}(\mathbf{X})\right] = r\mathbf{X}$

Dropout

Entrenamiento

Máscara que apaga aleatoriamente neuronas de una capa con probabilidad $1 - r$



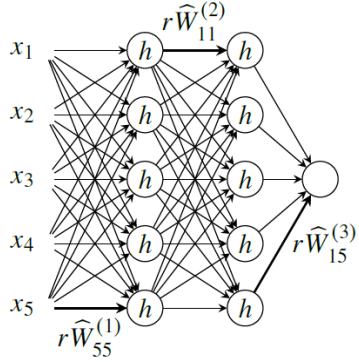
(a) A standard neural network

(b) Two sub-networks

Dropout

Predicción

Se multiplican los pesos por la probabilidad r



Batch Normalization (BN): estadísticos del mini-batch

- $\mathbf{X} \sim (n, c)$ salida de capa intermedia, $n =$ mini-batch y $c =$ características.
- Se normaliza **cada característica** (cada columna de $\mathbf{X}$) a media cero y varianza unitaria, usando sólo el mini-batch.

$$\text{Media de la columna } j: \quad \mu_j = \frac{1}{n}\sum_i X_{ij}$$

$$\text{Varianza de la columna } j: \quad \sigma_j^2 = \frac{1}{n}\sum_i (X_{ij} - \mu_j)^2$$

- Normalización (por columnas):

$$\mathbf{X}' = \frac{\mathbf{X} - \boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2 + \varepsilon}}$$

Definición de la capa BN y uso típico

- BN sobre el mini-batch $\mathbf{X} \sim (n, c)$:

$$\mathrm{BN}(\mathbf{X}) \;=\; \alpha \left(\frac{\mathbf{X} - \mu}{\sqrt{\sigma^2 + \varepsilon}} \right) + \beta,$$

  donde μ, σ^2 se computan como en la diapositiva anterior y $\alpha \sim (c),\ \beta \sim (c)$ son **parámetros entrenables**.

- Uso habitual:

$$\mathbf{H} = (\mathrm{ReLU} \circ \mathrm{BN} \circ \mathrm{Linear})(\mathbf{X})$$

  es decir, BN entre el componente lineal y la no linealidad.

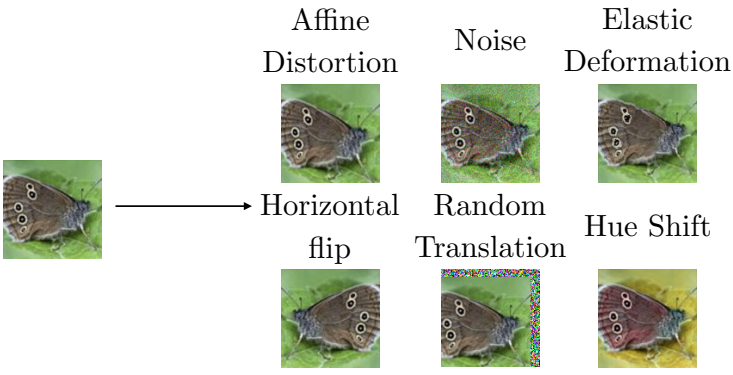
BN en inferencia

- Para tener predicciones deterministas y que no dependan de las estadísticas de cada lote, la capa de BN usa una media y varianza agregadas (o globales) que se van estimado durante el entrenamiento.

- Se suelen calcular usando un moving average (promedio móvil).

Regularización: aumento de los datos

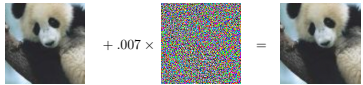
Transformación de imágenes



(Goodfellow 2016)

Regularización: otros mecanismos

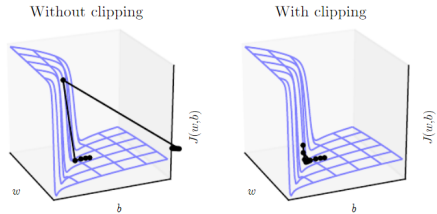
Generación de datos *adversarios*



Clipping del gradiente

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \eta \, \text{clip}(\nabla J(\mathbf{W}_t))$$

$$\text{clip}(\boldsymbol{g}) = \begin{cases} \boldsymbol{g} & \text{si } \|\boldsymbol{g}\| < \nu \\ \dfrac{\boldsymbol{g}}{\|\boldsymbol{g}\|} \nu & \text{si no} \end{cases}$$



Ruido en los coeficientes o en $\mathbf{y}$ (*label smoothing*)

Bibliografía

- Deep Learning. Ian Goodfellow, Yoshua Bengio, Aaron Courville. MIT Press, 2016.
- Prince, Simon JD. Understanding deep learning. MIT press, 2023.
- Bishop, Christopher M., and Hugh Bishop. Deep learning: Foundations and concepts. Springer Nature, 2023.
- Scardapane, Simone. "Alice's Adventures in a Differentiable Wonderland–Volume I, A Tour of the Land." arXiv preprint arXiv:2404.17625 (2024).