Centro de Investigación en Matemáticas (CIMAT)
Unidad Monterrey

Análisis de Datos Multimodales para MIR

Tarea 1

Introducción al Deep Learning

Diego Paniagua Molina
diego.paniagua@cimat.mx

22 de Septiembre del 2025

Problema 1

Calcula lo siguiente:

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} + \begin{pmatrix} 7 & 9 \end{pmatrix}$$

Usa broadcasting de tal forma que la operación esté bien definida. Antes, averigua y describe qué es broadcasting, en el contexto de numpy.

SOLUCIÓN

Tras revisar la documentacion oficial de la libreria `NumPy` se menciona que "el término **difusión** (broadcasting) describe cómo `NumPy` trata las matrices con diferentes formas durante las operaciones aritméticas. Sujeto a ciertas restricciones, la matriz más pequeña se *difunde* a través de la matriz más grande para que tengan formas compatibles. La difusión proporciona un medio para vectorizar las operaciones de matriz, de modo que el bucle se produce en `C` en lugar de en `Python`. Lo hace sin crear copias innecesarias de los datos y, por lo general, da lugar a implementaciones algorítmicas eficientes. Sin embargo, hay casos en los que la difusión no es una buena idea, ya que conduce a un uso ineficiente de la memoria que ralentiza el cálculo" [1]. Y en principio existen algunas reglas clave:

1. Se comparan las formas de derecha a izquierda (ejes finales primero).
2. En cada eje, dos tamaños son compatibles si son iguales o si alguno es 1.
3. Si son compatibles, el tamaño resultante en ese eje es el máximo de ambos; el arreglo con tamaño 1 se expande virtualmente (no se duplica en memoria).
4. Si existe algún eje no compatible, la operación es inválida.

Entonces, para que la operación dada este bien definida, es decir, sumar el vector fila a cada fila del producto matricial, sean:

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} 7 & 9 \end{pmatrix}$$

Primero calculamos el producto de matrices, sea $\mathbf{R} = \mathbf{AB}$, tal que:

$$\begin{aligned} \mathbf{R} &= \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \\ &= \begin{pmatrix} 1 \cdot 0 + 2 \cdot 2 & 1 \cdot 1 + 2 \cdot 3 \\ 3 \cdot 0 + 4 \cdot 2 & 3 \cdot 1 + 4 \cdot 3 \end{pmatrix} \\ &= \begin{pmatrix} 4 & 7 \\ 8 & 15 \end{pmatrix} \end{aligned}$$

Ahora realizamos la suma $\mathbf{R} + \mathbf{c}$ con broadcasting, expandiendo el vector $\mathbf{c}$ a dos filas de modo que:

$$\begin{aligned} \mathbf{R} + \mathbf{c} &= \begin{pmatrix} 4 & 7 \\ 8 & 15 \end{pmatrix} + \begin{pmatrix} 7 & 9 \end{pmatrix} \\ &= \begin{pmatrix} 4+7 & 7+9 \\ 8+7 & 9+15 \end{pmatrix} \\ &= \begin{pmatrix} 11 & 16 \\ 15 & 24 \end{pmatrix} \end{aligned}$$

Por lo tanto:

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} + \begin{pmatrix} 7 & 9 \end{pmatrix} = \begin{pmatrix} 11 & 16 \\ 15 & 24 \end{pmatrix}$$

Problema 2

Considera las redes multicapa (con funciones de activación lineal) que se muestran en la Figura 1:

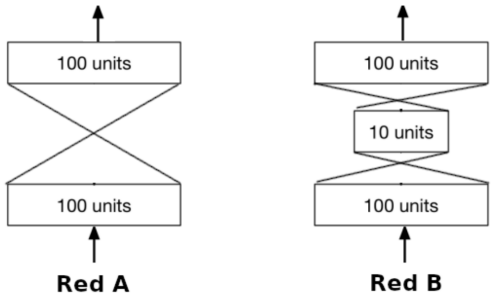

Figure 0.1: Esquema de dos redes neuronales.

1. Describe una ventaja (al menos) de la red A sobre la red B.

2. Describe una ventaja (al menos) de la red B sobre la red A.

SOLUCIÓN

De manera general, sabemos que en una red neuronal, cada capa aplica una transformación a sus datos de entrada. Si la función de activación f es lineal, la salida de una capa es simplemente una transformación lineal, específicamente es la multiplicación por una matriz de pesos $\mathbf{W}$ y suma de un sesgo $\mathbf{b}$:

$$\mathbf{y} = f(\mathbf{W}\mathbf{x} + \mathbf{b}) = \mathbf{W}\mathbf{x} + \mathbf{b} \tag{1}$$

Cuando apilamos múltiples capas lineales, la transformación total sigue siendo lineal. Por ejemplo, dos capas lineales consecutivas serían:

$$\mathbf{y} = \mathbf{W}_2(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2 = (\mathbf{W}_2\mathbf{W}_1)\mathbf{x} + (\mathbf{W}_2\mathbf{b}_1 + \mathbf{b}_2) \tag{2}$$

Esto se puede reescribir como una única capa lineal:

$$\mathbf{y} = \mathbf{W}_{\text{eq}}\mathbf{x} + \mathbf{b}_{\text{eq}} \tag{3}$$

donde $\mathbf{W}_{\text{eq}} = \mathbf{W}_2\mathbf{W}_1$ es la matriz de pesos equivalente.

A pesar de que ambas redes son equivalentes a una sola transformación lineal, su arquitectura interna impone una restricción muy importante sobre el tipo de matriz $\mathbf{W}_{\text{eq}}$ que pueden aprender. Esta restricción se relaciona con el **rango** de la matriz, que intuitivamente nos dice la dimensionalidad del espacio de salida que la transformación puede "*alcanzar*".

Ventaja de la Red A sobre la Red B

La Red A representa una única capa lineal con pesos que mapea una entrada $\mathbf{x} \in \mathbb{R}^{100}$ a una salida $\mathbf{y} \in \mathbb{R}^{100}$, siendo la transformación definida por una matriz de pesos $\mathbf{W}_{\text{A}} \in \mathbb{R}^{100\times100}$. El rango de la matriz resultante está limitado por la dimensión más pequeña de las matrices que se multiplican:

$$\text{rank}(\mathbf{W}_{\text{A}}) \leq \min(100, 100) = 100 \tag{4}$$

Ahora, la red B representa una composición de dos capas lineales, una capa de entrada que mapea $\mathbf{x} \in \mathbb{R}^{100}$ a una capa oculta $\mathbf{h} \in \mathbb{R}^{10}$, siendo la transformación definida por una matriz de pesos $\mathbf{W}_1 \in \mathbb{R}^{10\times100}$. Luego se tiene una capa de salida que mapea la capa oculta $\mathbf{h} \in \mathbb{R}^{10}$ a la salida final $\mathbf{y} \in \mathbb{R}^{100}$, siendo la transformación definida por una matriz de pesos $\mathbf{W}_2 \in \mathbb{R}^{100\times10}$. Tal que la matriz de pesos equivalentes para toda la red, $\mathbf{W}_\mathrm{B} \in \mathbb{R}^{100\times100}$ es:

$$\mathbf{W}_\mathrm{B} = \mathbf{W}_2\mathbf{W}_1 \tag{5}$$

Aquí, la capa intermedia de 10 unidades crea un *cuello de botella*, pues:

$$\mathrm{rank}(\mathbf{W}_\mathrm{B}) \leq \min(100, 10, 100) = 10 \tag{6}$$

Por lo tanto, la ventaja principal es que la Red A puede aprender cualquier transformación lineal de entrada a salida que tenga un rango de hasta 100 (suponiendo dimensiones de entrada/salida adecuadas). En contraste, la Red B está fundamentalmente limitada a aprender transformaciones con un rango máximo de 10. Por ejemplo, si la relación real entre los datos de entrada y salida es compleja y requiere una transformación de alto rango (por ejemplo, rango 50), la Red A podría aprenderla, mientras que la Red B sería incapaz de hacerlo. La Red B solo podría encontrar la mejor aproximación de rango 10 a esa transformación, perdiendo información en el proceso.

Ventaja de la Red B sobre la Red A

El cuello de botella de 10 unidades que mencionamos, obliga a la red B a aprender una representación comprimida y de baja dimensión de los datos de entrada. Para que la información pase por esta capa angosta y luego sea reconstruida, la red debe aprender a capturar las características más importantes de los datos, descartando el ruido o la información redundante.

Podemos ver este proceso como una forma de **regularización implícita**. Al restringir la complejidad del modelo, se reduce el riesgo de *sobre-ajuste*, lo que a menudo conduce a una mejor generalización en datos no vistos. También, la Red B es más ligera y rápida. Esto debido al número de parámetros (pesos + sesgos), asumiendo una entrada y salida de 100 unidades para una comparación justa:

Parámetros Red A:

- Transformación $(100 \to 100)$: $(100 \times 100) + 100 = 10,100$
- **Total:** 10,100 parámetros.

Parámetros Red B:

- Transformación 1 $(100 \to 10)$: $(100 \times 10) + 10 = 1,010$
- Transformación 2 $(10 \to 100)$: $(10 \times 100) + 100 = 1,100$
- **Total:** 2,110 parámetros.

Por lo tanto, la Red B es computacionalmente más eficiente (menos parámetros), lo que significa un entrenamiento más rápido y menor consumo de memoria. Más importante aún, su arquitectura impone una regularización que la obliga a aprender una representación de baja dimensión de los datos, lo que puede mejorar la generalización y hacerla más robusta al ruido, especialmente si la estructura intrínseca de los datos es de baja dimensión.

Problema 3

Considera la regularización l_1 de una función de costo L que asumimos es continuamente diferenciable:

$$\tilde{L}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = L(\mathbf{w}; \mathbf{X}, \mathbf{y}) + \alpha \sum_i |w_i|.$$

a) Como en clase, considera una aproximación de segundo orden alrededor de $\mathbf{w}^*$, y muestra que la aproximación regularizada de $\tilde{L}$

$$\hat{L}(\mathbf{w}) = L(\mathbf{w}^*) + \sum_i \left(\frac{1}{2} H_{ii} (w_i - w_i^*)^2 + \alpha |w_i| \right),$$

donde se asume que los datos están decorrelacionados (blanqueados), tal que H es una matriz diagonal con $H_{ii} > 0$.

b) Muestra que $\hat{L}$ se minimiza en

$$w_i = \text{signo}(w_i^*) \max \left\{ |w_i^*| - \frac{\alpha}{H_{ii}}, 0 \right\}$$

¿En qué casos tendremos soluciones sparse (donde $w_i = 0$)?

SOLUCIÓN

a)

Sabemos que para aproximar una función continuamente diferenciable podemos hacerlo a través de su **expansión en series de Taylor**. Entonces, si la función de costo $L(\mathbf{w})$ es dos veces diferenciable, su expansión en series de Taylor de segundo orden alrededor de un punto $\mathbf{w}_0$ es:

$$L(\mathbf{w}) \approx L(\mathbf{w}_0) + (\mathbf{w} - \mathbf{w}_0)^T \nabla L(\mathbf{w}_0) + \frac{1}{2} (\mathbf{w} - \mathbf{w}_0)^T \mathbf{H} (\mathbf{w} - \mathbf{w}_0) \tag{7}$$

donde $\mathbf{H} = \nabla^2 L(\mathbf{w}_0)$ es la matriz hessiana que nos indica la curvatura de la función.

Ahora sea $\mathbf{w}_0 = \mathbf{w}^*$ el punto que minimiza la función de costo L sin regularizar, esto significa que en un mínimo $\nabla L(\mathbf{w}^*) = \mathbf{0}$, sustituyendo en (7) obtenemos que:

$$L(\mathbf{w}) \approx L(\mathbf{w}_0) + \frac{1}{2} (\mathbf{w} - \mathbf{w}_0)^T \mathbf{H} (\mathbf{w} - \mathbf{w}_0) \tag{8}$$

Ahora, agregamos la regularización l_1 y de acuerdo a la definición del enunciado tenemos que la función de costo regularizada $\hat{L}$ con aproximación de segundo toma la siguiente forma:

$$\hat{L}(\mathbf{w}) = \left(L(\mathbf{w}^*) + \frac{1}{2} (\mathbf{w} - \mathbf{w}^*)^T \mathbf{H} (\mathbf{w} - \mathbf{w}^*) \right) + \alpha \sum_i |w_i| \tag{9}$$

Asumiendo que los datos están decorrelacionados implica que la matriz hessiana $\mathbf{H}$ se vuelve diagonal, es decir:

$$H_{ij} = 0 \quad \forall i \neq j$$

También se debe cumplir que $H_{ii} > 0$, por ende podemos expandir el termino cuadrático, tal que:

$$\begin{aligned}(\mathbf{w}-\mathbf{w}^*)^T\mathbf{H}(\mathbf{w}-\mathbf{w}^*) &= \sum_i \sum_j (w_i - w_i^*)H_{ij}(w_j - w_j^*) \\ &= \sum_i (w_i - w_i^*)H_{ii}(w_i - w_i^*) \\ &= \sum_i H_{ii}(w_i - w_i^*)^2 \end{aligned} \tag{10}$$

Sustituyendo (10) en (9) obtenemos que:

$$\begin{aligned}\hat{L}(\mathbf{w}) &= \left(L(\mathbf{w}^*) + \frac{1}{2}\sum_i H_{ii}(w_i - w_i^*)^2\right) + \alpha\sum_i |w_i| \\ &= L(\mathbf{w}^*) + \sum_i \left(\frac{1}{2}H_{ii}(w_i - w_i^*)^2 + \alpha|w_i|\right)\end{aligned} \tag{11}$$

Por lo tanto, queda demostrado que:

$$\boxed{\hat{L}(\mathbf{w}) = L(\mathbf{w}^*) + \sum_i \left(\frac{1}{2}H_{ii}(w_i - w_i^*)^2 + \alpha|w_i|\right)}$$

b)

Ya que $\hat{L}$ es separable podemos enfocarnos en minimizar la función para un solo peso w_i, sea esta función:

$$f(w_i) = \frac{1}{2}H_{ii}(w_i - w_i^*)^2 + \alpha|w_i| \tag{12}$$

El termino $L(\mathbf{w}^*)$ es una constante y no afecta la posición del mínimo, es por ello que podemos ignorarlo. Tal que, buscamos:

$$\min_{w_i} f(w_i) = \min_{w_i}\left(\frac{1}{2}H_{ii}(w_i - w_i^*)^2 + \alpha|w_i|\right) \tag{13}$$

Para minimizar debemos derivar e igualar a cero, el problema aquí es la función valor absoluto $|w_i|$ que no es diferenciable en $w_i = 0$. Entonces analicemos por partes.

Para $w_i > 0$ tenemos que $|w_i| = w_i$, calculamos la derivada de f:

$$\frac{df}{dw_i} = H_{ii}(w_i - w_i^*)^2 + \alpha, \qquad \forall w_i > 0 \tag{14}$$

Igualando a cero:

$$H_{ii}(w_i - w_i^*)^2 + \alpha = 0 \quad \Longrightarrow \quad w_i = w_i^* - \frac{\alpha}{H_{ii}} \tag{15}$$

Siendo que si $w_i > 0$, se satisface que:

$$w_i^* - \frac{\alpha}{H_{ii}} > 0 \quad \Longrightarrow \quad w_i^* > \frac{\alpha}{H_{ii}} \tag{16}$$

Para $w_i < 0$ tenemos que $|w_i| = -w_i$, calculamos la derivada de f:

$$\frac{df}{dw_i} = H_{ii}(w_i - w_i^*)^2 - \alpha, \qquad \forall w_i < 0 \tag{17}$$

Igualando a cero:

$$H_{ii}(w_i - w_i^*)^2 - \alpha = 0 \quad \Longrightarrow \quad w_i = w_i^* + \frac{\alpha}{H_{ii}} \tag{18}$$

Siendo que si $w_i < 0$, se satisface que:

$$w_i^* + \frac{\alpha}{H_{ii}} < 0 \quad \Longrightarrow \quad w_i^* < -\frac{\alpha}{H_{ii}} \tag{19}$$

Por ultimo el caso en que $w_i = 0$, es decir, cuando w_i^* cae en el rango $-\frac{\alpha}{H_{ii}} \leq w_i^* \leq \frac{\alpha}{H_{ii}}$. Para resolver este problema es necesario calcular el subgradiente. El subgradiente de $|w_i|$ en $w_i = 0$ es el intervalo completo $[-1, 1]$. La condición para que $w_i = 0$ sea un mínimo es que el cero este contenido en el subgradiente $f(w_i)$ evaluado en $w_i = 0$. Calculamos el subgradiente de f:

$$\partial f(w_i) = H_{ii}(w_i - w_i^*) + \alpha \, \partial |w_i| \tag{20}$$

Evaluando en $w_i = 0$:

$$\begin{aligned} \partial f(0) &= H_{ii}(0 - w_i^*) + \alpha \, \partial |0| \\ &= -H_{ii} w_i^* + \alpha \, [-1, 1] \\ &= [-H_{ii} w_i^* - \alpha, -H_{ii} w_i^* + \alpha] \end{aligned} \tag{21}$$

Para que $w_i = 0$ sea el mínimo, el valor 0 debe pertenecer a este intervalo. Esto requiere que se cumplan dos condiciones simultáneamente:

1. El límite inferior del intervalo debe ser menor o igual a 0.

$$-H_{ii} w_i^* - \alpha \leq 0 \implies -H_{ii} w_i^* \leq \alpha \implies w_i^* \geq -\frac{\alpha}{H_{ii}} \tag{22}$$

2. El límite superior del intervalo debe ser mayor o igual a 0.

$$-H_{ii} w_i^* + \alpha \geq 0 \implies \alpha \geq H_{ii} w_i^* \implies w_i^* \leq \frac{\alpha}{H_{ii}} \tag{23}$$

Juntando ambas condiciones, encontramos que $w_i = 0$ es la solución si y solo si:

$$-\frac{\alpha}{H_{ii}} \leq w_i^* \leq \frac{\alpha}{H_{ii}} \tag{24}$$

De forma compacta:

$$|w_i^*| \leq \frac{\alpha}{H_{ii}} \tag{25}$$

Por lo tanto, uniendo los tres casos posibles:

$$w_i = \begin{cases} w_i^* - \dfrac{\alpha}{H_{ii}} & \text{si } w_i^* > \dfrac{\alpha}{H_{ii}} \\ w_i^* + \dfrac{\alpha}{H_{ii}} & \text{si } w_i^* < -\dfrac{\alpha}{H_{ii}} \\ 0 & \text{si } |w_i^*| \leq \dfrac{\alpha}{H_{ii}} \end{cases} \tag{26}$$

Para simplificar este resultado podemos construir la función **soft-thresholding** que es la expresión a la que se nos pide llegar. Primero, podemos generalizar los casos 1 y 3 a través de la función $\max(A, B)$, tal que:

$$\max\left(|w_i^*| - \frac{\alpha}{H_{ii}}, 0\right) \tag{27}$$

- Si $|w_i^*| > \dfrac{\alpha}{H_{ii}} \implies |w_i^*| - \dfrac{\alpha}{H_{ii}}$ es positivo. La función max devolverá este valor.
- Si $|w_i^*| \leq \dfrac{\alpha}{H_{ii}} \implies |w_i^*| - \dfrac{\alpha}{H_{ii}}$ es negativo o cero. La función max devolverá 0.

Esta expresión nos da la magnitud correcta, pero el resultado es siempre no negativo. Sin embargo, el caso 2 nos dice que si w_i^* es negativo, la solución w_i también debe ser negativa. Necesitamos una forma de devolverle el signo original a la solución. Aquí es donde entra la función signo(w_i^*):

- Si $w_i^* > 0 \implies$ signo(w_i^*) es $+1$.
- Si $w_i^* < 0 \implies$ signo(w_i^*) es -1.
- Si $w_i^* = 0 \implies$ signo(w_i^*) es 0.

Multiplicando (27) por la función signo:

$$w_i = \text{signo}(w_i^*) \max\left\{|w_i^*| - \frac{\alpha}{H_{ii}}, 0\right\} \tag{28}$$

Verifiquemos que funciona para todos los casos:

- Si $w_i^* > \dfrac{\alpha}{H_{ii}} \implies w_i = (+1)\left(w_i^* - \dfrac{\alpha}{H_{ii}}\right) = w_i^* - \dfrac{\alpha}{H_{ii}}$
- Si $w_i^* < -\dfrac{\alpha}{H_{ii}} \implies w_i = (-1)\left(-w_i^* - \dfrac{\alpha}{H_{ii}}\right) = w_i^* + \dfrac{\alpha}{H_{ii}}$
- Si $|w_i^*| \leq \dfrac{\alpha}{H_{ii}} \implies w_i = 0 \cdot 0 = 0$

Por lo tanto, queda demostrado que $\hat{L}$ se minimiza en:

$$\boxed{w_i = \text{signo}(w_i^*) \max\left\{|w_i^*| - \frac{\alpha}{H_{ii}}, 0\right\}}$$

Tendremos resultados **sparse** (escasos) cuando el peso óptimo w_i es exactamente cero. Como se demostró en el análisis anterior, esto ocurre si y solo si el coeficiente del modelo no regularizado, w_i^*, se encuentra dentro de la región definida por:

$$|w_i^*| \leq \frac{\alpha}{H_{ii}}$$

Esta desigualdad es la clave de la capacidad de la regularización $l1$ para generar escasez, actuando en esencia como un mecanismo de selección de características. Esto debido a que un coeficiente w_i se anula si su magnitud en el modelo no regularizado, $|w_i^*|$, que representa la contribución de la característica a la reducción del error, no es lo suficientemente grande como para superar un umbral. Dicho umbral, $\frac{\alpha}{H_{ii}}$, está determinado por dos factores:

1. La fuerza de regularización α: Un valor mayor de α amplía esta "*zona de anulación*", promoviendo soluciones más escasas al imponer una penalización más severa a los coeficientes no nulos.

2. La curvatura de la función de costo H_{ii}: Si la función de costo es relativamente plana con respecto a un peso (H_{ii} pequeño), el umbral es más grande, lo que facilita la eliminación de dicho peso, ya que su impacto en la función de costo es menor. Por el contrario, si la función de costo es muy sensible al peso (H_{ii} grande), el umbral es estrecho y es más probable que el coeficiente se conserve.

Por lo tanto, la regularización $l1$ no solo encoge los coeficientes como lo hace la regularización $l2$, sino que realiza una forma de "*umbral suave*" (**soft-thresholding**) que elimina activamente las características cuya contribución al modelo no supera el costo impuesto por la penalización. Este comportamiento es una consecuencia directa de la no diferenciabilidad de la norma $l1$ en el origen, lo que permite que la solución óptima se sitúe exactamente en un eje donde uno o más coeficientes son cero.

Problema 4

Considera un problema de clasificación multiclase y una red neuronal densamente conectada con una capa oculta, como se muestra en la Figura 2. Considera también la función sigmoide como activación de las unidades ocultas, la función softmax para las estimaciones en la capa de salida y cross-entropy como función de costo.

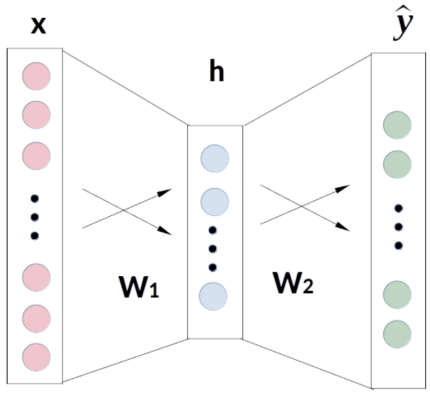

Figure 0.2: Red neuronal densamente conectada con una sola capa oculta.

a) Muestra que softmax es invariante a traslaciones (constantes) del vector de entrada, es decir, para cualquier vector $\mathbf{x}$ y cualquier constante c:

$$\text{softmax}(\mathbf{x}) = \text{softmax}(\mathbf{x} + c),$$

donde la operación $\mathbf{x} + c$ se realiza con broadcasting. Recuerda que

$$\text{softmax}(\mathbf{x})_i = \frac{e^{\mathbf{x}_i}}{\sum_j e^{\mathbf{x}_j}}$$

Lo anterior es útil cuando se escoge $c = -\max(\mathbf{x})$, es decir, quitando el valor mayor en todos los elementos de $\mathbf{x}$, para estabilidad numérica.

b) Para un escalar x, muestra que el gradiente de la función sigmoide es $\sigma(x)(1 - \sigma(x))$.

c) Muestra que el gradiente en la capa de salida es

$$\frac{\partial L(\mathbf{y}, \hat{\mathbf{y}})}{\partial \mathbf{z}} = \hat{\mathbf{y}} - \mathbf{y},$$

donde $\hat{\mathbf{y}} = \text{softmax}(\mathbf{z})$, para algún vector $\mathbf{z}$ que proviene de la capa de salida. ¿Qué interpretación puedes dar a ésa expresión? La función de costo, como mencionamos al inicio, es la cross-entropy:

$$L(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_i y_i \log(\hat{y}_i),$$

donde $\mathbf{y}$ es un vector one-hot de las clases y $\hat{\mathbf{y}}$ es el vector de probabilidades estimadas.

d) Considerando los incisos anteriores, obtén los gradientes respecto a los parámetros del modelo calculando

$$\frac{\partial L(\mathbf{y}, \hat{\mathbf{y}})}{\partial \mathbf{x}}$$

para obtener de ésta forma, las ecuaciones de backpropagation de la red. Recuerda que el paso forward calcula las activaciones: $\mathbf{h} = \sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)$ y $\hat{\mathbf{y}} = \text{softmax}(\mathbf{W}_2\mathbf{h} + \mathbf{b}_2)$. Recuerda también que la función de activación en un vector, se aplica entrada por entrada.

SOLUCIÓN

a)

Ya que la suma $\mathbf{x} + c$ se hace por broadcasting, donde $c \in \mathbb{R}$, quiere decir que:

$$(\mathbf{x} + c)_i = x_i + c \tag{29}$$

Tal que, partiendo de la definición de la función softmax para un elemento i del vector de salida aplicada al vector $\mathbf{x} + c$:

$$\begin{aligned}
\text{softmax}(\mathbf{x} + c)_i &= \frac{e^{(\mathbf{x}+c)_i}}{\sum_j e^{(\mathbf{x}+c)_j}} \\
&= \frac{e^{x_i + c}}{\sum_j e^{x_j + c}} \\
&= \frac{e^{x_i} e^c}{\sum_j e^{x_j} e^c} \\
&= \frac{e^{x_i} e^c}{e^c \sum_j e^{x_j}} \\
&= \frac{e^{x_i}}{\sum_j e^{x_j}} \\
&= \text{softmax}(\mathbf{x})_i
\end{aligned}$$

Ya que esta igualdad es cierta $\forall i$, por lo tanto queda demostrado que la función softmax es invariante ante traslaciones constantes, es decir:

$$\boxed{\text{softmax}(\mathbf{x}) = \text{softmax}(\mathbf{x} + c)}$$

b)

Sea la función sigmoide:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{30}$$

Para calcular el gradiente derivamos utilizando la regla de la cadena, tal que:

$$\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left(\frac{1}{1 + e^{-x}}\right) \\
&= \frac{d}{dx}\left(1 + e^{-x}\right)^{-1} \\
&= \left(1 + e^{-x}\right)^{-2} \frac{d}{dx}\left(1 + e^{-x}\right) \\
&= \left(1 + e^{-x}\right)^{-2} \left(-e^{-x}\right) \\
&= \frac{-e^{-x}}{\left(1 + e^{-x}\right)^2}
\end{aligned} \tag{31}$$

Siendo:

$$1 - \sigma(x) = 1 - \frac{1}{1+e^{-x}} = \frac{e^{-x}}{1+e^{-x}} \tag{32}$$

Entonces, tenemos que:

$$\begin{aligned}\sigma(x)\left(1-\sigma(x)\right) &= \frac{1}{1+e^{-x}}\left(\frac{e^{-x}}{1+e^{-x}}\right) \\ &= \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \frac{d}{dx}\sigma(x)\end{aligned}$$

Por lo tanto, queda demostrado que:

$$\boxed{\frac{d}{dx}\sigma(x) = \sigma(x)\left(1-\sigma(x)\right)}$$

c)

Primero calculamos el gradiente de L para una sola componente z_k mediante la regla de la cadena. Podemos ver que el costo L no depende directamente de z_k, sino a través de todas las salidas $\hat{y}_i$, tal que buscamos:

$$\frac{\partial L}{\partial z_k} = \sum_i \frac{\partial L}{\partial \hat{y}_i}\frac{\partial \hat{y}_i}{\partial z_k} \tag{33}$$

Calculando por partes, primero:

$$\frac{\partial L}{\partial \hat{y}_i} = \frac{\partial}{\partial \hat{y}_i}\left(-\sum_j y_j \log(\hat{y}_i)\right) = -\frac{y_i}{\hat{y}_i} \tag{34}$$

Todos los términos donde $j \neq i$ son constantes para $\hat{y}_i$. Ahora calculamos la derivada de la función softmax, para ello necesitamos considerar dos casos:

1. Caso $i = k$. Derivamos una salida $\hat{y}_k$ con respecto a una entrada z_k. Usamos la regla del cociente para calcular la derivada sobre $\hat{y}_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$:

$$\begin{aligned}\frac{\partial \hat{y}_k}{\partial z_k} &= \frac{(e^{z_k})'(\sum_j e^{z_j}) - (e^{z_k})(\sum_j e^{z_j})'}{(\sum_j e^{z_j})^2} \\ &= \frac{e^{z_k}(\sum_j e^{z_j}) - e^{z_k}(e^{z_k})}{(\sum_j e^{z_j})^2} \\ &= \frac{e^{z_k}}{\sum_j e^{z_j}}\left[\frac{(\sum_j e^{z_j}) - e^{z_k}}{\sum_j e^{z_j}}\right] \\ &= \hat{y}_k(1-\hat{y}_k)\end{aligned} \tag{35}$$

2. Caso $i \neq k$. Derivamos una salida $\hat{y}_i$ con respecto a una entrada diferente z_k. Nuevamente usamos la regla del cociente para calcular la derivada sobre $\hat{y}_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$:

$$\begin{aligned}\frac{\partial \hat{y}_i}{\partial z_k} &= \frac{(0)(\sum_j e^{z_j}) - (e^{z_i})(e^{z_k})}{(\sum_j e^{z_j})^2} \\ &= -\frac{e^{z_i}}{\sum_j e^{z_j}} \left[\frac{e^{z_k}}{\sum_j e^{z_j}}\right] \\ &= -\hat{y}_i \hat{y}_k \end{aligned} \tag{36}$$

Ahora unificando todo, separamos la suma en el termino donde $i = j$ y los términos donde $i \neq j$, respectivamente:

$$\frac{\partial L}{\partial z_k} = \left(\frac{\partial L}{\partial \hat{y}_k}\frac{\partial \hat{y}_k}{\partial z_k}\right) + \sum_{i \neq k}\left(\frac{\partial L}{\partial \hat{y}_i}\frac{\partial \hat{y}_i}{\partial z_k}\right) \tag{37}$$

Sustituyendo lo encontrado en (34), (35) y (36) tenemos que:

$$\begin{aligned}\frac{\partial L}{\partial z_k} &= \left(-\frac{y_k}{\hat{y}_k}\right)[\hat{y}_k(1-\hat{y}_k)] + \sum_{i \neq k}\left(-\frac{y_i}{\hat{y}_i}\right)(-\hat{y}_i\hat{y}_k) \\ &= -y_k(1-\hat{y}_k) + \sum_{i \neq k} y_i \hat{y}_k \\ &= -y_k + \hat{y}_k\left(y_k + \sum_{i \neq k} y_i\right)\end{aligned}$$

Dado que $\mathbf{y}$ es un vector **one-hot**, la suma de todos sus componentes es 1, es decir:

$$y_k + \sum_{i \neq k} y_i = 1$$

Por lo tanto:

$$\frac{\partial L}{\partial z_k} = \hat{y}_k - y_k \tag{38}$$

Siendo esto valido para cualquier componente k, podemos escribir el gradiente en su forma vectorial:

$$\boxed{\frac{\partial L(\mathbf{y}, \hat{\mathbf{y}})}{\partial \mathbf{z}} = \hat{\mathbf{y}} - \mathbf{y}}$$

Este vector resultante, $\hat{\mathbf{y}} - \mathbf{y}$, podemos interpretarlo como el **vector de error** de la predicción. Cada componente de este vector nos dice exactamente cómo debemos ajustar la entrada a la capa softmax, z_k, para que la predicción mejore y el costo disminuya.

- **Para la clase correcta**. Supongamos que el índice k corresponde a la clase verdadera. En este caso, el valor en el vector *one-hot* es $y_k = 1$. La componente k del gradiente es:

$$\frac{\partial L}{\partial z_k} = \hat{y}_k - y_k = \hat{y}_k - 1$$

Dado que $\hat{y}_k$ es una probabilidad y siempre está en el rango $[0, 1]$, el valor de $\hat{y}_k - 1$ será siempre negativo (o cero si la predicción es perfecta). En el algoritmo de descenso de gradiente, los parámetros se actualizan en la dirección **opuesta** al gradiente. Restar un valor negativo es sumar, por lo que el efecto neto seria **incrementar** el valor de z_k. Esto es precisamente lo que se busca: aumentar la confianza (el *logit*) en la clase correcta para que su probabilidad se acerque a 1.

- **Para cualquier clase incorrecta**. Ahora, considerando cualquier otro índice j que no corresponde a la clase verdadera. Para estos casos, el valor en el vector *one-hot* es $y_j = 0$. La componente j del gradiente es:

$$\frac{\partial L}{\partial z_j} = \hat{y}_j - y_j = \hat{y}_j - 0 = \hat{y}_j$$

El valor de $\hat{y}_j$ es una probabilidad, por lo que este gradiente será siempre positivo (o cero). Al actualizar los parámetros, el algoritmo restará este valor positivo, lo que provocará una **disminución** en el valor de z_j. Este es, de nuevo, el comportamiento deseado: queremos reducir la confianza en todas las clases que son incorrectas, empujando sus probabilidades hacia 0.

En conclusión, la expresión $\hat{\mathbf{y}} - \mathbf{y}$ esconde un mecanismo de corrección muy potente. El gradiente dirige de forma natural el proceso de optimización para que, iteración tras iteración, los *logits* de las clases incorrectas se reduzcan y el *logit* de la clase correcta aumente, mejorando así la precisión del modelo.

d)

Sabemos que el algoritmo de **backpropagation** (retropropagación del error) no es más que la aplicación sistemática de la regla de la cadena para calcular los gradientes de la función de costo L con respecto a todos los parámetros del modelo, (en este caso: $\mathbf{W}_2, \mathbf{b}_2, \mathbf{W}_1, \mathbf{b}_1$). El procedimiento sera ir hacia atrás en la red, desde el costo final hasta la entrada inicial.

Para el paso forward, tenemos que la salida de la red se calcula de la siguiente manera:

1. **Capa oculta**:
   - Entrada a la capa oculta:
   $$\mathbf{z}_1 = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1 \tag{39}$$
   - Activación de la capa oculta:
   $$\mathbf{h} = \sigma(\mathbf{z}_1) \tag{40}$$
2. **Capa de salida**:
   - Entrada a la capa de salida:
   $$\mathbf{z}_2 = \mathbf{W}_2\mathbf{h} + \mathbf{b}_2 \tag{41}$$
   - Activación de la capa de salida (predicción):
   $$\hat{\mathbf{y}} = \text{softmax}(\mathbf{z}_2) \tag{42}$$

Ahora para el paso backward:

1. **Gradientes de la capa de salida**.

   - Error en la salida, $\boldsymbol{\delta}_2$:
     Sea $\boldsymbol{\delta}_2$ el error de la capa de salida, del inciso **c)** ya conocemos el gradiente del costo L respecto a la entrada de la función softmax, $\mathbf{z}_2$, tal que:
     $$\boldsymbol{\delta}_2 \equiv \frac{\partial L}{\partial \mathbf{z}_2} = \hat{\mathbf{y}} - \mathbf{y} \tag{43}$$
   - Gradiente para los pesos $\mathbf{W}_2$:
     Mediante la regla de la cadena:
     $$\frac{\partial L}{\partial \mathbf{W}_2} = \frac{\partial L}{\partial \mathbf{z}_2}\frac{\partial \mathbf{z}_2}{\partial \mathbf{W}_2} \tag{44}$$
     Donde:
     $$\frac{\partial \mathbf{z}_2}{\partial \mathbf{W}_2} = \mathbf{h}^T \tag{45}$$
     Por lo tanto:
     $$\frac{\partial L}{\partial \mathbf{W}_2} = \boldsymbol{\delta}_2\mathbf{h}^T \tag{46}$$
   - Gradiente para el sesgo $\mathbf{b}_2$:
     Mediante la regla de la cadena:
     $$\frac{\partial L}{\partial \mathbf{b}_2} = \frac{\partial L}{\partial \mathbf{z}_2}\frac{\partial \mathbf{z}_2}{\partial \mathbf{b}_2} \tag{47}$$
     Donde:
     $$\frac{\partial \mathbf{z}_2}{\partial \mathbf{b}_2} = \mathbf{1}^T \tag{48}$$
     Por lo tanto:
     $$\frac{\partial L}{\partial \mathbf{b}_2} = \boldsymbol{\delta}_2 \tag{49}$$

2. **Retropropagación del error a la capa oculta**.

   - Error en la capa oculta, $\boldsymbol{\delta}_1$:
     Para encontrar los gradientes de $\mathbf{W}_1$ y $\mathbf{b}_1$, primero necesitamos propagar el error $\boldsymbol{\delta}_2$ hacia atrás a través de $\mathbf{W}_2$ y la función de activación sigmoide.
     Primero, calculamos el gradiente con respecto a la activación de la capa oculta, $\mathbf{h}$:
     $$\frac{\partial L}{\partial \mathbf{h}} = \frac{\partial L}{\partial \mathbf{z}_2}\frac{\partial \mathbf{z}_2}{\partial \mathbf{h}} \tag{50}$$
     Donde:
     $$\frac{\partial \mathbf{z}_2}{\partial \mathbf{h}} = \mathbf{W}_2^T \tag{51}$$
     Por lo tanto:
     $$\frac{\partial L}{\partial \mathbf{h}} = \boldsymbol{\delta}_2\mathbf{W}_2^T \tag{52}$$
     Ahora, calculamos el gradiente con respecto a la entrada de la capa oculta, $\mathbf{z}_1$. A esto lo llamaremos el error $\boldsymbol{\delta}_1$:
     $$\boldsymbol{\delta}_1 \equiv \frac{\partial L}{\partial \mathbf{z}_1} = \frac{\partial L}{\partial \mathbf{h}} \odot \frac{\partial \mathbf{h}}{\partial \mathbf{z}_1} \tag{53}$$
     Donde, $\odot$ denota el producto de Hadamard (elemento a elemento), ya que la función sigmoide se aplica a cada neurona individualmente. Del inciso **b)**, sabemos que la derivada de la sigmoide es $\sigma'(z) = \sigma(z)(1-\sigma(z))$, en este caso, $\mathbf{h}' = \mathbf{h} \odot (1-\mathbf{h})$. Por lo tanto:
     $$\boldsymbol{\delta}_1 = (\mathbf{W}_2^T\boldsymbol{\delta}_2) \odot (\mathbf{h} \odot (1-\mathbf{h})) \tag{54}$$

3. **Gradientes de la capa de entrada**.

   - Gradiente para los pesos $\mathbf{W}_1$:
     Ahora que tenemos el error $\boldsymbol{\delta}_1$, podemos calcular los gradientes para la primera capa de la misma forma que lo hicimos para la segunda.

$$\frac{\partial L}{\partial \mathbf{W}_1} = \frac{\partial L}{\partial \mathbf{z}_1} \frac{\partial \mathbf{z}_1}{\partial \mathbf{W}_1} = \boldsymbol{\delta}_1 \mathbf{x}^T \tag{55}$$

   - Gradiente para el sesgo $\mathbf{b}_1$.

$$\frac{\partial L}{\partial \mathbf{b}_1} = \frac{\partial L}{\partial \mathbf{z}_1} \frac{\partial \mathbf{z}_1}{\partial \mathbf{b}_1} = \boldsymbol{\delta}_1 \tag{56}$$

4. **Gradiente con Respecto a la Entrada x**.

   Finalmente, para obtener el gradiente con respecto a la entrada de la red, propagamos el error $\boldsymbol{\delta}_1$ un último paso hacia atrás a través de $\mathbf{W}_1$.

   - Gradiente para la entrada x:

$$\frac{\partial L}{\partial \mathbf{x}} = \frac{\partial L}{\partial \mathbf{z}_1} \frac{\partial \mathbf{z}_1}{\partial \mathbf{x}} \tag{57}$$

     Donde:

$$\frac{\partial \mathbf{z}_1}{\partial \mathbf{x}} = \mathbf{W}_1^T \tag{58}$$

     Por lo tanto:

$$\frac{\partial L}{\partial \mathbf{x}} = \boldsymbol{\delta}_1 \mathbf{W}_1^T \tag{59}$$

     Sustituyendo la expresión completa de $\boldsymbol{\delta}_1$ que encontramos antes, obtenemos:

$$\boxed{\frac{\partial L}{\partial \mathbf{x}} = \left[((\mathbf{W}_2)^T (\hat{\mathbf{y}} - \mathbf{y})) \odot (\mathbf{h} \odot (1 - \mathbf{h})) \right] \mathbf{W}_1^T}$$

Problema 5

En el moodle del curso, encontrarás un conjunto de datos que corresponden a un extracto del Free Music Archive (FMA) [1], que es una base de datos muy extensa de archivos de audio usada para diversas tareas de MIR. Encontrarás varios conjuntos de datos en formato csv que corresponden a dos grupos: uno de entrenamiento (archivos que empiezan con el prefijo `tracks_fma_train`) y uno de prueba (archivos que empiezan con el prefijo `tracks_fma_test`). Para cada uno hay un archivo con metadatos (`..._metadata.csv`) que incluye la variable de respuesta `track.genre` excepto para los datos de test. Este subconjunto de datos lo construí basado en los archivos originales y haciendo mucho preproceso de los mismos. En general, traté de incluir la información relevante para éste ejercicio, quitando valores nulos que no podían estimarse, entre otras cosas. Los datos originales son considerablemente mas grandes.

También, se proporcionan diversos archivos que contienen características de audio y de la señal (`..._features.csv`) en forma de indicadores o estadísticas de la señal, por lo que las escalas pueden variar.

a) Usando los datos de train, realiza un análisis exploratorio de los datos, usando métodos de reducción de dimensión que consideres apropiados. ¿Qué características tiene la variable de respuesta? ¿Puedes identificar los géneros, o al menos, algún subconjunto de ellos? Utiliza las características de audio y de la señal que creas convenientes y repórtalo junto con todos tus hallazgos.

b) Usando también los datos de entrenamiento, ajusta una red neuronal con pytorch para predecir el género. Genera un conjunto de datos de entrenamiento y validación para verificar su ajuste. Define la arquitectura de la red, y reporta todos los detalles del modelo y el preprocesamiento de los datos que hayas usado (características de audio y señal usadas, reducción de dimensión, regularización, optimizador, estandarización, escalamiento, etcétera). Reporta las métricas de desempeño y gráficas que creas conveniente.

c) Usando la red neuronal ajustada en el paso previo, predice el género de los datos de test. Debes entregar un archivo csv con el `track_id` de cada registro y el género que predijo tu modelo. El que obtenga el mejor resultado, recibirá puntos extra.

SOLUCIÓN

La discusión, solución y resultados obtenidos de este ejercicio se encuentra en el notebook adjunto llamado `Tarea01_Diego_Paniagua.ipynb` y en el arhivo `Tarea01_Diego_Paniagua.csv`.

Problema 6

Leé y haz un resúmen corto del ensayo “Una nota personal sobre música, sonido y electrónica”, de Daphne Oram. Incluye tu opinión personal sobre el mismo.

SOLUCIÓN

Resumen. Una nota personal sobre música, sonido y electrónica

Este capitulo del libro **La Materia del Sonido** presenta una analogía entre los principios de la electrónica y el sonido con el funcionamiento de la memoria y el pensamiento humano proponiendo que nuestra mente opera de manera similar a un complejo sistema de circuitos de retroalimentación.

La Memoria como un Circuito de Retroalimentación

El autor concibe la memoria no como una cinta de grabación, sino como millones de **circuitos en sintonía** que mantienen un “eco” continuo de las señales (recuerdos).

- **Concentración y Recuerdo:** Concentrarse en un recuerdo es como “subir el volumen” de su circuito, reforzando la señal.
- **Filtros de Personalidad:** Cada vez que un recuerdo es traído a la conciencia, pasa a través de “filtros formantes” personales, lo que significa que el recuerdo se colorea y modifica ligeramente con nuestra personalidad y estado actual.
- **Imperfección del Recuerdo:** El cerebro no almacena cada detalle. Recuerda los puntos clave y **rellena los huecos con razonamiento lógico**, lo que explica por qué los recuerdos pueden cambiar con el tiempo. Los recuerdos más simples y claros son los que perduran con mayor fidelidad.

El Peligro: La Retroalimentación sin Control

Así como en la electrónica, un bucle de **retroalimentación sin control** es peligroso. En la mente humana, esto se manifiesta como pensamientos obsesivos o corrosivos.

- **La Causa:** Ocurre cuando se le da “demasiado volumen” a un pensamiento, sobrecargando el sistema.
- **El Efecto:** El pensamiento se amplifica y distorsiona en cada ciclo, introduciendo “ruido” y matices que no estaban. Una pequeña preocupación (“un grano”) se convierte en una montaña, y una idea simple puede terminar sintiéndose como “ruido blanco” que consume toda la atención.

¿Cómo Romper el Bucle Obsesivo?

El autor explora varias formas de detener esta retroalimentación destructiva, comparándolas con soluciones del mundo del audio y la electrónica.

- **Cambiar el Foco:** La forma más simple es desviar la concentración hacia otro tema, esperando no sobrecargar el nuevo “circuito”.
- **El Shock o la Sobrecarga:** Un evento repentino y violento (un grito, un susto, un golpe de agua fría) puede “sobrecargar” el sistema momentáneamente, forzándolo a grabar “silencio” y rompiendo el bucle del pensamiento.

- **La Transducción (Hablar o Escribir):** Esta es la solución más sofisticada. Al verbalizar o escribir un pensamiento, lo forzamos a pasar por **filtros de censura personal** (lógica, contexto social, vergüenza, etc.). Luego, al escucharnos o leernos, el pensamiento regresa a nosotros como una señal externa, permitiéndonos **analizarlo desde un ángulo diferente** y más racional. Este proceso de externalizar y reinterpretar es análogo a un tratamiento psiquiátrico.

En esencia, el capítulo utiliza la música y la electrónica para argumentar que nuestros pensamientos y recuerdos no son estáticos, sino señales dinámicas que podemos aprender a modular para evitar caer en bucles mentales destructivos.

Opinión Personal

Me gusto mucho leer este capitulo del libro, me quede con ganas de seguir leyendo pues el tema que aborda resuena mucho con mi estado interior actualmente. Mi formación como físico me brindo un poco de conocimiento en electrónica y ondas (sonido), esto me ayudo a comprender (un poco) las analogías relacionadas con el funcionamiento del pensamiento humano, pero lo que mas me gusto fue que la lectura me hizo recordar como esta retroalimentación/repetición de pensamientos absurdos, obsesivos, traumas, son lo que realmente les sigue dando fuerza. A pesar de que en algún punto de mi vida me di cuenta de esto, pareciera que lo hubiera olvidado, ya sea porque cuando lo aprendí no quedo tan claro o porque la retroalimentación sin control últimamente a sido bastante alta. Reflexionando un poco, veo que de manera inconsciente he aplicado las técnicas mencionadas para romper estos bucles, cambiar el foco de la idea (hacer ejercicio, meditar, ejercicios de respiración), la terapia de shock (gritar, baños de agua fría) y la transducción (hablar o escribir con respecto a los problemas), ahora buscare aplicarlas de manera cociente. Así mismo el autor se pregunta si en el mundo cotidiano estas técnicas o mediante un cambio de entorno intencionado, mezclado con algo de incertidumbre puede ayudar con estos golpes de la retroalimentación, (me identifico de directamente con mi experiencia de venir a vivir a Apodaca y estudiar en CIMAT), y su respuesta rápida es que no, que se necesita mas que este mero cambio de entorno o hábitos. Y por experiencia propia puedo decir que tiene razón. También aborda la idea de observar las cosas desde distintos ángulos, y menciona “añadir variantes mas amplias que harán que la realidad sea aun mas aceptable”, contrastando con la analogía de que si la frase musical se vuelve mas armónica, o desarrollada de otras maneras, sera escuchada desde múltiples ángulos y me hace pensar si esto tiene alguna relación con mis ganas de hacer música sobre lo que siento y he pasado. Podría ser otro mecanismo inconsciente de defensa para el tratamiento de la retroalimentación. Tal vez, ya debería ir al psicólogo. Agradezco por la recomendación de la lectura.

References

[1] NumPy Developers. Broadcasting — numpy v2.3 manual, 2025.