Tarea 1

Fundamentos

Profesor: Fernando Crema Garcia
Fecha: 10 de Junio de 2026

Evaluación. La nota final se basa en el promedio ponderado de las tareas (70 %) y el proyecto final (30 %). La nota de cada informe se multiplica por un factor de calidad $\alpha \in [0{,}8,\ 1{,}2]$ según la calidad del reporte asociado.

Tarea	Tema	Peso
Tarea 1	Fundamentos	30 %
Tarea 2	Componentes	30 %
Tarea 3	Arquitecturas	30 %
Tarea 4	Optimizando	10 %

Entrega soft	24/06/2025, 23:59 (hora Caracas) - Máximo 20 ptos
Entrega final	02/08/2026, 23:59 (hora Caracas) - Máximo 10 ptos
Penalización	2 ptos/día de retraso hasta un máximo de 5 días. Después del 02/08/2025 la nota es 0.

Nota final:

$$\text{mín}\Big\{\alpha\,(0{,}3\,T_1 + 0{,}3\,T_2 + 0{,}3\,T_3 + 0{,}1\,T_4)\,,\ 20\Big\}$$

I. Introducción

I-A. Objetivos

El objetivo de esta actividad es consolidar los conocimientos fundamentales sobre redes neuronales profundas a través de la resolución de una serie de notebooks del libro Understanding Deep Learning de Simon Prince. Los notebooks seleccionados cubren aspectos clave del aprendizaje profundo, desde funciones de activación y diseño de redes profundas hasta funciones de pérdida, algoritmos de optimización y retropropagación (backpropagation).

Objetivos específicos

1. Resolver los siguientes notebooks del libro UDL de Simon Prince, utilizando Jupyter/Colab:
   - **Notebook 3.4** - Activation functions
   - **Notebook 4.1** - Composing networks
   - **Notebook 4.2** - Clipping functions
   - **Notebook 4.3** - Deep networks
   - **Notebook 5.1** - Least squares loss
   - **Notebook 5.2** - Binary cross-entropy loss
   - **Notebook 5.3** - Multiclass cross-entropy loss
   - **Notebook 6.2** - Gradient descent
   - **Notebook 6.3** - Stochastic gradient descent
   - **Notebook 6.4** - Momentum
   - **Notebook 6.5** - Adam
   - **Notebook 7.1** - Backpropagation in toy model
   - **Notebook 7.2** - Backpropagation
   - **Notebook 7.3** - Initialization
2. Integrar la pregunta en la sección Evaluación a los notebooks 7.2 y 7.3. Verifique que los resultados obtenidos son los mismos a los del notebook.

Entregables del informe

Aunado a la resolución de los notebooks[1] se desea la elaboración de un informe en PDF de no más de 10 páginas donde resuelvan las siguientes preguntas prácticas usando solamente el código generado dentro de sus notebooks o vistos en clases.

Importante:

1. Documentar el proceso de resolución de los notebooks en un Jupyter Notebook explicativo, incluyendo reflexiones personales sobre los conceptos fundamentales cubiertos y cómo se relacionan entre sí a lo largo del flujo de aprendizaje profundo.
2. Diseñar un experimento final donde se integre lo aprendido: entrenar un modelo simple utilizando los principios vistos (activaciones, arquitectura, función de pérdida, optimización, etc.) y almacenarlo como modelo óptimo.

II. Evaluación

II-A. Backpropagation

Considere la red neuronal de la Figura 1.

Cada neurona oculta utiliza una función de activación sigmoide:

$$\sigma(t) = \frac{1}{1+\exp(-t)}.$$

La función de pérdida es el *error cuadrático medio* (MSE), y los puntos de datos son:

$$(x_1, x_2, y)_i, \quad i = 1,2,3 = \{(1,1,1),(0,2,1),(-1,1,0)\}.$$

Las unidades de entrada son negras, los *biases* son verdes, las unidades ocultas son naranjas, y la unidad de salida es roja. Los pesos están indicados en las flechas correspondientes.

email: fernando.cremagarcia@kuleuven.be

[1] Que serán agregados en el entregable

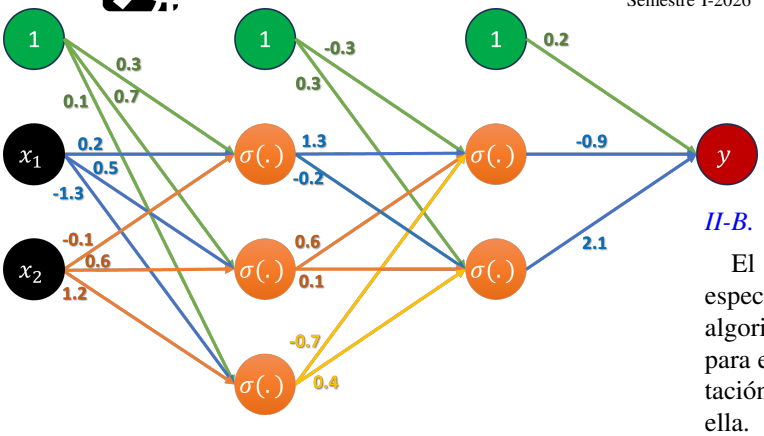

Figura 1. Red Neuronal $f(x) = y$ con $f : \mathbb{R}^2 \to \mathbb{R}$ los componentes de los sesgos en color verde, la de las neuronas ocultas en color naranja. Hay dos capas ocultas 0 y 1. Hay una sola salida en color rojo.

- Identifica todas las componentes explicadas en la sección 7.4 ecuación 7.24 en el caso del paso hacia adelante.

$$\begin{aligned}\mathbf{f}_0 &= \boldsymbol{\beta}_0 + \boldsymbol{\Omega}_0 \mathbf{x}_i \\ \mathbf{h}_k &= \mathbf{a}\left[\mathbf{f}_{k-1}\right] \qquad k \in \{1, 2, \ldots, K\} \\ \mathbf{f}_k &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k. \quad k \in \{1, 2, \ldots, K\}\end{aligned}$$

- Modifique la ecuación 7.25 para la nueva función de activación

$$\begin{aligned}\frac{\partial \ell_i}{\partial \boldsymbol{\beta}_k} &= \frac{\partial \ell_i}{\partial \mathbf{f}_k} && k \in \{K, K-1, \ldots, \\ \frac{\partial \ell_i}{\partial \boldsymbol{\Omega}_k} &= \frac{\partial \ell_i}{\partial \mathbf{f}_k} \mathbf{h}_k^T && k \in \{K, K-1, \ldots, \\ \frac{\partial \ell_i}{\partial \mathbf{f}_{k-1}} &= \mathbb{I}\left[\mathbf{f}_{k-1} > 0\right] \odot \left(\boldsymbol{\Omega}_k^T \frac{\partial \ell_i}{\partial \mathbf{f}_k}\right), && k \in \{K, K-1, \ldots,\end{aligned}$$

‘

- Identifica todas las componentes explicadas en la sección 7.4 de la ecuación modificada 7.25 en el caso del paso hacia atrás
- Luego, realiza un paso de descenso de gradiente (batch, simple) con **tasa de aprendizaje igual a 1**.
- ¿Cuáles son los nuevos pesos?

Nota: No es necesario usar una computadora. Todo puede resolverse con lápiz y papel.

Sugerencia:

$$\frac{d}{dt}\sigma(t) = \sigma(t)(1 - \sigma(t)).$$

Verificar que la nueva pérdida efectivamente ha disminuido también es una buena pista.

II-B. Algoritmos de optimización

El objetivo en este ejercicio es comprender mejor las especificidades, ventajas e inconvenientes de (algunos de) los algoritmos de optimización basados en gradientes disponibles para entrenar redes neuronales. Se proporciona una implementación en pytorch como material suplementario para jugar con ella.

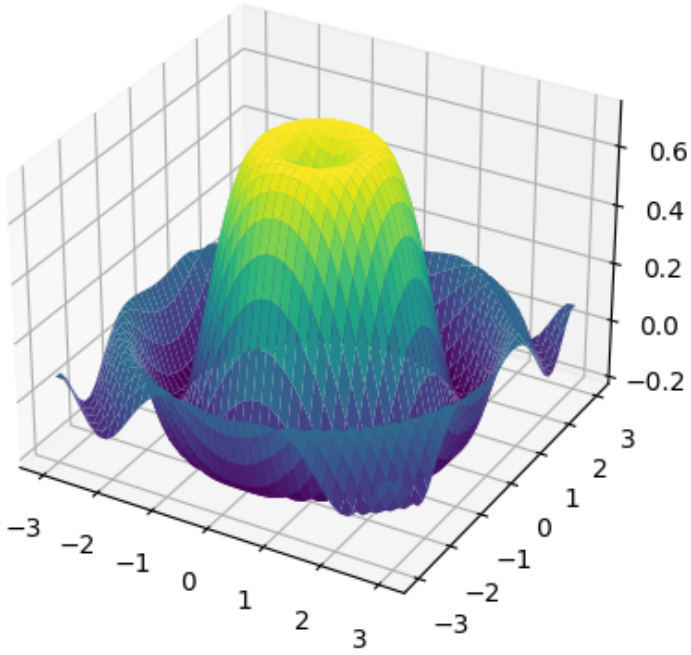

Figura 2. Superficie generada por la función f(x, y)

Considere la siguiente superficie de la figura 2 descrita de la siguiente manera

$$f(x, y) = \frac{\sin\left(0{,}8\left(x^2 + y^2\right)\right)}{\left(x^2 + y^2\right)^{0{,}9}}$$

El conjunto de datos consiste en puntos muestreados en esta superficie con o sin ruido, a partir de una cuadrícula regular en el plano bidimensional.

Para responder a las siguientes preguntas, puedes utilizar el código proporcionado en los notebooks y videos; modificarlo para adaptarlo a tus necesidades. Tendrás que generar gráficos para apoyar tus conclusiones y agregarlas en el informe. Es un buen paso revisar los videos incluídos en nuestro Discord.

- Cuál es el impacto del parámetro ‘ruido‘ en el proceso de optimización?
- Cómo se diferencia el método del descenso del gradiente con el descenso estocástico y las versiones aceleradas?

- Cómo impacta la dimensión de la red neuronal en el rendimiento del optimizador?
- Intente analizar la diferencia entre épocas y el tiempo de convergencia de los algoritmos.

II-C. Otros algoritmos

Invbestigue y elabore una página extra en el informe donde explique usando la notación del libro UDL los siguientes algoritmos de optimización.

- AdamW
- AdaDelta
- RMSDrop
- LBFGS

II-D. Inicialización

Considere nuevamente la red de la Figura 1, pero ahora con pesos inicializados aleatoriamente como $\Omega_{ij} \sim \mathcal{N}(0, \sigma_\Omega^2)$ y sesgos en cero, siguiendo la sección 7.5 del libro UDL.

- Usando el argumento de la ecuación 7.31 (preservación de la varianza de las preactivaciones $\mathbf{f}_k$ en el paso hacia adelante), derive el valor de σ_Ω^2 que propone la **inicialización He** para una capa con D_h unidades de entrada. ¿Qué supuesto sobre la función de activación se utiliza en esta derivación?
- La red de la Figura 1 usa activaciones sigmoides, no ReLU. Explique por qué el supuesto anterior deja de ser válido y derive (o justifique) la varianza que propone la **inicialización Xavier/Glorot**, $\sigma_\Omega^2 = 2/(D_{in} + D_{out})$. ¿Por qué Xavier promedia las dimensiones de entrada y salida mientras que He solo usa la de entrada?
- Para la arquitectura de la Figura 1, calcule numéricamente σ_Ω según He y según Xavier para cada capa ($\boldsymbol{\Omega}_0$, $\boldsymbol{\Omega}_1$ y la capa de salida).
- Modificando el Notebook 7.3, construya una red profunda ($K \geq 20$ capas) y grafique la varianza de las preactivaciones $\mathbf{f}_k$ en función de la profundidad k para tres esquemas:
  1. σ_Ω^2 demasiado pequeña
  2. σ_Ω^2 demasiado grande
  3. He/Xavier según corresponda a la activación. Relacione lo observado con los fenómenos de *vanishing* y *exploding gradients* discutidos en la sección de backpropagation.

Recordemos: Para $\Omega_{ij} \sim \mathcal{N}(0, \sigma_\Omega^2)$ i.i.d. e independiente de $\mathbf{h}$, se tiene $\text{Var}[f_i] = \sigma_\Omega^2 \sum_j \mathbb{E}[h_j^2]$. Para ReLU con entradas simétricas, $\mathbb{E}[h_j^2] = \frac{1}{2}\text{Var}[f_j']$.

III. Uso de Inteligencia Artificial

El uso de herramientas de inteligencia artificial generativa (ChatGPT, Claude, Copilot, Gemini, etc.) está permitido **únicamente** en las secciones indicadas a continuación. En el resto de la tarea, el trabajo debe ser realizado en su totalidad por el estudiante.

Componente	Uso de AI
Notebooks del libro UDL	No permitido [2]
II-A. Backpropagation	No permitido
II-B. Algoritmos de optimización	Permitido
II-C. Otros algoritmos	No permitido
II-D. Inicialización	Permitido sólo en código

Condiciones de uso. En las secciones donde el uso de AI está permitido, el estudiante debe declarar explícitamente en el informe qué herramienta utilizó y con qué propósito (por ejemplo: redacción, derivación de expresiones, verificación de resultados). El estudiante es responsable de la veracidad de todo el contenido entregado: los errores producidos por una AI se penalizan igual que los errores propios.

Incumplimiento. El uso de AI en componentes no autorizados se considera una falta de integridad académica y resulta en la anulación de la sección correspondiente.

[2] Esta es la sección más importante de la tarea. Si logran hacer esto solos el resto es sencillo.