Apuntes Semana 7 - 03/4/2025

Diana Sanabria Calvo 2021436548

***Abstract*— Estos apuntes corresponden a la segunda clase de la Semana 7 del curso de Inteligencia Artificial, impartida el día 03 de abril de 2025. Se abordan de forma detallada los fundamentos teóricos y prácticos de las redes neuronales densas, iniciando desde regresión logística multiclase y avanzando hasta perceptrones multicapa. Se analiza la estructura de una red neuronal, incluyendo el forward pass y el algoritmo de backpropagation, con énfasis en las funciones de activación y la forma en que se realiza la propagación del error. Además, se discuten temas esenciales como la maldición de la dimensionalidad, el aprendizaje jerárquico y el uso de embeddings para representar información semántica. Finalmente, se vinculan los conceptos vistos con casos de aplicación como la clasificación de dígitos en imágenes, proporcionando el fundamento para el estudio de redes convolucionales en clases posteriores.**

I. INTRODUCCIÓN

Durante esta clase se continuó con el estudio de **redes neuronales**, enfocándose en su estructura, comportamiento y capacidad para resolver problemas complejos de clasificación multiclase. Se partió desde el modelo de **regresión logística** extendido al caso multiclase, aplicándolo sobre imágenes del conjunto de datos **MNIST**.

Posteriormente, se profundizó en la transición hacia **perceptrones multicapa**, destacando el papel de las **funciones de activación no lineales** y su impacto en la capacidad de representación del modelo.

Asimismo, se introdujo el algoritmo de *retropropagación* (*backpropagation*) como método para la actualización de pesos mediante derivadas parciales de la función de pérdida. Se discutieron aspectos técnicos y conceptuales que permiten comprender cómo aprende una red neuronal, abordando también desafíos asociados como la **maldición de la dimensionalidad** y el **diseño jerárquico de las redes**.

Finalmente, se presentaron los *embeddings* como una herramienta fundamental para representar relaciones semánticas en un espacio vectorial, abriendo paso a técnicas más avanzadas que se explorarán en futuras sesiones.

I. NOTICIAS DE LA SEMANA

- **Sora de OpenAI:** El modelo Sora genera videos realistas a partir de texto. Aunque su capacidad es sorprendente, ha sido criticado por reforzar estereotipos de género (como mostrar a mujeres como azafatas y hombres como pilotos) y por la falta de diversidad en sus resultados. Esto ocurre porque los datos de entrenamiento reflejan los sesgos existentes en la sociedad, y la IA simplemente los replica.
- **Benchmark Paper-Bench (OpenAI):** Se presentó un nuevo benchmark que evalúa si una IA puede:
  1) Leer un paper académico de IA.
  2) Comprenderlo técnicamente.
  3) Implementar el modelo y obtener los resultados del estudio.

  Paper-Bench representa un paso importante hacia el desarrollo de IAs capaces de replicar investigaciones científicas reales.

II. REPASO DE CONCEPTOS

1) Regresión Lineal: La regresión lineal es un método estadístico para modelar la relación entre una variable dependiente y una o más variables independientes, suponiendo que esa relación es lineal.

2) Regresión Logística: Es una extensión de la regresión lineal para clasificación binaria, representa la probabilidad de pertenecer a la clase positiva. La salida es un número entre 0 y 1. Se aplica una función sigmoide a la salida de una regresión lineal:

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad \text{donde} \quad z = \mathbf{w}\mathbf{x} + b$$

III. MNIST

MNIST (Modified National Institute of Standards and Technology) es un conjunto de datos muy utilizado en machine learning. Contiene 70.000 imágenes de dígitos del 0 al 9. Cada imagen es de 28x28 píxeles (784 píxeles). Las imágenes están en escala de grises.

Objetivo: entrenar modelos que puedan reconocer números escritos a mano.

IV. REGRESIÓN LOGÍSTICA MULTICLASE

Es una extensión de la regresión logística binaria para más de dos clases. En vez de usar una sola salida, se usan varias salidas (una por clase).

1. Cada salida se calcula con una regresión logística independiente.

2. La clase con mayor probabilidad es la predicción final.

V. Conceptos Importantes

Codificación one-hot

Es una forma de representar clases con vectores binarios:
Clase 0 → [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
Clase 2 → [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

Matriz de pesos

Es una estructura que agrupa todos los pesos de una capa:
Si tenemos 10 salidas y 784 entradas:

$$\mathbf{W} \in R^{10\times 784}$$

Cada fila representa los pesos de una clase.

VI. Elementos de una red neuronal

A. Capa de entrada

Recibe los datos crudos (por ejemplo, 784 píxeles).

B. Capas ocultas

Procesan los datos y extraen características.

C. Capa de salida

Entrega la predicción final.

Cada neurona de una capa está totalmente conectada con la anterior → red densa.Cada conexión tiene un peso. Cada neurona tiene su sesgo.

VII. Funciones de activación

A. Sigmoid

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

→ Su salida está entre 0 y 1.

→ Muy usada para clasificación binaria.

→ Problema: en redes profundas puede causar desvanecimiento del gradiente.

B. ReLU (Rectified Linear Unit)

$$\text{ReLU}(x) = \max(0, x)$$

Muy eficiente y fácil de derivar. Problema: neuronas muertas si siempre reciben negativos.

C. Función Softmax

$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$$

Convierte el vector de salidas en una distribución de probabilidad. Se usa en la última capa para clasificación multiclase.

VIII. Forward Pass

El forward pass es el recorrido que hacen los datos desde la entrada hasta la salida.

1. Se pasa la entrada x a través de la red.
2. En cada capa:
Se calcula:

$$z = \mathbf{W}\mathbf{x} + \mathbf{b}$$

3. Se aplica la función de activación:

$$a = f(z)$$

4. Se repite hasta la capa final.
5. La capa de salida entrega el resultado.

IX. Backpropagation

Es el algoritmo que permite ajustar los pesos de la red a partir del error. Se basa en cálculo de derivadas (gradiente descendente).

1. Forward pass: Se calcula la predicción de la red.
2. Cálculo de la pérdida:

$$\mathcal{L} = -\sum_{i=1}^{K} y_i \log(\hat{y}_i)$$

donde:

- K es el número total de clases del problema.
- y_i es la etiqueta verdadera para la clase i (codificada en one-hot).
- $\hat{y}_i$ es la probabilidad predicha por el modelo para la clase i (después de aplicar softmax).
- $\mathcal{L}$ es el valor de la función de pérdida total.

3. Propagación del error hacia atrás: Se calculan los gradientes (derivadas parciales) desde la capa de salida hasta la capa de entrada.

4. Actualización de pesos: Con el gradiente, se actualiza:

$$\mathbf{W} := \mathbf{W} - \eta \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{W}}$$

donde:

- $\mathbf{W}$ es la matriz de pesos que se está actualizando.
- η es la tasa de aprendizaje (learning rate), un valor pequeño como 0.01.
- $\frac{\partial \mathcal{L}}{\partial \mathbf{W}}$ es el gradiente de la función de pérdida respecto a los pesos.
- El símbolo $:=$ indica que $\mathbf{W}$ será actualizado con el nuevo valor.

X. El problema del XOR y el perceptrón

Es un problema que no es separable linealmente → una línea recta no puede separarlo.

x_1	x_2	$x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	0

donde:

- x_1, x_2 son las variables de entrada (pueden ser 0 o 1).
- $\oplus$ representa la operación lógica **XOR** (exclusiva).
- El resultado es 1 si y solo si x_1 y x_2 son distintos.

A. *¿Qué es el perceptrón?*

Es el modelo más simple de red neuronal.

$$y = f(\mathbf{w} \cdot \mathbf{x} + b)$$

donde:

- $\mathbf{x}$ es el vector de entrada.
- $\mathbf{w}$ es el vector de pesos.
- b es el sesgo (bias).
- f es una función de activación, como `step` o `sigmoid`.
- y es la salida del perceptrón.

XI. Inspiración biológica

- **Dendritas:** entradas de la neurona biológica. En la red neuronal artificial, equivalen a las entradas del modelo, representadas como el vector $\mathbf{x}$.
- **Núcleo (soma):** parte encargada de procesar la información recibida. En el modelo artificial, realiza la combinación lineal de las entradas:

  $$z = \mathbf{w} \cdot \mathbf{x} + b$$

  donde $\mathbf{w}$ es el vector de pesos y b es el sesgo o bias.
- **Axón:** transmite la señal procesada hacia otras neuronas. En el modelo artificial, esto se representa como la salida activada:

  $$a = f(z)$$

  donde f es una función de activación como ReLU o σ (sigmoide).

XII. Maldición de la dimensionalidad

A medida que aumentan las dimensiones (features), el espacio se vuelve exponencialmente más grande.

Maldición de la dimensionalidad

- A mayor número de dimensiones, más datos se necesitan para cubrir el espacio de forma representativa.
- Las distancias entre puntos se vuelven menos significativas.
- La búsqueda de patrones se vuelve más costosa y menos efectiva.

Ejemplo del profe: Buscar una pelota en una casa (1D) es fácil. Buscarla en un edificio (2D) es más complejo. Buscarla en una ciudad entera con edificios y pisos (alta dimensión) es exponencialmente más difícil.
Solución: aplicar técnicas como selección de características o reducción de dimensionalidad (e.g., PCA).

XIII. Jerarquía en redes neuronales

Las redes neuronales aprenden en niveles progresivos:

- **Capas iniciales:** aprenden patrones básicos como bordes, líneas y formas simples.
- **Capas intermedias:** detectan combinaciones más complejas como ojos, letras o esquinas.
- **Capas finales:** reconocen conceptos completos como dígitos, gatos, rostros humanos, etc.

Este comportamiento emerge automáticamente durante el entrenamiento, sin intervención directa del programador.

XIV. Embeddings

Es una representación vectorial densa de un objeto (palabra, imagen, etc.) en un espacio numérico.

Palabras similares $\rightarrow$ vectores cercanos. Se pueden sumar, restar y comparar.

XV. vector latente

Es un vector oculto que representa la compresión de una entrada. Se genera dentro de la red durante el entrenamiento. Sirve para representar conceptos complejos en pocas dimensiones.

XVI. Conclusiones

Durante esta clase se consolidaron los fundamentos teóricos y prácticos de las redes neuronales densas, avanzando desde modelos lineales hasta estructuras más complejas como los perceptrones multicapa. Se comprendió la importancia de introducir funciones de activación no lineales para aumentar la capacidad representacional de los modelos, así como el rol central del algoritmo de retropropagación en el proceso de aprendizaje.

Además, se analizaron temas esenciales como la maldición de la dimensionalidad, el aprendizaje jerárquico y el uso de *embeddings* para representar información semántica en espacios vectoriales. Estos conceptos sientan las bases para comprender el funcionamiento de modelos más avanzados, como las redes convolucionales y los modelos generativos.