

Reporte de Proyecto

Nombre del estudiante:

Josué Aldair Romero Pineda

Número de cuenta:

22041249

Sede de estudio:

UNITEC SPS

Docente:

Ing. Ivan Deras

Sección:

119

Fecha de entrega:

27/03/25

Metodología

Arquitectura de la Red

La red neuronal implementada sigue una arquitectura feed-forward (propagación hacia adelante) completamente conectada con las siguientes características:

- **Capa de Entrada:** 784 neuronas (correspondientes a imágenes MNIST aplanadas de 28x28 píxeles).
- **Capas Ocultas:**
 - **Primera Capa Oculta:** 128 neuronas con activación ReLU.
 - **Segunda Capa Oculta (Opcional):** 128 neuronas con activación ReLU (activada mediante un flag).
- **Capa de Salida:** 10 neuronas (una por cada dígito del 0 al 9) con activación Softmax.

Detalles de Implementación

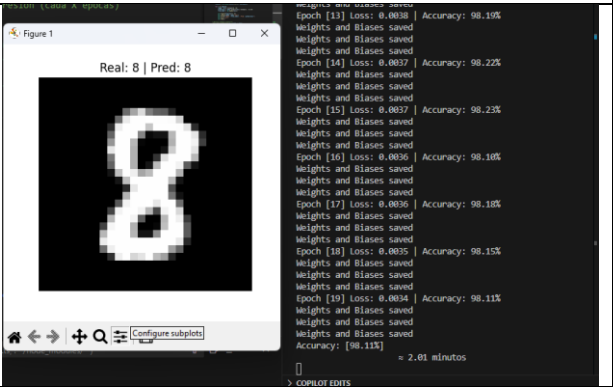
- **Funciones de Activación:**
 - **ReLU:** Utilizada en capas ocultas por su eficiencia computacional y no linealidad.
 - **Softmax:** Convierte logits en probabilidades para clasificación multiclase.
- **Función de Pérdida:** Entropía Cruzada (Cross-Entropy), optimizada para trabajar en conjunto con Softmax.
- **Optimizadores:**
 - **SGD (Descenso de Gradiente Estocástico):** Optimizador base con actualización manual de pesos.
 - **Adam:** Optimizador adaptativo con momento (opcional, configurable por flag).
- **Regularización:** Aplicación opcional de L2 durante el cálculo de gradientes.
- **Entrenamiento:**
 - Mini-lotes de (64 -32) muestras.
 - Validación en el conjunto de prueba tras cada época.
 - Guardado automático de pesos en formato.npy para continuar entrenamientos.

Almacenamiento de Pesos y Sesgos

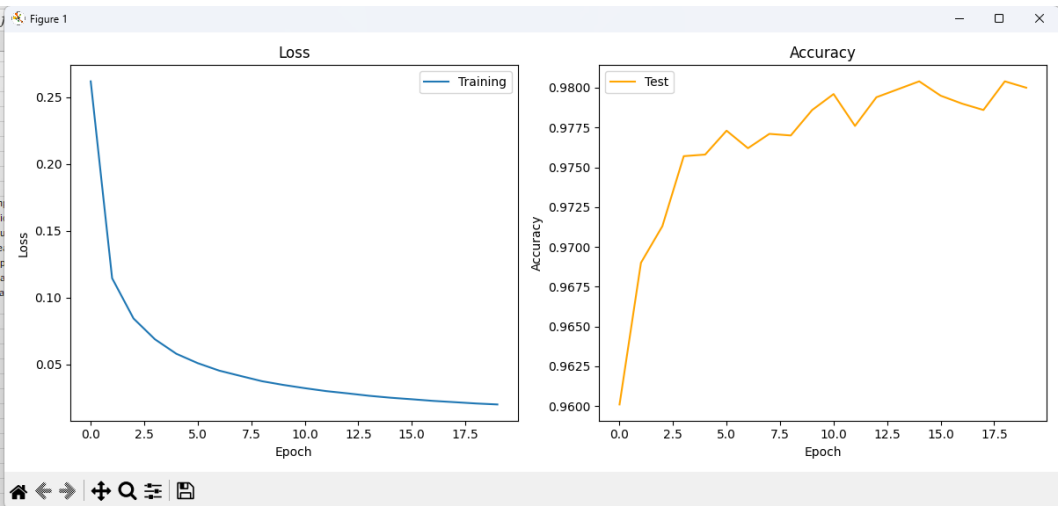
Los pesos y sesgos se guardan en carpetas que hacen función de “modelos” de la red neuronal. Los pesos y sesgos se cargan al inicializar la red neuronal y se guardan al final de cada época.

La documentación del código fue realizada por medio de Deepseek.

Resultados

Ejemplo de Parámetros de Ejecución:	Resultado de Ejecución:
input_size = 784 # 28x28 pixeles hidden_size = 128 output_size = 10 learning_rate = 0.001 epochs = 20 batchSize = 64 saveandprnteach = 1	

Graficas Resultantes de Ejecución:

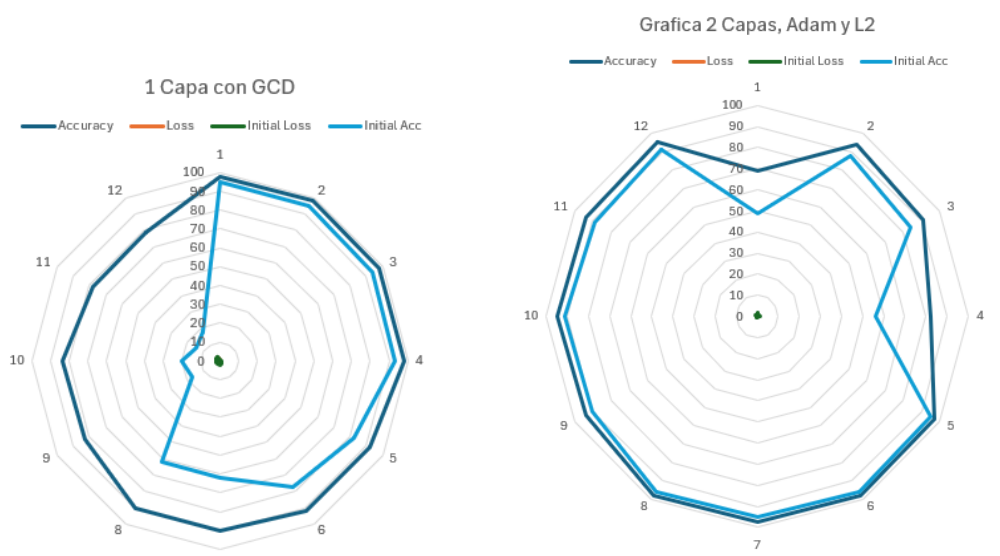


Experimentos con Hiperparametros:

Attempt	Learning Rate	Batch Size	Hidden Layer Size	Epochs	Accuracy	Loss	Initial Loss	Initial Accuracy
1	0.1	32	64	20	97.5	0.015	0.31	94.9
2	0.1	32	128	20	97.22	0.0086	0.29	94.8
3	0.1	64	64	20	97.7	0.032	0.37	93.35
4	0.1	64	128	20	97.96	0.023	0.37	92.9
5	0.001	32	64	20	91.9	0.311	0.68	82.25
6	0.001	32	128	20	91.78	0.309	1.63	77.43
7	0.001	64	64	20	90.52	0.3794	1.994	61.91
8	0.001	64	128	20	90.17	0.3774	2.0779	62.23
9	0.0001	32	64	20	82.71	0.7991	2.3266	16.47
10	0.0001	32	128	20	83.76	0.7607	2.27	19.88
11	0.0001	64	64	20	78.07	1.1057	2.3118	14.29
12	0.0001	64	128	20	78.5	1.121	2.3667	18.11

Experimentos Aplicando Segunda Capa, Regularización L2 y Optimizador Adam:

Attempt	Learning Rate	Batch Size	Hidden Layer Size	Epochs	Accuracy	Loss	Initial Acc	Initial Loss
1	0.1	32	64	20	69.11	0.9	1.54	48.76
2	0.1	32	128	20	93.95	0.1914	0.741	88.06
3	0.1	64	64	20	91.13	0.2953	0.8106	83.9
4	0.1	64	128	20	82.25	0.5915	1.3507	56.19
5	0.001	32	64	20	97.21	0.057	0.2879	94.99
6	0.001	32	128	20	97.91	0.03	0.2302	96.1
7	0.001	64	64	20	97.13	0.053	0.3224	94.78
8	0.001	64	128	20	98.02	0.0202	0.2582	96.03
9	0.0001	32	64	20	93.57	0.2351	0.7763	90.01
10	0.0001	32	128	20	94.61	0.1918	0.585	91.38
11	0.0001	64	64	20	93.71	0.2173	0.9429	88.89
12	0.0001	64	128	20	95.06	0.1729	0.7128	91.06



Curvas de Entrenamiento

- **Pérdida (Loss):**
 - La reducción es consistente, indicando aprendizaje efectivo.
 - Ejemplo de progresión (2 capas ocultas, Adam, LR=0.1):
 - Época 0: 1.3507
 - Época 10: 0.8912
 - Época 20: 0.5915
 - Con regularización L2 ($\lambda = 0.0001$), la pérdida aumenta ligeramente, pero mejora la generalización.
- **Precisión en Validación:**
 - Mejora progresiva sin sobreajuste severo.

- Ejemplo (2 capas ocultas, Adam, LR=0.1):

- Época 0: 56.19%
- Época 10: 80.32%
- Época 20: 82.25%

Precisión Final en Pruebas

- **Configuración Básica** (1 capa oculta, SGD, sin regularización): **~97.96%**.
- **Configuración Avanzada** (2 capas ocultas, Adam, L2): **~98.02%**.

Desafíos

1. Confusión en el Entrenamiento

Problema: Observé que la red neuronal sin Adam (usando SGD simple) con una tasa de aprendizaje alta (0.1) lograba mayor precisión en menos épocas que la configuración con Adam y la misma tasa. Esto contradecía lo esperado, ya que Adam debería optimizar el aprendizaje.

Causa: Adam tiene un mecanismo de tasa de aprendizaje adaptativa que ajusta automáticamente el paso de actualización. Al usar una tasa inicial alta (0.1) junto con Adam, las actualizaciones se volvían inestables debido a la división por la raíz cuadrada de la cache (segundo momento), lo que reducía la magnitud efectiva del paso.

Solución: Reduje la tasa de aprendizaje para Adam a 0.001, un valor estándar para el optimizador Adam. También añadí corrección de bias en los momentos (momentum y cache) para evitar pasos demasiado pequeños en las primeras iteraciones.

2. Inicialización de Pesos

Problema: Con inicialización tradicional ($\text{np.random.randn()} * 0.01$), la red no convergía o mostraba pérdidas extremas.

Causa: Pesos iniciales demasiado pequeños, lo que causaba gradientes cercanos a cero en capas con ReLU ("neuronas muertas").

Solución: Implementé inicialización de He para capas con ReLU: $\text{np.sqrt}(2. / \text{input_size})$

Esto escaló los pesos según la dimensión de entrada, manteniendo varianza óptima para la propagación de gradientes.

3. Dificultades en el Método Backward

Problema: Intenté implementar el gradiente de softmax y cross-entropy por separado, lo que generaba cálculos complejos e inestables.

Causa: La derivada de la pérdida con respecto a los logits (antes de softmax) requiere el Jacobiano de softmax, que es computacionalmente costoso.

Solución: Aproveché la derivada cuando se combinan softmax y cross-entropy: $(y_{\text{pred}} - y_{\text{true}}) / \text{batch_size}$. Con este cambio integré el cálculo del gradiente en una sola operación, evitando usar matriz Jacobiana y el backwards de cross-entropy.