Universidad del Valle de Guatemala

Facultad de Ingeniería

Data Science

**LABORATORIO #2**

**Redes Neuronales**

Gabriel Paz 221087

Carlos Valladares 221164

03 de agosto del 2025, Guatemala de la Asunción

**Resumen**

Este informe presenta los resultados de una serie de experimentos con redes neuronales sobre el conjunto MNIST, donde se analizaron hiperparámetros clave (ancho de capas, profundidad, funciones de activación, tamaño de batch, tasa de aprendizaje y regularización). Se incluye una tabla comparativa de todos los experimentos, gráficas ilustrativas, un análisis detallado de sus efectos, las conclusiones sobre el impacto de cada hiperparámetro y la descripción/justificación técnica del modelo óptimo que alcanza ≥ 98.5 % de precisión de validación.

**Métodos**

Se utilizó una función común run\_experiment(cfg, epochs, batch\_size) basada en TensorFlow 2/Keras que:

1. Construye un modelo secuencial a partir de la configuración cfg.
2. Entrena durante un número fijo de épocas y batch size.
3. Devuelve la precisión de validación y el tiempo de entrenamiento.

Todas las redes comparten:

* Conjunto de datos: MNIST, normalizado a [0,1] y aplanado a vectores de 784 entradas.
* Optimizer: SGD.
* Métricas: sparse\_categorical\_crossentropy y accuracy.

**Resultados**

**Tabla 1  
Comparación de configuraciones experimentales**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ejercicio** | **Configuración clave** | **Precisión validación** | **Tiempo entrenamiento (s)** |
| 1 | Ancho capa = [500] | 0.9382 | 17.2 |
| 2 | Capas ocultas = [200, 100] | 0.9435 | 15.38 |
| 3 | Profundidad = 5 capas × 100 neuronas | 0.9585 | 13.77 |
| 4 | Activación = sigmoid × 2 | 0.8025 | 14.72 |
| 5 | Activaciones = [ReLU, Tanh] | 0.9475 | 15.39 |
| 6 | Batch size = 10 000 | 0.613 | 3.74 |
| 7 | Batch size = 1 (SGD puro) | 0.9822 | 903.75 |
| 8 | Learning rate = 0.0001 | 0.464 | 13.6 |
| 9 | Learning rate = 0.02 | 0.9467 | 13.5 |
| 10 | Dropout = 0.5 + L2 = 1e-4 | 0.9358 | 15.81 |
| 12 | Óptimo: [300, 150], LR = 0.005, Dropout = 0.3, L2 = 1e-4 | 0.9862 | 27.85 |

*Nota: El ejercicio 11 corresponde a la generación de gráficas (no cuantifica en tabla).*

**Figuras relevantes**

* **Figura 1.** Val Accuracy vs Ancho de capa.

Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.

* **Figura 2.** Train Time vs Profundidad.

Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.

* **Figura 3.** Curvas de precisión y pérdida durante el entrenamiento.

Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.

Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.

* **Figura 4.** Comparación de val\_acc entre configuraciones seleccionadas.

Gráfico, Gráfico de barras

El contenido generado por IA puede ser incorrecto.

**Análisis detallado**

**Ancho vs Profundidad (Ej. 1–3):**

* Aumentar neuronas (50→500) mejora val\_acc (+0.0097) pero alarga el entrenamiento (+5.7 s).
* Profundizar (1→5 capas) aporta mayor ganancia (+0.025) con un sobrecoste moderado (< 1 s).

**Funciones de activación (Ej. 4–5):**

* Sigmoid sufre saturación y desvanecimiento (0.8025).
* ReLU+Tanh combina activaciones dispersas y centradas en cero, logrando 0.9475.

**Tamaño de batch (Ej. 6–7):**

* Batch grande (10 000) muy rápido (3.7 s) pero pobre generalización (0.6130).
* SGD puro (1) incurre en alto overhead (903.7 s) pero alcanza excelente val\_acc (0.9822) gracias al ruido regularizador.

**Tasa de aprendizaje (Ej. 8–9):**

* lr muy baja (0.0001) estanca la red (~0.46).
* lr alta (0.02) acelera convergencia pero con oscilaciones, alcanzando 0.9467.

**Regularización (Ej. 10):**

* Dropout+L2 (0.5,1e-4) reduce modesto overfitting, val\_acc 0.9358, +0.43 s de entrenamiento.

**Modelo Óptimo (Ej. 12)**

**Configuración**

* Capas ocultas: [300, 150]
* Activaciones: ReLU/ReLU
* Learning rate: 0.005
* Dropout: 0.3
* L2: 1×10⁻⁴
* Épocas: 15
* Batch size: 128

**Resultado:**

Óptimo → val\_acc = 0.9862, time = 27.85 s

**Justificación técnica**

1. Dos capas equilibran capacidad y costo computacional.
2. ReLU evita saturación y facilita convergencia.
3. LR 0.005 acelera sin perder estabilidad.
4. Dropout 0.3 regulariza sin dañar la señal.
5. L2 1e-4 penaliza pesos grandes y mejora robustez.
6. Batch 128 ofrece balance entre ruido y eficiencia.

**Conclusión**

Los experimentos muestran que optimizar hiperparámetros es clave para maximizar precisión y eficiencia. Aunque el SGD puro logra la mayor val\_acc, su costo de tiempo es prohibitivo. La configuración óptima propuesta (ReLU, dos capas, lr moderado y regularización combinada) alcanza ≥ 98.6 % en < 30 s, representando el mejor compromiso general para MNIST.

**Referencias**

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.  
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15, 1929–1958.  
TensorFlow Development Team. (2023). *TensorFlow Documentation*. <https://www.tensorflow.org/>