Laboratorio 2.
Redes Neuronales con MNIST

INTRODUCCIÓN:

Este laboratorio está diseñado para que los estudiantes experimenten con redes neuronales utilizando el conjunto de datos MNIST. El objetivo principal es comprender cómo diferentes hiperparámetros y configuraciones afectan el rendimiento y la eficiencia de los modelos de redes neuronales.

OBJETIVOS DEL APRENDIZAJE

Al finalizar este laboratorio, los estudiantes serán capaces de:

- Modificar y ajustar hiperparámetros en redes neuronales
- Analizar el impacto de diferentes arquitecturas en el rendimiento del modelo
- Evaluar el equilibrio entre precisión y tiempo de ejecución
- Documentar y comparar resultados de experimentos con redes neuronales
- Implementar técnicas de optimización para mejorar modelos existentes

REQUISITOS PREVIOS

- Jupyter Notebook con el código base visto en clase
- Conocimientos básicos de TensorFlow/Keras
- Comprensión de los conceptos fundamentales de redes neuronales

INSTRUCCIONES GENERALES

1. Utilice el código base proporcionado en clase como punto de partida
2. Para cada experimento, documente:
    - Configuración utilizada
    - Precisión de entrenamiento y validación
    - Tiempo de ejecución
    - Observaciones relevantes
3. Presente sus resultados en forma de tablas comparativas y gráficos
4. Incluya un análisis final con conclusiones sobre los experimentos realizados

EJERCICIOS

1. Modificación del Ancho de la Red (8 puntos)

- Modifique el tamaño de la capa escondida a 200 neuronas.
- ¿Cómo cambia la precisión de validación del modelo?
- ¿Cuánto tiempo tarda el algoritmo en entrenar?
- Experimente con diferentes tamaños de capa escondida (50, 100, 300, 500) y determine cuál ofrece el mejor rendimiento.

2. Modificación de la Profundidad de la Red (12 puntos)

Agregue una capa escondida adicional al modelo.

- Documente cuidadosamente las dimensiones de los pesos y sesgos
- Compare la precisión de validación con el modelo original
- Analice el impacto en el tiempo de ejecución
- Explique los cambios necesarios en el código para implementar esta modificación

3. Redes Profundas (12 puntos)

Experimente con arquitecturas más profundas, llegando hasta 5 capas escondidas.

- Ajuste el ancho de cada capa según considere conveniente
- Documente la precisión de validación para cada configuración
- Analice la relación entre profundidad y tiempo de ejecución
- Identifique posibles problemas de desvanecimiento del gradiente

4. Funciones de Activación I (8 puntos)

Aplique la función de activación sigmoidal a todas las capas.

- Compare el rendimiento con las activaciones originales
- Analice el impacto en la velocidad de convergencia

5. Funciones de Activación II (8 puntos)

Aplique ReLU a la primera capa escondida y tanh a la segunda.

- Compare el rendimiento con las configuraciones anteriores
- Explique las ventajas y desventajas de cada función de activación

6. Tamaño de Batch Grande (5 puntos)

Modifique el tamaño de batch a 10,000.

- Documente el cambio en el tiempo de entrenamiento
- Analice el impacto en la precisión del modelo
- Explique teóricamente por qué se observan estos cambios

7. Descenso de Gradiente Estocástico (SGD) (5 puntos)

Ajuste el tamaño de batch a 1 (SGD puro).

- Compare el tiempo de ejecución con configuraciones anteriores
- Analice la estabilidad y precisión del entrenamiento
- Explique si los resultados son coherentes con la teoría

8. Tasa de Aprendizaje Baja (4 puntos)

Modifique la tasa de aprendizaje a 0.0001.

- Documente el impacto en la convergencia del modelo
- Analice si el modelo alcanza mejor precisión o se queda atrapado en mínimos locales

9. Tasa de Aprendizaje Alta (4 puntos)

Ajuste la tasa de aprendizaje a 0.02.

- Documente el impacto en la estabilidad del entrenamiento
- Analice si se produce divergencia o mejora en la velocidad de convergencia

10. Optimización Avanzada (10 puntos)

Implemente técnicas de regularización:

- Agregue dropout entre capas (pruebe diferentes tasas)
- Experimente con regularización L2
- Documente el impacto en la generalización del modelo

11. Visualización (5 puntos)

Cree gráficos que muestren:

- Evolución de la precisión y pérdida durante el entrenamiento
- Comparación de rendimiento entre diferentes configuraciones
- Visualización de algunos filtros o características aprendidas por la red

12. Modelo Óptimo (14 puntos)

Combine todos los métodos anteriores para diseñar un modelo que:

- Alcance una precisión de validación de 98.5% o superior
- Optimice el tiempo de entrenamiento
- Presente la arquitectura más eficiente posible
- Justifique cada decisión de diseño

MATERIAL A ENTREGAR

1. Jupyter Notebook con todos los experimentos realizados y código comentado
2. Informe (máximo 5 páginas) que incluya:
    - Tabla comparativa de todos los experimentos realizados
    - Gráficos relevantes que ilustren los resultados obtenidos
    - Análisis detallado de los resultados, comparando las diferentes configuraciones
    - Conclusiones fundamentadas sobre el impacto de cada hiperparámetro
    - Descripción y justificación técnica del modelo óptimo desarrollado

Aspectos que se Evaluarán

- Claridad y estructura lógica del documento
- Precisión técnica en la descripción de los experimentos
- Profundidad del análisis comparativo entre configuraciones
- Calidad de las visualizaciones y tablas presentadas
- Solidez de las conclusiones basadas en evidencia experimental

Entregable

Un documento PDF o Word con el informe completo, siguiendo las pautas establecidas y manteniendo un formato académico apropiado

RÚBRICA DE EVALUACIÓN (100 puntos)

Criterio	Puntos	Descripción
Ejercicios Básicos (1 - 9)	66	Correcta implementación y análisis de los experimentos 1-9 según los puntos asignados a cada uno
Optimización Avanzada	10	Implementación efectiva de técnicas de regularización y análisis de su impacto
Visualización	5	Calidad y relevancia de las visualizaciones presentadas
Modelo Óptimo	14	Rendimiento del modelo final y justificación de las decisiones de diseño
Calidad del Informe	5	Claridad, organización y profundidad del análisis presentado

RECOMENDACIONES FINALES

- Gestione eficientemente el tiempo de ejecución de los experimentos
- Guarde los resultados intermedios para evitar repetir entrenamientos largos
- Documente meticulosamente cada experimento mientras lo realiza
- Analice patrones y tendencias en los resultados para informar experimentos posteriores
- Consulte la documentación oficial de TensorFlow/Keras para implementaciones específicas