FACULTAD DE INGENIERÍA
ESTADÍSTICA E INFORMÁTICA

UNIVERSIDAD NACIONAL DEL ALTIPLANO - PUNO

AJUSTES DE HIPERPARAMETROS

Integrantes:
Cacasaca Pilco Noemí
Condori Mamani Herson Romario
Muñoz Anccori Edilfonso
Mamani Rodriguez Marco Paul

6 de febrero de 2025

Contenido

Introduccion

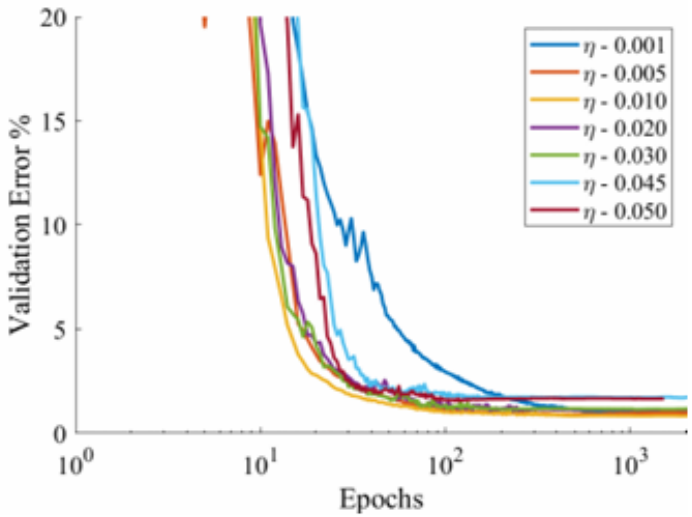
Introduccion

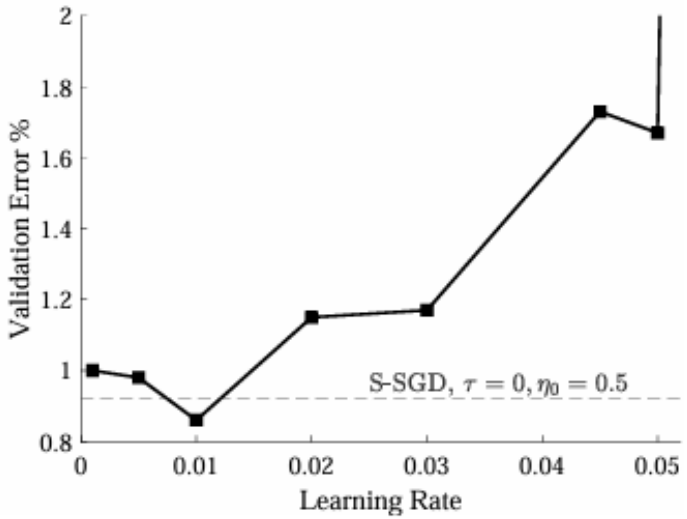
El entrenamiento de redes neuronales profundas (DNN) requiere grandes recursos computacionales, utilizando múltiples dispositivos durante días o semanas. Con el aumento del tamaño de los modelos y la cantidad de datos, el uso de cómputo ha crecido exponencialmente en los últimos años. Las operaciones de entrenamiento de DNN son adecuadas para la paralelización, comúnmente a través de "paralelización de datos", donde los datos se dividen en lotes distribuidos entre los dispositivos. Esto permite entrenar con una gran cantidad de muestras sin un alto costo en tiempo de ejecución. Sin embargo, el entrenamiento se realiza de forma síncrona, lo que genera un cuello de botella debido a la necesidad de sincronización entre dispositivos en cada iteración.

¿Por qué usamos DNN en aprendizaje automático?

- Capacidad de modelar relaciones complejas: Pueden capturar patrones no lineales en los datos, superando enfoques tradicionales como regresión lineal o máquinas de soporte vectorial
- Escalabilidad con grandes volúmenes de datos: A medida que los conjuntos de datos crecen, las DNN pueden aprovechar su profundidad para mejorar la precisión del modelo.
- Automatización del aprendizaje de características: En problemas como la clasificación de imágenes, las DNN aprenden representaciones directamente de los datos, eliminando la necesidad de ingeniería manual de características.
- Mejoras en hardware y computación distribuida: El avance en GPUs y TPUs permite entrenar modelos más grandes en tiempos razonables.

Tasa de Aprendizaje





Problema de minimización

Minimización

- **Problema :**
  - Minimizar la pérdida empírica $f(\mathbf{x}) = \sum_{i=1}^{N} f_i(\mathbf{x})$.
  - N es el número de muestras.
  - f_i son funciones continuamente diferenciables.

Regla de actualización

Regla de Actualización (A-SGD):

1. Utiliza el descenso de gradiente estocástico asíncrono.
2. Fórmula: $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \nabla f_{n(t-\tau)}(\mathbf{x}_{t-\tau}).$

Tasa de Aprendizaje (η) y Retraso (τ):

1. η es la tasa de aprendizaje.
2. τ es el retraso debido a la naturaleza asíncrona del entrenamiento.
3. Se enfoca en un retraso fijo τ.

Objetivo Principal

- .Comprender cómo el retraso del gradiente y la tasa de aprendizaje afectan el proceso de selección de mínimos en redes neuronales.
- Diferenciar entre mínimos donde los hiperparámetros interactúan: convergen o no convergen.

Interacción entre la Tasa de Aprendizaje y el Retraso

Ecuación Característica:

La ecuación dada es:

$$z^{\tau+1} - z^{\tau} + a\eta = 0$$

Este polinomio tiene $\tau + 1$ raíces. Para garantizar la estabilidad, la raíz con la magnitud máxima debe estar dentro del círculo unitario.

Tasa de Aprendizaje Umbral:

Para encontrar la tasa de aprendizaje umbral, se necesita que la raíz máxima esté exactamente en el círculo unitario, lo que asegura que estamos en el umbral de estabilidad. Se muestra que:

$$a\eta = 2 \sin\left(\frac{\pi}{\quad}\right)$$

Aplicaciones

Aproximación de Taylor: Usando la aproximación de Taylor, obtenemos:

$$a\eta = 2\left(\frac{\pi}{4\tau+2} + O\left(\frac{1}{\tau^3}\right)\right) \Rightarrow \frac{1}{a\eta} = \frac{2\tau+1}{\pi} + O\left(\frac{1}{\tau}\right)$$

Error Numérico: Se observa numéricamente que el error entre la solución exacta y la aproximación lineal es menor que 0.05 para $\tau \geq 1$. Esto demuestra la alta precisión de la aproximación analítica. **Implicaciones:** Para mantener la estabilidad para un punto mínimo dado, la tasa de aprendizaje debe mantenerse inversamente proporcional al retraso. Se evalúa la tasa de aprendizaje que asegura que este mínimo siga siendo estable para valores de retraso mayores.

Optimización con Momentum en A-SGD

Ecuaciones clave

$$\mathbf{v}_{t+1} = m\mathbf{v}_t - \eta(1-m)\nabla f_{n(t-\tau)}(\mathbf{x}_{t-\tau})$$

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{v}_{t+1}$$

- m: Parámetro de momentum.
- η: Tamaño del paso de aprendizaje.
- τ: Retardo.

1. **Relación inversa** entre la tasa de aprendizaje (η) y el retardo (τ).
2. **Momentum alto** requiere una tasa de aprendizaje más pequeña para mantener la estabilidad.

Relación con Estudios Previos:

- **Propuesta:** Intercambiar el término de velocidad completo en lugar de solo los gradientes.
- **Ecuación:**

$$\mathbf{v}_{t+1} = m\mathbf{v}_{t-\tau} - \eta(1-m)\nabla f_{n(t-\tau)}(\mathbf{x}_{t-\tau})$$
$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{v}_{t+1}$$

- **Beneficios:** Mejora la estabilidad y permite tasas de aprendizaje más grandes con momentum alto.
- Similar a enfoques de promediado de pesos (Luan et al. (2017); Kassan et al. (2019)).
- Hakimi et al. (2019) utilizaron una variación con momentum de Nesterov, mostrando beneficios empíricos.

Interpretación

Figura: El umbral de estabilidad se mantiene cuando $\eta \propto 1/\tau$. En la figura de la izquierda, se muestra el número de épocas que tarda en divergir de un mínimo en función de la tasa de aprendizaje η. Los círculos negros representan los umbrales de estabilidad, por debajo de los cuales no se escapa del mínimo. En la figura de la derecha, para cada valor de retardo τ, se muestra $1/\eta$, donde η es la tasa de aprendizaje máxima en la que no se observó divergencia. Debido a la resolución del muestreo, puede haber una desviación de hasta el 8 % de la tasa de aprendizaje máxima encontrada. Esta desviación se representa en las barras de error. VGG-11 entrenada con CIFAR10.

Figure

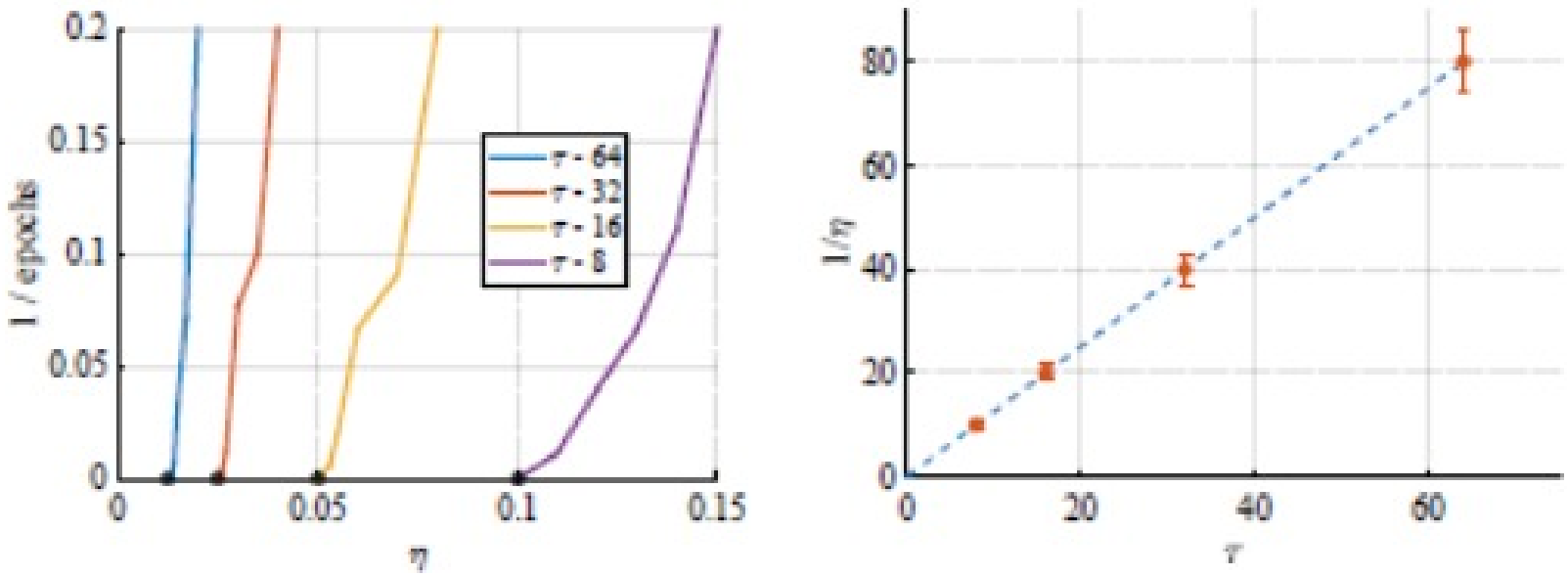


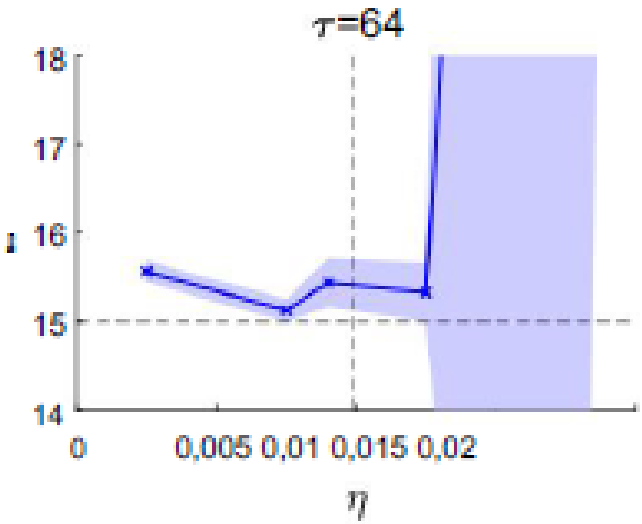
Figura: figue

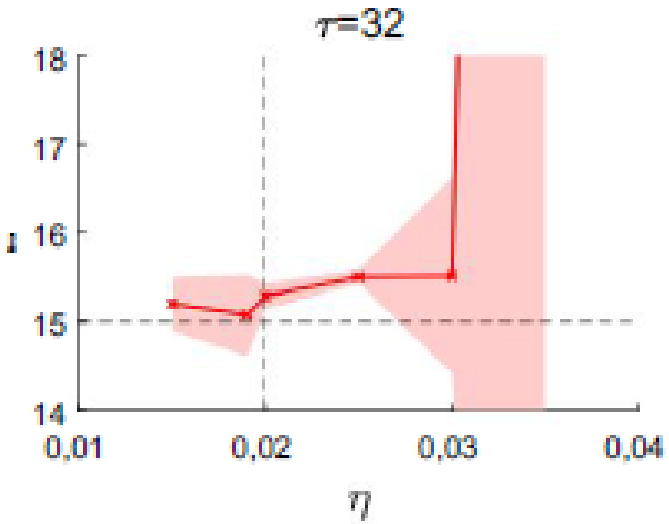
Contenido

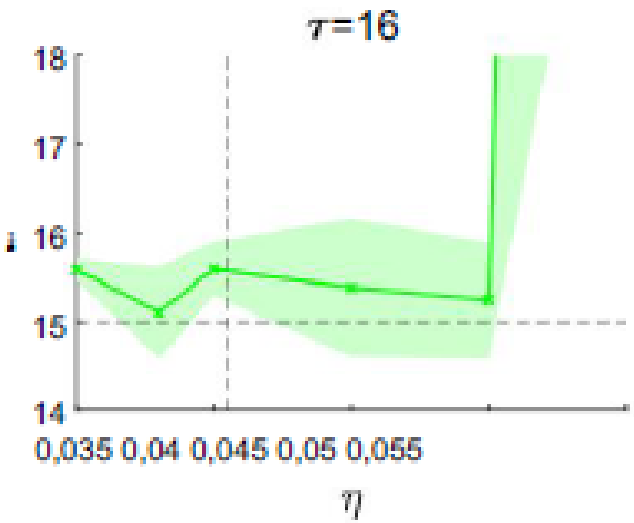
Selección de Mínimos y Hiperparámetros

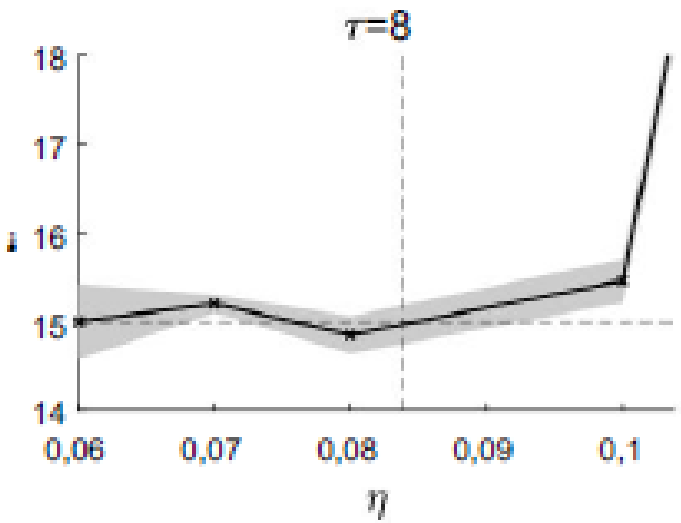
- Se realizaron experimentos con diferentes tasas de aprendizaje.
- Evaluación del impacto del retraso en la convergencia de los mínimos.
- Comparación entre distintos esquemas de optimización.
- Análisis de la estabilidad de los mínimos alcanzados.

GRAFICO


τ=64
18
17
16
15
14
0
0,005 0,01 0,015 0,02
η


τ=32
18
17
16
15
14
0,01
0,02
0,03
0,04
η


τ=16
18
17
16
15
14
0,035 0,04 0,045 0,05 0,055
η


τ=8
18
17
16
15
14
0,06
0,07
0,08
0,09
0,1
η

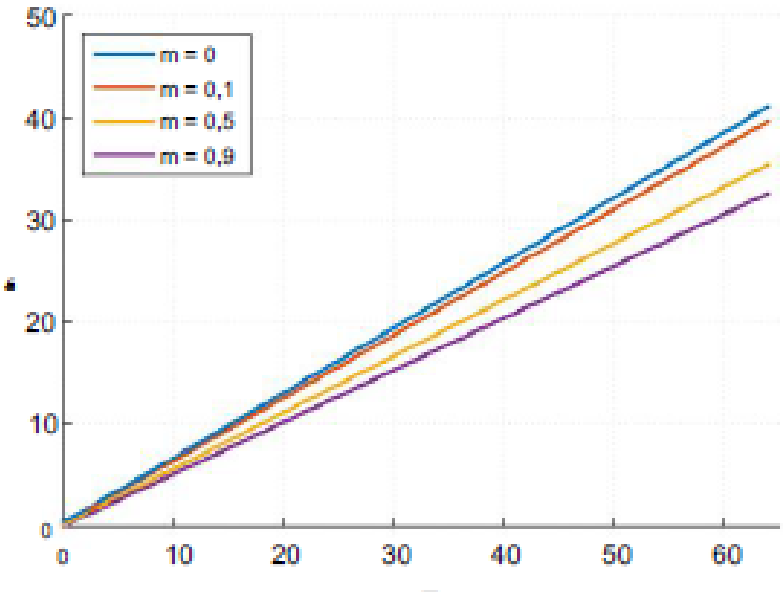
Interpretación del Gráfico: Selección de Mínimos

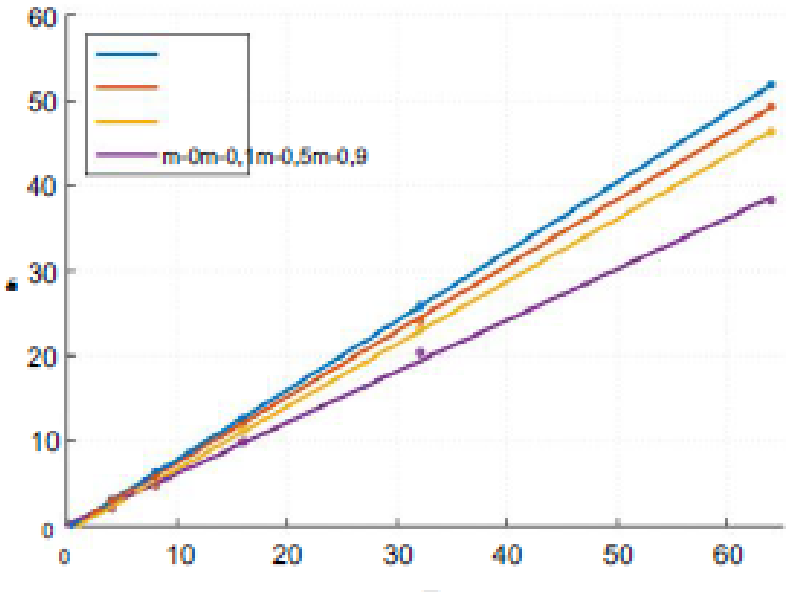
- Se observa que el ajuste de la tasa de aprendizaje tiene un impacto directo en la estabilidad de los mínimos alcanzados.
- Para valores altos de la tasa de aprendizaje, la convergencia puede ser inestable, afectando la selección de los mínimos globales.
- El gráfico muestra que a medida que el retraso aumenta, la calidad de los mínimos encontrados disminuye si no se ajustan adecuadamente los hiperparámetros.
- Se recomienda una tasa de aprendizaje adaptativa que varíe inversamente con el retraso para mantener la estabilidad en la optimización.

Optimización en Entrenamiento Asíncrono

- Evaluación del impacto del entrenamiento asíncrono sobre la generalización.
- Análisis de estrategias para mitigar la obsolescencia del gradiente.
- Ajuste dinámico de la tasa de aprendizaje en función del retraso.
- Comparación entre entrenamiento síncrono y asíncrono.

GRAFICO





Interpretación del Gráfico: Mejora del Entrenamiento Asíncrono

- El gráfico demuestra que la asincronía en el entrenamiento puede generar una degradación en la generalización si no se ajustan correctamente los hiperparámetros.
- Se observa que con un alto retraso en la actualización de gradientes, la convergencia del modelo se ralentiza y puede llevar a mínimos subóptimos.
- Reducir la tasa de aprendizaje de manera proporcional al retraso permite mitigar la inestabilidad observada en el gráfico.
- Se recomienda el uso de técnicas de estabilización, como la modificación del momentum o el ajuste adaptativo de los hiperparámetros, para mejorar el desempeño en entornos asíncronos.

Contenido

EJERCICIO

- Ajuste de hiperparámetros en redes neuronales:
  - Variación de la tasa de aprendizaje y el tamaño de mini-batch.
  - Registro del comportamiento de la función de costo y la exactitud en validación.
  - Determinación de la combinación óptima para un mejor balance entre velocidad de entrenamiento y calidad de la solución final.

  [**Arcos-Medina2019**, **tavara2019parallel**, **tavara2019parallel**]

Código de Python para Ajuste de Hiperparámetros

```
import torch
import torch.nn as nn
import torch.optim as optim
import torchvision
import torchvision.transforms as transforms
import matplotlib.pyplot as plt
import numpy as np

# Definir transformaciones para normalizar los dato
transform = transforms.Compose([transforms.ToTensor
                                transforms.Normaliz

# Cargar dataset FashionMNIST
trainset = torchvision.datasets.FashionMNIST(root='
testset = torchvision.datasets.FashionMNIST(root='.
```

Resultados del Ajuste de Hiperparámetros

- Combinaciones de hiperparámetros probadas:
  - Tasas de aprendizaje: 0.001, 0.01, 0.1.
  - Tamaños de mini-batch: 32, 64, 128.
- Mejor combinación:
  - Tasa de aprendizaje: 0.001.
  - Tamaño de mini-batch: 32.
  - Exactitud en validación: aproximadamente 91.71 %.

Gráfica de Resultados

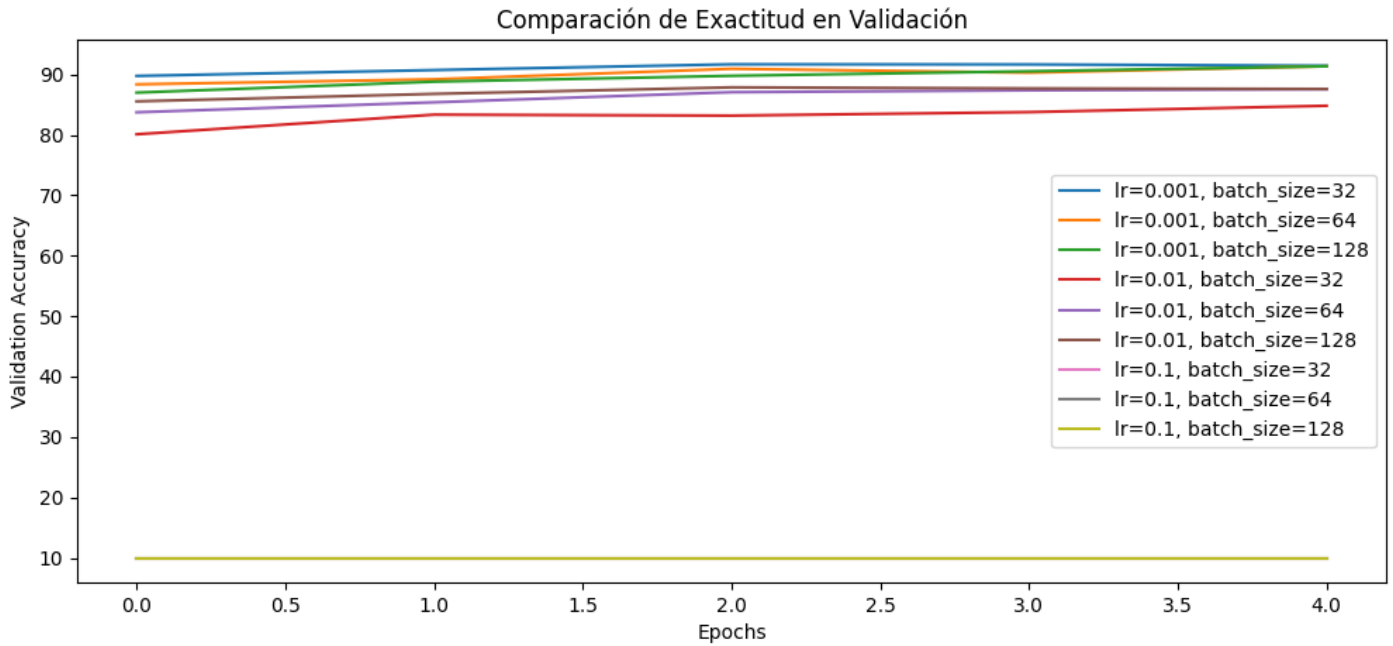


Figura: Comparación de Exactitud en Validación