TRABAJO ACADÉMICO 2

Reconocimiento de formas y apredizaje automático

Autora
Aitana Menárguez Box
Diciembre 2023

Índice

1 Introducción

El objetivo de este trabajo es el de experimentar con el uso de redes neuronales para el procesamiento de imágenes. En concreto, se trata de entrenar, optimizar y evaluar redes neuronales para la clasificación de diferentes colecciones de imágenes.
Se trata de una extensión del trabajo académico 1 y se utilizarán los mismos conjuntos de datos explicados en la memoria anterior: MNIS y Fashion-MNIST.

A lo largo de este trabajo se han probado diferentes arquitecturas neuronales, así como parámetros para entrenarlas. En primer lugar, se ha probado una red, que se ha entrenado con diferentes parámetros, para la tarea de MNIST. A partir de estos resultados, se ha intentado modificar la aquitectura para obtener un menor error de clasificación y se ha probado dicha arquitectura con el *dataset* de Fashion-MNIST. Por último, se ha experimentado con una arquitectura distinta para éste conjunto de datos.

2 Arquitecturas utilizadas

2.1 Red feedforward

Se ha utilizado el módulo *nn.Sequential* de PyTorch para crearla. A continuación se explican las capas empleadas:

- **Capa de entrada** (Linear 1): es una capa completamente conectada que tiene 784 neuronas de entrada (se corresponden con una imagen de 28x28 píxeles) y 1024 neuronas de salida a la capa oculta.
- **Función de activación** (ReLU 1): se aplica la función de activación *Rectified Linear Unit* que introduce no linealidades en la red; ayuda a aprender patrones más complejos.
- **Capa oculta 2** (Linear 2): otra capa completamente conectada que toma los 1024 valores de la capa anterior y produce 1024 salidas.
- **Función de activación** (ReLU 2)
- **Capa oculta 3** (Linear 3)
- **Función de activación** (ReLU 3)
- **Capa de salida** (Linear 4): la última capa completamente conectada tiene 1024 entradas y produce 10 salida, ya que, en este caso, el modelo está destinado a la clasificación en 10 clases (se trata de un problema de reconocimiento de dígitos).

Se ha realizado una versión de esta red con el uso de *dropout* (0.5) entre las capas para tratar de evitar que el modelo sobreajuste.
Por otro lado, cabe destacar que **no** se ha aplicado la función *softmax* en la capa de salida de la red, ya que se utilizará el criterio de optimización *cross entropy* en el entramiento y éste, internamente en PyTorch, aplica dicha función.

2.2 Red convolucional

En el caso de Fashion-MNIST se tienen imágenes más complejas que las de los dígitos (son prendas de ropa) y es por eso que pueden considerarse alguna arquitectura más elaborada para resolver la tarea. En este caso se ha optado por una red neuronal convolucional.
El esquema básico de esta red es el siguiente:

- **Capa de Convolución 1**
  - Entrada: Imágenes en escala de grises (1 canal).
  - Salida: 32 mapas de características.
  - Kernel/Filter: Tamaño 3x3.
  - Padding: 1 píxel para mantener el tamaño de la entrada.
  - Función de activación: ReLU (Rectified Linear Unit).
- **Capa de Pooling 1**
  - Tipo: MaxPooling.
  - Kernel/Filter: Tamaño 2x2.
  - Stride: 2 píxeles.
- **Capa de Convolución 2**
  - Entrada: 32 mapas de características.
  - Salida: 64 mapas de características.
  - Kernel/Filter: Tamaño 3x3.
  - Padding: 1 píxel para mantener el tamaño de la entrada.
  - Función de activación: ReLU.
- **Capa de Pooling 2**
  - Tipo: MaxPooling.
  - Kernel/Filter: Tamaño 2x2.
  - Stride: 2 píxeles.
- **Capa Totalmente Conectada (Fully Connected) 1**
  - Entrada: Flatten de la salida de la capa anterior.
  - Salida: 128 unidades.
  - Función de activación: ReLU.
- **Capa Totalmente Conectada 2**
  - Entrada: 128 unidades.
  - Salida: 10 unidades (clases; predicciones finales).
  - Activación lineal

3 Experimentación y resultados

3.1 MNIST

En primer lugar, se han probado diferentes valores de algunos de los hiperparámetros de la red *feedforward* para entrenarla con MNIST. En concreto se ha probado con:

- Tamaño de *batch*: 100, 200 y 500
- Factor de aprendizaje: 0.1, 0.01, 0.001 y 0.0001
- Optimizador: SGD (descenso por gradiente estocástico) y Adam.

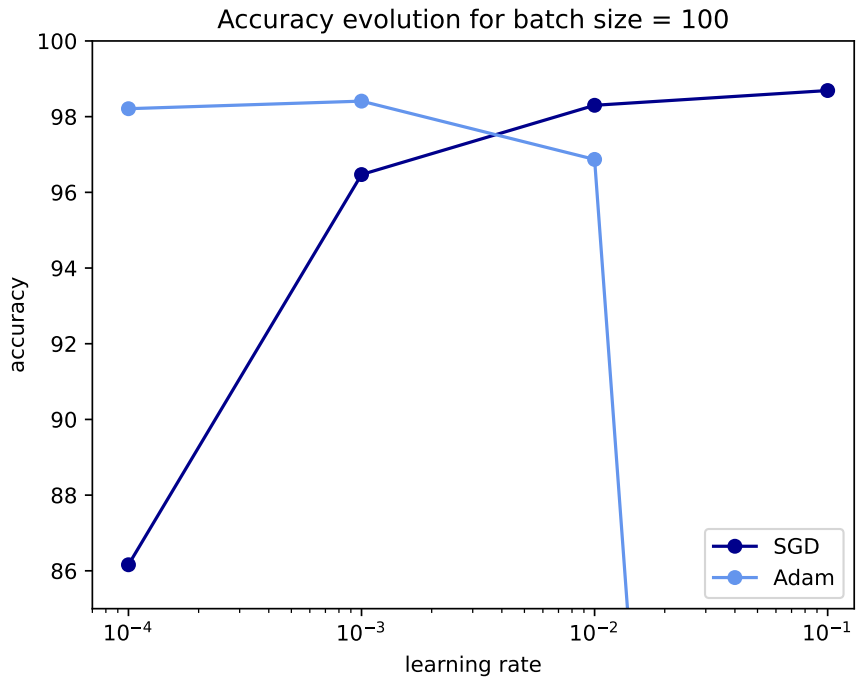

Figura 1: Máxima accuracy obtenida en cada entrenamiento del modelo feedforward para diferentes tasas de aprendizaje y optimizadores, para un tamaño de batch igual a 100.

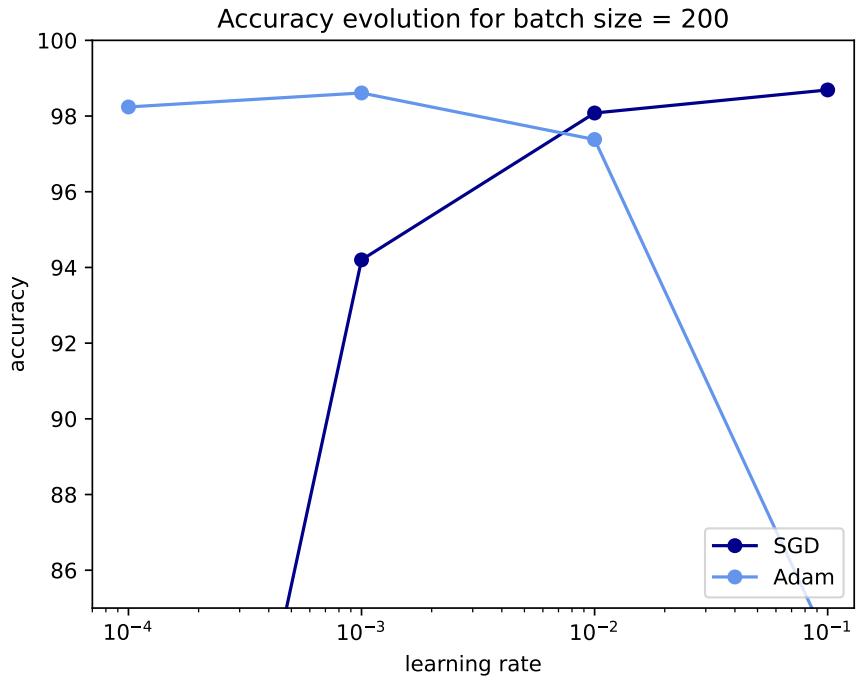

Figura 2: Máxima accuracy obtenida en cada entrenamiento del modelo feedforward para diferentes tasas de aprendizaje y optimizadores, para un tamaño de batch igual a 200.

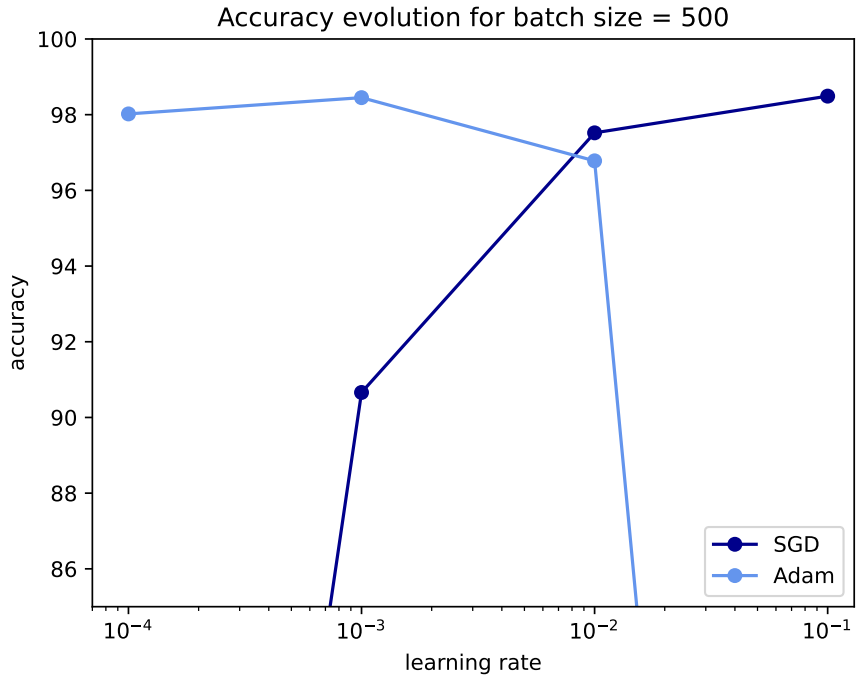

Figura 3: Máxima accuracy obtenida en cada entrenamiento del modelo feedforward para diferentes tasas de aprendizaje y optimizadores, para un tamaño de batch igual a 500.

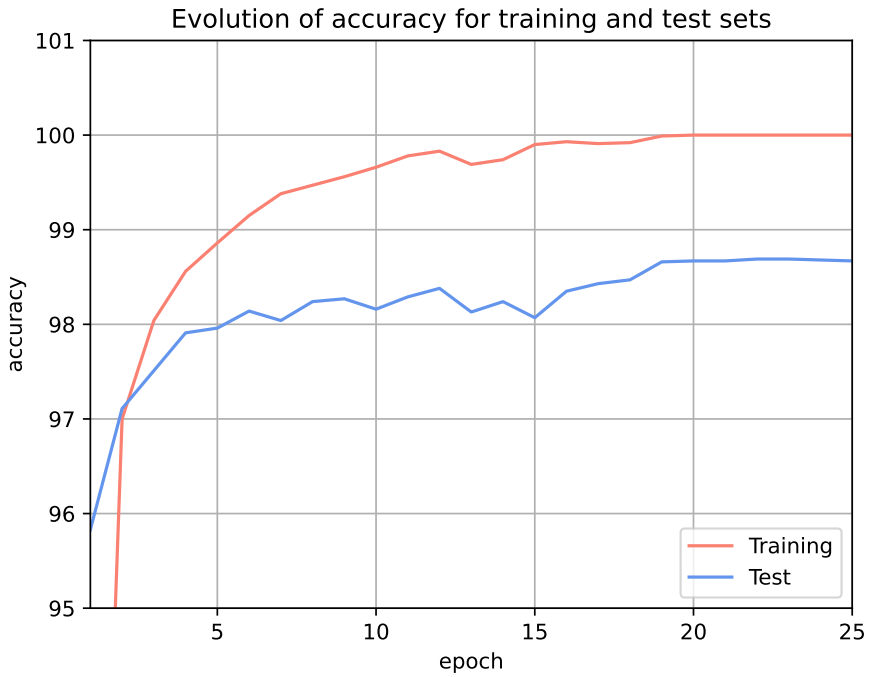

Figura 4: Evolución de la precisión en el entranamiento y test para el mejor modelo obtenido en la experimentación para MNIST.

Como se puede observar en las figuras 1, 2 y 3, la mejor combinación de parámetros es la de tamaño de batch de 200, tasa de aprendizaje igual a 0.1 y optimizador SGD. Su precisión exactamente es del 98,69%. En la figura 4 se tiene la evolución de la *accuracy* del entrenamiento y test del modelo. Se puede apreciar que llega al 100% en el entrenamiento. Esto puede llevar a pensar que se está sobrentrenando al modelo.
Por eso, se ha probado este mismo modelo modificando la arquitectura de la red, añadiendo un *dropout* entre las capas igual a 0.5, este apaga las neuronas de forma arbitraria, lo cual puede hacer que no se sobrentrene el modelo en exceso. La precisión máxima obtenida en este caso ha sido del 98.39%. Al inspeccionar la evolución de las precisiones en el entrenamiento y el test, se ve que no se consigue el 100% en el entrenamiento, pero eso no hace que mejore de ninguna forma la precisión máxima obtenida.

3.2 Fashion-MNIST

Con los parámetros óptimos encontrados para la red anterior, sin el *dropout*, la precisión obtenida al entrenarla con el *dataset* de Fashion-MNIST ha sido del 89.74%.
Por otro lado, se ha experimentado también con diferentes valores de hiperparámetros dentro de la red convolucional mencionada, y para este nuevo conjunto de datos. En concreto se han probado:

- Tamaño de *batch*: 100 y 200
- Factor de aprendizaje: 0.01 y 0.001
- Optimizador: SGD (descenso por gradiente estocástico) y Adam.

En este caso, se han probado menos valores dado que el tiempo de ejecución del entrenamiento y el test aumentaba, al tratarse de una red más compleja. Los resultados pueden verse en las figuras 5 y 6. En las cuales se observa que la mejor combinación de parámetros es la de tamaño de batch igual a 100, tasa de aprendizaje 0.01 y optimizador SGD, con una precisión de 91.72%.

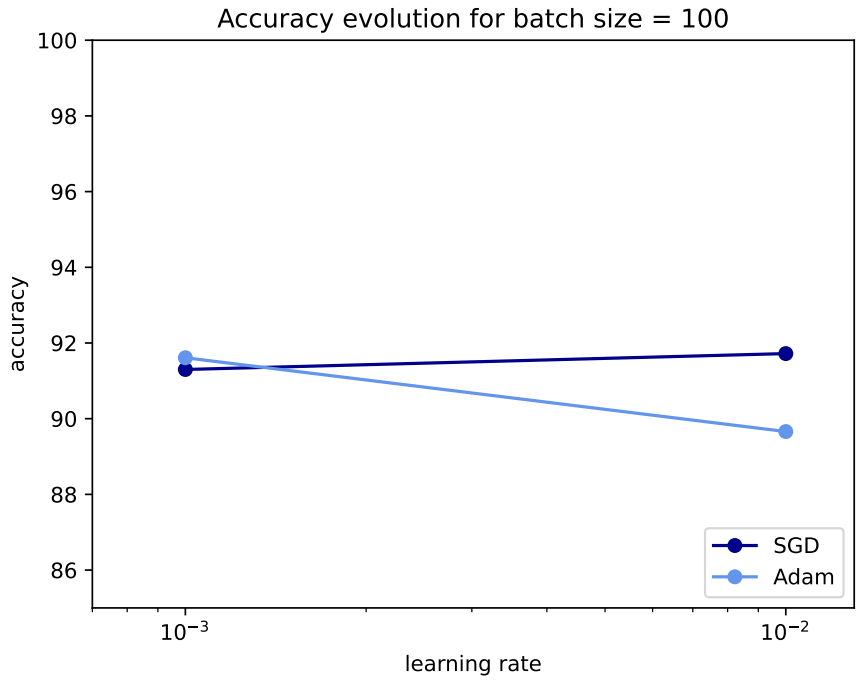

Figura 5: Máxima accuracy obtenida en cada entrenamiento del modelo convolucional para diferentes tasas de aprendizaje y optimizadores, para un tamaño de batch igual a 100.

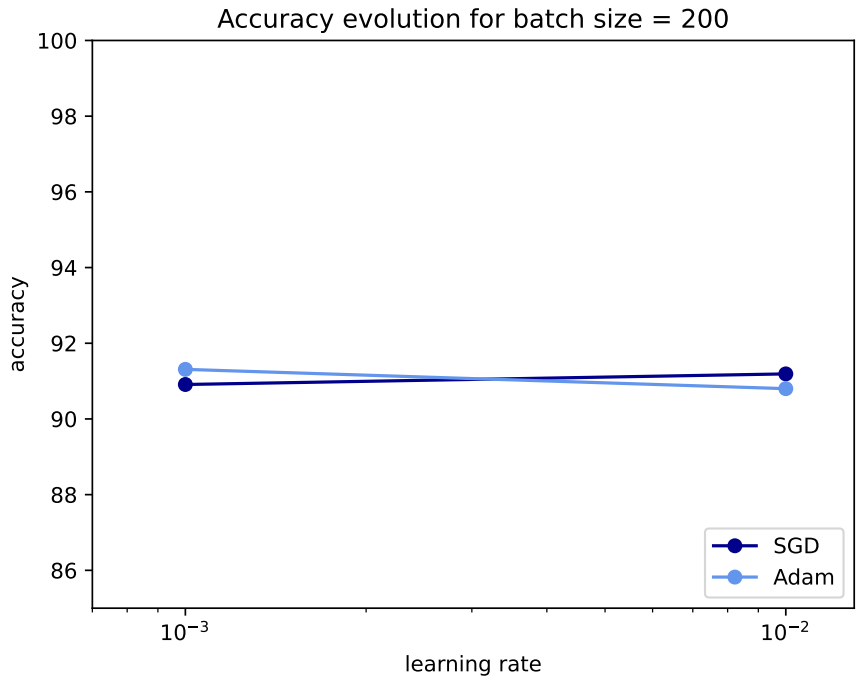

Figura 6: Máxima accuracy obtenida en cada entrenamiento del modelo convolucional para diferentes tasas de aprendizaje y optimizadores, para un tamaño de batch igual a 200.

4 Conclusiones y trabajos futuros

En este trabajo, se han explorado dos arquitecturas de redes neuronales para el reconocimiento de formas en los conjuntos de datos MNIST y Fashion-MNIST. La red feedforward alcanzó una precisión del 98.69% en MNIST, mientras que la red convolucional logró el 91.72% en Fashion-MNIST.
Una observación importante a la hora de analizar los resultados es cómo afecta la tasa de aprendizaje en la precisión del modelo. Mientras que el batch size o el optimizador no suponen una diferencia tan grande, es más sensible el modelo a los cambios en dicha tasa.

Como futuras ampliaciones a este trabajo, se podrían probar aún más combinaciones de hiperparámetros, nuevas arquitecturas o incluso utilizar métodos como *data augmentation* para mejorar la robustez y precisión de las predicciones.