Universidad del Valle de Guatemala
Data Science - Sección 20

Ale Gudiel 19232
Sofia Rueda 19099

Laboratorio 3 - MNIST

El modelo que se desarrolló en clase tiene una precisión ya bastante alta. Sin embargo, hay varios ajustes que se pueden intentar para mejorarlo. Es importante poner atención al tiempo que se tarda cada época en ejecutar. Utilizando el código visto en clase, experimenten con los hiperparámetros del algoritmo.

1. El ancho (tamaño de la capa escondida) del algoritmo. Intenten con un tamaño de 200. ¿Cómo cambia la precisión de validación del modelo? ¿Cuánto tiempo se tardó el algoritmo en entrenar? ¿Puede encontrar un tamaño de capa escondida que funcione mejor?

```
Epoch 1/5
540/540 - 5s - loss: 0.4067 - accuracy: 0.8834 - val_loss: 0.2132 - val_accuracy: 0.9393 - 5s/epoch - 9ms/step
Epoch 2/5
540/540 - 4s - loss: 0.1890 - accuracy: 0.9448 - val_loss: 0.1528 - val_accuracy: 0.9568 - 4s/epoch - 8ms/step
Epoch 3/5
540/540 - 3s - loss: 0.1417 - accuracy: 0.9574 - val_loss: 0.1202 - val_accuracy: 0.9662 - 3s/epoch - 5ms/step
Epoch 4/5
540/540 - 3s - loss: 0.1154 - accuracy: 0.9658 - val_loss: 0.1050 - val_accuracy: 0.9695 - 3s/epoch - 6ms/step
Epoch 5/5
540/540 - 3s - loss: 0.0967 - accuracy: 0.9709 - val_loss: 0.0912 - val_accuracy: 0.9733 - 3s/epoch - 6ms/step

<keras.callbacks.History at 0x127d5be80>
```

Entrenamiento del modelo con ancho de 50

```
Epoch 1/5
540/540 - 6s - loss: 0.2742 - accuracy: 0.9215 - val_loss: 0.1348 - val_accuracy: 0.9622 - 6s/epoch - 10ms/step
Epoch 2/5
540/540 - 4s - loss: 0.1130 - accuracy: 0.9663 - val_loss: 0.0906 - val_accuracy: 0.9742 - 4s/epoch - 8ms/step
Epoch 3/5
540/540 - 4s - loss: 0.0762 - accuracy: 0.9767 - val_loss: 0.0703 - val_accuracy: 0.9812 - 4s/epoch - 7ms/step
Epoch 4/5
540/540 - 4s - loss: 0.0550 - accuracy: 0.9826 - val_loss: 0.0530 - val_accuracy: 0.9830 - 4s/epoch - 7ms/step
Epoch 5/5
540/540 - 4s - loss: 0.0413 - accuracy: 0.9873 - val_loss: 0.0491 - val_accuracy: 0.9848 - 4s/epoch - 7ms/step

<keras.callbacks.History at 0x1273e51f0>
```

Entrenamiento del modelo con ancho de 200

Cuando el tamaño de la capa escondida era de 50, la precisión de validación del modelo estaba entre 0.9393 - 0.9733 y el algoritmo se entrenó en 19.7 segundos. Por otra parte, cuando su tamaño era de 200, la precisión de

validación estaba entre 0.9622 - 0.9848 y se entrenó en 21.6 segundos.

$$tamaño = \sqrt{entrada * salida} = \sqrt{784 * 10} \approx 89$$

No hay una forma exacta de determinar el tamaño óptimo para una capa escondida, pero se puede obtener una aproximación mediante la regla de la pirámide geométrica. Se puede observar, que según esta regla, el mejor tamaño sería de 89 neuronas para la capa escondida.

2. La profundidad del algoritmo. Agreguen una capa escondida más al algoritmo. Este es un ejercicio extremadamente importante! ¿Cómo cambia la precisión de validación? ¿Qué hay del tiempo que se tarda en ejecutar? Pista: deben tener cuidado con las formas de los pesos y los sesgos.

```
Epoch 1/5
540/540 - 6s - loss: 0.4148 - accuracy: 0.8782 - val_loss: 0.1880 - val_accuracy: 0.9442 - 6s/epoch - 11ms/step
Epoch 2/5
540/540 - 5s - loss: 0.1641 - accuracy: 0.9517 - val_loss: 0.1304 - val_accuracy: 0.9615 - 5s/epoch - 9ms/step
Epoch 3/5
540/540 - 4s - loss: 0.1228 - accuracy: 0.9626 - val_loss: 0.1118 - val_accuracy: 0.9667 - 4s/epoch - 8ms/step
Epoch 4/5
540/540 - 4s - loss: 0.0992 - accuracy: 0.9702 - val_loss: 0.0936 - val_accuracy: 0.9745 - 4s/epoch - 8ms/step
Epoch 5/5
540/540 - 4s - loss: 0.0831 - accuracy: 0.9744 - val_loss: 0.0943 - val_accuracy: 0.9730 - 4s/epoch - 8ms/step

<keras.callbacks.History at 0x1280dcb80>
```

Entrenamiento del modelo con una nueva capa escondida

Al agregar una nueva capa escondida, el modelo de entrenamiento se demoró 24.9 segundos para completar su ejecución. Se puede observar que la precisión de validación está entre 0.9442 - 0.9730. Por lo tanto, utilizar más de dos capas escondidas asegura mayor precisión en los datos entrenados.

3. El ancho y la profundidad del algoritmo. Agregue cuantas capas sean necesarias para llegar a 5 capas escondidas. Es más, ajusten el ancho del algoritmo conforme lo encuentre más conveniente. ¿Cómo cambia la precisión de validación? ¿Qué hay del tiempo de ejecución?

```
Epoch 1/5
540/540 - 7s - loss: 0.2757 - accuracy: 0.9164 - val_loss: 0.1267 - val_accuracy: 0.9643 - 7s/epoch - 14ms/step
Epoch 2/5
540/540 - 6s - loss: 0.1092 - accuracy: 0.9669 - val_loss: 0.0820 - val_accuracy: 0.9762 - 6s/epoch - 11ms/step
Epoch 3/5
540/540 - 6s - loss: 0.0797 - accuracy: 0.9753 - val_loss: 0.0730 - val_accuracy: 0.9793 - 6s/epoch - 12ms/step
Epoch 4/5
540/540 - 7s - loss: 0.0597 - accuracy: 0.9817 - val_loss: 0.0723 - val_accuracy: 0.9787 - 7s/epoch - 12ms/step
Epoch 5/5
540/540 - 6s - loss: 0.0541 - accuracy: 0.9836 - val_loss: 0.0692 - val_accuracy: 0.9800 - 6s/epoch - 11ms/step

<keras.callbacks.History at 0x128161b80>
```

Entrenamiento del modelo con una nueva capa escondida y ancho de 200

Al modificar el ancho del algoritmo y la cantidad de capas implementadas. Como resultado, la precisión de validación estuvo entre 0.9643 - 0.9800. A pesar de tomar más tiempo (36.4 segundos), la precisión fue mayor que en los casos anteriores.

4. Experimenten con las funciones de activación. Intenten aplicar una transformación sigmoidal a ambas capas. La activación sigmoidal se obtiene escribiendo “sigmoid”.

```
Epoch 1/5
540/540 - 5s - loss: 1.0295 - accuracy: 0.7670 - val_loss: 0.4230 - val_accuracy: 0.8988 - 5s/epoch - 10ms/step
Epoch 2/5
540/540 - 4s - loss: 0.3334 - accuracy: 0.9124 - val_loss: 0.2667 - val_accuracy: 0.9267 - 4s/epoch - 7ms/step
Epoch 3/5
540/540 - 4s - loss: 0.2422 - accuracy: 0.9321 - val_loss: 0.2090 - val_accuracy: 0.9420 - 4s/epoch - 7ms/step
Epoch 4/5
540/540 - 3s - loss: 0.1961 - accuracy: 0.9436 - val_loss: 0.1776 - val_accuracy: 0.9505 - 3s/epoch - 6ms/step
Epoch 5/5
540/540 - 4s - loss: 0.1664 - accuracy: 0.9522 - val_loss: 0.1543 - val_accuracy: 0.9573 - 4s/epoch - 7ms/step

<keras.callbacks.History at 0x128292940>
```

Entrenamiento del modelo con una transformación sigmoidal

La transformación sigmoidal en ambas capas escondidas disminuye la precisión de validación a un rango de 0.8988 - 0.9573 y aumenta un poco su tiempo de ejecución a 23.3 segundos.

5. Continúen experimentando con las funciones de activación. Intenten aplicar un ReLu a la primera capa escondida y tanh a la segunda. La activación tanh se obtiene escribiendo “tanh”.

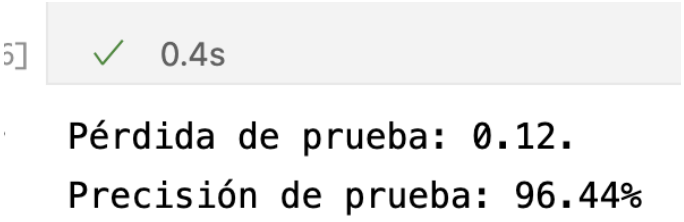
```
Epoch 1/5
540/540 - 7s - loss: 0.4013 - accuracy: 0.8912 - val_loss: 0.1909 - val_accuracy: 0.9470 - 7s/epoch - 14ms/step
Epoch 2/5
540/540 - 7s - loss: 0.1677 - accuracy: 0.9518 - val_loss: 0.1428 - val_accuracy: 0.9587 - 7s/epoch - 12ms/step
Epoch 3/5
540/540 - 7s - loss: 0.1244 - accuracy: 0.9629 - val_loss: 0.1161 - val_accuracy: 0.9643 - 7s/epoch - 13ms/step
Epoch 4/5
540/540 - 5s - loss: 0.1002 - accuracy: 0.9703 - val_loss: 0.0925 - val_accuracy: 0.9743 - 5s/epoch - 9ms/step
Epoch 5/5
540/540 - 5s - loss: 0.0828 - accuracy: 0.9754 - val_loss: 0.0857 - val_accuracy: 0.9765 - 5s/epoch - 8ms/step

<keras.callbacks.History at 0x12851ca30>
```

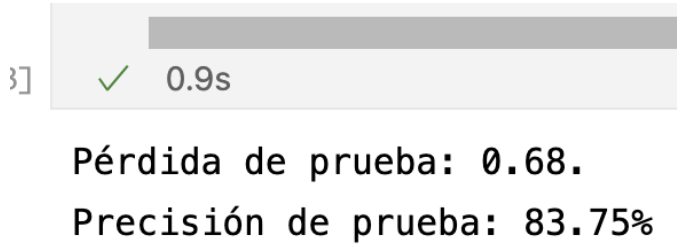
Entrenamiento del modelo con una transformación relu y tanh

El modelo de entrenamiento con las capas escondidas con transformaciones relu y tanh dieron mejores resultados que con las transformaciones sigmoidales. La precisión de valoración fue de 0.9470 - 0.9765 y su tiempo de ejecución fue de 34.2 segundos. Los porcentajes son mejores, pero sí tomó mucho más tiempo en ejecutar.

6. Ajusten el tamaño de la tanda. Prueben con un tamaño de tanda de 10,000. ¿Cómo cambia el tiempo requerido? ¿Cómo cambia la precisión?



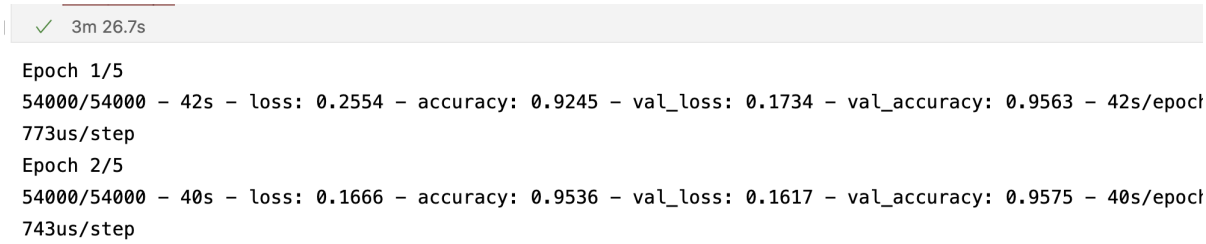
Tamaño de tanda en 100



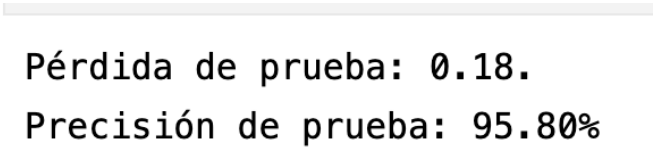
Tamaño de tanda en 10,000

7. Ajusten el tamaño de la tanda a 1. Eso corresponde al SGD. ¿Cómo cambian el tiempo y la precisión? ¿Es el resultado coherente con la teoría?

Al ajustar el tamaño de la tanda a 1 obtenemos los siguientes resultados:



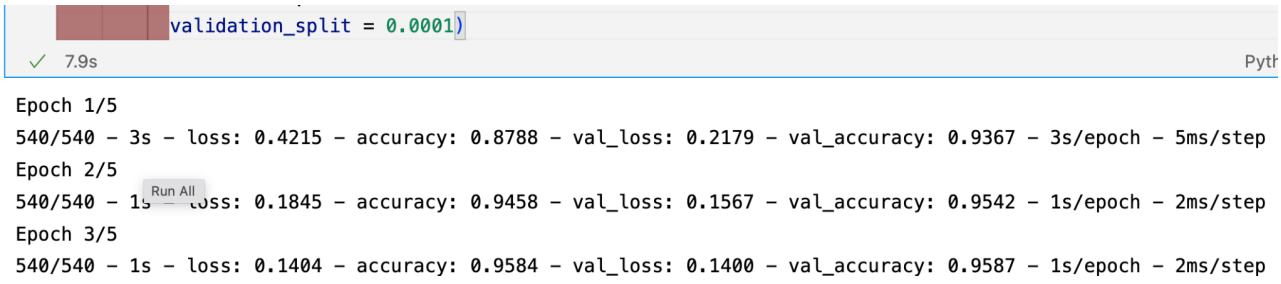
El tiempo de ejecución incrementó drásticamente ya que en vez correr en segundos, ya tardó varios minutos en ejecutar cada evento.



La precisión se mantiene constante ya que tenemos una pérdida de 0.18 y la precisión de prueba sigue manteniéndose en 95.8%. El resultado sí es coherente con la teoría ya que el tamaño del buffer y el número de muestras tienen una gran diferencia entre sí, por tanto, el barajeo no tiene toda la información y es menos uniforme comparado a un tamaño de buffer mayor a 1.

8. Ajusten la tasa de aprendizaje. Prueben con un valor de 0.0001. ¿Hace alguna diferencia?

Al ajustar el tamaño de la tanda a 0.0001 obtenemos los siguientes resultados:



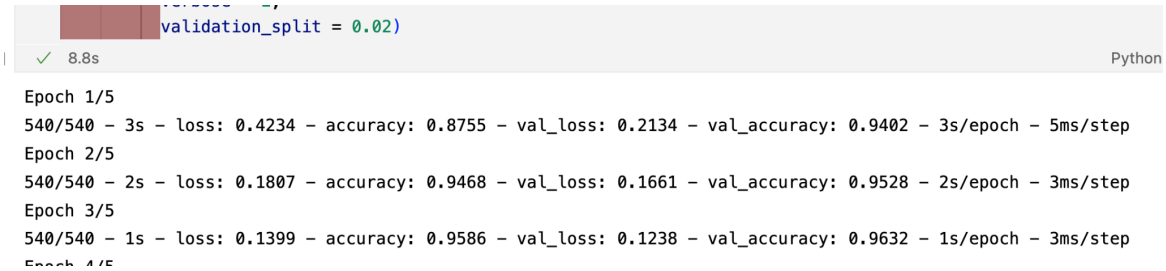
El tiempo de ejecución disminuyó drásticamente comparado con la tanda = 1.

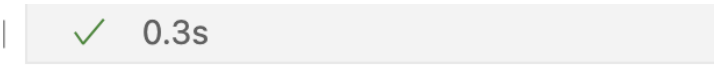
```
Pérdida de prueba: 0.11.
Precisión de prueba: 96.82%
```

La precisión de la prueba aumentó y podemos decir que la única diferencia es este dato junto con la pérdida de prueba.

9. Ajusten la tasa de aprendizaje a 0.02. ¿Hay alguna diferencia?

Al ajustar el tamaño de la tanda a 0.02 obtenemos los siguientes resultados:





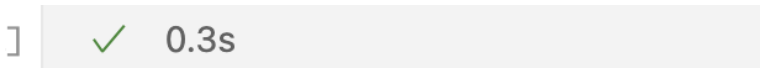

```
Pérdida de prueba: 0.10.
Precisión de prueba: 96.81%
```

La única diferencia es el aumento en la prueba de precisión.

10. Combinen todos los métodos indicados arriba e intenten llegar a una precisión de validación de 98.5% o más.

Usamos los siguientes métodos para llegar a una precisión de validación ≥ 98.5%

- No. Tanda: 100
- Tamaño entrada: 784
- Tamaño salida: 10
- Capas escondidas: 201
- No. épocas: 5
- Pasos de validación: 11
- Tasa de aprendizaje: 0.0001




```
Pérdida de prueba: 0.12.
Precisión de prueba: 98.29%
```