Universidad de Guanajuato.

División de Ciencias Económico Administrativas.

Emilio Alejandro Pérez Cerrillo.

Aprendizaje Profundo.

Andrés Espinal Jiménez.

Docente asesor.

MNIST1D: Shallow y Deep Neural Networks.

.

Guanajuato, Guanajuato.

30 de abril de 2024.

Introducción.

Como parte de nuestra asignatura Aprendizaje Profundo, se nos encargó la aplicación y análisis del comportamiento de algunas redes neuronales, a través de la base de datos conocida como MNIST1D.

MNIST1D (greydanus, 2020) es una simplificación de la base de datos MNIST, un conjunto de fotografías de trazado de números a mano. MNIST1D logra reducir la cantidad de características por vector sin dejar de lado la complejidad del reto original ya que añade ruido y otras modificaciones, pero reduciendo los vectores a solo 40 dimensiones, a comparación de las imágenes originales de 786 características.

Así mismo, este conjunto de datos es un gran elemento para evaluar modelos complejos, objetivo que buscaremos en este reporte. Mediante el uso de un framework de elección personal (Keras), implementaremos tres redes neuronales de creciente complejidad para posteriormente, y sin modificar las arquitecturas establecidas por el docente, tratar de mejorar el desempeño de dichas redes, a través de diversas técnicas adquiridas en clase. Las métricas utilizadas para evaluar el desempeño serán la exactitud y la pérdida tanto del entrenamiento como de la fase de pruebas.

Posteriormente, también se propondrá una red neuronal de arquitectura propia con la intención de mejorar el rendimiento de las anteriores.

Se espera que la revisión de este documento resulte tan interesante como su realización.

SNN

La primera red neuronal, de ahora en adelante referida como SNN, se trata de una red del tipo poco profundo o “shallow”, esta tiene 40 neuronas en su capa de entrada, 300 neuronas en su capa oculta con función de activación ReLU y 10 neuronas en su capa de salida con función de activación softmax. Adicionalmente, utiliza Categorical CrossEntropy como función de costo.

Originalmente la red utiliza el optimizador ADAM con tasa de aprendizaje 0.001 que, como puede apreciarse en la ilustración 1, presenta una brecha considerable en la exactitud y pérdida durante el entrenamiento y la validación.

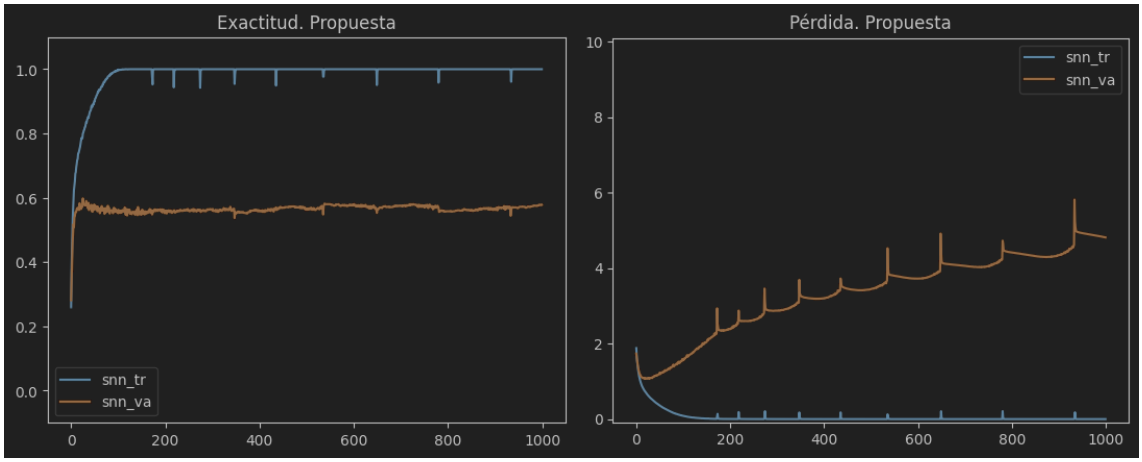

Ilustración 1. Desempeño de SNN con optimizador ADAM.

Como primera modificación, cambiaremos el optimizador por el descenso de gradiente estocástico, SGD por sus siglas en inglés, ya que este modifica más frecuentemente los pesos, además de que permite ajustar el momento con el que se moverán. Los parámetros utilizados fueron una tasa de aprendizaje de 0.05 y un momento de 0.01.

Puede apreciarse en la ilustración 2 que se disminuye la brecha en la pérdida entre el entrenamiento y la fase de validación, además de eliminar algunas crestas en la exactitud del entrenamiento.

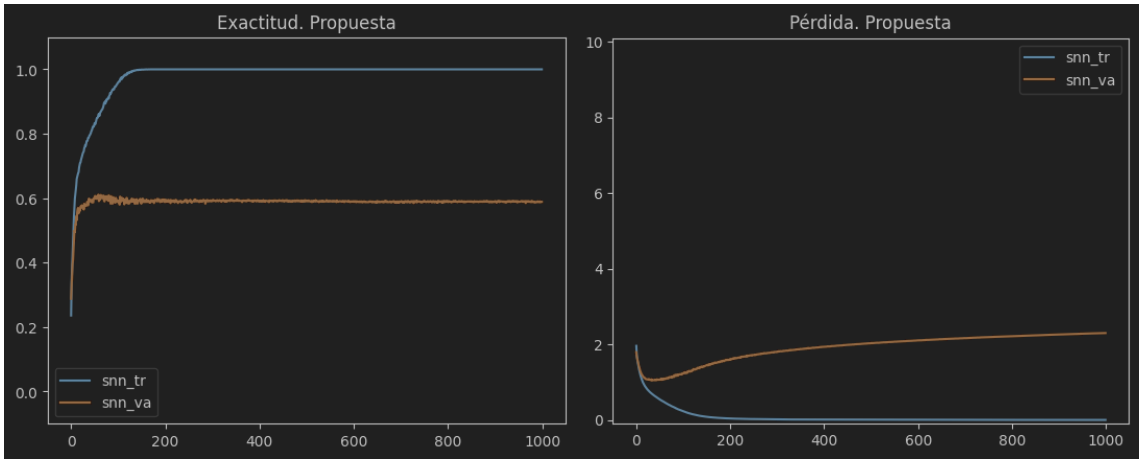

Ilustración 2. Desempeño de SNN con optimizador SGD.

Se puede detectar que existe un sobre ajuste del modelo debido a que la exactitud del conjunto de entrenamiento tiende a 1. Para atacar esto, aplicamos dos heurísticas: “early stopping” y “dropout”.

De acuerdo con Prince (2023), Early stopping, o parada temprana, es un método que detiene el proceso de entrenamiento cuando se alcanza cierta paciencia sin presentar mejoras en el desempeño. En esta ocasión la paciencia establecida fue de 200. Además de ayudar a evitar el sobre ajuste del modelo, mejora el tiempo en que este se ejecuta.

Por su parte el autor menciona que dropout o abandono es una heurística que pone ruido en las entradas que reciben las neuronas, lo que disminuye el grado de confianza que adquieren hacia los datos. El nivel de ruido utilizado fue de 0.2.

La implementación de estas técnicas mejora inmediatamente el sobreajuste durante la fase de entrenamiento y disminuye la brecha tanto en la exactitud como en la pérdida, como puede apreciarse en la ilustración 3.

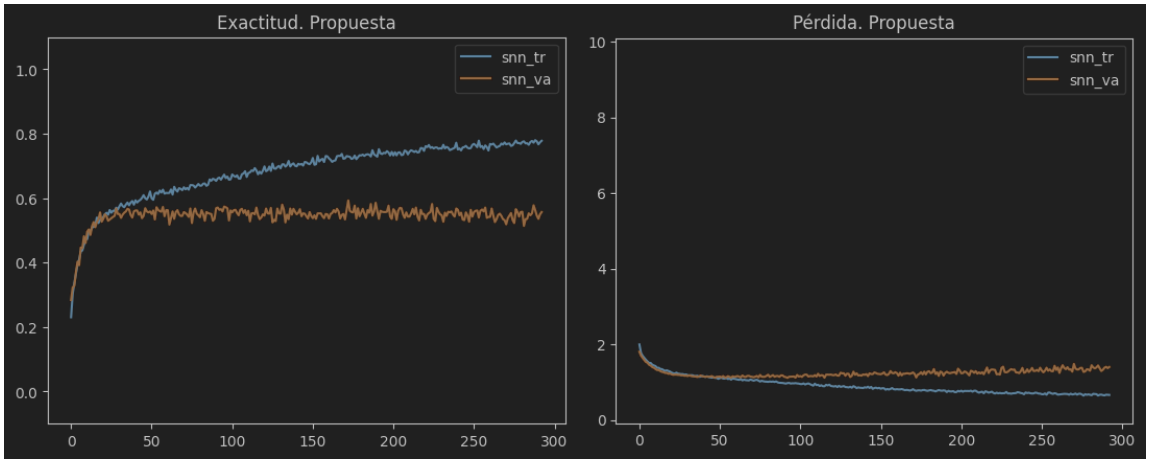

Ilustración 3. Desempeño de SNN con optimizador SGD, early stopping y dropout.

Finalmente, implementamos al modelo la técnica de regularización explícita conocida como L2, que penaliza la suma de los cuadrados de los valores de los parámetros, procura pesos pequeños, lo que produce una función de salida más suave. El parámetro utilizado con este regularizador fue 0.00001. Como puede apreciarse en la ilustración 4, este método disminuye aún más la brecha en la gráfica de la pérdida.

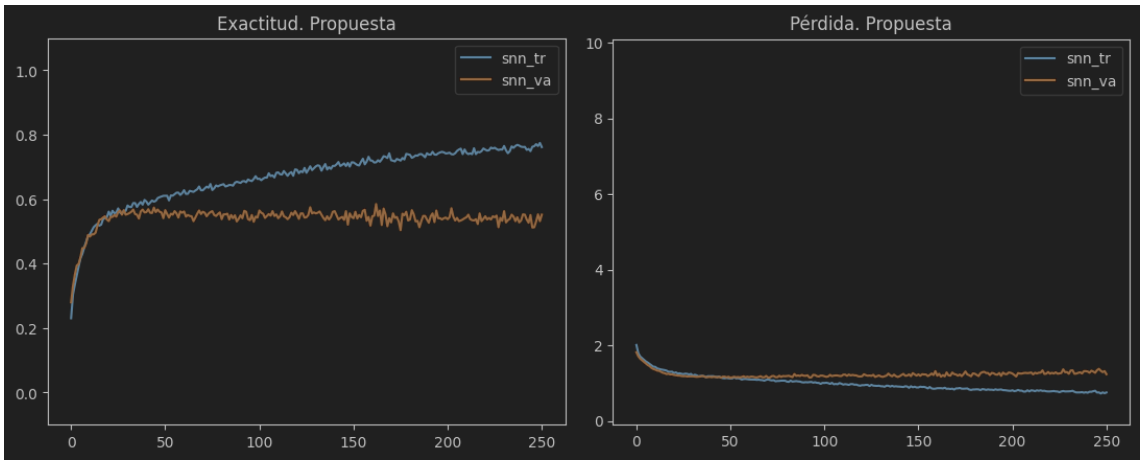

Ilustración 4. Desempeño de SNN con optimizador SGD, early stopping, dropout y regularización L2.

La elección de los cambios mencionados con anterioridad llegó del análisis de sus características vistas en clase. Dichas implementaciones al modelo mejoraron significativamente el desempeño del modelo, reduciendo en dos puntos la brecha en la exactitud y en casi cuatro en la brecha de la pérdida.

DNN1

La segunda red neuronal, de ahora en adelante referida como DNN1, se trata de una red profunda, esta tiene 40 neuronas en su capa de entrada, dos capas ocultas de 150 neuronas con función de activación ReLU cada una y 10 neuronas en su capa de salida con función de activación softmax. También utiliza Categorical CrossEntropy como función de costo.

La ilustración 5 muestra el desempeño de la red al utilizar el optimizador ADAM con tasa de aprendizaje 0.001. Se observa sobre ajuste durante el entrenamiento y brechas entre el entrenamiento y la fase de pruebas.

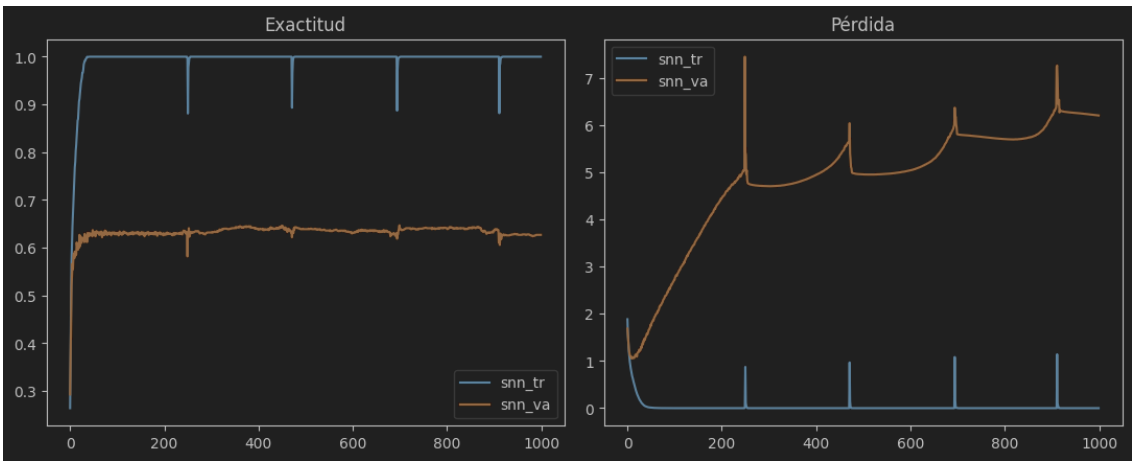

Ilustración 5. Desempeño de DNN1 con optimizador ADAM.

Se realiza el cambio al optimizador SGD con una tasa de aprendizaje de 0.05 y un momento de 0.01, por los motivos mencionados anteriormente. En la ilustración 6 puede apreciarse que este cambio suaviza las crestas en las gráficas de exactitud y pérdida, así como reducir la brecha entre la fase de entrenamiento y la de prueba en la gráfica de la pérdida.

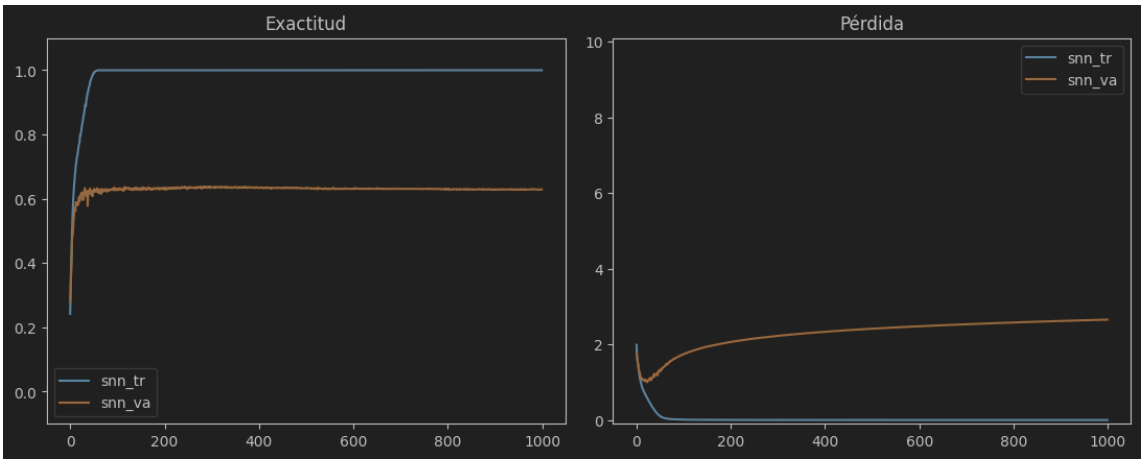

Ilustración 6. Desempeño de DNN1 con optimizador SGD.

En esta ocasión se aplicó primero la regularización L2 con parámetro 0.00001 en conjunto con la heurística early stopping, para ilustrar que por si sola, la heurística no evita por completo el sobre ajuste y en qué magnitud L2 reduce la brecha en la gráfica de la pérdida. Ambos cambios pueden apreciarse en la ilustración 7.

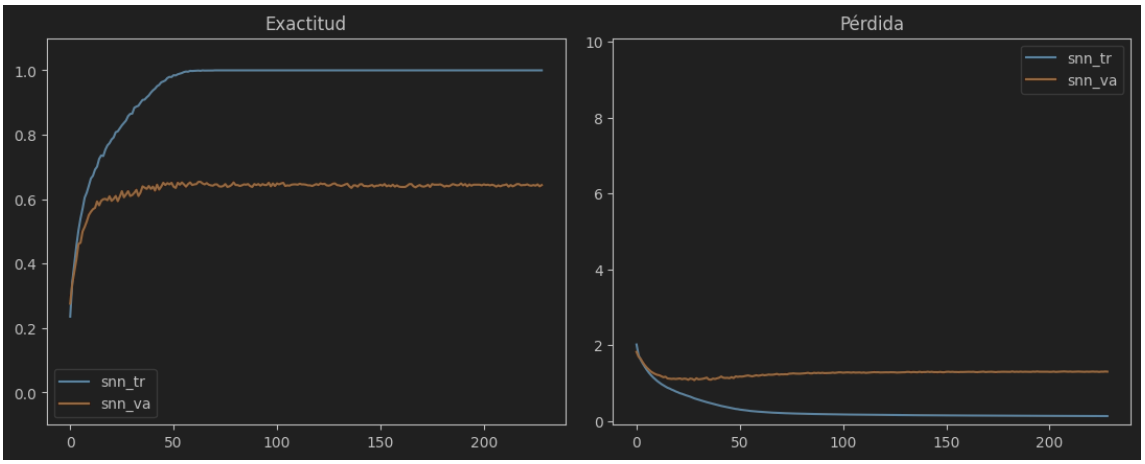

Ilustración 7. Desempeño de DNN1 con optimizador SGD, early stopping y regularizador L2.

Finalmente, aplicamos la heurística dropout para atacar directamente el sobre ajuste y reducir aún más la brecha en la gráfica de la pérdida, como puede apreciarse en la ilustración 8.

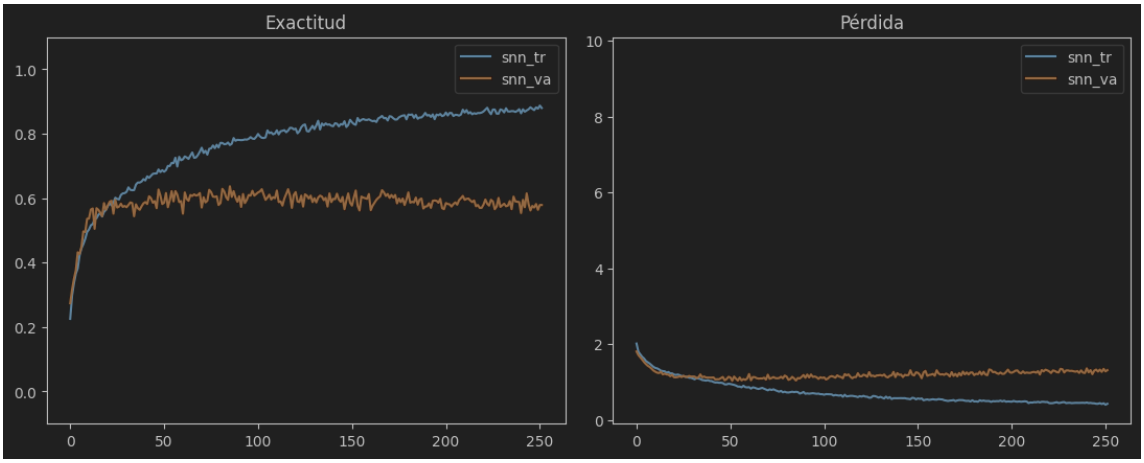

Ilustración 8. Desempeño de DNN1 con optimizador SGD, early stopping, dropout y regularizador L2.

Los cambios realizados en este modelo, a pesar de ser los mismos que para el anterior, probaron ser útiles ya que efectivamente lograron mejorar el desempeño, reduciendo las brechas en la exactitud y pérdida.

DNN2

La tercerae red neuronal, de ahora en adelante referida como DNN2, se trata de una red profunda, esta tiene 40 neuronas en su capa de entrada, tres capas ocultas de 100 neuronas con función de activación ReLU cada una y 10 neuronas en su capa de salida con función de activación softmax. También utiliza Categorical CrossEntropy como función de costo.

Comenzamos analizando el desempeño del modelo con el optimizador ADAM, y como puede apreciarse en la ilustración 9, sufre de los mismos errores

que los modelos anteriores, una brecha entre el desempeño durante la fase de entrenamiento como la de prueba, tanto en exactitud como la pérdida.

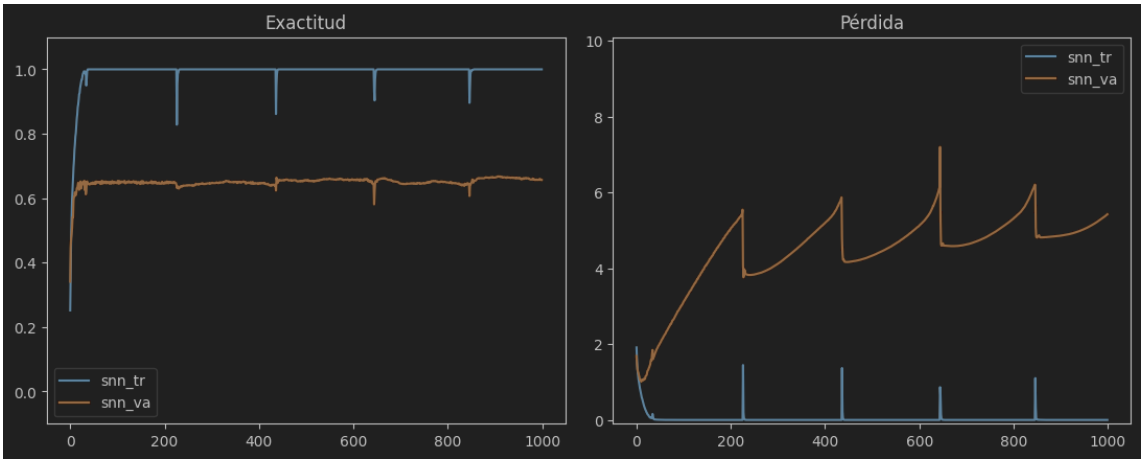

Ilustración 9. Desempeño de DNN2 con optimizador ADAM.

De manera similar a los análisis anteriores, el primer cambio que hacemos en cambiar de optimizador a SGD. Como se ve en la ilustración 10, el comportamiento del modelo mejora. Se reduce la pérdida en la fase de validación y se reducen crestas presentes en ambas gráficas.

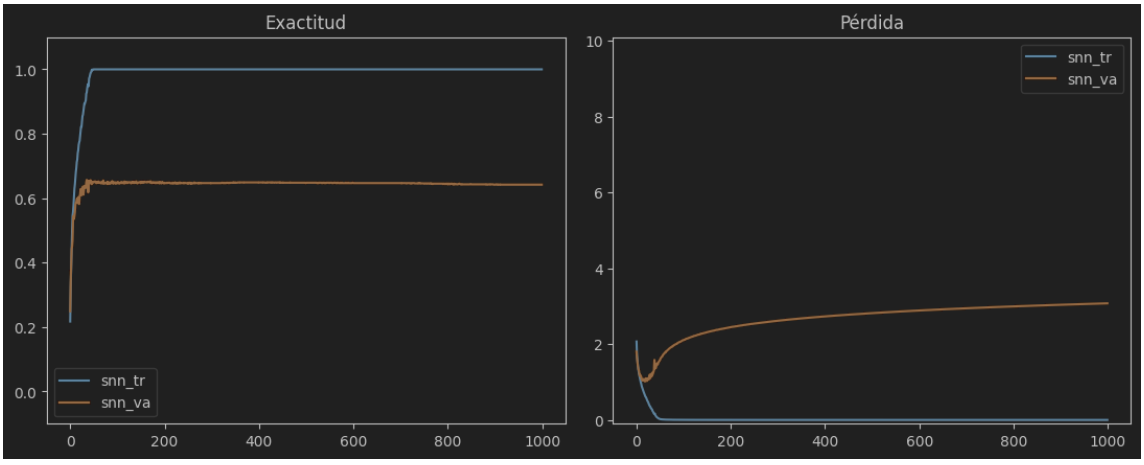

Ilustración 10. Desempeño de DNN2 con optimizador SGD.

Finalmente, implementamos en las neuronas el regularizador L2 para reducir lo más posible la brecha en la pérdida y las heurísticas early stopping y dropout para

reducir el sobre ajuste. Este cambio puede apreciarse en como la exactitud durante el entrenamiento deja de tender a 1 y la pérdida deja de tender a 0 (ilustración 11).

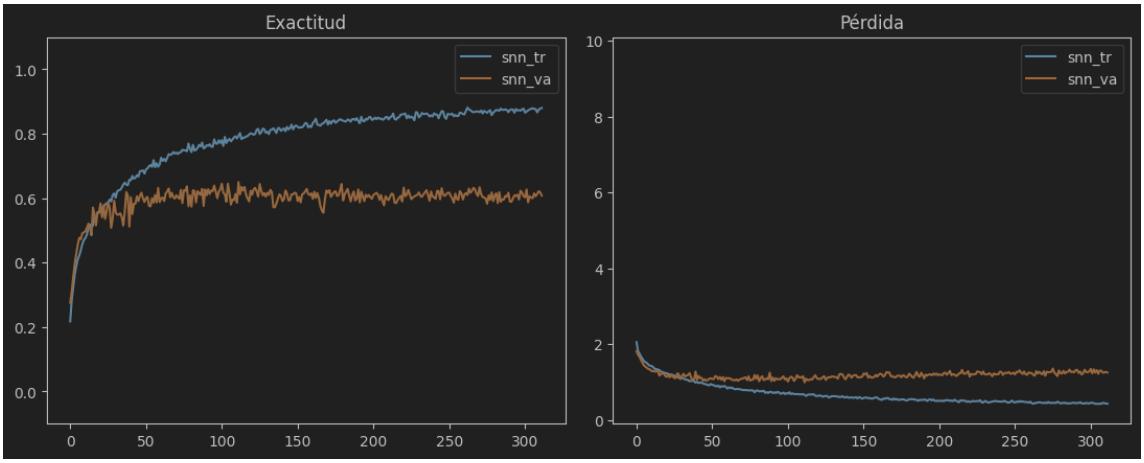

Ilustración 11. Desempeño de DNN2 con optimizador SGD, early stopping, dropout y regularizador L2.

Propuesta de red.

En esta última sección, se hará una descripción de una propuesta de red, con el objetivo de que esta sea mejor que los ejemplos anteriores.

Una de las desventajas de las redes analizadas anteriormente es la cantidad tan elevada de parámetros que necesitan. Esto se agrava especialmente después de realizar los cambios descritos en las secciones anteriores. A continuación, se presenta una tabla con la cantidad de parámetros utilizados.

Nombre del modelo.	Cantidad de parámetros original.	Cantidad de parámetros después de los cambios propuestos.
SNN	15,310 totales.	15,310 entrenables, 15,312 optimizador
		30,622 totales.
DNN1	30,310 totales.	30,310 entrenables,

		30,312 optimizador
		60,622 totales.
DNN2	25,310 totales.	25,310 entrenables, 25,312 optimizador
		50,622 totales.

Así pues, buscaremos obtener mejores comportamientos, utilizando menos parámetros. Proponemos una red neuronal profunda, de ahora en adelante referida como PROP, compuesta de 40 neuronas en su capa de entrada, tres capas ocultas de 40 neuronas con función de activación ReLU cada una y 10 neuronas en su capa de salida con función de activación softmax. Se mantiene el uso de Categorical CrossEntropy como función de costo.

El principal cambio de PROP frente a las otras redes analizadas es que utiliza menos neuronas, distribuidas en 3 capas ocultas, como se mencionó anteriormente, el objetivo es reducir los parámetros utilizados.

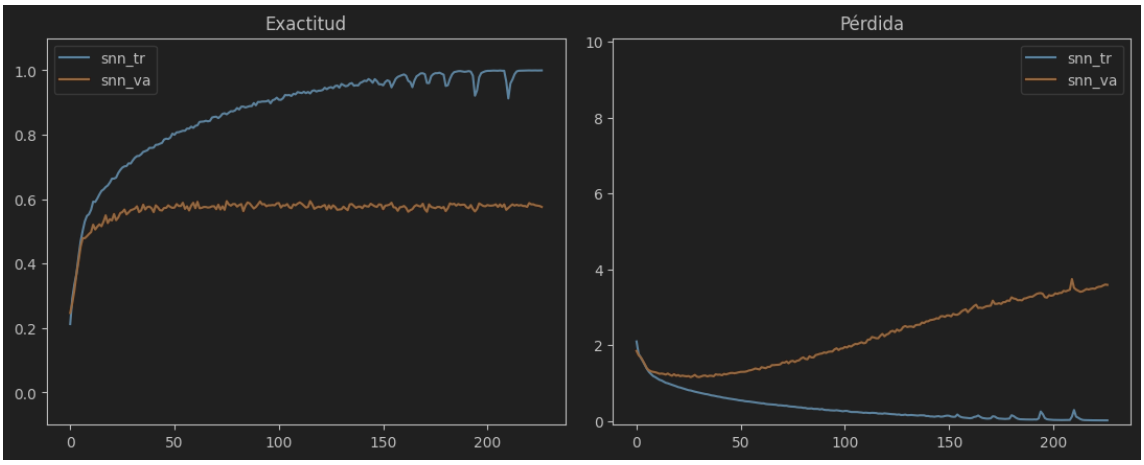

Ilustración 12. Desempeño de PROP con optimizador ADAM.

Continuando con lo aprendido en el análisis de los modelos anteriores, se hará el cambio al optimizador SGD que ha mostrado mejores resultados que ADAM, además de necesitar menos parámetros, con la desventaja de que los actualiza como conjunto más frecuentemente.

Para atacar el sobre ajuste se utilizan las heurísticas early stopping y dropout, así como el regularizador L2 en cada una de sus capas ocultas para reducir lo más posible la brecha en la pérdida. Tras aplicar todas estas técnicas, obtenemos el comportamiento descrito en la ilustración 13.

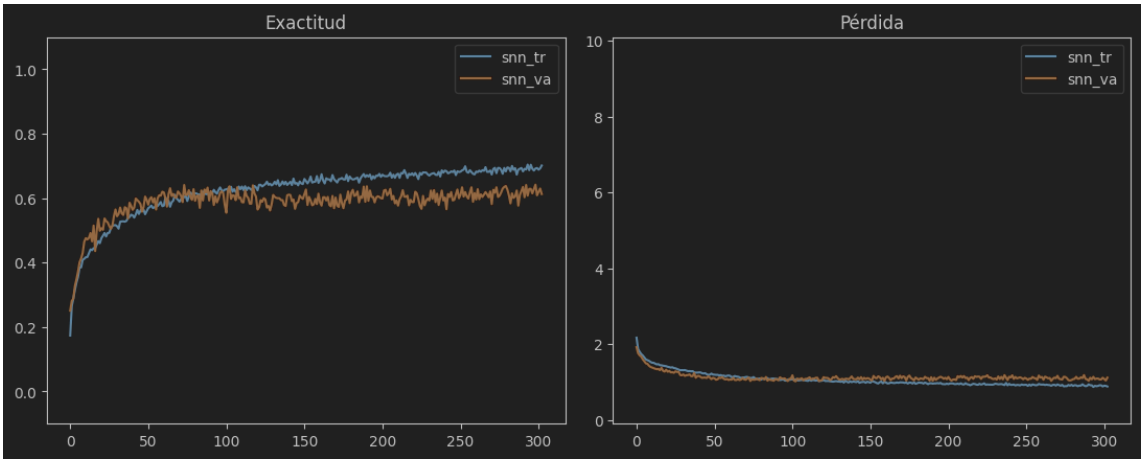

Ilustración 13. Desempeño de PROP con optimizador SGD, early stopping, dropout y regularizador L2.

Obtenemos un comportamiento similar a los últimos resultados obtenidos en los modelos anteriores, con la ventaja de que solo se utilizan 5,330 parámetros entrenables y 5,332 parámetros del optimizador, dando un total de 10,662 parámetros, cantidad que es menor incluso que la de la red poco profunda SNN.

Conclusión.

Este ejercicio de análisis de redes neuronales fue sumamente ilustrativo para identificar empíricamente los efectos de las diversas técnicas, heurísticas y arquitecturas que revisamos de manera teórica en clase durante los últimos meses.

He de mencionar que, si bien las soluciones encontradas se repiten en todos los modelos analizados, estoy seguro de que se tomaron las decisiones correctas, basadas en la razón y teoría revisada. Se espera que la justificación haya sido plasmada correctamente y no parezca que algo se dejó al azar.

Referencias.

greydanus. (2020). *mnist1d*. GitHub. https://github.com/greydanus/mnist1d

Prince, S. J. D. (2023). *Understanding Deep Learning*. MIT Press.