Redes Neuronales

Departamento de Computación

Facultad de Ciencias Exactas y Naturales

Universidad de Buenos Aires

Trabajo Práctico Nro 1
Redes feedforward multicapa

Integrantes del grupo

Santiago Iriarte	617/08	iriartesantiago@gmail.com

Indice:

1.Introducción

En el presente trabajo práctico se realizó una implementación de una red neuronal multicapa junto con distintas mejoras para la misma, y se estudió su comportamiento en dos problemas reales presentados por la materia: “Diagnóstico de cáncer de mamas” y “Eficiencia energética”. Se realizaron para los mismos distintos experimentos para poder estudiar el comportamiento de la red neuronal, tratando de encontrar los parámetros que logren una buena eficiencia de la misma.

1.1 Requerimientos

La implementación se realizó íntegramente en Python 3. A continuación se describen las librerías necesarias para su utilización:

- Numpy
- Matplotlib

1.2 Detalles de implementación

A continuación se listaran y detallarán los archivos de la implementación.

- **red_neuronal.py**: implementación de la red neuronal junto con todas sus funcionalidades específicas.
- **capa_neuronal.py**: implementación de la clase que modelo una capa de una red neuronal. Es utilizada por red_neuronal.py
- **fx_auxiliares.py**: contiene las funciones de activación y sus derivadas, utilizadas en la red neuronal.
- **parser_entrada.py**: contiene todas las funciones requeridas para procesar la entrada, para luego ser utilizada por la red neuronal.
- **ejX_prueba_Y_DescripciónPrueba.py**: donde *X* es el numero de ejercicio, *Y* el número de prueba y *DescripciónPrueba*, la descripción de la misma. Estos scripts reproducen las pruebas realizadas en este informe, y da como resultado el o los gráficos resultantes. Cada número de prueba coincide con el número de experimento de este informe, con el fin de poder replicarla. También al ejecutar cada una de estas pruebas, se genera en la carpeta “gráficos” los gráficos pertinentes a la prueba.
- **ejecutar_red.py**: script para poder ejecutar la red parametrizada.
- Carpeta **“gráficos”:** aquí se encuentran los gráficos resultantes de los experimentos.
- Carpeta **“resultados”**: aquí se encuentran los resultados (output por época) de los experimentos realizados.
- **mejores_configuraciones.txt:** aquí se encuentran valores de los parámetros de la red para los cuales logramos los mejores resultados para cada ejercicio.

1.3 Forma de uso

El script para ejecutar la red es “ejecutar_red.py”. En una ejecución deben setearse todos los parámetros que la red requiere. Aquellos que parametrizan mejoras, se setean en 0 para desactivarlas. Para todas las ejecuciones se toma del set de datos, el 80% para entrenar la red, un 10% para validar y el 10% restante para testing. El orden es el siguiente:

-Número de ejercicio: Valores posibles: ‘1’ y ‘2’
-Momentum: Número flotante, si es 0, se desactiva.
-Tamaño del batch: Número entero, si es 1, es estocástico.
-Early-Stopping: Número flotante, si es 0, se desactiva.
-Parametros adaptativos: Número entero. 1 significa activado, 0 desactivado.
-Función de activación: Número entero. 1 significa logistica, 0 tangencial.
-Distribución de los pesos: Número entero. 1 significa uniforme, 0 significa normal.
-Eta: Numero flotante
-Capas ocultas: Arreglo de enteros. Por ejemplo [10,10] son 2 capas de 10 neuronas. Sin espacios entre las comas.
-Cantidad de épocas de entrenamiento: Número entero

Ejemplo:

python ejecutar_red 2 0 1 0.2 0 1 1 0.05 [5,5] 400

- Ejecuta la red para el ejercicio 2
- Momentum desactivado
- Tamaño de batch igual a 1 (estocástico)
- Early stopping en 0.2
- Parámetros adaptativos desactivados
- Función de activación logística
- Distribución de peso uniforme
- Eta = 0.05
- Dos capas de 5 neuronas
- 400 Épocas

2.1 Preprocesamiento de los datos

En primera instancia se probaron los datos sin preprocesar para testear algunos resultados de básicos de la red, y se observaron problemas en la capa final, ya que los datos siempre estaban cercanos a los valores que saturan las funciones de activación. Al observar mejor los datos, se pudo observar que la variación de sus rangos era muy alta, y en los valores de las capas finales, los mismos también eran altos, lo que explicaba el problema.
La primera prueba que se realizó es normalizar todos los datos del ejercicio (entrada y salida), utilizando la media y la varianza de la columna. Para esto, se utilizó la librería numpy:

```
media = np.mean(vector)
desvio = np.std(vector)

For i in range(vector):
vector[i] = (vector[i] - media) / desvio
```

De esta forma se normalizo cada vector de la matriz de datos.
Esta forma de normalizar los datos reparaba el error anterior, para el ejercicio 1, pero para el ejercicio 2, el problema persistia. Es por esto que se decidió normalizar los vectores, pero entre -1 y 1, de la siguiente forma:

```
min_index = vector.argmin()
max_index = vector.argmax()
min = vector[min_index]
max = vector[max_index]


For i in range(vector):
vector[i] = 2*(vector[i]-min)/((max-min))-1
```

De esta forma, los valores quedan entre -1 y 1, lo cual solucionó el problema para el ejercicio 2 y también funcionó correctamente para el ejercicio 1. Por lo que fue elegida como la forma de preprocesar los datos.

2.2 Ejercicio 1

2.2.1. Variación de la función de activación

En este experimento se probaron las funciones tangencial y logística para observar el comportamiento de la red neuronal utilizando cada una. Se observó el error por época en la fase de entrenamiento.
Se corrieron 8 veces cada prueba, esto es, 8 entrenamientos para cada tipo de función de activación y se decidió, de cada uno de esos ocho, quedarse con el menor error para cada función y compararlos. ***Este mismo proceso fue utilizado para todos los experimentos del presente informe.***

Arquitectura de la red:

-Función de activación: **parámetro a estudiar**

-Tamaño de set entrenamiento: 80%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas ocultas de 10 neuronas

-ETA: 0.05

-Momentum: desactivado

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados:

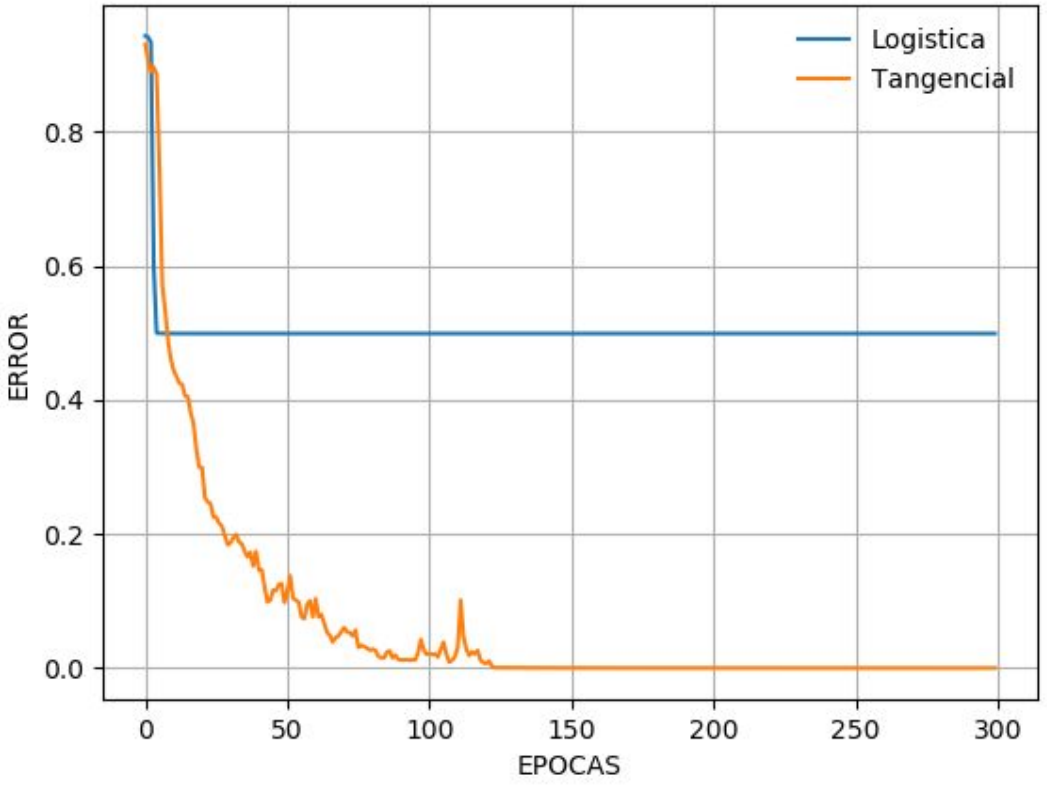

Gráfico 1: Función de logística vs. función tangencial

Observaciones del experimento:

Se puede ver que utilizando la función logística el error es muy alto, no baja de 0.5. Por otro lado con la función tangencial el resultado es óptimo.
A partir de esto, se decide utilizar la función tangencial como función de activación para el resto de los experimentos de este ejercicio.

2.2.2. Variación de número de capas ocultas y número de neuronas

En este experimento se verificó el comportamiento de la red neuronal variando la cantidad de capas ocultas y la cantidad de neuronas de las misma. Se probó con 1, 2, 3 y 5 capas ocultas, con 10 neuronas cada una. Luego una vez obtenida la que cantidad de capas que mejor resultado obtuvo, se varió el número de neuronas en 2, 5, 10 y 20. Se verificaron los errores por época de entrenamiento y validación.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: variable a estudiar

-ETA: 0.05

-Momentum: Desactivado

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - Variación número de capas ocultas y neuronas:

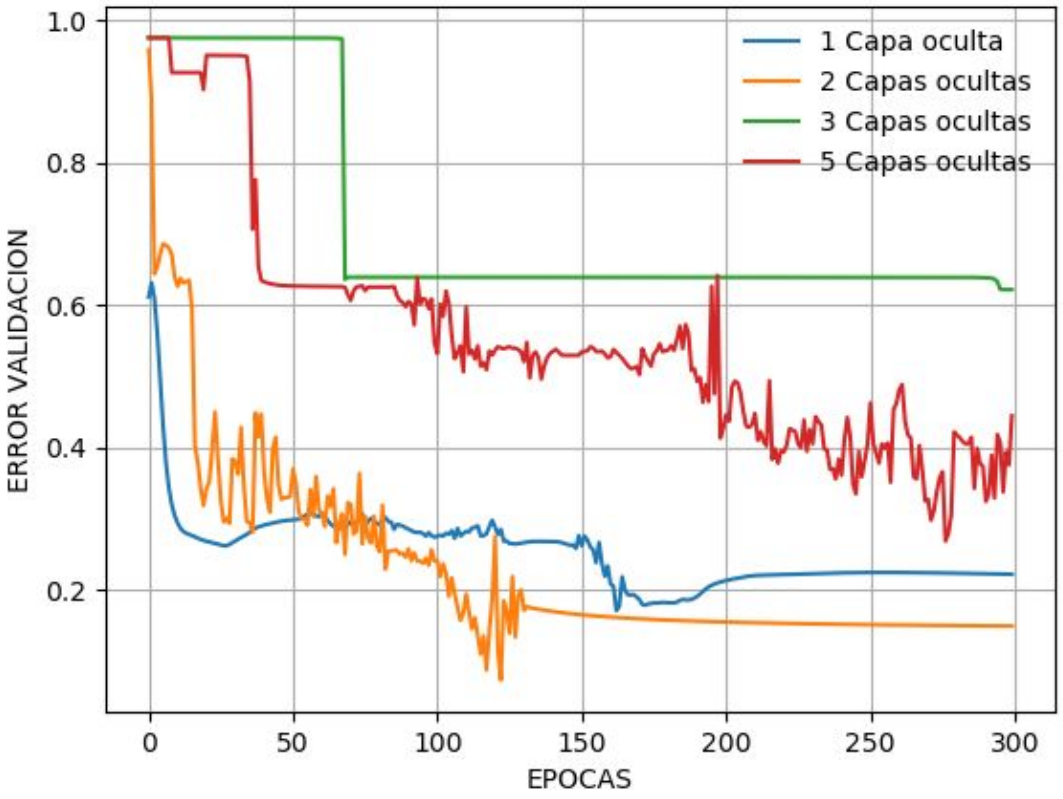

Gráfico 2: Ejercicio 1 - Variación capas ocultas - Error entrenamiento

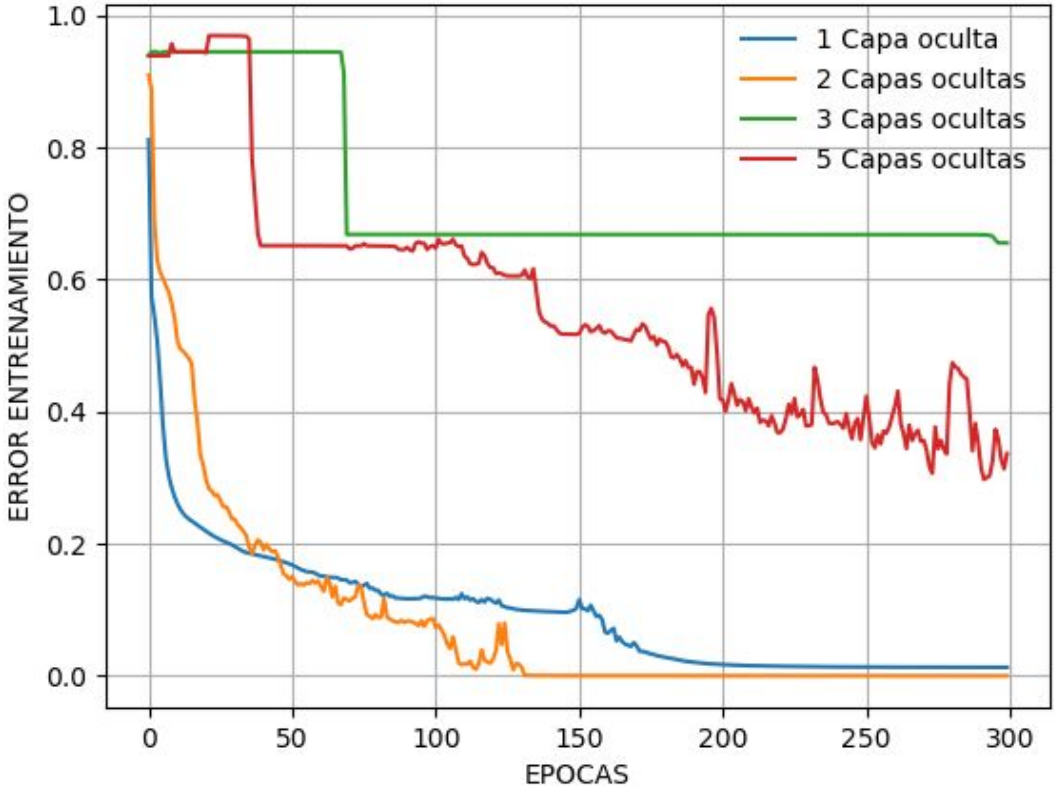

Gráfico 3: Ejercicio 1 - Variación capas ocultas - Error validación

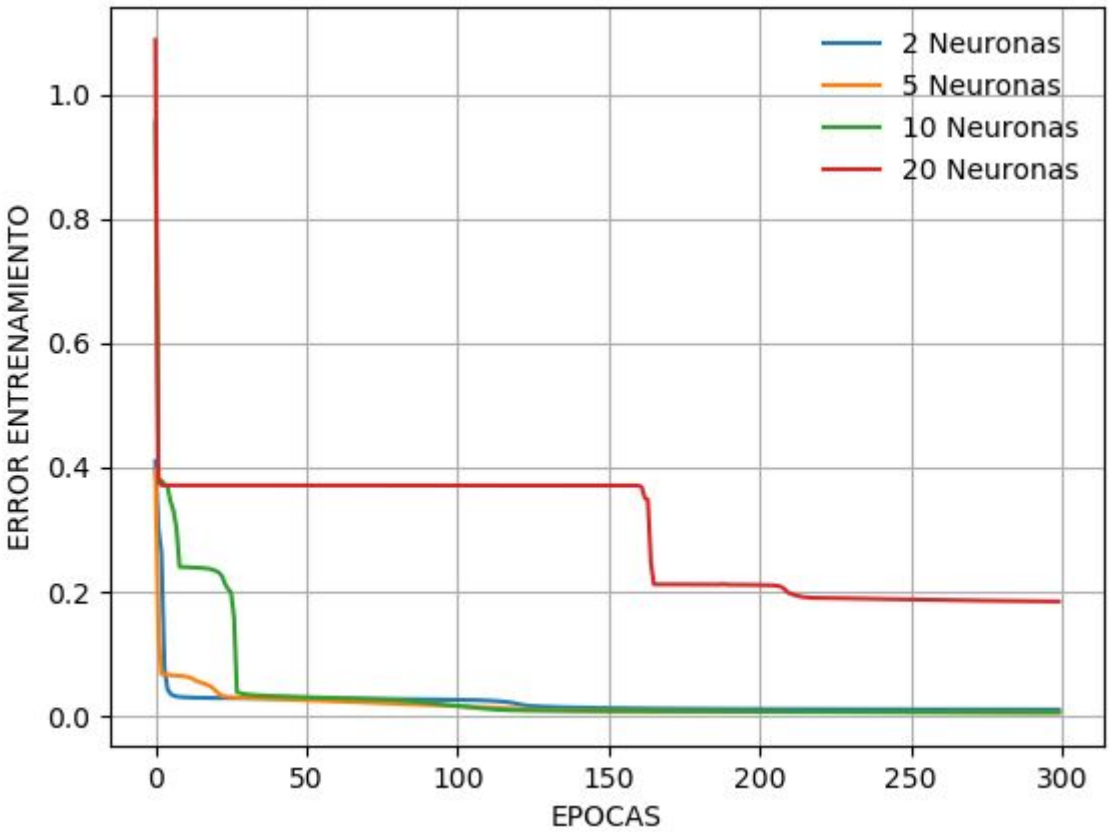

Gráfico 4: Ejercicio 1 - Variación cantidad de neuronas - Error entrenamiento

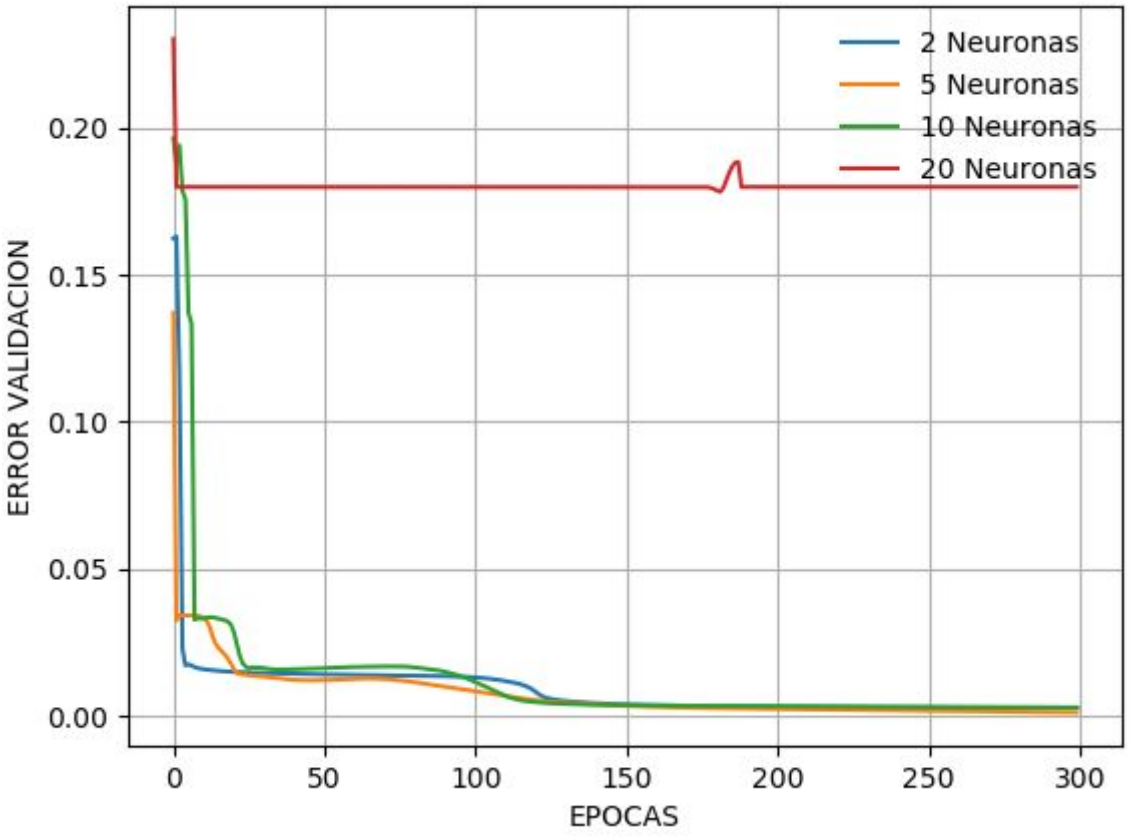

Gráfico 5: Ejercicio 1 - Variación cantidad de neuronas - Error entrenamiento

Observaciones del experimento:

Los gráficos muestran claramente que los mejores resultados tanto para el entrenamiento como para la validación se encuentran con 1 y 2 capas, para más capas los resultados no son alentadores. Esto tiene sentido, ya que teniendo en cuenta el tamaño de la entrada, más de dos capas de 10 neuronas, es una arquitectura demasiado grande para el problema que se quiere resolver.
A partir de esto, se decidió variar la cantidad de neuronas tomando 2 capas, ya que los resultados obtenidos fueron los mejores.
Con respecto a la cantidad de neuronas, los resultados no parecen ser distintos para 2, 5 y 10 neuronas. Se puede observar que el entrenamiento para 10 neuronas, toma más épocas en converger. Para 20 neuronas, los resultados son malos, y el problema posiblemente sea el mismo que para el exceso de capas.
Se tomará para las siguientes pruebas la arquitectura de 2 capas de 10 neuronas cada una.

2.2.3. Variación en la forma de iniciar los pesos

Se observará el comportamiento de la red iniciando los pesos de las neuronas con dos distribuciones distintas, la distribución normal y la distribución uniforme.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas de 10 neuronas.

-ETA: 0.05

-Momentum: Desactivado

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: **parámetro a estudiar**

Resultados - Forma de iniciar los pesos:

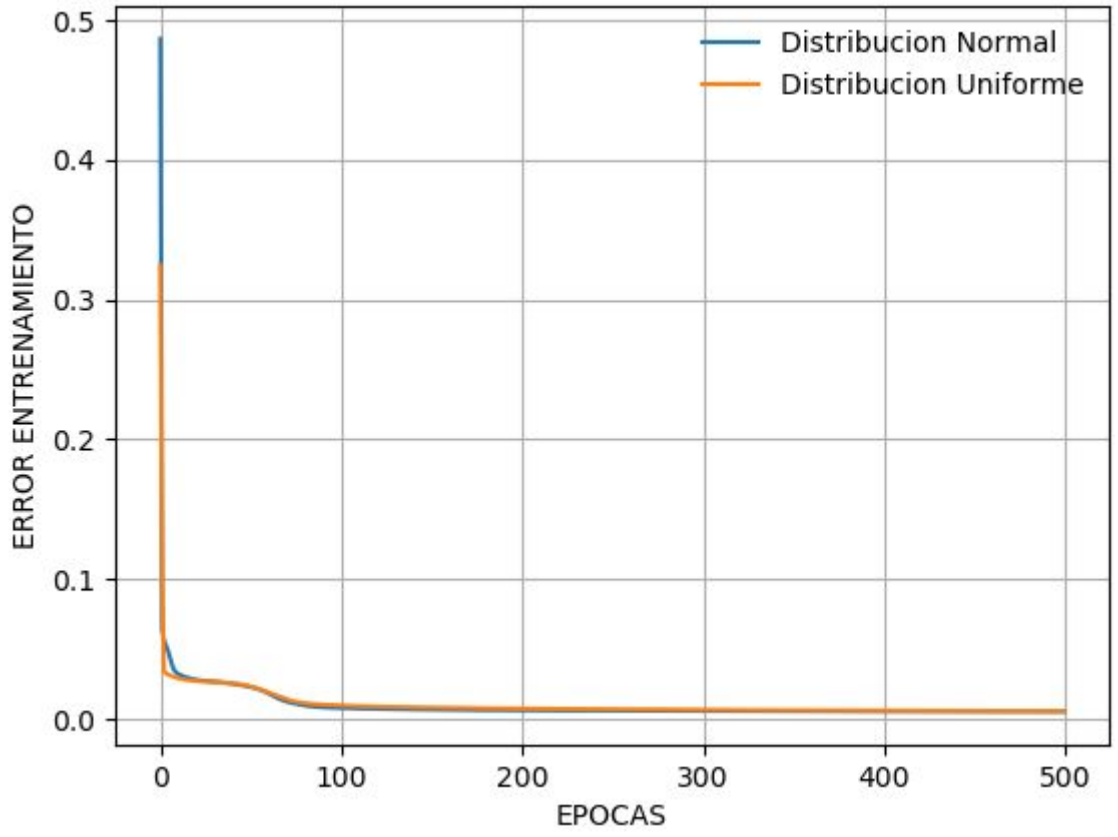

Gráfico 6: Ejercicio 1 - Variación distribución de pesos - Error entrenamiento

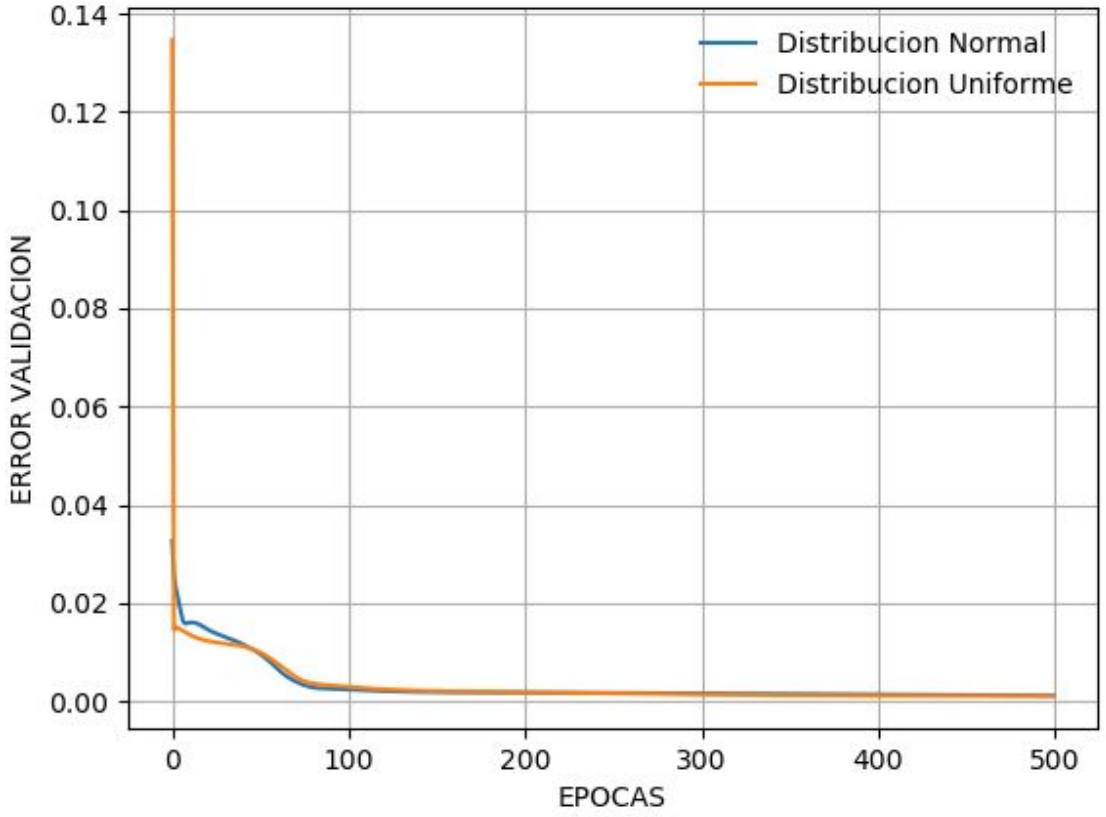

Gráfico 7: Ejercicio 1 - Variación distribución de pesos - Error validación

Observaciones del experimento:

Los resultados no muestran diferencias importantes entre los dos experimentos, ambos obtienen buenos resultados. ***Se utilizará para los experimentos que siguen de este ejercicio la distribución normal.***

2.2.4. Performance con Momentum

Se observará el comportamiento de la red con y sin momentum y su error en la etapa de entrenamiento y en la de validación para cada época. Se variara el parámetro con los siguientes valores: 0.1, 0.3, 0.5, 0.7 y 0.9.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas de 10 neuronas.

-ETA: 0.05

-Momentum: **parametro a estudiar**

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - Momentum:

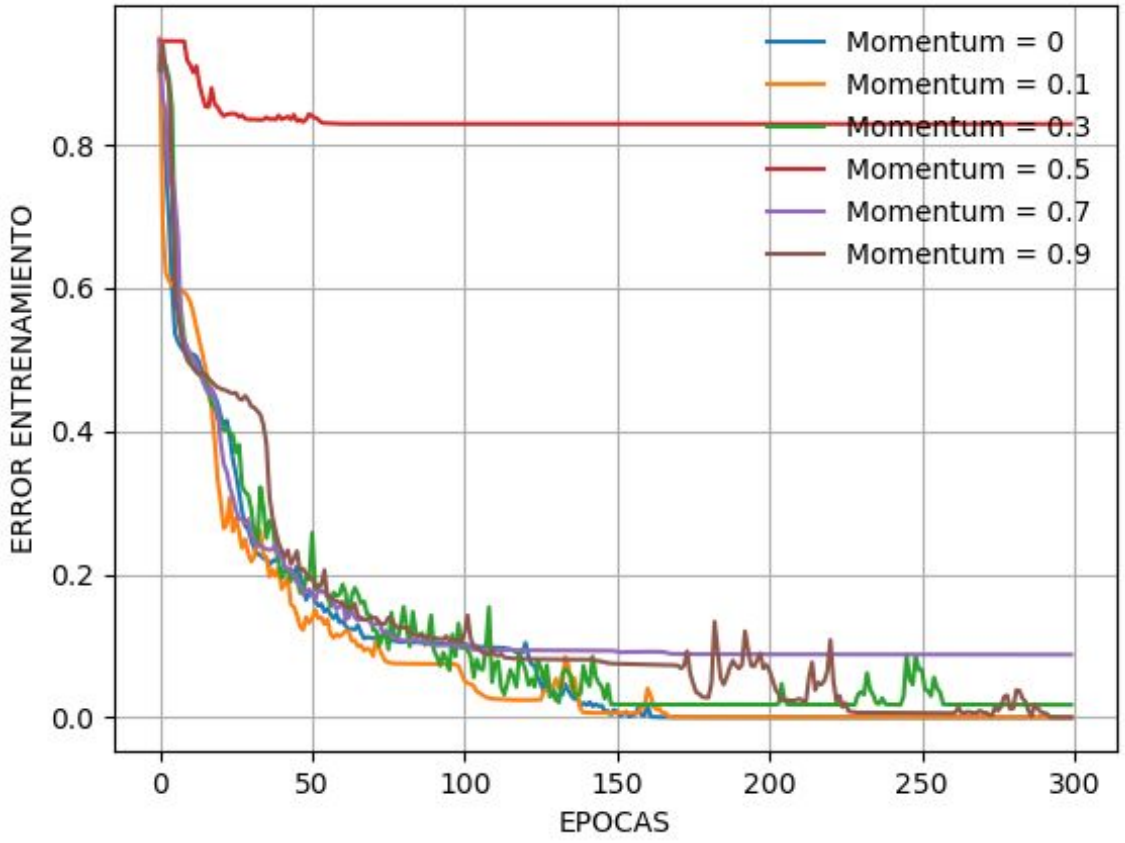

Gráfico 7: Ejercicio 1 - Variación parámetro Momentum - Error entrenamiento

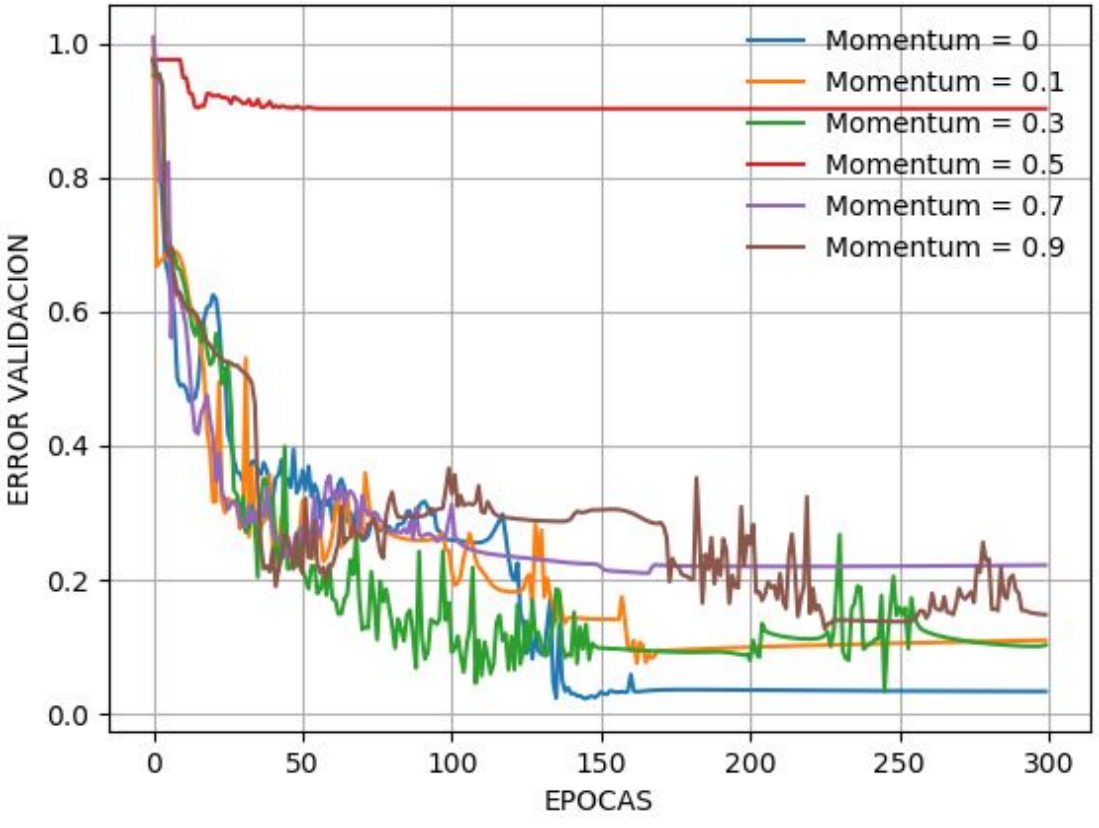

Gráfico 8: Ejercicio 1 - Variación parámetro Momentum - Error validación

Observaciones del experimento:

Los resultados muestran que en la etapa de entrenamiento, con momentum desactivado y con valores de 0.1, 0.3, 0.9, se obtienen buenos resultados. Los demás valores están lejos de ser buenos. Por otro lado en la validación, claramente se observa que momentum desactivado arroja el mejor resultado. El parámetro en 0.5 obtiene el peor error en ambos casos. Para los demás valores, los errores son muy cercanos, sus convergencias oscilan, pero lejos del mejor resultado.

2.2.5. Performance con Early-Stopping

Se observará el comportamiento de la red con y sin Early-Stopping y su error por época de entrenamiento y de validación. Se variará el parámetro con los siguientes valores: 0.1, 0.3, 0.5, 0.7 y 0.9.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas de 10 neuronas.

-ETA: 0.05

-Momentum: Desactivado

-Tamaño de batch: 1

-EarlyStopping: parametro a estudiar

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - Early-Stopping:

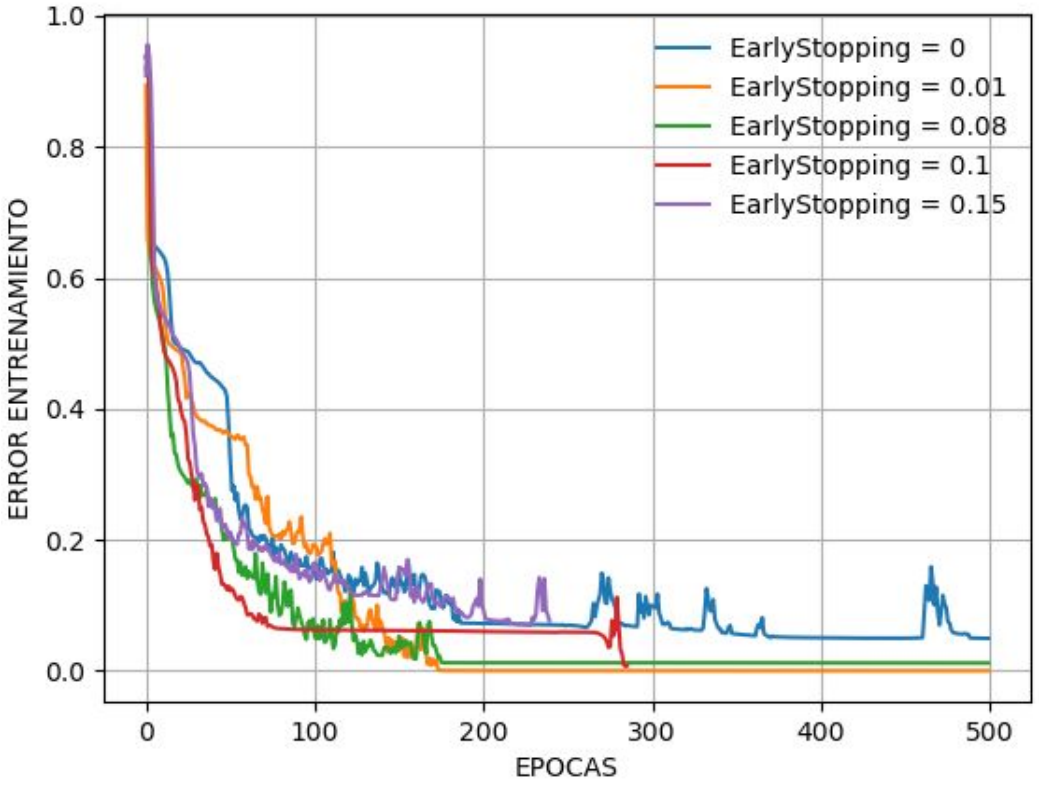

Gráfico 9: Ejercicio 1 - Variación parámetro Early-stopping - Error entrenamiento

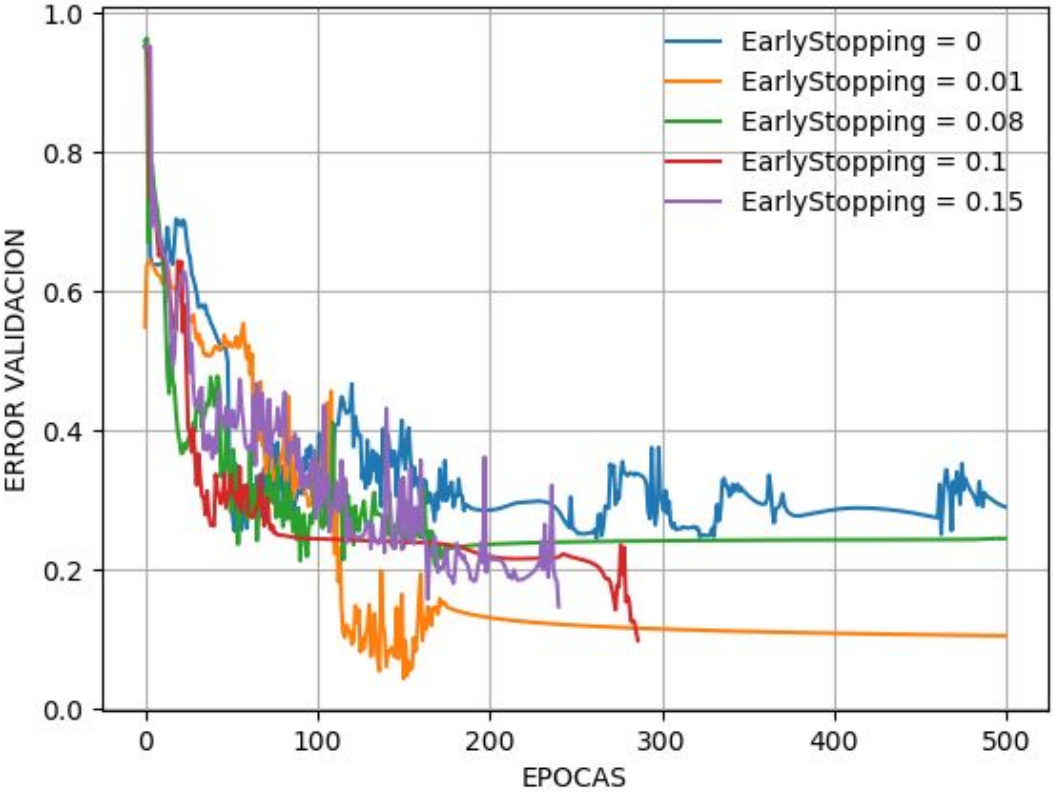

Gráfico 10: Ejercicio 1 - Variación parámetro Early-stopping - Error validación

Performance en etapa de testing para Early-Stopping:

Valor Early-Stopping	Performance de Testing
Desactivado	85%
0.01	90%
0.08	85%
0.1	92%
0.15	85%

Observaciones del experimento:

Se observa que el mejor resultado se encuentra con el valor en 0.1, donde se puede observar en los gráficos el corte por early-stopping y un buen resultado en la etapa de testing. También podemos ver que con el valor de 0.15 también existió el corte en el entrenamiento, pero en la etapa de testing no obtuvo tan buen resultado.
Para los demás valores, el corte de early-stopping no sucedió tras varias pruebas, por lo que no hay nada observable, obteniendo de todas formas buenos resultados.

2.2.6. Performance con Parametro Adaptativos

Se observo el comportamiento de la red con y sin parámetros adaptativos. La configuración de este feature, es la siguiente:

-Si el error de la época anterior es menor al actual, entonces sumó al eta el 1% de su valor.

-Si el error de la época anterior es mayor al actual, entonces resto al eta la mitad de su valor.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas de 10 neuronas.

-ETA: 0.05

-Momentum: Desactivado

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: parámetro a estudiar

-Distribución para los pesos: Normal

Resultados - Parámetros Adaptativos:

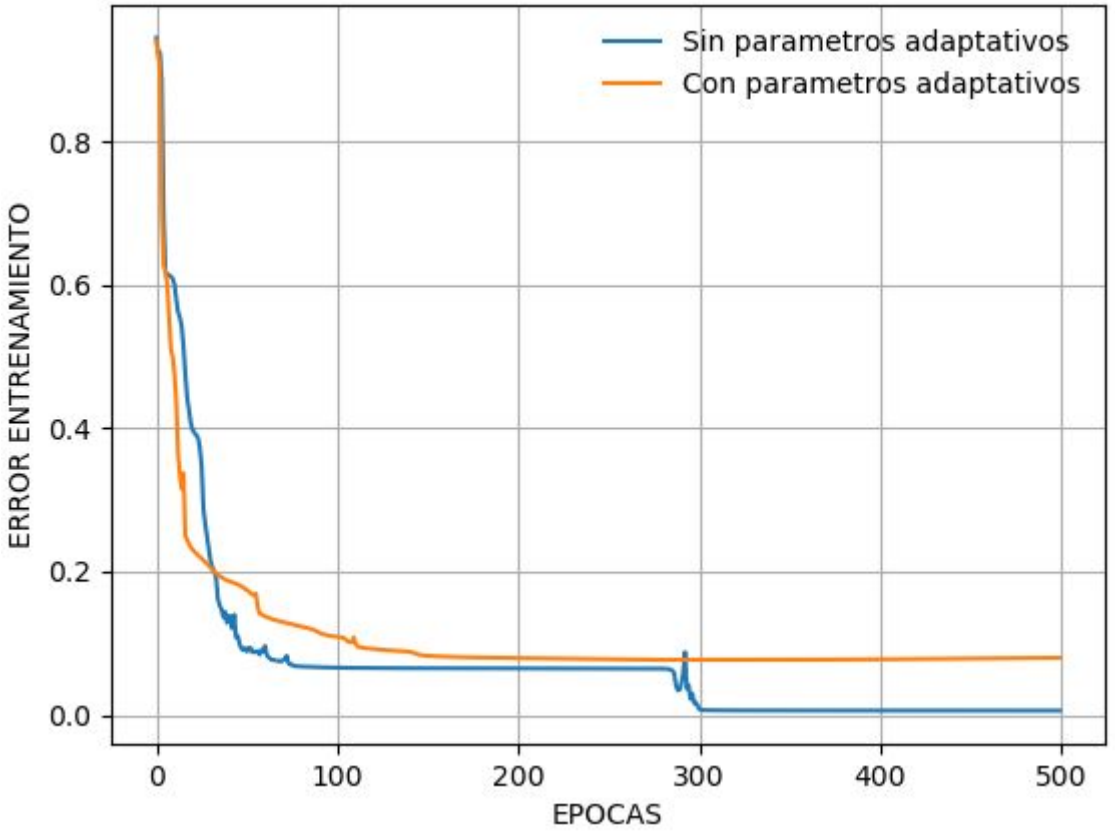

Gráfico 11: Ejercicio 1 - Variación Parámetros adaptativos - Error Entrenamiento

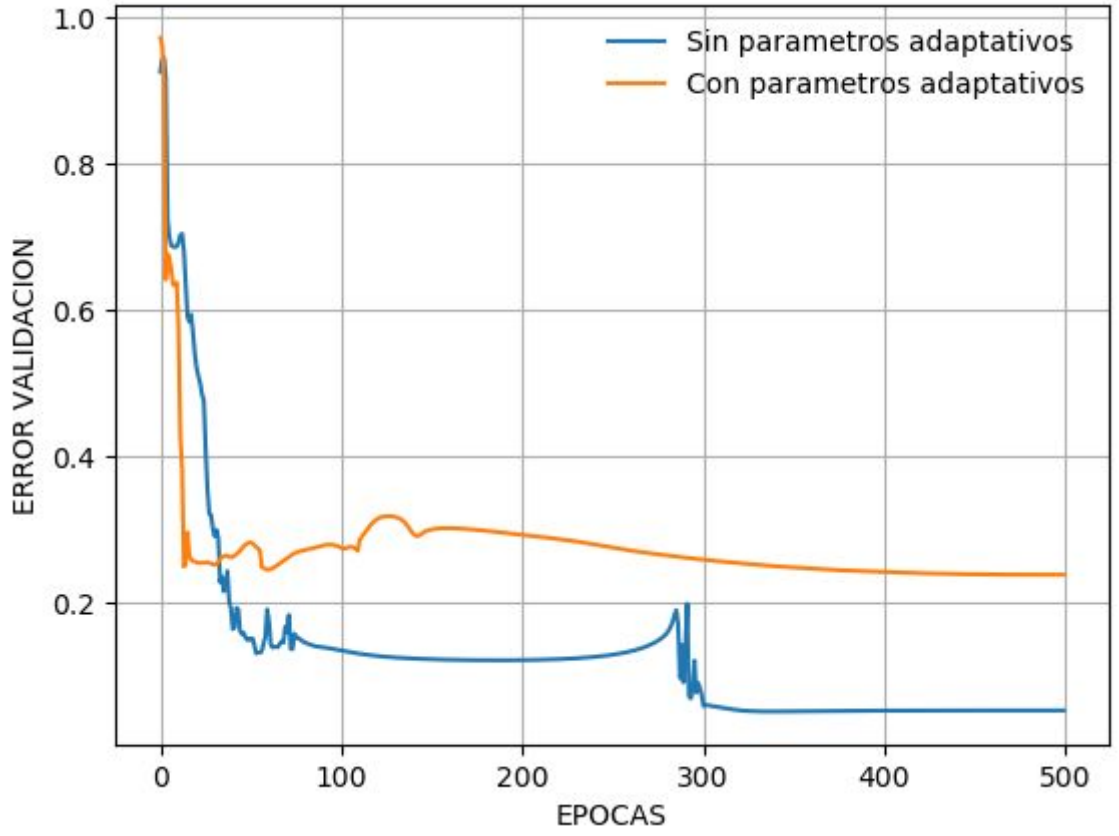

Gráfico 12: Ejercicio 1 - Variación Parámetros adaptativos - Error validación

Performance en etapa de testing para Parámetros Adaptativos:

Valor Early-Stopping	Performance de Testing
Desactivado	92%
Activado	85%

Observaciones del experimento:

Se observa que con este feature activado la red alcanza peores resultados que si el mismo está desactivado, tanto en la etapa de entrenamiento, validación y testing. De todas formas, para llegar a una conclusión más exacta respecto al funcionamiento de esta mejora, se deberían probar más variaciones en sus valores de crecimiento y decrecimiento del eta.

2.2.7. Performance variando factor de aprendizaje y parámetro del momentum

En este experimento se variarán conjuntamente los valores de eta y momentum, buscando combinaciones que logren buenos resultados. Para cada uno de los valores de eta 0.005, 0.05, 0.1, 0.3, se utilizaran valores de momentum 0.1, 0.3, 0.5, 0.7, 0.9, y también momentum desactivado. Se evaluará el error de entrenamiento y validación para cada época.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas de 10 neuronas.

-ETA: **parametro a estudiar**

-Momentum: **parametro a estudiar**

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - ETA / Momentum:

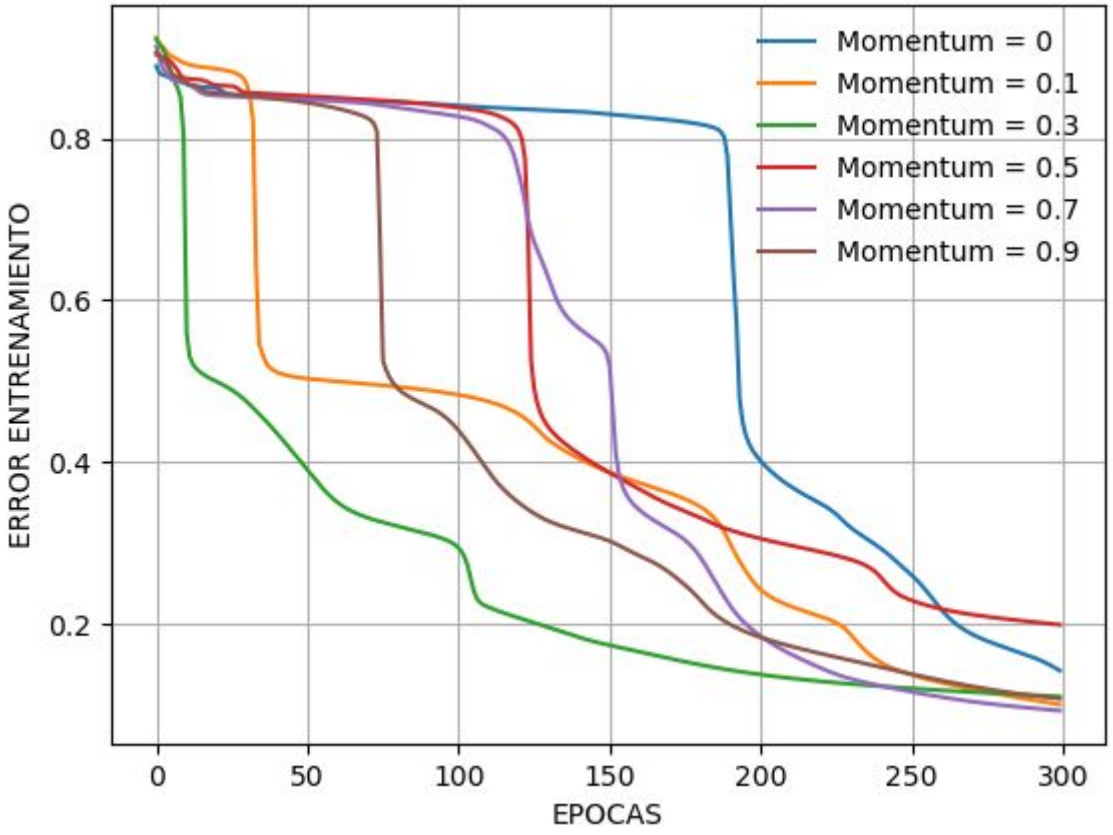

Gráfico 13: Ejercicio 1 - Variación ETA/Momentum - Eta = 0,005 - Error entrenamiento

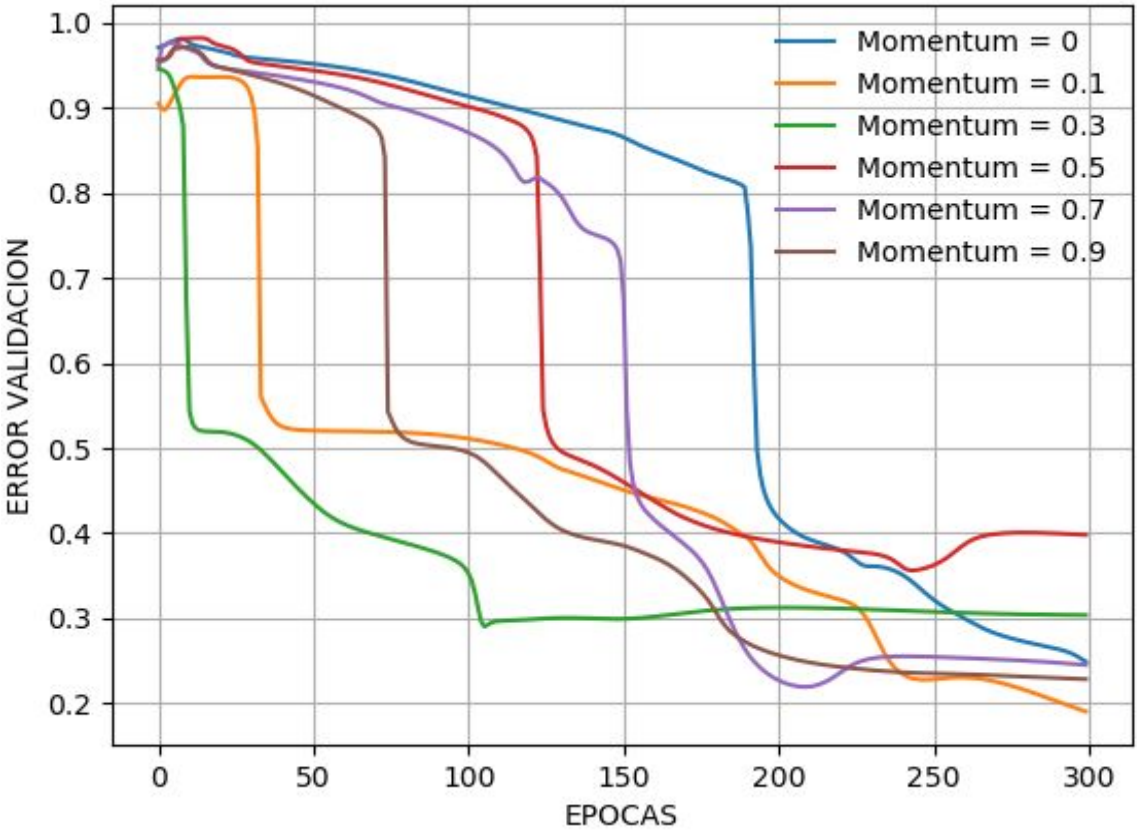

Gráfico 14: Ejercicio 1 - Variación ETA/Momentum - Eta = 0,005 - Error validación

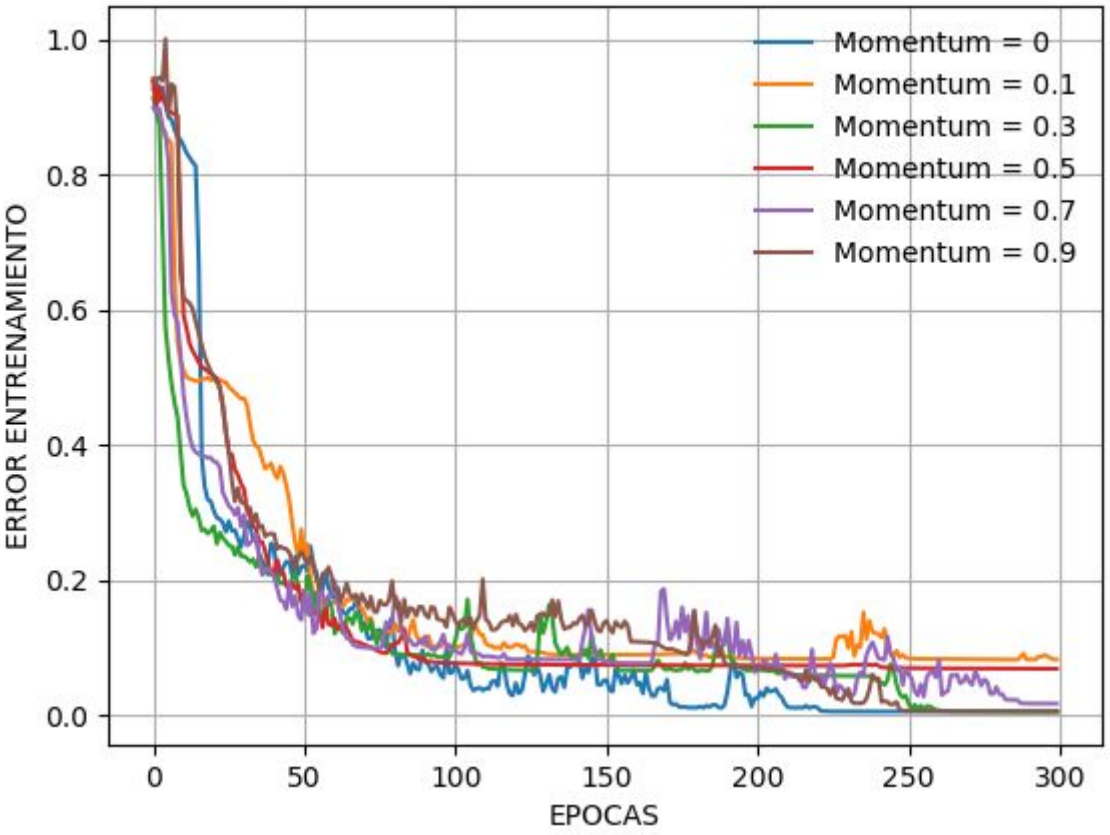

Gráfico 15: Ejercicio 1 - Variación ETA/Momentum - Eta = 0,05 - Error validación

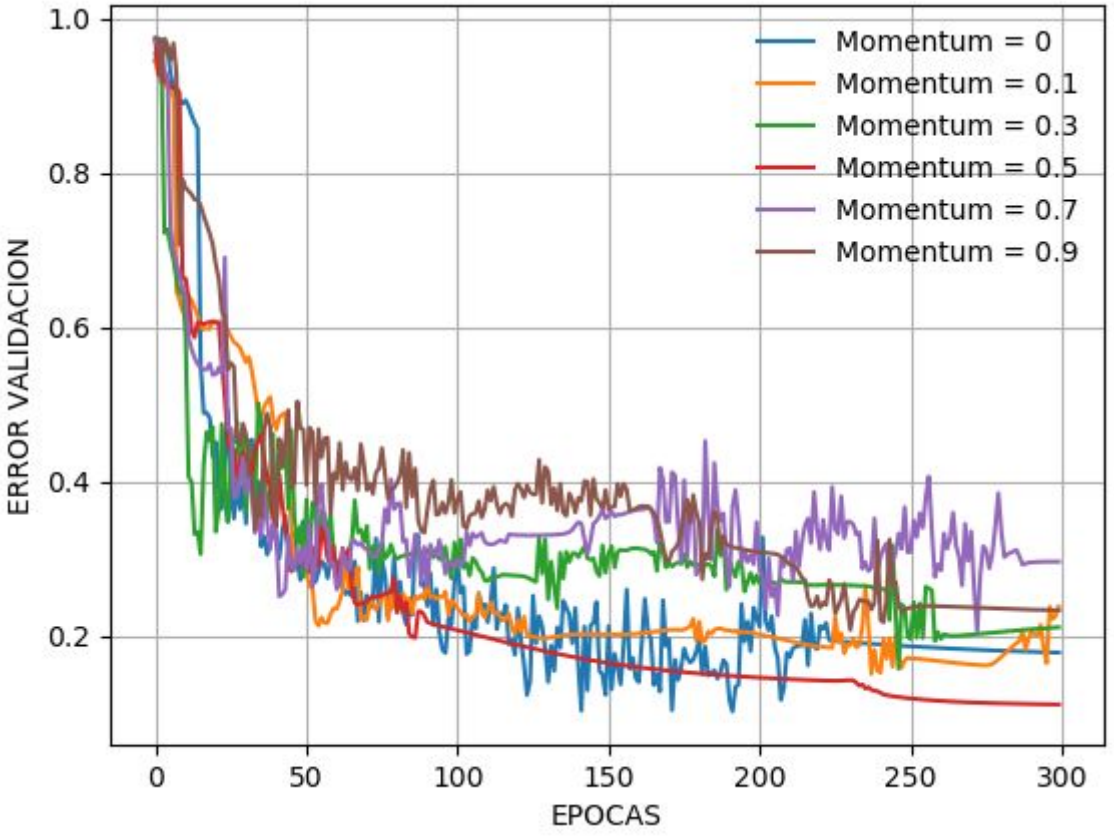

Gráfico 16: Ejercicio 1 - Variación ETA/Momentum - Eta = 0,05 - Error validación

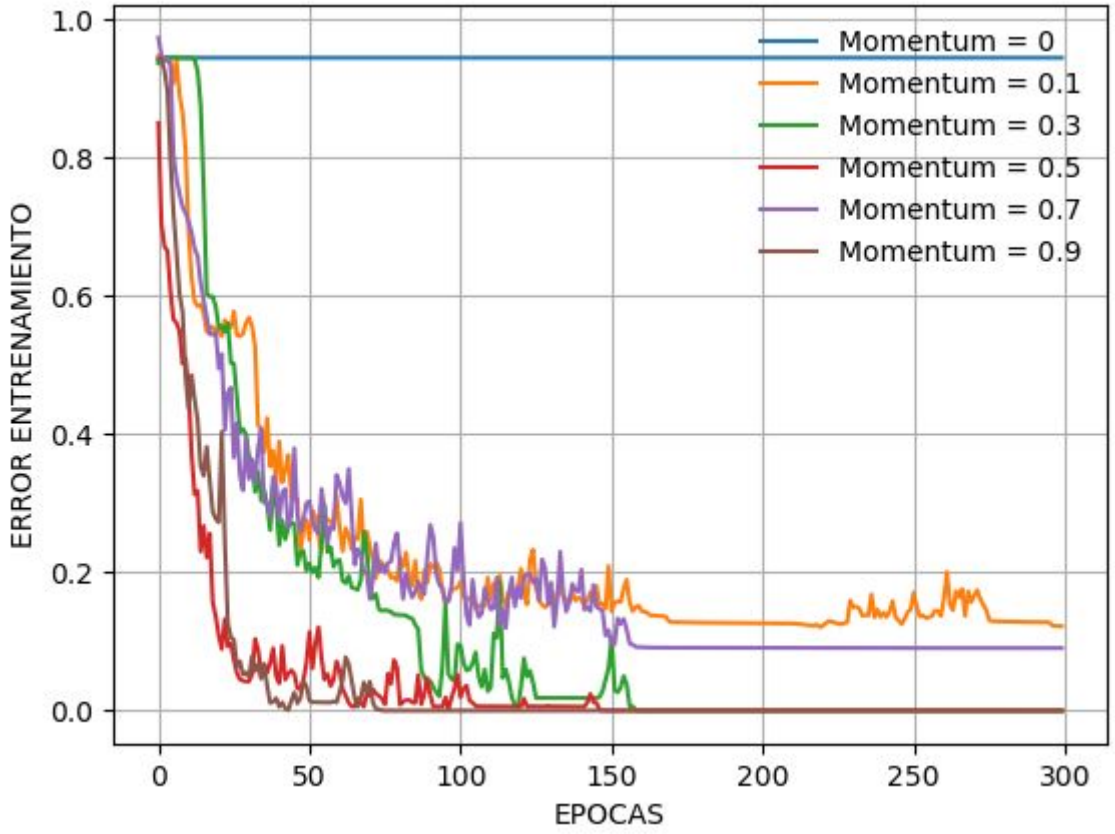

Gráfico 17: Ejercicio 1 - Variación ETA/Momentum - Eta = 0,1 - Error entrenamiento

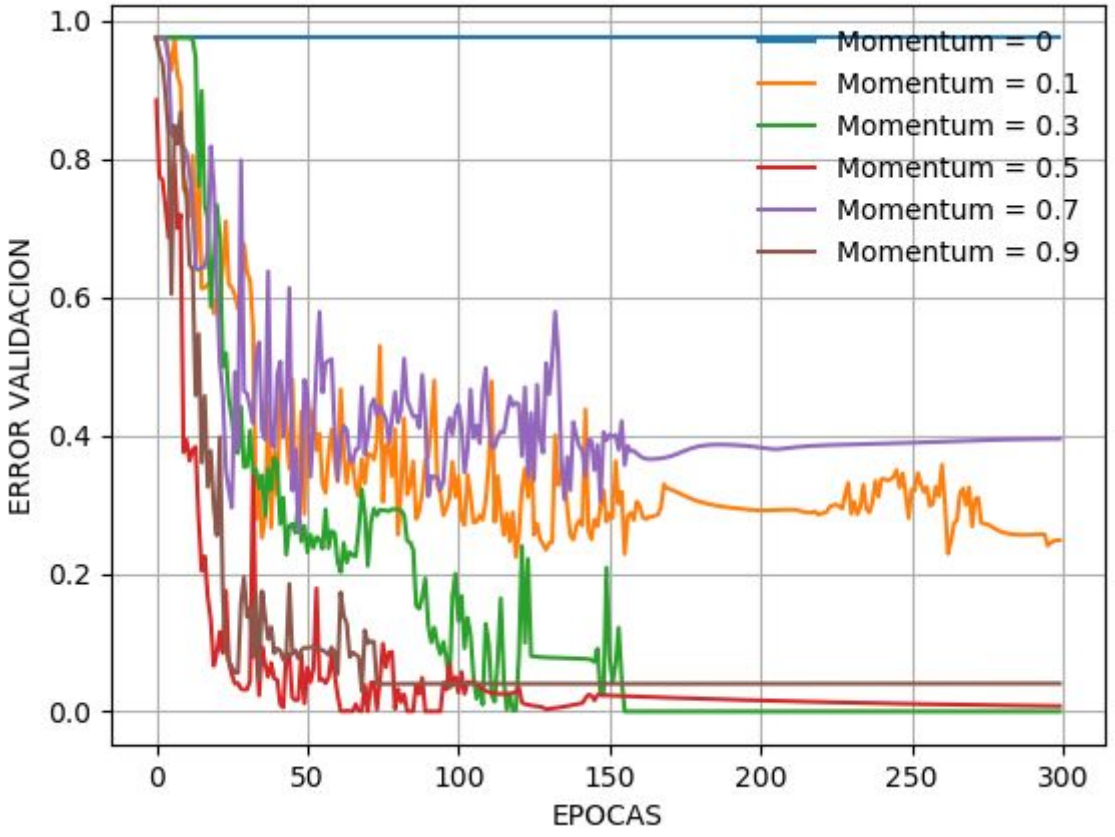

Gráfico 18: Ejercicio 1 - Variación ETA/Momentum - Eta = 0,1 - Error validación

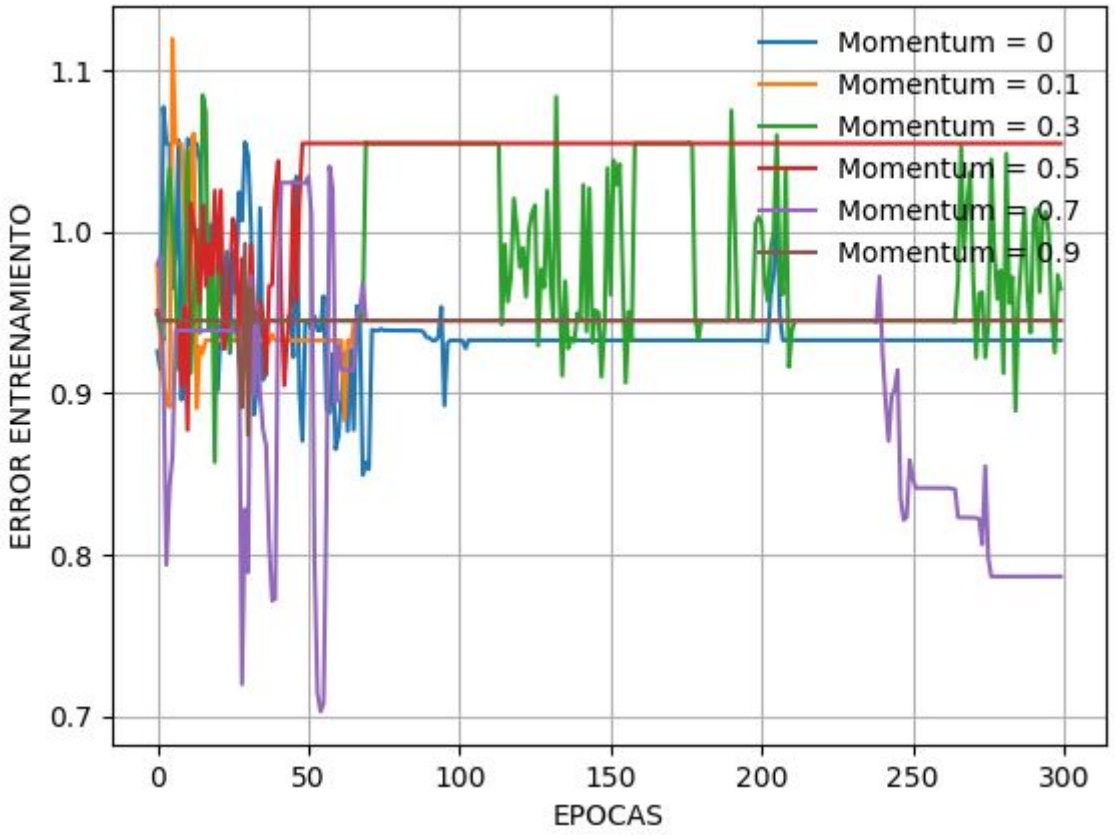

Gráfico 19: Ejercicio 1 - Variación ETA/Momentum - Eta = 0,3 - Error entrenamiento

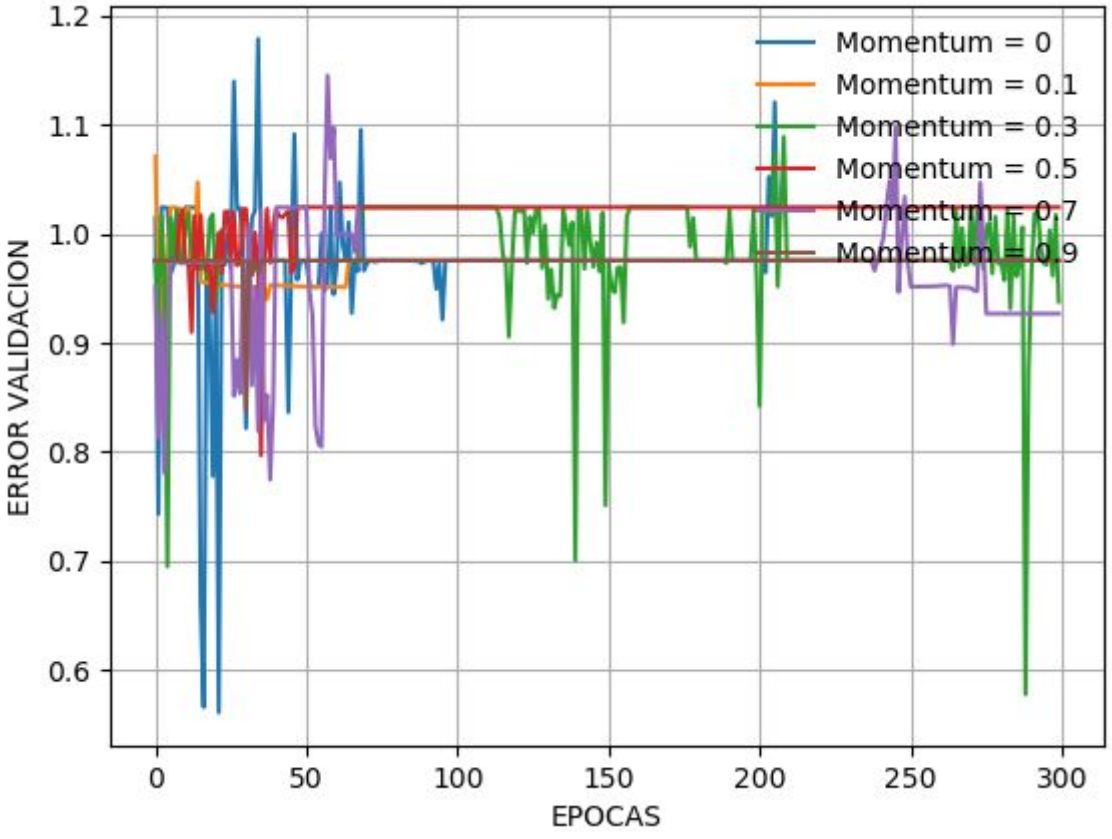

Gráfico 20: Ejercicio 1 - Variación ETA/Momentum - Eta = 0,3 - Error entrenamiento

Observaciones del experimento:

Se observará en principio que para el valor eta=0.3, los resultados son muy malos tanto en entrenamiento como validación, para todo valor del momentum.
Para eta = 0.1, se alcanzan buenos valores con momentum=0.3 y 0.5, no tan malos para 0.9, y para momentum desactivado los resultados fueron los peores.
Para eta = 0.05, en general mejoraron todos los resultados en entrenamiento, pero en validación, no fueron tan buenos, alcanzado el error más bajo para momentum=0.5.
Para eta = 0.005 el error en entrenamiento parece converger más rápido para valores de momentum más grande, pero finalmente todos converger casi al mismo error, que no es muy bajo. En cambio el error de validación es alto para casi todos los valores de momentum, menos para momentum = 0.1, que tiene un error menor a 0.2.
Como conclusión puede verse que el mejor comportamiento se encuentra utilizando eta = 0.1 y momentum con valores 0.9, 0.5 y 0.3.

2.2.8. Performance variando entrenamiento estocástico, batch y mini-batch

En este experimento observaremos el comportamiento de la red asignando distintos valores al batch de procesamiento. Se utilizaron valores de batch = 1(estocástico),5, 10, 100 y 500. Se midió el error de entrenamiento y validación en cada época, y la performance con el set de testing.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%
-Tamaño de set validación: 10%
-Tamaño de set testing: 10%

-Numero de epocas: 500

-Cantidad de capas ocultas y neuronas: 2 capas de 10 neuronas.

-ETA: 0.05

-Momentum:. Desactivado

-Tamaño de batch: **parámetro a estudiar**

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - Tamaño de Batch:

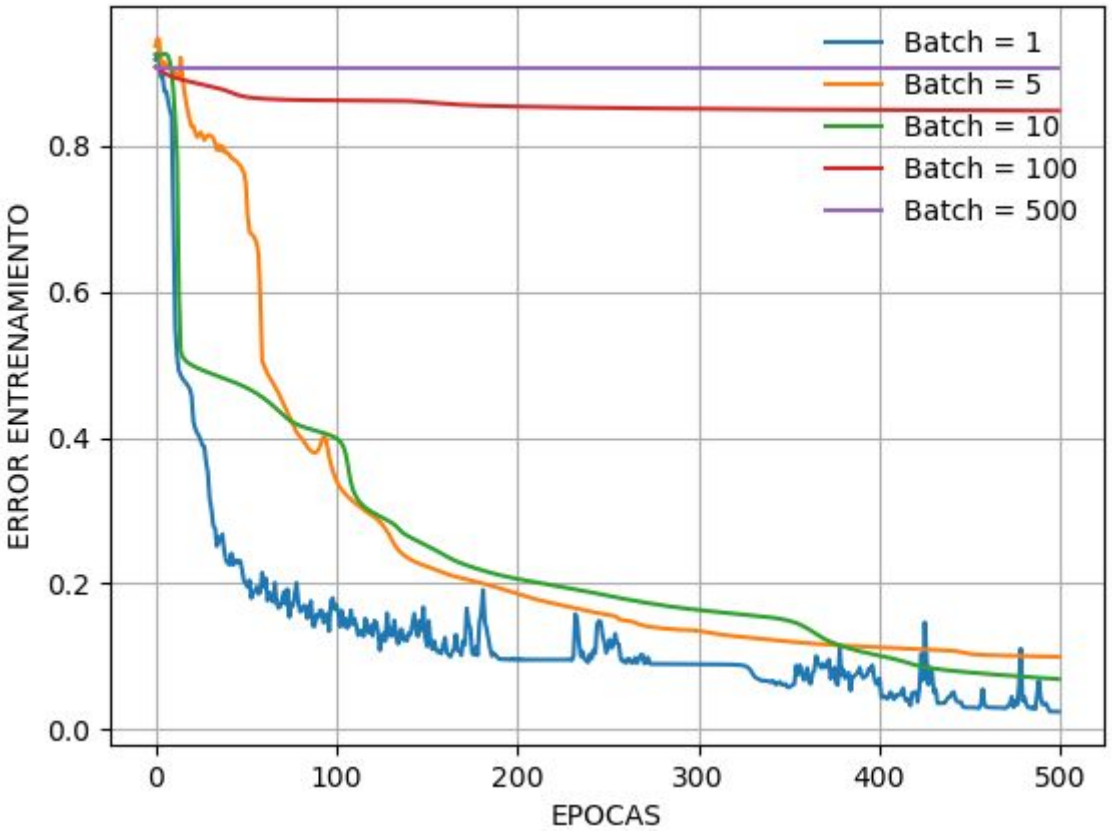

Gráfico 21: Ejercicio 1 - Variación batch - Error entrenamiento

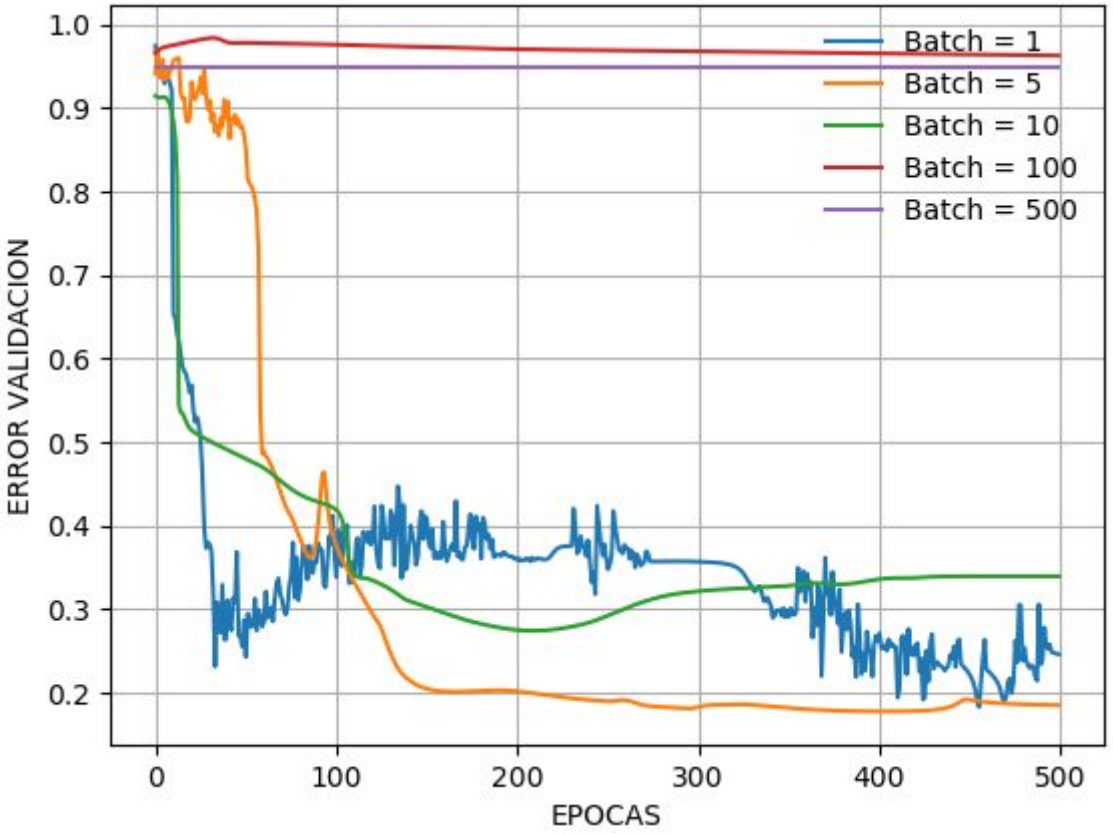

Gráfico 22: Ejercicio 1 - Variación batch - Error validación

Performance en etapa de testing para tamaño de batch:

Tamaño de batch	Performance de Testing
1	90%
5	85%
10	65%
100	48%
500	48%

Observaciones del experimento:

Se observa que los peores resultados se encuentran tomando batch = 100 y 500, para entrenamiento, validación y testing.
Para batch de tamaño 10, el error de entrenamiento y validación no parecen tan malos, pero sin embargo en testing, la performance es muy baja.
Los mejores resultados en entrenamiento y validación son para batch con valor 1 y 5. En error de entrenamiento tienen valores muy cercanos, pero en error de validación, con batch=5, parece tener mejores resultados, pero en testing, batch=1, tiene un 5% más de aciertos.

2.3 Ejercicio 2

2.3.1. Variación de la función de activación

En este experimento se probaron las funciones tangencial y logística para ver cómo se comporta la red neuronal utilizando cada una. Se observó el error por época, en la fase de entrenamiento.

Arquitectura de la red:

-Función de activación: **variable a estudiar**

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas ocultas de 8 neuronas

-ETA: 0.05

-Momentum: desactivado

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados:

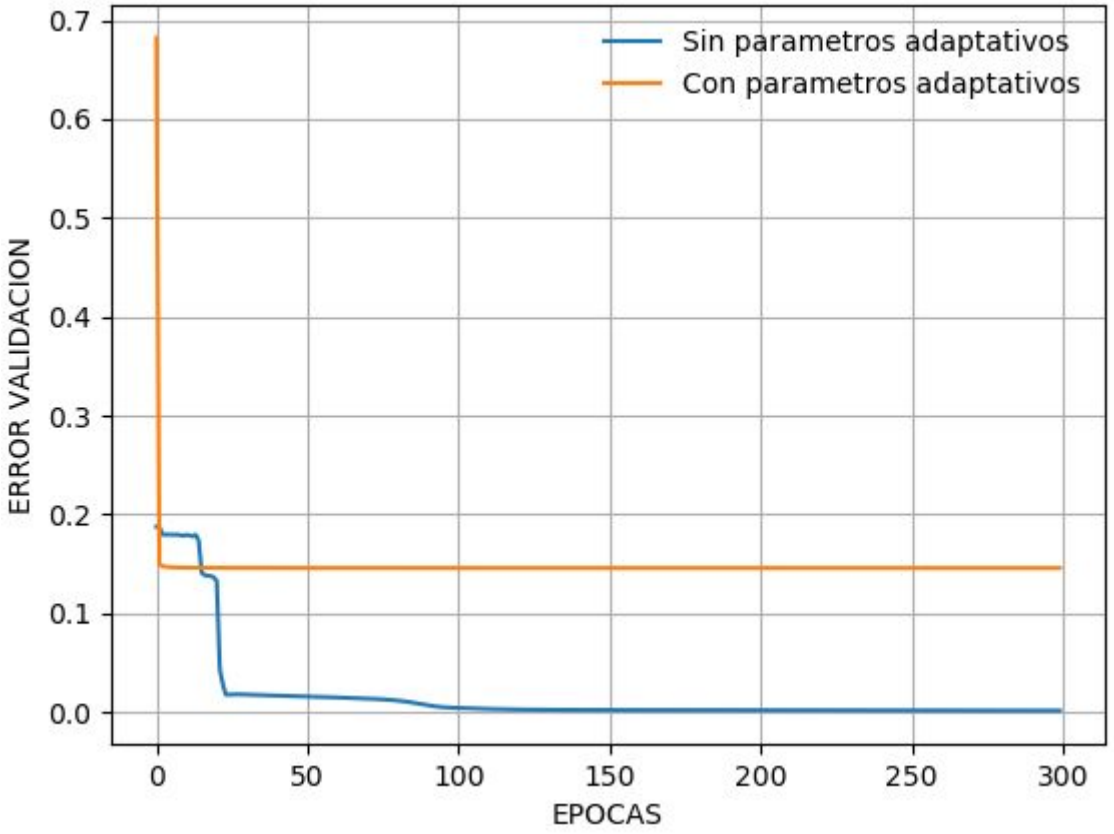

Gráfico 23: Ejercicio 2 - Función de logística vs. función tangencial

Observaciones del experimento:

Se puede ver al igual que en el ejercicio 1, que al utilizar la función logística el error es muy alto, no baja de 0.15. En cambio con la función tangencial el resultado es óptimo.
A partir de esto, se decide utilizar la función tangencial como función de activación para el resto de los experimentos de este ejercicio.

2.2.3. Variación de número de capas ocultas y número de neuronas

Este este experimento se verificó el comportamiento de la red neuronal variando las capas ocultas y la cantidad de neuronas de las misma. Se probó con 1, 2, 3 y 5 capas ocultas, con 8 neuronas cada una. Luego una vez obtenida la que cantidad de capas que mejor resultado obtuvo, se varió el número de neuronas en 2, 5, 8 y 16. Se verificaron los errores por época tanto de entrenamiento, como de validación.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: **parámetro a estudiar**

-ETA: 0.05

-Momentum: Desactivado

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - Variación número de capas ocultas y neuronas:

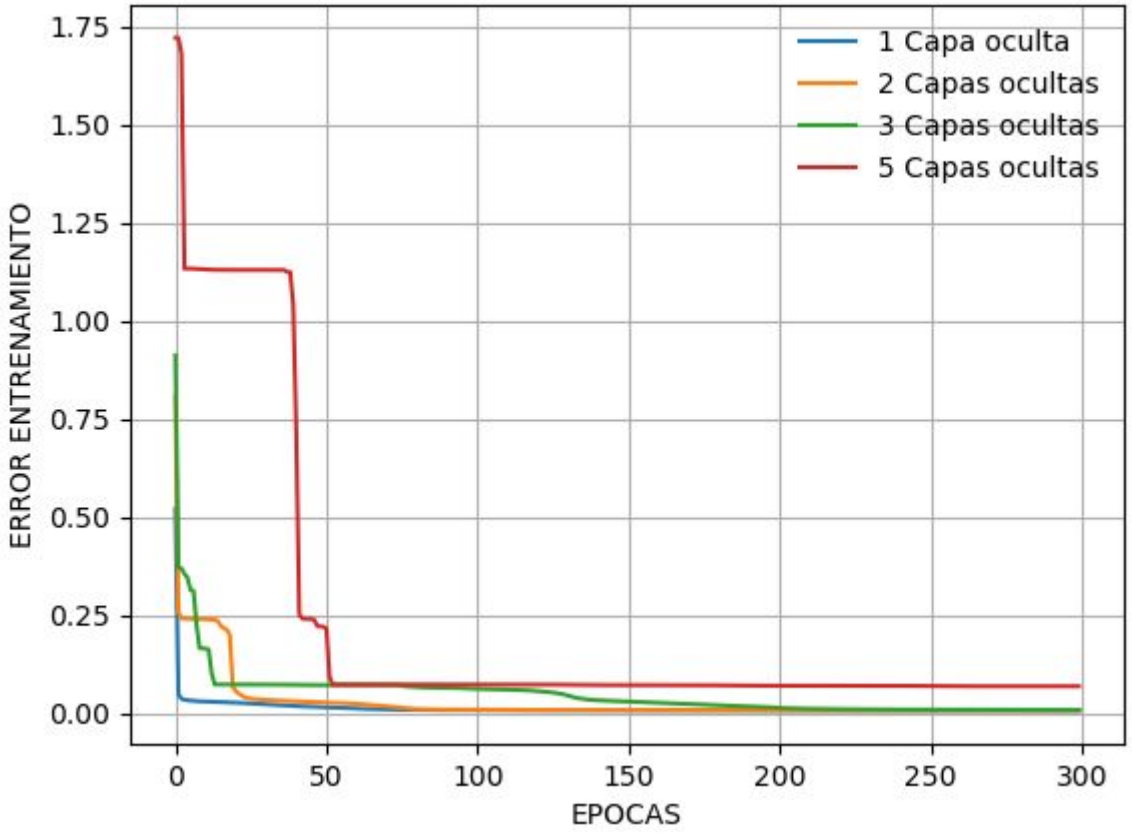

Gráfico 24: Ejercicio 2 - Variación capas ocultas - Error entrenamiento

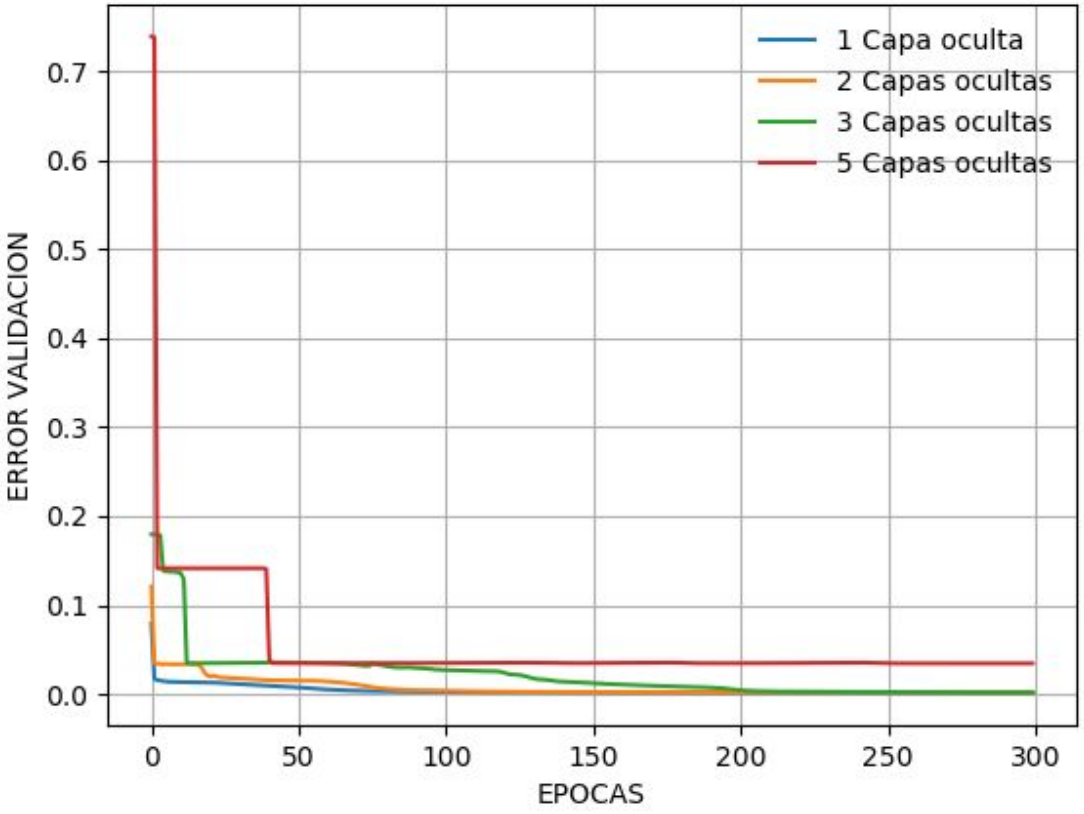

Gráfico 25: Ejercicio 2 - Variación capas ocultas - Error validación

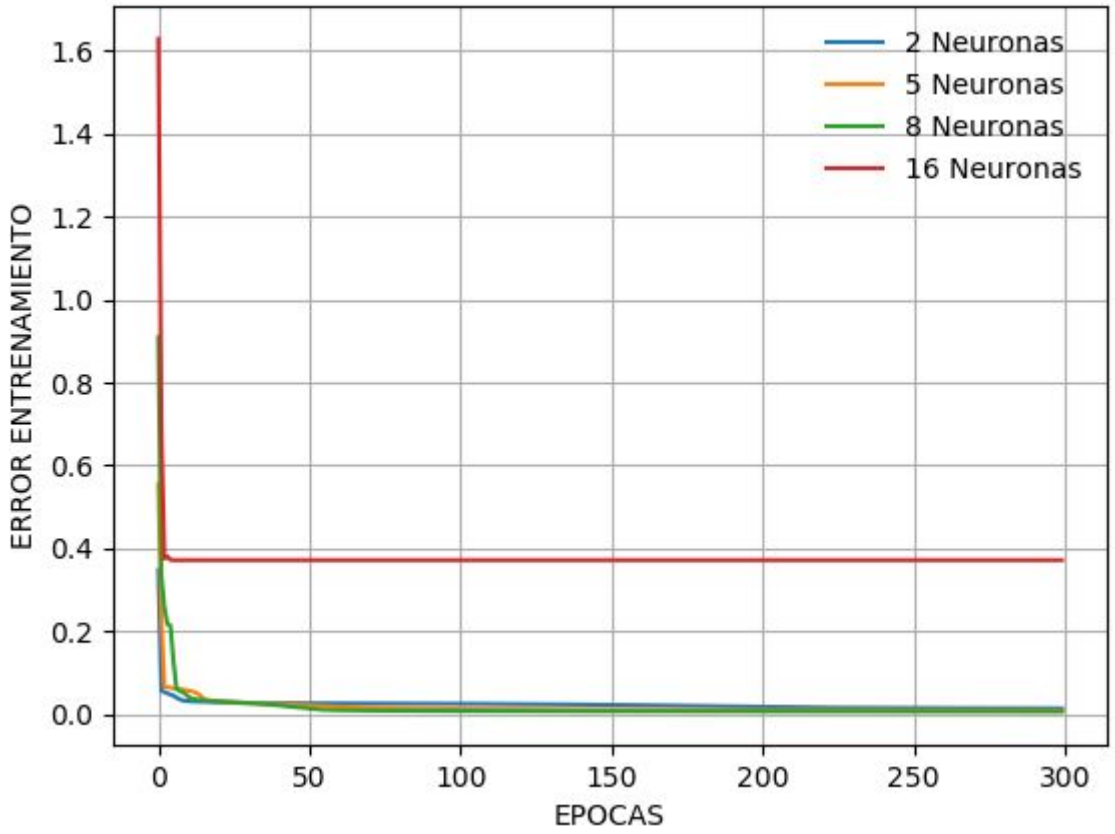

Gráfico 26: Ejercicio 1 - Variación cantidad de neuronas - Error entrenamiento

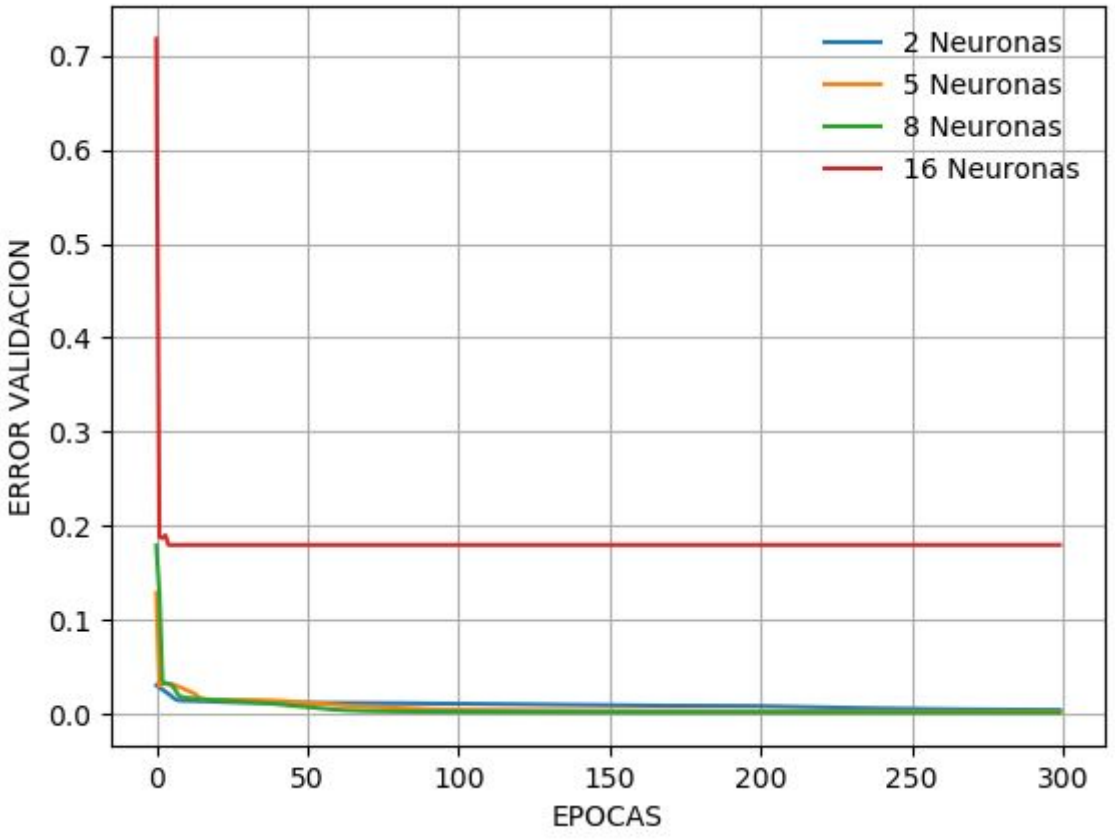

Gráfico 27: Ejercicio 1 - Variación cantidad de neuronas - Error entrenamiento

Observaciones del experimento:

Se puede observar que para 1, 2 y 3 capas de 8 neuronas, el comportamiento es bastante bueno, el error tanto en entrenamiento como validación converge a valores muy cercanos a 0 rápidamente. El peor resultado se encuentra con 5 capas, lo cual es entendible, porque 5 capas de 8 neuronas cada una, parece una arquitectura muy grande para este problema. ***Se decidió de aquí en adelante usar 2 capas ya que parece ser la configuración que más rápido converge.***
Para las pruebas de variación de cantidad de neuronas, se puede observar algo similar, para 2, 5 y 8 neuronas, el comportamiento es muy bueno, pero para 16, es muy malo. ***Se decide tomar a partir de este experimento la configuración de 5 neuronas,*** por tener buenos resultados y mejor velocidad de ejecución.

2.2.3. Variación en la forma de iniciar los pesos

Se observo el comportamiento de la red iniciando los pesos de las neuronas con dos distribuciones distintas, la distribución normal y la distribución uniforme.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas de 5 neuronas.

-ETA: 0.05

-Momentum: Desactivado

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: **parámetro a estudiar**

Resultados - Forma de iniciar los pesos:

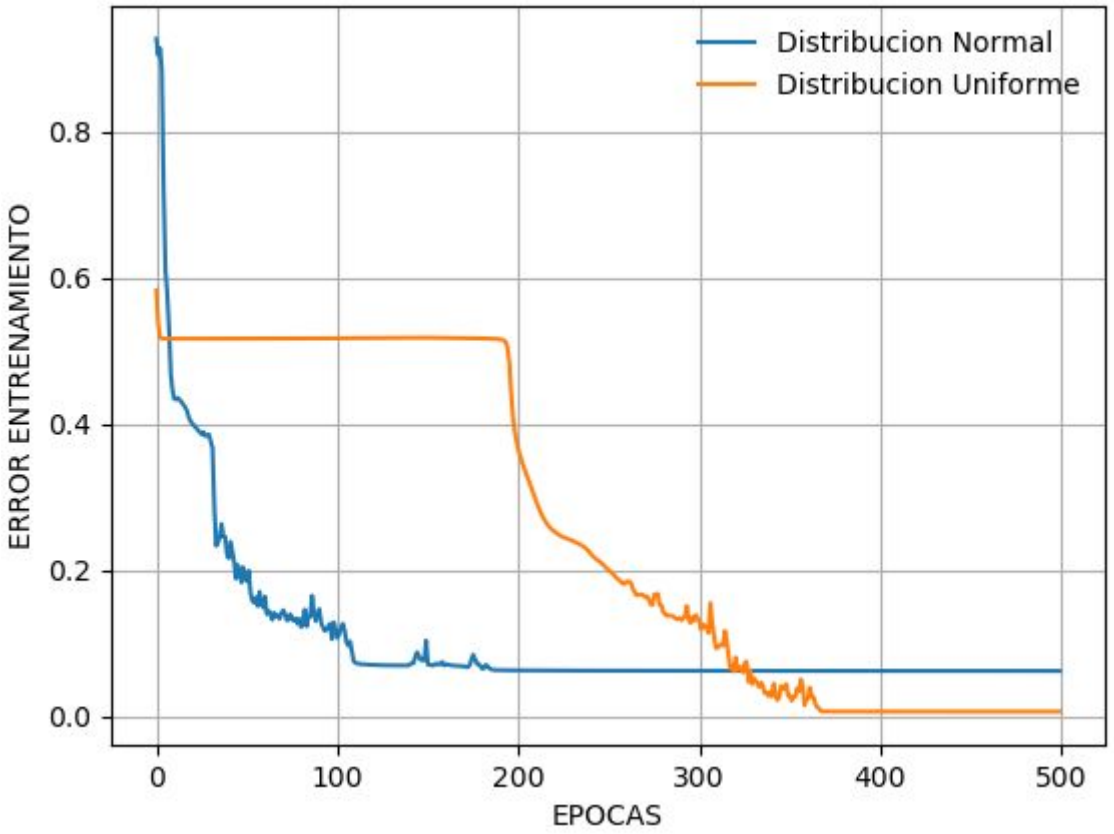

Gráfico 28: Ejercicio 2 - Variación distribución de pesos - Error entrenamiento

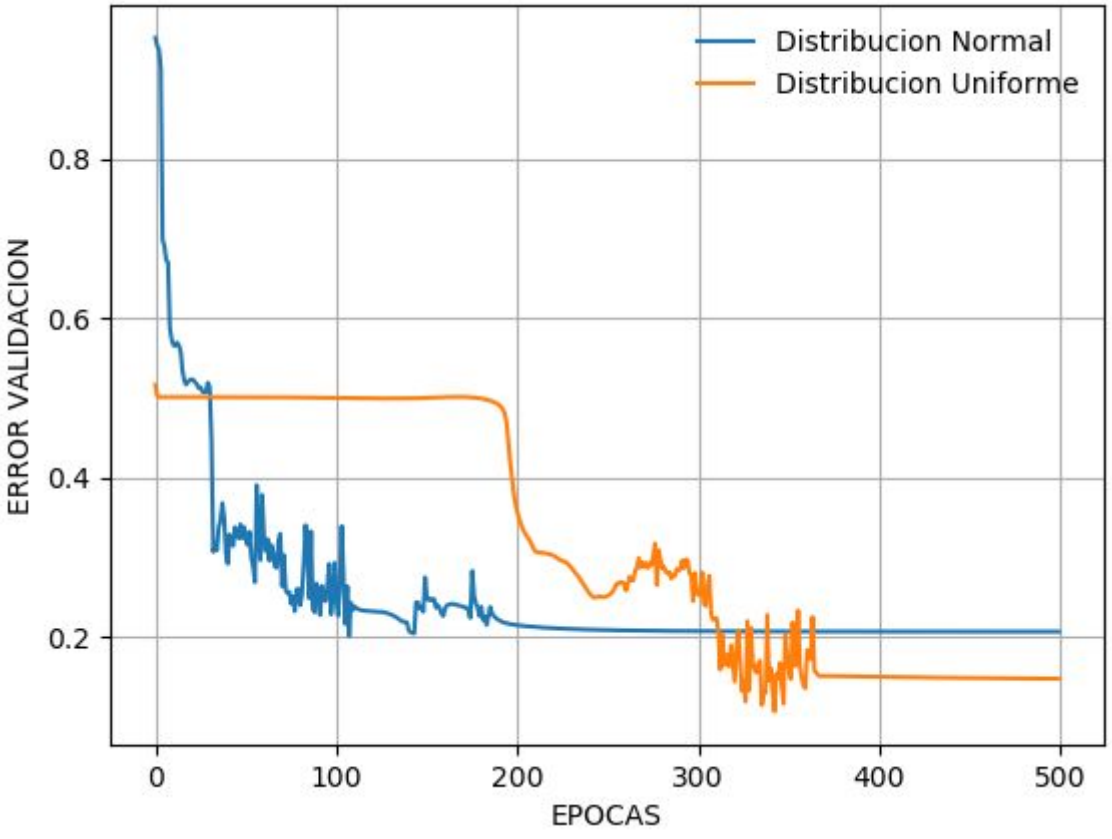

Gráfico 29: Ejercicio 2 - Variación distribución de pesos - Error validación

Observaciones del experimento:

Los resultados muestran que la distribución uniforme demora más en converger a un nivel de error bajo, pero que al hacerlo, el valor es más bajo que con la distribución normal. Este comportamiento se observa tanto en el error de entrenamiento como de validación.
Además se observa que antes de las 300 épocas, la distribución normal parece funcionar mejor, pero después con la uniforme se logra minimizar más el error.

2.2.4. Performance con Momentum

Se observó el comportamiento de la red con y sin momentum y su error en la etapa de entrenamiento y en validación para cada época. Se varió el parámetro con los siguientes valores: 0.1, 0.3, 0.5, 0.7 y 0.9.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 500

-Cantidad de capas ocultas y neuronas: 2 capas de 5 neuronas.

-ETA: 0.05

-Momentum: **valor a estudiar**

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Uniforme

Resultados - Momentum:

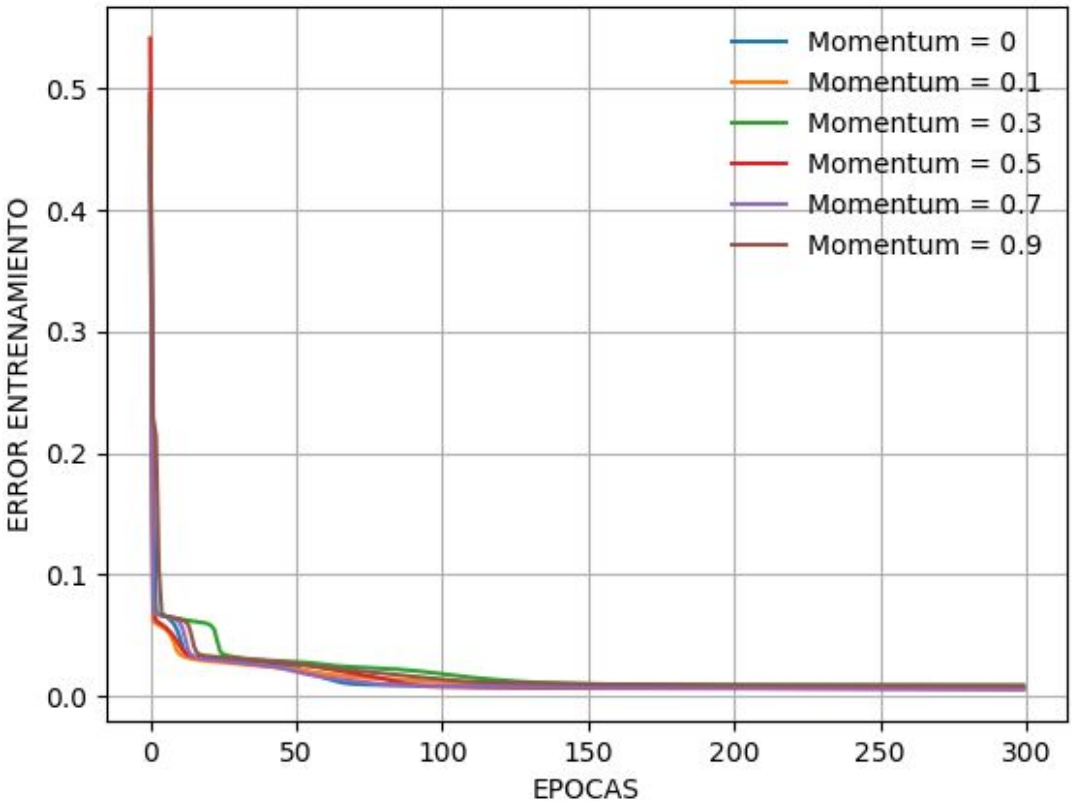

Gráfico 30: Ejercicio 2 - Variación parámetro Momentum - Error entrenamiento

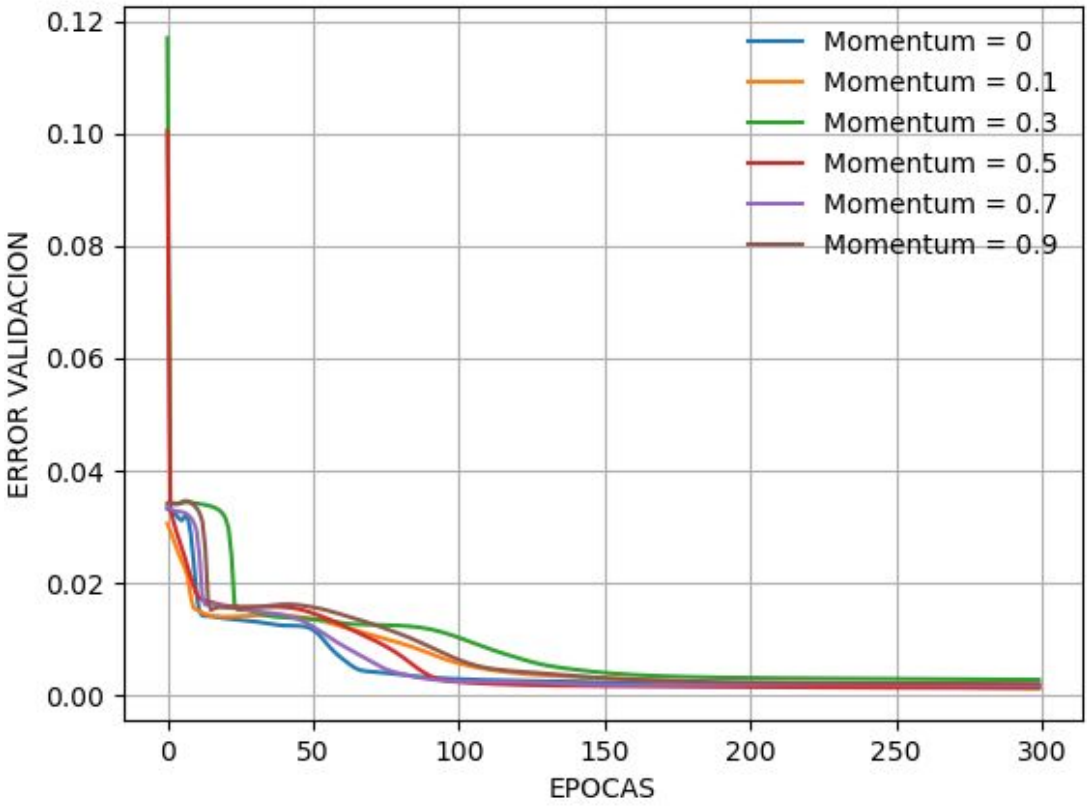

Gráfico 31: Ejercicio 2 - Variación parámetro Momentum - Error validación

Observaciones del experimento:

Se observa que para todos los valores de momentum que obtuvieron buenos resultados. A partir de la época número 150, se converge a un valor muy cercano a cero en el error de entrenamiento y de validación.

2.2.5. Performance con Early-Stopping

Se observó el comportamiento de la red con y sin Early-Stopping y su error en la etapa de entrenamiento y validación para cada época. Se varió el valor del parámetro con los siguientes valores: 0.001, 0.005, 0.009. Se decidió utilizar valores tan cercanos a 0, porque se pudo observar que el error de validación en los anteriores experimentos de este ejercicio es muy bajo.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 500

-Cantidad de capas ocultas y neuronas: 2 capas de 5 neuronas.

-ETA: 0.05

-Momentum: Desactivado

-Tamaño de batch: 1

-EarlyStopping: **parametro a estudiar**

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - Early-Stopping:

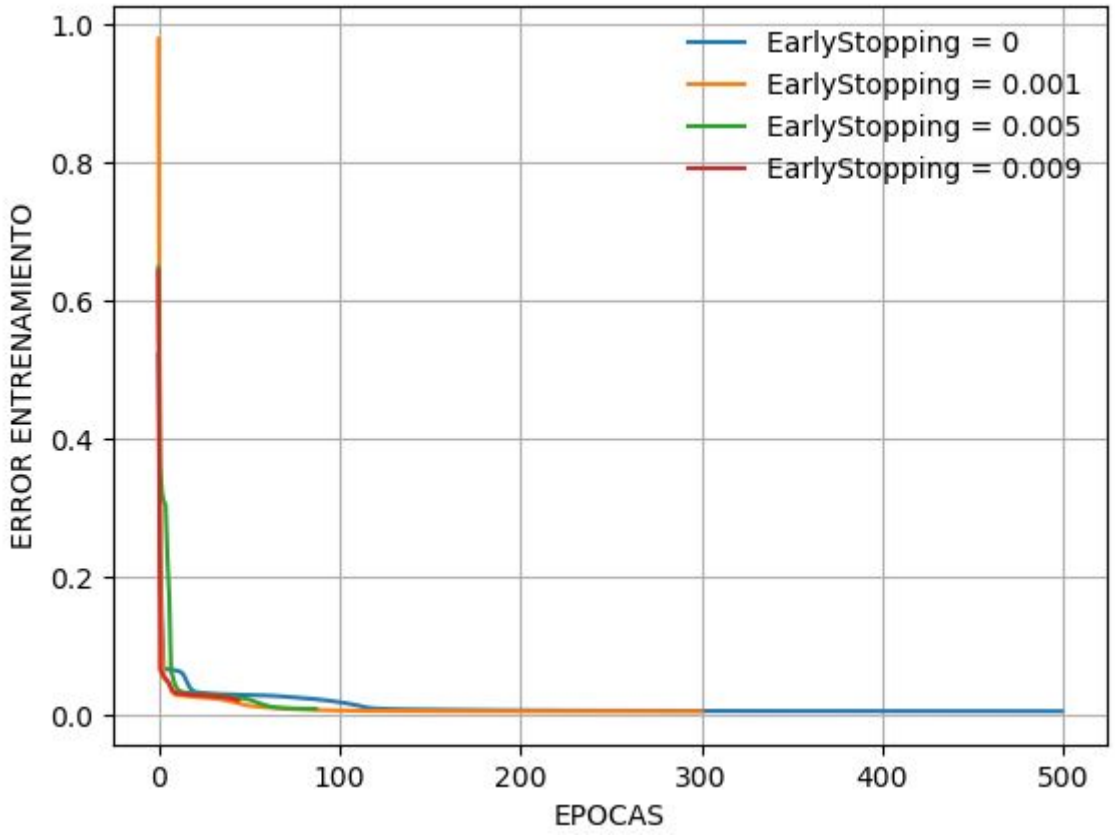

Gráfico 32: Ejercicio 2 - Variación parámetro Early-stopping - Error entrenamiento

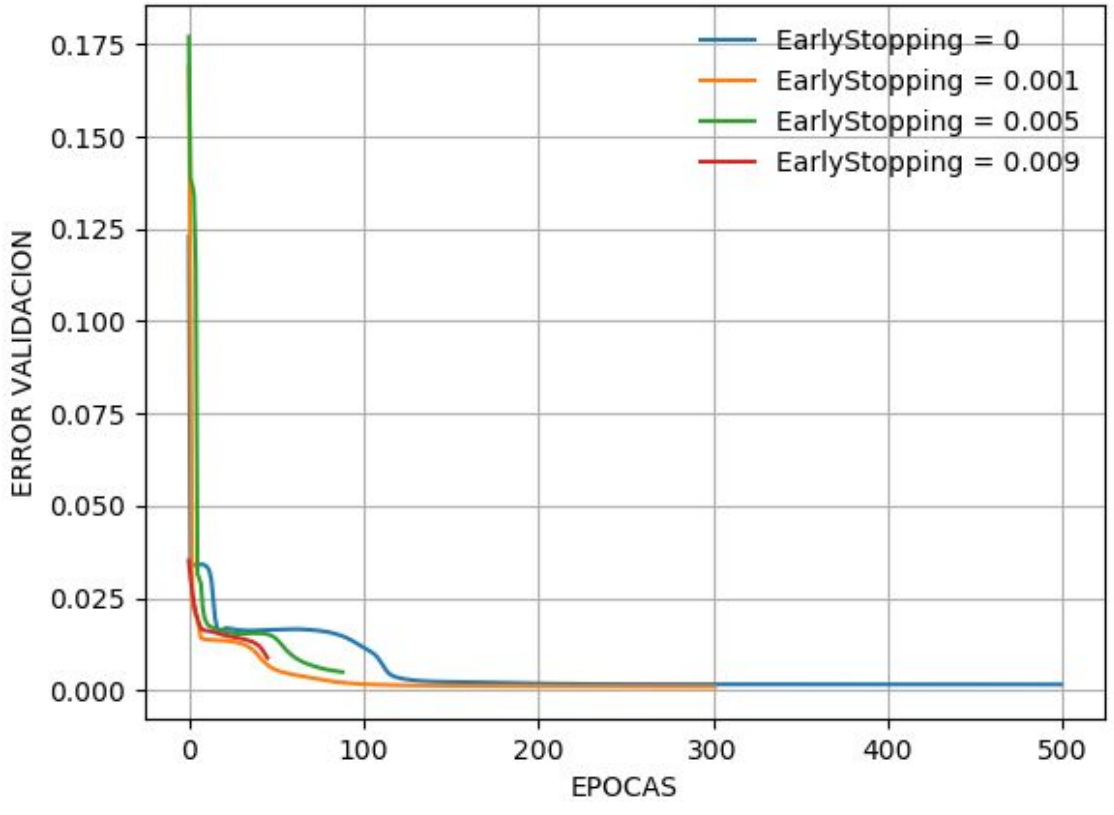

Gráfico 33: Ejercicio 2 - Variación parámetro Early-stopping - Error validación

Performance en etapa de testing para Early-Stopping:

Valor Early-Stopping	Performance de Testing
Desactivado	86%
0.001	98%
0.005	80%
0.009	70%

Observaciones del experimento:

Se observa que para los valores 0.005 y 0.009, el entrenamiento se corta rápidamente antes de la epoca numero 100 y que para 0.001 el entrenamiento no se corta. Sin embargo en la etapa de testing los mejores resultados se encuentran con early-stopping desactivado (86%) y con early-stopping = 0.001 (98%) sabiendo que este último valor nunca es alcanzado en el entrenamiento (por lo que no corta). De esta forma se puede concluir que este feature no parece ser del todo útil cuando los valores de la validación son tan bajos.

2.2.6. Performance con Parametros Adaptativos

Se observó el comportamiento de la red con y sin parámetros adaptativos. La configuración de este feature, es la siguiente:

-Si el error de la época anterior es menor al actual, entonces sumó al eta el 1% de su valor.

-Si el error de la época anterior es mayor al actual, entonces resto al eta la mitad de su valor.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 500

-Cantidad de capas ocultas y neuronas: 2 capas de 5 neuronas.

-ETA: 0.05

-Momentum: Desactivado

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: **parámetro a estudiar**

-Distribución para los pesos: Normal

Resultados - Parámetros Adaptativos:

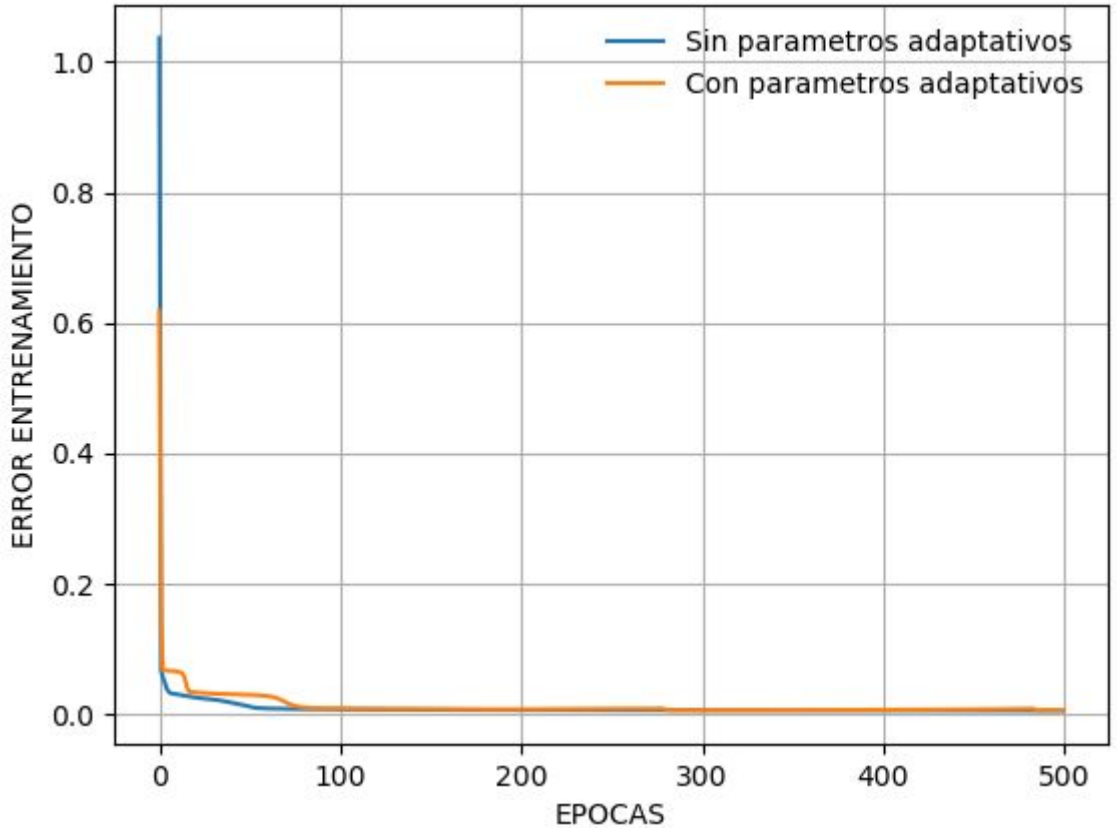

Gráfico 34: Ejercicio 2 - Variación Parámetros adaptativos - Error Entrenamiento

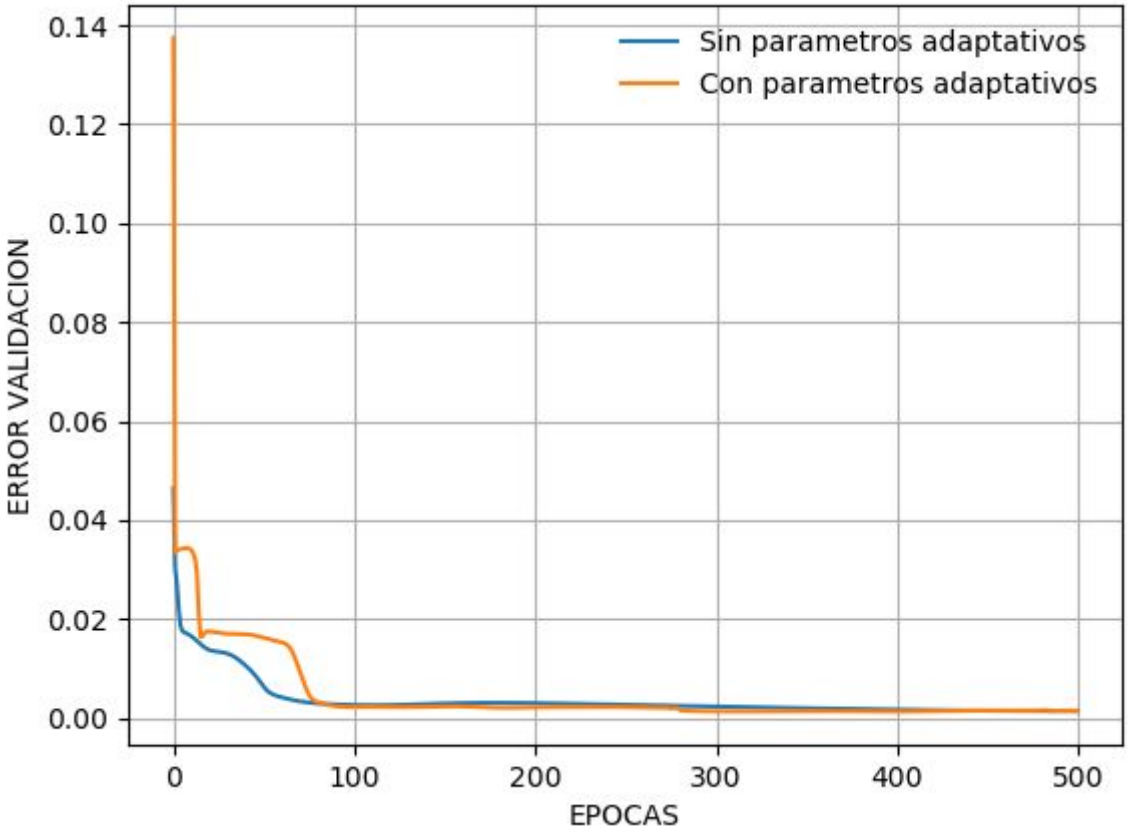

Gráfico 35: Ejercicio 2 - Variación Parámetros adaptativos - Error validación

Performance en etapa de testing para Parámetros Adaptativos:

Valor Early-Stopping	Performance de Testing
Desactivado	94%
Activado	88%

Observaciones del experimento:

Se observa algo similar al experimento anterior, una rápida convergencia a un nivel de error muy bajo tanto si el feature está o no activado, en el entrenamiento y en la validación. Lo que podemos ver en los gráficos, es que con el feature activado se tardan algunas épocas más en converger.
También se puede observar un mejor rendimiento en la etapa de test, de más del 6%, si los parámetros adaptativos están desactivados.

2.2.7. Performance variando factor de aprendizaje y parámetro del momentum

En este experimento se varió conjuntamente los valores de eta y momentum, buscando combinaciones que logren buenos resultados. Para cada uno de los valores de eta 0.005, 0.05, 0.1, 0.3, se utilizaron valores de momentum 0.1, 0.3, 0.5, 0.7, 0.9, y también momentum desactivado. Se evaluó el error de entrenamiento y validación para cada época.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 300

-Cantidad de capas ocultas y neuronas: 2 capas de 5 neuronas.

-ETA: **parametro a estudiar**

-Momentum: **parametro a estudiar**

-Tamaño de batch: 1

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - ETA / Momentum:

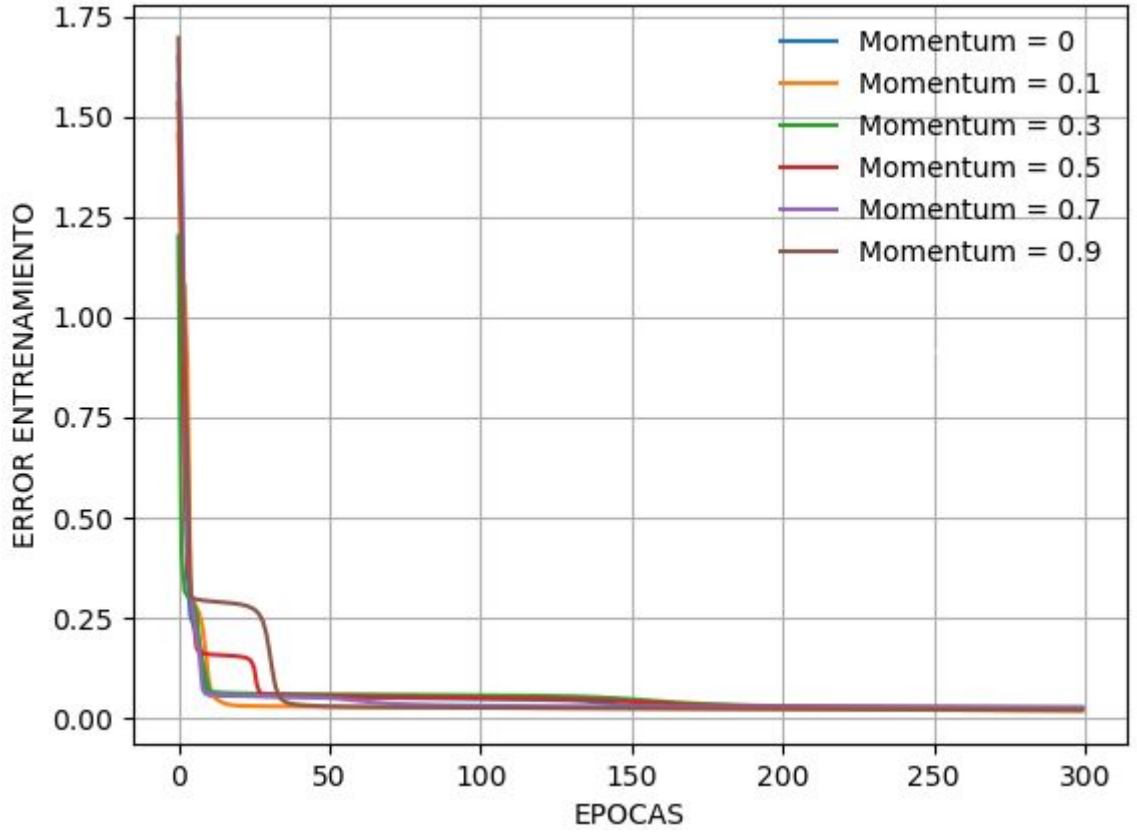

Gráfico 36: Ejercicio 2 - Variación ETA/Momentum - Eta = 0,005 - Error entrenamiento

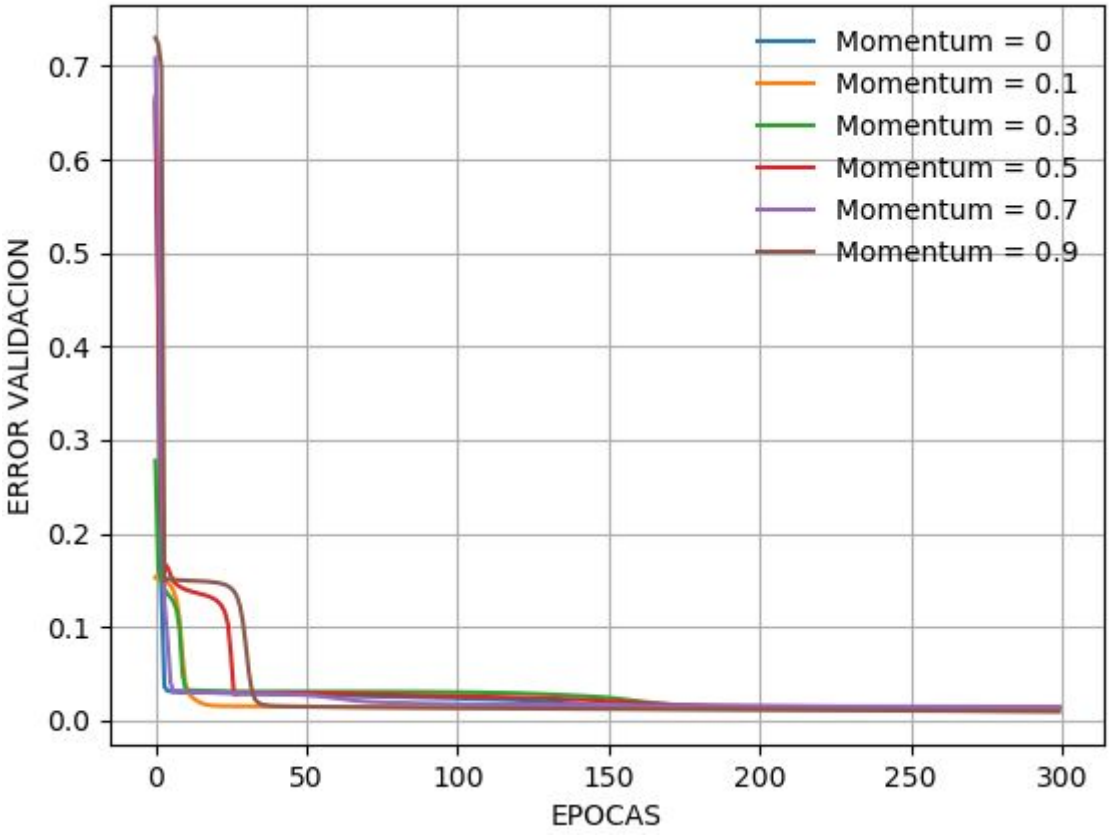

Gráfico 36: Ejercicio 2 - Variación ETA/Momentum - Eta = 0,005 - Error validación

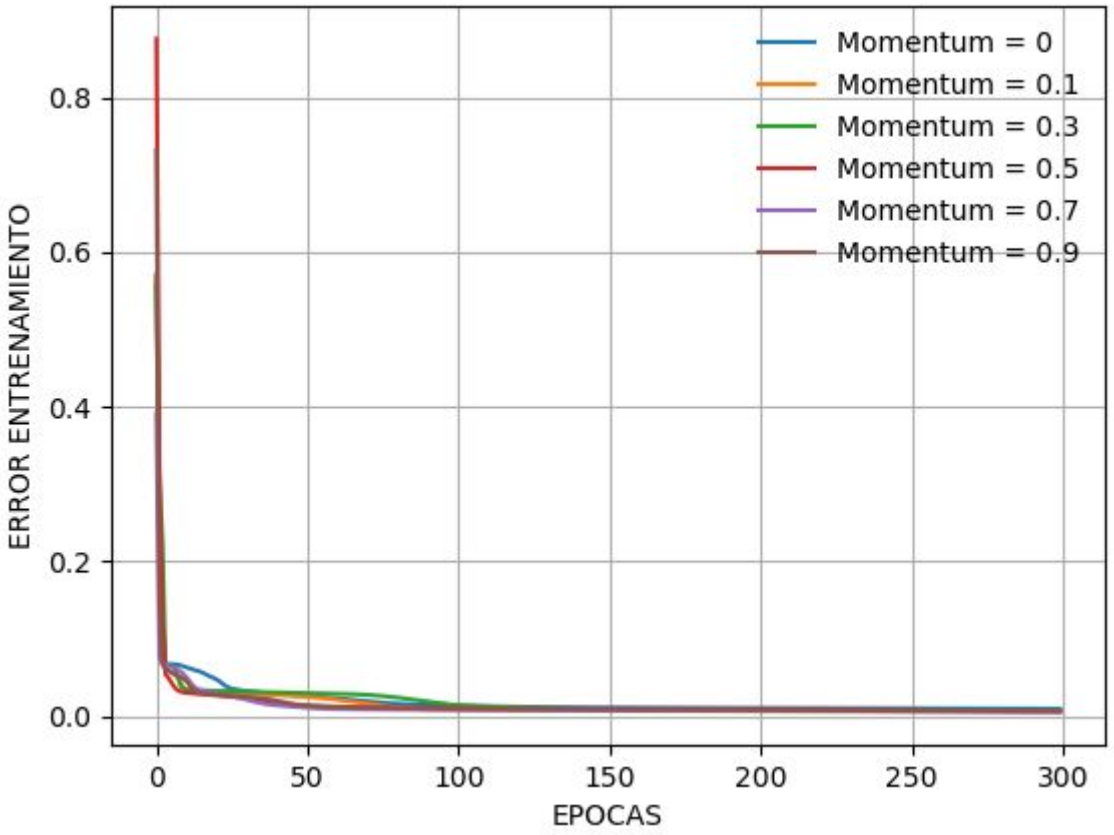

Gráfico 37: Ejercicio 2 - Variación ETA/Momentum - Eta = 0,05 - Error validación

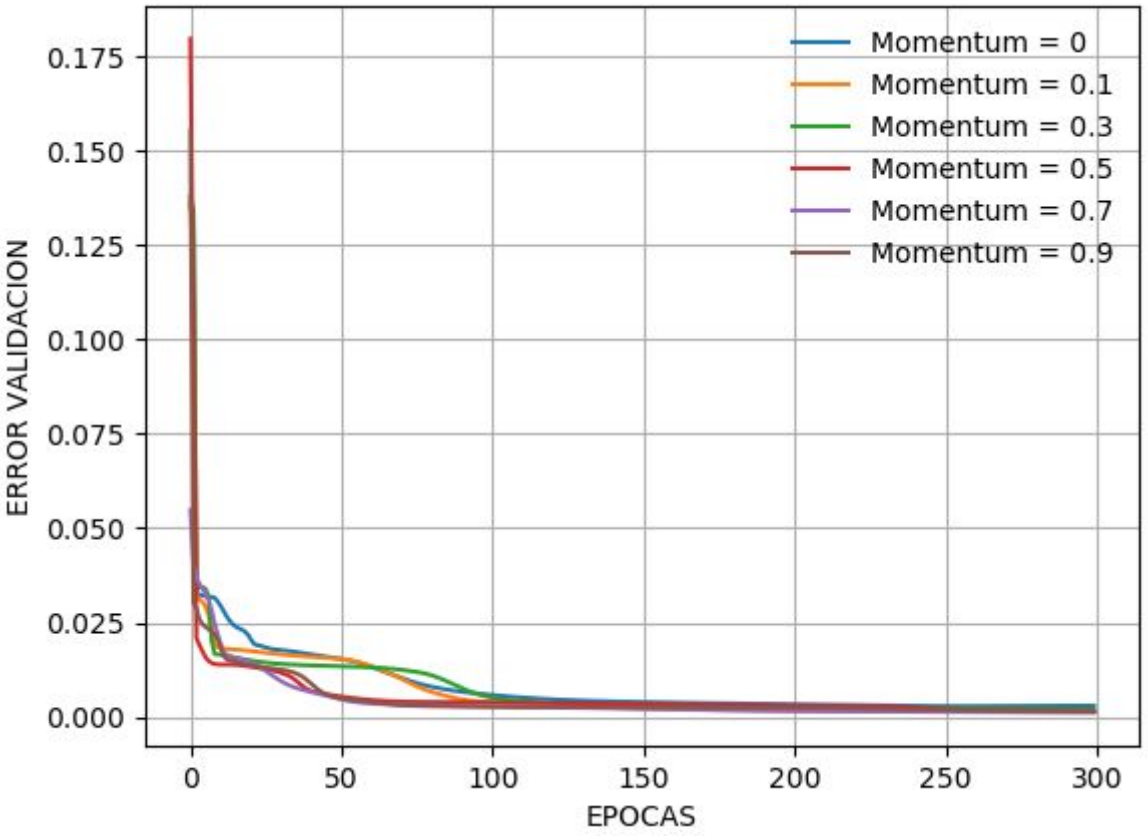

Gráfico 38: Ejercicio 2 - Variación ETA/Momentum - Eta = 0,05 - Error validación

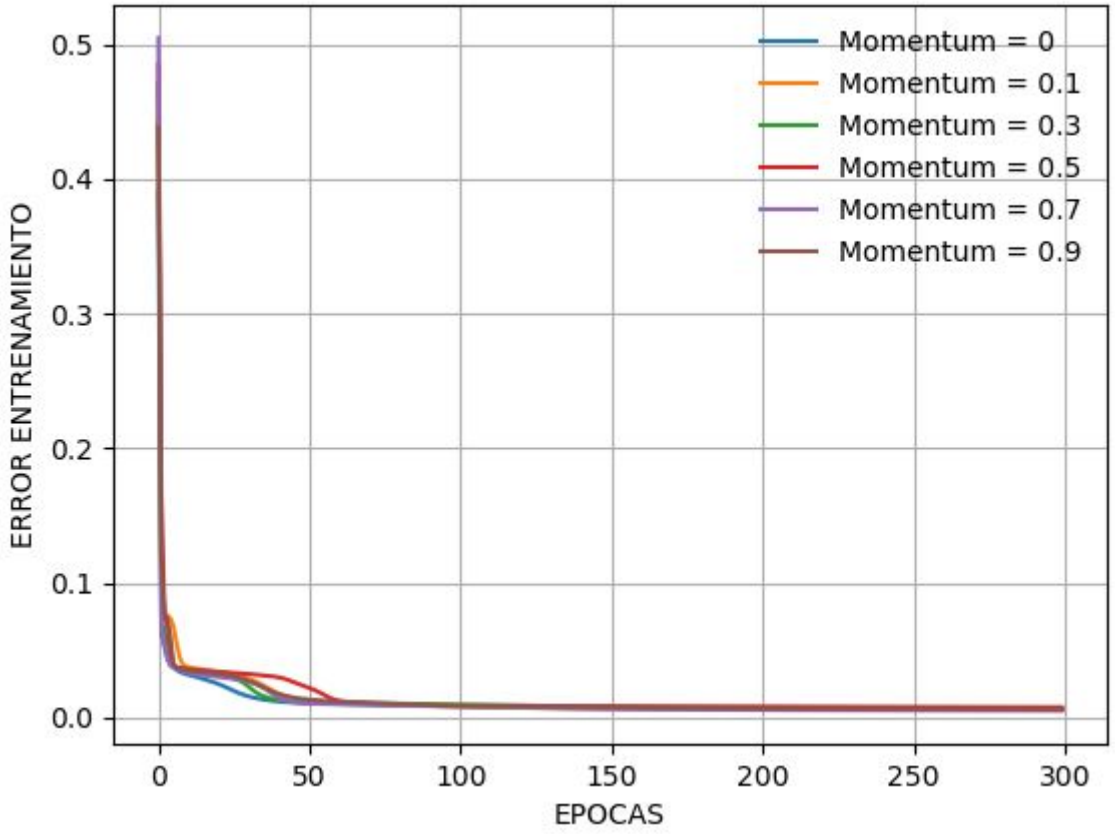

Gráfico 39: Ejercicio 2 - Variación ETA/Momentum - Eta = 0,1 - Error entrenamiento

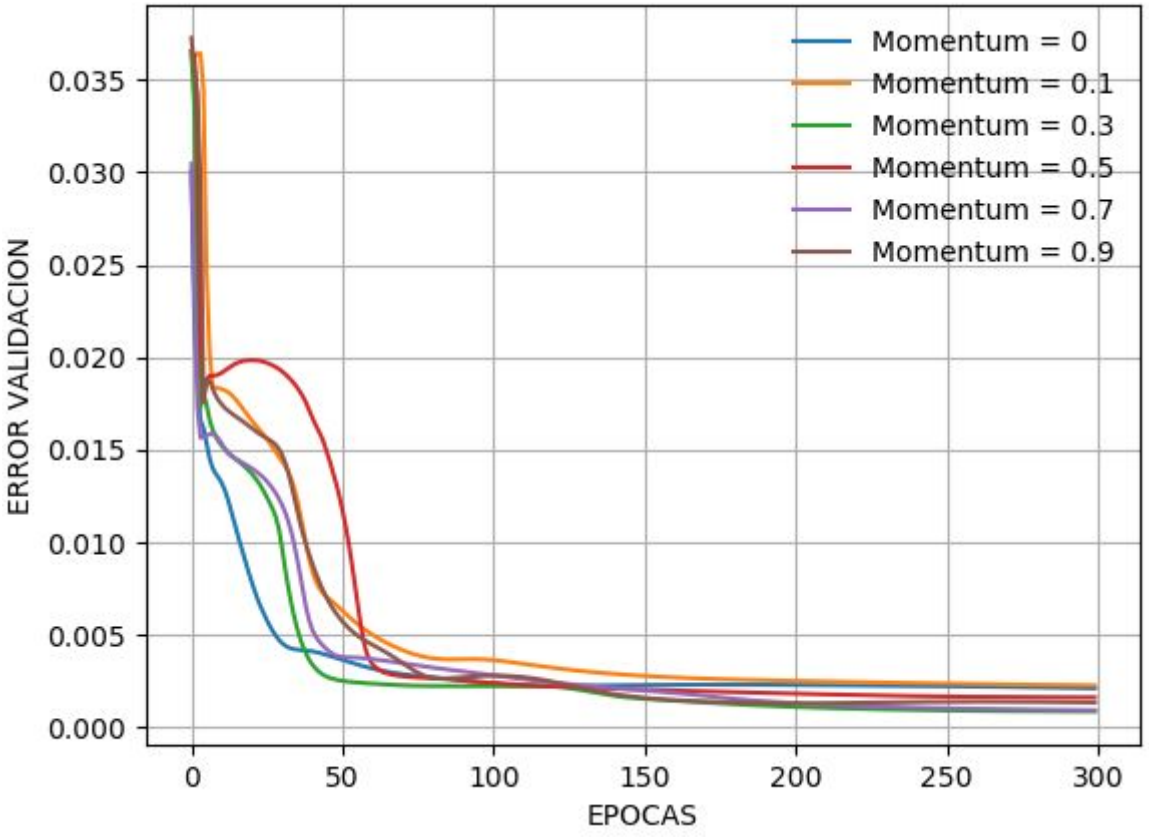

Gráfico 40: Ejercicio 2 - Variación ETA/Momentum - Eta = 0,1 - Error validación

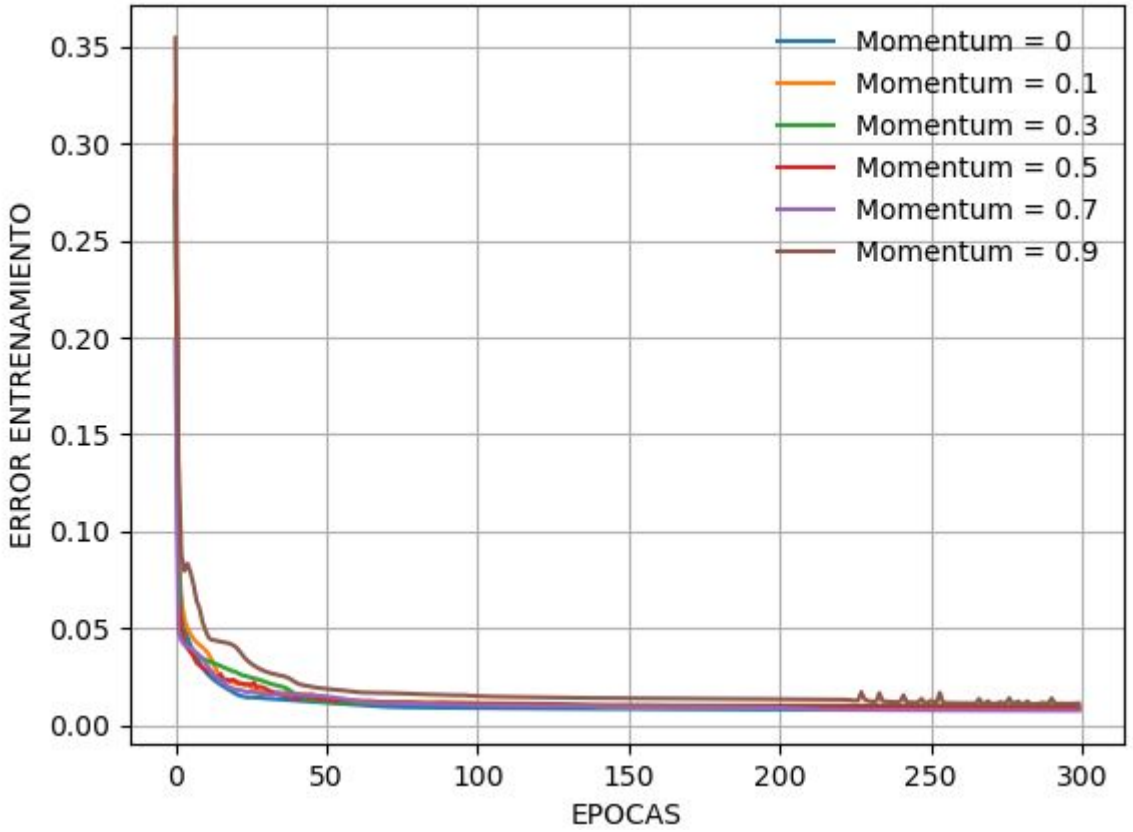

Gráfico 41: Ejercicio 2 - Variación ETA/Momentum - Eta = 0,3 - Error entrenamiento

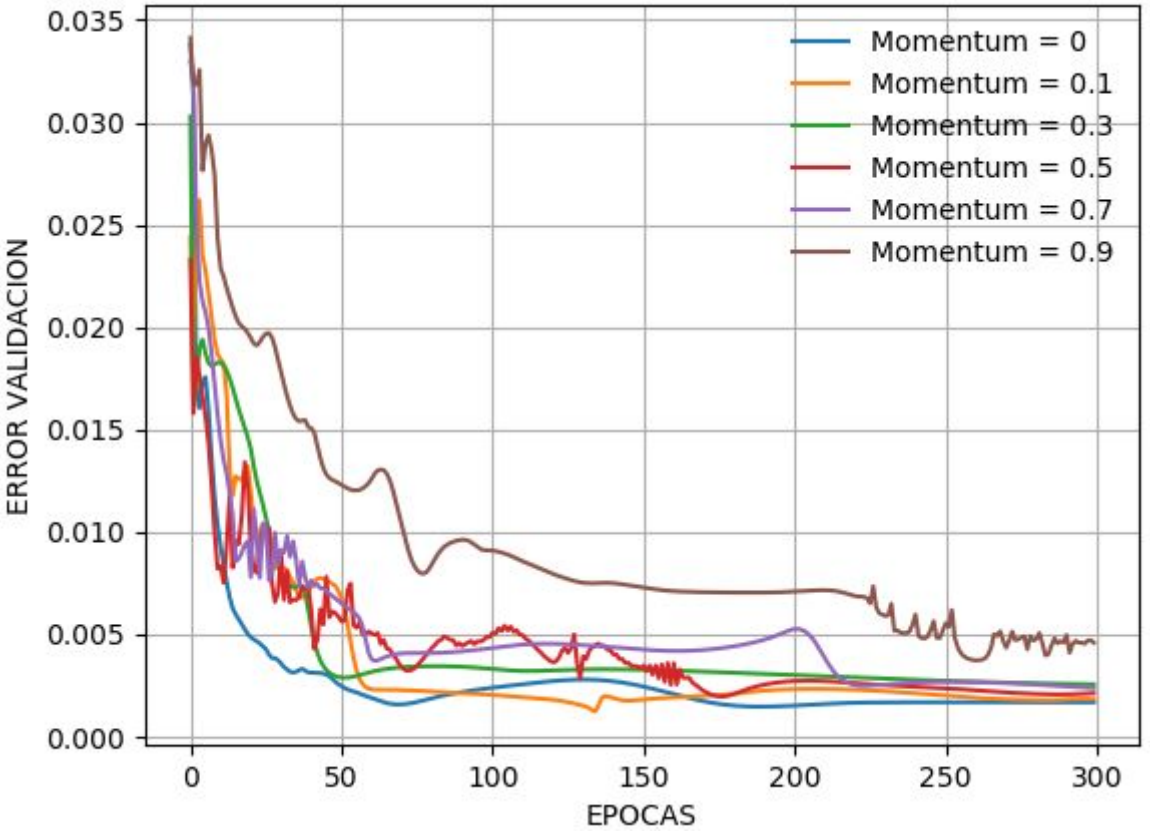

Gráfico 42: Ejercicio 2 - Variación ETA/Momentum - Eta = 0,3 - Error entrenamiento

Observaciones del experimento:

Se observa que para eta=0.3, si bien los resultados parecen ser buenos en el error de entrenamiento, el error de validación es el que más tarda en converger para todos los valores de momentum. Para el valor 0.09, el valor de error en la validación, es el peor de todos.
Para eta=0.005 y eta=0.05, los comportamientos son muy similares, con variación en la cantidad de épocas que tarda en converger a un nivel de error bajo, siempre cercano a las 100 épocas para casi todos los valores de momentum.
Para el valor de eta = 0.1, existe un comportamiento particular para todos los valores de momentum, donde hay un salto del error de validación en las primera 50 epocas, y luego converge más suavemente para todos los valores de momentum.
Dado que prácticamente para todos los valores de eta (menos para 0.3) y de momentum, se obtiene un buen comportamiento, no se puede concluir cuál combinación es la óptima.

2.2.8. Performance variando entrenamiento estocástico, batch y mini-batch

En este experimento observamos el comportamiento de la red asignando distintos valores al batch de procesamiento. Se utilizaron valores de batch = 1(estocástico), 5, 10, 100 y 500. Se midió el error de entrenamiento y validación en cada época, y la performance del testing.

Arquitectura de la red:

-Tamaño de set entrenamiento: 80%

-Tamaño de set validación: 10%

-Tamaño de set testing: 10%

-Numero de epocas: 500

-Cantidad de capas ocultas y neuronas: 2 capas de 5 neuronas.

-ETA: 0.05

-Momentum:. Desactivado

-Tamaño de batch: **parámetro a estudiar**

-EarlyStopping: Desactivado

-Parámetros adaptativos: Desactivado

-Distribución para los pesos: Normal

Resultados - Tamaño de Batch:

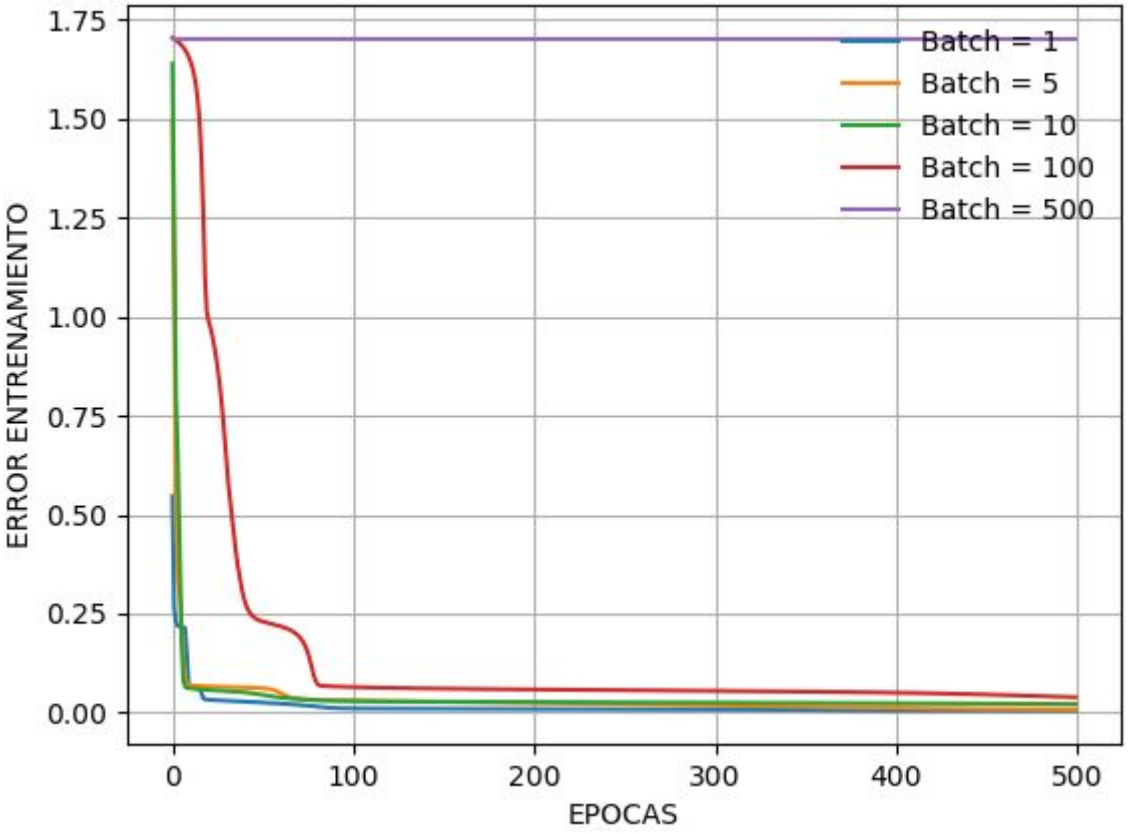

Gráfico 43: Ejercicio 1 - Variación batch - Error entrenamiento

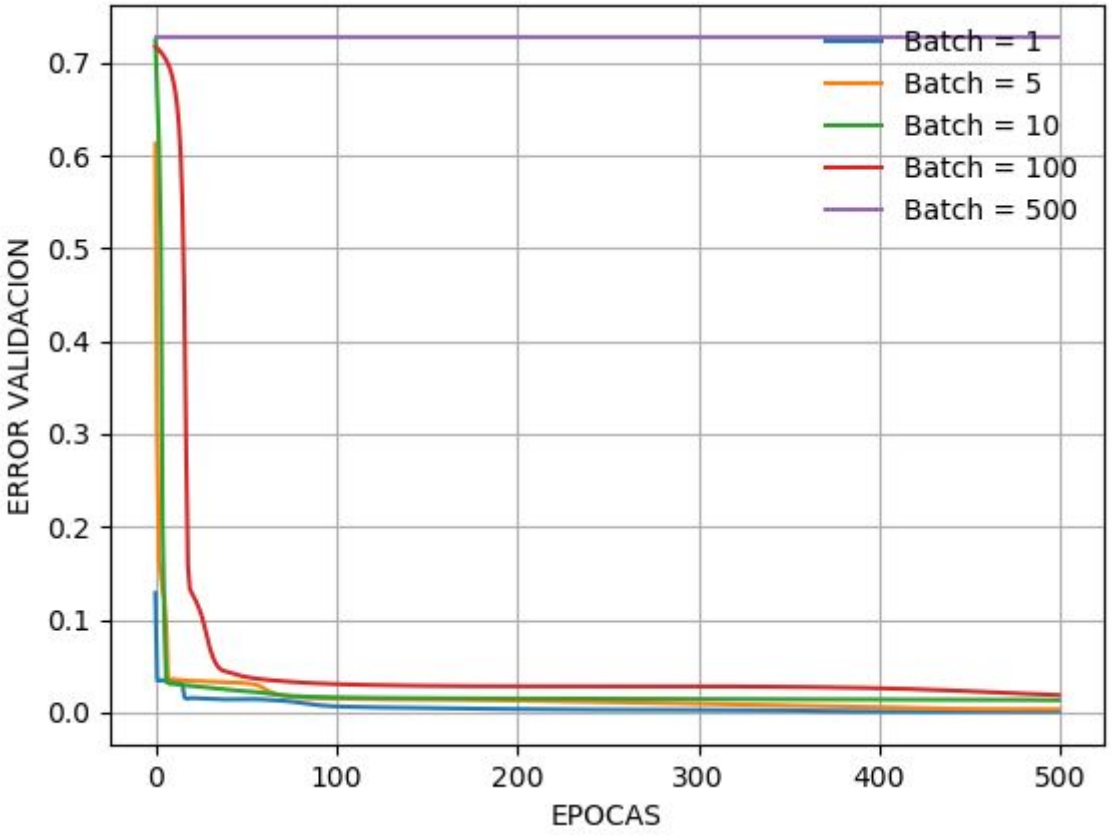

Gráfico 44: Ejercicio 1 - Variación batch - Error validación

Performance en etapa de testing para tamaño de batch:

Tamaño de batch	Performance de Testing
1	98%
5	74%
10	60%
100	52%
500	04%

Observaciones del experimento:

Se observa que los peores resultados se encuentran tomando batch = 100 y 500, para testing, sin embargo el error de entrenamiento y validación para batch = 100, no es malo, pero si para batch=500
Para batch de tamaño 10, los errores de entrenamiento y validación tampoco son malos, pero su performance en testing sí lo es (60%).
Los mejores resultados en entrenamiento y validación son para batch con valor 1 y 5 al igual que en el ejercicio 1. En entrenamiento tienen valores de error muy cercanos, en validación, batch=5, parece tener mejores resultados, pero en testing batch=1, tiene un 98% de efectividad.