Práctica de laboratorio 4: Redes neuronales recurrentes

Castro Elvira D. [2022710039], Pineda Rugerio N. [20222710240], Castañón Hernández V. J. [2022710020], Galicia Cocoletzi N. [2022710234], Sánchez Zanjuampa M. A. [2022710029], y Nava Mendez E. U. [2021710144].

Profesor: Lauro Reyes Cocoletzi

***Resumen*—En la presente práctica se utilizan redes neuronales recurrentes (RNN) para analizar y modelar una serie temporal de manchas solares para predecir valores futuros. Se llevó acabo la experimentanción con diveros valores de hiperparámetros para optimizar el rendimiento del modelo. Se implementó un modelo RNN con una capa SimpleRNN y una capa densa utilizando Keras de python, entrenándolo con el conjunto de datos "Monthly Mean Total Sunspot Number", que ofrece una serie temporal histórica de las manchas solares registradas. Posteriormente, se realizaron predicciones en los conjuntos de entrenamiento y validación para evaluar el modelo. Se evaluó la precisión del modelo mediante el cálculo del error cuadrático medio (RMSE), una métrica ampliamente reconocida para evaluar la exactitud de las predicciones. Finalmente, se realizó la visualización de los resultados para evaluar y analizar el rendimiento del modelo en la predicción de la serie temporal de manchas solares con los datos reales.**

***Palabras clave*—Redes neuronales recurrentes, función de activación, pesos, sesgos, predicciones**

I. Introducción

I-A. Marco teórico

I-A1. ¿Qué es una red neuronal recurrente?:

Las redes neuronales recurrentes se tratan de un modelo de aprendizaje profundo que está entrenado para procesar y convertir una entrada de datos secuencial en una salida de datos en secuencia específica. Para el caso de las redes neuronales recurrentes, las salidas dependen de los elementos anteriores dentro de la secuencia.

Dentro de las aplicaciones de las RNN podemos encontrar lo siguiente:

- Traducción automática. Las redes neuronales recurrentes se utilizan para traducir texto de un idioma a otro.
- Reconocimiento de voz. Las redes neuronales recurrentes se utilizan para convertir el habla en texto.
- Generación de texto. Las redes neuronales recurrentes se utilizan para generar texto, como subtítulos de imágenes o artículos de noticias.
- Análisis del sentimiento. Las redes neuronales recurrentes se utilizan para determinar el sentimiento de un texto, como positivo, negativo o neutral.
- Predicción de series temporales. Las redes neuronales recurrentes se utilizan para predecir valores futuros en una serie temporal, como el precio de las acciones o el clima.

I-A2. Estructura básica de una RNN:

Las redes neuronales recurrentes contienen la siguiente estructura básica, en función del tipo de RNN que se está usando

- Capas de entradas. Recibe la secuencia de datos de entrada
- Capas ocultas. Contienen las neuronas recurrentes de la red. Estas neuronas tienen conexiones recurrentes, lo que significa que pueden recibir información de sí mismas además de la entrada y la capa anterior.
- Capa de salida. Produce la secuencia de salida deseada.

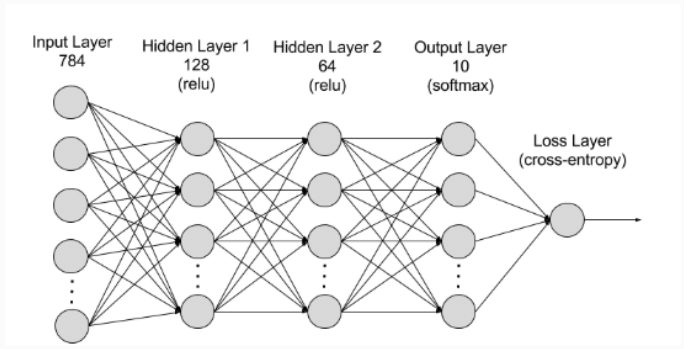


Figura 1: Estructura básica de una red neuronal

Las neuronas recurrentes en las capas ocultas son el núcleo de una RNN. Estas neuronas realizan una función no lineal sobre la suma de sus entradas y almacenan el resultado en un estado oculto. El estado oculto se utiliza luego como entrada para la siguiente neurona recurrente, así como para la capa de salida.

I-A3. Optimizador Adam:

Adam, o **Adaptive Moment Estimatio**n, es un algoritmo de optimización que combina las ventajas de los algoritmos RMSprop (adapta la tasa de aprendizaje para cada parámetro basándose en una media exponencialmente decreciente de los gradientes al cuadrado pasados, buen rendimiento en diversos ámbitos) y Momentum (rápida convergencia, mejor estabilidad y la habilidad de superar los minimos locales) para mejorar el proceso de aprendizaje de un modelo.

Al igual que Momentum, Adam utiliza una estimación del momento y de la magnitud de los gradientes anteriores para actualizar los parámetros del modelo en cada iteración.

Sin embargo, en lugar de utilizar una tasa de aprendizaje constante para todos los parámetros, Adam **adapta la tasa de aprendizaje de cada parámetro individualmente en función de su estimación del momento y de la magnitud del gradiente**. Esto permite que el modelo se ajuste de manera más eficiente y efectiva a los datos de entrenamiento, lo que puede llevar a una mayor precisión de la predicción en comparación con otros métodos de optimización.

I-A4. Base de datos:

- Base de datos "Monthly Mean Total Sunspot Number". Se trata de una base de datos que se recopila el aproximado de manchas solares de forma mensual desde 1749 hasta julio de 2018

I-B. Objetivos

- Comprender la estructura de una red neuronal recurrente, así como el proceso que esta lleva a cabo para obtener una predicción
- Analizar la importancia de selección de los hiperparámetros para el entrenamiento de la red

I-C. Aporte

- El principal aporte de este trabajo se encuentra en la aplicación de redes neuronales recurrentes para predecir valores acerca de una serie temporal de manchas solares.
- Se buscan las mejores combinaciones de hiperparámetros del modelo, lo que permite mejorar la precisión de las predicciones.
- Además, se pueden analizar las variaciones en el número de manchas solares, que tienen implicaciones importantes en la comprensión del clima espacial.
- Los resultados obtenidos demuestran que la correcta combinación de parámetros permite optimizar el rendimiento del modelo, y precisamente lograr errores menores en la predicción.

II. Desarrollo

1. **Carga y división de los datos.**
   - Se carga la base datos mediante la url proporcionada para escalar los datos y dividirlos para entrenamiento y prueba.
   - Se divide el conjunto de datos en un 80 % para el entrenamiento y el 20 % para prueba.
2. **Preparar la entrada en el formato Keras requerido.**
   - Se utiliza la función *get_XY*, la cual toma los datos y los divide en segmentos que tienen una longitud *time_steps*, estos se pasan como entradas al modelo RNR. Y los puntos "que son desplzados un paso adelante"son considerados los objetivos o targets.
3. **Revisar modelo RNN y entrenarlo.**
   - Se crea un modelo RNR de python, que lo contiene la función *create_RNN*, se le pasa el número de unidades ocultas, el número de unidades densas, los datos de entrada, y la función de activación.
   - Se entrena el modelo utilizando Keras.
4. **Realizar las predicciones en los conjuntos de entrenamiento y prueba e imprimir el error cuadrático medio en ambos conjuntos.**
   - Se hacen las predicciones con el modelo.
   - Se calcula el error cuadrático medio RMSE para entrenamiento y prueba, y se imprime el resultado.
5. **Ver los resultados.**
   - Para ello se hace un plot, para observar la gráfica de lo valores reales y los valores predichos.

III. Resultados

IV. Código 1 Predicción de series temporales

Se realizaron diferentes ajustes a los hiperparámetros que permiten la creación del modelo de la red RNN, para este caso se consideraron los valores de hidden_units como 3 y dense_units como 1:

Tabla I: **Hiperparametros**

Prueba	Epochs	Batch Size	time_steps	Activation
1	20	1	12	tanh
2	30	1	10	tanh
3	35	1	10	tanh
4	25	1	10	tanh
5	35	1	10	linear
6	25	1	10	linear
7	100	16	20	tanh
8	100	16	20	tanh & relu
9	100	8	10	tanh
10	100	16	20	tanh
11	60	2	14	linear

Conn lo anterior se obtuvieron los siguientes resultados del error cuadrático medio.

Tabla II: **Error cuadratico medio**

Prueba	Entrenamiento RMSE	Test RMSE
1	0.060	0.102
2	0.061	0.087
3	0.062	0.103
4	0.063	0.084
5	0.060	0.079
6	0.066	0.081
7	0.055	0.090
8	0.058	0.085
9	0.060	0.092
10	0.053	0.088
11	0.057	0.069

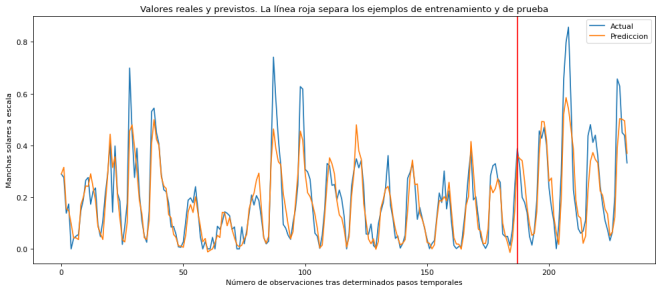
IV-A. Graficas de valores reales vs. valores predichos.

A continuación, se presentan gráficas que comparan los valores reales (línea azul) y los valores predichos (línea anaranjada) a lo largo de una serie de observaciones temporales.

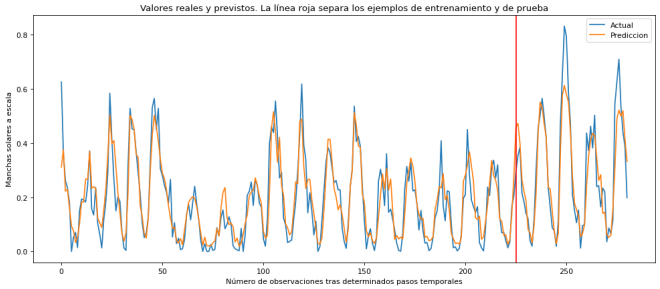
La línea roja divide los ejemplos de entrenamiento y prueba, lo que significa que los datos a la izquierda de esa línea se usaron para entrenar el modelo, mientras que los datos a la derecha se utilizaron para comprobar el desempeño del mismo.

La gráfica muestra cómo el modelo predice los valores a lo largo del tiempo, comparándolos con los valores reales observados. Esto permite evaluar qué tan bien el modelo está realizando sus predicciones en comparación con los datos reales.

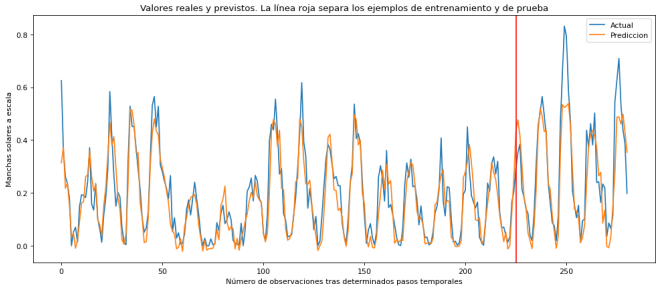
IV-A1. Prueba 1:



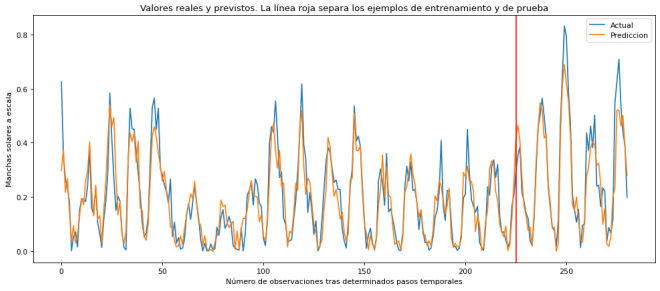
IV-A2. Prueba 2:



IV-A3. Prueba 3:

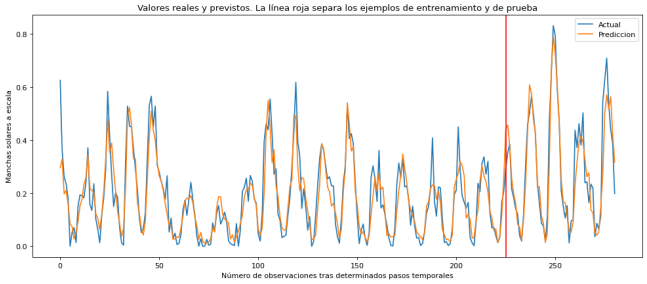


IV-A4. Prueba 4:

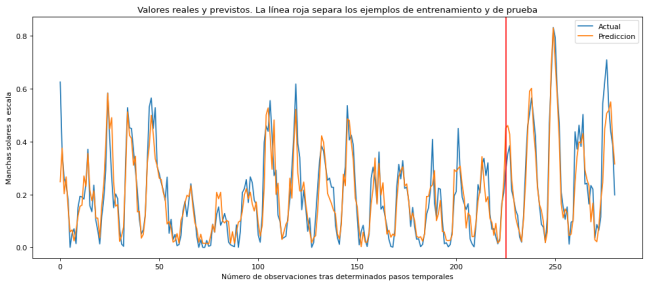


IV-A5. Prueba 5:

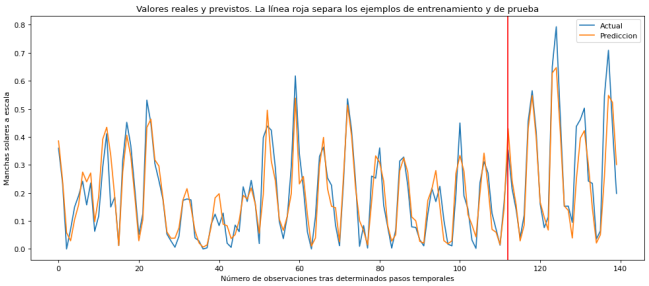
[H]



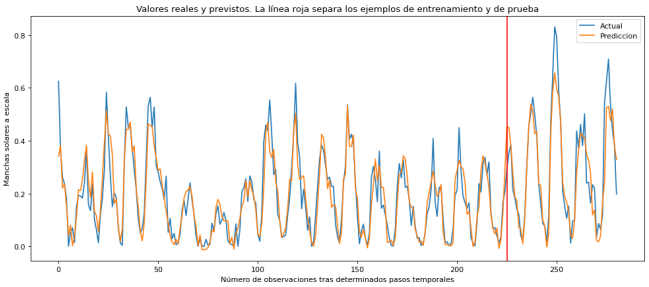
IV-A6. Prueba 6:



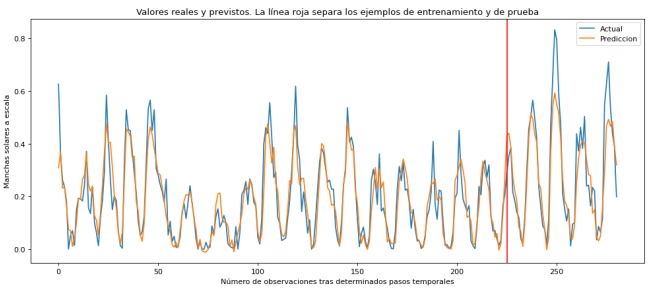
IV-A7. Prueba 7:



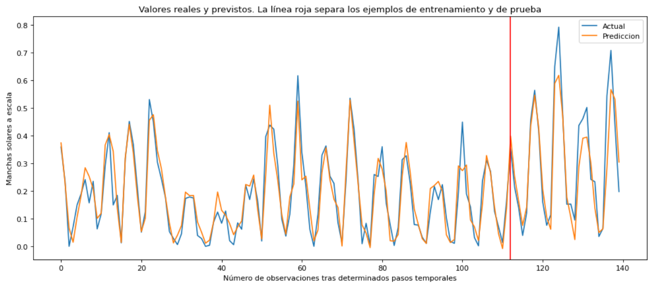
IV-A8. Prueba 8:



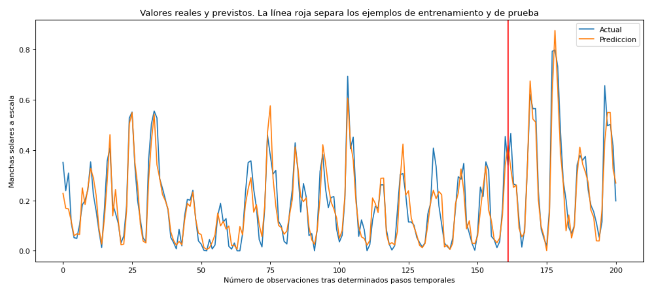
IV-A9. Prueba 9:



IV-A10. Prueba 10:

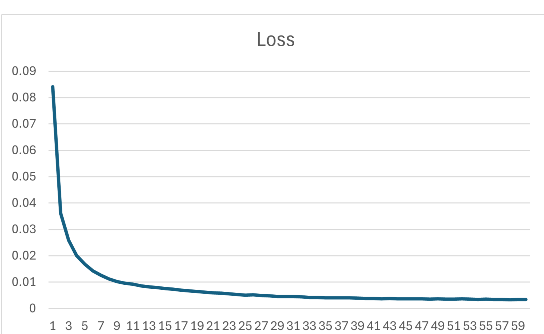


IV-A11. Prueba 11:



Al revisar los resultados de las pruebas, se observa una tendencia que sugiere una relación entre la elección de la función de activación y el tamaño del lote (Batch size) en el rendimiento del modelo, puesto que se observo una disminución del error cuadrático medio (RMSE) al comparar los resultados de la primera prueba con la última, donde se aplicaron valores mas elevados en los hiperparámetros y se implementó la función de activación lineal. Sin embargo, aunque los resultados parecen prometedores en términos de la reducción del RMSE en el conjunto de entrenamiento, surge una señal de alerta al examinar la evolución de la función de pérdida a lo largo del entrenamiento. La gráfica muestra que la función de pérdida se estabiliza mucho antes de que se completen todas las épocas, sugiriendo que el modelo podría estar alcanzando su capacidad máxima de aprendizaje lo que resulta en un sobreajuste. Esta sospecha de sobreajuste también se hace notar al observar la diferencia entre los valores de RMSE de entrenamiento y prueba. Mientras que el RMSE de entrenamiento es bajo (0.057), indicando un buen ajuste a los datos utilizados para entrenar el modelo, el RMSE de prueba es más elevado (0.069), lo que sugiere una pérdida de capacidad de generalización en datos nuevos.

IV-A12. Loss:



V. CÓDIGO 2. BASADO EN CARACTERES

El siguiente código consiste en una predicción realizada por caracteres, es un algoritmo parecido a una predicción por palabras, pero este tiene la ventaja de que se entrena más rápido.

V-A. Prueba 1

Parámetros utilizados:
Número de iteraciones = 25
Número de épocas por iteración = 1
Número de predicciones por época = 100
SEQLEN = 10

Tabla III: **Prueba 1**

Iteración	Loss	Semilla	Resultado
1	2.5842	my poor h	*my poor har se the gat ing the gat ing the gat ing the gat ing the gat ing the gat ing the gat ing the gat in*
13	1.5271	rried on,	*rried on, and the mock turtle she was she was she was she was she was she was she was she was she was*
25	1.3837	* *	* *

Acá podemos apreciar que la última iteración nos dió "malos resultados", esto debido a que anteriormente no se habían presentado esos caracteres, y como en la semilla se encuentran 2 caracteres consecutivos y el modelo no reconoció esos mismos caracteres anteriormente, el resultado lo interpretó de esa manera, llenándolo de los mismos caracteres. En cambio, si tomamos como referencia la penúltima iteración, nos da el siguiente resultado final (100 caracteres):

s sort of the foundation of the foundation of the foundation of the foundation of the foundation of the founda

V-B. Prueba 2

Parámetros utilizados:
Número de iteraciones = 25
Número de épocas por iteración = 1
Número de predicciones por época = 100
SEQLEN = 12

Tabla IV: **Prueba 2**

Iteracion	Loss	Semilla	Resultado
1	2.5780	, but it was	*, but it was in the the the the the the the the the the the the the the the the the the the the the the the the the the the*
13	1.5164	ct gutenberg	*ct gutenberg-tm electronic work in a the gryphon was the mouse in a the dormouse the morst it was the mouse in a*
25	1.3789	crab, he was	*crab, he was the sort of the sort of the sort of the sort of the sort of the sort of the sort of the sort of the*

V-C. Prueba 3

Parámetros utilizados:
Número de iteraciones = 25
Número de épocas por iteración = 2
Número de predicciones por época = 100
SEQLEN = 15

Tabla V: **Prueba 3**

Iteracion	Loss	Semilla	Resultado
1	2.0781	so far, said t	*so far, said the groject gutenberg-the said the groject gutenberg-the said the groject gutenberg-the said the groj*
13	1.3753	see, because so	*see, because so mock turtle she was no the could not look a little with a shatch her face all come of the could not*
25	1.2894	top of her head	*top of her head for a little conther the gryphon in the same with a shoulder and the cat heard i wonder the course,*

En esta prueba podemos apreciar que la pérdida de información representada con el parámetro de Loss es muy baja, esto gracias a que en cada iteración se fueron entrenando 2 épocas, donde la segunda siempre representaba un Loss menos que el de la primera. Además también podemos apreciar que la salida final de la iteración 25 tiene una mejor relación de palabras y llega a tener algún sentido.

V-D. Prueba 4

Parámetros utilizados:
Número de iteraciones = 10
Número de épocas por iteración = 3
Número de predicciones por época = 150
SEQLEN = 20

Tabla VI: **Prueba 4**

Iteracion	Loss	Semilla	Resultado
1	1.9468	ermanent future for	*ermanent future for said the king the page and the said the downot on the came the down the king the page and the said the downot on the came the down the king the page a*
5	1.4826	ctly what they said.	*ctly what they said. and she was a little said to herself, and she said to herself, and she said to herself, and she said to herself, and she said to herself, and she sai*
10	1.3526	and perhaps as this	*and perhaps as this distributed in a long sure a confused and replied to herself a this eare the poor little poor alice as she had not gon it was not gone a great down a*

En esta otra prueba decidimos aumentar las épocas por iteración y reducir las iteraciones, y podemos apreciar que desde la primer iteración, al finalizar la tercer época, ya nos daba una pérdida bastante baja, además si tomamos en cuenta que tomamos más de 10 caracteres anteriores para predecir el siguiente, y analizamos el texto de salida, podemos apreciar que desde ahí ya tenemos mejores resultados que en las pruebas anteriores en la primer iteración.

V-E. Prueba 5

Parámetros utilizados:
Número de iteraciones = 40
Número de épocas por iteración = 5
Número de predicciones por época = 200
SEQLEN = 30

Tabla VII: **Prueba 5** (Parte 1)

Iteracion	Loss	Semilla	Resultado
1	1.5772	ened the door and went in. the	*ened the door and went in. the dinted the really see to the dinted the really see to the dinted the really see to the dinted the really see to the dinted the really see to the dinted the really see to*
10	1.2843	he moral of that is–be what y	*he moral of that is–be what you to speak of the fan the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact of the fact*

Tabla VIII: **Prueba 5** (Parte 2)

Iteracion	Loss	Semilla	Resultado
20	1.2268	was out of sight before the of	*was out of sight before the offended to alice took to herself, and they had to states the trees began the tone of them as they were alice and the court all the terms of the three gardeners, and she was a project gutenberg-tm work*
30	1.2105	rly as large as himself, and t	*rly as large as himself, and the party with a she said to herself in a thing as it was a long as it was a long as she could not thinking the looked at the white rabbit a little thing it any any part of the great of the same thing*
40	1.2051	s would all wash off in the se	*s would all wash off in the sea–now you make out a long to the whole party with the white rabbit asking of the soldiers and the white rabbit asking of the soldiers and the white rabbit asking of the soldiers and the white rabbit*

Como podemos apreciar, en esta prueba extrema, los resultados finales del texto tienen mucho más sentido que las pruebas anteriores, aunque el programa completo tardó alrededor de 30 minutos en terminar, obtuvo mejores resultados tanto en la pérdida como en el resultado final. Aunque es importante mencionar que se nota una convergencia en el modelo en cuanto a la pérdida, ya que en la mitad de las épocas era donde por promedio, tenía menor pérdida, llegando a tener un valor de Loss del 1,1972 en la iteración 35, época 5.

Pero debido al gran número de resultados que nos da cada iteración, decidimos capturar los resultados de cada 10 iteraciones en este caso, en donde el valor de Loss de cada iteración era el de la última época de esa iteración.

V-F. Prueba 6

En la primera iteración las predicciones parecen no tener un buen resultado pues al final solo imprime de manera cíclica la palabra 'the' y el valor de la perdida es alto. Sin embargo se puede mejorar una clara mejora en la iteración numero 15 donde la perdida se redujo considerablemente y las palabras formadas tienen mas sentido. En la ultima iteración la perdida mejoro un poco y la predicción de caracteres también, ero comienza a repetir un conjunto de palabras.

Parámetros utilizados:
Número de iteraciones = 30
Número de épocas por iteración = 1
Número de predicciones por época = 100
SEQLEN = 30

Tabla IX: **Prueba 6**

Iteracion	Loss	Semilla	Resultado
1	2.5743	sneeze, were the cook, and a	*sneeze, were the cook, and a d aller in the the the the the the the the the the the the the the the the the the the the the t*
15	1.4630	w. so you see, miss, were doin	*w. so you see, miss, were doing the caterpillar she thought alice was not the did not a little she had not got the caterpillar she*
30	1.3546	lice,) well, i hardly know–no	*lice,) well, i hardly know–now alice to the mock turtle was a very a little with the duchess to the mock turtle was a very a litt*

V-G. Prueba 7

Esta es la prueba con la semilla mas corta de la practica pues solo se han usado 6 caracteres. Inicialmente la perdida de la primera iteración es algo alta y parece disminuir a lo largo de 42 iteraciones. En cuanto a los caracteres predichos, estos crean solo algunas palabras las cuales se repiten mucho a lo largo del texto generado en los resultados. Aunque el resultado de la ultima iteración mejora mucho, tomando en cuenta el numero de iteraciones y que la mejora en LOSS se hace cada vez mas pequeña, esta prueba es quizá una de las peores.

Parámetros utilizados:
Número de iteraciones = 42
Número de épocas por iteración = 1
Número de predicciones por época = 100
SEQLEN = 6

Tabla X: **Prueba 7**

Iteracion	Loss	Semilla	Resultado
1	2.5683	, and	*, and the mere the mere the mere the mere the mere re the mere the mere the mere the mere the mere the mere the mere t*
21	1.4282	to her	*to herself, i think the little thing the some of the white rabbit the white rabbit the white rabbit the wh*
42	1.3221	es adv	*es adventures of the caterpillar. alice to herself a gutting of the caterpillar. alice to herself a guttin*

V-H. Prueba 8

En esta prueba se duplicó el numero de muestras de SEQLEN a 30 caracteres y los demás parámetros se completaron con los de la prueba numero 3. La perdida no vario mucho con la prueba numero 3 pero en este caso la repetición de palabras se ha reducido mucho como se puede ver en la iteración 30. Aun así Existen algunos caracteres que no se redijeron correctamente en las iteraciones anteriores. Como nota extra, el programa no parece tener información suficiente como para

cerrar el signo de paréntesis que se abrió en la semilla.

Parámetros utilizados:
Número de iteraciones = 25
Número de épocas por iteración = 2
Número de predicciones por época = 100
SEQLEN = 30

Tabla XI: **Prueba 8**

Iteracion	Loss	Semilla	Resultado
1	2.0725	d to stand on their hands and	*d to stand on their hands and the gat in whe dore the gat in whe dore the gat in whe dore the gat in whe dore*
13	1.3842	and the little golden key was	*and the little golden key was all the restanted alice had not some was a little she said to herself, and she was not got to the b*
25	1.2903	wimming about here, o mouse! (	*wimming about here, o mouse! (which was so ot end she could have the way of the white rabbit to the large rather she was a little*

V-I. Prueba 9

Para esta prueba se tomo como referencia la prueba numero 4, de la cual solo se aumento el numero de iteraciones a 25. Al aumentar las iteraciones el valor de perdida se logro reducir hasta 1.2695, además de que la predicción de caracteres resulto en la formación de mas mas palabras correctas, aunque también llega a caer ocasionalmente en un ciclo de repeteción de palabras.

Parámetros utilizados:
Número de iteraciones = 25
Número de épocas por iteración = 3
Número de predicciones por época = 150
SEQLEN = 20

Tabla XII: **Prueba 9**

Iteracion	Loss	Semilla	Resultado
1	1.9389	did not venture to	*did not venture to the dound the mous the mad the mous the mad the mous the mad the mous the mad the mous the mad the mous the mad the*
10	1.3527	and then quietly ma	*and then quietly made of the wasch i dont know when it was a long as it was a little been what i dont know when it was a long as it was a little been what i dont know wh*
20	1.2695	e, however, she agai	*e, however, she again, and the party of the poor all the same and down and she said the party of the poor all the same and down and she said the party of the poor all the*

V-J. Prueba 10

En este caso se han repetido algunos parámetros de la prueba 5 a excepción del numero de épocas por cada iteración cual redujimos a solo una iteración. Como resultado, la perdida es mayor en comparación con la prueba 5, por otro lado el tiempo de ejecución en este caso se redujo considerablemente, aunque el sentido de las palabras que se formaron con los caracteres predichos no tuvieron mucha mejora y tampoco empeoraron. En la iteración. Es de destacar que algunas palabras o grupos de palabras se repitieron constantemente, tal es el caso de la iteración 30 donde se puede apreciar mas y sorprendentemente en la ultima iteración la predicción de caracteres fue mucho mejor, pero no así el sentido de las palabras.

Parámetros utilizados:
Número de iteraciones = 40
Número de épocas por iteración = 1
Número de predicciones por época = 200
SEQLEN = 30

Tabla XIII: **Prueba 10** (Parte 1)

Iteracion	Loss	Semilla	Resultado
1	2.5827	asis, looking hard at alice as	*asis, looking hard at alice as the wast the wast the the there the said the sald the the there the said the sald the the there the said the sald the the there the said the sald the the there the said the sald the the there the said the sald the the there the sai*
10	1.5883	mouse did not notice this ques	*mouse did not notice this question a great down the dormouse of the stopling to the dormouse of the stopling to the dormouse of the stopling to the dormouse of the stopling to the dormouse of the stopling to the dormouse of the stopling to the dormouse of the st*
20	1.4236	nd just as id taken the highes	*nd just as id taken the highess was a little began in a long on the three was a little began in a long on the three was a little began in a long on the three was a little began in a long on the three was a little began in a long o*
40	1.3169	him declare, you have baked me	*him declare, you have baked me seep to say when they were the work it was not a little way up and here the hatter were shall she was not a learnd of course, the hatter were shall she was not a learnd of course, the hatter were sha*

VI. Conclusiones

Los objetivos de la práctica que consisten en comprender la estructura de una RNN así como el proceso que se lleva a cabo para obtener una predicción se cumplieron, tras realizar diversas pruebas tanto en el programa 1 (predicción de series temporales) como en el programa 2 (predicciones de caracteres) se pudo notar la importancia del segundo objetivo: analizar la importancia de la selección de los hiperparámetros para el entrenamiento de la RNN.

En el programa 1 se notó que entre mayor número de épocas se le asignaba al entrenamiento de la RNN, mejor era el resultado que se reflejaba en el error cuadrático medio (RMSE) de los conjuntos de entrenamiento y prueba, aunque esto tendía a variar con los tipos de función de activación colocadas en la red; por lo que el RMSE de ambos conjuntos son similares quiere decir que el modelo está siendo entrenado correctamente, ya que no tiende a un sobreajuste ni a un subajuste; por otro lado, si los valores del RMSE son muy diferentes entre sí puede significar que nuestro modelo puede tener problemas de rendimiento y/o generalización.

Por otro lado en el programa 2 se puede notar una mejora en la función de pérdida del algoritmo cuando se aumenta el número de épocas por iteración así como la cantidad de caracteres utilizados para predecir el siguiente valor; mientras que en caso de disminuir el número de caracteres utilizados para realizar la predicción los valores de la función de pérdida tienen a empeorar y esto se nota en la calidad de las predicciones, que comienzan a repetir caracteres llegando a ciclarse.

Referencias

[1] ¿Qué son las RNN? -AWS https://aws.amazon.com/es/what-is/recurrent-neural-network/#:~:text=Una%20red%20neuronal%20recurrente%20(RNN,salida%20de%20datos%20secuencial%20espec%C3%ADfica.. Recuperado el 12 de abril de 2024