Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Uso de redes LSTM para el diagnóstico de enfermedades neurodegenerativas

Use of LSTM networks for the diagnosis of neurodegenerative diseases

Diego Hugo Hamilton López

La Laguna, 14 de junio de 2022

D. **Patricio García Báez**, con N.I.F. 43356987D profesor Contratado Doctor adscrito al Departamento de Ingienería Informática y de Sistemas de la Universidad de La Laguna, como tutor

D. **Pablo Carmelo Fernández López**, con N.I.F. 42868276W profesor Contratado Doctor adscrito al Departamento de Informática y Sistemas de la Universidad de Las Palmas de Gran Canaria, como cotutor

C E R T I F I C A (N)

Que la presente memoria titulada:

”*Uso de redes LSTM para el diagnóstico de enfermedades neurodegenerativas*”

ha sido realizada bajo su dirección por D. **Diego Hugo Hamilton López**, con N.I.F. 54111147J.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 14 de junio de 2022

Agradecimientos

Quiero agradecer a todas las personas que me han acompañado a lo largo de mi formación. También a mis padres por siempre brindarme su apoyo y confianza durante todo el tiempo que he estado sacando la carrera. A mis amigos, por haber hecho de la carrera una experiencia más amena e inolvidable. A mis tutores y profesores que he tenido durante las distintas asignaturas de la carrera, de los cuales he podido aprender tanto y me han brindado todo su apoyo. Y por último a mi tutor del TFG, por el esfuerzo que ha hecho por ayudarme a sacar este proyecto, y la ayuda que me ha brindado para encaminarme por él y haber hecho que sepa como desarrollarlo.

Licencia

Resumen

Actualmente las enfermedades neurodegenerativas son un gran problema debido a su dificultad de diagnóstico y la necesidad de pruebas complejas. En este trabajo se estudiará si el uso de la red LSTM mejorará el diagnóstico de estas enfermedades por su mayor capacidad para analizar datos longitudinales. Se comparará los resultados obtenidos por la LSTM con una red convencional que no haga uso de datos longitudinales, para concluir el beneficio de usar este tipo de datos y si se podría llegar a diagnosticar estas enfermedades con un conjunto de pruebas sencillas realizadas en distintas citas con el paciente.

Palabras clave: Red Neuronal Artificial, LSTM, diagnóstico, datos longitudinales, demencia, deterioro cognitivo.

Abstract

Neurodegenerative diseases are currently a major problem due to their difficulty in diagnosis and the need for complex tests. In this work, we will study whether the use of the LSTM network will improve the diagnosis of these diseases due to its greater capacity to analyze longitudinal data. The results obtained by the LSTM will be compared with a conventional network that does not use longitudinal data, to conclude the benefit of using this type of data and whether these diseases could be diagnosed with a set of simple tests performed at different appointments with the patient.

Keywords: Artificial Neural Network, LSTM, diagnosis, longitudinal data, dementia, cognitive impairment.

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

Las enfermedades neurodegenerativas son aquellas provocadas por una degeneración progresiva de la estructura y función del sistema nervioso central o periférico, entre las más comunes y conocidas están el Alzheimer y el Parkinson. Los casos de estas enfermedades están en crecimiento. En España en 2017 había ya un total de 441.912 personas con demencia diagnosticada y, ya que están asociadas al aumento de esperanza de vida de los humanos [2], se prevé que siga teniendo este aumento. Esto está suponiendo un gran impacto en la salud pública debido a que, en sus etapas tardías, las enfermedades neurodegenerativas son mucho más difíciles de tratar y tienen un costo más elevado en sus tratamientos.

El procedimiento médico para detectar estas enfermedades se basa principalmente en el historial clínico y pruebas cognitivas sencilla. Aun así, estas enfermedades se manifiestan de formas que todavía no se terminan de comprender y hace que estas pruebas tengan una baja precisión a la hora del diagnóstico, especialmente en la atención primaria. Para su correcto diagnóstico es necesario el uso de pruebas mucho más costosas y menos accesibles, como son las resonancias magnéticas. Por este problema se abre pie a la utilización de redes neuronales para ayudar en el diagnóstico, aprovechando su habilidad para generar relaciones entre variables y su capacidad de aprendizaje.

Con la utilización de la redes neuronales se pretende mejorar la detección temprana del deterioro cognitivo para poder aplicar un tratamiento lo antes posible y así evitar que este deterioro se convierta en una enfermedad más avanzada, como es el Alzheimer o el Parkinson, dando así una mayor esperanza de recuperación a los pacientes.

1.1. Objetivos

1.1.1. Objetivo general del proyecto

El objetivo de este TFG es desarrollar una red neuronal recurrente la cual sea capaz de predecir si una persona es sana, tiene trastorno cognitivo leve o padece demencia. Se hará uso de los datos de dos o más citas médicas en distintas fechas, en las que han sido realizadas las mismas pruebas. Estas deben de ser sencillas, poco costosas y accesibles para cualquier persona, de forma que el sistema desarrollado se pueda utilizar en atención primaria.

Una vez desarrollada esta red neuronal se comparará con una red neuronal convencional, para saber si la memoria de las redes neuronales recurrentes ayudan a hacer este diagnóstico. Al final se concluirá si llevar un seguimiento del avance cognitivo de las personas es más eficaz, a la hora de diagnosticar, que simplemente realizar una única prueba en un momento determinado.

1.1.2. Objetivo específicos del proyecto

Los objetivos específicos del proyecto son los necesarios para poder realizar el objetivo general del proyecto de forma efectiva, y se pueden dividir en 5 distintos:

1. La realización de tratamiento previo a los datos, haciendo una división, balanceo y normalización mediante el uso de estadísticos para tener la cantidad y criterios clínicos más adecuados para ser usados como entrada de entrenamiento, test y validación de la red neuronal.

2. Implementar la red neuronal Long Short-Term Memory (LSTM) [3] junto algún método de validación, además de haciendo uso de distintos métodos de evaluación de resultados como, gráficas de pérdidas, informes de clasificación, matrices de confusión y gráficas ROCs.

3. Una vez implementada una primera red LSTM, buscar la mejor configuración posible de la red para así quedarnos el conjunto de parámetros que dé mejor resultado de validación.

4. Crear una red convencional equivalente, haciendo los mismo procesos hechos a la LSTM, pero que use información de una sola cita y comparar los resultados de esta nueva red con la anterior red LSTM, llegando a una conclusión dependiendo de los datos obtenidos.

5. Realizar un estudio y representación gráfica visual de los resultados obtenidos, de forma coherente que valide la conclusión obtenida.

1.2. Estado del arte

Este tema cuenta con amplios antecedentes, en los últimos años se han hecho muchas investigaciones con el uso de redes neuronales para diagnosticar enfermedades, y el diagnóstico de enfermedades neurodegenerativas es uno de los más tratados debido a su dificultad. Ya existen varios proyectos con redes neuronales capaces de predecir esto con más de un 99 por ciento de acierto, lo que estas redes son redes de tipo convolucional y utilizan pruebas de resonancia y otras que son mucho más costosas que las pruebas con las que se va a realizar este proyecto. Hay muchos ejemplos de tipos de redes para este tema como son las redes neuronales híbridas y los ensambles neuronales para asistir a la detección de DCL [4], o redes neuronales convencionales [5], pero no se encuentran estudios haciendo uso de LSTM.

El uso de LSTM para este diagnóstico es más novedoso y se pueden ver en TFG de otros compañeros [6] [7], en las que me apoyaré en parte, pero este TFG tiene un enfoque distinto al de los otros compañeros ya que no trataremos de predecir como va a evolucionar el paciente, sino de mejorar el acierto de la predicción haciendo uso de la memoria a largo plazo que posee la red LSTM.

El mayor problema actual de la creación de estas redes es la falta de información, porque aunque existen varias bases de datos, como es el caso de la cual se va a utilizar en este proyecto ya mencionada, no hay mucha información debido a que son datos privados de pacientes, y también el costo de estas pruebas y la necesidad de alguien que esté introduciendo a mano estas pruebas en las bases de datos lo complica. Aun así, cada día va aumento la cantidad de información y se va concienciando de la importancia de almacenar los resultados de estas pruebas.

Capítulo 2

Método y herramientas utilizadas

En el capítulo anterior se vio la introducción al proyecto, en este se verá las herramientas software utilizadas, la estructura de la red LSTM y la evaluación de resultados.

2.1. Herramientas software

Las herramientas software utilizadas durante el desarrollo del TFG y las cuales permitieron los resultados del mismo son las siguientes:

- **Entorno de desarrollo:**

  El entorno de desarrollo para realizar el programa ha sido Jupyter-Notebook, dentro de los enviroments de Anaconda [8], que es una distribución de Python para computación científica. Se escogió esta herramienta debido a lo simple que era la gestión e implementación de paquetes, gestionando sus versiones y dependencias correctamente. También permitía usar los recursos del ordenador local, en vez de utilizar servicios de recursos en la nube.

- **Librerías utilizadas:** Las librerías utilizadas más importantes han sido:

  - Pandas [9], librería de Python como extensión de Numpy para la manipulación y análisis de datos, la cual es la más utilizada en el ámbito de Machine Learning.
  - Keras/Tensorflow [10], para el desarrollo de la red neuronal se escogió Keras, que es la API de alto nivel de Tensorflow, la cual permite una rápida creación de prototipos de redes neuronales.
  - Scikit-learn [11], para el uso de algoritmos de clasificación, la creación de matrices de confusión y otros métodos para comprobar los resultados de la red como el accuracy.

- Matplotlib [12], como librería para representar las gráficas obtenidas.

2.2. LSTM

LSTM es una red neuronal de tipo recurrente. Estas redes son capaces de analizar una secuencia de datos de entrada y producir una predicción, por lo que se adapta bien para el estudio de datos longitudinales. La red LSTM es capaz de "recordar" un dato importante y preservarlo, por tanto puede tener una memoria a corto o largo plazo.

2.2.1. Celdas

La red LSTM está formada por celdas, como se ve en la figura 2.1, las cuales son como las de las redes recurrentes básicas pero tienen una entrada y una salida adicional (c_{t-1} y c_t). En esta celda se pueden añadir o mover datos que no queremos que queden en la memoria de red.

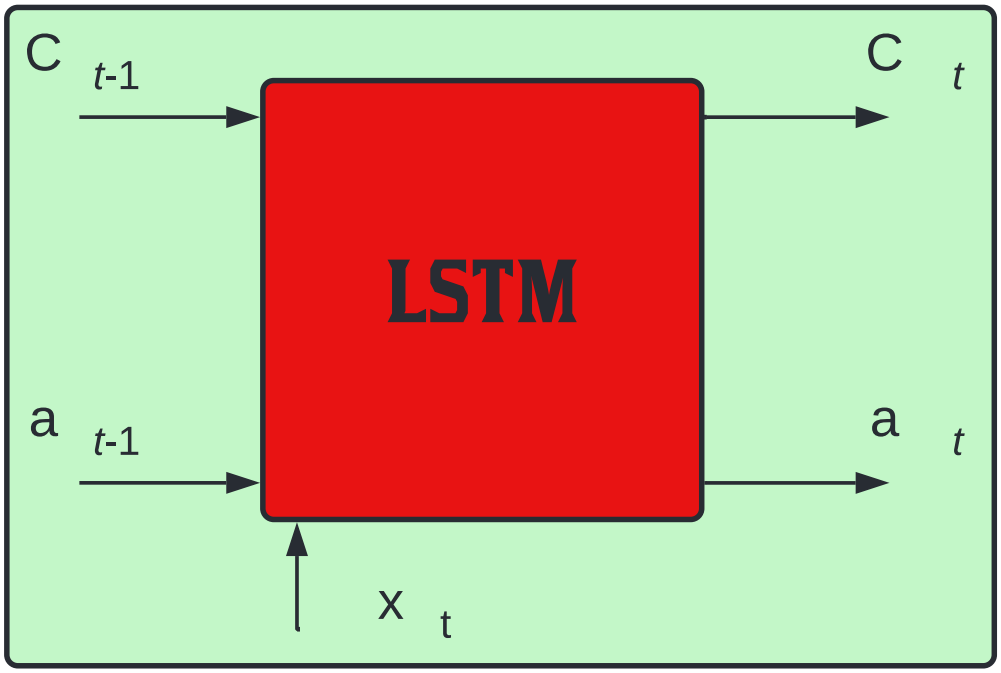

Figura 2.1: Celda de una red LSTM

2.2.2. Compuertas

Las celdas están compuestas por compuertas (figura 2.2). Están las compuertas *forget*, *update* y *output*, las cuales sirven para eliminar, añadir datos y crear el estado oculto actualizado, respectivamente. Cada una de estas compuertas está conformada por una función de activación y un elemento multiplicador. La función de activación puede ser por ejemplo una sigmoidal, la cual tiene un comportamiento de válvula, ya que puede alcanzar valores entre 0 y 1 “cerrando” o “abriendo” la válvula.

Entrando en detalle, la compuerta forget permite decidir qué información descartar. Para ello toma el estado oculto anterior y la entrada actual, los transforma y los lleva a la función sigmoidal, dependiendo de si está más cerca de 0 o de 1 decidirá si eliminarla o mantenerla.

La compuerta update nos permite actualizar la memoria de la celda, para ello tomamos el estado oculto anterior y la entrada actual, los transformamos y llevamos a otra función sigmoidal.

La compuerta output, nos permite calcular el estado oculto. Este estado es simplemente una versión filtrada del estado de la celda generado.

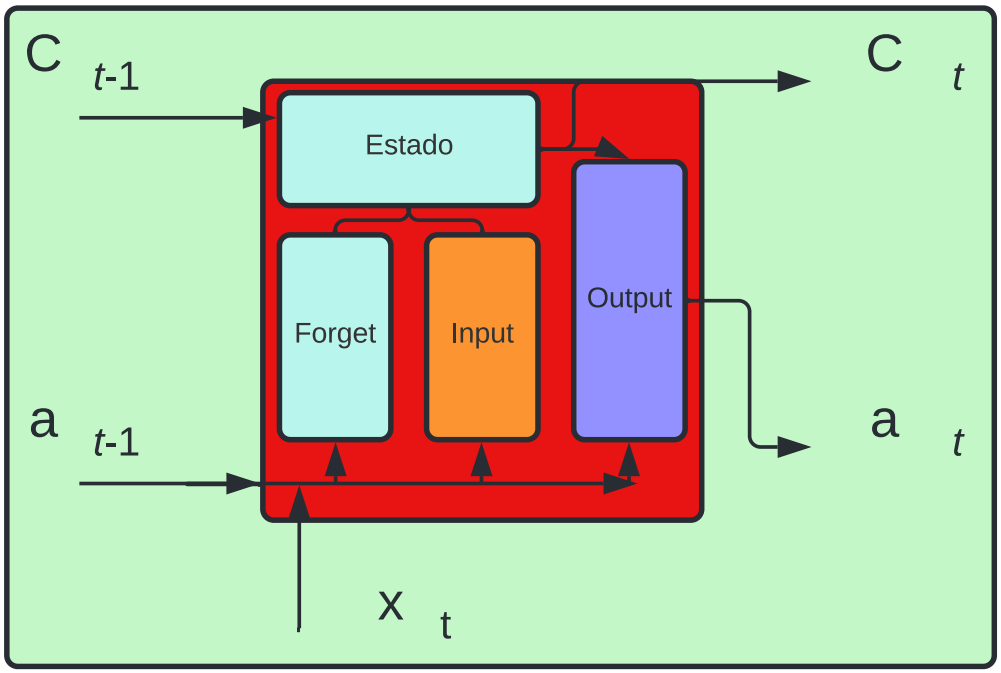

Figura 2.2: Compuertas de una celda LSTM

2.2.3. Actualización de estado

Con estas compuertas podemos actualizar el estado de la celda, para esto lo que hacemos es eliminar la información irrelevante de la celda de estado, multiplicando el valor anterior de esta celda por el vector de la puerta forget. Luego creamos el vector de valores candidatos para la nueva memoria y filtramos estos valores, multiplicando el vector que acabamos de obtener con el generado por la compuerta update y el resultado lo sumamos a los valores anteriores de la celda de estado.

2.2.4. Ventajas de LSTM

En el funcionamiento de la red cuando analizamos una secuencia podemos ver que tenemos réplicas de la celda LSTM vistas anteriormente, cada una corresponde a un instante de tiempo (figura 2.3). Así se puede ver como la información se va añadiendo o eliminando de la memoria y la información relevante se va propagando hasta el estado final y la irrelevante es eliminada.

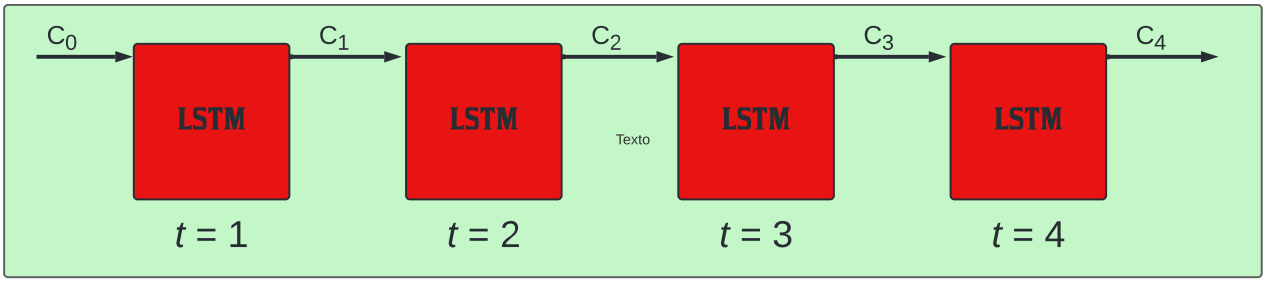

Figura 2.3: Propagación de la información en LSTM [1]

Esto resuelve el problema de las redes recurrentes convencionales, que tienen una memoria a corto plazo, debido a las transformaciones que sufre el estado oculto. De esta forma las redes LSTM son mucho más robustas y tienen una mejor memoria a largo plazo, lo cual es el motivo por el que la hemos escogido, ya que nos interesa ver cual es la mejora de utilizar memoria en una red neuronal, frente a no usarla, a la hora de diagnosticar utilizando varias citas temporales.

2.3. Evaluación de los resultados

Para evaluar los resultados haremos principalmente uso de la comparación de los porcentaje de acierto de las distintas redes neuronales.También utilizaremos distintos métodos de validación, para poder elegir los parámetros de la red adecuadamente, y nos aseguraremos de que aunque la red de mejor resultado no sea debido a que esta sobrentrenada o hace algún tipo de proceso que la invalide. Estos métodos de validación serán la matriz de confusión para comprobar que la red no tiene sobreentrenamiento, los informes de clasificación y las curvas ROC para ver cómo está resultando el diagnóstico en distintos casos. Veremos más sobre estos métodos más adelante.

Capítulo 3

Conjuntos de datos

En el capítulo se verán los conjuntos de datos utilizados, extraídos de la base de datos ADNI, como son estos datos, el preproceso y balanceo por el que pasaron y además la validación que se uso con ellos tras ser utilizados en las distintas redes.

3.1. Base de datos ADNI

Los datos utilizados en la preparación de este artículo se obtuvieron del Alzheimer's Disease Neuroimaging Base de datos de la Iniciativa (ADNI) [13]. El ADNI se puso en marcha en 2003 como una iniciativa público-privada asociación, dirigida por el investigador principal Michael W. Weiner, MD. El objetivo principal de ADNI ha sido probar si la resonancia magnética nuclear (RMN) en serie, la tomografía por emisión de positrones (PET), otros marcadores biológicos y la evaluación clínica y neuropsicológica se pueden combinar para medir la progresión del deterioro cognitivo leve (MCI, del inglés *mild cognitive impairmen*) y la enfermedad de Alzheimer (EA) temprana (Manuscrito para la citación de ADNI [14]).

Esta base de datos reúne las citas de distintos pacientes a los que se les han hecho las pruebas mencionadas anteriormente. Cada una de estas citas cuenta con un diagnóstico, que puede dar que el paciente está sano (CN), padece MCI o que tiene demencia (Dementia).

3.2. Test utilizados

Dentro de las pruebas de la base de datos se encuentran test cognitivos. Dentro de estos test se ha escogido el protocolo ADAS y el protocolo MMSE, los cuales son unos test neuropsicológicos utilizados para diagnóstico de deterioro cognitivo y demencia. Se ha elegido esto ya que el objetivo del trabajo es lograr diagnosticar esta enfermedad haciendo uso de recursos baratos y que puedan utilizarse de forma sencilla. Estos test se componen de preguntas orales o escritas, que se pueden realizar bajo la supervisión de cualquier médico. Estas pruebas se realizarán varias veces en distintos meses y se compararán los resultado obtenidos, es muy importante el orden y la fecha en la que se realizan los test ya que es la línea en la que avanza el paciente. El formato de estos protocolos es:

- Alzheimer's Disease Assessment Scale (ADAS), se compone de 30 preguntas, que puntúan de 0 a 70, siendo el 0 el mejor resultado posible y 70 el peor. Son preguntas que evalúan la memoria, los reconocimientos de preguntas y otras capacidades neurocognitivas.
- Mini-Mental State Examination (MMSE), se compone en 10 secciones en las que se puntúa de 0 a 30, siendo 0 el peor resultado posible y 30 el mejor. Evalúa orientación, memoria inmediata, atención, recuerdo diferido, lenguaje, comandos, repetición , lectura, escritura y construcción.

3.3. Datos iniciales

Los datos con los que empezamos a trabajar son datos de citas de pacientes en las cuales se le realizan los test ADAS y MMSE. Cada cita está asociada a un diagnóstico del paciente, este diagnóstico es apoyado por otras pruebas pero nos quedaremos solo con los datos de los test. De todas estas citas solo nos quedamos con las que pertenezcan a pacientes que tengan 5 o más citas en la base de datos, ocurridas en distintas fechas, y descartamos todas las demás citas. Al final nos quedamos con un total de 407 pacientes, los cuales tienen una cantidad de citas que varía entre 5 y 7 citas, como se puede ver representado en la figura 3.1.

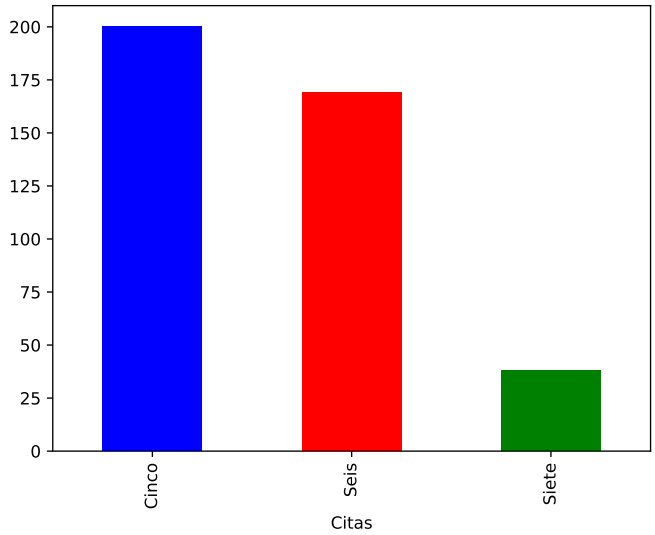

Figura 3.1: Citas de pacientes

Como se puede apreciar, hay una mayoría de pacientes que tienen 5 citas. Cuantas más citas tenga cada paciente es más beneficioso ya que podemos sacar más conjuntos de datos, pero 5 citas es la cantidad mínima porque como veremos más adelante llegaremos a hacer agrupaciones de como máximo 5 citas.

De estos pacientes tenemos un diagnóstico de cada una de sus citas, pero agrupándolos por el diagnóstico de su última cita terminaríamos con un total de cada diagnóstico como se ven en la figura 3.2.

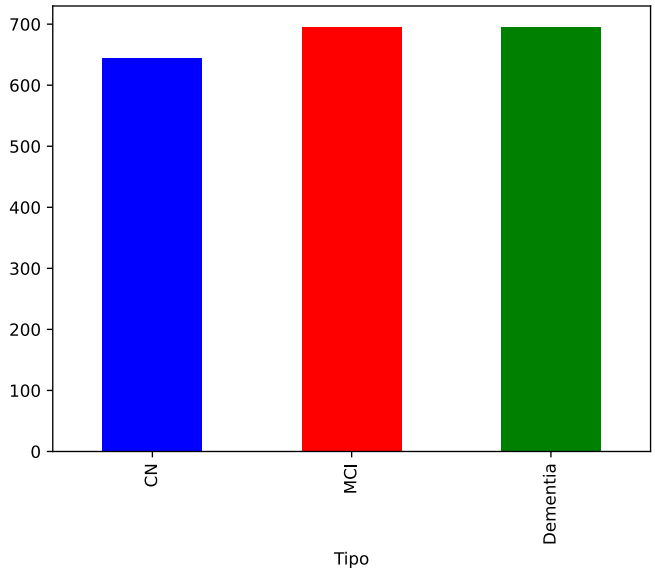

Figura 3.2: Diagnósticos de pacientes

Se puede apreciar que la base de datos está bastante balanceada en un comienzo, pero esto cambiará ya que no utilizaremos todas las citas de un paciente juntas.

La diferencia de tiempo de entre una cita y otra del mismo paciente es de un mínimo de 5 meses. No se ha podido poner un intervalo concreto de tiempo entre citas a los pacientes, debido a la disparidad de estas, pueden tener una cita a los 6 meses y la siguiente al año. Pocos pacientes tenían un intervalo de tiempo regulado entre citas, y debido a la poca cantidad de datos no se pudo permitir eliminar a los pacientes que incumplían esto. La única regla ha sido que tengan 5 meses de diferencia entre citas como mínimo.

3.4. División y balanceo de los datos

Para introducir los datos a la red lo que haremos será introducir agrupando entre n citas de un mismo paciente, y trataremos de predecir la última cita de esta agrupación. Por lo que dividiremos en grupos de n citas y 1 diagnóstico.

Haremos agrupaciones de 2, 3, 4 y 5 citas. Tomaremos como ejemplo el proceso de agrupar en 3 citas a un paciente que tenga un total de 5 citas, como se puede ver en la figura 3.3.

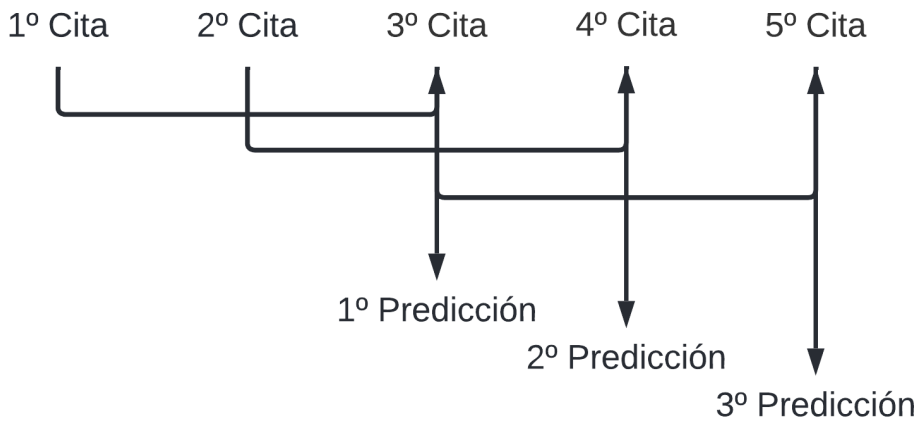

Figura 3.3: División citas

De esta forma un paciente con 5 citas acabará dividido en 3 agrupaciones junto con 3 predicciones, un paciente con 7 citas acabará con 5 agrupaciones y 5 perdiciones.Estas agrupaciones se harían de la misma manera uniendo en grupos 2, 4 y 5 citas. Las fechas de las citas serán modificadas para adaptarse a estas agrupaciones. La fecha de la primera cita de la agrupación tendrá valor 0, y la fecha de las siguientes citas de la misma agrupación tendrán como valor el tiempo en días que ha pasado desde la primera cita de la agrupación. Esto se hace para que las agrupaciones sean independientes entre ellas y no tengan un valor que las relacione con algo externo.

Al agrupar de esta manera se rompe el balanceo, ya que el diagnóstico no será el final de todas las citas del paciente sino del final de la agrupación, por lo que terminaría como se muestra en la figura 3.4

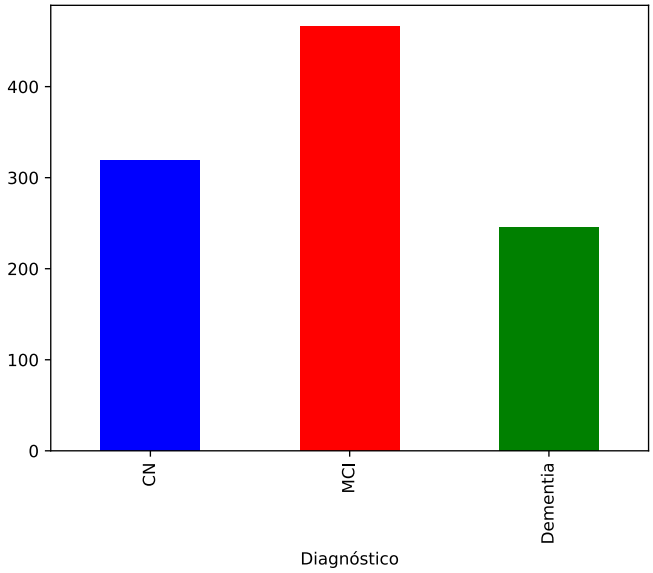

Figura 3.4: Desbalanceo en agrupaciones de 3 citas

Para arreglar este balanceo y nivelar los 3 posibles diagnósticos que hay, lo que se hizo fue eliminar agrupaciones de los pacientes que tenían más de 5 citas, para así no eliminar ningún paciente ni dejarlo con menos agrupaciones que los demás. Esto también implico que no hubiera un balanceo perfecto en algunos casos, pero bastante simétrico como se observa en la figura 3.5.

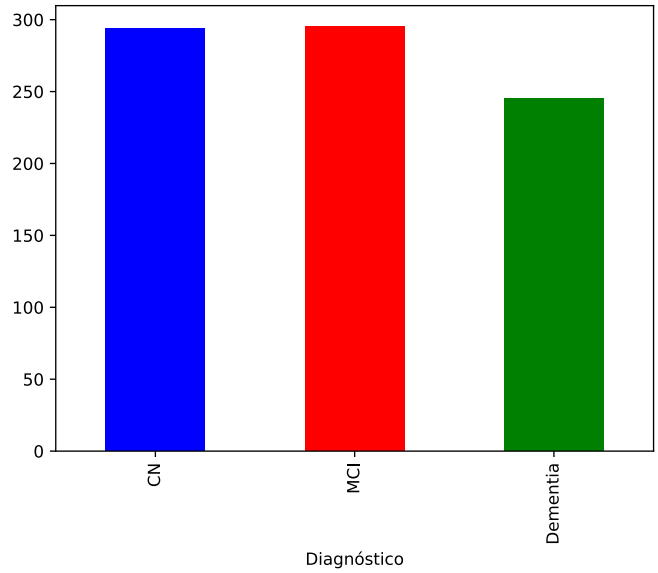

Figura 3.5: Agrupaciones de 3 citas balanceado

Este balanceo también se hará para cada una de las otras agrupaciones, debido a que al cambiar el tamaño de la agrupación también cambiará el número de casos de cada diagnóstico.

3.5. Normalización de los datos

Lo siguiente es normalizar los datos y eliminar los valores no nulos. Se normalizaron entre 0 y 1 todos los datos menos el diagnóstico, el cual una persona sana es representada por 0, una con trastorno cognitivo medio es un 1 y una con demencia es un 2. Esto no se normalizó por el hecho de que va a ser una red neuronal categórica y se transformará más adelante a categórico. Los datos ya nos los dieron sin valores nulos por lo que no se tuvo que hacer ningún tratado más de estos.

3.6. División del conjunto de datos

Lo último que queda hacer con los datos es dividirlos en los grupos de entrenamiento, test y validación. Este es un paso sencillo en el que simplemente separamos los datos. La separación sera de un 15 por ciento de los datos para test, un 15 por ciento para validación y un 70 por ciento para entrenamiento como podemos ver en la figura 3.6.

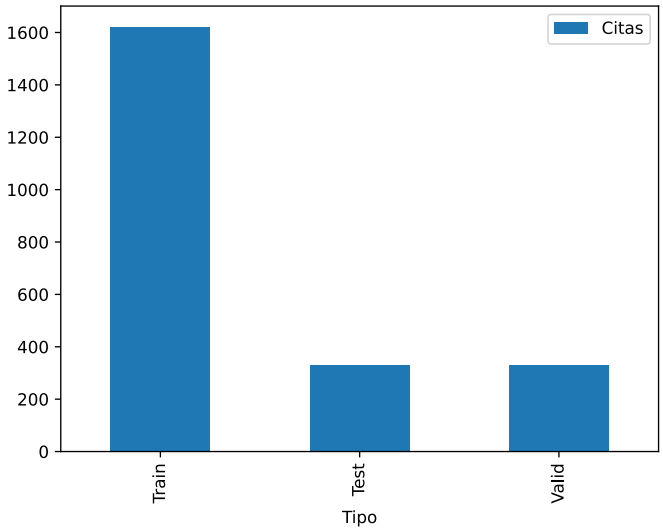

Figura 3.6: Separación de datos en 3 conjuntos

Al hacer la separación nos aseguramos que todas las agrupaciones de citas de un mismo paciente estén en un mismo conjunto, de forma que no pueda quedar 1 cita de un paciente en entrenamiento y otra cita del mismo paciente en test.

3.7. Evaluación de los resultados

Para evaluar estos datos tras haber sido pasados por la red se usarán 5 distintos métodos, los cuales confirmarán la eficiencia de la red.

3.7.1. Validación por evaluación

La primera forma de evaluar los resultados es mediante el método *evaluation* de Keras, el cual con el conjunto de validación, devuelve el valor de pérdida y el valor de las métricas del modelo, que en este caso es del porcentaje de acierto. Por lo que podremos sacar los siguientes datos:

```
Validation results — Loss: 0.6236170530319214 — Accuracy: 69.2307710647583
```

Esta es la forma más simple de comprobar que tan buena es la red, ya que directamente devuelve el porcentaje de acierto de esta, al enfrentarse a datos con los que no ha entrenado.

3.7.2. Gráfica de pérdida

La gráfica de pérdida va devolviendo la cantidad de pérdida que ocurre durante cada ciclo de entrenamiento, cuanto más cercano a 0 y más estable sea es mejor, porque significa que la red no está cambiando y ya está lo suficientemente entrenada. Con esta gráfica se puede ver si hace falta aumentar el tiempo de entrenamiento o disminuirlo como se observa en la figura 3.7.

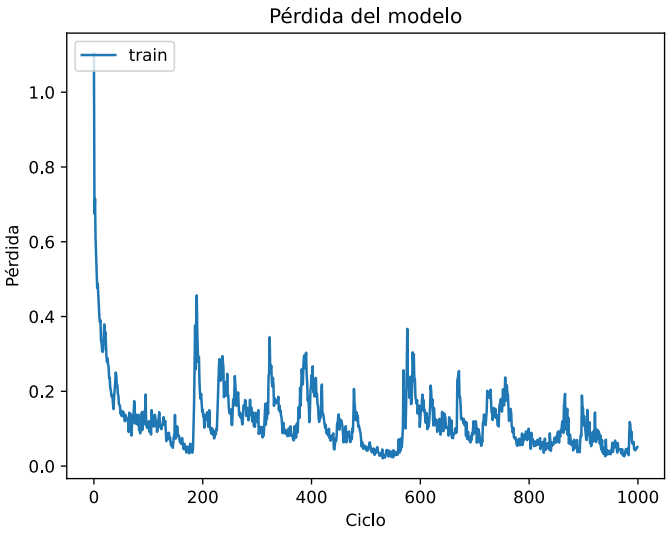

Figura 3.7: Gráfica de pérdida

3.7.3. Matrices de confusión

Las matrices de confusión son otra forma de evaluar los resultados. Se realiza una predicción con la red y esta predicción se le pasa a la función *confusion matrix* de Keras, donde se puede ver de mejor manera donde falla la red neuronal. Esta función devuelve una matriz en la que las filas corresponden al tipo de dato que tendría que haber salido, siendo la 1ºfila el diagnóstico de persona sana, la 2º el de trastorno cognitivo leve y la 3º demencia, mientras que las columnas muestran los datos que predijo la red neuronal, respetando el mismo valor orden de las filas, por lo que cuanto mayor sea el número de la diagonal mayor es el acierto y los números en otras casillas muestran dónde se confundió, se puede ver un ejemplo en la figura 3.8.

```
[[27  2  0]
 [16 10 12]
 [ 0 10 25]]
```

Figura 3.8: Ejemplo de matriz de confusión

Como se ve en la figura 3.8 se podría ver que lo que más le cuesta a la red es diferenciar cuando una persona tiene trastorno cognitivo leve, porque es en esta fila donde más se confunde, mientras que se aprecia que nunca llega a confundir una persona sana con una persona con demencia.

3.7.4. Informes de clasificación

Los informes de clasificación de la misma manera que las matrices de clasifiación, se consiguen al pasarle una predicción a la función *classification report* de Keras. Este método es mucho más sencillo, ya que nos devuelve el porcentaje de precisión de cada diagnóstico, así como otros datos de cada diagnóstico.

Test classification report:

	precision	recall	f1-score	support
0.0	0.63	0.93	0.75	29
1.0	0.45	0.26	0.33	38
2.0	0.68	0.71	0.69	35
accuracy			0.61	102
macro avg	0.59	0.64	0.59	102
weighted avg	0.58	0.61	0.58	102

Figura 3.9: Ejemplo de informe de clasificación

3.7.5. Curva ROC

La Receiver Operating Characteristic (ROC) curva, sirve para medir la salida del clasificador, representa una tasa de verdaderos positivos en el eje *y* y falsos positivos en el eje *x*. De esta forma el punto en la esquina superior izquierda de la gráfica es el punto ideal, teniendo 0 falsos positivos y el máximo de verdaderos positivos. El área debajo de la curva ROC se conoce como Area Under The Curve (AUC) [15]. Este área representa el grado de separación, indicando cuanto es capaz el modelo de distinguir entre las clases. Cuanto más cerca esté de 1 mejor será el modelo, cuando está cerca de 0,5 significa que no es capaz de diferenciar la clase, y cerca de 0 significa que confunde completamente las clases.

Se dibujará una curva de cada diagnóstico, además de una curva del macro promedio, que es el peso sumado de los diagnósticos dándoles el mismo valor. Se analizará lo que ocurre para cada modelo con las curvas ROC y el AUC, para luego comparar entre modelos con esto.

Capítulo 4

Desarrollo y resultados

En este capítulo se verá como se desarrollaron las redes neuronales, se compararon las redes y se presentarán los resultdos obtenidos.

4.1. Creación la red LSTM

Primero se creó una red LSTM con la estructura deseada, pero con unos valores por defecto de los parámetros de la misma que posteriormente habrán de optimizarse:

```
Targets_training = to_categorical(y_train.reshape((-1,1)))
Targets_valid = to_categorical(y_valid.reshape((-1,1)))
Targets_test = to_categorical(y_test.reshape((-1,1)))

loss_function_used = CategoricalCrossentropy(from_logits=True)
opt = Adam(learning_rate=0.01, decay=1e-6)
model = Sequential()
model.add(LSTM(100, return_sequences=False, input_shape=(n_past, n_features-1)))
model.add(Dropout(0.2))
model.add(Dense(units=2))
model.add(Activation('softmax'))
model.compile(loss=loss_function_used, optimizer=opt, metrics=['acc'])
```

Lo primero que se hace es convertir las salidas deseadas, o sea la predicción, a tipo categórico y esto lo que hace es que como tenemos 3 elecciones los convertirá como [0,0,1], [0,1,0] o [1,0,0], siendo cada uno de estos el equivalente a un diagnóstico. Como se puede ver es de tipo *CategoricalCrossentropy*, que es una predicción categórica como ya hemos dicho, que devolverá el peso de su predicción. Si devuelve [0.35,0.6,0.5] esto equivaldrá a quedarse la posición con mayor peso como 1 y el resto como 0, en este caso sería [0,1,0].

Los parámetros de entrada con los que la red neuronal cuenta son:

- Una capa de entrada con 100 neuronas o bloques LSTM.
- Un dropout layer [16] que pone de forma aleatoria entradas a 0 para evitar el sobreajuste.
- Una capa de salida con total de 3 elementos, ya que la red devolverá el resultado graduando las 3 posibles elecciones.
- La optimización es de tipo *Adam* [17] con un ratio de aprendizaje de 0,01, el ratio de aprendizaje indica que tan rápido se adapta el modelo al aprendizaje, cuanto mayor sea más rápido aprende.
- La función de activación es *softmax* [18], que devolverá la probabilidad de cada diagnóstico de ser el correcto como un valor entre 0 y 1.
- Se indica que las métricas [19] de la red sean de tipo acierto, por lo que devolverá el porcentaje de acierto que tiene.

```
1 history = model.fit(X_train,Targets_training, batch_size=64, epochs=64,verbose=0)
```

Lo que queda es entrenar el modelo [20], pasándole el conjunto de entrenamiento de entrada y los valores deseados de salida que tiene que predecir. Los únicos parámetros que se le pasa son:

- El *batch*, que indica el número de entradas que se le introducen a la red antes de actualizar los pesos de la misma.
- El *epoch* que es el número máximo de ciclos. En cada ciclo la red va a entrenar con todo el conjunto de datos de entrada.
- El último parámetro es el *verbose*, que simplemente indica si quieres que la red mande una salida por pantalla de como va avanzando el entrenamiento.

4.2. Búsqueda de mejores parámetros

La búsqueda de mejores parámetros se realizó mediante *grid search*, lo cual es probar la mayor combinación posible de parámetros y luego quedarnos con la combinación que dio el mejor resultado. Se crearon la mayor cantidad posible de combinaciones y se probaron todas, guardando los resultados de validación para cada una, y luego comparándolas se encontró la mejor. Esto se hizo para los 4 casos de agrupaciones de citas.

4.3. Creación de una red convencional

Ahora crearemos una red convencional, siguiendo un proceso similar al realizado para la red LSTM, lo único que cambiará será que los datos no estarán agrupados, si no serán los datos de simplemente una visita y no tendrán variable temporal.También la red no será de tipo LSTM, sino tipo *Dense* [21]:

```
1  Targets_training = to_categorical(target_train.reshape((-1,1)))
2  Targets_valid = to_categorical(target_valid.reshape((-1,1)))
3  Targets_test = to_categorical(target_test.reshape((-1,1)))
4
5  loss_function_used = CategoricalCrossentropy(from_logits=True)
6  opt = Adam(learning_rate=lr, decay=1e-6)
7
8  model = Sequential()
9  model.add(Dense(n_hidden1, input_dim=n_features, activation='relu'))
10 model.add(Dense(n_hidden2, activation='relu'))
11 model.add(Dense(3, activation=activation))
12 model.compile(loss=loss_function_used, optimizer=opt, metrics=['accuracy'])
13
14 history = model.fit(input_train,Targets_training, batch_size=n_batch, epochs=n_epoch,
       verbose=0)
```

Cuenta con la misma conversión categórica y parámetros y también se le aplica una búsqueda de parámetros para encontrar los mejores.

4.4. Resultados

Tras conseguir los mejores parámetros para cada red hecha, se lograron obtener los siguientes resultados en cada una, comparando mediante el acierto de la red y su curva ROC:

4.4.1. LSTM con 2 citas

En el caso de la red LSTM haciendo agrupaciones de 2 citas, da un porcentaje final de acierto del 71,49 por ciento en la validación, y un 67,56 por ciento en el test final. Los mejores parámetros para esta red son:

- Test de función de activación, *linear*.
- *Ratio de aprendizaje* de 0,05.
- 16 *neuronas ocultas*.
- 400 *número máximo de ciclos*.
- 500 *batch*.

Como se puede ver en la matriz de confusión de la tabla 4.1, el mayor problema de esta red es diferenciar el diagnóstico de MCI. También confunde a veces Dementia con CN.

Predicho / Real	CN	MCI	Dementia
CN	70	10	0
MCI	27	43	10
Dementia	4	21	37

Tabla 4.1: Matriz de confusión de 2 citas

Viéndolo en la curva ROC de la figura 4.1, se aprecia como el valor del AUC del diagnóstico 1 (MCI) está muy cerca de valer 0,5 en algunos puntos. El valor 0,5 representa que la red no es capaz de diferenciar la clase.

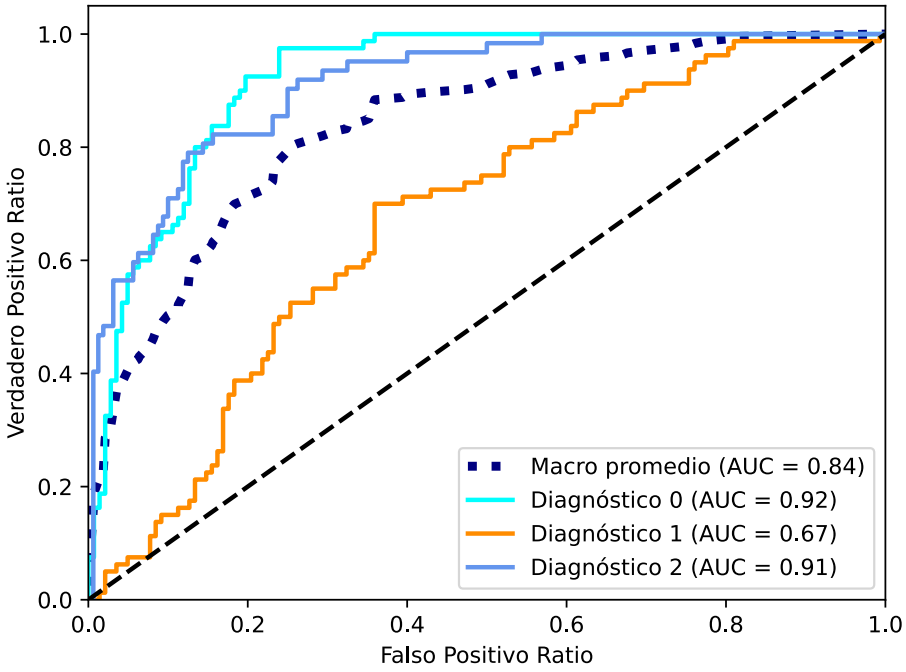

Figura 4.1: Gráfica ROC de LSTM con 2 citas

4.4.2. LSTM con 3 citas

En el caso de la red LSTM haciendo agrupaciones de 3 citas, da un porcentaje final de acierto del 69,94 por ciento en la validación, y un 70,5 por ciento en el test final. Los mejores parámetros para esta red son:

- Test de función de activación, *linear*.
- *Ratio de aprendizaje* de 0,1.
- 32 *neuronas ocultas*.
- 1000 *número máximo de ciclos*.
- 248 *batch*.

Como se puede ver en la matriz de confusión de la tabla 4.2, el mayor problema de esta red sigue siendo diferenciar el diagnóstico de MCI, aunque se ve una mejoría en este diagnóstico frente a la anterior red. También ya no tiene problemas para diferenciar entre CN y Dementia.

Predicho / Real	CN	MCI	Dementia
CN	60	11	0
MCI	16	40	15
Dementia	0	17	41

Tabla 4.2: Resultado de las redes

Viéndolo en la curva ROC de la figura 4.2, se aprecia como el AUC del diagnóstico 1 se separa más del valor de 0,5 que la anterior red. El AUC de los otros diagnósticos también mejora, por lo que el promedio sube.

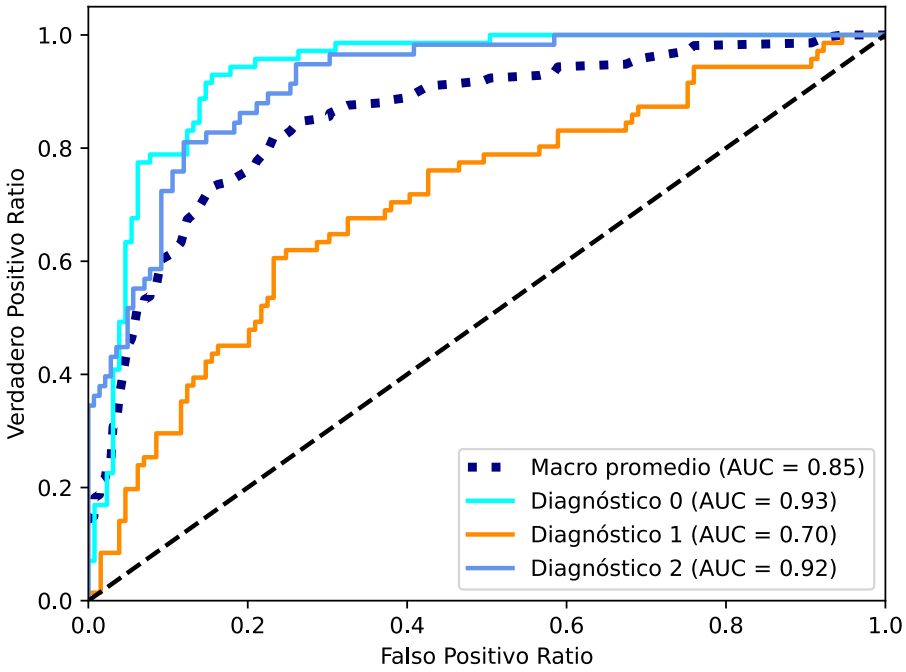

Figura 4.2: Gráfica ROC de LSTM con 3 citas

4.4.3. LSTM con 4 citas

En el caso de la red LSTM haciendo agrupaciones de 4 citas, da un porcentaje final de acierto del 74,65 por ciento en la validación, y un 75 por ciento en el test final. Los mejores parámetros para esta red son:

- Test de activación, *linear*.
- *Ratio de aprendizaje* de 0,1.
- 128 *neuronas ocultas*.
- 248 *número máximo de ciclos*.
- 1000 *batch*.

Como se observa en la matriz de confusión de la tabla 4.3, el problema de diferenciar el diagnóstico MCI sigue mejorando. La mayor mejoría es en el diagnóstico CN, en el que solo tiene 2 confusiones.

Predicho / Real	CN	MCI	Dementia
CN	50	2	0
MCI	9	36	7
Dementia	0	20	28

Tabla 4.3: Resultado de las redes

Viéndolo en la curva ROC de la figura 4.3, todos los AUC mejoran haciendo que el área del macro promedio aumente.

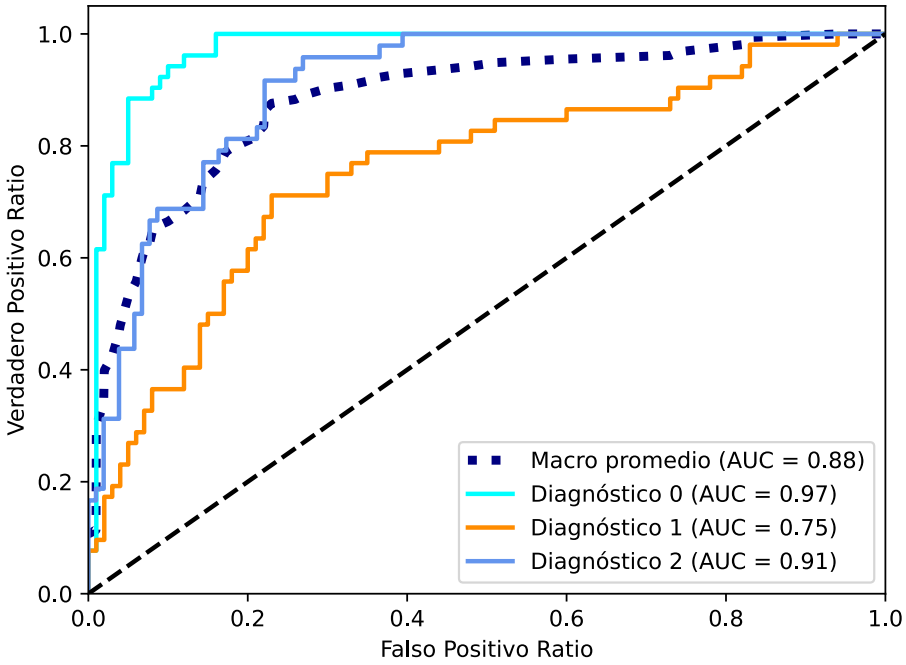

Figura 4.3: Gráfica ROC de LSTM con 4 citas

4.4.4. LSTM con 5 citas

En el caso de la red LSTM haciendo agrupaciones de 5 citas, da un porcentaje final de acierto del 68 por ciento en la validación, y un 74,49 por ciento en el test final. Los mejores parámetros para esta red son:

- Test de función de activación, *linear*.
- *Ratio de aprendizaje* de 0,1.
- 16 *neuronas ocultas*.
- 45 *número máximo de ciclos*.
- 128 *batch*.

Como se observa en la matriz de confusión de la tabla 4.4, el problema de diferenciar el diagnóstico MCI vuelve a empeorar levemente. Los otros diagnósticos se mantienen igual.

Predicho / Real	CN	MCI	Dementia
CN	28	1	0
MCI	7	15	12
Dementia	0	5	30

Tabla 4.4: Resultado de las redes

Viéndolo en la curva ROC de la figura 4.4, se puede ver como el AUC del diagnóstico 1 ha vuelto a empeorar, haciendo bajar el área promedio.

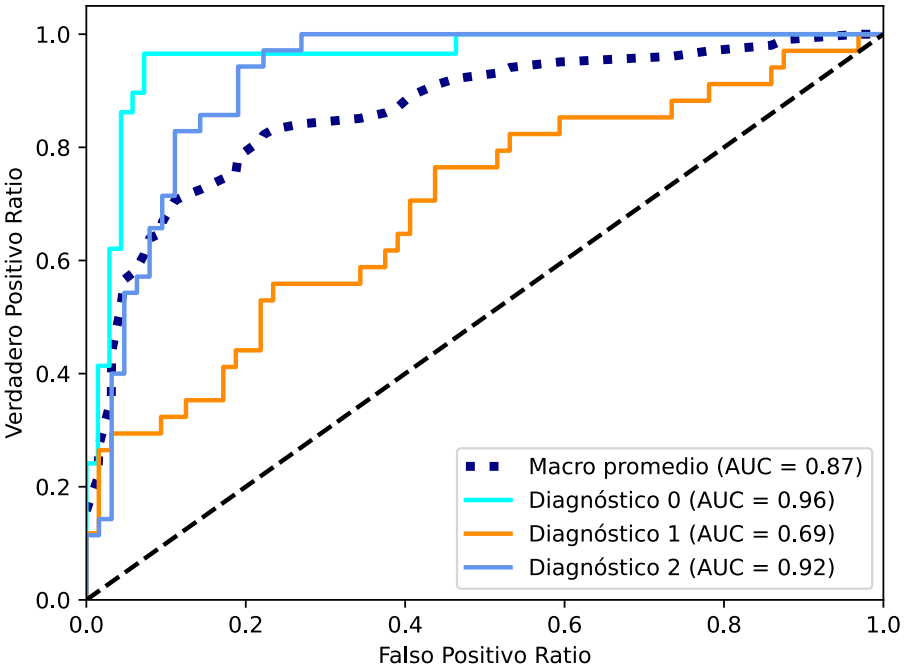

Figura 4.4: Gráfica ROC de LSTM con 5 citas

4.4.5. Red convencional

En el caso de la red convencional da un porcentaje final de acierto del 55 por ciento en la validación, y un 60,23 por ciento en el test final. Los mejores parámetros para esta red son:

- Test de función de activación, *linear*.
- *Ratio de aprendizaje* de 0.001.
- Una primera capa de 16 *neuronas ocultas*.
- Una segunda capa de 16 *neuronas ocultas*.
- 32 *número máximo de ciclos*.
- 248 *batch*.

Como se observa en la matriz de confusión de la tabla 4.5, el problema de diferenciar el diagnóstico MCI es mucho más grave que en cualquiera de las redes LSTM. Los otros diagnósticos también empeoran pero en menor medida.

Predicho / Real	CN	MCI	Dementia
CN	45	11	1
MCI	27	9	21
Dementia	2	6	49

Tabla 4.5: Resultado de las redes

Viéndolo en la curva ROC de la figura 4.5, se puede ver como el AUC del diagnóstico 1 llega a incluso pasar por debajo de la línea del valor 0,5. Esto hace que el diagnóstico 1 sea prácticamente aleatorio.

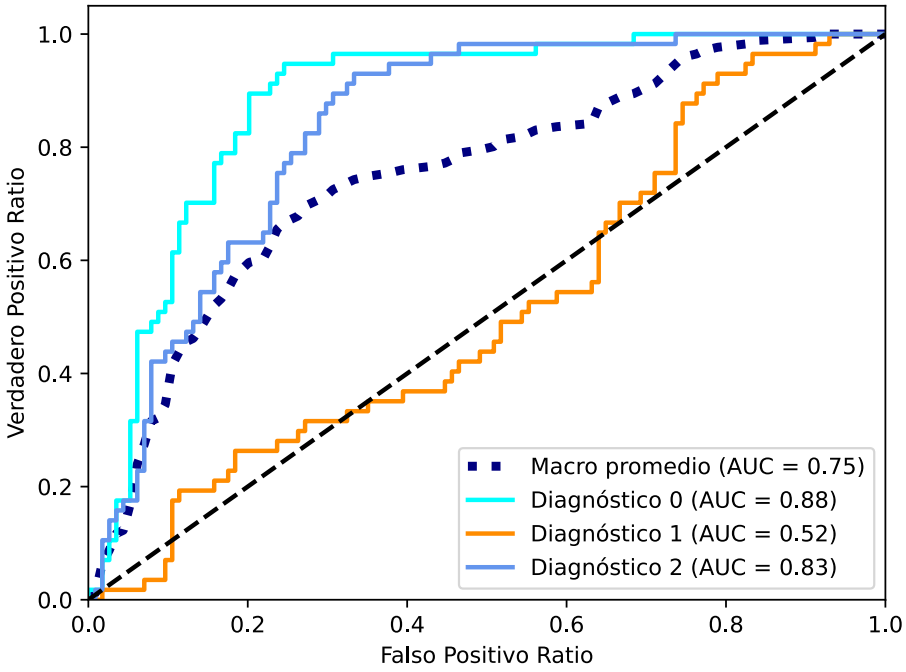

Figura 4.5: Gráfica ROC de red convencional con 1 citas

4.4.6. Tabla de resultados de test

	Prec CN	MCI	Dem	Recall MCI	MCI	Dem	AUC CN	MCI	Dem	Macro
LSTM 2	0,69	0,58	0,79	0,88	0,54	0,60	0,92	0,67	0,91	0,84
LSTM 3	0,79	0,59	0,73	0,85	0,56	0,71	0,93	0,70	0,92	0,85
LSTM 4	0,85	0,62	0,80	0,96	0,69	0,58	0,97	0,75	0,91	0,88
LSTM 5	0,80	0,71	0,71	0,97	0,44	0,86	0,96	0,68	0,92	0,87
Conven	0,61	0,35	0,69	0,79	0,16	0,86	0,88	0,52	0,83	0,75

Tabla 4.6: Resultado de las redes

Esta tabla 4.6 formada por la información de precisión (Prec) y recall de cada uno de los diagnósticos, siendo estos CN, MCI ,Dem (Dementia), de las distintas redes LSTM y la red convencional (conven). También está el AUC de los diagnósticos junto con el macro promedio sobre el conjunto de test.

Analizando estos datos se ve claramente como la red LSTM con agrupación de 5 visitas es la mejor con diferencia. Esta mejora es en todos los diagnósticos, pero el que marca la diferencia es el diagnóstico de MCI, dando así un mejor porcentaje de acierto y un AUC mayor.

La clase con mejor eficiencia es el diagnóstico CN, ya que es el que tiene mayor porcentaje de acierto y menor cantidad de fallos, el diagnóstico Dementia está bastante parejo con esta primera clase. El diagnóstico con peor eficiencia es MCI, que como ya se ha mencionado antes es el que mayor problema trae, siendo este el que condiciona la eficiencia general de las redes.

Capítulo 5

Conclusiones y líneas futuras

5.1. Conclusión

Como conclusión de este proyecto se puede ver como el uso de una red LSTM marca una mejoría frente al de una red convencional, siendo un total de un 9,5 por ciento de mejora en la versión de la red LSTM más eficiente.

Esta es debido a la capacidad de la red LSTM, al ser una red de tipo recurrente, de procesar información temporal en los datos memorizando información en sus estados internos de los datos ya vistos. Esto le ha permitido no solo saber los datos actuales del paciente, sino saber también su historial médico que es una ayuda ya que puede ver si este está empeorando y desarrollando una enfermedad, o se está manteniendo estable y no tiene ningún problema.

Dentro de los distintas redes LSTM llevadas a cabo, se puede observar como cuanto mayor es el historial médico o cantidad de citas de un mismo paciente que se le pasa a la red, esta aumenta su eficacia hasta cierto punto. El hecho de que la red con agrupación de 5 se haya salido de la línea que estaba llevando y haya empeorado es posible que haya ocurrido debido a la falta de datos. Esto ocurre ya que cuanto mayor es el número de citas dentro de una agrupación, menor es el número de agrupaciones final con las que la red contaba y la cantidad de información. Esto ya era un grave problema desde el inicio, por lo que reducir aun más la cantidad de información solo lo empeoraba. Se debería de hacer un estudio con una mayor cantidad de datos para poder determinar si realmente 4 es el número máximo de citas que da el mejor rendimiento, o si esta bajada fue resultante de una escasez de datos.

El último punto que se puede concluir es el problema de diferenciar el trastorno cognitivo leve cuando es muy leve o muy grave, ya que los límites de estos son muy finos. Al ser test de puntuación, un diagnóstico de otro se puede diferenciar por pequeñas cantidades de puntos. Dentro de un mismo margen de puntuación puede ser un diagnóstico u otro para distintos pacientes. Esto puede ser debido a algún fallo humano en un test o porque estas enfermedades al todavía no ser lo suficientemente conocidas, los test hechos no sirvan para todas las personas.

Aun con los fallos de la red, esta podría llegar a ser útil, ya que aunque no tiene un porcentaje de acierto lo suficientemente alto como para dar un diagnóstico con muy alta fiabilidad, podría dar diagnósticos preventivos tras los cuales si da un indicio de tener un mal diagnóstico se realizarían pruebas mucho más complejas para aclarar la situación del paciente. Podría servir como un primer filtro para realizar estas pruebas usándose como sistema de cribado, debido a lo fácil de realizar que son los test necesarios para que dé un diagnóstico la red.

Además, a nivel personal, durante el desarrollo del proyecto me he familiarizado con el lenguaje de programación *python*, aprendiendo a utilizarlo con más habilidad y eficiencia. También he aprendido como hacer tratado y manejo de datos, filtrándolos y preprocesándolos. Las redes neuronales ha sido el tema en el que más he aprendido y descubierto, debido a que era el que más desconocía y ha sido en el que más he tenido que investigar, ahora ya puedo entender como funcionan y trabajar con ellas sin problema.

5.2. Líneas futuras

Las líneas futuras para este trabajo serían principalmente la búsqueda de una mayor cantidad de datos para utilizar, ya que un gran problema ha sido éste, y también tratar de conseguir información de pacientes con un mayor número de visitas para poder probar con agrupaciones mayores.

Otra necesidad es la búsqueda de otras pruebas tan sencillas como las dos utilizadas, que pudieran lograr una mayor diferenciación a las personas con trastorno cognitivo leve, ya que en las pruebas utilizadas se ha podido ver que no termina de haber una diferenciación clara en esta categoría.

La última opción sería utilizar validaciones como *cross-validation one leave out* [22], que necesitan de una menor cantidad de datos. Esta validación también se empezó a usar en el proyecto pero se abandonó debido a la gran cantidad de tiempo necesaria para su uso y a la falta de potencia computacional para llevarla a cabo.

Capítulo 6

Summary and Conclusions

6.1. Conclusion

As a conclusion of this project, it can be seen that the use of an LSTM network marks an improvement over that of a conventional network, being a total of 9.5 percent improvement in the most efficient version of the LSTM network.

This is due to the ability of the LSTM network, being a recurring network, to process temporary information in the data by storing information in its internal states of the data already viewed. This has allowed the network, not only to know the patient's current data, but also to know their medical history which is helpful as you can see if the patient is getting worse and developing a disease, or is staying stable and not having any problems.

Within the various LSTM networks carried out, it can be observed that the greater the medical history or the number of appointments of the same patient that is transferred to the network, the network increases its effectiveness to a certain extent. The fact that the network with grouping of 5 went off the line it was carrying and got worse is likely to have happened due to a lack of data. This is because the higher the number of appointments within a group, the lower the number of final groupings that the network had and the amount of information. This was already a serious problem from the start, so reducing the amount of information even further only made it worse. A study with more data should be done to determine if 4 is really the maximum number of citations that gives the best performance, or if this drop was the result of a lack of data.

The last point that can be concluded is the problem of differentiating mild cognitive impairment when it is very mild or very severe, since the boundaries of these are very fine. Being a score test, one diagnosis can be differentiated by small amounts of points. Within the same scoring range there may be one diagnosis or another for different patients. This may be due to some human error in a test or because these diseases are not been sufficiently known, the tests done do not work for all people.

Even with network failures, it could be useful, because although it does not have a sufficiently high percentage of success to make a very reliable diagnosis, it could give preventive diagnoses after which if it gives an indication of a misdiagnosis, much more complex tests would be performed to clarify the patient's situation. It could serve as a first filter to perform these tests by using it as a screening system, due to how easy it is to perform the tests necessary for a network diagnosis.

Also, on a personal level, during the development of the project I have become familiar with the *Python* programming language, learning to use it with more skill and efficiency. I have also learned how to treat and handle data, filtering and preprocessing them. Neural networks have been the subject in which I have learned and discovered the most, because it was the one I didn't know the most and it has been the one I have had to research the most, now I can understand how they function and work with them without problem.

6.2. Future lines

Future lines for this work would be mainly to search for more data to use, since this has been a big problem, and also to try to get information from patients with a larger number of visits to be able to test with larger groups.

Another need is the search for other tests as simple as the two used, which could achieve a greater differentiation for people with mild cognitive impairment, since the tests used have shown that there is no clear differentiation in this category.

The last option would be to use validations such as *cross-validation one leave out* [22], which require less data. This validation was also started in the project but was abandoned due to the large amount of time required for its use and the lack of computational power to carry it out.

Capítulo 7

Presupuesto

Todas las tecnologías usadas son de código abierto, y no es necesario pagar ninguna licencia.

El único coste sería el sueldo de informático contratado y del equipo en el que se entrena la red. El tiempo destinado para el desarrollo del proyecto fue de unas 300 horas. El sueldo de un informático junior promedio es de unos 20€ la hora, por lo que costaría 6000€ el trabajador.

El precio de un ordenador promedio con la potencia de entrenar redes neuronales de forma eficiente es de unos 1300€.

Con estos datos viendo la tabla 7.1, observamos que el total de presupuesto necesario sería de unos 7300€.

	Costo
Sueldo	6000€
Equipo	1300€
Total	7300€

Tabla 7.1: Desglose del presupuesto

Bibliografía

[1] Codificandobits, LSTM:. https://www.codificandobits.com/blog/redes-lstm/.

[2] Número de casos de demencia registrados en España:. https://es.statista.com/estadisticas/1037990/numero-de-casos-de-demencia-en-espana/#:~:text=Esta%20estad%C3%ADstica%20muestra%20la%20evoluci%C3%B3n,aproximadamente%20442.000%20casos%20en%20Espa%C3%B1a.

[3] LSTM layer:. https://Keras.io/api/layers/recurrent_layers/lstm/.

[4] Artículo, A Real-Time Clinical Decision Support System, for Mild Cognitive Impairment Detection, Based on a Hybrid Neural Architecture:. https://www.hindawi.com/journals/cmmm/2021/5545297/.

[5] Artículo, Machine learning for modeling the progression of Alzheimer disease dementia using clinical data:. https://pubmed.ncbi.nlm.nih.gov/34350389/.

[6] Ardiel García Rodríguez. TFG, Cogmultest: Aplicación web y móvil para la gestión de múltiples test congitivos:. https://www.webs.ull.es/users/pgarcia/public/alzh/tfgs/TFGMemoriaProyectoTFG_08_07_2015.pdf.

[7] Anayara Moreno Merino. TFG, Sistema basado en redes neuronales profundas, para la búsqueda y análisis de patrones estructurados en datos longitudinales, de criterios diagnósticos de la Enfermedad del Alzheimer y del Deterioro Cognitivo Leve.

[8] Jupyter-notebook Anaconda:. https://docs.anaconda.com/ae-notebooks/4.3.1/user-guide/basic-tasks/apps/jupyter/.

[9] Pandas:. https://pandas.pydata.org/docs/.

[10] Keras TensorFlow. https://www.tensorflow.org/guide/Keras?hl=es-419.

[11] Scikit learn:. https://scikit-learn.org/stable/user_guide.html.

[12] Matplotlib. https://matplotlib.org/stable/users/index.

[13] BBDD, Alzheimer's Disease Neuroimaging Initiative (ADNI):. https://adni.loni.usc.edu/.

[14] Manuscrito de citaciones de ADNI:. https://adni.loni.usc.edu/wp-content/themes/freshnews-dev-v2/documents/policy/ADNI_Manuscript_Citations.pdf.

[15] Introduction to LSTM Units in RNN:. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5.

[16] Dropout layer Keras:. https://Keras.io/api/layers/regularization_layers/dropout/.

[17] Optimizador Adam de Keras:. https://Keras.io/api/optimizers/adam/.

[18] Función de activación softmax:. https://Keras.io/api/layers/activations/.

[19] Métricas de Keras:. https://Keras.io/api/metrics/.

[20] Metodo fit de Keras:. https://Keras.io/api/models/model_training_apis/.

[21] A Complete Understanding of Dense Layers in Neural Networks:. https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/.

[22] Cross-validation one leave out:. https://link.springer.com/referenceworkentry/10.1007/978-0-387-30164-8_469#:~:text=Definition,a%20single%2Ditem%20test%20set.