UNIVERSIDAD DE VALLADOLID

Escuela Técnica Superior de Ingenieros de Telecomunicación

Universidad de Valladolid

TRABAJO DE FIN DE GRADO

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

REDES RECURRENTES PROFUNDAS PARA EL DIAGNÓSTICO DEL TDAH EN LA INFANCIA

Autor: PAULA ÁLVAREZ TAVERA

Tutor: J.P. CASASECA DE LA HIGUERA

CARLOS ALBEROLA LÓPEZ

TÍTULO: REDES RECURRENTES PROFUNDAS PARA EL DIAGNÓSTICO DEL TDAH EN LA INFANCIA
AUTOR: PAULA ÁLVAREZ TAVERA
TUTOR: J.P. CASASECA DE LA HIGUERA
CARLOS ALBEROLA LÓPEZ
DEPARTAMENTO: TSCIT

Miembros del Tribunal
PRESIDENTE: CARLOS ALBEROLA LÓPEZ
SECRETARIO: RODRIGO DE LUIS GARCÍA
VOCAL: J.P. CASASECA DE LA HIGUERA
SUPLENTE 1: SANTIAGO AJA FERNÁNDEZ
SUPLENTE 2: FEDERICO SIMMROSS WATTENBERG
CALIFICACIÓN:

Resumen

El trastorno por déficit de atención e hiperactividad es un trastorno crónico y genético que se da principalmente en la infancia. Las técnicas actuales de diagnóstico de este trastorno se basan en pruebas incapaces de detectar la presencia del TDAH de forma objetiva. Debido a esto surge la necesidad de crear un sistema de diagnóstico imparcial y fiable. En este trabajo se describe un sistema que combina el uso de la actigrafía para obtener datos de un niño, junto con técnicas de *Deep Learning* para detectar la presencia de TDAH. Para poder llevar a cabo este modelo se han utilizado unidades de procesamiento gráfico o GPU. Los datos obtenidos del actígrafo se han clasificado en distintos periodos y divididos en actividades de diferente duración. Estos datos se transforman en imágenes mediante la técnica del espectrograma y se utilizan como la entrada de una red convolucional preentrenda. Esta red extrae las características espaciales, las cuales se emplean para entrenar una red recurrente que utiliza secuencias de las imágenes de un paciente para clasificar los datos. Estas redes analizan las dependencias temporales y pueden llevar a cabo el diagnostico.

Palabras clave

Diagnóstico automático, TDAH, LSTM, CNN, espectrograma, Deep Learning, actigrafía

Abstract

Attention Deficit Hyperactivity Disorder (ADHD) is a chronic, genetic disorder that occurs primarily in childhood. Current diagnostic techniques for this disorder are based on tests that are unable to detect the presence of ADHD objectively. Because of this, the need to create an impartial and reliable diagnostic system arises. This paper describes a system that combines the use of actigraphy to obtain data from a child, along with Deep Learning techniques to detect the presence of ADHD. In order to develop this model, graphic processing units or GPUs have been used. The data obtained from the actigraph have been classified into different periods and divided into activities of different duration. Activity windows are transformed into time-frequency images using spectrograms and are used as the input to a pre-trained convolutional network. This network extracts spatial features, which are used to train a recurrent network that uses sequences of a patient's images to classify the data. This network analyses time dependencies and can carry out diagnosis.

Keywords

Automatic diagnosis, ADHD, LSTM, CNN, spectrogram, Deep Learning, actigraphy

Agradecimientos

En primer lugar, me gustaría dar las gracias a mis tutores Pablo Casaseca y Carlos Alberola por todo el tiempo que han dedicado a ayudarme y por todo lo que he aprendido con este trabajo. Especialmente quiero dar las gracias a Patricia Amado Caballero quien comenzó este estudio sobre el TDAH con *Deep Learning* en su TFG y cuya ayuda ha sido fundamental para poder realizar este proyecto.

También me gustaría agradecer al laboratorio de procesado de imagen el haberme dado acceso a las máquinas del laboratorio, pudiendo hacer así las simulaciones necesarias para entrenar las redes neuronales.

Finalmente quiero dar las gracias a mi familia y amigos por todo el apoyo que he recibido durante estos meses. En concreto, quiero agradecérselo a mi hermana por confiar siempre en mí y animarme constantemente a continuar trabajando y esforzándome. Gracias a todos.

Índice general

Índice de figuras

Índice de tablas

1. INTRODUCCIÓN

1.1. Planteamiento del problema

El TDAH o trastorno por déficit de atención e hiperactividad es un trastorno crónico y genético que puede darse tanto en la infancia como en la etapa adulta [1] . Los principales síntomas en los niños son las dificultades para mantener la atención, concentrarse, recordar tareas cotidianas y controlar sus impulsos. Para identificar este trastorno se utilizan tanto pruebas clínicas como pruebas médicas. El diagnostico de este problema es una tarea larga y tediosa puesto que requiere mucho tiempo analizando el entorno familiar y académico del paciente, así como diferentes tipos de pruebas realizadas por neurólogos, psiquiatras infantiles, etc... [2] [3]

La dificultad para determinar si un niño tiene TDAH o no puede llevar al sobrediagnóstico [4], cuyas consecuencias afectan considerablemente tanto a un niño sano al que se le diagnostica este trastorno de forma incorrecta, como a un niño con este desorden que no recibe la ayuda necesaria. Con el propósito de evitar el sobrediagnóstico, surge la necesidad de encontrar un método capaz de detectar la presencia del trastorno por déficit de atención e hiperactividad de forma objetiva.

1.2. Objetivos

Para solucionar el problema planteado se busca crear un sistema capaz de detectar correctamente la presencia del TDAH. Se ha demostrado que el procesado de datos actimétricos utilizando métodos no lineales obtiene buenos resultados, y el uso de

Deep Learning incluso los mejora [5] [6] [7]. La actigrafía puede ser una solución a tener en cuenta puesto que permite recopilar datos de la actividad de un niño sin interferir en la vida del paciente y es un método económico. Teniendo en cuenta esto, este trabajo pretende hacer uso de las técnicas de *Deep Learning* para obtener imágenes de los datos proporcionados por el acelerómetro y usarlos para entrenar una red que determinará si una secuencia de imágenes pertenece a una persona con TDAH o sin este trastorno. EL uso de unidades de procesamiento gráficos, más conocidas como GPUs, permite enseñar a una red neuronal mediante un conjunto de datos de entrenamiento que previamente han sido clasificados como niños sanos o niños con déficit de atención e hiperactividad. Los objetivos de este trabajo son:

- Crear un sistema basado en *Deep Learning* que realice el diagnóstico teniendo en cuenta las características espaciales y temporales de los datos. Para esto será necesario identificar la arquitectura óptima de la red. En base a otros estudios [5] se han demostrado que extrayendo las características espaciales se puede conseguir una alta precisión de diagnóstico. Por tanto, el propósito de este trabajo es considerar también la dependencia temporal y ver si los resultados mejoran.
- Evaluar la precisión del sistema comparándolo con lo existente en la literatura actual.

1.3. Fases y métodos

Para conseguir los objetivos propuestos en la sección anterior y obtener una red neuronal capaz de diferenciar los datos obtenidos de una persona sana o de un niño con trastorno por déficit de atención e hiperactividad es necesario completar las siguientes fases:

- Clasificación de los datos en dos periodos estudiados: actividad diurna y nocturna. Esto permite ver como se distingue la actividad de un niño con TDAH de uno sano, tanto durante sus actividades cotidianas como en etapas de descanso.
- Adaptación de los datos para ser procesados por la red neuronal. La red híbrida está formada por una red convolucional preentrenada y una red recurrente. La red convolucional extrae las características tiempo-frecuencia de tramos de actividad mientras que la recurrente analiza la relación temporal.

- División de los datos en ventanas para estudiar periodos de distinta duración. Se normalizan los datos, se crean las imágenes a partir de espectrogramas extraídos de la señal y se calcula la potencia de cada una de ellas.
- Creación de secuencias de una determinada longitud, se eliminan aquellas secuencias con potencia muy baja, puesto que se corresponden a periodos de inactividad que no aportan información útil. Se adaptan las dimensiones de las imágenes obtenidas mediante espectrogramas a la red preentrenada.
- Obtención de las características de las imágenes mediante la red convolucional preentrenada.
- Diseño de la red recurrente que va a evaluar cómo evolucionan las características calculadas en la fase anterior a lo largo de la secuencia.
- Entrenamiento de la red recurrente con diferentes longitudes de secuencias de imágenes y como clasificador de pacientes.
- Análisis de los resultados obtenidos tanto para el periodo diurno como nocturno.

1.4. Estructura del documento

Este documento está formado por cinco capítulos teniendo en cuenta este capítulo de introducción. A continuación, se describe el contenido de cada uno de ellos.

- **Diagnóstico del TDAH**: es esta sección se explica en qué consiste el trastorno de déficit de atención e hiperactividad, el problema del sobrediagnóstico y las principales pruebas médicas y clínicas que se llevan a cabo. Además, se explica en que consiste la actigrafía y cómo esta técnica se ha utilizado para obtener los resultados de este trabajo.

- **Aprendizaje máquina y *Deep Learning***: en este capítulo se muestra cómo las técnicas de *Deep Learning* permiten diagnosticar si un niño tiene o no TDAH. Se describe el funcionamiento de las redes neuronales, en concreto de las convolucionales y de las recurrentes. Además, se profundiza en la arquitectura de las celdas LSTM.

- **Procesamiento de datos y diseño de las redes neuronales:** en este apartado se describen los datos obtenidos del actígrafo y el procesamiento que se lleva a cabo para obtener las imágenes utilizadas para entrenar la red neuronal. Se explica la fragmentación en ventanas de los datos, así como la técnica del espectrograma. Además, se muestra la red híbrida utilizada para diagnosticar si un niño tiene o no TDAH.

- **Resultados:** en este capítulo se indican los resultados de las simulaciones realizadas para diagnosticar si un paciente pertenece al grupo de Caso o Control.

- **Conclusiones y líneas futuras:** en esta última sección se presentan las conclusiones de este trabajo, así como las posibles líneas de investigación a seguir en base a los resultados obtenidos.

Por último, se incluye un anexo que pretende ilustrar el funcionamiento de la red híbrida. Este apartado incluye el código de Matlab necesario para entrenar la red. También se muestra cómo crear una red con parte convolucional y recurrente.

2. DIAGNÓSTICO DEL TDAH

2.1. ¿Qué es el TDAH?

Según se recoge en [8] en 1902 los médicos *Still y Tredglon* publicaron en el Real colegio de médicos las principales características de este trastorno en un grupo de niños. A partir de 1917 se comienzan a estudiar los síntomas de lo que actualmente se conoce como TDAH, aunque se empleaba el término "Síndrome del daño cerebral" vinculado en aquel momento a una incorrecta educación familiar. En la década de 1950 se empieza a hablar de "Síndrome hipercinético" y "Síndrome del niño hiperactivo" desarrollándose la idea de que el origen de este síndrome no reside en la educación familiar sino en un funcionamiento cerebral incorrecto. El término TDAH se implanta en 1980 definiéndose diferentes tipos en función de la presencia o no de hiperactividad en el paciente. En 1990 se dividen todas las clases establecidas anteriormente en tres subtipos. Además, se estudia su origen y se emplean técnicas de neuroimagen para confirmar biológicamente la presencia de este trastorno. [9]

La sintomatología del TDAH se puede mostrar de muchas formas distintas en función de la edad del niño, así como de otros factores. Además, se debe desarrollar tanto en el ámbito familiar como en el académico. El principal impedimento que experimentan los niños afectados por este trastorno es la dificultad para atender en un ambiente escolar pero también en actividades cotidianas. Teniendo en cuenta diferentes estudios, este desorden es más frecuente entre niños que entre niñas. El origen del TDAH reside en un fallo en el desarrollo de circuitos cerebrales que ayudan al autocontrol.

De acuerdo con el DSM-V [2] *(Manual diagnóstico y estadístico de trastornos mentales de la asociación americana de psiquiatría)* hay tres tipos principales de TDAH:

- **Combinado**: el paciente presenta al menos seis síntomas de atención y seis síntomas de hiperactividad durante un periodo de tiempo no inferior a seis meses.
- **Predominantemente inatento**: el paciente presenta al menos seis síntomas de atención, pero menos de seis síntomas de impulsividad durante un periodo de tiempo como mínimo de seis meses.
- **Predominantemente hiperactivo o impulsivo**: el paciente presenta al menos seis síntomas de hiperactividad, pero menos de seis síntomas de atención durante un intervalo de tiempo no inferior a seis meses.

Para poder comprender esta clasificación hay que conocer que síntomas se corresponden con falta de atención y cuales con hiperactividad. Los comportamientos que reflejan inatención son: dificultad para prestar la atención necesaria a los detalles, a las actividades escolares, así como problemas para escuchar cuando se habla a una persona con este trastorno. Otros síntomas son la complejidad para seguir instrucciones u organizar trabajos, la facilidad para perder cosas necesarias para llevar a cabo actos cotidianos e incluso es frecuente que se olviden de realizar tareas rutinarias. Los comportamientos que indican que un niño tiene hiperactividad son: juega, corre y se levanta continuamente, es incapaz de llevar a cabo tranquilamente actividades lúdicas y habla excesivamente. Otros síntomas son la necesidad de responder impulsivamente, no esperar su turno e interrumpir frecuentemente.

Cabe destacar la importancia de diferenciar entre el TDAH en la infancia y en la edad adulta. Un niño que sufre este desorden se mueve constantemente, no respeta los turnos y tiende a tener conflictos por su incapacidad para controlar sus impulsos. En cambio, una persona adulta con TDAH no tiene por qué moverse continuamente como es el caso de un niño. Además, tiende a realizar los trabajos individualmente, tiene dificultades para delegar y muestra una falta de control emocional.

2.2. Diagnóstico

El diagnóstico del TDAH puede ser realizado por un médico o por un psicólogo clínico. Hoy en día no existe una única prueba que diagnostique este trastorno con un alto porcentaje de acierto. Se realizan principalmente dos tipos de pruebas [10]:

- **Pruebas médicas**: análisis de sangre, medida de la tensión arterial, pruebas de audición y visión.
- **Pruebas clínicas:** entrevista con el niño, los padres y los profesores, uso de criterios diagnósticos y evaluación de la capacidad intelectual.

Otras pruebas cada vez más frecuentes son electroencefalograma (EEG), resonancia magnética (RM) o TAC (tomografía axial computarizada), así como pruebas psicopedagógicas.

El diagnóstico del TDAH requiere tiempo y está formado de varias etapas, que no se pueden omitir. Es necesario conocer la historia clínica del niño, observar su conducta, realizar una exploración física y obtener información de personas del entorno del niño tanto familiar como escolar.

2.2.1. El problema del sobrediagnóstico

El problema del sobrediagnóstico se basa en atribuir este trastorno a personas que realmente no lo tienen. El diagnóstico del TDAH requiere tiempo con el paciente y conocimiento de su entorno familiar y académico. Las principales causas que llevan al sobrediagnóstico se muestran en la figura 1 [11].

Figura 1: Causas del sobrediagnóstico

En lo que concierne a los criterios diagnósticos, el DSM V indica los comportamientos que reflejan la presencia de este desorden, pero pueden ser difíciles de diagnosticar o confundirse con otros. Muchas veces se asigna el TDAH sin cumplir todas las características requeridas olvidando que debe haber un deterioro del rendimiento escolar. Respeto a los criterios clínicos, muchos

profesionales que intentan diagnosticar este trastorno en niños no tienen un conocimiento lo suficientemente amplio. Otra de las principales causas del sobrediagnóstico es fijar los límites de un desarrollo típico, para logar esto se plantean dos preguntas: ¿La conducta del niño es la correspondiente a su edad? y ¿El patrón de conducta es ocasional o persistente? Además, para conocer en profundidad como afecta este desorden es necesario invertir en investigación [4].

Algunas consecuencias del sobrediagnóstico son la sobremedicación, el uso de recursos innecesarios para el apoyo escolar o incluso estrés. Un diagnóstico erróneo puede tener graves consecuencias en un niño por tanto en este trabajo se pretende hacer obtener un sistema capaz de diagnosticar correctamente el TDAH mediante un sistema autónomo.

2.3. *Evaluación*

En esta sección se explican algunas de las pruebas que se realizan para detectar la presencia del TDAH en niños cuyos comportamientos indican que pueden tener este desorden.

2.3.1. *Escala EDAH*

Entre las pruebas clínicas una de las más utilizadas actualmente es la escala EDAH (Evaluación del Trastorno por Déficit de Atención con Hiperactividad) [12] destinada a evaluar a los niños entre 6 y 12 años. Tiene una duración de entre 5 y 10 minutos y la escala está formada por 20 ítems repartidos en dos subescalas: hiperactividad / impulsividad / inatención, y trastornos de conducta. Cada cuestionario debe ser respondido por el profesor del niño. De acuerdo con el DSM-V superar el punto de corte en una de las escalas es suficiente para indicar la presencia de TDAH. Sin embargo, según la opinión de los autores hay que superar el punto de corte en la suma de ambas subescalas. La EDAH es un instrumento de evaluación muy útil como primera aproximación, es simple y requiere poco esfuerzo, pero no demuestra objetivamente la presencia del trastorno de déficit de atención e hiperactividad

2.3.2. *Resonancia magnética*

Una de las pruebas médicas más empleadas para detección del TDAH es la resonancia magnética. Los niños con trastorno por déficit de atención e hiperactividad presentan un retraso madurativo de algunas áreas cerebrales, lo cual

se pretende determinar con esta prueba. Según un niño recibe estímulos se crean nuevas conexiones neuronales, este proceso debe seguir un orden en base a la información, pero si los estímulos son demasiado intensos el sistema nervioso no funciona correctamente. La importancia de esta prueba reside en que es capaz de demostrar médicamente y de forma objetiva la presencia de partes cerebrales con un desarrollo atípico [13].

2.3.3. *Hemoencefalografía*

También conocida como HEG, esta prueba pretende obtener mayor vascularización sanguínea en el lóbulo prefrontal. Se requiere un sensor de luz infrarroja que mide la cantidad de sangre en el córtex prefrontal y un software concreto. A medida que se desarrolla la prueba el paciente aporta una mayor cantidad de oxígeno provocando un aumento de los capilares de la zona estudiada. Una de las características médicas que tiene un niño con TDAH es una baja actividad del lóbulo prefrontal puesto que este se encarga de la organización de actividades, planificación y control de los impulsos y emociones [14].

2.3.4. *Pulsioximetría*

Esta técnica utiliza un sensor situado en el dedo pulgar para comprobar si el paciente presenta una desaturación de oxígeno o hipoxia, esta aparece cuando la saturación de O_2 es inferior al 90%. Si la hipoxia es frecuente puede que el paciente tenga síndrome de apneas-hipopneas de sueño (SAHS), el cual es común entre las personas con ciertos tipos de TDAH. Si la pulsioximetría es negativa no implica que el paciente no tenga trastorno de déficit de atención e hiperactividad, sino que hay que realizar otras pruebas como puede ser la polisomnografía.

2.3.5. *Polisomnografía*

Desde los primeros estudios del TDAH se observaron las diferencias en el sueño de un niño afectado por este trastorno y uno sano. La relación entre el patrón de sueño y este desorden es compleja y depende de múltiples variables. Para poder estudiar esto se utiliza la polisomnografía. Esta prueba monitoriza los estados en los que el paciente esta despierto y dormido. Se estudia el comportamiento del cerebro, de los pulmones y de los ojos especialmente. Los niños con TDAH presentan un sueño

REM no constante con rápidos movimientos oculares. Además, el sueño no es estable, los niños tienden a tener más pesadillas y el síndrome de piernas inquietas es frecuente. Por tanto, es lógico estudiar el comportamiento de un niño con TDAH durante periodos de descanso [15].

2.3.6. *Actigrafía*

Esta prueba usa un sensor, concretamente un acelerómetro que se sitúa en la muñeca o tobillo para almacenar información de la actividad muscular durante un periodo de tiempo. La principal ventaja de esta técnica respecto a las anteriores es que es una prueba no intrusiva. No es necesaria toda la tecnología de una resonancia ni el control médico de una polisomnografía, basta con colocar el acelerómetro y recoger los datos tras un tiempo determinado. Puesto que el sensor también recoge información durante las fases de sueño, esta técnica puede competir con la polisomnografía siendo en ciertas situaciones incluso más útil. [16]

Otras ventajas de la actigrafía o actimetría es el coste económico y el hecho de que solo es necesaria la colaboración del niño. Otras pruebas como la escala EDAH necesitan la participación de múltiples personas de la vida del niño y está sujeta a la subjetividad de estas. En cambio, la actimetría recoge información que describe de forma completamente objetiva el comportamiento del paciente. En base a todos estos motivos se considera la actigrafía uno de los métodos más fiables para diagnóstico del TDAH en la infancia.

2.4. *Estado del arte*

En este apartado se va a analizar diferentes estudios sobre diagnóstico del TDAH llevados a cabo hasta la fecha de hoy. Concretamente se va a profundizar en trabajos basados en la actimetría para ilustrar la eficacia de esta técnica para extraer datos que permiten diagnosticar el déficit de atención e hiperactividad.

En primer lugar, se analiza el estudio realizado por Diego Martín Martínez y colaboradores [16]. Este trabajo se realizó con los datos de 31 Casos y 32 Controles de niños de hospitales públicos y centros de salud de Palencia. Se obtienen los datos mediante actimetria y se procesan dividiéndolos en ventanas de 15 y 30 segundos, así como de 1.5 y 15 minutos. Después se extraen las características, principalmente se analiza el CTM (*central tendency measure*) que aumenta para señales regulares y el SD (*symbolic dinamics*). Para este estudio se obtuvo una precisión de 85.71%, una

sensibilidad de 90.32% y una especificidad del 81.25%. La sensibilidad es la capacidad que tiene el sistema para identificar correctamente a aquellos datos que corresponden a un Caso; es la tasa de verdaderos positivos. La especificidad cuantifica el número de datos que corresponden a un Control, pero se clasifican como Caso, es decir, es la tasa de falsos positivos. La precisión tiene en cuenta tanto la tasa de verdaderos positivos como la tasa de verdaderos negativos. La principal dificultad de este método es que las características no se extraen automáticamente como el caso de las redes profundas. Se seleccionan las características que se emplean para llevar a cabo el diagnóstico.

También se ha consultado el estudio realizado por Patricia Amado Caballero [5] puesto que este trabajo ha contado con su ayuda y parte de los mismos datos. La investigación de Patricia también se basa en la actimetría y utiliza una red convolucional para clasificar representaciones tiempo-frecuencia de la actividad, en Caso (TDAH) o Control (individuo sano). Se obtuvo una precisión de hasta 75.38% en clasificación por imágenes. Puesto que cada paciente tiene un número de imágenes, para diagnosticar se usaron múltiples clasificadores que, en base a todas las imágenes de un Caso o Control, tomaban una decisión. En diagnóstico se obtuvo una precisión de 99.55%, demostrándose de esta forma que la actigrafía junto con *Deep Learning* permiten crear un sistema capaz de detectar de forma eficiente y objetiva el trastorno por déficit de atención e hiperactividad. La principal desventaja de este trabajo es que la red convolucional distingue las imágenes de Caso o Control extrayendo las características espaciales de las imágenes, pero no se tiene en cuenta la dependencia temporal. Es decir, esta red no es capaz de analizar cómo cambian en el tiempo las imágenes de los niños estudiados para poder diferenciar una persona con TDAH de una sana. Además, requiere de clasificadores adicionales a la red para poder diagnosticar.

Estos dos estudios se basan en la actigrafía, pero también existen trabajos con *Deep Learning* que obtiene los datos mediante otras técnicas como las comentadas en este capítulo. Por ejemplo, la investigación realizada por Riaz y colaboradores [17] utilizó la técnica de resonancia magnética funcional (fMRI) para obtener información médica relevante tanto de niños diagnosticados con TDAH como de niños sanos. También se empleó un clasificador SVM (*support vector machine*) para diagnosticar, alcanzando una precisión máxima de 86.7 %.

El trabajo llevado a cabo por Junqiang Du et al [18] se basa en el estudio de las redes cerebrales de los pacientes con TDAH. Este trastorno no sólo se puede detectar en regiones individuales del cerebro sino también en las conexiones entre ellas. Para obtener los datos también se ha empleado la técnica de resonancia magnética funcional (fMRI). Se creó un método de selección de subredes discriminativas para

extraer las subredes que caracterizan tanto a Casos como a Controles. Para realizar esta investigación se utilizaron los datos de 118 Casos y 98 Controles. Con el objetivo de diagnosticar correctamente se empleó un SVM obteniéndose una precisión del 94.91%. Además, este método permite localizar las regiones cerebrales afectadas por el TDAH para su posterior estudio e investigación.

Hay numerosas ventajas al usar técnicas de actigrafía frente a otros métodos. Se trata de una prueba que no requiere elementos muy complejos o costosos y es poco intrusiva en la vida del paciente. Además, permite recoger datos del movimiento de un niño tanto por el día como por la noche, permitiendo así observar las diferencias en ambos periodos. Por otra parte, el uso de técnicas de *Deep Learning* permite extraer las características de forma automática lo cual incrementa la precisión de diagnóstico como se ha podido ver en [5] [16]. En análisis de las relaciones temporales a mayores de las espaciales permite analizar cómo cambian las señales estudiadas a lo largo del tiempo. Esto supone el uso de más información a la hora de diagnosticar. Sin embargo, analizar las dependías temporales añade complejidad puesto que no es sencillo extraer las relaciones espaciotemporales [19].

3. APRENDIZAJE MAQUINA Y DEEP LEARNING

3.1. Deep Learning

El *Deep Learning* es un conjunto de algoritmos de aprendizaje automático (*Machine Learning*) que pretenden modelar abstracciones de alto nivel sobre un grupo de datos [21] [22]. Generalmente estos sistemas usan una serie de capas para extraer y transformar los datos. Cada capa usa como entrada la salida de la capa anterior. Los algoritmos empleados pueden utilizar distintos tipos de aprendizaje, los más comunes son:

- **Aprendizaje supervisado**: es el tipo más sencillo de implementar, se basa en un conocimiento previo (a priori). Se dispone de unos datos (X) y de su salida correspondiente (Y), a partir de estos se pretende buscar la función que mejor se ajuste, tal que $f(X) = Y$. Se logra obtener una estimación de la función f minimizando el error. Este tipo de aprendizaje se usa para clasificación y regresión, es el aprendizaje empleado en este trabajo de fin de grado para entrenar la red.

- **Aprendizaje no supervisado**: en este caso no se tiene conocimiento a priori. Se dispone de los datos (X) pero no de la salida (Y), se estudia la distribución de los datos para aprender más sobre ellos. En este tipo de aprendizaje el modelo se ajusta a los datos observados, las variables de entrada se usan como variables aleatorias. Se considera más subjetivo puesto que no se conoce la salida deseada y se usa principalmente para algoritmos de *clustering* y de asociación. También se combina con la teoría bayesiana de la probabilidad para estudiar las probabilidades condicionadas de los datos.

- **Aprendizaje semisupervisado:** este tipo de aprendizaje es un punto intermedio entre los dos vistos previamente. Se emplean datos (X) de los cuales se conoce su salida (Y) pero también hay otros datos cuya salida es desconocida, se les conoce como datos etiquetados y no etiquetados. Además, es necesario que los dos tipos de datos tengan la misma distribución.

- **Aprendizaje por refuerzo**: en el aprendizaje por refuerzo hay datos no supervisados que reciben refuerzos o realimentaciones. El objetivo es determinar qué acciones debe realizar un agente para maximizar una recompensa o premio. Este agente aprende mediante prueba y error en un ambiente dinámico, en cada iteración el agente recibe el estado actual y selecciona la acción que maximice el refuerzo o recompensa. El proceso de decisión se estudia como un proceso de Markov.

3.2. Redes neuronales artificiales (RNAs)

Las redes neuronales artificiales [21] fueron presentadas en 1943 por el neurofisiólogo *Warren McCulloch* y el matemático *Walter Pitts*. Ambos investigadores presentaron un modelo computacional simplificado de cómo las neuronas del cerebro de los animales trabajaban todas juntas para realizar cálculos complejos.

Las redes neuronales artificiales pretenden simular el funcionamiento de las redes neuronales del cerebro humano. Están formadas por un grupo de neuronas que operan de forma simultánea y síncrona. Al igual que el cerebro de una persona con la información recibida y la experiencia, se crean conexiones y se aprende una tarea determinada. Las neuronas biológicas reciben impulsos eléctricos llamados señales de otras neuronas. Cuando recibe un número suficiente de señales mediante el axón crean su propia señal y la transmite al resto de neuronas de la red. Se ha descubierto que las neuronas se organizan en capas consecutivas, idea que también se ha empleado en las RNAs.

Una de las RNA más simples es el *Perceptron* [21] [22] desarrollado en 1957 y basado en la LTU (*línear treshold unit)*. Su esquema principal se muestra en la figura 13.

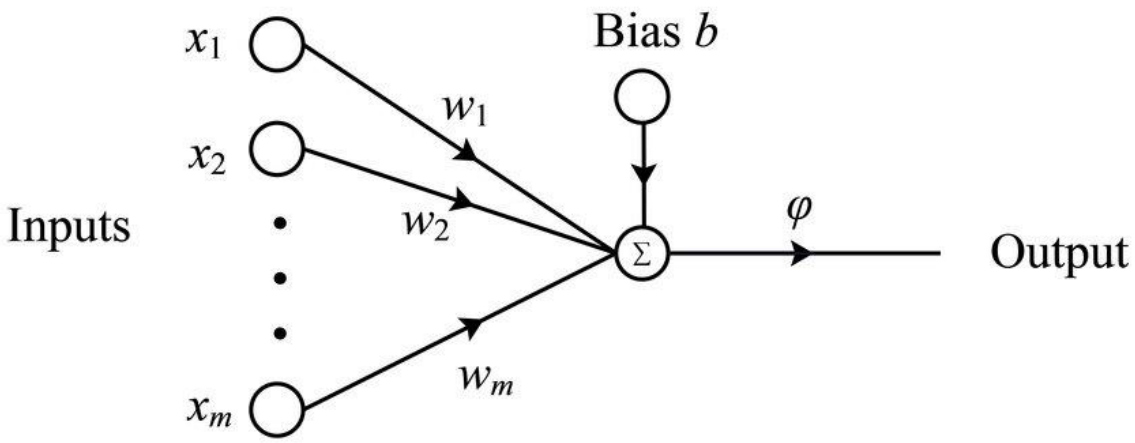

Figura 2: esquema del precepton

A cada una de las entradas x_m se le aplica un peso w_m , posteriormente se suman todas las entradas con sus pesos correspondientes y se aplica la función de activación $y(x)$ para obtener la salida. El término de *bias* se resta directamente a las entradas con sus pesos. La fórmula empleada es:

$$output = y(\sum_{i=1}^{m}(x_i * w_i) - b) \quad (1)$$

Hay muchas funciones de activación [21] [23], pero principalmente se pueden dividir en dos grupos:

- **Funciones lineales:** la más relevante es la función lineal o de identidad. Permite que lo que hay a la entrada aparezca en la salida, si se emplea una red de varias capas con esta función de activación se denomina regresión lineal.

- **Funciones no lineales:** destacan la función umbral, la sigmoide, la hiperbólica y la función ReLu (unidad lineal rectificada). Esta última es muy frecuente en redes convolucionales puesto que permite un aprendizaje rápido en las redes neuronales. Para valores muy negativos el resultado es cero mientras que para valores positivos la entrada es igual a la salida. Cuando la función es cero y su derivada también lo es se produce la muerte de neuronas; esto puede ser un problema por tanto se usa la función *LeakyRelu* que presenta una pequeña pendiente para valores negativos. Para la red recurrente

comúnmente se usa la función de activación sigmoide. Tanto la función ReLu como la sigmoide se pueden ver en la ilustración 14.

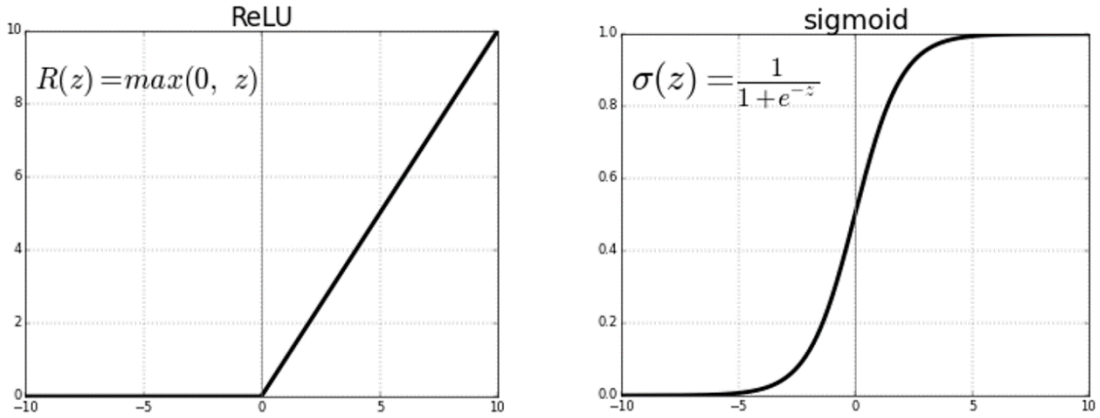

Figura 3: funciones de activación RELU y sigmoide

3.2.1. *Propagación hacia atrás (BackPropagation)*

Para conseguir que la red realice la función buscada hay que entrenarla; durante este proceso se ajustan los pesos. En función del resultado que se obtenga se modifican los pesos intentando minimizar el error. Este error es la diferencia entre la estimación de la función de la red y la función deseada. Los pesos cambian en base a la contribución de cada neurona al resultado. Esta técnica se conoce como *BackPropagation* [22] [21] y consigue que le modelo sea más preciso.

Este método emplea un ciclo de propagación en dos fases:

- En la primera fase la entrada se propaga desde la capa inicial hasta la última. Una vez que se genera la salida se compara con la deseada y se obtiene una señal de error.
- En la segunda etapa la señal de error se propaga desde la capa de salida hasta la inicial pasando por todas las capas ocultas intermedias. Esto se repite hasta que cada neurona recibe una señal de error que indica su aportación a dicho error. De esta forma cada neurona puede saber el error relativo cometido y ajustar sus pesos en consecuencia.

Aunque las redes neuronales existen desde hace décadas, la poca potencia que presentaban hizo que fueran muy poco usadas. Algoritmos como éste y el uso de la GPU permitió llevar a cabo grandes optimizaciones para este tipo de cálculos. Gracias a estas mejoras ha sido posible el desarrollo de *Deep Learning* puesto que requiere redes neuronales de muchas capas que realizan tareas muy complejas y computacionalmente costosas.

3.2.2. *Descenso del gradiente*

El algoritmo que se emplea para lleva a cabo el aprendizaje de la red es el descenso del gradiente [21] . En esta sección se explica cómo funciona.

El descenso del gradiente es un algoritmo de optimización genérico capaz de dar solución a una gran variedad de problemas. La idea general del descenso del gradiente es ajustar los parámetros iterativamente para minimizar una función de coste. El propósito de este algoritmo consiste en calcular el gradiente de la función de error y buscar la dirección en la que este disminuye. Cuando llega al valor de cero se ha alcanzado un mínimo.

Uno de los parámetros más relevantes a configurar es la tasa de aprendizaje (α), si esta tiene un valor muy pequeño serán necesarias muchas iteraciones para que el algoritmo converja por lo que se requerirá mucho tiempo. Por otra parte, si la tasa de aprendizaje es muy grande es posible que se evite el mínimo de la función. Esto provocará que el algoritmo diverja. El efecto de la tasa de aprendizaje se puede ver en la figura 11.

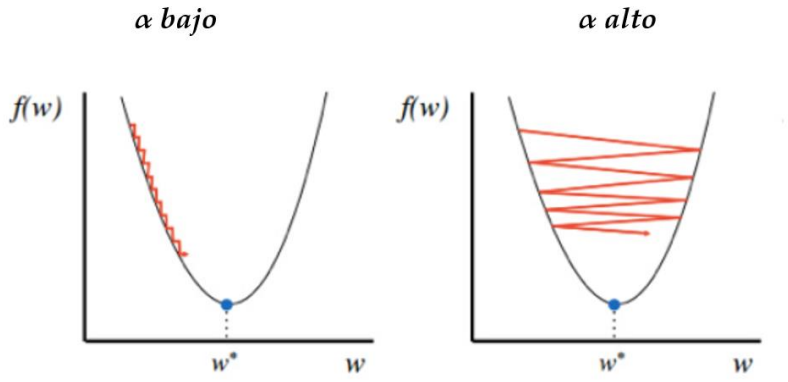

Figura 4: efecto de la tasa de aprendizaje (α) en el algoritmo del descenso del gradiente estocástico

No todas las funciones de coste son como las representadas en la ilustración 11. Hay funciones que presentan mínimos locales que pueden provocar que el algoritmo converja en un mínimo local en lugar de alcanzar el mínimo absoluto de la función de coste, depende de donde se inicialice el algoritmo. Sin embargo, si el problema es de regresión lineal la función de coste no tiene mínimos locales solo un mínimo absoluto. Esto tiene una consecuencia muy relevante, se garantiza que el descenso del gradiente se aproximara a un valor cercano al mínimo global de la función de coste.

Se pueden distinguir tres tipos de descenso del gradiente:

- **Descenso del gradiente de Batch (GD)**: se utiliza para ajustar los pesos de una función que modela el funcionamiento de una aplicación de aprendizaje automático. Se define una función de coste (ecuación 3) y se calcula el gradiente de la función. Con este resultado se ajustan los pesos como se puede ver en la ecuación 4. Siendo w los pesos de la red, x los datos de entrada, y la salida de la red y $\hat{y}$ la salida real del sistema que se pretende modelar.

$$y = w^T * x \quad (2)$$

$$J(w) = \frac{1}{N}\sum_{n=1}^{N} w^T * x_n - \hat{y}_n \quad (3)$$

$$w = w - \alpha \nabla J(w) \quad (4)$$

- **Descenso del gradiente estocástico (SGD):** es una variante del descenso del gradiente de *Batch* que consiste en utilizar solamente un dato en cada paso de optimización. En la ecuación 5 se actualizan los pesos con el gradiente de la función de error considerando un solo dato.

$$w = w - \alpha \nabla j_n(w) \quad (5)$$

- **Descenso del gradiente miniBatch:** en este algoritmo en cada paso de optimización no se usa un solo dato, pero tampoco todos, sino un conjunto de datos. Se busca que el tamaño del lote (*miniBatchSize*) se una potencia de 2 puesto que es más adecuado para el funcionamiento de unidades de procesamiento gráfico, más conocidas como GPU. En este proyecto se han obtenido los mejores resultados con un *miniBatchSize* de 64.

Estas tres variantes del descenso del gradiente tienen sus ventajas y aspectos negativos. El descenso del gradiente de *Batch* consigue una precisión alta, pero requiere mucho tiempo de computación, el SGD en cambio es más rápido puesto que en cada iteración solo trabaja con un dato, pero la precisión es más baja. El descenso del gradiente de *miniBatch* se presenta como una variante con mayor velocidad de convergencia que el GD y una precisión más alta que el SGD.

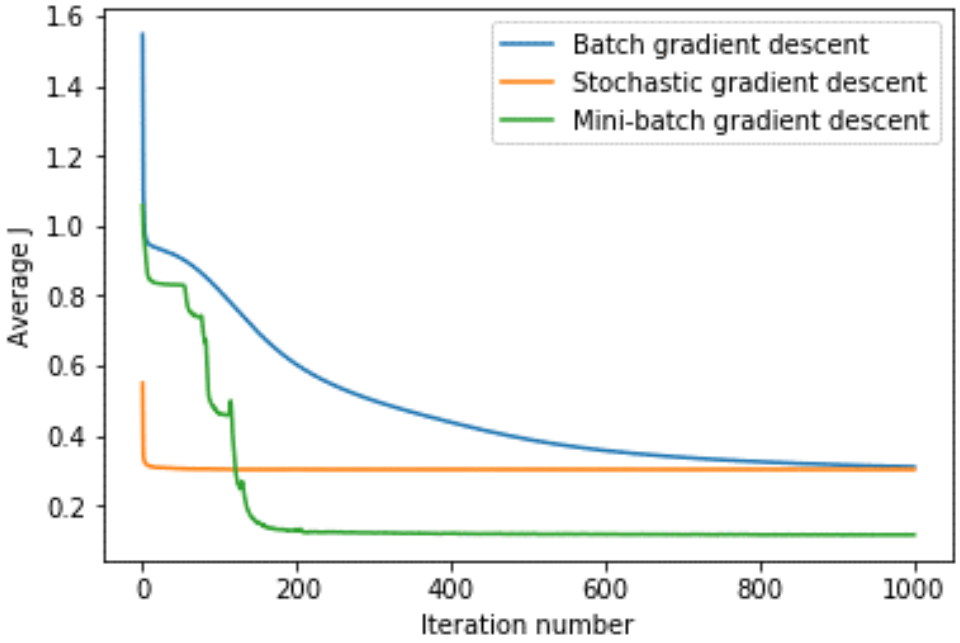

Figura 5: velocidad de convergencia de las tres variantes del descenso del gradiente

En la Ilustración 12 se puede ver como varia la velocidad de convergencia para los tres algoritmos del descenso del gradiente. El descenso del gradiente de Batch requiere muchas iteraciones para converger mientras que el descenso del gradiente estocástico converge tras muy poco tiempo. Para entrenar la red recurrente se utiliza el descenso del gradiente estocástico (SGD) con momento y *miniBatchSize*. Esto permite conseguir una velocidad de convergencia alta con la ventaja de poder determinar el tamaño del lote con el que se entrena en cada iteración.

3.3. *Tipos de redes neuronales*

Existen muchos tipos de redes neuronales artificiales entre ellas el perceptrón simple, el perceptrón multicapa, la red neuronal prealimentada, las redes neuronales convolucionales o las redes recurrentes. En esta sección se van a explicar las redes convolucionales y recurrentes, que son las empleadas en este proyecto para determinar si un grupo de imágenes se etiqueta como Caso o Control.

3.3.1. *Redes neuronales convolucionales (CNN)*

David H. Hubel y Torsten Wiesel llevaron a cabo una serie de experimentos que demostraban que ciertas neuronas reaccionan solo a estímulos ubicados en una región del campo visual mientras que otras neuronas de nivel superior solo

reaccionan a patrones más complejos. Estas observaciones llevaron a la idea de que las salidas de las neuronas de nivel superior se basan en la salida de las neuronas de nivel inferior. Se probó que esta arquitectura es capaz de detectar patrones complejos y se aplicó a las redes neuronales artificiales [21] [24].

Este tipo de redes están formadas por múltiples capas convolucionales, así como por capas de reducción. Esta arquitectura está pensada principalmente para imágenes, aunque también se pueden introducir secuencias. Se utilizan como redes de clasificación, pero también para extracción de características. Este último uso es el que se le da a la red convolucional de este proyecto, de esta forma se extraen las características más relevantes de las imágenes para entrenar la red recurrente.

La red convolucional que se usa para extracción de características cuenta con las siguientes capas:

- **Capa convolucional:** la operación de convolución consiste en aplicar el producto y la suma entre la imagen de entrada y el kernel, obteniéndose así una matriz de características de la señal de entrada. El proceso de convolución se puede apreciar en la ilustración 15. Una vez llevada a cabo esta operación es frecuente aplicar la función de activación *RELU*, explicada en la sección 4.1. [25] [26] [27]

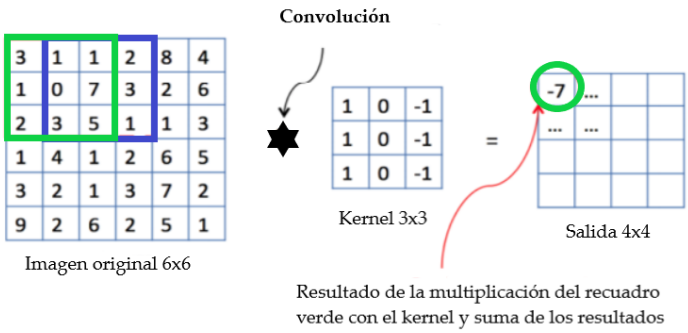

Figura 6: operación de convolución

- **Capa de normalización:** esta capa no modifica las dimensiones de los datos ni realiza ninguna operación compleja. Es un método usado para hacer las redes neuronales mucho más rápidas y estables mediante la normalización de los datos de la capa de entrada. La normalización de *Batch* se consigue mediante iteraciones en las que se fijan las medias y varianzas de los datos de entrada en cada capa. Idealmente se emplearían todos los datos de

entrenamiento, pero en realidad esto no es práctico, por lo que se aplica a pequeños lotes de los datos de entrenamiento.

- **Capa de Maxpooling:** esta capa se utiliza para reducir las dimensiones espaciales de los datos de entrada. En este caso, de las imágenes que entran en la red convolucional. Esta operación también se conoce como reducción de muestreo, puesto que se está eliminando información. Tener un menor tamaño de datos supone menor sobrecarga de cálculo en las próximas capas. La capa *MaxPooling* divide la imagen de entrada en rectángulos y se queda con el máximo valor de cada una de las secciones. Si la imagen de entrada es de dimensiones 129x55 y el kernel es [2,1] la salida de la capa será de tamaño 64x55 reduciéndose cada eje de la imagen en función del kernel indicado. El funcionamiento de esta capa se muestra en la ilustración 16. [25] [28]

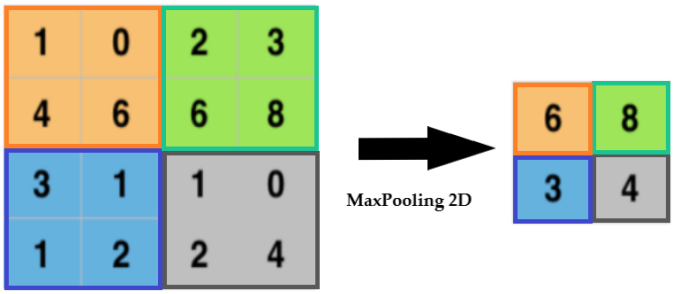

Figura 7: operación MaxPooling

Estas son las capas principales con las que cuenta cualquier red convolucional. Para extraer características se requiere una red convolucional que tras múltiples capas de convolución sea capaz de extraer un número adecuado de características de cada imagen.

3.3.2. *Redes neuronales recurrentes (RNN)*

Es un tipo de red neuronal artificial que presenta bucles de realimentación; esto permite que la información se mantenga durante varias épocas de entrenamiento [22] [29] [30]. Esta clase de redes son eficientes para analizar datos de forma secuencial puesto que permiten recordar la salida anterior como una entrada. Además, pueden analizar cada uno de los elementos de secuencias muy largas. En la ilustración 17 se puede apreciar como en cada instante de tiempo cada neurona recibe la salida de la capa anterior, así como su entrada correspondiente.

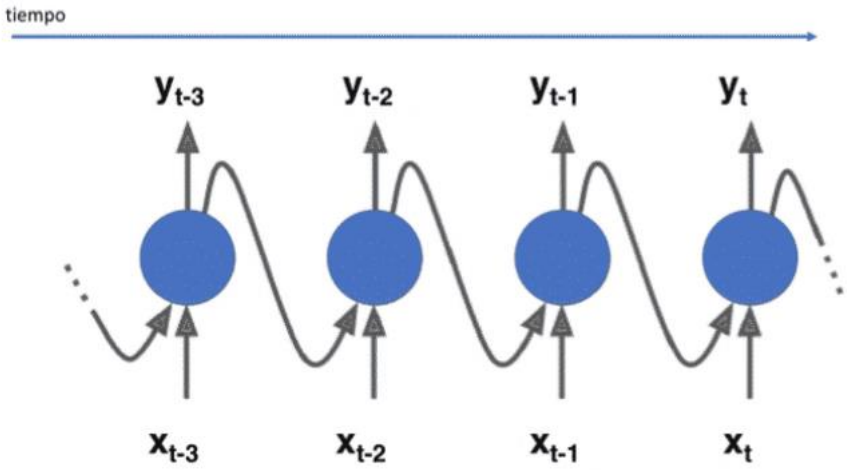

Figura 8: Neuronas de una red recurrente

La función de activación utilizada para producir la salida de cada neurona se aplicaría a las dos fuentes de datos de entrada, haciendo uso de la siguiente ecuación [21]

$$y_t = f(Wx_t + Uy_{t-1}) \qquad (6)$$

Los principales elementos son:

- **La matriz de pesos:** W
- **La entrada en el instante *t:*** x_t
- **La salida de la capa anterior:** y_{t-1}
- **La matriz de pesos que indica el estado de la red en el instante de tiempo anterior:** U
- **La función de activación correspondiente:** $f(x)$

Teniendo en cuenta esto se puede determinar que la salida de una neurona en un instante t es función de las entradas en instantes previos, por tanto, tiene memoria. Este hecho es lo que las hace tan relevantes: pueden recordar características sobre los datos de entrada permitiendo obtener una mayor precisión. Las redes recurrentes se emplean especialmente en traductores automáticos y reconocimiento de voz. En este tipo de redes se hace uso de la técnica de *BackPropagation* para buscar las derivadas parciales del error en base a los pesos (W). Mediante *BackPropagation* se busca ajustar los pesos durante el entrenamiento de la red neuronal recurrente.

3.3.2.1. *Tipos de redes recurrentes*

Los dos tipos más empleados de redes recurrentes son:

- **Las redes LSTM *(Long Short Term Memory):*** están formadas por unidades LSTM y fueron desarrolladas en 1997. El principal problema de las redes recurrentes es que los gradientes usados para *BackPropagation* tienden a desaparecer con el tiempo puesto que no solo depende del error actual sino también de los anteriores. La acumulación de errores da lugar a problemas a la hora de memorizar secuencias a largo plazo. Este problema se solucionó con las redes LSTM que permiten seleccionar qué información debe ser recordada con el tiempo y cual olvidada. Las redes con celdas LSTM se suelen emplear para reconocimiento de texto, voz o de escritura a mano. Este tipo de unidades se ha utilizado en este proyecto y se explican más en detalle en la siguiente sección [31] [32] [33] [34].

- **Las redes GRU *(Gated Recurrent Unit)*:** son un tipo de redes recurrentes creadas en 2014. Las celdas GRU están formadas por una puerta de actualización que indica que información hay que mantener. También disponen de una puerta de reajuste que incorpora los nuevos datos a la información previa de la celda [35] [36].

Comparando ambas redes se puede considerar que las GRU son más simples que las LSTM. Las redes GRU tiene menos parámetros y generalmente son más eficientes en su ejecución. Se ha demostrado que tienen un mejor rendimiento cuando los grupos de datos son más pequeños, sin embrago a medida que las secuencias de datos son mayores las redes LSTM son una mejor opción. Estas redes obtienen mejores resultados en contextos más complejos y con datos más amplios. Uno de los parámetros que más interesa analizar con las redes recurrente que usan celdas LSTM es la longitud de las secuencias.

3.3.2.2. *Celda LSTM (Long Short-Term Memory)*

En este apartado se explica el funcionamiento de este tipo de celdas a partir de su esquema representado en la ilustración 18 [34] [21].

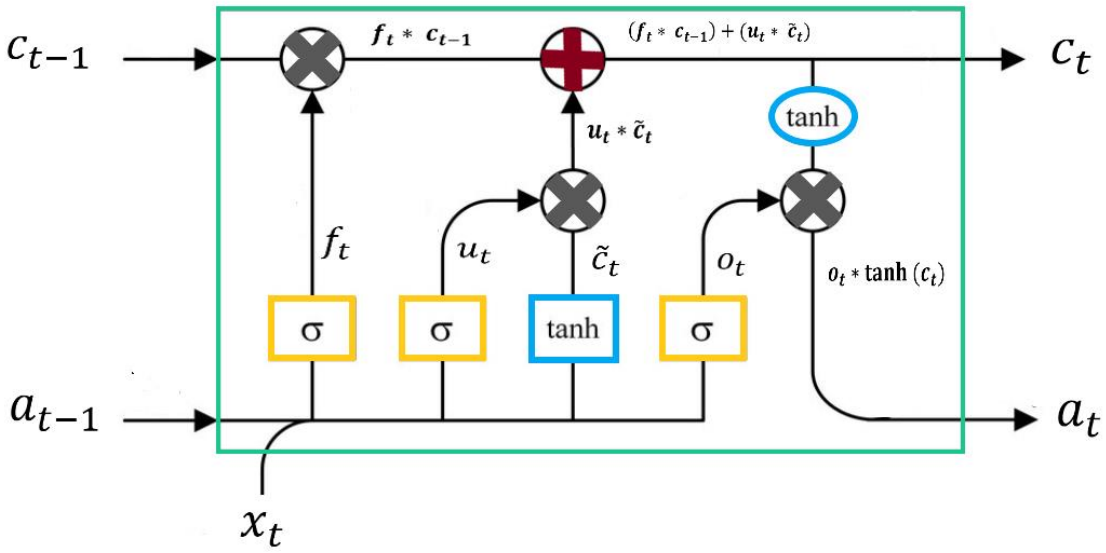

Figura 9: esquema de una celda LSTM

El funcionamiento de las capas LSTM se basa en la incorporación de la celda de estado, que permite añadir o quitar datos de la memoria. Para lograr esto se usan tres puertas o redes neuronales que funcionan como puertas lógicas [36].

- **Puerta de olvido:** permite decidir qué información se va a descartar y por tanto no pasa a la celda de estado. Usa el estado oculto anterior y la entrada actual.
- **Puerta de actualización:** permite actualizar la información de la celda LSTM. Se usan los datos de entrada y el estado oculto anterior.
- **Puerta de salida:** se emplea para calcular el nuevo estado ocultado, es simplemente un filtrado del estado de memoria calculada previamente.

Los datos recibidos por la celda LSTM en el instante de tiempo t son x_t, mientras que a_{t-1} y c_{t-1} hacen referencia al estado oculto y al estado de memoria en el instante anterior.

La salida de la puerta de olvido es $f_t * c_{t-1}$ siendo

$$f_t = sigmoidal(W_f[a_{t-1}, x_t] + b_f) \quad (7)$$

Los coeficientes $W_f y b_f$ se ajustan durante el entrenamiento de la red. f_t se multiplica por el estado de memoria en el instante anterior para eliminar información de la celda de estado.

Después se calcula:

$$\tilde{c}_t = tanh(W_c[a_{t-1}, x_t] + b_c) \quad (8)$$

que incluye los posibles datos que hay que añadir a la memoria de la celda. También es necesario obtener:

$$u_t = sigmoidal(W_i[a_{t-1}, x_t] + b_i) \quad (9)$$

Para saber que parte de $\tilde{c}_t$ hay que incorporar a la celda de estado. Una vez conocidos u_t y $\tilde{c}_t$ ambos se multiplican, obteniendo así la salida de la puerta de actualización. El resultado de esta operación se suma a la salida de la puerta de olvido para conocer el nuevo estado de memoria de la celda, c_t.

$$c_t = (f_t * c_{t-1}) + (u_t * \tilde{c}_t) \quad (10)$$

Para finalizar, hay que conseguir el nuevo estado oculto de la celda, a_t. El resultado de la puerta de salida es:

$$o_t = sigmoidal(W_o[a_{t-1}, x_t] + b_o) \quad (11)$$

Se calcula que partes del estado de memoria forman parte del nuevo estado oculto mediante la operación $tanh(c_t)$. Se filtran los valores del estado de memoria (c_t con el resultado de la puerta de salida (o_t) obteniéndose así el nuevo estado oculto de la celda.

$$a_t = o_t * tanh\,(c_t) \quad (12)$$

.

4. PROCESAMIENTO DE DATOS Y DISEÑO DE LAS REDES NEURONALES

4.1. Señales de entrada

Los datos con los que se va a trabajar en este proyecto provienen de un actígrafo de muñeca (ActiGraph GT3X) formado por un acelerómetro capaz de medir el movimiento del paciente que lo lleva. Como ya se mencionó en el primer apartado de este documento la principal ventaja de esta técnica es su carácter no intrusivo. También permite realizar una evaluación continua de modo que se registra también el movimiento de un paciente durante la noche. Este acelerómetro funciona a una frecuencia de muestreo de 1 Hz permitiendo así obtener un valor numérico del movimiento de un niño cada segundo.

Para este estudio se dispone de un total de 148 pacientes entre 6 y 15 años, de los cuales 73 están diagnosticados con TDAH de acuerdo con el *DSM-V* [2] y se les denomina Casos. Los 75 restantes, no están diagnosticados con este trastorno y se les conoce como Controles. En este proyecto no se va a tener en cuenta el sexo de los pacientes, siendo indiferente si se trata de un niño o niña, pero sí cabe destacar que el 80.8% de los pacientes diagnosticados con TDAH son niños y que el 76.27% de estos tienen una edad inferior a 10 años.

El objetivo de este trabajo es usar una red neuronal implementada con *Deep Learning* para determinar si un niño tiene o no trastorno de déficit de atención e hiperactividad. Para lograrlo, es necesario entrenar la red con imágenes que se crean a partir de los datos obtenidos del actígrafo. El procesamiento que se lleva a cabo con los datos se muestra en la siguiente sección.

4.2. *Adecuación de los datos*

Los cambios que se han realizado en los datos obtenidos del actígrafo se representan en la ilustración 2. En las siguientes secciones se explica cada una de estas etapas.

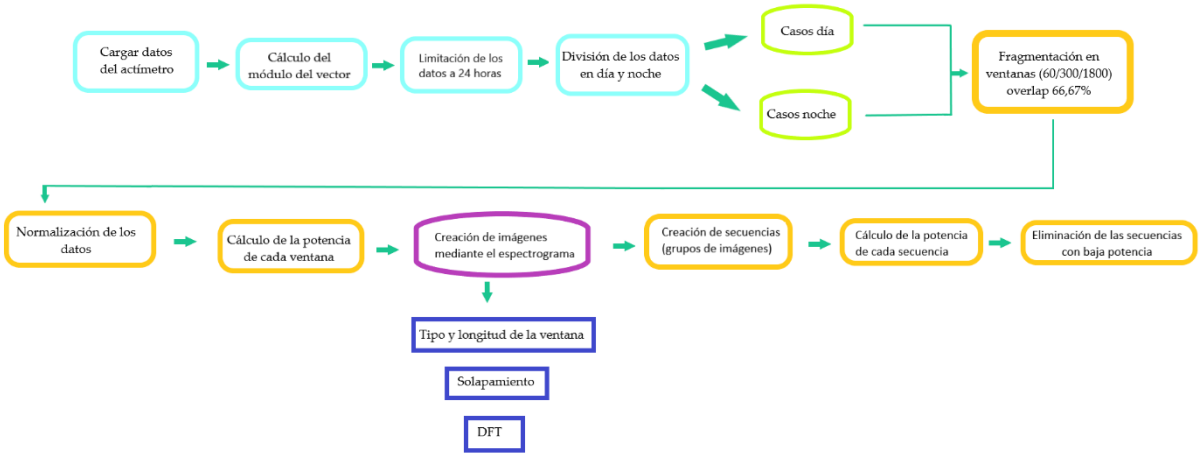

Figura 10: procesamiento de los datos

4.2.1. *Carga de los datos y obtención del módulo del vector*

Inicialmente se leen los ficheros con extensión .data de cada paciente y se almacena en una estructura: los datos del acelerómetro, el número de serie, la fecha y hora en la que se conectó el actígrafo, la fecha y hora de la descarga de los datos y la frecuencia de muestreo. El acelerómetro obtiene un valor de cada eje cartesiano *(x,y,z)* de modo que se calcula el módulo del vector para tener una única medida cada segundo. Se redimensionan los datos para obtener una matriz de tres columnas; los números de cada fila se elevan al cuadrado, se suman y se calcula la raíz cuadrada.

$$A = \sqrt{x^2 + y^2 + z^2} \qquad (13)$$

4.2.2. *Limitación del número de muestras y división en periodo diurno y nocturno*

De acuerdo con la frecuencia de muestreo se obtienen una muestra cada segundo. Se limita el número de datos a los que el acelerómetro puede adquirir en un periodo de 24 horas. En base a esto se reduce el número de muestras a 86400.

Se estudia el movimiento de un Caso o Control en dos etapas diferentes respecto a la actividad diurna o nocturna. Estos dos conjuntos de datos se van a analizar de forma completamente independiente. Para separar los datos en estos dos periodos se aplica una máscara que devuelve como resultado 0, 1 o 2 en función de si un dato corresponde al periodo de día (1 y 2) o de noche (0). El método utilizado para dividir los datos en dos periodos se especifica en [16].

4.2.3. *Fragmentación y normalización de los datos*

Los datos de un Caso o Control se dividen en fragmentos más pequeños, se estudian periodos de actividad corta (1 min), media (5 y 10 min) y larga (30 min). Debido a que la frecuencia de muestreo es de 1 segundo se va a realizar una fragmentación en ventanas de tamaño 60, 300, 600 y 1800 correspondientes a las cuatro actividades mencionadas previamente.

En esta fragmentación se usa un solapamiento del 66,67% [5]; por lo que el *overlap* es de 40 muestras para actividades de corta duración, 200 para actividades de 5 min, 400 para periodos de 10 min y 1200 para actividades largas.

Además, cada una de las ventanas de un Caso o Control se normaliza de modo que el valor máximo que pueden tomar los datos es 1.

4.2.4. *Cálculo de la potencia de cada ventana*

En esta etapa del procesamiento de los datos se calcula la potencia de cada ventana de todos los niños. Si un caso o control tiene 100 ventanas se obtiene un vector de 100 elementos que contiene la potencia de cada ventana. Para calcular la potencia de una ventana de N muestras se usa la siguiente fórmula

$$Potencia\ de\ una\ ventana = \frac{1}{N}\sum_{n=0}^{N-1} |x[n]|^2 \quad (14)$$

Se calcula la potencia de cada ventana para poder obtener posteriormente la potencia de una secuencia. Las secuencias con potencia baja se eliminan, esta reducción de datos no se realiza por ventanas sino por secuencias porque la red recurrente analiza la dependencia temporal, la cual se perdería si se eliminasen imágenes en vez de secuencias.

4.2.5. *Espectrograma*

Una vez normalizados los datos y calculadas las ventanas es necesario transformar las señales 1D en señales 2D, para esto se usa el espectrograma. Consiste en calcular el espectro de ventanas de una señal. Representa la energía del contenido frecuencial de la señal a medida que varía el tiempo, por tanto, los ejes de esta imagen son tiempo y frecuencia.

La función *spectrogram* de la librería *Matlab* [20] realiza los siguientes pasos para obtener las señales en dos dimensiones:

- Se enventanan los datos, para esto se ha utilizado una ventana de *Hamming* de longitud el 10% de la ventana utilizada para la fragmentación. El solapamiento de esta ventana es del 83%.

- De cada una de las ventanas se calcula la transformada de Fourier de tiempo corto o reducido en tiempo discreto (*short time Fourier transform, STFT)*. Su fórmula matemática se muestra a continuación:

$$X(m,\Omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\Omega n} \quad (15)$$

  Donde x[n] es cada uno de los fragmentos de los cuales queremos calcula la STFT y w[n] es la ventana de *Hamming* anteriormente descrita. De cada ventana se realiza la transformada de Fourier de tiempo corto obteniéndose un resultado complejo que se almacena en una matriz.

- Por último, se calcula el valor absoluto del cuadrado de la STFT para obtener el espectrograma. Se aplica la siguiente ecuación:

$$espectrograma\{x(t)\} = |X(m,w)|^2 \quad (16)$$

Los parámetros necesarios para generar el espectrograma se han fijado de forma que las imágenes obtenidas sean de tamaño 129x55x1 independientemente de si se evalúan actividades de 1, 5, 10 o 30 minutos.

4.2.5.1. *Parámetros del espectrograma*

A la hora de realizar el espectrograma para obtener las imágenes que pasarán a la red es necesario fijar una serie de parámetros, los principales son el tamaño y tipo de ventana, el solapamiento y la longitud de la DFT. A continuación, se explica qué valores se han elegido y se muestra cómo afectan a las imágenes obtenidas.

- **Tipo de ventana**: se ha seleccionado una ventana de tipo *Hamming*. Las principales ventajas de esta clase de ventana es que tiene un rizado bajo y presenta una zona de transición reducida. En la ilustración 3 se muestra la ventana usada para el espectrograma en el caso de actividades de larga duración.

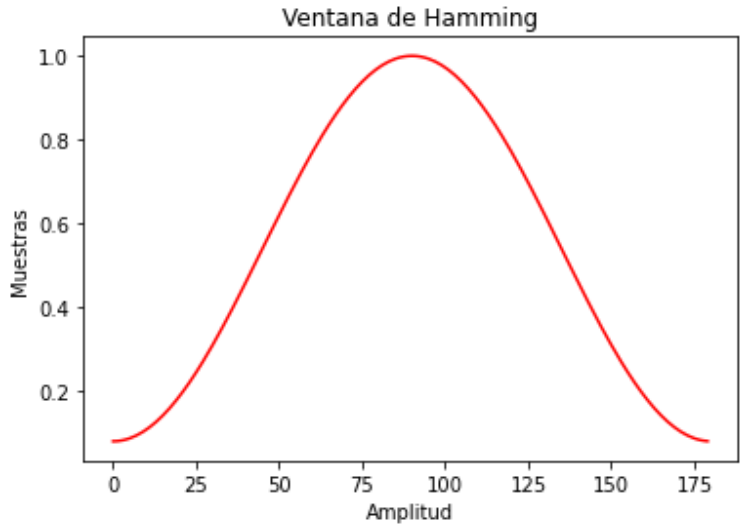

Figura 11: Ventana de Hamming para actividades de 30 min

- **Longitud de la ventana:** la ventana del espectrograma se ha calculado como el 10% de la ventana de fragmentación, con un solapamiento del 83%. En la tabla 1 se muestra la longitud de la ventana de *Hamming* usada para cada tipo de actividad junto con el *overlap*.

Tipo de Actividad	Ventana Espectrograma	Solapamiento
Duración corta	6	5
Duración media	30	25
Duración media	60	50
Duración larga	180	150

Tabla 1: Ventanas empleadas para realizar el espectrograma

En la ilustración 4 se puede ver cómo afecta el tamaño de la ventana a la imagen que se obtiene, así como a la resolución. Se puede observar que para el caso de actividades de 5 min (ventanas de 300 segundos) un tamaño de ventana de 6 segundos da como resultado una imagen de 129x55 mientras que si la ventana es de longitud 100 la imagen resultante tiene unas dimensiones de 129x12. Se puede determinar que al aumentar la ventana del espectrograma la resolución temporal empeora considerablemente. Las imágenes deben tener una resolución y tamaño adecuado para la red neuronal.

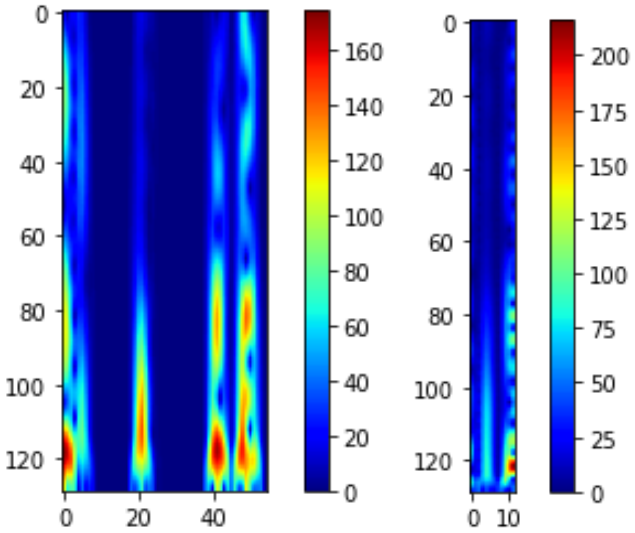

Figura 12: Espectrogramas para actividades de duración media con ventana de longitud 6 (izquierda) y 100 (derecha)

- **Solapamiento de la ventana:** otro de los factores que afecta a las imágenes obtenidas haciendo uso de la técnica del espectrograma, es el solapamiento de la ventana. Se ha podido comprobar que con un solapamiento del 50% se obtiene una imagen de dimensiones 129x19 en cambio si el solapamiento es

del 90% el espectrograma es de 129x91. Se puede afirmar que cuanto mayor sea el *overlap*, hay más ventanas de los datos y por tanto mayor es la imagen resultante. Esta conclusión se puede observar en la ilustración 5.

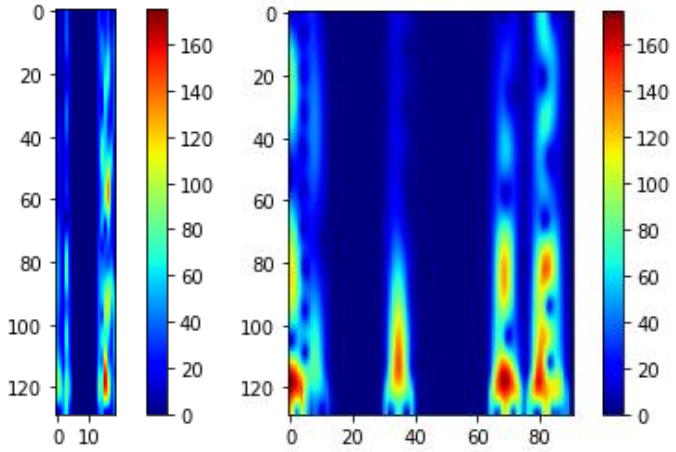

Figura 13: Espectrogramas de actividades de duración media con ventana de tamaño 30 y solapamiento del 50% (izquierda) y 90% (derecha)

- **Tamaño de la DFT:** este parámetro permite cambiar el número de puntos usados para realizar la transformada discreta de Fourier o DFT. Generalmente este número es una potencia de 2, se ha fijado este valor en 256 para obtener imágenes de 129x55, pero en la ilustración 6 se puede ver cómo cambian las dimensiones de los espectrogramas variando este dato. Si la DFT es de tamaño 256 las imágenes presentan un tamaño de 129x55; pero si el tamaño de la DFT es 1024 los espectrogramas son de 513x55.

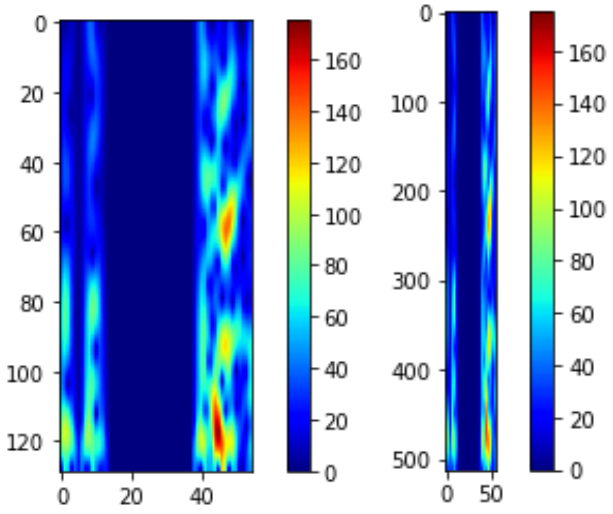

Figura 14: Espectrogramas de actividades de duración media con DFT de tamaño 256 (izquierda) y 1024 (derecha)

Una vez obtenidas las imágenes se agrupan formando secuencias que recibe la red recurrente. En la tabla 2 se muestra un resumen de todas las ventanas utilizadas tanto para la fragmentación como para los espectrogramas de todas las clases de actividades.

Tipo de Actividad	Ventana fragmentación	*Overlap* fragmentación	Ventana Espectrograma	*Overlap* espectrograma
Corta	60	40	6	5
Media	300	200	30	25
Media	600	400	60	50
Larga	1800	1200	180	150

Tabla 2: longitud de las ventanas y solapamiento para fragmentación y espectrograma

4.2.6. *Creación de secuencias y cálculo de la potencia*

Una vez obtenidas todas las imágenes de dimensiones 129x55x1 se forman secuencias de imágenes que pasaran a la red recurrente. Estas secuencias están formadas por 3, 5 y 10 imágenes obteniéndose así matrices de dimensiones 10x129x55x1 en el caso de agrupar las imágenes de diez en diez. Para calcular la potencia de las secuencias se realiza la media aritmética de las potencias de las imágenes obtenidas previamente. Con el propósito de obtener la potencia de una secuencia de N imágenes se hace uso de la siguiente fórmula:

$$Potenciadeunasecuencia = \frac{1}{N}\sum_{n=0}^{N-1} P[n] \qquad (17)$$

Siendo P[n] la potencia de las imágenes que conforman la secuencia.

4.2.7. *Eliminación de secuencias*

Analizando las potencias de las secuencias se puede observar que muchas de ellas presentan una potencia muy baja; por lo que no aportan información útil a la hora de diagnosticar si un niño tiene o no TDAH. Para evitar introducir datos irrelevantes en la red se realiza un histograma de las potencias tanto de Casos como de Controles, estos se muestran en las ilustraciones 7 y 8.

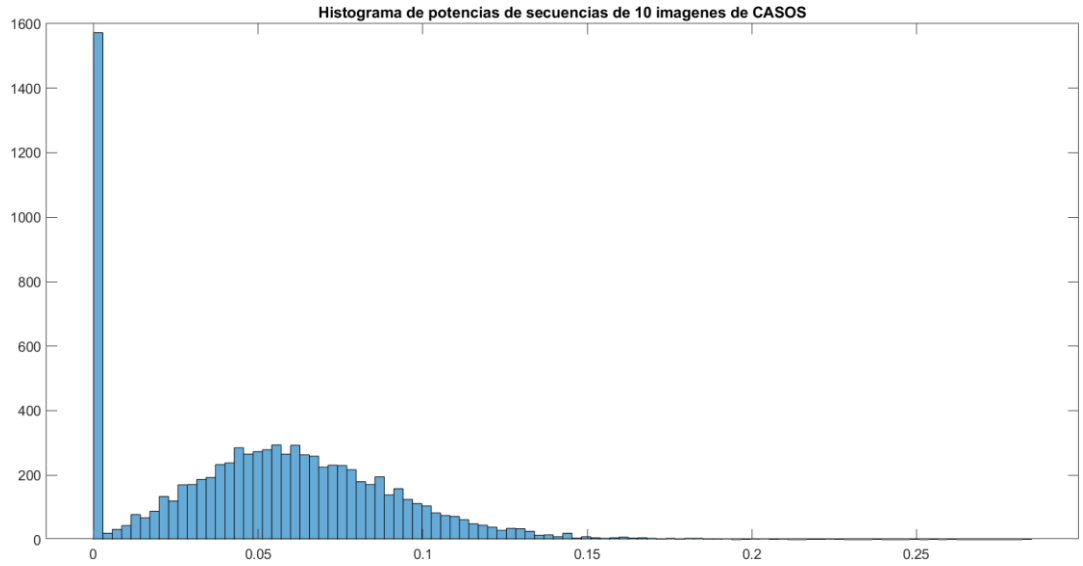

Figura 15: histograma de las potencias de secuencias de 10 imágenes de Casos

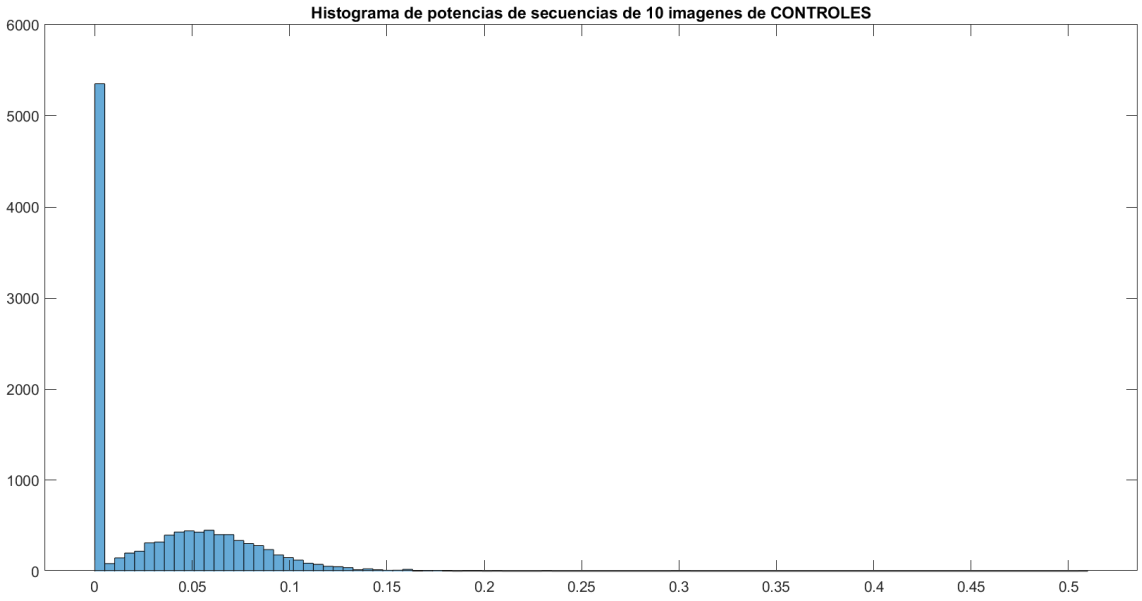

Figura 16: histograma de las potencias de secuencias de 10 imágenes de Controles

Como se puede observar en las dos figuras de esta página hay grupos de imágenes cuya potencia es prácticamente nula. Estas son las secuencias que se van a eliminar. Se establece un umbral inferior de potencia para que todas las secuencias que se introduzcan en la red neuronal tengan una potencia superior a este valor. El límite inferior de potencia se fija en 0.020. Los histogramas obtenidos tras eliminar las secuencias cuya potencia es inferior a este umbral se muestran en las ilustraciones 9 y 10.

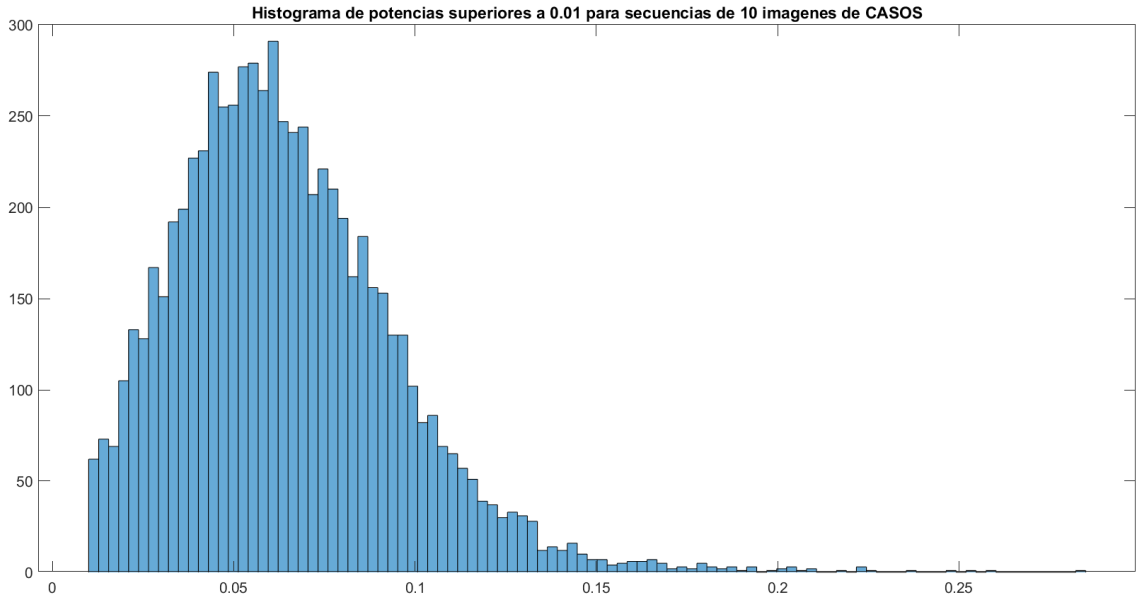

Figura 17: histograma de las potencias de secuencias de 10 imágenes de Casos cuya potencia supera el umbral de 0.020

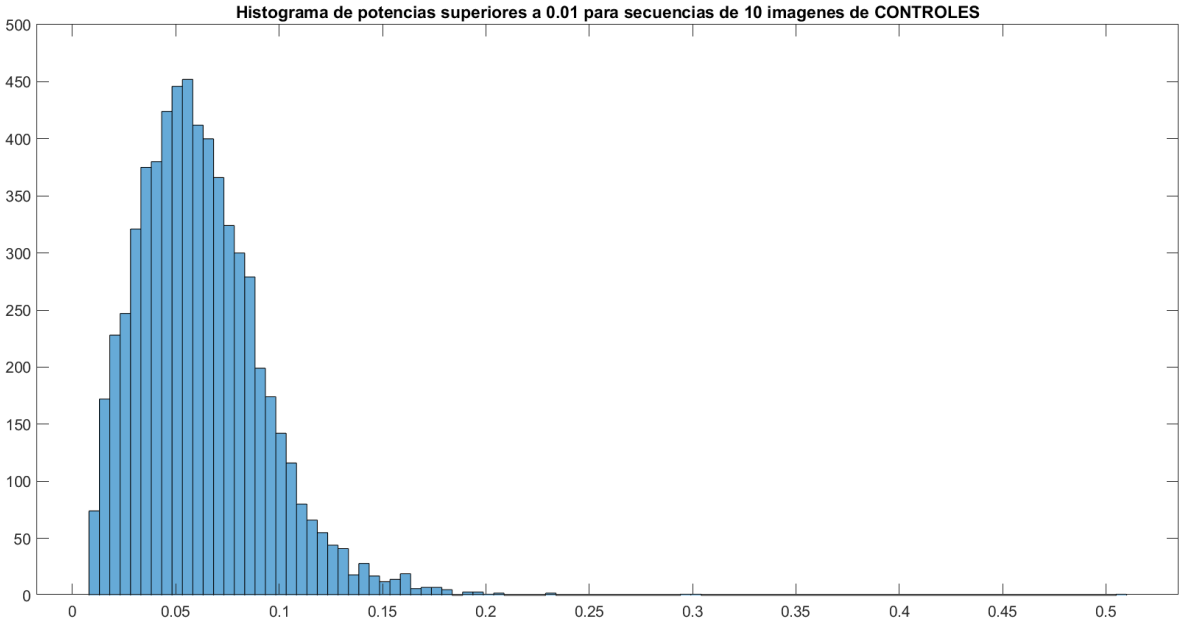

Figura 18: histograma de las potencias de secuencias de 10 imágenes de Controles cuya potencia supera el umbral de 0.020

Tanto para Casos como para Controles se obtienen secuencias cuya potencia media se encuentra aproximadamente entre 0.01 y 0.25. Eliminando las secuencias con potencia baja o en muchos casos nula, se puede entrenar la red con datos que realmente aporten información importante para diferenciar entre un Caso y un Control.

Para finalizar este apartado sobre el procesamiento de datos, se incluyen las tablas 3 y 4 que muestran la cantidad de imágenes obtenidas para cada tipo de actividad durante el periodo nocturno y diurno estudiados.

Tipo de Actividad	Imágenes Casos Dia	Imágenes Controles Dia
Corta (1 min)	180692	165332
Media (5 min)	35989	32915
Media (10 min)	17903	16360
Larga (30 min)	5853	5331

Tabla 3: Número de imágenes para Casos y Controles para cada tipo de actividad durante el día

Tipo de Actividad	Imágenes Casos Noche	Imágenes Controles Noche
Corta (1 min)	120704	127303
Media (5 min)	24004	25320
Media (10 min)	11916	12576
Larga (30 min)	3850	4075

Tabla 4: Número de imágenes para Casos y Controles para cada tipo de actividad durante la noche

4.3. Red híbrida

Con el objetivo de estudiar las características temporales y espaciales de las imágenes se utiliza tanto una red convolucional como una red recurrente, esta red se denomina red hibrida. La CNN se emplea para extraer características de cada una de las imágenes de cada secuencia y la LSTM se usa para evaluar una matriz cuyas dimensiones coinciden con el número de características y la longitud de la secuencia [37] [38] [39] .

Para extraer características de las imágenes se emplea la técnica de *transfer Learning* [40], consiste en hacer uso de redes ya entrenadas para resolver un problema. En este caso se emplea la red *Googlenet* [41] para extraer las características de las imágenes de Casos y Controles. Una vez obtenidas estas características se usan para entrenar a la red recurrente que determina si la secuencia de imágenes introducidas es etiquetada como Caso o Control.

4.3.1. *GoogleNet*

Esta red preentrenada está disponible en Matlab [41] y consiste en una red convolucional formada por 144 capas que es capaz de distinguir entre 1000 objetos. Esta red fue la ganadora de la competición *ImageNet Large Scale Visual Recognition Challenge*, que consistía en crear una red capaz de clasificar imágenes que mostraban distintos objetos. Hace uso de las capas *convolucion2D*, *maxPooling y Relu*, pero también incorpora la capa *Dropout* para evitar *overfitting* y el módulo *inception* que permite a las redes CNN ser más eficientes [42].

Estos módulos fueron diseñados para reducir la carga computacional y resolver el problema del *overfiiting*. Se basan en usar varios tamaños de filtros de kernel dentro de la CNN, pero en lugar de aplicarlos secuencialmente se ordenan para que operen al mismo nivel. El funcionamiento de este módulo se muestra en la ilustración 19.

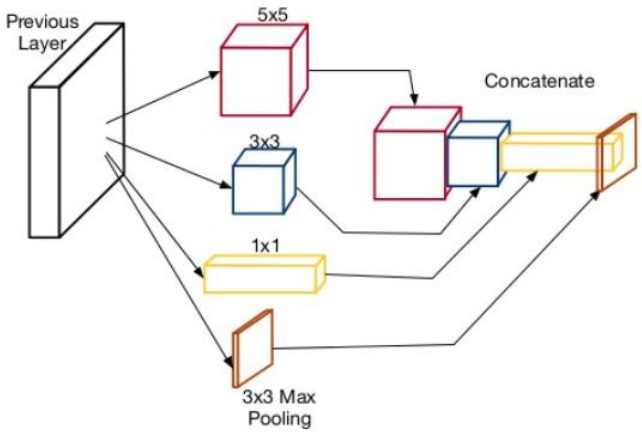

Figura 19: modulo inception

En el caso de GoogleNet se usan tres kernels de tamaño de 3x3, 5x5 y 1x1 además de la capa de Max*pPooling* explicada en la sección 4.3.1. Las salidas se concatenan y se pasan a la siguiente capa.

En la figura 20 se muestran las capas de *GoogleNet*. Con el objetivo de usar la red como extractor de características, se obtienen los datos tras la última capa convolucional, las siguientes capas de la red no se emplean puesto que se estarían clasificando imágenes no secuencias. Por cada imagen se obtiene un vector de 1024 características las cuales se van a utilizar para entrenar la red LSTM.

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Figura 20: capas de Googlenet

4.3.2. *Red LSTM*

Una vez comprendido el funcionamiento de las celdas LSTM y obtenidas las características de las imágenes con la red *GoogleNet*, se utiliza una red recurrente para clasificar las secuencias de imágenes. Por cada imagen se obtiene un vector de 1024 características, de modo que si la secuencia de imágenes está formada por N elementos; los datos de entrenamiento de la red recurrente serán de dimensiones 1024xN. De esta forma se consigue que la red evalúe como varían las características de las imágenes a lo largo de la secuencia de N componentes.

Se trata de una red sencilla que incluye las siguientes capas:

- **BiLSTM:** es una capa LSTM bidireccional [43]. Esto implica que los datos se propagan hacia delante y hacia atrás. Esto permite a estas capas aprender más rápido respecto a las LSTM. El parámetro más relevante que hay que seleccionar es el número de unidades ocultas, que define las dimensiones de un vector de estado interno que afecta al estado oculto de la celda. Tiene una gran importancia porque un valor alto dará lugar a mayor precisión, pero también aumenta la probabilidad de que haya *overfitting*. En cambio, un número bajo de unidades ocultas dará lugar a una red con una precisión baja.

- **Drop**: se utilizan para evitar *overfiting*. La probabilidad de *drop* tiene un valor de 0.5 [44].
- **FullyConected**: se indica el número de clases en las que se pueden etiquetar los datos, para este problema son solo dos, Caso o Control. Esta capa utiliza toda la información de la que dispone para identificar patrones más grandes [45].
- **Softmax**: capa que generalmente se encuentra a continuación de la FullyConnected y se emplea para obtener las probabilidades de cada una de las clases [46].

En la ilustración 21 se muestra un resumen de las capas que conforman la red recurrente empleada.

	Name	Type	Activations	Learnables
1	sequence Sequence input with 1024 dimensions	Sequence Input	1024	-
2	bilstm BiLSTM with 200 hidden units	BiLSTM	400	InputWeigh… 1600×… RecurrentW… 1600×… Bias 1600×1
3	drop 50% dropout	Dropout	400	-
4	fc 2 fully connected layer	Fully Connected	2	Weights 2×400 Bias 2×1
5	softmax softmax	Softmax	2	-
6	classification crossentropyex	Classification Output	-	-

Figura 21: capas de la red recurrente

Para finalizar este capítulo, se incluye una imagen de la estructura de la red híbrida que muestra como las secuencias de imágenes pasan por la red convolucional y tras extraer las características más importantes se entrena la red recurrente. Esta última es capaz de recordar la información más relevante gracias a la capa biLSTM y clasifica las secuencias de imágenes en Caso o Control.

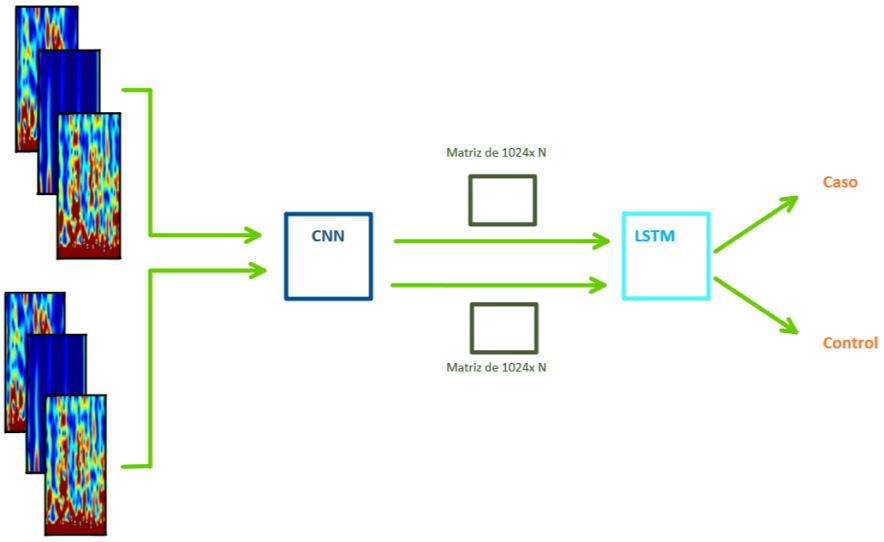

Figura 22: esquema de la red hibrida con N=3

5. RESULTADOS

5.1. Entrenamiento de la red

En esta sección se van a mostrar los resultados obtenidos para las simulaciones realizadas. Se han dividido las secuencias de imágenes en grupos de entrenamiento (80%) y test (20%). Se ha trabajado también con un grupo de validación utilizado para evitar el sobreentrenamiento de los datos. Para crear el conjunto de validación se selecciona el 10% de los datos de entrenamiento.

5.1.1. Parámetros de entrenamiento

Se ha utilizado el algoritmo de entrenamiento *stochastic gradient descent with momentum* o SGDM puesto que es muy eficiente en este tipo de problemas. En este trabajo se han realizado pruebas con otros algoritmos como Adam, obteniéndose siempre mejores resultados con SGDM. Los parámetros fijados para entrenar la red se muestran en la tabla

MiniBatchSize	LearningRate	Momentum	Epochs
64	0.01	0.9	150

Tabla 5: parámetros de entrenamiento

El parámetro *miniBatchSize* indica la longitud de un subgrupo de los datos de entrenamiento que se usa para evaluar el gradiente de la función de pérdidas y actualizar los pesos en cada iteración, como se comentó en la sección 4.1. Respecto al

número de épocas se ha fijado en 150 porque realizando múltiples simulaciones se pudo ver que con este valor la precisión y la tasa de pérdidas se mantenían estables.

5.2. *Resultados para los datos de día*

En este apartado se muestran los resultados obtenidos al entrenar la red recurrente con las características extraídas de *GoogleNet*. Se ha probado con diferentes longitudes de secuencia (3, 5 y 10) hasta llegar a hacer un clasificador de pacientes. Para que la red recurrente clasifique pacientes los datos de entrenamiento tienen unas dimensiones de 1024xN siendo 1024 las características extraídas de cada imagen y N el número de imágenes que se hayan obtenido de cada Caso o Control.

En la siguiente tabla se muestra la precisión de la red para secuencias de distinta longitud para actividades de 5 minutos.

hola

Longitud de la secuencia	Precisión Train	Precisión Test
3	76.00%	55.94%
5	76.72%	56.60%
10	99.98%	56.67%

Tabla 6: resultados de las simulaciones variando la longitud de las secuencias para espectrogramas de 300 segundos

A continuación, se realizan las mismas simulaciones para espectrogramas de 1800 segundos.

Longitud de la secuencia	Precisión Train	Precisión Test
3	76.57%	52.60%
5	76.03%	52.11%
10	73.33%	47.65%

Tabla 7: resultados de las simulaciones variando la longitud de las secuencias para espectrogramas de 1800 segundos

En base a estos resultados se puede ver la importancia de la longitud de la secuencia. Para espectrogramas de 5 min cuanto más larga es la secuencia de imágenes mejor es la precisión tanto de entrenamiento como de evaluación de la red. En el caso de espectrogramas de 1800 segundos hay muchas menos imágenes; por lo que cuanto mayor es la longitud de la secuencia, se dispone de menos datos de entrenamiento. Esto se refleja en precisiones de test inferiores. Para espectrogramas de 60 segundos

se han realizado pruebas con longitudes de secuencias superiores debido a que en este caso hay muchos datos y las secuencias son demasiado largas.

Longitud de la secuencia	Precisión Train	Precisión Test
500	71.95%	60.78%
200	69.52%	58.15%

Tabla 8: resultados de las simulaciones variando la longitud de las secuencias para espectrogramas de 60 segundos

También se han hecho pruebas usando la red como clasificador de pacientes, esto implica que cada secuencia está formada por todas las imágenes que tiene un Caso o Control. En la siguiente tabla se muestran los resultados obtenidos para ventanas de 1, 5, 10 y 30 min.

Tipo de actividad	Número medio de imágenes por Caso o Control	Precisión Train	Precisión Test
Corta (60 seg)	2000	71.33%	57.69%
Media (5 min)	500	96.00%	**78.87%**
Media (10 min)	200	98.33%	69.23%
Larga (30 min)	90	92.11%	53.85%

Tabla 9: precisión de la red como clasificador de pacientes para distintas longitudes de ventanas

Observando la tabla 9 se puede ver que la precisión de la red empeora considerablemente para ventanas de 60 y 1800 segundos; esto demuestra la importancia de la longitud de la secuencia. En el caso de espectrogramas de 60 segundos las secuencias de imágenes son muy grandes (del orden de 2000 imágenes por Caso o Control), lo cual no permite un entrenamiento correcto de la red. Para espectrogramas de media hora ocurre lo contrario, las secuencias son demasiado cortas por lo que la red no tiene suficientes datos como para distinguir un niño con TDAH de uno sano. Se puede determinar que las secuencias de muchas imágenes o muy pocas no son adecuadas para entrenar la LSTM.

En la ilustración 23 se puede ver el progreso del entrenamiento y la tasa de pérdidas a medida que pasan las épocas. La precisión en los datos de entrenamiento crece progresivamente mientras que la tasa de perdidas decrece.

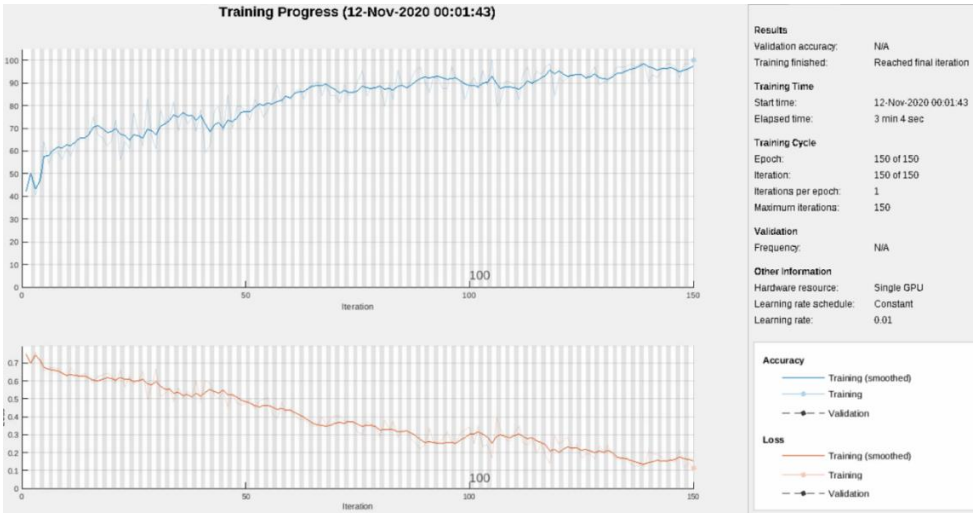

Figura 23: entrenamiento de la red para clasificar pacientes con espectrogramas de 600 segundos

Respecto a la duración del entrenamiento una de las ventajas de la red recurrente es que al operar con las características en lugar de las imágenes procesa menos datos y requiere poco tiempo de entrenamiento. En la siguiente tabla se puede ver los tiempos de entrenamiento para el clasificador de pacientes de la tabla 9.

Tipo de actividad	Tiempo de entrenamiento (min)
Corta (60 seg)	10 min y 15 segundos
Media (5 min)	5 min y 17 segundos
Media (10 min)	3 min y 40 segundos
Larga (30 min)	1 min y 20 segundos

Tabla 10: tiempos de entrenamiento de la red como clasificador de pacientes

5.3. *Resultados para los datos de noche*

Durante el procesamiento de los datos se aplicó una máscara para diferenciar que datos corresponden al periodo nocturno y diurno. Como se puede ver en las tablas 3 y 4 generalmente se dispone menos datos para el periodo nocturno.

Se han realizado las mismas pruebas que con los datos tomados durante el día, agrupando las imágenes en secuencias de 3, 5 y 10. Los resultados han sido muy similares a los de la sección anterior de modo que se utiliza la red como clasificador de pacientes. Los datos de entrenamiento de la red están formados por una matriz de 1024xN siendo N el número de imágenes por pacientes. Los resultados obtenidos se muestran en la siguiente tabla.

Longitud de ventanas para espectrogramas	Número medio de imágenes por Caso o Control	Precisión Train	Precisión Test
60	1400	74.23%	61.54%
300	250	94.17%	70.00%
600	150	95.00%	71.43%
1800	60	95.00%	60.00%

Tabla 11: precisión de la red como clasificador de pacientes para distintas longitudes de ventanas

Puesto que hay menos imágenes para el periodo nocturno se puede ver que la longitud de las secuencias de la tabla de 10 disminuye respecto a la de la tabla 9. Para espectrogramas de 1 y 30 min, el entrenamiento de la red con los datos obtenidos durante la noche mejora ligeramente respecto al periodo diurno. Cabe destacar la importancia de la longitud de las secuencias. Si cada Caso o Control tienen muchas imágenes o muy pocas la red recurrente no es capaz de entrenar correctamente.

6. CONCLUSIONES Y LÍNEAS FUTURAS

6.1. Conclusiones

El TDAH es un trastorno mental que sufren muchos niños y que en muchas ocasiones se diagnostica incorrectamente. Las consecuencias del sobrediagnóstico pueden ser muy negativas para el desarrollo de una persona. Este estudio surge de la necesidad de buscar una forma objetiva de diagnosticar el TDAH.

En lo que respecta al procesamiento de datos se puede concluir que crear espectrogramas de 60, 300, 600 o 1800 segundos afecta directamente al número de imágenes que se obtiene para cada Caso o Control lo cual condiciona a la red. Aunque hay imágenes con potencia nula estas no se eliminan puesto que se va a evaluar la dependencia temporal.

Respecto al entrenamiento de la red, cabe destacar la necesidad de utilizar una red convolucional capaz de extraer características relevantes de las imágenes. Además, se requiere un número de características adecuado; si este es muy alto la red recurrente no es capaz de etiquetar los datos correctamente. La precisión de la red LSTM depende de la longitud de los datos y del número de unidades ocultas de esta capa. Respecto al tamaño de la secuencia, se puede concluir que los resultados empeoran con secuencias excesivamente largas o cortas. El número de unidades ocultas es otro parámetro fundamental, un número muy alto causará *overfit*. La principal ventaja de este sistema frente a otros es que se evalúa la dependencia

temporal y al introducir todos los datos de un paciente se consigue que la red sea capaz de diagnosticar sin la necesidad de usar clasificadores.

Teniendo en cuenta los resultados de la red y las diferentes simulaciones llevadas a cabo se puede concluir:

- El uso de la actimetría para la extracción de los datos junto con técnicas de *Deep Learning* forman un sistema eficiente para diagnóstico del TDAH como se ha indicado en el segundo capítulo.
- Las redes recurrentes permiten estudiar la dependencia temporal de los datos, tal y como se ha podido ver en el apartado 3.3.2.
- Puesto que los datos de entrada son imágenes también es necesario hacer uso de una red convolucional. De este modo surge la necesidad de recurrir a la red híbrida explicada en la sección 4.3.
- La longitud de la secuencia es un parámetro fundamental en las redes LSTM. Como se ha demostrado en el capítulo de resultados, se han hecho pruebas para estudiar que valores de esta variable consiguen incrementan la precisión de la red.

6.2. *Líneas futuras de investigación y limitaciones*

Una vez analizados estos resultados se plantean algunos aspectos que se podrían mejorar para incrementar la precisión de diagnóstico. En primer lugar, convendría buscar una red convolucional capaz de clasificar las imágenes de Casos y Controles con una alta precisión y con muchas capas convolucionales. De este modo el número de características obtenidas tras la última capa convolucional sería adecuado para poder entrenar la LSTM. Este método permitiría conseguir las características principales de las imágenes con una red que ha sido entrenada con el conjunto de datos estudiado. Otro aspecto a tener en cuenta seria añadir múltiples capas LSTM para comprobar si esto permite incrementar la precisión en los datos de test.

Respecto a las limitaciones, cabe destacar que al utilizar la red como clasificador de pacientes el número de datos de entrenamiento se reduce notablemente en comparación con en caso de las secuencias de 3, 5 y 10 imágenes. Una idea que considerar sería entrenar las redes recurrentes con los datos de más niños.

En resumen, los resultados de este trabajo de fin de grado son adecuados, pero probablemente serían mejores con una red convolucional entrenada para estos datos en lugar de usar una preentrenda.

Bibliografía

[1] Fundacion ADAH, «FundacionADAH,» [En línea]. Available: https://www.fundacioncadah.org/web/articulo/tdah-dsm-iv-tr.html. [Último acceso: 21 Noviembre 2020].

[2] American Psychiatric Association, Diagnostic And Statistical Manual Of Mental Disorders, 5ª edicion, 2013.

[3] Wilma, Rivera-Flores Gladys, «Etiología del Trastorno por Déficit de Atención e Hiperactividad y Características Asociadas en la Infancia y Niñez,» *Acta de investigación psicológica,* vol. 3, nº 2, pp. 1079-1091, 2013.

[4] FundacionADAH, «FundacionADAH_sobrediagnostico,» [En línea]. Available: https://www.fundacioncadah.org/web/articulo/el-problema-del-sobrediagnostico-en-tdah.html. [Último acceso: 21 Noviembre 2020].

[5] Patricia Amado-Caballero, P Casaseca de la Higuera, S Alberola López, J Andrés de Llano, JA López Vilalobos, J. Ramón Garmendia-Leiza y C Alberola López, «Objective ADHD diagnosis using convolutional neural networks over Daily-Life activity records,» *IEEE Journal of Biomedical and Health Informatics,* 2020.

[6] M. Muñoz-Organero, L. Powell, B. Heller, V. Harpin and J. Parker, «Automatic extraction and detection of characteristic movement patterns,» *Sensors,* vol. 18, nº 11, p. 3924, 2018.

[7] A. Riaz, M. Asad, E. Alonso, and G. Slabaugh, «Fusion of fMRI and nonimaging data for ADAH classification,» *Computerized Medical Imaging and Graphics* , vol. 65, pp. 115-128, 2018.

[8] FundacionADAH, «FundacionADAH_historiaTDAH,» [En línea]. Available: https://www.fundacioncadah.org/web/articulo/historia-del-tdah.html. [Último acceso: 21 Noviembre 2020].

[9] XavierCastells, RuthCunill, «Trastorno por deficit de atencion e hiperactividad,» *Medicina clínica,* vol. 144, nº 8, pp. 370-375, 2014.

[10] M. L. L. G. C. M. Quintero, «Protocolo de evaluación y diagnóstico del trastorno por déficit de atención e hiperactividad,» *Medicine-Programa de Formación Médica Continuada Acreditado,* vol. 11, nº 86, pp. 5153-5156, 2015.

[11] «Fundacion CADAH,» [En línea]. Available: https://www.fundacioncadah.org/web/articulo/el-problema-del-sobrediagnostico-en-tdah.html. [Último acceso: 12 Agosto 2020].

[12] Trinidad García, Paloma González-Castro, Celestino Rodríguez Pérez*, Marisol Cueli, David Álvarez García, «Alteraciones del funcionamiento ejecutivo en el trastorno por déficit de atención,» *Psicología educativa,* vol. 20, nº 1, pp. 23-32, 2014.

[13] M.J. CarmonaFranceschi, J.L .AscencioLancheros, J.F. Ochoa Gómez, M.T. Rueda Nobmann, J.H. Donado Gómez, L. Blazicevich Carrillo, «Resting state functional magnetic resonance imaging inattention deficit hyperactivity disorde,» *Radiologia,* vol. 62, nº 2, pp. 139-147, 2018.

[14] Paloma González-Castro, Celestino Rodríguez Ángel, López Marisol Cueli, Luis Álvarez, «Attention Deficit Hyperactivity Disorder, differential diagnosis with blood oxygenation, beta/theta ratio, and attention measures,» *International Journal of Clinical and Health Psychology,* vol. 13, nº 2, pp. 101-109, 2013.

[15] Noelia Ruiz Herrera, Alejandro Guillén-Riquelme, Amparo Díaz-Román, Nicola Cellini, Gualberto Buela-Casal, «Sleep among presentations ofAttention-Deficit/Hyperactivity Disorder: Analysis ofobjective and subjective measures,» *International Journal of Clinical and Health Psychology,* vol. 20, nº 1, pp. 54-61, 2020.

[16] D Martín Martínez, P Casaseca de la Higuera, S Alberola López, J Andrés de Llano, JA López, «Nonlinear Analysis Of Actigraphic Signals For The Assessment Of The Attention-Deficit/Hyperactivity Disorder,» *Medical Engineering & Physics,*, vol. 34, nº 9, pp. 1317-1329, 2012.

[17] Atif Riaz, Muhammad Asad, Eduardo Alonso, Greg Slabaugh, «DeepFMRI: End-to-end deep learning for functional connectivity and classification of ADHD using fMRI,» *Journal of Neuroscience Methods,* vol. 335, p. 108506, 2020.

[18] PCAJunqiang Du, Lipeng Wang, Biao Jie, Daoqiang Zhang, «Network-based classification of ADHD patients using discriminativesubnetwork selection and graph kernel PCA,» *omputerized Medical Imaging and Graphics,* vol. 52, pp. 82-88, 2016.

[19] Guangyin Jin, Qi Wang, Cunchao Zhu, Yanghe Feng, Jincai Huang, Xingchen Hu, «Urban Fire Situation Forecasting: Deep sequence learning with,» *Applied Soft Computing,* p. 106730, 2020.

[20] MathWorks, «Matlab_spectrogram,» [En línea]. Available: https://es.mathworks.com/help/signal/ref/spectrogram.html. [Último acceso: 10 septiembre 2020].

[21] A. Géron, Hands-on Machine Learning with Scikit-Learn & Tensorflow, Sebastopol, California : O'Reilly Media, Inc., 2017.

[22] F. Chollet, Deep Learning with python, Shelter Island, New York: Maining Publications, 2018.

[23] LongweiWang, Chengfei Wang, YupengLi, Rui Wang, «Explaining the Behavior of Neuron Activations in Deep Neural Networks,» *Ad Hoc Networks,* p. 102346, 2020.

[24] MathWorks, «Matlab_CNN,» [En línea]. Available: https://es.mathworks.com/solutions/deep-learning/convolutional-neural-network.html. [Último acceso: 15 Septiembre 2020].

[25] D. Calvo, «CNN,» Diego Calvo, [En línea]. Available: https://www.diegocalvo.es/red-neuronal-convolucional/. [Último acceso: 11 Noviembre 2020].

[26] MathWorks, «Matlab_convolution2D,» [En línea]. Available: https://es.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.convolution2dlayer.html. [Último acceso: 11 Septiembre 2020].

[27] Benesty, J., Chen, J., & Huang, Y., «Automatic Speech Recognition: A Deep Learning Approach.,» 2008.

[28] MathWorks, «Matlab_MaxPooling,» [En línea]. Available: https://es.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.maxpooling2dlayer.html. [Último acceso: 20 septiembre 2020].

[29] Wennian YuIl, Yong Kim, Chris Mechefske, «Analysis of different RNN autoencoder variants for time seriesclassification and machine prognostics,» *Mechanical Systems and Signal Processing,* vol. 149, p. 107322, 2019.

[30] A. Sherstinsky, «Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network,» *Physica D: Nonlinear Phenomena* , vol. 404, p. 132306, 2020.

[31] Wen Yu, Jesus Gonzalez, Xiaoou Li, «Fast training of deep LSTM networks with guaranteed stability fornonlinear system modeling,» *Neurocomputing,* vol. 422, pp. 85-94, 2019.

[32] Xiao Zheng, Wanzhong Chen, «An Attention-based Bi-LSTM Method for Visual Object Classification via EEG,» *Biomedical Signal Processing and Control,* vol. 63, p. 102174.

[33] Farah Shahid, Aneela Zameer, Muhammad Muneeb, «Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM,» *Chaos, Solitons & Fractals,* vol. 140, p. 110212, 2020.

[34] A. Graves, Supervised Sequence Labelling with Recurrente Neural Networks, Toronto, Canada, 2011.

[35] Rahul Dey, Fathi M. Salem, «Gate-Variants of Gated Recurrent Unit (GRU) Neural Networks,» *IEEE 60th international midwest symposium on circuits and systems (MWSCAS),* pp. 1597-1600, 2017.

[36] Diego Calvo, «Red neuronal recurrente,» [En línea]. Available: https://www.diegocalvo.es/red-neuronal-recurrente/. [Último acceso: 2 Noviembre 2020].

[37] Yonsei University, Department of Computer Science, Republic of Korea, «Combinatorial feature embedding based on CNN and LSTM for biomedical name identity recognition,» de *Journal of Biomedical Informatics*, Korea.

[38] Aichun Zhu, Qianyu Wu, Ran Cui, Tian Wang, Wenlong Hang, Gang Hua, Hichem Snoussi, «Exploring a rich spatial–temporal dependent relational model forskeleton-based action recognition by bidirectional LSTM-CNN,» *Neurocomputing,* vol. 414, pp. 90-100, 2020.

[39] Md. Zabirul Islam, Md. Milon Islam, Amanullah Asraf, «A combined deep CNN-LSTM network for the detection of novel coronavirus (COVID-19) using X-ray images,» *Informatics in Medicine Unlocked* , vol. 20, p. 100412, 2020.

[40] Zitong Wan, Rui Yang, Mengjie Huang, Nianyin Zeng, Xiaohui Liu, «A review on transfer learning in EEG signal analysis,» *Neurocomputing,* vol. 421, nº 2020, pp. 1-14, 2020.

[41] MathWorks, «MathLab_googleNet,» [En línea]. Available: https://es.mathworks.com/help/deeplearning/ref/googlenet.html. [Último acceso: 5 Noviembre 2020].

[42] Jahandada, Suriani Mohd Sama, Kamilia Kamardinb, Nilam Nur Amir Sjarif b, Norliza Mohamed, «Offline Signature Verification using Deep Learning Convolutional Neural Network (CNN) Architectures GoogLeNet Inception-v1 and Inception-v3,» *Procedia Computer Science,* vol. 161, pp. 475-483, 2019.

[43] MathWorks, «Matlab_biLSTM,» [En línea]. Available: https://es.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.bilstmlayer.html. [Último acceso: 5 Noviembre 2020].

[44] MathWorks, «Matlab_Dropout,» [En línea]. Available: https://es.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.dropoutlayer.html. [Último acceso: 5 Noviembre 2020].

[45] MathWorks, «Matlab_fullyConnected,» [En línea]. Available: https://es.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.fullyconnectedlayer.html. [Último acceso: 11 Noviermbre 2020].

[46] MathWorks, «Matlab_softmax,» [En línea]. Available: https://es.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.softmaxlayer.html. [Último acceso: 10 Noviembre 2020].

[47] Wentao Wang, Canhui Liao, Quan Cheng, Pengju Wang, «Mixed Convolutional Recurrent Neural Networks: Learning sor Signal Classifications,» *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing,* pp. 1-6, 2019.

7. Anexo: Uso de GoogleNet con una red LSTM y creación de una red convolucional y recurrente

7.1. Uso de GoogleNet y entrenamiento de la red LSTM con Matlab

En este anexo se muestra el código utilizado para obtener las características de la red preentrenada y como se usan estas para entrenar la red recurrente. Se realizan los siguientes pasos:

- Se extraen las características de GoogleNet, tanto las de Casos como las de Controles

```
clear;
clc;

net = googlenet;
inputSize = net.Layers(1).InputSize;
analyzeNetwork(net);
rutaCasos=[pwd '/Imagenes/Casos'];
layer = 'pool5-7x7_s1';

featuresCasos={};
numCasos=0;
for i=1:1:73
```

```
    rutaCaso=[rutaCasos '/Caso' int2str(i)];

    rutaImagenes={};
    if(length(dir(rutaCaso))~=0)
        for j=1:1:length(dir(rutaCaso))-2
            rutaImagen=[rutaCaso '/Imagen' int2str(j) '.png'];
            rutaImagenes{j,1}=rutaImagen;
        end
        imds=imageDatastore(rutaImagenes);
        imds2= augmentedImageDatastore(inputSize(1:2),imds);
        numCasos=numCasos+1;
        featuresCasos{numCasos,1}=
activations(net,imds2,layer,'OutputAs','columns');
        featuresCasos{numCasos,2}='Caso';
        disp(['Añadidas caracteristicas de caso ' int2str(i)]);
    end

end

featuresControles={};
numControles=0;
rutaControles=[pwd '/Imagenes/Controles'];
for i=1:1:75  %length(ind_train2)-2
    rutaControl=[rutaControles '/Control' int2str(i)];

    rutaImagenes={};
    if(length(dir(rutaControl))~=0)
        for j=1:1:length(dir(rutaControl))-2
            rutaImagen=[rutaControl '/Imagen' int2str(j) '.png'];
            rutaImagenes{j,1}=rutaImagen;
        end
        imds=imageDatastore(rutaImagenes);
        imds2= augmentedImageDatastore(inputSize(1:2),imds);
        numControles=numControles+1;
        featuresControles{numControles,1}=
   activations(net,imds2,layer,'OutputAs','columns');
        featuresControles{numControles,2}='Control';
        disp(['Añadidas caracteristicas del control ' int2str(i)]);
    end
end


save('Features.mat', 'featuresCasos', 'featuresControles');
```

- Se dividen los datos en grupos de entrenamiento y Test

```
Train={};
numTrain=0;
```

```
for i=1:1:58

    numTrain=numTrain+1;
    Train{numTrain,1}=featuresCasos{i,1};
    Train{numTrain,2}='Caso';

end
for i=1:1:58

    numTrain=numTrain+1;
    Train{numTrain,1}=featuresControles{i,1};
    Train{numTrain,2}='Control';

end

Test={};
numTest=0;
for i=59:1:72

    numTest=numTest+1;
    Test{numTest,1}=featuresCasos{i,1};
    Test{numTest,2}='Caso';

end
for i=59:1:72

    numTest=numTest+1;
    Test{numTest,1}=featuresControles{i,1};
    Test{numTest,2}='Control';

end
Train_datos=Train(:,1);
Train_labels=categorical(Train(:,2));
Test_datos=Test(:,1);
Test_labels=categorical(Test(:,2));
```

- Se define la red LSTM

```
numFeatures = size(featuresCasos{1,1},1); %1024
numClasses = 2;

layers = [
    sequenceInputLayer(numFeatures,'Name','sequence')
    bilstmLayer(200,'OutputMode','last','Name','bilstm')
    dropoutLayer(0.5,'Name','drop')
    fullyConnectedLayer(numClasses,'Name','fc')
    softmaxLayer('Name','softmax')
    classificationLayer('Name','classification')];
```

- Se defienden las opciones de entrenamiento

```
miniBatchSize = 64;
numObservations = numel(Train_datos);
numIterationsPerEpoch = floor(numObservations / miniBatchSize);

options = trainingOptions('sgdm',...
   'MiniBatchSize',miniBatchSize, ...
   'MaxEpochs', 150,…
   'InitialLearnRate',1e-4, ...
   'GradientThreshold',2, ...
   'Shuffle','every-epoch', ...
   'Plots','training-progress', ...
   'Verbose',true, ...
   'ExecutionEnvironment','gpu',...
   'InitialLearnRate',0.01);
```

- Se entrena la red y se evalúan los resultados obtenidos

```
[netLSTM,info] = trainNetwork(Train_datos, Train_labels, layers,options)

predicted=classify(netLSTM, Train_datos);
real=Train_labels;
accuracyTrain = sum(predicted==real)/numel(real)
predicted2=classify(netLSTM, Test_datos);
real2=Test_labels;
accuracyTest = sum(predicted2 ==real2)/numel(real2)
```

7.2. *Diseño de una única red con una parte convolucional y otra recurrente con Matlab*

Otra de las pruebas que se ha realizado es crear una red tanto con parte convolucional como con parte recurrente y entrenar ambas. Para la red convolucional se ha utilizado la red diseñada en el trabajo de Patricia Amado Caballero *[5]*. Después se incorpora la red recurrente. El código de Matlab se muestra a continuación.

```
numClasses = 2;
inputSize = [129 55 1];

layers = [ ...
   sequenceInputLayer(inputSize,'Name','input')
   sequenceFoldingLayer('Name','fold')
```

```
    convolution2dLayer([10 10],32,'Padding',0,'Name','Convolucion 1')
    batchNormalizationLayer('Name','BatchNormalization 1')
    reluLayer('Name','Relu 1')
    maxPooling2dLayer([2 1],'Stride',[2 1],'Name','MaxPooling 1')
    dropoutLayer(0.7,'Name','Drop 1')

    convolution2dLayer([10 10],64,'Padding',[0 0],'Name','Convolucion 2')
    batchNormalizationLayer('Name','BatchNormalization 2')
    reluLayer('Name','Relu 2')
    maxPooling2dLayer([2 1],'Stride',[2 1],'Name','MaxPooling 2')
    dropoutLayer(0.5,'Name','Drop 2')

    convolution2dLayer([10 10],128,'Padding',[0 0],'Name','Convolucion 3')
    batchNormalizationLayer('Name','BatchNormalization 3')
    reluLayer('Name','Relu 3')


    sequenceUnfoldingLayer('Name','unfold')
    flattenLayer('Name','flatten')

    lstmLayer(200,'OutputMode','last','Name','lstm')

    fullyConnectedLayer(numClasses, 'Name','fc')
    softmaxLayer('Name','softmax')
    classificationLayer('Name','classification')

    ];

lgraph = layerGraph(layers);
lgraph = connectLayers(lgraph,'fold/miniBatchSize','unfold/miniBatchSize');
```

El objetivo de esta red es que realice las operaciones convolucionales sobre cada una de las imágenes de la secuencia, almacene los datos en un vector y forme un matriz con todos los resultados de las imágenes de dicha secuencia. Para lograr esto es necesario utilizar tres capas:

- **Flatten:** convierte los datos obtenidos tras la tercera etapa convolucional en un vector. Esta operación es necesaria para que a la salida de la red convolucional haya una matriz de CxN siendo C el número de características y N la longitud de la secuencia.
- **SequenceFoldingLayer:** se utiliza para aplicar las operaciones convolucionales a las imágenes de forma independiente. Convierte grupos se secuencias de imágenes en un conjunto de imágenes.
- **SequenceUnFoldingLayer:** restablece la estructura de la secuencia tras la capa de *Folding*.

Aunque es posible crear una única red con una parte convolucional y otra recurrente, como la mostrada en esta sección. El número de características que llegan a la LSTM es muy grande (C=57344), esto hace que la red no entrene correctamente y se obtiene una precisión de test del 55%. Debido a esto se ha recurrido a la *Googlenet* puesto que el número de características que es capaz de obtener de las imágenes (C=1024), permite entrenar la red recurrente correctamente.

Como se ha indicado en la sección 5.5 una futura línea de investigación podría centrarse en conseguir una CNN con más capas convolucionales que permitiera obtener un número de características adecuado para entrenar la LSTM.