UNIVERSIDAD DE VALLADOLID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

MÁSTER EN INGENIERÍA DE TELECOMUNICACIÓN

DETECCIÓN AUTOMÁTICA DE SPINDLES EN NIÑOS CON APNEA OBSTRUCTIVA DEL SUEÑO MEDIANTE TÉCNICAS DE DEEP LEARNING

Autor:

Dña. Victoria Pacho Velasco

Tutores:

Dr. D. Fernando Vaquerizo Villar

Dr. D. Roberto Hornero Sánchez

Valladolid, 15 de Septiembre de 2023

TÍTULO:	**Detección automática de *spindles* en niños con apnea obstructiva del sueño mediante técnicas de *deep learning***
AUTOR:	**Dña. Victoria Pacho Velasco**
TUTORES:	**Dr. D. Fernando Vaquerizo Villar** **Dr. D. Roberto Hornero Sánchez**
DEPARTAMENTO:	**Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

TRIBUNAL

PRESIDENTE:	**Dra. Dña. María García Gadañón**
VOCAL:	**Dr. D. Ramón de la Rosa Steinz**
SECRETARIO:	**Dra. Dña. Miriam Antón Rodríguez**
PRESIDENTE SUPLENTE:	**Dr. D. Salvador Dueñas Carazo**
SECRETARIO SUPLENTE:	**Dr. D. Ignacio de Miguel Jiménez**
VOCAL SUPLENTE:	**Dr. D. Luis Miguel San José Revuelta**

Agradecimientos

La culminación de este TFM no sólo marca para mi la finalización de un proyecto, sino que también simboliza el cierre de mi etapa como estudiante y el comienzo de una nueva fase en mi vida. El haber llegado hasta aquí no ha sido trivial, por ello debo agradecer a diversas personas que han contribuido en mi desarrollo como ingeniera de telecomunicaciones de manera directa e indirecta.

En primer lugar, me gustaría expresar mi sincero agradecimiento a mis tutores Fernando y Roberto, por su invaluable apoyo y orientación durante todo el proceso de mi TFM. Han mostrado disponibilidad e interés durante la evolución del proyecto a la vez que han sido clave a la hora de resolver mis dudas en todo momento.

Por otro lado, no me puedo olvidar de todos los que de una forma u otra me han apoyado de manera incondicional y que han sido esenciales a lo largo de toda mi carrera. Gracias a todos mis amigos por darme la fuerza que necesitaba, especialmente a Lausta, Paula y Carretero, los cuales me han sabido escuchar y acompañar durante este tiempo. Además, debo agradecer a la persona que ha sido hogar, cariño y luz a lo largo de todos estos años. Gracias Andrés de corazón por todo lo que he aprendido y vivido contigo.

Por último, pero no menos importante, agradecer el apoyo de mi familia, la cual me ha dado la seguridad y confianza que en todo momento necesitaba. Gracias por todos los valores que me habéis dado y que han sido indispensables para convertirme en la persona que hoy soy. Sin vosotros no hubiera podido llegar hasta aquí, y sois la base de toda mi vida.

Resumen del TFM

La Apnea Obstructiva del Sueño (AOS) infantil es un trastorno muy prevalente a nivel mundial (1-5% de la población), que tiene graves consecuencias en la salud y afecta el desarrollo de los niños afectados. Gracias a los nuevos avances tecnológicos, se ha observado que pequeñas oscilaciones presentes fundamentalmente durante las fases N2 y N3 del sueño entre 11-16 Hz, conocidas como *spindles* del sueño. Estos *spindles* están íntimamente relacionadas con el proceso cognitivo de las personas en general y de los niños en particular. Este descubrimiento abre una nueva línea de investigación orientada a desarrollar algoritmos que detecten automáticamente *spindles* empleando para ello la señal de electroencelafograma (EEG), y que sirvan para estudiar posibles trastornos en función de su número, densidad y características concretas. Además, estas nuevas técnicas permiten disminuir la carga de trabajo y la variabilidad inter-observador en la labor de detección de dichas oscilaciones en las señales EEG.

El objetivo de este Trabajo Fin de Máster ha sido evaluar la utilidad de algoritmos basados en técnicas de *deep learning* para detectar *spindles* del sueño en señales EEG de niños de entre 6 y 9 años con sospecha de AOS. La mayoría de los estudios científicos publicados hasta la fecha actual sobre la detección de *spindles* se ha centrado principalmente en pacientes adultos, lo que, junto con las diferencias de la AOS en sujetos adultos provoca que los modelos de detección de *spindles* no sean fácilmente generalizables a la población infantil.

Para lograr dicho objetivo, se ha utilizado una base de datos formada por 9 niños y que ha sido proporcionada por el Hospital Universitario Rio Hortega de Valladolid. Esta base de datos se separó en distintos grupos: uno de entrenamiento (4 sujetos), otro de validación (1 sujeto) y otro de test (4 sujetos). Después de preprocesar la señal EEG, se han aplicado distintos modelos de *deep learning* basados en redes neuronales convolucionales (CNN) y U-Net para la detección automática de *spindles*. Con el fin de reducir el desequilibrio entre las clases “spindle” y “no spindle”, se han aplicado varias técnicas de balanceo de datos durante el entrenamiento de los distintos modelos de *deep learning*. Por último, se ha incluido una etapa de post-procesado con la finalidad de mejorar el rendimiento del clasificador eliminando falsos positivos de la salida de los modelos de *deep learning*.

Los resultados obtenidos difieren de unos modelos a otros. Por un lado, aplicando una estructura CNN se han conseguido sensibilidades en torno a 61.60%-84.43% con valores predictivos positivos de 36.43%-60.64% y F1-score de 53.22%-61.12%. Características como la implementación de una CNN profunda o la modificación del parámetro *dropout*, así como técnicas de balanceo como N2-N3, ventanas deslizantes o aplicación de ruido gaussiano para generar muestras de *spindles* sintéticos, han ayudado a mejorar considerablemente los resultados adquiridos con este tipo de redes. Por otro lado, con la arquitectura U-Net se refleja una alta sensibilidad del 85.42%, un valor predictivo positivo (PPV) del 57.13% y un F1-score del 68.48%. Todos los resultados obtenidos mejoran considerablemente a los adquiridos por el previo Trabajo Fin de Grado (TFG), donde se utilizó una metodología basada en la extracción de características y un algoritmo de clasificación *Random Forest*, ya que las PPV obtenidas son superiores en todos los casos al 12.56%. Sin embargo, en la mayoría de los modelos, el aumento de PPV y F1-score,

infiere directamente en una disminución de la sensibilidad del clasificador. Por último, añadir que los resultados de este TFM son comparables a estudios previos realizados en adultos encontrados en la literatura científica.

Estos resultados sugieren que el análisis de la señal EEG mediante algoritmos de *deep learning* es de gran utilidad para la detección automática de *spindles* en niños con AOS. Sería interesante que trabajos futuros evaluaran la asociación entre el número de *spindles* y su duración con parámetros neurocognitivos en niños con AOS.

PALABRAS CLAVE: apnea obstructiva del sueño (AOS) infantil, *deep learning,* electroencefalograma (EEG), Red Neuronal Convolucional (CNN), *spindles* del sueño, U-Net.

Abstract

Pediatric Obstructive Sleep Apnea Syndrome (OSA) in children is a high prevalent global disorder (affecting 1-5% of the population), adversely impacting the health and development of affected children. Recent technological advances have revealed that small oscillations, primarily occurring during the N2 and N3 sleep stages within the 11-16 Hz frequency range, known as sleep spindles. Sleep spindles are closely related to the cognitive process of both the general population and, particularly, children. This discovery has prompted research focused on developing automated algorithms for sleep spindle detection using electroencephalogram (EEG) signals, enabling the examination of potential disorders based on spindle number, density, and specific characteristics. Furthermore, these new techniques hold the potential to alleviate the workload and inter-observer variability associated within the task of detecting these oscillations with EEG recordings.

The aim of this Master's Thesis was to evaluate the usefulness of deep-learning algorithms for detecting sleep spindles in EEG signals of children between 6 and 9 years old with suspected OSA as a continuation of the previous Bachelor's degree thesis. Prior research spindle detection has mainly focused on adult patients, which, together with the differences in OSA characteristics between adults and children, leads to spindle detection models that are not easily generalizable to the pediatric population.

To achieve this objective, we used a database of 9 children provided by the Hospital Universitario Rio Hortega in Valladolid, Spain. This database was divided into different groups: a training group (4 subjects), a validation group (1 subject) and a test group (4 subjects). After EEG signal preprocessing, different deep learning models based on convolutional neural networks (CNN) and U-Net have been applied to automatically detect sleep spindles. To mitigate class between the "spindle" and "non-spindle" instances, several data balancing techniques were applied during the training of the diverse deep-learning models. Finally, post-processing procedures were conducted to improve the performance of the classifier by removing false positives from the output of the deep-learning algorithms.

The results obtained differed from one model to another. On the one hand, the application of a CNN structure yielded sensitivities ranging from 31.60% to 84.43%, positive predictive values (PPV) spanning from 36.43% to 60.64% and F1-scores ranging from 53.22% and 61.12%. Features such as the implementation of a deep CNN modification of the dropout parameter, as well as balancing techniques such as N2-N3, sliding windows or the application of Gaussian noise to generate synthetic spindle samples significantly contributed to the substantial improvement in the performance of these networks. On the other hand, the U-Net architecture demonstrated a high sensitivity (Se) of 85.42%, a positive predictive value (PPV) of 57.13% and a F1-score of 68.48%. All achieved results markedly outperformed the outcomes of the preceding TFG study, where a methodology based on feature extraction and a random forest classifier was used. The PPV obtained in all cases exceed 12.56%. However, in most of the models, the increase in precision and F1-score directly results in a reduction in classifier sensitivity. Finally, it is noteworthy that the results of this TFM are comparable to previous studies conducted on adult subjects.

These findings underscore the utility of deep learning algorithms in the analysis of EEG signals for the automatic detection of spindles in children with OSA. Subsequent research endeavors should consider assessing the association between the spindle quantity and spindle duration with neurocognitive parameters in children with OSA.

KEYWORDS: pediatric obstructive sleep apnea (OSA), deep learning, electroencephalogram (EEG), Convolutional Neural Network (CNN), sleep spindles, U-Net.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1: INTRODUCCIÓN

1.1 SEÑALES BIOMÉDICAS

El sistema nervioso central establece comunicación con los demás sistemas del cuerpo humano a través de señales de naturaleza eléctrica o química. Analizar estas señales, portadoras de información fundamental, es esencial para que los especialistas puedan diagnosticar diversas patologías relacionadas con ellas.

En el pasado, las señales biomédicas se evaluaban visualmente mediante métodos manuales, lo que propiciaba la aparición de errores en la identificación de patrones. Por ende, uno de los objetivos clave en el desarrollo del procesamiento de señales biomédicas consiste en reducir significativamente la subjetividad de estas mediciones mediante la implementación de métodos informáticos, lo que no solo mejora su precisión sino también su exactitud (Sörnmo & Laguna, 2005).

Otra ventaja importante que ha brindado el procesamiento de señales biomédicas es la capacidad de extraer características, lo cual permite caracterizar de forma más completa las señales bajo estudio y comprender la información que contienen. La obtención de detalles de las señales no perceptibles mediante una simple inspección visual es uno de los propósitos principales del diseño de los métodos de extracción (Sörnmo & Laguna, 2005). Un ejemplo para ello sería el estudio de oscilaciones conocidas como *spindles* del sueño, tema que se desarrollará a lo largo de este Trabajo Fin de Máster (TFM) como continuación al previo Trabajo Fin de Grado (TFG). A pesar de ser deseable que las características obtenidas tengan un significado intuitivo para los especialistas, no siempre es así y se necesita por ello cierto conocimiento de la técnica para poder interpretar los resultados.

Existen diversos tipos de señales biomédicas, todas relacionadas con procesos de homeostasis en órganos internos o desencadenadas por estímulos externos. Estas señales se pueden clasificar según su origen, y entre ellas destacan (Martinek *et al.*, 2021):

i. **Señales bioacústicas**: corresponden a la detección de sonidos producidos por órganos internos debido a movimientos mecánicos o flujos de fluidos en el cuerpo. Los sonidos generados por del pulmón (*lung sounds*, LS) y los del corazón (*heart sounds*, HS) son especialmente valiosos para propósitos de diagnóstico médico. Para capturar estas señales, es necesario contar con herramientas como el estetoscopio (Druzgalski et al., 2016).

ii. **Señales biomecánicas**: son el resultado de procesos mecánicos que ocurren en el cuerpo humano y proporcionan mediciones sobre cambios en la posición, ubicación, aceleración, precisión y velocidad (Martinek et al., 2021).

iii. **Señales bioquímicas**: ofrecen información sobre la concentración de sustancias y el pH del cuerpo humano (Martinek *et al*., 2021).

iv. **Señales bioeléctricas**: este estudio se centra en especialmente en este tipo de señales, las cuales se originan en eventos eléctricos que tienen lugar en las membranas celulares y provienen principalmente de órganos como el cerero, el corazón y otros músculos. Dentro de esta categoría se pueden identificar varios tipos (Martinek *et al.*, 2021) (Sörnmo & Laguna, 2005):

   a. **Electroencefalograma (EEG)**: refleja la actividad eléctrica de cerebro y se utiliza para diagnosticar enfermedades tales como la epilepsia. Se explorará más en detalle esta técnica, ya que las señales empleadas para este TFM provienen del EEG de varios pacientes (Sörnmo & Laguna, 2005).

   b. **Electrocardiograma (ECG)**: refleja la actividad eléctrica del corazón mediante la colocación de electrodos sobre el pecho, las manos y los pies. Los impulsos eléctricos generados por el músculo cardíaco regulan los latidos del corazón y permiten investigar enfermedades tan comunes como el infarto de miocardio (Sörnmo & Laguna, 2005).

   c. **Potenciales evocados (*evoked potentials,* EP)**: representan manifestaciones eléctricas del cerebro en respuesta a diversos estímulos externos, típicamente visuales o auditivos. Esta señal se obtiene mediante electrodos parecidos a los empleados en el EEG y son valiosos para diagnosticar trastornos relacionados con la vía visual y el tronco encefálico (Chiappa, 1997).

   d. **Electromiograma (EMG)**: esta señal registra la actividad neuromuscular al medir las corrientes eléctricas que aparecen durante las contracciones y relajaciones musculares. La complejidad del EMG radica en su control por el sistema nervioso y su dependencia de las características anatómicas y fisiológicas de los músculos del cuerpo (Sadikoglu *et al*., 2017). Enfermedades como la distrofia muscular o la información de los músculos se pueden identificar gracias al electromiograma.

   e. **Electroneurograma (ENG)**: este tipo de señal se obtiene al estimular un nervio periférico con una descarga eléctrica, permitiendo así medir la respuesta a lo largo de dicho nervio. La velocidad de conducción de un nervio estimulado se obtiene gracias a unos electrodos de aguja y con ello, se puede identificar si dicho nervio se encuentra dañado o no (Sörnmo & Laguna, 2005).

   f. **Electroretinograma (ERG)**: se emplea en el estudio de los potenciales eléctricos generados por la retina en respuesta al estímulo de la luz. Para obtener esta señal se coloca un electrodo encapsulado en una lente sobre la córnea del ojo, lo que permite evaluar la respuesta eléctricas de los conos y los bastones que conforman la parte posterior de la retina ocular (Sörnmo & Laguna, 2005).

Existen muchos más tipos de señales bioeléctricas de las cuales no se va a profundizar en este estudio. Se pueden identificar algunas de ellas en función de su localización alrededor del cuerpo humano en la Figura 1.1.

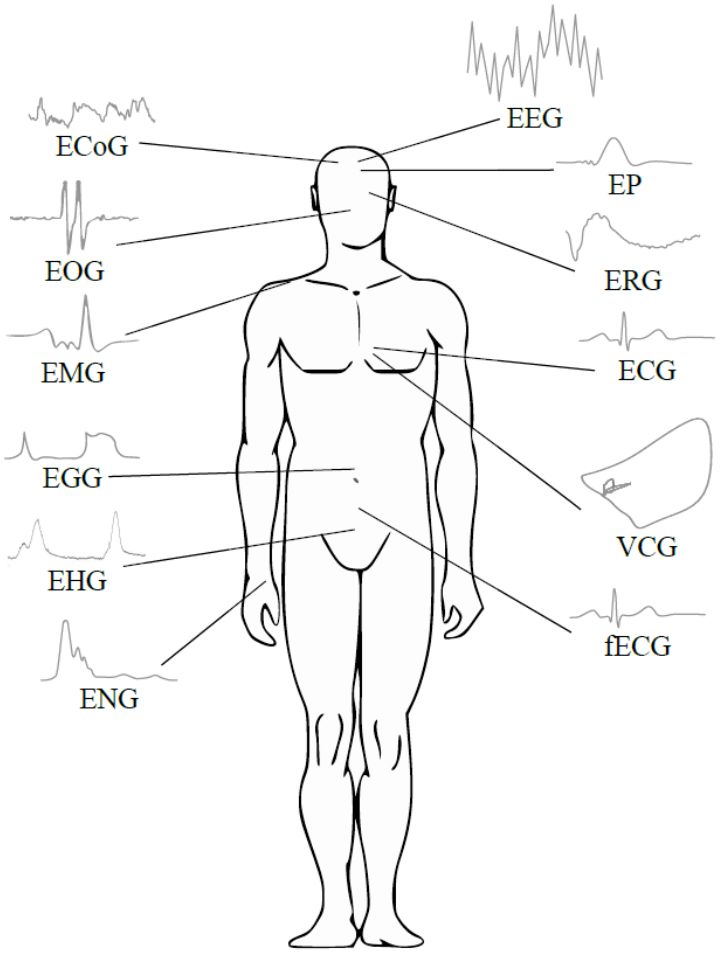

Figura 1.1: Esquema general de distintas señales biomédicas. En él se muestran: electroencefalograma (EEG), potenciales evocados (EP), electroretinograma (ERG), electrocardiograma (ECG), vectorcardiograma (VCG), electrocardiograma fetal (fECG), electroneurograma (ENG), electrohisterograma (EHG), electrogastrograma (EGG), electromiograma (EMG), electrooculograma (EOG), y electrocorticograma (ECoG) (Martinek *et al*., 2021)

1.2 ELECTROENCEFALOGRAMA (EEG)

El cerebro humano se destaca como uno de los órganos más intricados conocidos. Comprender su comportamiento y estudiar su función a lo largo del tiempo es esencial y, para lograrlo, se utilizan técnicas electrofisiológicas que permiten registrar los procesos eléctricos que tienen lugar e las membranas neuronales que conforman el cerebro (Sörnmo & Laguna, 2005).

El descubrimiento de fenómenos eléctricos que tenían lugar en los nervios y músculos suscitó un gran interés entre varios investigadores, quienes llevaron a cabo diversos

estudios al respecto. Fue el electrofisiólogo inglés Richard Caton en 1874 el pionero de la existencia de la actividad eléctrica del cerebro quien, junto con la influencia de Eduard Hitzig y Gustav Theodor Fritsch, demostraron la evidencia de respuestas motoras eléctricas en el cerebro de otros animales y consiguieron producir convulsiones en dicho órgano después de estimularlo con intensos impulsos eléctricos. No obstante, el EEG fue descubierto por Hans Berger en 1924 cuando registró por primera vez las oscilaciones rítmicas en una persona joven utilizando un galvanómetro de cuerda. Este avance condujo al desarrollo del electroencefalograma (Palacios, 2002; Sörnmo & Laguna, 2005).

En la actualidad, el EEG se emplea para registrar la actividad eléctrica de los tejidos cerebrales mediante la medición diferencial de potencial entre un electrodo de exploración y una referencia (Peralta *et al.*, 2004). La información contenida en el EEG es sumamente variada y depende en gran medida del estado del sujeto en cuestión, ya sea que esté despierto, dormido, realizando actividad física o relajado (ver Figura 1.2). Estos ritmos se caracterizan en función del rango de frecuencias en el que se encuentra y en base a su amplitud (Sörnmo & Laguna, 2005).

La amplitud de la señal EEG guarda una estrecha relación con el grado de sincronización entre las interacciones (Sörnmo & Laguna, 2005). Cuando un grupo de neuronas se excita de manera síncrona, se observan amplitudes elevadas en la señal debido a la superposición de las señales individuales generadas por las neuronas. La repetición de varias excitaciones síncronas neuronales es lo que provoca una señal rítmica en el EEG. Por el contrario, la excitación asíncrona genera amplitudes generalmente bajas, aunque dependiendo de la dispersión temporal, se puede obtener como resultado variaciones en las amplitudes del EEG. (Sörnmo & Laguna, 2005).

En lo que respecta a la frecuencia o velocidad de oscilación de las señales EEG, esta se encuentra vinculada con la actividad del tálamo. La información rítmica en forma de patrón se genera gracias a las capacidades neuronales localizadas en esta parte junto con las interacciones que presentan las neuronas de la zona del cortex entre sí (Sörnmo & Laguna, 2005).

Señales con alta frecuencia y pequeñas amplitudes son propias de situaciones de alerta o de sueño no profundo, mientras que señales con baja frecuencia y grandes amplitudes están asociadas a estados de somnolencia o sueño profundo. Generalmente, el rango de amplitudes en el que se suele encontrar este tipo de señales es de unos pocos microvoltios, aproximadamente 100 µV. Para el caso de la frecuencia, esta se suele mover entre los rangos de 0.5 a 40 Hz(Sörnmo & Laguna, 2005).

En función de la frecuencia registrada del EEG, se pueden distinguir cinco formas de onda categorizadas en sub-bandas:

1. **Delta (< 4 Hz)**: caracterizadas por darse en una situación de sueño profundo y presentan amplitudes altas. Estas oscilaciones pueden informar de daños cerebrales o discapacidades en caso de detectarse cuando el sujeto está despierto.
2. **Theta (4-8 Hz)**: se caracterizan por darse durante sueño liviano. En los niños, es la principal banda de frecuencia.
3. **Alpha (8-13 Hz)**: las oscilaciones detectadas en esta banda se suelen dar en situaciones de un sujeto despierto, pero relajado y con ojos cerrados. Normalmente

las amplitudes son mayores en las regiones occipitales. Es la principal banda de frecuencia en los adultos.

4. **Beta (13-30 Hz)**: esta forma de onda presenta frecuencias altas y amplitudes bajas, que son asociadas a la actividad del córtex. Se suelen dar durante la fase REM del sueño y en situaciones de alerta.
5. **Gamma (> 30 Hz)**: esta banda se puede observar cuando se realizan se realizan movimientos conscientemente. Está relacionado con el estado activo de procesamiento de información del córtex.

Cabe destacar que, dependiendo de la fase del sueño, el EEG puede presentar distintas formas de onda en forma de picos o pequeñas oscilaciones impredecibles con un patrón irregular cuya presencia puede informar a especialistas de la existencia o no de ciertos trastornos o enfermedades.

En la Tabla 1.1 se recogen los tipos de oscilaciones más comunes que aparecen en función de las distintas fases del sueño: la etapa REM (*Rapid Eye Movement*) y la etapa no-REM (*non-rapid Eye Movement*, NREM), la cual está separada en tres fases distintas (N1, N2 y N3). Ambas explicadas con mayor detenimiento en apartados posteriores. En este TFM se va a continuar con el estudio de *spindles* que, como se puede apreciar, aparecen durante las fases N2 y N3 del sueño.

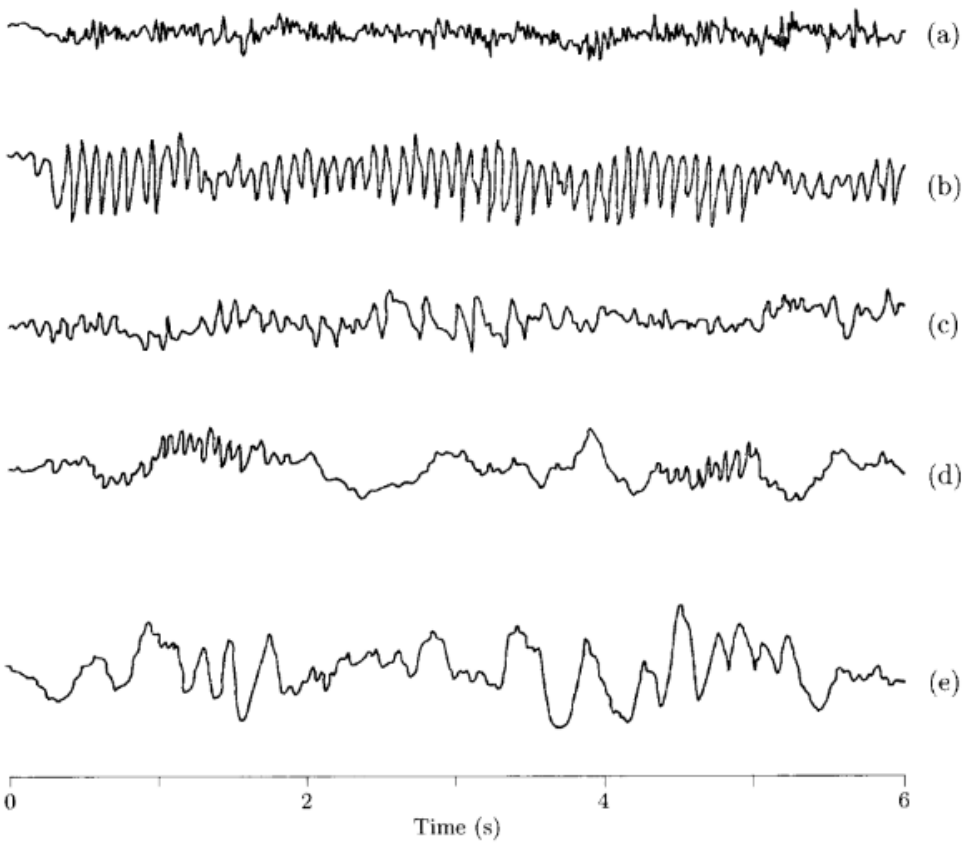

Figura 1.2: Ritmos del electroencefalograma dependiendo del estado del sujeto: (a) excitado, (b) relajado, (c) adormilado, (d) dormido y (e) profundamente dormido (Sörnmo & Laguna, 2005).

Fase del sueño	Profundidad del sueño	Formas de onda
N1	Somnolencia	Ondas de vértice, ondas alpha
N2	Sueño ligero	Ondas de vértice, *spindles*, *k-complex*
N3	Sueño profundo	Ondas delta, *spindles*
REM	Fase REM	Desincronización con frecuencias altas

Tabla 1.1: Características esenciales de las fases del sueño

1.2.1 ADQUISICIÓN DEL EEG

Clínicamente, se utiliza el sistema internacional 10/20 para la obtención del EEG, que implica la colocación de electrodos según una disposición específica en el cuero cabelludo (ver Figura 3). Este sistema se denomina así debido a que la distancia relativa entre las localizaciones de los electrodos dentro del perímetro craneal está equiespaciada entre el 10% y el 20%. En los últimos años, el uso de este sistema ha experimentado un notable aumento, especialmente en aplicaciones relacionadas con la cognición, así como en investigaciones de neurociencia y terapias psiquiátricas (Herwig et al., 2003; Sörnmo & Laguna, 2005).

En particular, el sistema 10/20 emplea un conjunto de hasta 21 electrodos que se colocan en diversas ubicaciones en el cuero cabelludo. Para medir la señal de manera precisa, se emplean referencias, las cuales pueden consistir en electrodos posicionados a cierta distancia de la zona que se pretende medir o en el promedio de todas las señales captadas por los electrodos del sistema (Herwig et al., 2003; Sörnmo & Laguna, 2005).

En cuanto al espaciado de los electrodos, éste suele ser bastante disperso. La distancia entre electrodos aproximada para un adulto suele ser de unos 4.5 cm. En caso de querer aumentar la precisión y así conseguir un mayor detalle de las señales, hay estudios que indican que se debe emplear 64 electrodos o más. Esto interesa especialmente cuando el mapeo del cerebro es de gran interés (Sörnmo & Laguna, 2005).

En este sistema, los electrodos se identifican con una letra dependiendo de la zona craneal a la que estén conectados. En concreto se emplean las letras F, P, C, T, O y A para identificar las zonas frontal, parietal, central, temporal, occipital y aurícula, respectivamente. Este trabajo se va a realizar a partir de la señal obtenida por el canal C3, ya que es el más común utilizado en estudios del sueño, localizado en la zona central izquierda como se puede ver en la Figura 1.3.

Las señales utilizadas en este TFM han sido registradas a una frecuencia de muestreo de 500 Hz, siendo la frecuencia recomendada para el EEG por la AASM (Richard B. Berry, MD; Rita Brooks, MEd, RST, RPSGT; Charlene E. Gamaldo & Susan M. Harding, MD; Robin M. Lloyd, 2016) y la más utilizada en estudios previos (Wei, Ventura, Ryan, et al., 2022).

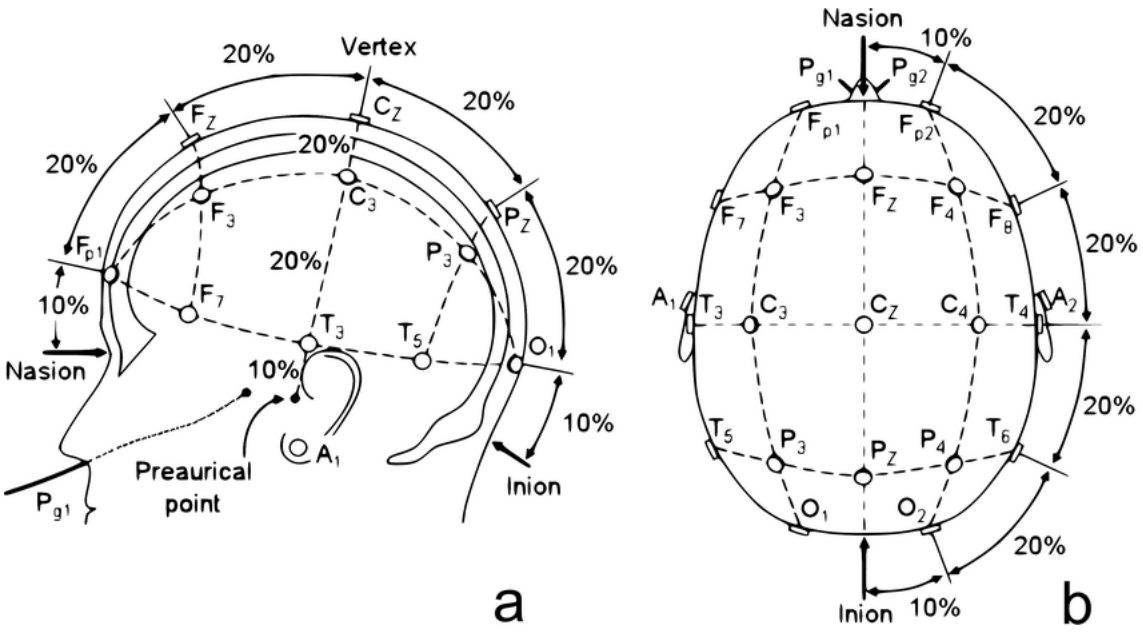

Figura 1.3: Sistema internacional 10/20 para la adquisición del EEG(Parra Sepúlveda, Roberto Humberto Montiel Ross et al., 2015).

1.3 APNEA DEL SUEÑO EN NIÑOS

La apnea del sueño pediátrica (AOS) es un trastorno crónico, común y de desarrollo gradual que afecta al 5% de la población adulta (Henry Olivi, 2013) y entre el 1-3% de la población infantil (Uong *et al*., 2005). Este trastorno se caracteriza por episodios repetitivos durante el sueño provocados por la obstrucción completa o parcial de la faringe, que ocasionan hipoxia seguida de re-oxigenación y despertares transitorios, responsable de alteraciones fisiopatológicas (Henry Olivi, 2013). Además de afectar a la calidad del sueño, la AOS está relacionada con un deterioro de la calidad de vida, un mayor riesgo de problemas cardiovasculares, hipertensión y trastornos metabólicos. También se ha observado que la AOS puede tener un impacto negativo en el desarrollo y el rendimiento escolar de los niños (Mohammadi et al., 2021; Uong et al., 2005).

Este trastorno ha sido reconocido científicamente en las últimas décadas debido a su alta prevalencia en la población y su impacto significativo en la salud (Henry Olivi, 2013).

En comparación con los adultos, las causas y características de la apnea del sueño en niños son distintas, pero sus consecuencias para la salud son mayores debido a que en esta etapa tiene una gran importancia el desarrollo neurocognitivo. La AOS en niños puede conllevar a serias complicaciones pulmonares, neurocognitivas y conductuales ya que procesos como el crecimiento, la memoria, el desarrollo físico y mental o el desarrollo del cuerpo se realizan durante el sueño principalmente en la etapa de niñez y de adolescencia. Por ello, en los últimos años, se está tratando este tipo de problemas para evitar un efecto negativo que infiera de manera directa en la salud de los niños (Uong *et al*., 2005). En concreto, para el desarrollo de esta investigación se ha trabajado con varios EEG de niños que presentan este trastorno de edades entre 6 y 9 años.

La obesidad es un factor de riesgo significativo en niños que puede aumentar las posibilidades de padecer apnea en un 4.5%. Su investigación ha confirmado la relación bidireccional entre la obesidad y la apnea del sueño, ya que los problemas de sueño pueden contribuir al aumento de peso y, a su vez, la obesidad infiere en el aumento de somnolencia (Hermida L, 2016).

Por otro lado, el ronquido es otro indicador significativo a la hora de sospechar sobre la existencia de este trastorno en tempranas edades. Presentar este síntoma durante la noche parece un signo importante y por ello se recomienda realizar exámenes clínicos detallados para comprobar la existencia o no de AOS (Hermida L, 2016).

Estudios recientes han demostrado que las oscilaciones conocidas como *spindles* del sueño están relacionadas con procesos cognitivos, emocionales y sociales. Una alta densidad de *spindles* infiere en un aumento en la consolidación de la memoria, habilidades lingüísticas y aprendizaje, mientras que el caso contrario suele estar asociado a pacientes con demencia (Mohammadi *et al*., 2021). Existen diferencias en las características de *spindles* entre la población que sufre AOS y la que no. Además, se ha demostrado que los pacientes con AOS que están siendo tratados presentan un aumento en la actividad *spindle*. Por este motivo, resulta de gran interés desarrollar algoritmos que detecten estas oscilaciones de forma automática y así poder diagnosticar enfermedades asociadas (Mohammadi *et al*., 2021).

1.4 SPINDLES DEL SUEÑO

A lo largo del proceso del sueño, la actividad cerebral humana experimenta diferentes etapas, cada una caracterizada por patrones u oscilaciones que reflejan los estados cerebrales en ese momento. Dentro de cada fase del sueño coexisten micro eventos que ejercen un fuerte impacto en la actividad neuronal y que pueden ser identificados mediante herramientas como el EEG. Existen muchos tipos de oscilaciones como *ripples* o *K-complex*, pero este proyecto se centrará únicamente en *spindles* (Figura 1.4) (Gomez-Pilar et al., 2021).

El análisis de estas complejas interacciones es de gran importancia, ya que proporciona información valiosa para la detección y prevención de ciertas enfermedades. Debido a ello, en los últimos años, se han realizado diversas investigaciones que han demostrado que los *spindles* del dueño desempeñan un papel crucial en los procesos cognitivos, especialmente en la memoria y el aprendizaje. (Gomez-Pilar *et al*., 2021). Un claro ejemplo de trastorno relacionado con la falta de *spindles* durante la noche es la esquizofrenia *(Manoach et al.*, 2016).

La mayoría de los expertos caracterizan los *spindles* en función de su duración, número de apariciones por minuto, por su apariencia e incluso en base a su frecuencia de oscilación. Los *spindles* del sueño, generados por la interacción del núcleo reticular talámico y otros núcleos talámicos (Kulkarni *et al*., 2019), se producen mayoritariamente durante la fase N2 del sueño, aunque también durante N3, entre 11-16 Hz (banda frecuencial sigma) y presentan una duración entre 0.5 y 3 segundos (Gomez-Pilar *et al.*, 2021).

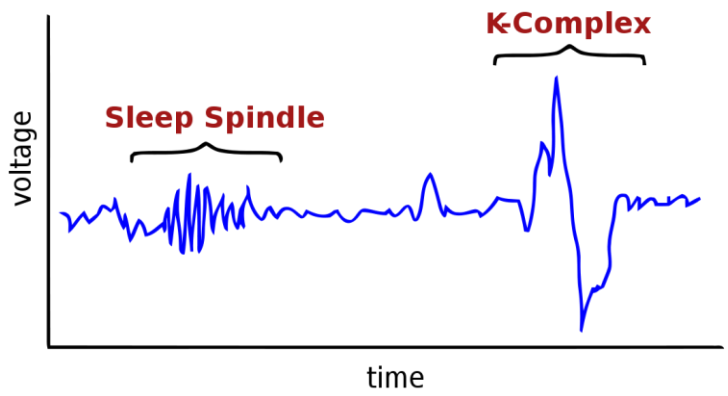

Figura 1.4: Representación de un *spindle* del sueño frente a un *k-complex.*

Varios estudios proponen distinguir entre dos tipos de *spindles* según su frecuencia, ya que presentan diferencias en términos de topología y función. Los denominados *fast spindles*, por encima de los 13 Hz originados principalmente sobre la zona centroparietal del cerebro, y los *slow spindles* por debajo de los 13 Hz producidos con mayor frecuencia en la zona frontal (Purcell *et al*., 2017).

Es importante tener en cuenta que la actividad *spindle* puede varias según el sexo y la edad, lo que significa que se requieren algoritmos específicos para detectarlos en adultos o en niños. En adultos, tanto la densidad de *spindle* como la potencia en la banda sigma disminuyen con la edad. Por otro lado, en niños la actividad *spindle* y la potencia en su banda aumentan con la edad alcanzando su pico más alto en la adolescencia. En cuanto a la duración de los *spindles*, tiende a disminuir gradualmente desde la infancia hasta la edad adulta. Los mismo ocurre con la amplitud, aunque se caracteriza por mantenerse estable durante las etapas de niñez y adolescencia. Por último, la frecuencia de oscilación presenta un comportamiento contrario, ya que va a aumentando conforme lo hace la edad. Típicamente en niños suele ir de 8 a 15 Hz. Todas estas características se pueden apreciar en la Figura 1.5 (Purcell *et al*., 2017).

Además, se ha demostrado que hay diferencias en la densidad de *spindles* en función del sexo. El sexo femenino presenta 0.16 *spindles*/segundo a mayores que el sexo masculino. Estas diferencias se han observado principalmente en personas con edades en torno al rango de 15-45 años (Purcell *et al*., 2017).

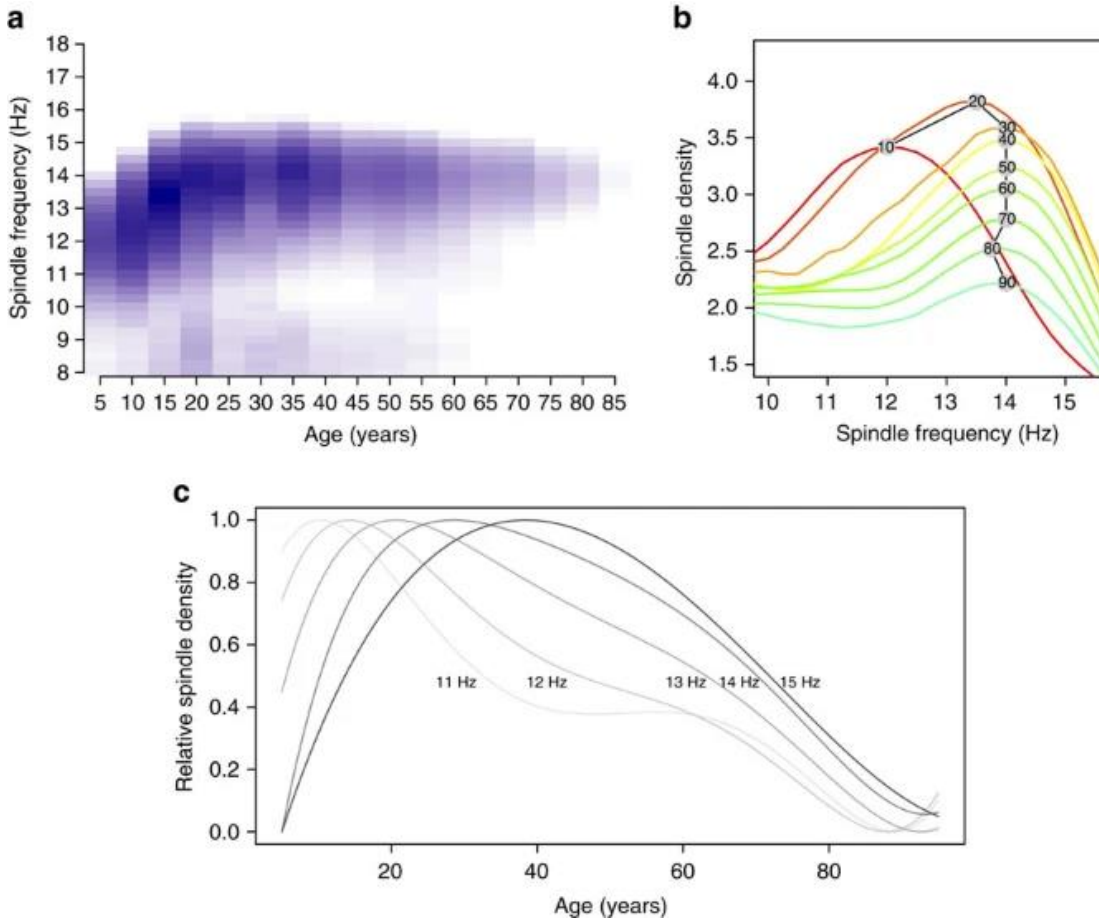

Figura 1.5: Estas imágenes representa la estimación media de la densidad de *spindles* en función de la frecuencia: (a) banda de frecuencias *spindle* en función de la edad. Las sombras más oscuras indican una mayor densidad *spindle*. Se puede ver cómo la densidad va disminuyendo con la edad. (b) En las ordenadas se representa la densidad media de *spindle* de grupos de personas con la misma edad y en las abscisas se representa la frecuencia. (c) Se representan curvas de densidad relativa agrupadas por frecuencias entre 11-15 Hz en función de la edad (Purcell *et al.*, 2017).

1.5 DEEP LEARNING

La inteligencia artificial (IA) hace posible que las máquinas u ordenadores sean capaces de aprender y actuar ante distintas situaciones de manera semejante a la inteligencia humana (Hulsen, 2022).

Dentro de la IA, se encuentra el *deep learning* (DL), que está teniendo un gran impacto en la investigación y en aplicaciones biomédicas gracias a su gran flexibilidad a la hora de relacionar las entradas con las etiquetas de salida y a su capacidad para integrar grandes volúmenes de datos y aprender relaciones complejas (Wainberg et al., 2018). El *deep learning* forma parte del *machine learning* (ML), sin embargo, este se caracteriza en que se permite el procesamiento de datos en crudo (*raw data*) sin necesidad de una previa etapa de extracción de características, tal y como era necesario aplicar en el previo TFG, donde

se aplicaron técnicas de extracción de características temporales y espectrales del EEG para posteriormente aplicar un clasificador *Random Forest* para detectar los *spindles*.

Esta tecnología se basa en la idea de las redes neuronales artificiales, las cuales se inspiraron en la estructura de la red neuronal humana. Consiste en una serie de capas, en las cuales, el término *deep* hace referencia a que el tamaño y número de capas es muy elevado. De esta forma se consigue extraer de manera mucho más eficiente y con una mayor precisión la información presente en los datos a estudiar (Ghaderi & Khotanlou, 2019).

Actualmente estamos expuestos a una cantidad masiva de datos que no se podrían haber aprovechado con los algoritmos tradicionales. Una de las ventajas principales de utilizar *deep learning* es que permite trabajar con conjuntos de datos muy grandes y es gracias a las redes neuronales las que consiguen procesar toda esa información. Por contrapartida, el tiempo de entrenamiento de estos algoritmos será mayor con el aumento del tamaño de los datos (Lecun et al., 2015).

Dentro del ámbito de *deep learning*, destacan dos tipos de redes: redes neuronales convolucionales (*convolutional neual networks*, CNN) y redes neuronales recurrentes (*recurrent neural networks*, RNN). Las CNN es uno de los algoritmos más ampliamente utilizados. Estos modelos se emplean en aplicaciones de clasificación de imágenes, detección de objetos 3D así como en aplicaciones 1D y 2D en la que se incluyen señales EEG (Craik et al., 2019). Las redes RNN, por su parte, se suelen utilizar para procesar datos con secuencia de datos como reconocimiento de voz o traducción de idiomas. También es habitual combinar CNN con RNN para obtener mejores resultados en aplicaciones biomédicas (Yu et al., 2022). En este TFM se han utilizado distintos modelos de CNN para detectar los *spindles* del sueño.

1.6 HIPÓTESIS Y OBJETIVOS

El desarrollo de este TFM parte de la hipótesis planteada en el previo TFG de que es posible detectar de manera automática *spindles* del sueño por medio de la información extraída del procesado de la señal de EEG (Pacho Velasco, 2022). Teniendo en cuenta los avances en múltiples campos de investigación gracias a la aplicación de los métodos de *deep learning* en este TFM se trabaja bajo la hipótesis de que la aplicación de técnicas de *deep learning* permite detectar con gran precisión los *spindles* del sueño. Persiguiendo esta hipótesis, el objetivo general se centra en mejorar el algoritmo desarrollado en el TFG de esta misma autora empleando metodologías de *deep learning* en lugar de *machine learning* para ser capaz de detectar estos micro eventos en niños diagnosticados con apnea del sueño, tal y como se indicó en sus líneas futuras. Para poder comparar los resultados, se ha trabajado con los mismos registros EEG de niños proporcionados por el Hospital Universitario Río Hortega de Valladolid (9 niños con edades comprendidas entre 6 y 9 años).

Una vez detallado el objetivo general, se muestran los objetivos específicos para alcanzarlo:

1. Revisar el estado del arte sobre los algoritmos de *deep learning* que ya han sido publicados en revistas científicas relacionados con la detección de *spindles* del sueño.
2. Seleccionar la arquitectura más apropiada de *deep learning* teniendo en cuenta el tipo de registros *a priori* que se tienen para poder reproducirlo e implementarlo.
3. Estudiar e implementar la arquitectura seleccionada para poder llevar a cabo este trabajo y aportar ideas de mejora o cambios, en caso de ser necesarios.
4. Extraer conclusiones a cerca del estudio desarrollado.

Estos objetivos específicos se pueden dividir en 3 fases, las cuales se han ido siguiendo para completar el trabajo:

1.6.1 FASE 1: DOCUMENTACIÓN Y REVISIÓN BIBLIOGRÁFICA

- Desarrollo del curso introductorio a *deep learning*, en base al cual se ha podido realizar todo el estudio del TFM, utilizando bibliotecas de Python.
- Revisión bibliográfica de métodos ya implementados con el objetivo de conocer el estado del arte y poder seleccionar la arquitectura que mejor se adapte a las condiciones iniciales.

1.6.2 FASE 2: ESTUDIO DEL ALGORITMO E IMPLEMENTACIÓN

- Información sobre el algoritmo seleccionado para conocer en profundidad su funcionamiento interno y cómo trabajar con él.
- Implementación de este método en Python siguiendo aproximadamente los pasos indicados en la publicación.
- Modificación de ciertos aspectos para la implementación del método que permitan una mejor integración para las señales y población con las que se está trabajando.

1.6.3 FASE 3: EXTRACCIÓN DE RESULTADOS Y CONCLUSIONES

- Entrenamiento de las arquitecturas CNN y U-Net para obtener resultados a cerca del algoritmo implementado.
- Análisis de los resultados.
- Modificación de la estrategia para tratar de obtener mejores resultados y comparación con otros modelos.
- Extracción de conclusiones.
- Redacción del informe final.

1.7 ESTRUCTURA DEL TFM

El informe de este TFM consta de un total de 6 capítulos en los cuales se ha ido detallando el proceso a seguir para la realización del estudio. Estos capítulos se dividen en: “Introducción”, “Estado del Arte”, “Materiales y métodos”, “Resultados”, “Discusión”, “Conclusiones y líneas futuras”. A continuación, se va a hablar de la estructura que presenta el resto del documento.

En el capítulo 2 se lleva a cabo la revisión del estado del arte dentro del contexto de detección automática de *spindles* del sueño basado en algoritmos de *deep learning.* Este apartado comienza con una explicación en la que se habla de cómo se debe preparar la base de datos con la que se desea trabajar, su preprocesado y técnicas de balanceo de datos para evitar el desequilibrio entre clases. Por último, se detallan varios métodos utilizados en diversos artículos como detectores de *spindles* y algunas estrategias de post-procesado empleadas en ellos.

Los sujetos y señales con las que se ha trabajado se describen en el tercer capítulo. Asimismo, se documenta la metodología que se ha utilizado para el desarrollo del TFM comentando todos los pasos seguidos para conseguir el objetivo final de la implementación. Este capítulo finaliza con una serie de métricas de evaluación, las cuales permiten caracterizan los resultados predictivos obtenidos por los distintos modelos.

Es en el capítulo 4 donde se reflejan los resultados obtenidos para los distintos modelos desarrollados. En concreto, se han diseñado modelos basados en una arquitectura CNN convencional y otros basados en una U-Net. Por cada uno de ellos se muestra la matriz de confusión, así como métricas de rendimiento obtenidas ilustrados con algún ejemplo de predicción.

Posteriormente, en el quinto capítulo, se realiza una discusión sobre los valores obtenidos en el capítulo anterior para los distintos modelos desarrollados. Además, se exponen los resultados de artículos y TFG previos para poder compararlos con los modelos desarrollados. Finalmente, se concluye con una serie de limitaciones con las que se ha topado a lo largo del desarrollo del proyecto.

Finalmente, se expone en el último capítulo las conclusiones extraídas del proyecto junto con posibles líneas futuras y contribuciones al campo de investigación de la ingeniería biomédica.

CAPÍTULO 2: ESTADO DEL ARTE

A lo largo de esta investigación, ha sido necesaria una revisión del estado del arte con el fin de comprender los avances que se han realizado hasta la fecha. Durante este periodo de documentación, se han revisado artículos publicados en los últimos años y se ha podido analizar cuál de ellos era el más apropiado para replicar en este estudio. A lo largo de ese capítulo se recoge la evolución que ha sufrido la detección de *spindles* hasta la actualidad estableciendo una clara diferencia entre los artículos basados en *feature engineering* y *deep learning*. Además, se lleva a cabo una explicación sobre metodologías y bases de datos utilizadas dentro del ámbito de *deep learning.*

2.1 INTRODUCCIÓN

Los estudios relacionados con el sueño constituyen una línea de investigación que ha tenido un gran auge a en las últimas décadas. El interés que ha despertado el descubrimiento de *spindles* del sueño y su fuerte implicación en el proceso cognitivo de las personas ha hecho que investigadores busquen métodos para poder detectar estas oscilaciones y así ayudar a comprender posibles enfermedades (Parkinson, Alzheimer, etc.) (Manoach et al., 2016).

Tradicionalmente, la identificación de *spindles* se hacía de forma manual mediante la inspección del EEG por expertos en laboratorios clínicos, los cuales eran entrenados en la interpretación de señales polisomnográficas. La variabilidad inter-observador junto con lo tediosa que es esta labor, conlleva a dedicarle una excesiva cantidad de tiempo a la detección de *spindles*, lo cual resulta perjudicial para el interés clínico y biológico. Por ello, este motivo es una de las principales razones por las que se busca automatizar esta detección (Warby *et al*., 2014).

Hay varias estrategias para la detección automáticas de *spindles*, y cada una de ellas ha dado lugar a distintos detectores estrechamente relacionados. Uno de los primeros detectores de *spindles* se desarrolló en el año 1994 y estaba basado en tres etapas: i) el EEG se pasaba por un filtro paso banda entre las frecuencias 1.2-16 Hz; ii) se estableció un criterio de amplitud respecto a un umbral de 25μV; y iii) se calculó la duración del tiempo en épocas en las zonas donde la amplitud superaba dicho umbral (Schimicek *et al*., 1994).

Este método sirvió de base a muchos posteriores, los cuales introdujeron algunos cambios al respecto como son:

- Cambiar el filtro paso banda estandarizado por uno particular para cada sujeto.
- Reemplazar dicho filtro paso banda por transformaciones de la señal.

- Otras modificaciones empleando la forma del *spindle* para determinar su comienzo y su final en vez de establecer un criterio de umbral para la amplitud.

Gracias al gran avance tecnológico de los últimos años, estos algoritmos de detección automática han evolucionado a técnicas basadas en ML y posteriormente en DL. Acorde con el pionero de estas herramientas, Arthur Samuel, ML se define como el campo que estudia la habilidad de los ordenadores para aprender a partir de cierta información dada (Batta, 2020). De esta forma las máquinas son capaces de trabajar de manera mucho más eficiente y rápida. El propósito principal reside en aprender de los datos y en base a ellos obtener un resultado con sentido (Batta, 2020). Estas técnicas poco a poco han ido tomando importancia dentro del ámbito del sueño y actualmente es uno de los métodos más utilizados para la detección y clasificación de eventos.

Los primeros artículos que hablan de detección de *spindles* empleando estas técnicas de ML, se basan en redes neuronales artificiales (ANN, *Artificial Neural Network*) *(Ventouras et al.*, 2005). Además, hay publicaciones en las que primeramente emplean una pre-clasificación para eliminar de la señal elementos *no-spindle* y quedarse únicamente con los elementos *spindle* con los que se trabajará en una etapa posterior, denominada post-clasificación. Máquinas de vectores de soporte (SVM, *Support Vector Machine*) o perceptrones multicapa (MLP, *Multilayer Perceptron*) han sido necesarios para implementar esta técnica en el artículo (Acir & Güzeliş, 2004). Con el paso de los años se ha ido evolucionando hacia técnicas de mayor complejidad, que dejan atrás las redes neuronales convencionales. Duman et al. (2009) utiliza en su publicación un único árbol de decisión que realiza la clasificación en función de tres técnicas propuestas (Duman et al., 2009), mientras que Kinoshita et al. (2020) optó por una combinación de la transformada SST (*Synchrosqueezed Transform*) y un sobremuestreo aleatorio implementado con técnicas de boosting (Kinoshita *et al*., 2020). Además, Wei et al (2022) consiguió unos buenos resultados utilizando el algoritmo de *Random Forest*, metodología que se llevó a cabo en el previo TFG (Pacho Velasco, 2022).

Con el avance de esta tecnología y con el aumento masivo de datos, los métodos basados en ML se han quedado anticuados y se ha dado paso al DL. Con esta nueva técnica, deja de ser necesario un trabajo previo de extracción de características como entrada de los algoritmos. El aprendizaje profundo trata de encontrar y aprender por sí mismo las características de la señal reduciendo el trabajo a los expertos de esta tarea. Esto ha llevado a un aumento en el número de estudios que están utilizando métodos de DL para explorar nuevas formas de detectar automáticamente *spindles* del sueño (Wei, Ventura, Ryan, et al., 2022).

Dentro de esta rama, se han publicado algoritmos como “SpindleNet” formado por la combinación de una CNN y una RNN (Kulkarni *et al.*, 2019). Siguiendo el mismo ejemplo, Wei, Ventura, Ryan, et al. (2022) desarrollaron un algoritmo similar variando la estructura de las redes neuronales. Además existen otros algoritmos que solo utilizan CNN (Chen et al., 2021), así como otros más innovadores que emplean la estructura de una U-Net para detectar *spindles* (Kaulen et al., 2022; You et al., 2021).

En los siguientes apartados se trata de forma breve las etapas empleadas en los algoritmos basados en *machine learning* y, de forma más extensa, la metodología de los modelos *deep learning*, ya que son objeto de estudio en este TFM.

2.2 MACHINE LEARNING

Todos los algoritmos basados en ML convencional presentan una serie de etapas en común sobre las que destacan: adquisición de datos, preprocesado, extracción de características y, por último, clasificación. En este apartado, nos vamos a centrar principalmente en la extracción de características y clasificación, ya que la adquisición y preprocesado de datos son similares a las que se realiza con DL y se tratarán en esa sección.

La actividad *spindle* presenta características propias que la diferencian del resto de la señal de EEG. La extracción de estas características son las que permiten aplicar un algoritmo de clasificación que permita detectar para cada época la presencia o no de un *spindle* del sueño. Se pueden distinguir dos conjuntos de características en función de si se realiza un análisis temporal (media, varianza, desviación típica, *skewness*, valor cuadrático medio, etc) o frecuencial de la señal (aplicando la densidad espectral de potencia, transformada wavelet, etc) (Gomez-Pilar et al., 2021; Wei, Ventura, Mathieson, et al., 2022).

Antes de la etapa de clasificación, resulta importante conocer aquellas características que presentan una alta correlación con otras, ya que no suelen proporcionar información adicional y se puede prescindir de ellas. Se requiere de técnicas que permitan seleccionar de entre todas las características extraídas, aquellas que realmente aporten contenido relevante. Hay tres técnicas dentro de ML que se encargan de esta selección: métodos de envoltura (*wrapped*), métodos de filtrado y métodos integrados o híbridos (Khalid *et al.*, 2014).

Antes de comenzar con la implementación del algoritmo clasificador de ML, conviene realizar una normalización de las características seleccionadas para evitar que unas destaquen por encima de otras. Hay varios métodos que permiten realizar dicha normalización de los cuales destacan: escalador estándar (s*tandard scaler*), escalador de valor absoluto máximo (*maximum absolute value scaler*) y la transformación cuantil (*quantile transform*) (Ferreira *et al*., 2019).

Durante esta última etapa de calificación, se consigue predecir si el segmento de señal pertenece a un *spindle* o no. En los últimos años este proceso de clasificación se ha llevado a cabo gracias a algoritmos basados en ML supervisados. Dentro de este ámbito hay diversas técnicas disponibles y muchas de ellas se pueden aplicar a la detección automática de *spindles* del sueño. SVM (Acir & Güzeliş, 2004), ANN (Ventouras et al., 2005) (Güneŝ et al., 2011), árboles de decisión (Duman et al., 2009) o *random forest* (Wei, Ventura, Mathieson, et al., 2022) son los modelos más utilizados en este ámbito para la detección de *spindles* del sueño.

2.3 DEEP LEARNING

Este proyecto se centra en detectar *spindles* del sueño aplicando técnicas de DL. Por ello, la mayor parte de la investigación se ha realizado en este ámbito. En este apartado se pretenden explicar las bases de datos más populares y las metodologías y técnicas más comunes utilizadas en los últimos años para la realización de esta tarea.

2.3.1 ADQUISICIÓN DE DATOS

Trabajar dentro de la rama de investigación dedicada a la apnea del sueño, implica tratar con un gran número de señales biomédicas procedentes de distintos pacientes. Normalmente estas señales se adquieren de bases de datos ya existentes, aunque hay estudios en los que registran y analizan sus propias señales. En los últimos estudios, las bases de datos que más han sido utilizadas son:

- **MASS** (Montreal, Canadá): contiene 200 registros polisomnográficos (PSG) de 97 hombres y 103 mujeres de edad comprendidas entre 18 y 76 años. Estas señales se encuentran almacenadas en el formato europeo de datos (EDF, *European Data Format*) (Kulkarni *et al*., 2019) (Wei, Ventura, Ryan, et al., 2022). Las señales de 200 sujetos de todos ellos contienen marcas de *spindles* establecidas por expertos en el canal C3 del EEG.
- **DREAMS** (Universidad de Bruselas, Bélgica): contiene registros de 30 minutos de PSG provenientes de 8 sujetos formados por 4 hombres y 4 mujeres de edades comprendidas entre 35 y 50 años. Todas estas señales presentan marcas de *spindles* anotadas por expertos (Kulkarni *et al*., 2019) (Wei, Ventura, Ryan, et al., 2022).
- **MrOS** (*Osteoporotic Fractures in Men Study*): se encuentra en *the National Sleep Research Resource* (NSRR). Almacenado en una base de datos que se diseñó con el objetivo de facilitar la investigación de estudios relacionados con el sueño. En él se encuentran 26000 PSGs completos y sin ningún tipo de anotación de *spindles* junto con datos clínicos de varios sujetos durante toda la noche (Kulkarni et al., 2019).
- **CHAT** (*Childhood Adenotosillectomy Trial*): se encuentra almacenado también en NSRR. En particular, contiene un total de 1447 registros EEG de los canales C3 y C4 en niños. Es importante indicar que esta base de datos no contiene *spindles* anotados, sino que se utiliza para testear y visualizar el comportamiento de algoritmos entrenados previamente con otras bases de datos (Kulkarni *et al*., 2019).
- **MODA** (*Massive Online Data Annotation*): 180 registros EEG anotados de 115 segundos. Contiene un total de 5342 spindles anotados en los canales C3-LE y C3-M2 (Kaulen et al., 2022).
- **CLINICAL RESEARCH ETHICS COMITTEE** (*Cork Teaching Hospitals, Cork, Ireland*): base de datos privada que contiene 141 EEGs de niños de 4 meses. Cada registro dura 70 minutos y su frecuencia de muestreo es de 500 Hz. Los

spindles están anotados a mano en los canales F4-C4 y F3-C3 (Wei, Ventura, Ryan, et al., 2022).

2.3.2 PREPROCESADO DE DATOS

El pre-procesado de datos engloba aquellas técnicas de análisis de datos que permiten mejorar la calidad de las señales con la finalidad de proporcionar una mayor y mejor información. El ruido dentro de la señal, así como las posibles interacciones entre canales infieren en gran medida en los resultados obtenidos (Herrera, 2004). Por ello, uno de los aspectos más cruciales en el preprocesado de señales biomédicas consiste en adquirir conocimientos sobre el ruido o artefactos presentes en la señal con el objetivo de minimizar su influencia (Sörnmo & Laguna, 2005). Las técnicas más empleadas en este ámbito son:

- **Referenciado promedio de la señal**: antes de filtrar la señal en las bandas de interés, es habitual realizar un referenciado promedio de todos los canales del EEG y restarlo a cada elemento de la señal. Esta técnica se suele emplear para eliminar posibles interferencias que haya entre unos canales y otros.
- **Filtrado digital**: es conveniente filtrar la señal EEG para eliminar ruido y componentes de información innecesaria. Es una práctica habitual emplear un filtro *Nocht* lineal invariante en el tiempo (filtro elimina banda) para cancelar la interferencia de línea eléctrica que aparece al grabar los registros (50 Hz en Europa y 60 Hz en Estados Unidos) (Wei, Ventura, Ryan, et al., 2022). Además, el uso de filtros paso banda permiten trabajar únicamente con las bandas de interés. Esto mejora considerablemente la calidad y precisión de la señal. Aplicar filtrados paso bajo suele ser muy útil para reducir la interferencia de la actividad EMG cuando interesa especialmente frecuencias bajas en el EEG. Por último, es importante llevar a cabo un buen diseño de estos filtros ya que en el caso contrario podrían añadir espurios de ruido a la señal (Sörnmo & Laguna, 2005).
- **Eliminar *offset* de la señal**: otra técnica de preprocesado en señales biomédicas consiste en eliminar la componente de continua de la señal para cada canal de EEG extraído. Normalmente esta práctica consiste en hacer el promediado de toda la señal y restárselo a cada elemento (Wei, Ventura, Mathieson, et al., 2022).
- **Reducción de la amplitud de EEG**: en el estudio de You et al. (2021), la amplitud de las señales originales del EEG se limita al rango -150 µV a 150 µV para eliminar *outliers.* Posteriormente se realiza una transformación lineal al rango [-0.5 0.5] y se emplea esta señal como entrada al modelo. Aplicando esta técnica deja de ser necesario aplicar ningún tipo de filtro a la señal.
- **División del EEG en fragmentos**: una vez aplicados los preprocesados anteriores, se divide el EEG en segmentos de la misma duración para posteriormente poder extraer características de ellos. Concretamente, para el estudio de *spindles* del sueño, la mayoría de las publicaciones dividen el EEG en fragmentos entre 0.5 y 3 segundos con un *overlap* de entre 0.25 y 0.5 segundos. Esto es así debido a que la duración de un *spindle* se encuentra comprendida entre esos rangos (Wei, Ventura, Mathieson, et al., 2022; Wei, Ventura, Ryan, et al., 2022).

2.3.3 TÉCNICAS DE BALANCEAO DE DATOS

La aparición de *spindles* se distribuye en las fases N2 y N3 del sueño (NREM) con una densidad que varía de unos sujetos a otros. El número y duración de estos eventos es substancialmente menor que los eventos de no *spindles,* generando un problema de desequilibrio entre los datos de gran relevancia para la clasificación ya que hay muchas más muestras de la clase *no-spindle* que de la clase *spindle*. En diversos artículos se emplean varias técnicas con el objetivo de paliar esta cuestión:

i. **TÉCNICAS DE OVERSAMPLING**: se basan en crear *spindles* sintéticos con el fin de tratar de igualar la clase negativa con la positiva (You et al., 2021). Formas de generar nuevos *spindles* son:
   a. **Adición de ruido gaussiano**: Se generan muestras a partir de las originales añadiendo ruido gaussiano aleatorio de media 0 (You et al., 2021).
   b. **Sliding window**: la técnica de ventana deslizante se basa en generar varias muestras de *spindle* sobre un mismo *spindle*. Se considera que mínimo el 50% de esa muestra pertenece a uno de ellos y se genera una ventana deslizante de tal forma que cada cierto número de muestras se obtenga un nuevo *spindle*. Sería equivalente a decir que se generan muestras de *spindles* con distinto inicio y final pero de la misma duración (Kulkarni et al., 2019).
   c. ***Mirror***: se basa en generar muestras sintéticas realizando un espejo sobre las muestras de *spindles* originales y/o sobre muestras sintéticas de *spindles*.

ii. **TÉCNICAS DE UNDERSAMPLING**: todas estas técnicas se basan en la idea de disminuir el número de muestras de la clase mayoritaria para tratar de paliar el desequilibrio entre clases. Algunos ejemplos para aplicar *undersampling* son:
   a. **Trabajar con N2 y N3**: esta es una de las soluciones más eficaces para tratar de reducir el número de clases “no spindle” ya que estos eventos únicamente se dan en las etapas N2 y N3 del sueño. Además, como la mayor parte de *spindles* se dan en la tapa N2, hay artículos que restringen aún más la señal quedándose únicamente con esta etapa (Wei, Ventura, Ryan, et al., 2022).
   b. **Selección random**: se basa en eliminar de forma aleatoria (*random*) muestras de la clase dominante de toda la señal EEG. Un método para ello puede ser *RandomUnderSample* implementado en librerías de Python.

iii. **CUSTOM LOSS WEIGHTS FUNCTION**: son técnicas que se aplican en los algoritmos a la hora de generar los pesos de las redes neuronales, de tal forma que se le da más importancia a la clase minoritaria a la hora de realizar la clasificación. Un ejemplo de ello sería aplicar una ponderación de 0.7 para la clase minoritaria y 0.3 para la mayoritaria, asignando así más “importancia” a la clase con problemas. para asignar un peso distinto a cada elemento (Ebert-Uphoff et al., 2021).

iv. **LOSS FUNCTION**: son funciones de pérdidas que se aplican a los algoritmos para reducir la influencia demuestras desequilibradas. Entre ellas destacan:
    a. **Focal loss**: este método permite aprender los ejemplos más difíciles de clasificar haciendo que estos contribuyan más a las pérdidas. Se utiliza cuando se emplean arquitecturas de CNN básicas (Chen et al., 2021).
    b. **Dice loss**: este método actúa de forma similar al anterior pero es más utilizado cuando se emplea una arquitectura U-Net (You et al., 2021).

2.3.4 ALGORITMOS DE CLASIFICACIÓN *DEEP LEARNING*

Durante esta última etapa se consigue predecir si el segmento de señal pertenece a un *spindle* o no utilizando herramientas de DL. Dentro de este ámbito una gran variedad de redes y de arquitecturas han surgido y muchas de ellas se pueden aplicar a la detección automática de *spindles* del sueño. A continuación, se recoge una breve explicación de varios de los algoritmos utilizados en artículos con el objetivo de detectar estas oscilaciones en distintas bases de datos.

2.3.4.1 CONVOLUTIONAL NEURAL NETWORK (CNN)

Las CNN son unas de las arquitecturas más comunes y usadas en DL. Originalmente se utilizaban para aplicaciones como reconocimiento de imágenes y vídeo, aunque también utilizado en otras áreas como procesado de voz y visión computacional. Su estructura es similar a las redes neuronales, inspiradas en el comportamiento de las neuronas del cerebro humano (Pouyanfar et al., 2018).

Como se puede ver en la Figura 2.1, la arquitectura de una CNN de manera general está formada por varias capas convolucionales seguidas de capas de submuestreo (o *pooling*). Al final de todas estas capas normalmente se utilizan capas *fully-connected* y una capa de salida (típicamente una función *softmax* cuando se utilizan las redes para clasificación) (Pouyanfar et al., 2018).

Las capas de convolución son necesarias para extraer las características más importantes de los datos y suelen estar seguidas de filtros o *kernels* que se convolucionan con la señal de entrada. Normalmente, cuanto más se profundice en la red, se pueden extraer características más complejas (Pouyanfar et al., 2018).

Gracias a las capas de *pooling* se consigue reducir la dimensionalidad de las características, consiguiendo disminuir el tiempo de entrenamiento y se hace un control del sobreentrenamiento (*overfitting*) (Pouyanfar et al., 2018).

Después de pasar por varias capas de convolución y *pooling*, es habitual agregar una o más capas *fully connected* (*dense*) al final de la red al igual que se hace en las redes neuronales convencionales. Estas capas generan una abstracción de alto nivel a partir de los datos y preparan a la red para la clasificación (Pouyanfar et al., 2018).

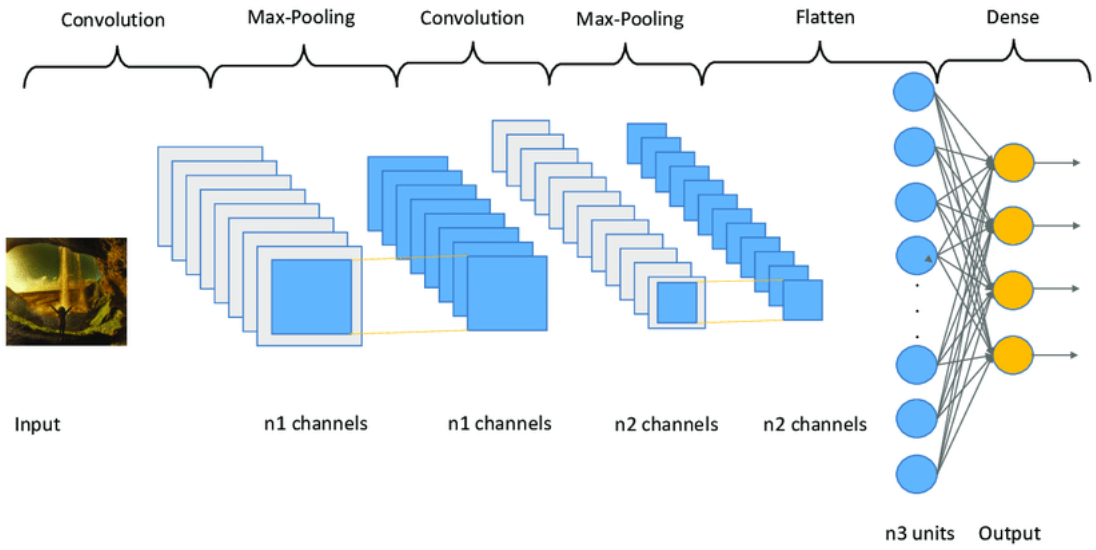

Figura 2.1: Ejemplo de una arquitectura general de una red CNN.

En la última capa (capa de salida) se suele utilizar una función *Softmax* en aplicaciones de clasificación. Gracias a ella, se obtiene la probabilidad con la que la entrada pertenece a una determinada clase (Pouyanfar et al., 2018).

El artículo de Chen et al. (2021) se utiliza este tipo de redes para la detección de *spindles* del sueño en el dataset de DREAMS. Su arquitectura está formada por un total de 6 capas: 3 convolucionales con función de activación ReLU y normalización, 1 capa pooling y 2 capas *dense*. Además, se obtiene información a partir de la entropía obtenida a partir de las épocas EEG que, combinado con la CNN se obtiene la predicción final. En este TFM se ha utilizado este tipo de arquitectura para la elaboración de distintos modelos detallados en capítulos posteriores.

2.3.4.2 RECURRENT NEURAL NETWORK (RNN)

Otro tipo de redes ampliamente utilizadas en la dinámica del *deep learning* son las RNNs. Estas se caracterizan por utilizar la información de manera secuencial en la red. Esta propiedad es esencial en ciertas aplicaciones para conocer la secuencia de los datos. Un ejemplo típico de aplicación es en reconocimiento de voz, aunque también puede utilizarse en otros ámbitos como es la ingeniería biomédica (Pouyanfar et al., 2018).

Uno de los problemas que presenta es su alta sensibilidad a los gradientes ya que estos pueden decaer o aumentar exponencialmente durante el entrenamiento del sistema. Para evitar esto se utilizan bloques de memoria que al almacenan los estados temporales de la red (Pouyanfar et al., 2018). Los más utilizados en análisis de señales biomédicas son los *Long short-term memory cells* (LSTM) para el análisis de EEG (Kulkarni et al., 2019).

La manera de utilizar estas redes para la detección de *spindles* es combinando CNN con RNN como sucede con la arquitectura de “SpindleNet” utilizada en el artículo de Kulkarni et al. (2019). Otro ejemplo de estas arquitecturas lo vemos en Wei, Ventura, Ryan, et al.

(2022) donde se utiliza una LSTM bidireccional que mejora el procesado de la señal. Debido al alto coste computacional que implica el uso de RNNs, no se han utilizado en este TFM.

2.3.4.3 U-NET

U-Net es un tipo de arquitectura dentro de las CNN muy utilizada para la segmentación de imágenes médicas. Las CNN convencionales se utilizan principalmente para tareas de clasificación. Sin embargo, en muchos casos como en la segmentación de imágenes biomédicas, se desea que la salida sea muestra a muestra y que permita, a partir de una señal, extraer la localización de los eventos de interés. La manera de poder conseguir esto es utilizando redes U-Net (Taniguchi et al., 2015).

La estructura general de estas redes se puede ver en la Figura 2.2, donde se puede ver que la red consta de dos "caminos". El primero de ellos conocido como codificador (o *encoder*) que es el encargado del *downsampling* para extraer la información de los datos (la imagen) mediante bloques convolucionales. El segundo camino es el decodificador (o *decoder*) encargado del *upsampling* que permite obtener la localización precisa de los eventos de interés mediante etapas de convolución inversa. Ambos caminos están formados por un conjunto de capas de convolución y *pooling* lo cual le permite a la arquitectura profundizar en las características de los datos de entrada (Taniguchi et al., 2015).

Este tipo de redes son muy utilizadas en la detección de *spindles* del sueño ya que permiten obtener el inicio, final y duración de los *spindles*. Información muy útil cuando se está trabajando con señales EEG. Ejemplos de estas redes las podemos encontrar en el artículo Kaulen et al. (2022), donde se emplea una arquitectura con un menor número de capas que en el estudio de You et al. (2021) con el modelo "SpindleU-Net". Comparando los resultados de ambos artículos se puede concluir que cuanto más se profundice en este tipo de redes, mejores resultados se obtendrán. Además, gracias al uso de las U-Net, se pueden obtener medidas de error en cuanto a la duración y al comienzo del *spindle*, las cuales no se obtienen con otro tipo de arquitecturas. En este TFM se han utilizado arquitecturas de U-Net para así poder comparar los resultados obtenidos con este modelo respecto al uso de una CNN convencional.

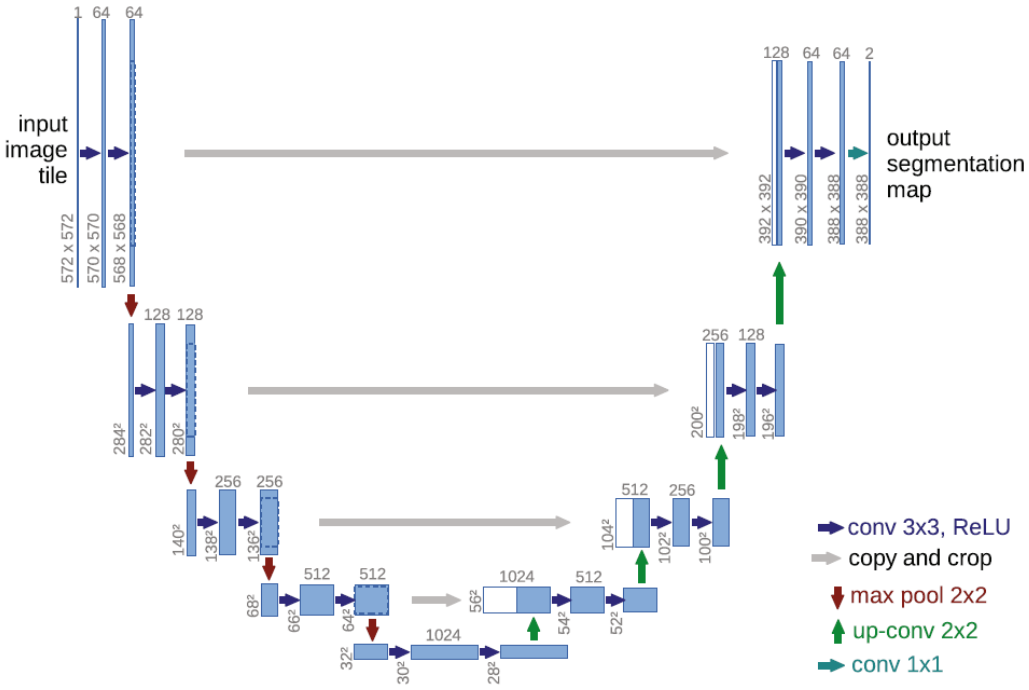

Figura 2.2: Arquitectura general de una U-Net: red convolucional para segmentación de imágenes biomédicas (Taniguchi et al., 2015)

2.3.5 ALGORITMOS DE POST-PROCESADO

Una vez obtenidos los resultados de clasificación, es habitual utilizar un post-procesado de la señal con el obtenido de reducir los falsos positivos en la predicción. Una técnica muy utilizada para ello es modificar las detecciones de *spindles* cuando la duración sea menos que 0,5 segundos (Wei, Ventura, Ryan, et al., 2022).

Si se emplea una CNN convencional, la forma típica de hacerlo sería eliminar las predicciones marcadas como “spindles” cuando las etapas adyacentes están etiquetadas como “no spindles”, dando a entender que lo que se ha marcado como clase positiva en realidad no lo sea (Wei, Ventura, Ryan, et al., 2022). Sin embargo, si utilizamos un modelo U-Net, la forma de eliminar este *spindle* simplemente se reduce a obtener su duración y si resulta menor que 0.5 segundos se procede a eliminar ese falso positivo (You et al., 2021).

CAPÍTULO 3: MATERIALES Y MÉTODOS

En este capítulo se pretende detallar la metodología llevada a cabo en este estudio para conseguir desarrollar un algoritmo que detecte de forma automática *spindles* del sueño en niños aplicando técnicas de *deep learning*.

En el primer apartado "3.1 Sujetos y señales" se describe la población bajo estudio y señales con las que finalmente se ha trabajado del EEG. Posteriormente se detallan todas las fases por las que se ha pasado para lograr el objetivo final: pre-procesado de datos, balanceo de datos, modelos implementados, post-procesado de las predicciones y validación de los algoritmos.

Debido a que se han utilizado dos arquitecturas diferenciadas (CNN convencional y U-Net), hay etapas que se han realizado de distinta forma. Los modelos basados en CNN se han generado siguiendo aproximadamente las indicaciones del artículo de Wei, Ventura, Ryan, et al. (2022), mientras que los modelos de U-Net se han basado en el estudio de You et al. (2021).

3.1 SUJETOS Y SEÑALES

En este apartado se pretende presentar la base de datos que se ha empleado para realizar el estudio, así como presentar al tipo de pacientes sobre los cuales se han recogido los distintos datos. Además, se detalla el canal del EEG con el que se ha trabajado a partir del cual se han podido extraer y seleccionar las distintas características.

3.1.1 POBLACIÓN BAJO ESTUDIO Y SEÑALES A TRABAJAR

El objetivo principal de este TFM se centra en desarrollar un algoritmo que detecte de manera automática *spindles* del sueño en niños que sufren AOS. Se ha trabajado concretamente con el EEG de 9 sujetos de edades comprendidas entre 6 y 9 años, de los cuales 4 de ellos son de sexo femenino y 5 masculino. Estas señales han sido proporcionadas por el Hospital Universitario Rio Hortega de Valladolid y son las mismas que se utilizaron para el TFG con el objetivo de que la comparación de los resultados sea lo más fiable posible.

Sujeto	Edad (años)	Peso (kg)	Altura (cm)	Sexo	Fecha PSG
COG001	6.7	28	124	Femenino	28/01/2020
COG002	7.3	31	130	Femenino	04/02/2020
COG003	8.7	40.5	132	Masculino	25/02/2020
COG004	8.1	36	138	Masculino	26/02/2020
COG005	9.6	33.8	138	Femenino	04/03/2020
COG006	7.6	28	125	Femenino	05/03/2020
COG007	7.5	32	135	Masculino	10/03/2020
COG008	6.4	17	113	Masculino	11/03/2020
COG009	6.6	23	117	Masculino	12/03/2020

Tabla 3.1: Características de los sujetos participantes en el estudio.

Las bases de datos no solo contienen las señales de cada sujeto junto con las marcas de *spindles*, sino que también cuentan con ciertas características de cada uno de ellos (ver Tabla 3.1). Un experto en el ámbito de la AOS ha sido el encargado del etiquetado de los *spindles* para los canales C3, C4, Fp1 y Fp2 del EEG.

Las señales EEG se grabaron, para cada sujeto, empleando los canales C3, C4, F3, F4, O1 y O2 referenciados al mastoide M2. Todos los registros presentan una duración de 10 horas y todas las señales han sido muestreadas a la misma frecuencia de 500 Hz. De todos los canales mencionados, únicamente se ha empleado el C3-M2 para la realización del algoritmo clasificador debido a que este canal es el más utilizado en las publicaciones del estado del arte además de que este fue el canal utilizado en el previo TFG.

3.1.2 ENTRENAMIENTO, VALIDACIÓN Y TEST

Cuando se aplican algoritmos basados en IA, una práctica habitual consiste en dividir la base de datos globa en tres subbases de datos con distintas finalidades: conjunto de entrenamiento (*training set*), conjunto de validación (*validation set*) y conjunto de test (*test set*). Estos grupos se forman de tal manera que cada sujeto sólo pertenezca a uno de ellos, por lo que los datos que contienen cada uno de ellos son independientes entre sí (Eagan et al., 2019).

El **conjunto de entrenamiento** se emplea para entrenar los modelos. Por ello, es importante que los datos que forman este grupo sean de calidad y estén bien preparados, ya que el modelo irá aprendiendo en base a ello y generará las predicciones. Además, el *training set* es el que presenta una mayor cantidad de datos en comparación con los otros. El **conjunto de validación**, por su parte, se utiliza para evaluar el modelo entrenado y así poder ajustar el valor de los hiperparámetros del modelo hasta optimizarlos al máximo. Normalmente se emplea también este grupo de validación para evitar el problema del sobreajuste en los modelos modificando los parámetros. Por último, el objetivo del **conjunto de test** reside en evaluar el algoritmo final después de ser entrenado y ajustado (Eagan et al., 2019).

La estrategia que se ha seguido para dividir la base de datos en los distintos subgrupos ha sido repartir los sujetos de forma similar a la que se realizó con el TFG. (Pacho Velasco, 2022). Teniendo en cuenta que en dicho trabajo no fue necesario un conjunto de validación y que se repartieron 5 sujetos a entrenamiento y 4 a *test*, para este nuevo proyecto se ha optado por utilizar 4 sujetos para entrenamiento, 1 para validación y 4 para

test. Cabe destacar que se han hecho las divisiones utilizando los mismos sujetos para ambos proyectos. De esta forma el 44.44% de los datos son para entrenamiento, el 11,11% para validación y el 44.5% restante para test.

3.2 PREPROCESADO DE DATOS

Para poder caracterizar los *spindles* del sueño en niños, resulta necesario realizar varias transformaciones a las señales originales con el objetivo de eliminar ruido e interferencias. Por ello se ha filtrado la señal original entre 0.5-30 Hz para eliminar, no solo la componente de ruido procedente de la corriente eléctrica a 50 Hz, sino también para descartar información no necesaria superior a 30 Hz. En la Figura 3.1 se representa el canal original junto con el canal filtrado en las frecuencias anteriormente indicadas.

Debido a que se han empleado dos arquitecturas de *deep learning* para la detección de *spindles*, las señales finalmente a utilizar en cada caso han sido tratadas de forma distinta.

Las señales utilizadas con los modelos basados en una CNN convencional han sido divididas en épocas de duración 0.5 segundos con un *overlap* de 0.25 segundos siguiendo las recomendaciones del estudio de (Wei, Ventura, Ryan, et al., 2022) y de la misma forma que se realizó previamente con el TFG (Pacho Velasco, 2022). Además, la elección de estos tiempos no ha sido al azar, sino que se basa en que el tiempo mínimo requerido para la duración de un *spindle* son 0.5 segundos.

Por otro lado, las señales utilizadas con los modelos basados en U-Net han sido divididas en fragmentos de 20 segundos siguiendo las recomendaciones del artículo de (You et al., 2021). De esta forma la salida de la U-Net tendrá las mismas dimensiones que la entrada y se localizarán los *spindles* muestra a muestra.

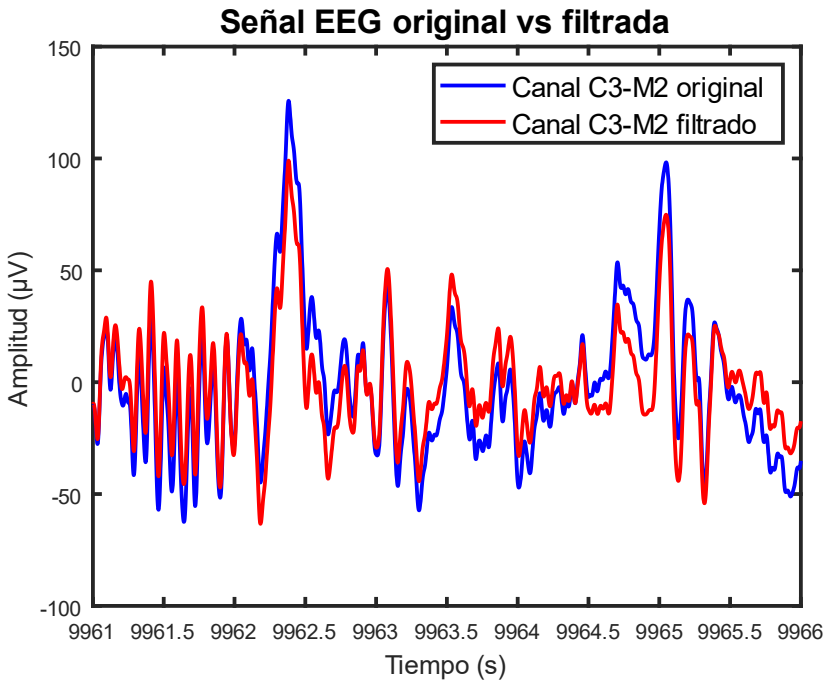

Figura 3.1: Representación del canal original con respecto al canal filtrado entre 0.5-30 Hz

3.3 BALANCEO DE DATOS

El desequilibrio entre datos es un problema bastante frecuente en aplicaciones de señales biomédicas que ocurre cuando el tamaño de las clases de datos presenta una distribución irregular. La mayoría de algoritmos basados en DL trabajan bien cuando los datos están balanceados, sin embargo, cuando no lo están tienden a estar sesgados hacia la clase mayoritaria provocando que sus predicciones sean ineficientes y erróneas (Boughorbel *et al.*, 2017).

En este estudio se ha trabajado con una base de datos que presenta un 1.15% de clase positiva (*spindle*) y un 98.85% de clase negativa (*no spindle*), por lo que se puede ver que el problema del desequilibrio entre los datos es muy significativo y conducirá a problemas en las predicciones. Para paliar esta cuestión se ha trabajado con distintas soluciones para ver cuál de todas ellas tiene mejores resultados. En este apartado se van a diferenciar las soluciones implementadas para los modelos con CNN de forma separada a las implementadas con U-Net.

En primer lugar, se va a hablar de las soluciones de los modelos con CNN y as combinaciones que se han propuesto para ello. Como los *spindles* se localizan únicamente durante las etapas N2 y N3 del sueño, todas las señales utilizadas se han recortado a estas dos etapas utilizando los algoritmos del TFG de Calvo Merino (2020) reduciendo en gran medida el número de elementos pertenecientes a la clase mayoritaria (no *spindle*).

En vista que esta técnica era eficiente pero no lo suficiente, se optó por utilizar algoritmos de *undersample* para equilibrar ambas clases. Se utilizó la técnica de *RandomUnderSample* (RUS) (AT et al., 2016) de tal forma que el conjunto de entrenamiento final estaba formado por el mismo número de muestras pertenecientes a la clase positiva que a la clase negativa.

Otra técnica que se utilizó fue ajustar los pesos de pérdida durante la etapa de entrenamiento aplicando la técnica *custom loss weights*. Ajustar estos pesos implica dar más importancia a la clase minoritaria haciendo que el modelo le preste más atención a esta clase. De esta forma se consigue que, durante el entrenamiento del sistema, los parámetros que se van escogiendo dentro del proceso iterativo minimicen la función de pérdidas y el error consiguiendo que el algoritmo aprenda a clasificar correctamente todas las muestras (Ebert-Uphoff et al., 2021).

Para abordar el problema de desequilibrio entre clases también se ha utilizado la función de pérdidas *focal loss*, diseñada específicamente para este tipo de tareas. Esta función aborda el problema modificando los pesos en función de la dificultad que presentan las muestras, siendo mayor cuando se trata de un ejemplo perteneciente a la clase minoritaria (Lin et al., 2017).

Además, también se han utilizado técnicas de *oversampling* de tal forma que se han generado muestras de *spindles* sintéticas para aumentar el número de elementos pertenecientes a la clase minoritaria. Dentro de este ámbito se han utilizado distintas técnicas y, en algunos casos, se han combinado varias soluciones con el fin de aumentar el número de muestras. Se detallan los recursos utilizados:

1. Repetición de segmentos: esta solución implica aumentar el número de *spindles* repitiendo las muestras de *spindles* que ya había previamente.
2. *Sliding window* o técnica de la ventana deslizante: esta técnica consiste en aumentar el número de *spindles* creando una ventana deslizante sobre el *spindle* real de tal forma que se obtienen varias muestras del mismo *spindle* modificando el inicio y final, pero sin variar la duración. Esta técnica permite aumentar considerablemente el número de elementos de la clase minoritaria sin alterar la señal del EEG.
3. Adición de ruido gaussiano: con esta técnica se consigue aumentar el número de muestras añadiendo ruido gaussiano de media 0. Esta es una buena solución para generar una gran variedad de *spindles.*
4. *Mirror* o técnica del espejo: esta técnica consiste en generar muestras de *spindles* sintéticos a parir de los ya existentes dándoles la vuelta. De tal forma que así se consigue obtener el doble de muestras sin modificar la frecuencia de la señal.

Como ya se ha comentado anteriormente, se han generado modelos en los que se combinan varias soluciones para tratar de paliar el problema del desequilibrio entre las clases.

Por último, se va a hablar de la técnica utilizada para los modelos con U-Net. Como este tipo de redes trabajan de diferente manera, no es posible aplicar las soluciones anteriores. Para evitar en mayor medida el problema del balanceado, simplemente se ha recortado la señal original del EEG de tal forma que se ha trabajado únicamente con las etapas N2 y N3 del sueño, sin aplicar ningún otro tipo de algoritmo.

3.4 ALGORITMOS DE CLASIFICACIÓN IMPLEMENTADOS

En este apartado se presentan las distintas versiones de los modelos de DL utilizados identificándolos con un nombre representativo y detallando las características de cada uno de ellos. Se puede ver que hay dos tipos de modelos. Por un lado, se encuentran las versiones realizadas cuando se implementa una CNN convencional y, por otro lado, está el modelo basado en una U-Net.

3.4.1 MODELO 1: CNN BASE (CNN_V1)

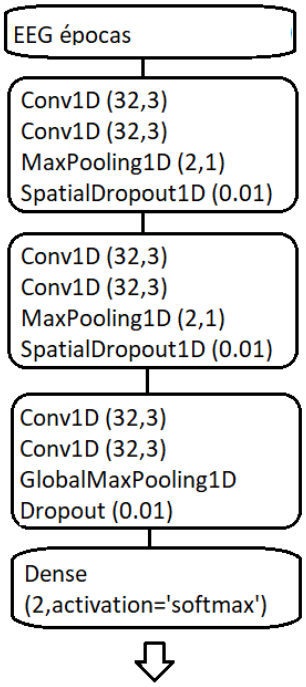

Figura 3.2: Estructura CNN inicial basada en el artículo Wei, Ventura, Ryan, et al., (2022).

En una primera toma de contacto con el objetivo de desarrollar un algoritmo capaz de detectar *spindles* del sueño con técnicas de DL, se llevó a cabo la implementación de un primer modelo con nombre cnn_v1 a partir del cual se han ido modificando diversos aspectos dando lugar al resto de versiones posteriores.

Este modelo se caracteriza por seguir al pie de la letra la estructura de CNN (ver Figura 3.2) propuesta por el artículo de Wei, Ventura, Ryan, et al., (2022). Esta arquitectura está formada por capas Conv1D, Maxpooling y SpatialDropout principalmente empleadas para extraer las características más relevantes de los datos y su repetición múltiple que permite un aprendizaje jerárquico de estas características antes de su salida final de la red. Finalmente, la red termina con una capa *dense* que combina todas las características extraídas para realizar con todo ello la clasificación. Además, en este caso no se aplica ninguna técnica específica para el balanceo de datos (ni siquiera se ha recortado la señal en las etapas N2 y N3 del sueño) y la entrada al algoritmo son épocas de 0.5 segundos con un *overlap* de 0.25 segundos.

Para la implementación de este modelo se ha utilizado como optimizador la función *stochastic grdient descent* y como función de pérdidas *binary crossentropy* (Wei, Ventura, Ryan, et al., 2022) (ver código en Anexo B.1).

Por último, destacar que la repartición de sujetos en este modelo entre los grupos de entrenamiento (5 sujetos), validación (2 sujetos) y test (2 sujetos) ya que inicialmente se trató de replicar el artículo original. Sin embargo, debido a que posteriormente el objetivo

era comparar los resultados obtenidos con el TFG, tanto este modelo como los modelos cnn_v2 y cnn_v3 no servirán como comparación en el siguiente capítulo.

3.4.2 MODELO 2: CNN CON BALANCEO RUS EN ENTRENAMIENTO (CNN_V2)

La novedad de este nuevo modelo es que se implementa el algoritmo RUS en el conjunto de entrenamiento para el balanceo de datos. Se emplea la misma red y las mismas características que el modelo anterior. En la Tabla 3.2 se puede ver cómo se ha conseguido reducir el número de muestras pertenecientes a la clase "no *spindle*" del conjunto de entrenamiento con el algoritmo.

	DATASET INICIAL	DATASET BALANCEADO
NO SPINDLES	782889	9106
SPINDLES	9106	9106

Tabla 3.2: Número de muestras final empleando *RandomUnderSample* para el modelo cnn_v2

3.4.3 MODELO 3: CNN CON BALANCEO RUS EN ENTRENAMIENTO Y VALIDACIÓN (CNN_V3)

Este modelo se diferencia del anterior en que el balanceado de datos utilizando el algoritmo *RandomUnderSample* (AT et al., 2016) se realiza no solo en el conjunto de entrenamiento, sino también en el conjunto de validación. El resto de las características son similares.

3.4.4 MODELO 4: CNN ANTERIOR Y CNN PROFUNDA (CNN_V4)

Dentro de este modelo, se han aplicado varias novedades que difieren del resto de las versiones anteriores. Por un lado, se han diferenciado pruebas realizadas con la misma red que las anteriores aplicando diferentes técnicas de balanceo de datos. Por otro lado, estas mismas técnicas han sido aplicadas a un modelo formado por una CNN mucho más profunda que la anterior con el objetivo de caracterizar la señal de entrada mucho mejor y así ser capaz de extraer información mucho más útil a la hora de realizar las predicciones. Por este motivo, este modelo consta de dos partes: CNN anterior y CNN profunda.

Las señales utilizadas en esta versión de modelo y en las sucesivas han sido recortadas en las etapas N2 y N3 del sueño. Además, a partir de esta nueva versión la repartición de sujetos se ha realizado de la misma forma que se hizo en el TFG: 4 sujetos para entrenamiento, 1 sujeto para validación y 4 sujetos para test. Por consiguiente, a partir de

esta versión, los resultados obtenidos podrán ser directamente comparados con los resultados del TFG.

3.4.4.1 CNN ANTERIOR

Aplicando la misma CNN que en versiones anteriores, se han aplicado diversos algoritmos con el objetivo de evitar el desequilibrio entre datos. En concreto se han utilizado las siguientes técnicas:

- Únicamente empleando las señales en N2 y N3.
- N2 y N3 junto con la función de *Custom Loss Weights* (ver código en Anexo B.2).
- N2 y N3 junto con la función de *Focal Loss*. Los parámetros utilizados para ello han sido gamma=2, y Alpha=0.25 (ver código en Anexo B.3).

3.4.4.2 CNN PROFUNDA

Las mismas técnicas de balanceado de datos han sido utilizadas con esta red. La estructura de la red está basada en el artículo de Sors et al. (2018), donde se construye una CNN para clasificar las fases del sueño del EEG. En la Figura 3.3 se pueden ver las capas que constituyen esta nueva CNN la cual será sustituida por la anterior y utilizada por las nuevas versiones del modelo. Como se puede ver, se aumenta de 3 bloques a 7 en esta nueva arquitectura, así como se incrementa el tamaño del filtro con la profundidad de la red (ver código en Anexo B.4).

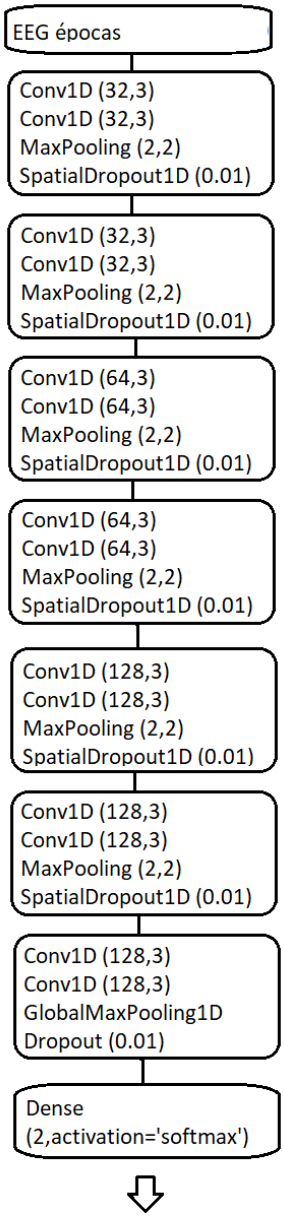

Figura 3.3: Estructura de la CNN profunda

3.4.5 MODELO 5: CNN PROFUNDA CON DROPOUT (CNN_V5)

En este modelo se ha utilizado la CNN profunda de la versión anterior y se ha modificado el parámetro *dropout_rate* para averiguar con qué valor de todos ellos se obtienen los mejores resultados y así conseguir minimizar el *overfitting*. En concreto se han probado los valores: 0.00, 0.01, 0.1, 0.05 y 0.08.

Gracias al modelo cnn_v5 se ha podido ver que el parámetro que hace que el modelo se ajuste mejor a los datos ha sido para un *dropout* de 0.05, por lo que en el resto de los modelos este será el valor del parámetro utilizado.

3.4.6 MODELO 6: CNN CON OVERSAMPLING (CNN_V6)

En esta nueva versión del modelo, se han hecho cambios aplicando técnicas de *oversampling* para el problema del desequilibrio entre clases, manteniendo la misma CNN profunda con un *dropout* de 0.05. En este caso se han probado tres combinaciones distintas para ver qué técnica de todas ellas genera mejores resultados.

3.4.6.1 SLIDING WINDOW

En primer lugar, se han generado nuevas muestras de *spindles* aplicando una ventana deslizante sobre los *spindles* ya existentes en el conjunto de entrenamiento y validación. El tamaño de la ventana ha sido de 0.5 segundos (250 muestras), al igual que el de las épocas anteriores, y se ha establecido un *step* de 5 muestras. Es decir, cada 5 muestras de obtiene un nuevo elemento *spindle*. En la Tabla 3.3 se puede ver el número de *spindles* totales conseguidos gracias a este método. Se ha conseguido que aproximadamente el 35% de las muestras pertenezcan a *spindle*, frente al 1% que había inicialmente.

	SPINDLES INICIALES	SPINDLES GENERADOS	SPINDLES TOTALES	NO SPINDLES TOTALES
ENTRENAMIENTO	7788	193973	201761	385845
VALIDACIÓN	1198	30165	31363	96635

Tabla 3.3: Relación *spindles* vs no *spindles* tras aplicar la ventana deslizante.

3.4.6.2 SLIDING WINDOW + MIRROR

A pesar de que con el método anterior se han conseguido aumentar considerablemente el número de muestras, se han generado *spindles* sintéticos aplicando la técnica del espejo sobre los *spindles* que había inicialmente sin técnicas de balanceo. En la Tabla 3.4 se puede ver el número de *spindles* finales conseguidos con esta metodología. Se ha conseguido que las muestras de *spindles* constituyan el 36% de la base de datos.

	SPINDLES INICIALES	SPINDLES GENERADOS	SPINDLES TOTALES	NO SPINDLES TOTALES
ENTRENAMIENTO	7788	193973 + 7788	217337	385845
VALIDACIÓN	1198	30165 + 1198	32561	96635

Tabla 3.4: Relación *spindles* vs no *spindles* tras aplicar la ventana deslizante y espejo.

3.4.6.3 SLIDING WINDOW + GAUSSIAN NOISE

En esta última prueba se han combinado las técnicas de ventana deslizante junto con la adición de ruido gaussiano para generar nuevas muestras. Es decir, se genera un número

de muestras sintéticas aplicando *sliding window* y posteriormente, se genera el mismo número de muestras sumándole ruido gaussiano a las anteriores.

En este caso se ha utilizado un *step* de 10 muestras para la ventana deslizante y además se han generado muestras sintéticas sobre las nuevas obtenidas con distinta SNR para ver cuál mejoraba los resultados. Los valores de SNR probados han sido: 0dB, 10 dB, -10 dB y -20 dB. Con ello se ha podido observar que la SNR de 0 dB ha sido la mejor. En la Tabla 3.5 se puede ver la relación *spindle vs.* no-*spindle* final. El 34% de los datos ahora pertenece a la clase *spindle*.

	SPINDLES INICIALES	SPINDLES GENERADOS	SPINDLES TOTALES	NO SPINDLES TOTALES
ENTRENAMIENTO	7788	97010 + 97010	201808	385845
VALIDACIÓN	1198	15090 + 15090	31378	96635

Tabla 3.5: Relación *spindles* vs no *spindles* tras aplicar la ventana deslizante y ruido gaussiano.

3.4.7 MODELO 7: CNN Y RUIDO GAUSSIANO (CNN_V7)

Este modelo corresponde a la última versión obtenida empleando una arquitectura de CNN convencional. En vista de que la adición de ruido gaussiano de media 0 y desviación típica 1 (SNR=0) ha tenido resultados interesantes, se han generado dos modelos variando la base de datos aplicando esta técnica.

3.4.7.1 ADICIÓN DE RUIDO GAUSSIANO

En esta primera prueba simplemente se ha entrenado al sistema con los *spindles* iniciales y el mismo número de *spindles* sintéticos añadiéndoles ruido gaussiano con SNR 0 dB. En la Tabla 3.6 se puede ver la relación de *spindles* finalmente adquiridos.

	SPINDLES INICIALES	SPINDLES GENERADOS	SPINDLES TOTALES	NO SPINDLES TOTALES
ENTRENAMIENTO	7788	7788	15576	385845
VALIDACIÓN	1198	1198	2396	96635

Tabla 3.6: Relación *spindles* vs no *spindles* tras aplicar ruido gaussiano sobre las muestras originales de *spindles*.

3.4.7.2 ADICIÓN DE RUIDO GAUSSIANO DOBLE

En esta nueva ocasión se ha entrenado al sistema aplicando la ventana deslizante (*step*=10) y se ha añadidos dos nuevas tandas de *spindles* añadiendo sobre éstas ruido gaussiano (SNR=0 dB). En la Tabla 3.7 se recoge el número de muestras final obtenido, perteneciendo el 43% de la base de datos a la clase *spindle*.

	SPINDLES INICIALES	SPINDLES GENERADOS	SPINDLES TOTALES	NO SPINDLES TOTALES
ENTRENAMIENTO	7788	97010 + 2*97010	298818	385845
VALIDACIÓN	1198	15090 + 2*15090	46468	96635

Tabla 3.7: Relación *spindles* vs no *spindles* tras aplicar *sliding window* y ruido gaussiano sobre las muestras originales de spindles.

3.4.8 MODELO 8: U_NET

Este último modelo es independiente de los demás. La clasificación de *spindles* en este caso se hace muestra a muestra y para ello de aplica una arquitectura de red U-Net similar a la del estudio de You et al. (2021) (ver Figura 3.4).

En la parte del codificador (*downsampling*), se puede ver que el número de capas va aumentando al doble en cada bloque convolucional consiguiendo profundizar más en la red. La entrada a dicha red corresponde a un segmento del EEG de duración 20 segundos (un total de 10000 muestras teniendo en cuenta que la frecuencia de muestreo de la señal es de 500 Hz). Los bloques que constituyen esta red están formados por varias capas convolucionales de tamaño 11 seguido de una capa *dropout* de tasa 0.2 (ver código en Anexo B.5).

En cuanto a la parte del decodificador, se realiza un sobremuestreo (*upsampling*) empleando bloques de convolución transpuestos. Debido a la simetría de la estructura U-Net, el tamaño de la salida corresponde al de la entrada, obteniendo, por lo tanto, una predicción muestra a muestra de la entrada. Por último, indicar que en la salida de esta red se emplea una función de activación *softmax* para generar la probabilidad de que esa muestra pertenezca o no a la clase *spindle* (ver código en Anexo B.5).

Para paliar el problema del desequilibrio entre datos simplemente se ha recortado la señal del EEG a las fases N2 y N3 del sueño.

Es importante recalcar que en este modelo se caracteriza por realizar la predicción muestra a muestra de la señal de entrada y que se han obtenido resultados para distinta probabilidad de predicción en la función *softmax*. Se han utilizado los umbrales de 0.5 y 0.6, obteniendo mejores resultados para el valor de 0.5.

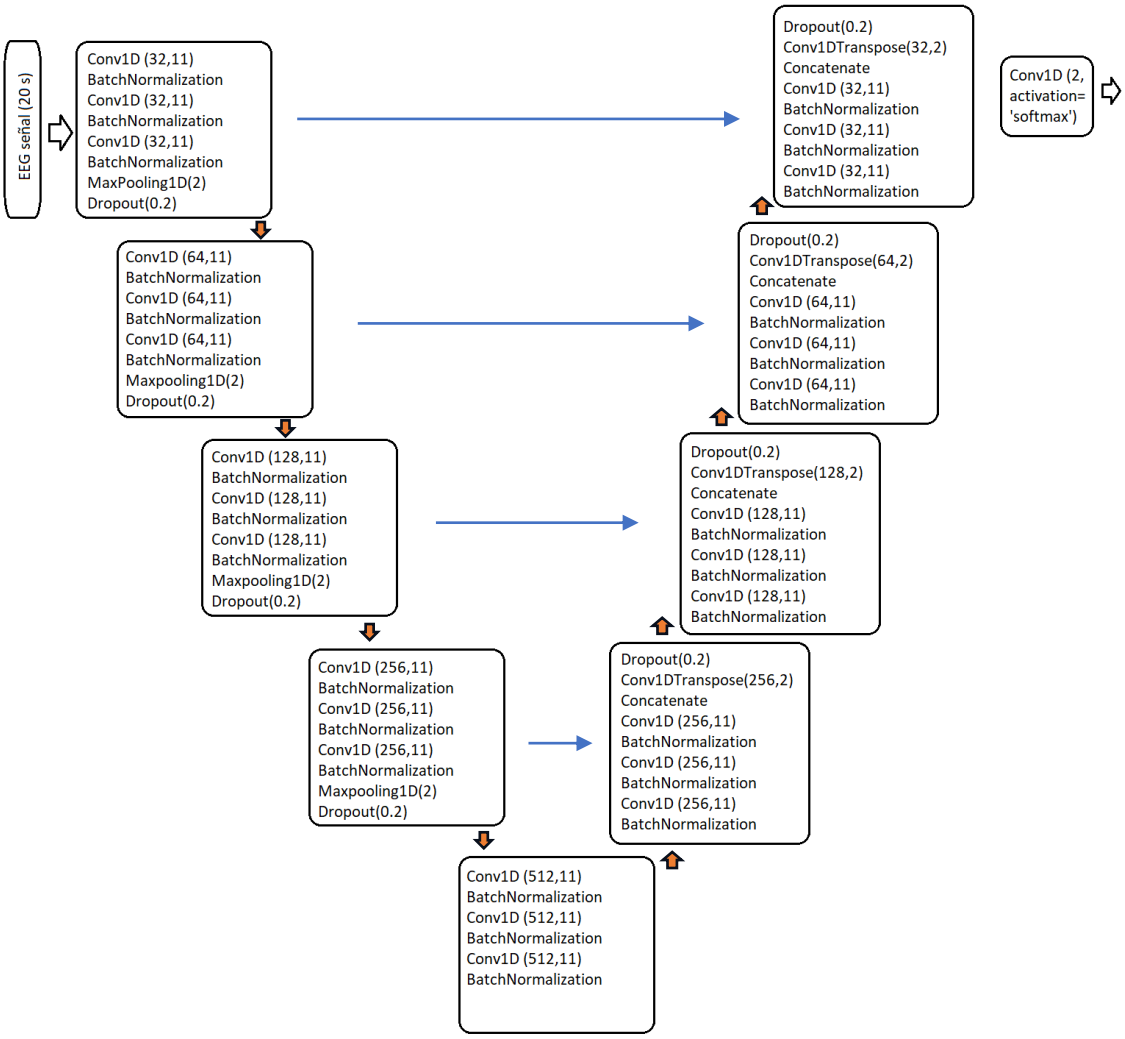

Figura 3.4: Arquitectura de la red U-Net.

3.5 POST-PROCESADO DE LA SEÑAL

Una vez obtenidas las predicciones de los modelos, los resultados finales se han post-procesado con el objetivo de evitar falsos positivos. En los algoritmos basados en CNN convencional, se han eliminado las etiquetas predichas como “spindle” siempre que las etapas adyacentes a esta estén etiquetadas como “no spindle” (Wei, Ventura, Ryan, et al., 2022). En el caso de la U-Net, se han eliminado las predicciones de “spindle” en función de su duración, de tal forma que si se detecta un *spindle* de duración menor a 0.5, este será descartado (You et al., 2021). Todos los resultados obtenidos se han post-procesado excepto la salida de los modelos cnn_v1, cnn_v2 y cnn_v3.

3.6 MÉTRICAS DE RENDIMIENTO

En los problemas de clasificación, existen métricas empleadas para evaluar el rendimiento de los algoritmos con el objetivo de mejorar los hiperparámetros de optimización del modelo. En este proyecto, se han utilizado herramientas para evaluar la efectividad del clasificador en la fase de test. Además, debido a que los modelos presentan dos tipos de arquitecturas diferenciadas (CNN convencional y U-Net), se han obtenido diferentes tipos de métricas que se muestran a continuación.

3.6.1 MÉTRICAS EMPLEADAS CON CNN

La evaluación de los modelos que presentan una arquitectura basada en CNN convencional se ha realizado empleando métodos de discriminación basados en lo que se conoce como matriz de confusión y en el cálculo del error cuadrático medio.

En primer lugar, se van a tratar las métricas obtenidas a partir de la matriz de confusión. En la Tabla 3.8 se representan en fila las clases predichas por el clasificador y en columna las clases reales, de tal forma que se reflejan en ella los valores TP (*true positive*), TN (*true negative*), FP (*false positive*) y FN (*false negative*). Los valores TP y TN denotan el número de valores predichos positivos y negativos que concuerdan con los valores reales, mientras que FP y FN se emplean para identificar el número de falsos positivos y falsos negativos que se han dado tras la clasificación (Hossin & Sulaiman, 2015).

	Clase predicha negativa	Clase predicha positiva
Clase real negativa	TN	FP
Clase real positiva	FN	TP

Tabla 3.8: Matriz de confusión

En base a estos valores, se obtienen varias medidas de evaluación que permiten discriminar la solución más optima. Entre todas ellas destacan (Hossin & Sulaiman, 2015):

- **Exactitud** (*accuracy*, Acc): es la métrica de evaluación más utilizada en la práctica tanto para problemas de clasificación binaria como multiclase. Se basa en medir el porcentaje de predicciones correctas frente al total de las predicciones (*ecuación 14*). Una desventaja de este parámetro reside en que cuando hay desequilibrio entre los datos, suele ser favorable a la clase mayoritaria, proporcionando así un resultado poco convincente.

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} * 100 \quad (14)$$

- **Sensibilidad o exhaustividad** (*sensitivity* / *Recall*, Se): este parámetro permite conocer la proporción de casos positivos que fueron correctamente clasificados (*ecuación 15*). Para un modelo perfecto, este valor debería ser 1 o el 100%.

$$Se = \frac{TP}{TP + FN} * 100 \quad (15)$$

- **Especificidad** (*specificity*, Sp): se emplea para medir la fracción de casos negativos correctamente clasificados (*ecuación 16*).

$$Sp = \frac{TN}{TN + FP} * 100 \quad (16)$$

- **Valor predictivo positivo** (*positive predictive value / precision*, PPV): el valor predictivo positivo se utiliza para evaluar el número correcto de casos positivos obtenidos frente a todos los casos positivos, reales o falsos que se han clasificado (*ecuación 17*).

$$PPV = \frac{TP}{TP + FP} * 100 \quad (17)$$

- **Valor-F** (*F1-score*): esta métrica se utiliza para combinar las medidas de *precision* y *recall* en un solo valor. Esto es muy práctico porque facilita la comparación del rendimiento de los modelos. Se calcula con la media ponderada entre ambas (*ecuación 18*).

$$F1 = 2 * \frac{precision * recall}{precision + recall} * 100 \quad (18)$$

- **Coeficiente de correlación de Matthews, MCC** (*Matthews correlation coefficient, MCC*): esta medida estadística informa sobre la concordancia que hay entre los datos reales y las predicciones. El MCC oscila entre -1 y 1, donde el 1 corresponde a la mejor relación entre datos reales y predicciones y el 0 indica que el modelo se comporta de manera aleatoria (Baldi et al., 2000). En esta medida intervienen todos los parámetros de la matriz de confusión (*ecuación 19*).

$$MCC = \frac{TN * TP - FN * FP}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}} * 100 \quad (19)$$

Por otro lado, se ha calculado la raíz del error cuadrático medio (RMSE) ya que es una métrica muy utilizada en problemas de clasificación y permite medir la diferencia entre las soluciones predichas y las deseadas o reales. El RMSE presenta la ventaja de que se puede interpretar como la desviación estándar de la varianza y, además, permite trabajar en las mismas unidades que la variable predicha. Cuando más pequeño es su valor, mejor ajuste tendrá el modelo implementado (*ecuación 20*) (Hossin & Sulaiman, 2015).

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(P_i - A_i)^2} \quad (20)$$

3.6.2 MÉTRICAS EMPLEADAS CON U-NET

La evaluación de los modelos desarrollados con una arquitectura de U-Net es ligeramente distinta a la realizada con los modelos CNN. Gracias al uso de este tipo de redes se pueden obtener resultados muestra a muestra (*point by point*), por evento (*by event*) así como medidas de errores en la duración y comienzo de los spindles de forma similar a como se realiza en el artículo de You et al. (2021).

Las métricas obtenidas con los resultados *point by point* han sido los mismos que se han comentado para el caso de los modelos basados en CNN convencional. Sin embargo, para los resultados por evento ha sido necesario calcular la IoU (*Intersection over Union*). Esta herramienta se emplea para evaluar algoritmos de detección comparando el área de detección de un *spindle* real con el predicho en función de un umbral ɣ. De esta forma se realiza la proporción entre el área del solapamiento de ambos *spindles* frente al área total que forman el *spindle* real y el detectado (*ecuación 21*).

$$\boldsymbol{IoU} = \frac{\boldsymbol{\text{Área de solapamiento}}}{\boldsymbol{\text{Área de la unión}}} \tag{21}$$

A partir de esta medida se han obtenido los parámetros TP, FP y FN siguiendo la siguiente estrategia considerando ɣ de valor 0.2 (You et al., 2021):

1. *Cálculo de TP*: se obtiene un TP cuando el IoU es igual o superior a 0.2, empleando como referencia la etiqueta de las predicciones.
2. *Cálculo de FP*: se obtiene un FP cuando el IoU es inferior a 0.2, empleando como referencia la etiqueta de las predicciones.
3. *Cálculo de FN*: se obtiene un FN cuando el IoU es inferior a 0.2, empleando como referencia la etiqueta real.

Con estos parámetros se han calculado los valores de *precision*, *recall* y *F1-score*.

Por otro lado, se han obtenido los errores de duración y comienzo de un *spindle*, los cuales se han representado en histogramas representando el intervalo de confianza del 95% y el valor medio.

Por último, se ha podido generar una tabla por cada sujeto con la información del número de *spindles* reales y detectados y la duración promedio de ellos.

CAPÍTULO 4: RESULTADOS

Este capítulo está orientado a exponer los resultados obtenidos tras la implementación de las metodologías de DL desarrolladas en el capítulo anterior.

En primer lugar, se expone el análisis de un espectrograma de la señal original con la que se ha trabajado. Esto ha permitido identificar de manera cualitativa los *spindles* con las marcas reales puestas por expertos y así verificar que tanto las señales como las marcas se correspondían para todos los sujetos de la base de datos.

Una vez analizado el espectrograma, se muestran los resultados de cada uno de los modelos diseñados. Cabe destacar que se diferencian dos tipos de resultados diferentes. Por un lado, todos los que tienen que ver con los algoritmos basados en redes CNN y por otro, los basados en la arquitectura U-Net. Además, se han elaborado gráficas con las predicciones obtenidas para los modelos cnn_v7 y U_net.

4.1 ANÁLISIS PRELIMINAR DEL ESPECTROGRAMA

En primer lugar, para verificar que tanto las señales como las marcas etiquetadas por expertos se correspondían a *spindles* del sueño, se han realizado varios espectrogramas de 30 segundos de duración sobre cada uno de los sujetos para estudiar su contenido frecuencial y su distribución de la energía en función de tiempo. Como ya se ha comentado en anteriores capítulos, los *spindles* corresponden a oscilaciones que se dan mayoritariamente en la fase N2 del sueño, aunque también en la fase N3, a frecuencias entre los 8-15 Hz en niños. En la Figura 4.1 se muestra un ejemplo representativo en el que se visualiza el espectrograma del sujeto COG001 en un segmento de 30 segundos. Además, se representan las marcas de *spindles* que se corresponden en ese tiempo, así como la señal correspondiente en el dominio temporal.

Como se puede apreciar en la imagen, los *spindles* anotados por los expertos (Figura 4.1b) sí parecen coincidir con una de estas oscilaciones puesto que la frecuencia obtenida en esos puntos se encuentra entre 8-15 Hz (Figura 4.1a), tal y como se debe esperar teniendo en cuenta que se trata de un EEG de niños entre 6 y 9 años. Además, en la señal EEG (Figura 4.1c) se puede observar que, por su forma, esa anotación sí que corresponde a un *spindle*.

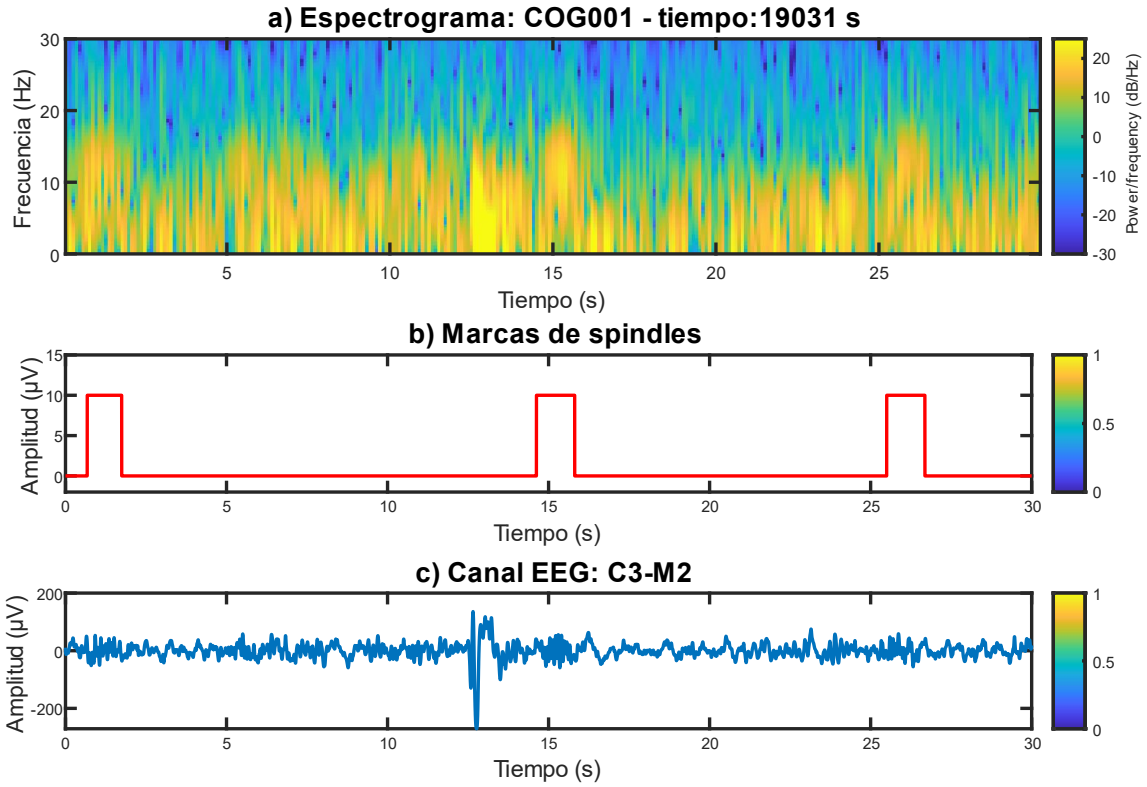

Figura 4.1: Representación de 30 segundos del sujeto COG001 a partir del segundo 19031 de grabación: a) representación del espectrograma, b) representación de las marcas de *spindles* etiquetadas por un experto y c) canal C3-M2 del EEG.

4.2 RESULTADOS DE LOS MODELOS CNN

4.2.1 MODELO CNN_V1

En la Figura 4.2 se puede ver la matriz de confusión obtenida con el modelo CNN base (cnn_v1). Posteriormente, en la Tabla 4.1 se reflejan las métricas obtenidas, muchas de ellas, a partir de la matriz de confusión. Aunque la mayoría de los segmentos “no spindle” se detecten correctamente (Acc=99.80%, Sp=99.91%), el modelo cnn_v1 no tiene una gran capacidad para identificar los *spindles* (Se=30.48%, PPV=33.11%, F1-score=32.00%).

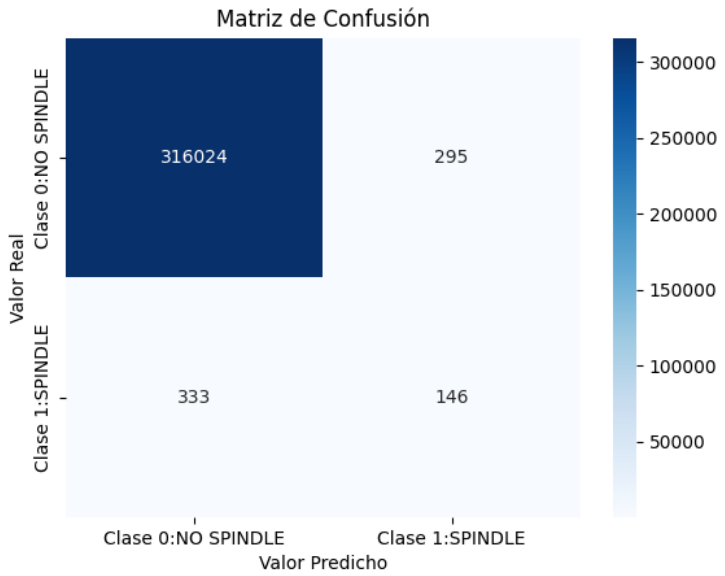

Figura 4.2: Matriz de confusión para el modelo cnn_v1.

Métricas de evaluación	Valor en %
Acc	99.80
Se	30.48
Sp	99.91
PPV	33.11
F1-score	31.74
MCC	31.66
RMSE	4.45

Tabla 4.1: Métricas de evaluación del modelo cnn_v1.

4.2.2 MODELO CNN_V2

En la Figura 4.3 se puede ver la matriz de confusión obtenida con el modelo CNN que incorpora CNN con balanceo RUS en entrenamiento (cnn_v2). Posteriormente, en la Tabla 4.2 se reflejan las métricas obtenidas, muchas de ellas, a partir de la matriz de confusión. Aunque haya más *spindles* correctamente identificados en comparación con el modelo cnn_v1 (Se=90.61% *vs.* Se=30.48%), el balanceo RUS en entrenamiento resulta en una gran cantidad de falsos positivos, obteniéndose valores inferiores de Acc (81.79% *vs*. 99.80%), Sp (81.78% *vs*.99.91%), PPV (0.75% *vs*. 33.11%) y F1-score (1.49% *vs*. 31.74%).

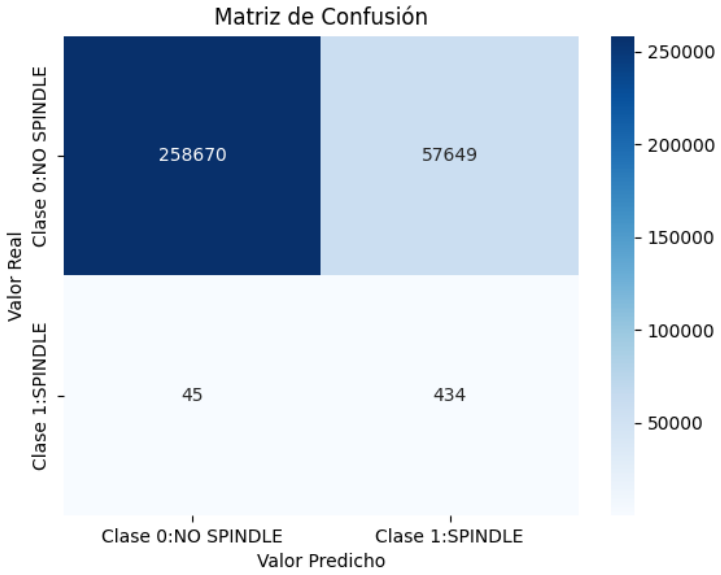

Figura 4.3: Matriz de confusión para el modelo cnn_v2.

Métricas de evaluación	Valor en %
Acc	81.79
Se	90.61
Sp	81.78
PPV	0.75
F1-score	1.49
MCC	7.27
RMSE	42.68

Tabla 4.2: Métricas de evaluación del modelo cnn_v2.

4.2.3 MODELO CNN_V3

En la Figura 4.4 se puede ver la matriz de confusión obtenida con el modelo CNN con balanceo RUS en entrenamiento y validación (cnn_v3). Posteriormente, en la Tabla 4.3 se reflejan las métricas obtenidas, muchas de ellas, a partir de la matriz de confusión. Se puede observar cómo las métricas de rendimiento (Acc, Se, Sp, PPV y F1-score) toman valores intermedios a las obtenidas con los modelos cnn_v1 y cnn_v2.

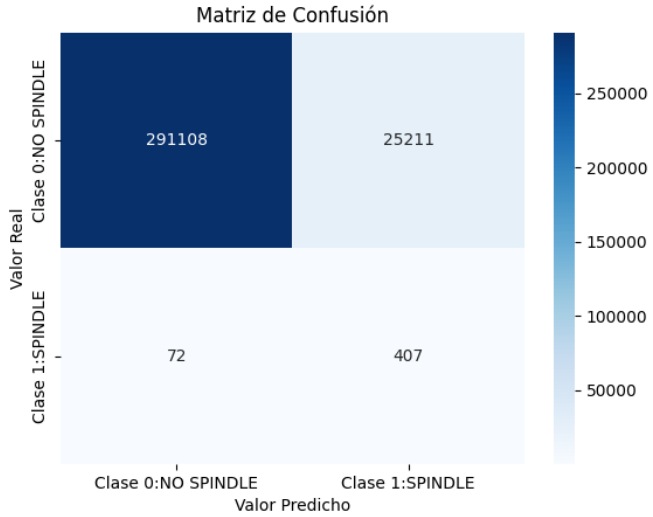

Figura 2: Matriz de confusión para el modelo cnn_v3.

Métricas de evaluación	Valor en %
Acc	92.02
Se	84.97
Sp	92.03
PPV	1.59
F1-score	3.12
MCC	10.97
RMSE	28.25

Tabla 4.3: Métricas de evaluación del modelo cnn_v3.

4.2.4 MODELO CNN_V4

Para este modelo, podemos recordar que se han generado varias propuestas con el fin de paliar el problema del desequilibrio entre datos empleando distintas técnicas:

1. Señales N2-N3
2. Señales N2-N3 + Custom Loss Weights (CLW)
3. Señales N2-N3 + Focal Loss (FL)

Además, por cada método se han utilizado dos tipos de arquitecturas: una basada en el artículo de Wei, Ventura, Ryan, et al. (2022) y otra más profunda basada en el estudio de Sors et al. (2018). En la Tabla 4.4 se pueden ver los resultados obtenidos con ambas arquitecturas. En ella se puede ver que la identificación de los *spindles* aplicando una CNN profunda mejora considerablemente las métricas de Se (74.67% *vs*. 33.52%) y F1-score (53.22% *vs*. 41.86%) dando a entender con esta nueva red se extraen características mucho más relevantes a la hora de realizar la clasificación. Por ello, en el resto de los modelos se ha empleado esta nueva arquitectura CNN profunda.

En las Figuras 4.5, 4.6 y 4.7 se pueden ver las matrices de confusión obtenidas para todos los modelos con CNN profunda aplicando las técnicas N2_N3, CLW y FL. Posteriormente, en la Tabla 4.5 se reflejan las métricas obtenidas. En comparación con los modelos cnn_v1, cnn_v2 y cnn_v3, se puede observar cómo abordar el desequilibrio entre clases utilizando únicamente las fases N2 y N3 permite mejorar la identificación de *spindles* (Se=74.67%, PPV=42.34%, F1-score=53.22%). En cambio, las métricas de rendimiento de los modelos CNN que han utilizado *Custom Loss Weights* y *Focal Loss* durante el entrenamiento no mejoran la detección de *spindles*.

Métricas de evaluación	Valor en %	
METODOLOGÍA	N2-N3 CNN anterior	N2-N3 CNN profunda
Accuracy	99.44	99.21
Recall / Sensitivity	33.52	74.67
Specificity	99.84	99.36
Precision	55.73	41.34
F1-score	41.86	53.22
MCC	42.97	55.21
RMSE	7.46	8.86

Tabla 4.4: Resultados del modelo cnn_v4 con CNN anterior y CNN profunda.

Métricas de evaluación	Valor en %		
METODOLOGÍA	N2-N3	N2-N3 + CLW	N2-N3 + FL
Acc	99.21	94.69	99.20
Se	74.67	98.31	72.89
Sp	99.36	94.66	99.36
PPV	41.34	9.98	40.70
F1-score	53.22	18.12	52.23
MCC	55.21	30.45	54.12
RMSE	8.86	23.05	8.93

Tabla 4.5: Métricas de evaluación de los modelos cnn_v4.

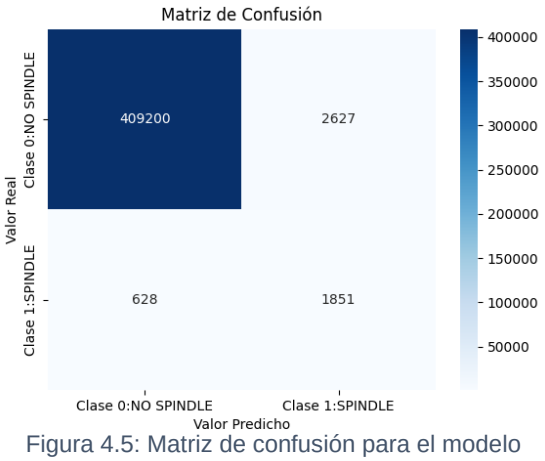

Figura 4.5: Matriz de confusión para el modelo cnn_v4 con señales N2-N3 y CNN profunda.

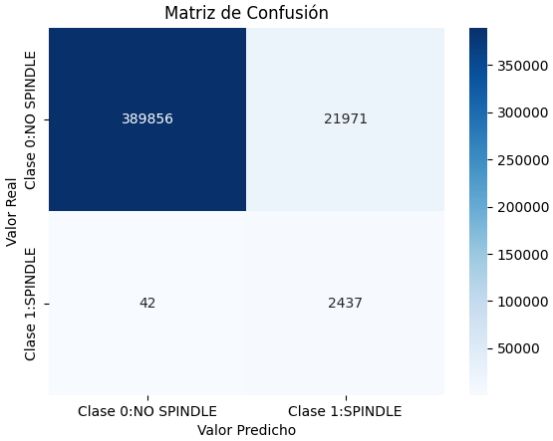

Figura 4.6: Matriz de confusión para el modelo cnn_v4 con señales N2-N3 + Custom Loss Weights y CNN profunda.

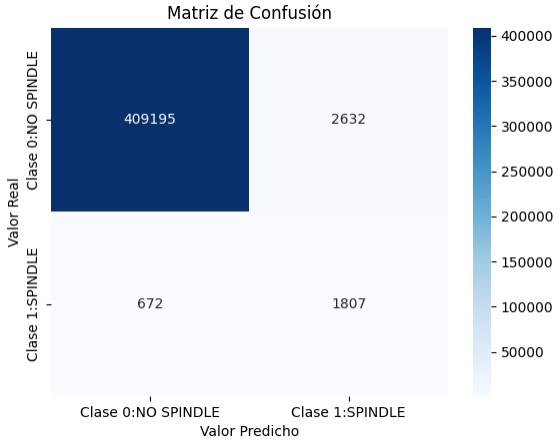

Figura 4.7: Matriz de confusión para el modelo cnn_v4 con señales N2-N3 + Focal Loss y CNN profunda.

4.2.5 MODELO CNN_V5

Para este modelo, se ha utilizado la CNN profunda modificando el parámetro *dropout_rate* de la red con los valores 0.00, 0.01, 0.1, 0.05 y 0.08. En este apartado se muestran tanto las matrices de confusión (Figuras 4.8-4.12) como las métricas obtenidas por cada uno de estos valores (Tabla 4.6).

Como se puede ver en los resultados de la Tabla 4.6, a medida que se aumenta el *dropout*, la sensibilidad del sistema disminuye, sin embargo, aumenta la precisión o valor predictivo positivo de ella. Es decir, poco a poco van disminuyendo los TP a consta que el número de FP y FN sean más similares. Se puede ver en la matriz de confusión para un *dropout* de 0.05 (Figura 4.10), se consigue reducir casi el 40 % de los FP aumentando ligeramente los FN.

Por lo tanto, gracias a esta nueva versión, se ha podido optimizar el valor del parámetro *dropout*. Además, dado que se busca maximizar los parámetros de PPV y F1-score sin

perder demasiada sensibilidad en el sistema, se ha seleccionado de todos los parámetros el de 0.05, el cual será utilizado en los posteriores modelos. Con este parámetro se obtienen los mejores valores de PPV (60.64%) y F1-score (61.12%) obtenidos hasta estas pruebas, aunque esto infiere directamente en una disminución de la sensibilidad (61.60%).

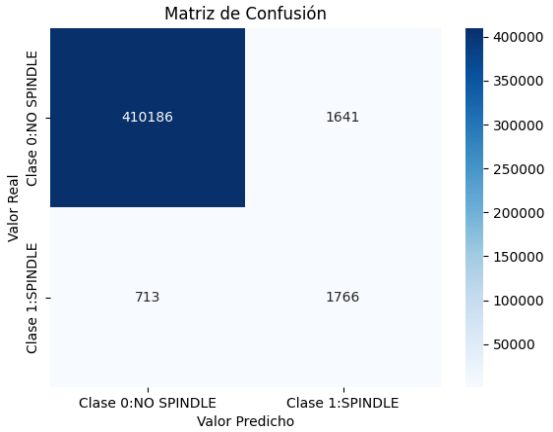

Figura 4.8: Matriz de confusión para el modelo cnn_v5 con *dropout*=0.00.

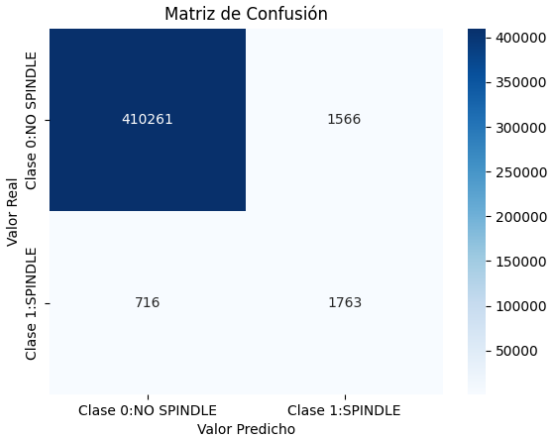

Figura 4.9: Matriz de confusión para el modelo cnn_v5 con *dropout*=0.01.

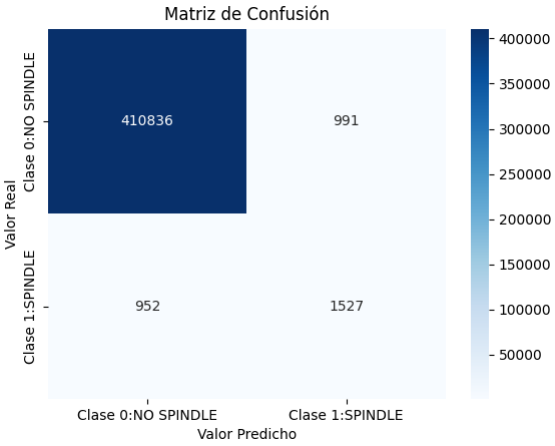

Figura 4.10: Matriz de confusión para el modelo cnn_v5 con *dropout*=0.05.

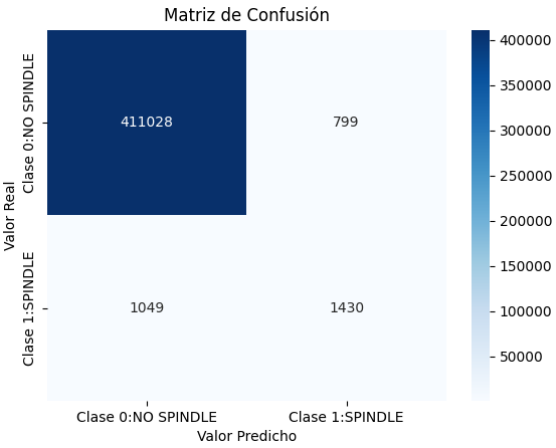

Figura 4.11: Matriz de confusión para el modelo cnn_v5 con *dropout*=0.08.

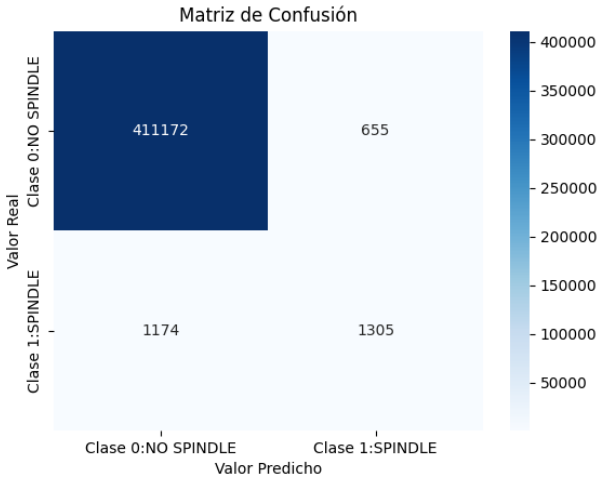

Figura 4.12: Matriz de confusión para el modelo cnn_v5 con *dropout*=0.1.

Métricas de evaluación	Valor en %				
DROPOUT	0.00	0.01	0.05	0.08	0.1
Acc	99.43	99.45	99.53	99.55	99.55
Se	71.24	71.12	61.60	57.68	52.64
Sp	99.60	99.62	99.76	99.81	99.84
PPV	51.83	42.96	60.64	64.15	66.58
F1-score	60.00	53.56	61.12	60.74	58.80
MCC	60.50	61.11	60.88	0.61	58.99
RMSE	7.54	7.42	6.85	6.68	6.64

Tabla 4.6: Métricas de evaluación del modelo cnn_v5 para los distintos valores de *dropout*.

4.2.6 MODELO CNN_V6

En este modelo se presentan los resultados aplicando post-procesado con las siguientes medidas para evitar el balanceado:

1. *Sliding window* (SW)
2. *Sliding sindow + mirror* (SW+M)
3. *Sliding window + gaussian noise* (SW+GN): dentro de este modelo se emplearon distintos parámetros de SNR para el ruido, siendo el más eficaz el de 0 dB. Por tanto, los resultados únicamente se muestran para una SNR = 0dB.

En las Figuras 4.13, 4.14 y 4.15 se pueden ver las matrices de confusión obtenidas con el modelo cnn_v6. Posteriormente, en la Tabla 4.7 se reflejan las métricas obtenidas. Estos modelos no superan en cuanto a precisión y F1-score a los modelos cnn_v5, lo cual sugiere que estas técnicas de balanceo de datos no funcionen del todo bien. Sin embargo, parece destacar que la técnica de aplicar ruido gaussiano para generar muestras sintéticas de *spindles* parece ser la mejor de ellas al obtener un mayor PPV (28.07% *vs.* 20.04% *vs.* 16.66%) y F1-score (43.02% *vs*. 33.15% *vs.* 28.45%).

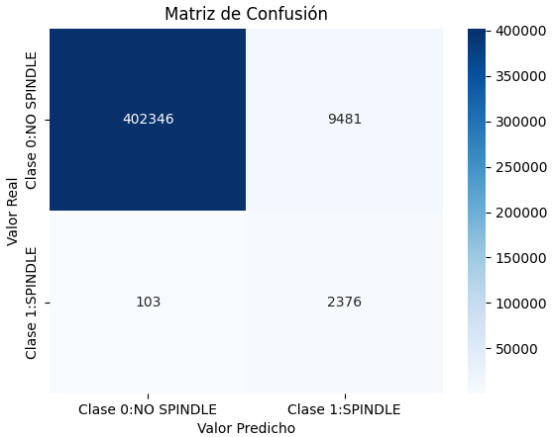

Figura 4.13: Matriz de confusión para el modelo cnn_v6 con *Sliding window*.

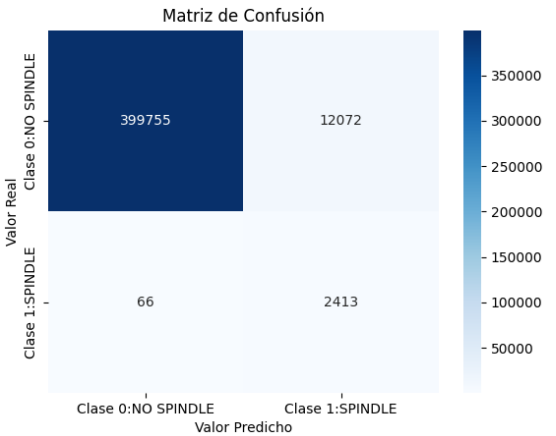

Figura 4.14: Matriz de confusión para el modelo cnn_v6 con *Sliding window + mirror*.

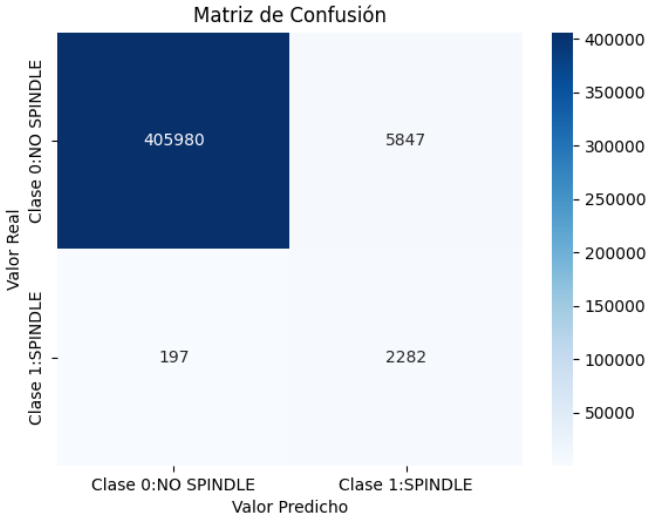

Figura 4.15: Matriz de confusión para el modelo cnn_v6 con *Sliding window + gaussian noise 0 dB*.

Métricas de evaluación	Valor en %		
METODOLOGÍA	SW	SW + M	SW +GN
Acc	97.69	97.08	98.54
Se	95.85	97.34	92.05
Sp	97.70	97.07	98.58
PPV	20.04	16.66	28.07
F1-score	33.15	28.45	43.02
MCC	43.27	39.64	50.40
RMSE	15.21	17.12	12.08

Tabla 4.7: Métricas de evaluación de los modelos cnn_v6.

4.2.7 MODELO CNN_V7

En este modelo se presentan los resultados aplicando post-procesado con las siguientes medidas para evitar el desequilibrio entre los datos:

1. *Gaussian noise* 0 dB (GN)
2. *Sliding window* + 2 * *Gaussian noise* 0 dB (SW+2*GN)

En las Figuras 4.16 y 4.17 se puedes ver las matrices de confusión obtenidas con el modelo cnn_v7. Posteriormente, en la Tabla 4.8 se reflejan las métricas obtenidas, muchas de ellas, a partir de dichas matrices de confusión. En comparación con los modelos cnn_v6, se puede ver que generando únicamente muestras sintéticas con ruido gaussiano (GN), se obtienen mejores resultados en cuanto a PPV (54.84% *vs.* 28.07%) y F1-score (60.96% *vs*. 43.02%), aunque la sensibilidad del sistema disminuye Se (68.62% *vs*. 92.05%). Además, los resultados de este modelo GN, se pueden comparar con los obtenidos en cnn_v5 ya que son bastante similares (PPV=54.84% *vs.* 60.64%, F1-score=60.96% *vs.* 61.12%, Se=68.62% *vs.* 61.60%).

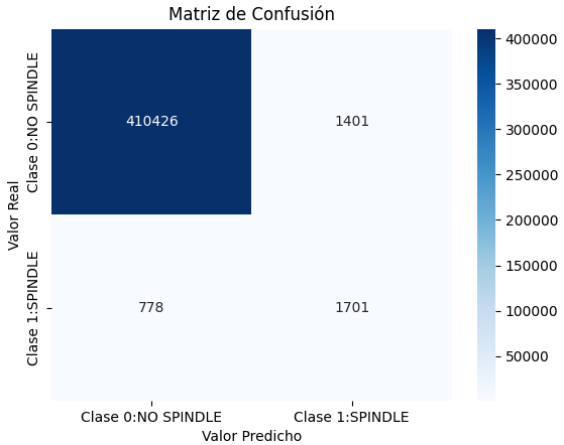

Figura 4.16: Matriz de confusión para el modelo cnn_v7 con *gaussian noise*.

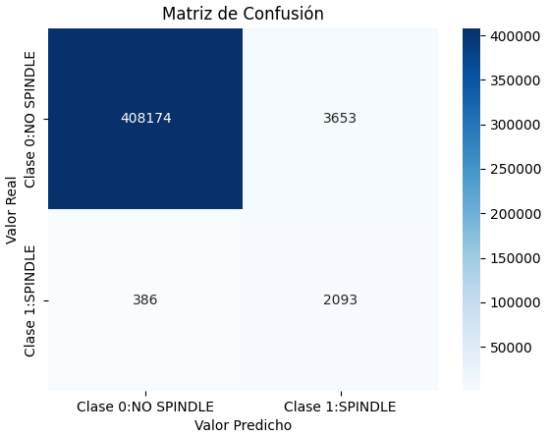

Figura 4.17: Matriz de confusión para el modelo cnn_v7 con *sliding window + 2*gaussian noise*.

Métricas de evaluación	Valor en %	
METODOLOGÍA	GN	SW + 2* GN
Accuracy	99.47	99.03
Recall / Sensitivity	68.62	84.43
Specificity	99.66	99.11
Precision	54.84	36.43
F1-score	60.96	50.90
MCC	61.08	55.09
RMSE	7.25	9.87

Tabla 4.8: Métricas de evaluación de los modelos cnn_v7.

Por último, se muestra un ejemplo de predicción para el modelo cnn_v7 cuando se emplea únicamente ruido gaussiano para generar muestras sintéticas de *spindles* (Figura 4.18). En el resto de los modelos no se muestra este ejemplo ya que las imágenes obtenidas son similares.

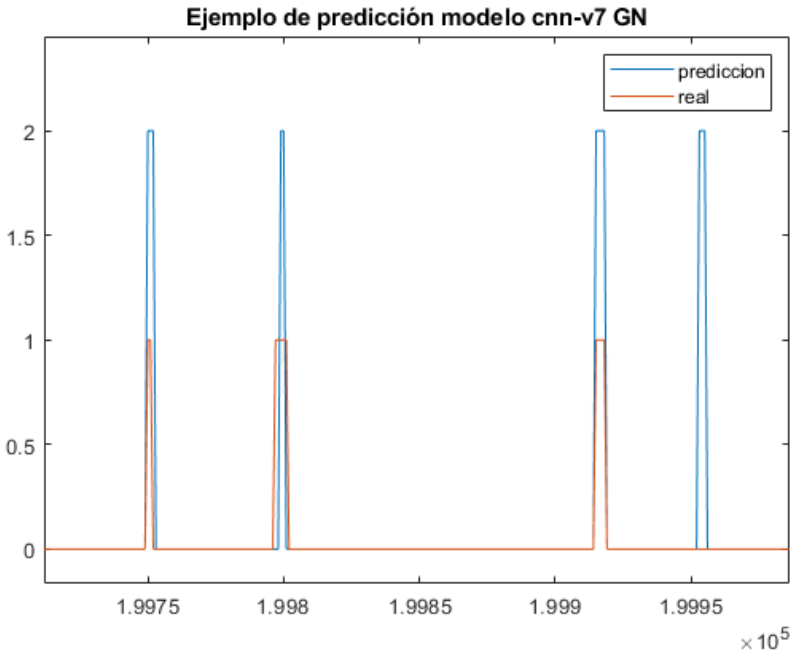

Figura 4.18: Ejemplo de predicción para el modelo cnn_v7 GN.

4.3 RESULTADOS DEL MODELO U_NET

En esta nueva versión, se ha probado la arquitectura de red U-Net propuesta en el artículo de You et al. (2021). Como ya se comentó en apartados anteriores, para el desarrollo de este nuevo modelo se empleó distinta probabilidad de predicción en la función *softmax* (valores de 0.5 y 0.6). Sin embargo, sólo se presentan los resultados para el umbral de 0.5 debido a sus mejores resultados. Además, se ha utilizado un post-procesado diferente, que considera *spindle* si el número de muestras es superior a 200 (0.4 segundos).

En primer lugar, se muestra en la Figura 4.19 la matriz de confusión obtenida con la metodología muestra a muestra y en la Tabla 4.9 los valores de las métricas obtenidas. Aunque los resultados no sean directamente comparables, se observa que se ha obtenido un rendimiento ligeramente mejor que en las arquitecturas CNN (F1-score=63.48% y MCC=64.88%).

A continuación, se muestran los resultados obtenidos por evento. Tras calcular el IoU, se han obtenido los siguientes parámetros:

1. TP= 621
2. FP= 466
3. FN= 106

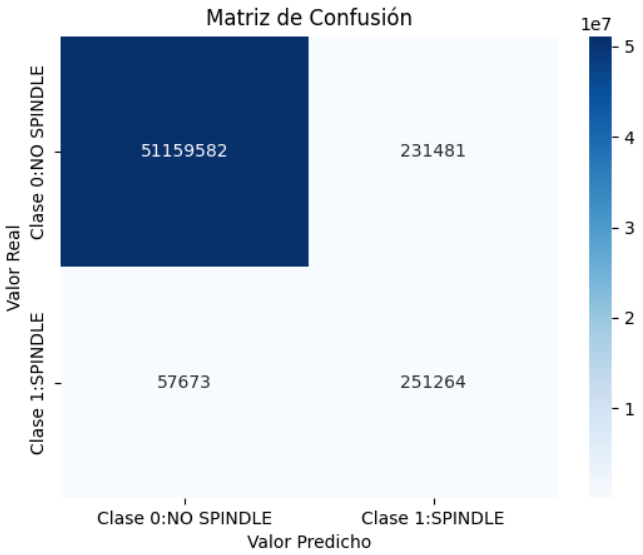

Figura 4.19: Matriz de confusión del modelo U_net para los resultados muestra a muestra.

Métricas de evaluación	Valor en %
Acc	99.44
Se	81.33
Sp	99.55
PPV	52.05
F1-score	63.48
MCC	64.88
RMSE	7.48

Tabla 4.9: Métricas para el modelo U_net empleando los resultados muestra a muestra.

Métricas de evaluación	Valor en %
Se	85.42
PPV	57.13
F1-score	68.48

Tabla 4.10: Métricas para el modelo U_net empleando los resultados por evento.

En la Tabla 4.10 se pueden ver las métricas de rendimiento obtenidas con estos parámetros. En este caso, como estas medidas son directamente comparables a los resultados obtenidos con CNN, se puede decir que los obtenidos con la U-Net son mejores tanto en Se (85.42% *vs.* 68.62%) como en PPV (57.13% *vs.* 54.84%) y F1-score (68.48% *vs.* 60.96%).

Posteriormente, se han obtenido las métricas relacionadas con los errores en duración y en comienzo de los *spindles*. Estos errores han sido traducidos en términos de tiempo y representados en histogramas tal y como se puede ver en las Figuras 4.20 y 4.21. Además de representar los histogramas junto con los intervalos de confianza, se ha calculado la media e intervalo de confianza 965% de ambos errores (Tabla 4.11).

Además, por cada sujeto perteneciente al conjunto de test, se muestra en la Tabla 4.12 el número de *spindles* detectados frente a los reales, así como su duración promedio. Por lo general, se puede ver que hay una gran concordancia entre los *spindles* detectados y los reales en todos los sujetos excepto en el último, donde hay una gran diferencia entre el número de spinles detectados (44) y los reales (101).

Por último, en la Figura 4.22 se puede ver un ejemplo de predicción hecha con este modelo en la que se representa sobre la señal de EEG, las etiquetas reales junto con las predicciones.

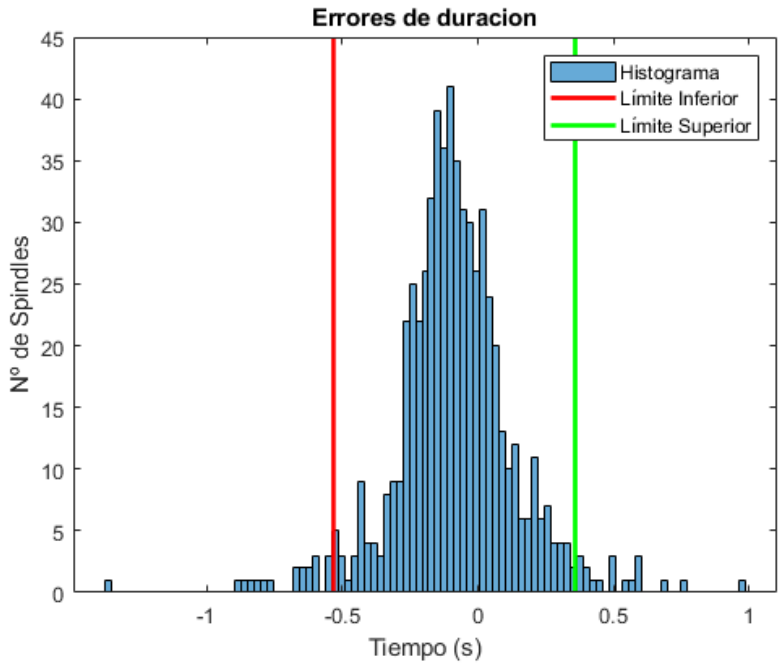

Figura 4.20: Histograma que representa los errores en duración obtenidos en el modelo U_net.

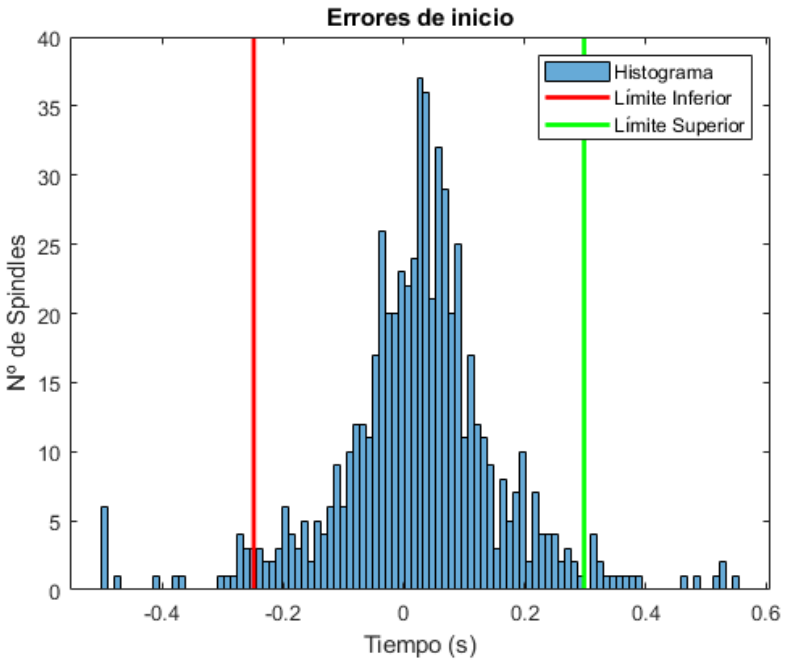

Figura 4.21: Histograma que representa los errores en inicio obtenidos en el modelo U_net.

Métricas de evaluación	Errores en duración	Errores en comienzo
Media	0.0876 s	0.025 s
Intervalo confianza 95%	0.890641 s	0.546609 s

Tabla 4.11: Media y desviación típica para los histogramas de los errores del modelo U_net.

ID SUJETO	Número spindles detectados	Número spindles reales	Duración promedio spindles detectados	Duración promedio spindles reales
COG006	201	214	0.993527 s	0.8469752 s
COG007	338	368	0.9776686 s	0.8891952663 s
COG008	35	44	0.8224 s	0.804742856 s
COG009	44	101	0.7925 s	0.928318181 s

Tabla 4.12: Características de predicción por cada sujeto para el modelo U_net.

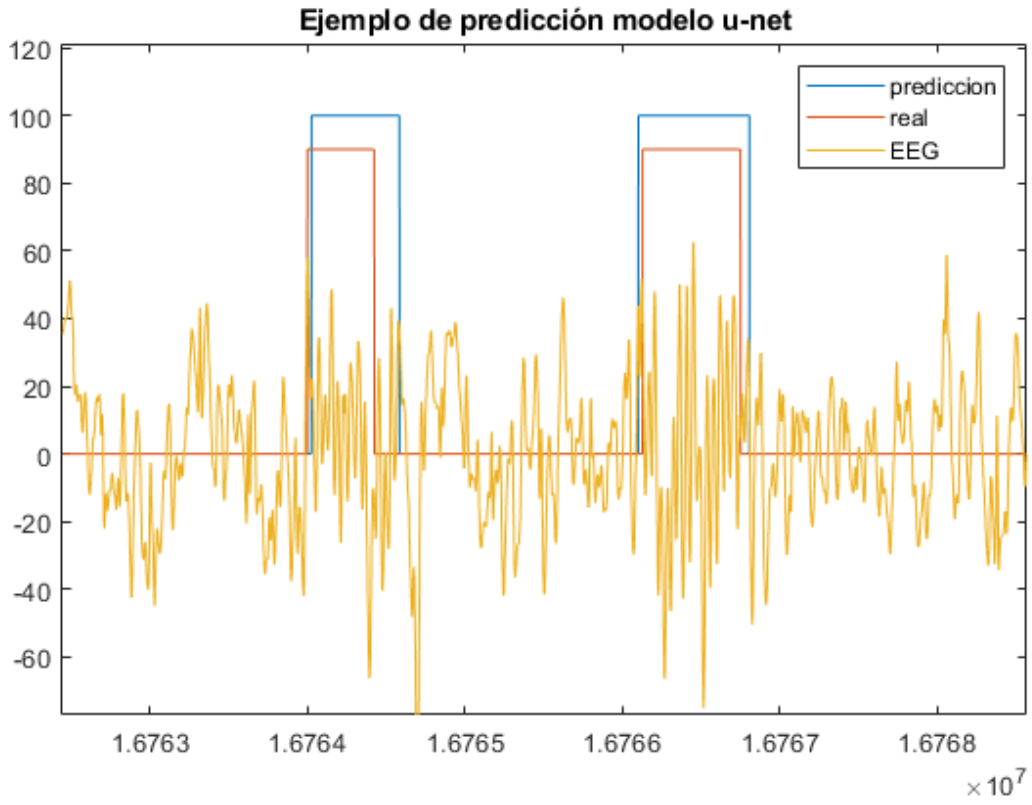

Figura 4.22: Ejemplo de predicción para el modelo U_net.

CAPÍTULO 5: DISCUSIÓN

En este estudio se presentan varios métodos de detección de *spindles* basados en arquitecturas *deep learning* empleando para ello registros EEG de niños pedriático con AOS. Los resultados obtenidos, sugieren que es posible conseguir un clasificador que presente buenas prestaciones al respecto. La forma de proceder en este estudio ha estado guiada por el TFG de la misma autora (Pacho Velasco, 2022) y los artículos de You et al. (2021) y Wei, Ventura, Ryan, et al. (2022).

El objetivo principal de esta detección se centra en ayudar a los expertos para el marcaje de *spindles,* ya que es un trabajo tedioso que implica una gran inversión de tiempo. Gracias a ello y conociendo la densidad de *spindles* que presenta una persona, se pueden diagnosticar o anticipar enfermedades relacionadas con el proceso cognitivo para poder tratarlo.

En el capítulo anterior se trató de exponer los resultados obtenidos tras la evaluación de varios modelos, sin embargo, es en este capítulo donde se pretenden analizar e interpretar además de comparar unos modelos con otros. Asimismo, se exponen los resultados del TFG y los artículos a replicar, además de una comparación donde se exponen las diferencias que hay con estos modelos. Para finalizar, se exponen las limitaciones presentes en este estudio.

5.1 DISCUSIÓN DE LOS RESULTADOS OBTENIDOS

En esta discusión se pretenden discutir los resultados obtenidos en este estudio que fueron mostrados en el capítulo anterior. Para ello, en un primer lugar se hablará de las versiones de modelos CNN y posteriormente, se comparan esos resultados con los obtenidos empleando una U-Net.

5.1.1 MODELOS CNN

Antes de discutir los resultados obtenidos, hay que recordar que los modelos **cnn_v1**, **cnn_v2** y **cnn_v3**, tienen una distribución de los sujetos en los conjuntos de entrenamiento, validación y test diferente al resto de los modelos, lo cual invalida en cierto modo una comparación directa con el resto de las versiones. Además, la red utilizada también es diferente al resto. Esto es debido a que inicialmente, se trató de replicar el artículo Wei, Ventura, Ryan, et al. (2022) y una vez adquiridos ciertos resultados, mejorar la red y emplear la misma repartición de sujetos que en el TFG (Pacho Velasco, 2022)

para poder estudiar las diferencias entre aplicar algoritmos de inteligencia artificial basados en *machine learning* y *deep learning*.

En cuanto a estos tres modelos, se puede ver que el que mejores prestaciones obtiene de los tres es el primero (**cnn_v1**). En el resto de ellos se emplea la técnica *RandomUnderSampling* (AT et al., 2016) para tratar de paliar los problemas en cuanto al desequilibrio entre los datos aplicándose únicamente al conjunto de entrenamiento (**cnn_v2**) y el de entrenamiento y validación (**cnn_v3**). Sin embargo, no se consigue mejorar la situación y se obtienen clasificadores con muy baja precisión.

Por lo tanto, con la técnica de balanceo de datos *RandomUnderSample* no se consigue mejorar la situación de desequilibrio entre datos y la clasificación de *spindles* se realiza mucho peor. El motivo de ello reside en que al eliminar segmentos "no *spindle*" al azar, el algoritmo elimina de toda la señal completa de EEG elementos que pueden aportar información relevante a la hora de realizar la clasificación. Además, al trabajar con toda la señal EEG el algoritmo elimina más del 70% de estos elementos con el objetivo de que tanto las muestras de "*spindles"* como "no *spindles*" sean las mismas. Se puede ver claramente cómo el número de falsos positivos crece en gran medida para los modelos **cnn_v2** y **cnn_v3**. Por ello, en el resto de los modelos, no se emplea esta técnica como solución al desequilibrio entre clases.

A partir del modelo **cnn_v4**, además de modificar la repartición de los sujetos entre los diferentes subgrupos y emplear una CNN profunda, se intentan aplicar diferentes técnicas de balanceo entre datos comenzando con el recorte de la señal EEG en las fases N2 y N3 del sueño, eliminando así una gran cantidad de datos sobrantes pertenecientes a la clase negativa. Dentro de las tres técnicas que se han utilizado en esta versión, se puede ver que los resultados obtenidos empleando N2-N3 y N2-N3 junto con *Focal Loss* (*API TensorFlow*, 2023), no parecen una gran diferencia entre ellos, por lo que la función de pérdidas no ha vuelto a ser utilizada como solución. Sin embargo, estos resultados difieren bastante a lo obtenido con la técnica de *Custom Loss Weights* (CLW) (*API TensorFlow*, 2023).

El CLW aumenta la Se del clasificador, consiguiendo detectar prácticamente todos los *spindles* presentes en la señal. Sin embargo, vemos que la PPV obtenida decae considerablemente en comparación con los otros dos modelos. Esto se ve reflejado en el aumento de FP en la matriz de confusión. Debido a que la medida de F1-score es bastante más elevada para N2-N3, se ha continuado en posteriores versiones con esa idea, descartando el uso de CLW.

Respecto a la nueva versión **cnn_v5**, se ha tratado de mejorar la red modificando el parámetro *dropout_rate* como técnica de regularización para evitar un posible *overfitting* del modelo. Como ya se comentó previamente, se ha seleccionado un *dropout* de 0.05 ya que se reducen en gran medida los FP a pesar de que los TP bajen. Con 0.05, las precisiones obtenidas son muy altas, a pesar de que la sensibilidad del sistema disminuye en respecto a un *dropout* menor. En comparación con el modelo anterior, con **cnn_v5** se consigue un PPV (60.64%) y F1-score (61.12%) superior.

Con el siguiente modelo **cnn_v6**, se consigue reducir el número de FP aplicando una técnica de post-procesado a la predicción del clasificador. Además, se generan *spindles*

sintéticos para evitar el desequilibrio entre los datos. Ante todas las soluciones propuestas en esta versión, optar por añadir a la base de datos muestras sintéticas con ruido gaussiano de SNR 0 dB parece una buena opción ya que las métricas de F1-score y precisión aumentan en comparación con el resto de las pruebas. Por contrapartida, la sensibilidad del algoritmo disminuye al 92.05%, aunque sigue siendo un valor elevado. Es por ello, que en la última versión con CNN (**cnn_v7**) se ha probado el algoritmo aplicando diversas técnicas con ruido gaussiano.

La última versión empleando una red CNN profunda corresponde a **cnn_v7**. En ella se prueban diferentes técnicas para generar *spindles* sintéticos con la idea de emplear ruido gaussiano para ello. Respecto a los resultados obtenidos en la *Tabla 16*, llama la atención en primer lugar que generando muestras de ruido gaussiano se consigue una precisión y F1-score mucho mayores a lo que se obtenía en **cnn_v4** sin generar muestras sintéticas. Sin embargo, esto provoca que la sensibilidad del sistema caiga en comparación disminuyendo el número de TP clasificados. Por otra parte, aplicando una generación de muestras sintéticas con *sliding window* y ruido gaussiano, ocurre el efecto contrario. Con ello se consigue mejorar la sensibilidad con la pérdida de precisión y F1-score.

Por otro lado, merece la pena destacar la semejanza de resultados que se obtienen con el modelo **cnn_v7** aplicando GN y **cnn_v5** con un *dropout* de 0.05 (PPV=54.84% *vs.* 60.64%, F1-score=60.96% *vs.* 61.12%, Se=68.62% *vs.* 61.60%). Con ello, se puede decir que al añadir muestras sintéticas de *spindles*, lo que se consigue es aumentar la Se del sistema a cambio de experimentar una leve disminución el PPV.

Como resumen de todos estos algoritmos basados en CNN, se considera complicado seleccionar cuál de todos ellos proporciona un mejor rendimiento. Si lo que se busca es que el algoritmo sea capaz de detectar la mayoría de *spindles* presentes en la señal y así tener una sensibilidad alta (a pesar de aumentar el número de FP), quizá nos interese **cnn_v6** cuando se aplica SW+GN 0 dB o **cnn_v7** con SW+2*GN. Sin embargo, si lo que buscamos es un alto F1-score y una precisión alta, aunque esto implique en cierta forma perder sensibilidad, puede que nos interese aplicar el algoritmo **cnn_v4** aplicando N2-N3 o **cnn_v7** cuando se emplea únicamente GN. En estudios futuros, se debería ver con cuál de estas aproximaciones el número de *spindles* detectados presenta una mayor correlación con los parámetros cognitivos en niños con AOS.

De todas formas, siempre hay que tener en cuenta que puede darse el caso en el que un FP sea en realidad un TP que no haya sido marcado por los expertos. En la Figura 5.1 se ha representado un ejemplo correspondiente a un FP obtenido en el modelo **cnn_v4** para N2-N3. En esta imagen se representa una época del EEG clasificada como falso positivo y debajo de ella su correspondiente espectrograma. Como ya se ha comentado en anteriores capítulos, un *spindle* en niños se localiza entre 8-15 Hz y, tal y como se puede ver en el espectrograma, el segmento bajo estudio oscila mayoritariamente entre esas frecuencias. La conclusión de todo esto reside en que puede que haya falsos positivos que el algoritmo detecte correctamente, pero los expertos hayan clasificado como no *spindle*. Estas marcas servirían de apoyo a expertos para comprobar si efectivamente lo que el algoritmo detecta como *spindle* lo es o no.

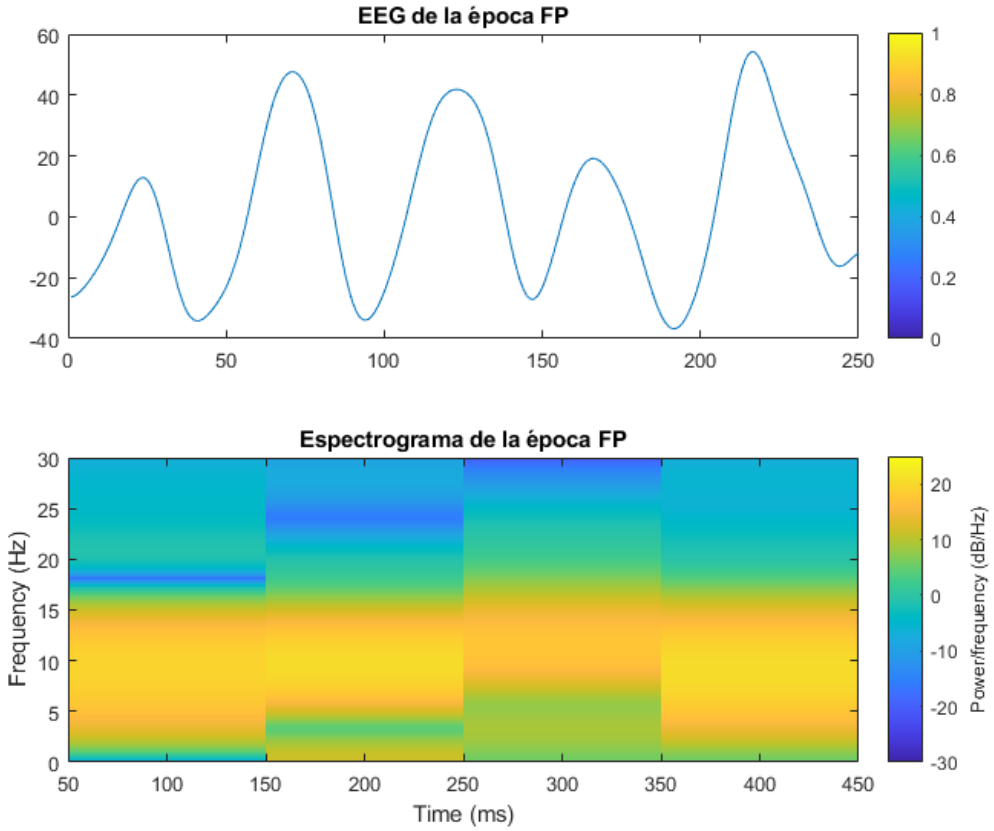

Figura 5.1: Señal EEG y espectrograma de un falso positivo.

5.1.2 ARQUITECTURA U-NET

Con respecto a la U-Net, se van a analizar principalmente las métricas de rendimiento por evento (Tabla 19), ya que es la forma más común de evaluar la detección de spindles, así como los errores en duración (Figura 4.20) y comienzo (Figura 4.21) de los *spindles*.

En cuanto a las métricas por evento, los resultados son muy satisfactorios. El F1-score de 68.48% es el mayor valor obtenido hasta el momento en comparación con el resto de los modelos. Además, se obtiene un PPV del 57.13% junto con un valor de Se bastante elevado, del 85.42%. En los anteriores modelos CNN, nunca se había dado la situación de obtener un F1-score y PPV tan altos con una Se del algoritmo tan elevada.

Por otro lado, al observar los histogramas de los errores en duración y comienzo, se puede deducir que presentan una distribución aproximadamente gaussiana, donde la mayor parte de las muestras se concentran en la media, la cual es muy próxima a 0 segundos en ambos casos. Gracias a los intervalos de confianza se puede ver que la gran mayoría de los errores en duración no superan 0.5 segundos y los de inicio 0.2 segundos.

Para finalizar, en la *Tabla 21* se muestra una tabla comparativa en la que se puede ver que tanto el número de *spindles* detectados como la duración media de ellos predicha se acerca mucho a la realidad. Eso podría ser de gran utilidad para evaluar la asociación entre el número de *spindles* y su duración con los parámetros neurocognitivos en niños con AOS.

5.2 COMPARATIVA DE RESULTADOS

En este apartado se pretenden comparar los modelos generados en este TFM con el modelo implementado en el artículo que se quiere replicar. Para ello, y antes de comenzar con la comparativa, en la Tabla 5.1 se presentan brevemente los resultados alcanzados en las publicaciones (Wei, Ventura, Ryan, et al., 2022; You et al., 2021), así como los obtenidos en el TFG (Pacho Velasco, 2022). Por su parte, en la Tabla 5.2 se muestran los resultados alcanzados por este TFM a modo de resumen.

En la Tabla 5.2 hay que tener en cuenta que los resultados de **cnn_v4** se corresponden a la CNN profunda N2-N3, los resultados de **cnn_v5** se corren a los obtenidos con un *dropout* de 0.05, los resultados de **cnn_v6** se corresponden a la metodología SW+GN, los resultados de **cnn_v7** se corresponden a la metodología GN y los resultados de **U_net** corresponden a los de por evento (*by event*).

La estructura de este apartado va a consistir de tres partes. Por un lado, se comentarán los resultados obtenidos con los modelos empleando una CNN convencional con el artículo de Wei, Ventura, Ryan, et al. (2022), el cuál ha sido objeto a seguir al comienzo del estudio. A continuación, se compararán los resultados de la U-Net con el artículo de You et al. (2021), ya que ha sido el modelo a seguir para este otro tipo de algoritmos. Por último, se compararán los resultados obtenidos en el TFG de la misma autora con los obtenidos en este nuevo proyecto.

PUBLICACIÓN	SUJETOS	SEÑALES	MODELO	Acc	Se	Sp	PPV	F1	MCC	RMSE
Wei, Ventura, Ryan, et al. (2022)	141 recién nacidos (4 meses)	C3 y C4. Etapa N2	CNN + RNN		91.90	96.70	95.20	93.50	88.70	
You et al. (2021)	19 PSG de jóvenes	C3	U-Net		83.40		76.80	79.10		
Victoria et al. (2022)	Misma base de datos que el TFM	C3	Random Forest	95.99	95.76	95.99	12.56	22.21	34.00	17.00

Tabla 5.1: Métricas de evaluación obtenidas en las publicaciones (Wei, Ventura, Ryan, et al., 2022), (You et al., 2021) y (Pacho Velasco, 2022).

Métricas de evaluación	Valor en %							
MODELO	cnn_v1	cnn_v2	cnn_v3	cnn_v4	cnn_v5	cnn_v6	cnn_v7	U_net
Acc	99.80	81.79	92.02	99.21	99.53	98.54	99.47	
Se	30.48	90.61	84.97	74.67	61.60	92.05	68.62	85.42
Sp	99.91	81.78	92.03	99.36	99.76	98.58	99.66	
PPV	33.11	0.75	1.59	41.34	60.64	28.07	54.84	57.13
F1-score	31.74	1.49	3.12	53.22	61.12	43.02	60.96	68.48
MCC	31.66	7.27	10.97	55.21	60.88	50.40	61.08	
RMSE	4.45	42.68	28.25	8.86	6.85	12.08	7.25	

Tabla 5.2: Resumen de las métricas de evaluación obtenidas en este TFM.

En primer lugar, el estudio realizado por Wei, Ventura, Ryan, et al. (2022) ha obtenido unos parámetros de evaluación elevados (ver Tabla 5.1). Llama la atención que estos resultados presentan una sensibilidad, precisión y F1-score más altos a los que se han conseguido con los modelos CNN (**cnn_v1** a **cnn_v7**) de este proyecto (ver Tabla 5.2). Hay varios motivos por los que se obtienen estas diferencias:

- La división de sujetos en los distintos subconjuntos de entrenamiento, validación y test se ha realizado de forma diferente.
- Wei, Ventura, Ryan, et al. (2022) emplean las señales C3 y C4 para la clasificación de *spindles*, mientras que en este TFM sólo se utiliza la señal C3.
- Para el entrenamiento y validación del modelo la señal utilizada sólo presenta la etapa N2 del sueño mucho más recortada que la que se utiliza en este proyecto.
- El algoritmo empleado no sólo está formado por una CNN, sino que está precedido por una RNN antes de obtener la predicción final.
- Las señales EEG pertenecen a recién nacidos en vez de niños entre 6 y 9 años, como se realiza en este trabajo. Esto hace que las características del EEG sean diferente para ambos grupos de edad. En cuanto al rango de frecuencia de los *spindles*, en recién nacidos está entre 10-16 Hz, mientas que para niños de 6 a 9 años entre 8-15 Hz. Por otro lado, en cuando a la densidad de *spindles*, esta es relativamente baja en recién nacidos en comparación con niños de 6 a 9 años, donde la densidad de *spindles* tiende a aumentar.
- En este TFM se ha trabajado con señales pertenecientes a niños con sospecha de AOS, por lo que esto infiere en una disminución significativa del número de *spindles* presentes en cada EEG en comparación con niños sanos. Además, las características del EEG en presencia de AOS se ven modificadas, pudiendo llegar a confundir unos eventos conocidos como *arousals* con *spindles*.

Todas estas diferencias influyen en la diferencia de resultados con respecto a este trabajo. Sin embargo, se puede decir que las métricas conseguidas bajo este estudio también son muy buenas, siendo capaces de obtener sensibilidades superiores al 91.90% a pesar de tener valores de PPV y F1-score menores. Además, este algoritmo presenta la ventaja de que no sólo se pueden detectar estas oscilaciones en la etapa N2 del sueño, sino también en N3.

En segundo lugar, se van a comparar los algoritmos que emplean una red basada en U-Net (You et al., 2021). Observando las Tablas 5.1 y 5.2, se puede ver que el algoritmo elaborado en este proyecto (**U_net**), tiene una mayor sensibilidad que el presentado en el artículo. Sin embargo, los valores de PPV y F1-score son algo más bajos. De todas maneras, en comparación con los resultados anteriores, los obtenidos con esta arquitectura no difieren tanto. Por otro lado, si observamos los errores en duración que se expone en You et al. (2021), presentan una mayor desviación que los obtenidos para este TFM. Por último, destacar las diferencias entre este artículo y el TFM:

- You et al. (2021) entrena al sistema empleando únicamente la etapa N2 del sueño del EEG de sus pacientes.
- Se realiza una normalización de las señales en el preprocesado de datos.
- Las señales EEG pertenecen a jóvenes sanos, mientras que en este TFM se emplean señales de niños con sospecha de AOS.
- Las señales EEG han sido remuestreadas a 256 Hz, mientras que en este estudio la frecuencia de muestreo es de 500 Hz.

- Se emplean técnicas de *data augmentation* para evitar el desequilibrio entre los datos, mientras que en el modelo U_net propuesto en este proyecto, no se emplea ningún tipo de técnica de balanceo de datos.

Por último, se comparan todos los resultados con los obtenidos en el anterior TFG (ver Tabla 5.1). Las evaluaciones de los modelos obtenidos en este TFM pueden ser directamente confrontados con los del TFG puesto que se han empleado las mismas señales en ambos proyectos, además se han dividido los sujetos en los mismos subgrupos y se ha entrenado a los diferentes sistemas finalmente con las etapas N2 y N3 del sueño. Al detenerse un poco sobre los resultados se puede decir que fácilmente se han superado los valores del TFG. El modelo **cnn_v6** presenta mejores prestaciones en todas las métricas que con lo que se obtuvo en su día implementando un modelo de *Random Forest* como clasificador de *spindles* para el TFG. En términos generales, uno de los objetivos de este TFM era superar la precisión obtenida por el previo TFG y se puede ver que efectivamente se ha conseguido. Hay casos en los que se ha logrado conseguir un PPV del 60.64% y F1-score del 61.12%. Sin embargo, en la mayoría de los casos esto ha inferido en una disminución de la sensibilidad del sistema. Sin duda, el mejor valor de precisión y F1-score conseguido, manteniendo un valor de sensibilidad elevado, se ha obtenido en el modelo **U_net** (85.42% de Se, 57.13% de PPV y 68.48% de F1-score). Con ello se puede concluir que los resultados de los modelos DL desarrollados en este TFM han sido superiores a los basados en ML por el TFG.

En resumen, se puede decir que el objetivo principal de toda esta investigación centrada en la detección automática de *spindles* del sueño aplicando técnicas de *deep learning* se ha conseguido de manera satisfactoria y podría ser utilizada en el futuro como ayuda a expertos en su trabajo de detección de *spindles*.

5.3 LIMITACIONES

Antes de finalizar este capítulo se detallan varias limitaciones que han estado presentes a lo largo del desarrollo del TFM.

La primera de las limitaciones, y la cual se puede considerar más importante, ha sido el hecho de trabajar con una base de datos en la que únicamente formaban parte de ella 9 sujetos. Esto implica que el alcance de los resultados obtenidos está limitado a un número reducido de sujetos. No obstante, el hecho de que los artículos apliquen sus algoritmos con una amplia base de datos sugiere que la detección automática de *spindles* pueda ser continuada en estudios futuros con poblaciones más amplias y con diferentes características. Sin embargo, esto ha permitido poder comparar los resultados directamente con los obtenidos en el anterior TFG.

Otras de las limitaciones encontradas es el desbalanceo entre clases “*spindle*” y “no *spindle*”, presentando un mayor de segmentos esta última. Esta anterior limitación es aún mayor ya que se ha trabajado con niños con sospecha de AOS, y la AOS conlleva a una reducción de la densidad de *spindles* a lo largo del sueño en las fases N2 y N3 y, como consecuencia, puede inferir en el número de muestras pertenecientes a esa clase. Para tratar de minimizar este problema se han propuesto varias soluciones de *undersampling* y

oversampling en los distintos modelos: *Custom Loss Weights*, *Focal Loss*, *Sliding Window*, ruido gaussiano, *mirror,* N2-N3, incluyendo combinaciones entre estas soluciones.

No obstante, el problema no se ha logrado reducir suficientemente y, como consecuencia, los clasificadores entrenados generan un gran número de falsos positivos con relación al número de verdaderos positivos.

Otra limitación está ligada a que en este TFM se han desarrollado algoritmos basados en CNN y U-Net, sin embargo, no se ha probado a combinar otras arquitecturas dentro de la rama de DL. Incluso se podría modificar la estructura de la U-Net pudiendo obtener mejores resultados, lo cual no se logró dada la limitación temporal.

La última limitación podría estar relacionada con el hecho de que el modelo formado únicamente por las fases N2 y N3 del sueño está limitado por el algoritmo de *deep learning* desarrollado en el TFG de Eva Calvo (Calvo Merino, 2020). Sin embargo, el rendimiento de este algoritmo es superior a la concordancia inter-observador en la detección de fases del sueño y, por lo tanto, no comprometen las conclusiones obtenidas en este trabajo.

Por último, las arquitecturas de *deep learning* son vistas como una caja negra. Es decir, no se puede saber exactamente en qué características o patrones se ha fijado para detectar *spindles*, lo que dificulta el análisis de aquellos casos donde no se han detectado *spindles* o la detección es incorrecta.

CAPÍTULO 6: CONCLUSIONES Y LÍNEAS FUTURAS

Durante este TFM se ha trabajado con señales EEG en niños con el objetivo de elaborar un algoritmo capaz de detectar *spindles* del sueño aplicando técnicas de *deep learning* continuando con la investigación del anterior TFG. Después de explicar la metodología utilizada, los resultados obtenidos y la discusión sobre ellos, se finaliza este informe exponiendo las contribuciones realizadas al campo de investigación, las conclusiones de todo el proyecto y posibles líneas futuras que puedan continuar con el estudio.

6.1 CONTRIBUCIONES AL CAMPO DE INVESTIGACIÓN

Al finalizar este estudio, se deben reflejar las contribuciones que dicho trabajo aporta al campo de investigación, las cuales se enumeran a continuación:

1. Desarrollo de un algoritmo capaz de detectar *spindles* del sueño en señales procedentes del EEG en niños empleando técnicas de DL. Se han explorado varios tipos de redes y arquitecturas (CNN y U-Net), las cuales son de actual interés y han permitido caracterizar estas oscilaciones en los distintos modelos.
2. Detección del comienzo y duración de los *spindles* en niños con sospecha de AOS mediante una U-Net. Aunque la U-Net se habían empleado en población adulta, este es el primer trabajo que las utiliza en sujetos pedriáticos.
3. Comparación directa de los rendimientos que se obtienen con redes de ML (*Random forest*) y DL (CNN, U-Net) en la detección de *spindles*. Gracias al TFG previo, se ha podido hacer esta comparación directa, ya que tanto las señales EEG como la repartición de sujetos ha sido la misma para ambos estudios.

6.2 CONCLUSIONES

A continuación, se exponen las diferentes conclusiones extraídas en base al trabajo desarrollado a lo largo de este TFM:

1. El desarrollo de ese TFM como línea futura del previo TFG, afirma la gran capacidad que presentan las técnicas de inteligencia artificial para desarrollar algoritmos clasificadores que detecten de manera automática *spindles* del sueño

en niños entre 6 y 9 años. Gracias a ellos se consigue facilitar la labor de su marcaje a expertos y así poder diagnosticar, en caso de detectar irregularidades en la densidad de *spindles* a lo largo del sueño, enfermedades relacionadas con el proceso cognitivo del paciente más rápidamente. Además, las herramientas como DL son de gran utilidad para la detección automática de *spindles*, automatizando así el proceso y reduciendo los costes en comparación con su clasificación de forma manual.

2. Un solo canal de la señal EEG (C3) contiene información relevante y adecuada para caracterizar las oscilaciones producidas en el EEG por los *spindles* durante el sueño de los niños con sospecha de AOS.
3. Las técnicas de DL (CNN y U-Net) consiguen un rendimiento superior a las metodologías de ML (*Random forest*) en la clasificación automática de *spindles* aumentando su PPV y F1-score considerablemente.
4. Utilizar las señales de EEG correspondientes a las fases N2 y N3 del sueño permite reducir el desbalanceo entre clases y mejorar los resultados en la detección de *spindles*. Asimismo, el uso de técnicas para la generación de *spindles* sintéticos como la adición de ruido gaussiano con SNR de 0 dB o el uso de ventanas deslizantes han sido adecuadas para evitar la clase mayoritaria y así conseguir reducir los FP y aumentar la precisión del sistema. Por el contrario, el empleo de funciones de pérdida (*Focal Loss*) o funciones de ajuste de pesos del sistema (*Custom Loss Weights*) no han proporcionado una buena solución ante este problema.
5. El empleo del post-procesado a la salida de los métodos de detección de *spindles* ha sido muy adecuado, permitiendo reducir el número de FP consiguiendo mejorar las medidas de rendimiento del sistema: Se, PPV y F1-score.
6. El uso de redes neuronales convolucionales más profundas, implementadas a partir del modelo **cnn_v4**, ha permitido al clasificador extraer información más precisa y concreta de los *spindles*, mejorando así su rendimiento. Además, modificar el *dropout* de la red ha permitido mejorar su PPV y F1-score, aunque es cierto que la sensibilidad del sistema disminuye. Con todo ello, se han alcanzado PPV=54.84%, F1-score=60.96% y Se=68.52%.
7. La arquitectura U-Net tiene un alto potencial en la detección de *spindles* ya que permite adoptar una estrategia de predicción muestra a muestra. Gracias a ello, se consiguen identificar directamente en la señal el comienzo y duración de los *spindles*.
8. En comparación con los modelos CNN desarrollados, la arquitectura U-Net es la que mejores rendimientos obtiene tanto en PPV (57.13% *vs.* 54.84%) como en F1-score (68.48% *vs.*60.96%) y Se (85.42% *vs.* 68.52%).

6.3 LÍNEAS FUTURAS

Con el objetivo de continuar este proyecto en líneas futuras, conviene resaltar algunos aspectos de este TFM que podrían estudiarse con mayor detalle y poder conseguir así un modelo con mejores prestaciones. Principalmente, dichos aspectos se encuentran relacionados con las limitaciones expuestas en el capítulo anterior.

1. Uno de los primeros pasos a realizar sería aumentar el tamaño de la base de datos con la que se trabaja. Además, sería interesante no solo realizar el estudio con niños en presencia de AOS, sino ampliar el número de sujetos con pacientes que presenten otro tipo de enfermedades con el objetivo de llevar a cabo una investigación más exhaustiva acerca del comportamiento que presentan los *spindles* en función de ello. Por otro lado, también sería una buena idea variar el rango de edades y el sexo de los pacientes con los que se realiza el estudio, aunque es cierto que hay un mayor número de investigaciones realizadas para este campo en adultos que en niños. Así, se podrían obtener modelos de *deep learning* más generalizables
2. Con respecto a los algoritmos basados en *deep learning* desarrollados en este TFM, sería recomendable continuar la investigación empleando arquitecturas U-Net, ya que esta rama ha sido la menos explorada en el estudio y ha generado unas buenas expectativas de cara a un trabajo futuro. Con ella se podrían tratar de aplicar técnicas para evitar el desequilibrio entre datos y sería interesante modificar ciertos hiperparámetros de la red para mejorar su potencial predictivo.
3. Otra línea futura interesante podría ser la aplicación de técnicas de inteligencia artificial explicable para detectar nuevos patrones/atributos inherentes a la señal de EEG vinculados con los *spindles* del sueño.
4. Otra línea de investigación innovadora sería ampliar el modelo desarrollado a un clasificador multiclase para que detecte no solo *spindles* del sueño, sino también otro tipo de señales contenidas en el EEG como son *cyclic alternating patterns* o *k-complex*, las cuales también están relacionadas con el desarrollo cognitivo. Esto permitiría llevar a cabo un estudio mucho más amplio en el que se podría ayudar a detectar otro tipo de trastornos, que no se pueden diagnosticar únicamente observando *spindles* del sueño.
5. En relación con el preprocesado de la señal, sería interesante probar a modificar el tipo de referenciado y verificar si este cambio ha sido significativo para los resultados o no. Asimismo, realizar pruebas con otros canales del EEG e incluso introducir algunos nuevos para el estudio sería una alternativa bastante original y prometedora en cuanto a los resultados esperados.
6. Por otra parte, a la vista de que el mayor problema que presenta el modelo desarrollado es el alto número de falsos positivos, podría ser interesante aplicar dentro del algoritmo algún tipo de arquitectura que diferencie las etapas del sueño N2 y N3 del resto, sin necesidad de invertir tiempo en modificar la señal original cada vez que se desee estimar la densidad de *spindles* para un determinado

sujeto. En este sentido, sería interesante evaluar los resultados utilizando únicamente la fase N2 para así poder compararse con estudios previos.

7. Para finalizar, sería interesante estudiar la asociación entre número de *spindles* y su duración con los parámetros neurocognitivos en niños con AOS u otras patologías. En este sentido, sería interesante desarrollar algún programa o aplicación, en base a los modelos obtenidos para la detección de *spindles*, que permita realizar una prueba a cada sujeto desde cualquier dispositivo electrónico en el domicilio del paciente y determinar si presentan algún tipo de afectación neurocognitiva. De este modo, los niños se beneficiarían de una detección y tratamiento tempranos para así mejorar su calidad de vida.

REFERENCIAS

Acir, N., & Güzeliş, C. (2004). Automatic recognition of sleep spindles in EEG by using artificial neural networks. *Expert Systems with Applications*, *27*(3), 451–458. https://doi.org/10.1016/j.eswa.2004.05.007

API TensorFlow. (2023). https://www.tensorflow.org/api_docs

AT, E., M, A., F, A.-M., & M, S. (2016). Classification of Imbalance Data using Tomek Link (T-Link) Combined with Random Under-sampling (RUS) as a Data Reduction Method. *Global Journal of Technology and Optimization*, *01*(S1). https://doi.org/10.4172/2229-8711.s1111

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A. F., & Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: An overview. *Bioinformatics*, *16*(5), 412–424. https://doi.org/10.1093/bioinformatics/16.5.412

Batta, M. (2020). Machine Learning Algorithms - A Review. *International Journal of Science and Research (IJ*, *9*(1), 381–386. https://doi.org/10.21275/ART20203995

Boughorbel, S., Jarray, F., & El-Anbari, M. (2017). Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric. *PLoS ONE*, *12*(6), 1–17. https://doi.org/10.1371/journal.pone.0177678

Calvo Merino, E. (2020). *Análisis de electroencefalogramas para la detección automática de las fases del sueño* (p. 99).

Chen, P., Chen, D., Zhang, L., Tang, Y., & Li, X. (2021). Automated sleep spindle detection with mixed EEG features. *Biomedical Signal Processing and Control*, *70*(August), 103026. https://doi.org/10.1016/j.bspc.2021.103026

Chiappa, K. H. (1997). *Evoked potentials in clinical medicine* (3rd ed). Lippincott-Raven. https://books.google.es/books?hl=es&lr=&id=h5mOjnTimT8C&oi=fnd&pg=PP11&dq=evoked+potentials&ots=uNoCVw0eOt&sig=t446oo33SjyCFpTTN9LVVlUB6vw#v=onepage&q=evoked potentials&f=false

Craik, A., He, Y., & Contreras-Vidal, J. L. (2019). Deep learning for electroencephalogram (EEG) classification tasks: A review. *Journal of Neural Engineering*, *16*(3). https://doi.org/10.1088/1741-2552/ab0ab5

Druzgalski, C., Vega, J. E. M., & Zeljkovic, V. (2016). *Determinación del Tamaño Óptimo de Modelos HMM-GMM para Clasificación de las Señales Bioacústicas*. *37*(1), 63–79.

Duman, F., Erdamar, A., Erogul, O., Telatar, Z., & Yetkin, S. (2009). Efficient sleep spindle detection algorithm with decision tree. *Expert Systems with Applications*, *36*(6), 9980–9985. https://doi.org/10.1016/j.eswa.2009.01.061

Eagan, B., Misfeldt, M., & Siebert-Evenstone, A. (2019). Advances in Quantitative Ethnography. In *Advances in Quantitative Ethnography* (Vol. 1). https://doi.org/10.1007/978-3-030-33232-7

Ebert-Uphoff, I., Lagerquist, R., Hilburn, K., Lee, Y., Haynes, K., Stock, J., Kumler, C., & Stewart, J. Q. (2021). *CIRA Guide to Custom Loss Functions for Neural Networks in Environmental Sciences -- Version 1*. 1–37. http://arxiv.org/abs/2106.09757

Ferreira, P., Le, D. C., & Zincir-Heywood, N. (2019). Exploring Feature Normalization and Temporal Information for Machine Learning Based Insider Threat Detection. *15th International Conference on Network and Service Management, CNSM 2019*, *Cnsm*. https://doi.org/10.23919/CNSM46954.2019.9012708

Ghaderi, Z., & Khotanlou, H. (2019). Weakly supervised pairwise Frank–Wolfe algorithm to

recognize a sequence of human actions in RGB-D videos. In *Signal, Image and Video Processing* (Vol. 13, Issue 8). https://doi.org/10.1007/s11760-019-01504-6

Gomez-Pilar, J., Gutiérrez-Tobal, G. C., Poza, J., Fogel, S., Doyon, J., Northoff, G., & Hornero, R. (2021). Spectral and temporal characterization of sleep spindles - methodological implications. *Journal of Neural Engineering*, *18*(3). https://doi.org/10.1088/1741-2552/abe8ad

Günesˆ, S., Dursun, M., Polat, K., & Yosunkaya, Ŝ. (2011). Sleep spindles recognition system based on time and frequency domain features. *Expert Systems with Applications*, *38*(3), 2455–2461. https://doi.org/10.1016/j.eswa.2010.08.034

Henry Olivi, R. (2013). Apnea del sueño: cuadro clínico y estudio diagnóstico. *Revista Médica Clínica Las Condes*, *24*(3), 359–373. https://doi.org/10.1016/s0716-8640(13)70173-1

Hermida L, R. C. (2016). El síndrome de apnea del sueño en los niños. *Revista Odontopediatria Latinoamericana*, *6*(2), 1–19.

Herrera, F. (2004). *Preprocesamiento de Datos en Mineria de Datos*. 6–7.

Herwig, U., Satrapi, P., & Schönfeldt-Lecuona, C. (2003). Using the International 10-20 EEG System for Positioning of Transcranial Magnetic Stimulation. *Brain Topography*, *16*(2), 95–99. https://doi.org/10.1023/B:BRAT.0000006333.93597.9d

Hossin, M., & Sulaiman, M. . (2015). a Review on Evaluation Metrics for Data. *International Journal of Data Mining & Knowledge Management Process (IJDKP)*, *5*(2), 1–11.

Hulsen, T. (2022). Literature analysis of artificial intelligence in biomedicine. *Annals of Translational Medicine*, *10*(23), 1284–1284. https://doi.org/10.21037/atm-2022-50

Kaulen, L., Schwabedal, J. T. C., Schneider, J., Ritter, P., & Bialonski, S. (2022). Advanced sleep spindle identification with neural networks. *Scientific Reports*, *12*(1), 1–10. https://doi.org/10.1038/s41598-022-11210-y

Khalid, S., Khalil, T., & Nasreen, S. (2014). A survey of feature selection and feature extraction techniques in machine learning. *Proceedings of 2014 Science and Information Conference, SAI 2014*, 372–378. https://doi.org/10.1109/SAI.2014.6918213

Kinoshita, T., Fujiwara, K., Kano, M., Ogawa, K., Sumi, Y., Matsuo, M., & Kadotani, H. (2020). Sleep Spindle Detection using RUSBoost and Synchrosqueezed Wavelet Transform. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *28*(2), 390–398. https://doi.org/10.1109/TNSRE.2020.2964597

Kulkarni, P. M., Xiao, Z., Robinson, E. J., Jami, A. S., Zhang, J., Zhou, H., Henin, S. E., Liu, A. A., Osorio, R. S., Wang, J., & Chen, Z. (2019). A deep learning approach for real-time detection of sleep spindles. *Journal of Neural Engineering*, *16*(3). https://doi.org/10.1088/1741-2552/ab0933

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal Loss for Dense Object Detection. *2020 7th International Conference on Advanced Informatics: Concepts, Theory and Applications, ICAICTA 2020*, 2980–2988. https://doi.org/10.1109/ICAICTA49861.2020.9428882

Manoach, D. S., Pan, J. Q., Purcell, S. M., & Stickgold, R. (2016). Reduced Sleep Spindles in Schizophrenia: A Treatable Endophenotype That Links Risk Genes to Impaired Cognition? *Biological Psychiatry*, *80*(8), 599–608. https://doi.org/10.1016/j.biopsych.2015.10.003

Martinek, R., Ladrova, M., Sidikova, M., Jaros, R., Behbehani, K., Kahankova, R., & Kawala-Sterniuk, A. (2021). Advanced bioelectrical signal processing methods: Past, present and future approach—Part II: Brain signals. *Sensors*, *21*(19), 1–32. https://doi.org/10.3390/s21196343

Mohammadi, H., Aarabi, A., Rezaei, M., Khazaie, H., & Brand, S. (2021). Sleep Spindle Characteristics in Obstructive Sleep Apnea Syndrome (OSAS). *Frontiers in Neurology*, *12*(February), 1–14. https://doi.org/10.3389/fneur.2021.598632

Pacho Velasco, V. (Universidad de V. (2022). *Detección de spindles del sueño mediante técnicas de intelogencia artificial*. *2100*, 45510.

Palacios, L. (2002). Breve historia de la electroencefalografia. *Acta Neurológica Colombiana*, *2*(2), 104–107.

Parra Sepúlveda, Roberto Humberto Montiel Ross, O., Díaz, G., & Gutierrez, D. (2015). Classification of Encephalographic signals using artificial neural networks. *Computacion y Sistemas*, *19*, 69–88. https://doi.org/10.13053/CyS-19-1-1570

Peralta, R. G., González-andino, S., & Gómez-gonzález, C. M. (2004). *grave_REN_2004*. *39*(8), 748–756.

Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M. P., Shyu, M. L., Chen, S. C., & Iyengar, S. S. (2018). A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys*, *51*(5). https://doi.org/10.1145/3234150

Purcell, S. M., Manoach, D. S., Demanuele, C., Cade, B. E., Mariani, S., Cox, R., Panagiotaropoulou, G., Saxena, R., Pan, J. Q., Smoller, J. W., Redline, S., & Stickgold, R. (2017). Characterizing sleep spindles in 11,630 individuals from the National Sleep Research Resource. *Nature Communications*, *8*(May), 1–16. https://doi.org/10.1038/ncomms15930

Richard B. Berry, MD; Rita Brooks, MEd, RST, RPSGT; Charlene E. Gamaldo, M., & Susan M. Harding, MD; Robin M. Lloyd, M. C. L. M. (2016). American Academy of Sleep Medicine. The AASM Manual for the Scoring of Sleep and Associated Events : Rules, Terminology, and Technical Specifications, Version 2.2. *American Academy of Sleep*, *28*(3), 391–397.

Sadikoglu, F., Kavalcioglu, C., & Dagman, B. (2017). Electromyogram (EMG) signal detection, classification of EMG signals and diagnosis of neuropathy muscle disease. *Procedia Computer Science*, *120*, 422–429. https://doi.org/10.1016/j.procs.2017.11.259

Schimicek, P., Zeitlhofer, J., Anderer, P., & Saletu, B. (1994). Automatic Sleep-Spindle Detection Procedure: Aspects of Reliability and Validity. *Clinical EEG and Neuroscience*, *25*(1), 26–29. https://doi.org/10.1177/155005949402500108

Sörnmo, L., & Laguna, P. (2005). *Bioelectrical signal processing in cardiac and neurological applications* (N. Stjernan (Ed.); Elsevier A). https://books.google.es/books?hl=es&lr=&id=RQv7tFFXYyIC&oi=fnd&pg=PP5&dq=bioelectrical+signal+processing&ots=hGBr8i9QxH&sig=SLUiiNg_FJat8rugnftejkJ3eq8#v=onepage&q=bioelectrical signal processing&f=false

Sors, A., Bonnet, S., Mirek, S., Vercueil, L., & Payen, J. F. (2018). A convolutional neural network for sleep stage scoring from raw single-channel EEG. *Biomedical Signal Processing and Control*, *42*, 107–114. https://doi.org/10.1016/j.bspc.2017.12.001

Taniguchi, H., Kohira, N., Ohnishi, T., Kawahira, H., von und zu Fraunberg, M., Jääskeläinen, J. E., Hauta-Kasari, M., Iwadate, Y., & Haneishi, H. (2015). Improving convenience and reliability of 5-ALA-induced fluorescent imaging for brain tumor surgery. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 9351). https://doi.org/10.1007/978-3-319-24574-4_25

Uong, E. C., Jeffe, D. B., Gozal, D., Arens, R., Holbrook, C. R., Palmer, J., Cleveland, C., & Schotland, H. M. (2005). Development of a measure of knowledge and attitudes about obstructive sleep apnea in children (OSAKA-KIDS). *Archives of Pediatrics and Adolescent Medicine*, *159*(2), 181–186. https://doi.org/10.1001/archpedi.159.2.181

Ventouras, E. M., Monoyiou, E. A., Ktonas, P. Y., Paparrigopoulos, T., Dikeos, D. G., Uzunoglu, N. K., & Soldatos, C. R. (2005). Sleep spindle detection using artificial neural networks trained with filtered time-domain EEG: A feasibility study. *Computer Methods and*

Programs in Biomedicine, *78*(3), 191–207. https://doi.org/10.1016/j.cmpb.2005.02.006

Wainberg, M., Merico, D., Delong, A., & Frey, B. J. (2018). Deep learning in biomedicine. *Nature Biotechnology*, *36*(9), 829–838. https://doi.org/10.1038/nbt.4233

Warby, S. C., Wendt, S. L., Welinder, P., Munk, E. G. S., Carrillo, O., Sorensen, H. B. D., Jennum, P., Peppard, P. E., Perona, P., & Mignot, E. (2014). Sleep-spindle detection: Crowdsourcing and evaluating performance of experts, non-experts and automated methods. *Nature Methods*, *11*(4), 385–392. https://doi.org/10.1038/nmeth.2855

Wei, L., Ventura, S., Mathieson, S., Boylan, G., Lowery, M., & Mooney, C. (2022). Spindle-AI: Sleep Spindle Number and Duration Estimation in Infant EEG. *IEEE Transactions on Biomedical Engineering*, *69*(1), 465–474. https://doi.org/10.1109/TBME.2021.3097815

Wei, L., Ventura, S., Ryan, M. A., Mathieson, S., Boylan, G. B., Lowery, M., & Mooney, C. (2022). Deep-spindle: An automated sleep spindle detection system for analysis of infant sleep spindles. *Computers in Biology and Medicine*, *150*(August). https://doi.org/10.1016/j.compbiomed.2022.106096

You, J., Jiang, D., Ma, Y., & Wang, Y. (2021). SpindleU-Net: An Adaptive U-Net Framework for Sleep Spindle Detection in Single-Channel EEG. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *29*, 1614–1623. https://doi.org/10.1109/TNSRE.2021.3105443

Yu, J., de Antonio, A., & Villalba-Mora, E. (2022). Deep Learning (CNN, RNN) Applications for Smart Homes: A Systematic Review. *Computers*, *11*(2), 1–32. https://doi.org/10.3390/computers11020026

ANEXO A: GLOSARIO DE SIGLAS

Acc: exactitud.

ANN: *Artificial Neural Network*.

AOS: apnea obstructiva del sueño.

CHAT: *Childhood Adenotosillectomy Tria*l.

CLW: *Custom Loss Weights.*

CNN: Red Neuronal Convolucional.

DL: *deep learning.*

ECG: electrocardiograma.

EEG: electroencefalograma.

EMG: electromiograma.

ENG: electroneurograma.

EP: potenciales evocados.

ERG: electroretinograma.

F1-score: valor-F.

FL: *Focal Loss.*

FN: *false negative.*

FP: *false positive.*

GN: *gaussian noise.*

HS: *heart sounds.*

IA: inteligencia artificial.

IoU: *intersection over union.*

LS: *lung sounds.*

M: *mirror.*

MCC: coeficiente de correlación de Matthews.

ML: *machine learning.*

MLP: *Multilayer Perceptron*.

MODA: *Massive Online Data Annotation.*

MrOS: *Osteoporotic Fractures in Men Study.*

NREM: *non-rapid Eye Movement.*

PSG: registro polisomnográfico.

PPV: valor predictivo positivo.

RNN: Red Neuronal Recurrente.

RMSE: raíz del error cuadrático medio.

RUS: *Random Under Sample.*

Se: sensibilidad o exhaustividad.

Sp: especificidad.

SST: *Synchrosqueezed Transform.*

SVM: *Support Vector Machine.*

SW: *sliding windorw.*

TFG: Trabajo Fin de Grado.

TFM: Trabajo Fin de Máster.

TP: *true positive.*

TN: *true negative.*

REM: *Rapid Eye Movement.*

ANEXO B: CÓDIGO DESARROLLADO

B.1 CNN DEL MODELO CNN_V1

```
model = models.Sequential()
model.add(layers.Conv1D(filters=64,           kernel_size=3,           activation='relu',
input_shape=(250,1,)))
model.add(layers.Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(layers.MaxPooling1D(pool_size=2,strides=1))
model.add(layers.SpatialDropout1D(rate=0.01))

model.add(layers.Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(layers.Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(layers.MaxPooling1D(pool_size=2,strides=1))
model.add(layers.SpatialDropout1D(rate=0.01))

model.add(layers.Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(layers.Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dropout(rate=0.01))

model.add(layers.Dense(units=2, activation='softmax')) # 2 neuronasde salida: 1 o 0
sgd_optimizer = tf.keras.optimizers.SGD(learning_rate=0.001, momentum=0.8)
model.compile(optimizer=sgd_optimizer, loss='binary_crossentropy', metrics=['accuracy'])
```

B.2 CLW DEL MODELO CNN_V4

```
################################################################################
#                            CUSTOM LOSS WEIGHTS                               #
################################################################################

def binary_crossentropy(weights):
    """
    A weighted version of keras.objectives.categorical_crossentropy

    Variables:
        weights: numpy array of shape (C,) where C is the number of classes

    Usage:
        weights = np.array([0.5,2,10]) # Class one at 0.5, class 2 twice the normal
weights, class 3 10x.
        loss = weighted_categorical_crossentropy(weights)
        model.compile(loss=loss,optimizer='adam')
    """

    weights = K.variable(weights)
```

```
    def loss(y_true, y_pred):
        # scale predictions so that the class probas of each sample sum to 1
        y_pred /= K.sum(y_pred, axis=-1, keepdims=True)
        # clip to prevent NaN's and Inf's
        y_pred = K.clip(y_pred, K.epsilon(), 1 - K.epsilon())
        # calc
        loss = y_true * K.log(y_pred) * weights
        loss = -K.sum(loss, -1)
        return loss

    return loss

weights=np.array([1,100]) # estoy dando peso 1 a la clase 0 (mayoritaria) y 100 a la
clase 1 (minoritaria)

model.compile(loss=binary_crossentropy(weights),
              optimizer=sgd_optimizer,
              metrics=["accuracy"])
```

B.3 FL DEL MODELO CNN_V4

```
################################################################################
#                                   FOCAL LOSS                                 #
################################################################################

def binary_focal_loss(gamma=2., alpha=.25):
    """
    Binary form of focal loss.
      FL(p_t) = -alpha * (1 - p_t)**gamma * log(p_t)
      where p = sigmoid(x), p_t = p or 1 - p depending on if the label is 1 or 0,
respectively.
    References:
        https://arxiv.org/pdf/1708.02002.pdf
    Usage:
     model.compile(loss=[binary_focal_loss(alpha=.25,  gamma=2)],  metrics=["accuracy"],
optimizer=adam)
    """

    def binary_focal_loss_fixed(y_true, y_pred):
        """
        :param y_true: A tensor of the same shape as `y_pred`
        :param y_pred:  A tensor resulting from a sigmoid
        :return: Output tensor.
        """
        y_true = tf.cast(y_true, tf.float32)
        # Define epsilon so that the back-propagation will not result in NaN for 0
divisor case
        epsilon = K.epsilon()
        # Add the epsilon to prediction value
        # y_pred = y_pred + epsilon
        # Clip the prediciton value
        y_pred = K.clip(y_pred, epsilon, 1.0 - epsilon)
        # Calculate p_t
        p_t = tf.where(K.equal(y_true, 1), y_pred, 1 - y_pred)
```

```
        # Calculate alpha_t
        alpha_factor = K.ones_like(y_true) * alpha
        alpha_t = tf.where(K.equal(y_true, 1), alpha_factor, 1 - alpha_factor)
        # Calculate cross entropy
        cross_entropy = -K.log(p_t)
        weight = alpha_t * K.pow((1 - p_t), gamma)
        # Calculate focal loss
        loss = weight * cross_entropy
        # Sum the losses in mini_batch
        loss = K.mean(K.sum(loss, axis=1))
        return loss

    return binary_focal_loss_fixed

#--------------------------------END FOCAL LOSS---------------------------------

model.compile(loss=binary_focal_loss(gamma=2., alpha=.25),
                  optimizer=sgd_optimizer,
                  metrics=["accuracy", "AUC"])
```

B.4 CNN PROFUNDA DEL MODELO CNN_V4

```
model = models.Sequential()
model.add(layers.Conv1D(filters=32,   kernel_size=3,padding='same',   activation='relu',
input_shape=(250,1,))) #  input_shape=(col,1)??
model.add(layers.Conv1D(filters=32, kernel_size=3,padding='same', activation='relu'))
model.add(layers.MaxPooling1D(pool_size=2,strides=2))
model.add(layers.SpatialDropout1D(rate=0.01))

model.add(layers.Conv1D(filters=32, kernel_size=3,padding='same', activation='relu'))
model.add(layers.Conv1D(filters=32, kernel_size=3,padding='same', activation='relu'))
model.add(layers.MaxPooling1D(pool_size=2,strides=2))
model.add(layers.SpatialDropout1D(rate=0.01))

model.add(layers.Conv1D(filters=64, kernel_size=3,padding='same', activation='relu'))
model.add(layers.Conv1D(filters=64, kernel_size=3,padding='same', activation='relu'))
model.add(layers.MaxPooling1D(pool_size=2,strides=2))
model.add(layers.SpatialDropout1D(rate=0.01))

model.add(layers.Conv1D(filters=64, kernel_size=3,padding='same', activation='relu'))
model.add(layers.Conv1D(filters=64, kernel_size=3,padding='same', activation='relu'))
model.add(layers.MaxPooling1D(pool_size=2,strides=2))
model.add(layers.SpatialDropout1D(rate=0.01))

model.add(layers.Conv1D(filters=128, kernel_size=3,padding='same', activation='relu'))
model.add(layers.Conv1D(filters=128, kernel_size=3,padding='same', activation='relu'))
model.add(layers.MaxPooling1D(pool_size=2,strides=2))
model.add(layers.SpatialDropout1D(rate=0.01))

model.add(layers.Conv1D(filters=128, kernel_size=3,padding='same', activation='relu'))
model.add(layers.Conv1D(filters=128, kernel_size=3,padding='same', activation='relu'))
model.add(layers.MaxPooling1D(pool_size=2,strides=2))
model.add(layers.SpatialDropout1D(rate=0.01))

model.add(layers.Conv1D(filters=128, kernel_size=3,padding='same', activation='relu'))
```

```
model.add(layers.Conv1D(filters=128, kernel_size=3,padding='same', activation='relu'))
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dropout(rate=0.01))

model.add(layers.Dense(units=2, activation='softmax'))
```

B.5 ARQUITECTURA U-NET

```
# Encoder:
conv1=                          layers.Conv1D(filters=32,kernel_size=11,padding='same',
activation='relu')(inputs)
conv1= layers.BatchNormalization()(conv1)
conv1=                     layers.Conv1D(filters=32,                    kernel_size=11,
padding='same',activation='relu')(conv1)
conv1= layers.BatchNormalization()(conv1)
conv1=         layers.Conv1D(filters=32,         kernel_size=11,         padding='same',
activation='relu')(conv1)
conv1= layers.BatchNormalization()(conv1)
pool1= layers.MaxPooling1D(pool_size=2)(conv1)
drop1= layers.Dropout(rate=0.2)(pool1)

conv2= layers.Conv1D(filters=64,kernel_size=11,padding='same', activation='relu')(drop1)
conv2= layers.BatchNormalization()(conv2)
conv2=                      layers.Conv1D(filters=64,                    kernel_size=11,
padding='same',activation='relu')(conv2)
conv2= layers.BatchNormalization()(conv2)
conv2=         layers.Conv1D(filters=64,         kernel_size=11,         padding='same',
activation='relu')(conv2)
conv2= layers.BatchNormalization()(conv2)
pool2= layers.MaxPooling1D(pool_size=2)(conv2)
drop2= layers.Dropout(rate=0.2)(pool2)

conv3=                          layers.Conv1D(filters=128,kernel_size=11,padding='same',
activation='relu')(drop2)
conv3= layers.BatchNormalization()(conv3)
conv3=                     layers.Conv1D(filters=128,                   kernel_size=11,
padding='same',activation='relu')(conv3)
conv3= layers.BatchNormalization()(conv3)
conv3=         layers.Conv1D(filters=128,         kernel_size=11,         padding='same',
activation='relu')(conv3)
conv3= layers.BatchNormalization()(conv3)
pool3= layers.MaxPooling1D(pool_size=2)(conv3)
drop3= layers.Dropout(rate=0.2)(pool3)


conv4=                          layers.Conv1D(filters=256,kernel_size=11,padding='same',
activation='relu')(drop3)
conv4= layers.BatchNormalization()(conv4)
conv4=                     layers.Conv1D(filters=256,                   kernel_size=11,
padding='same',activation='relu')(conv4)
conv4= layers.BatchNormalization()(conv4)
conv4=         layers.Conv1D(filters=256,         kernel_size=11,         padding='same',
activation='relu')(conv4)
conv4= layers.BatchNormalization()(conv4)
pool4= layers.MaxPooling1D(pool_size=2)(conv4)
```

```
drop4= layers.Dropout(rate=0.2)(pool4)

#Capa central
conv5=                          layers.Conv1D(filters=512,kernel_size=11,padding='same',
activation='relu')(drop4)
conv5= layers.BatchNormalization()(conv5)
conv5=                    layers.Conv1D(filters=512,                    kernel_size=11,
padding='same',activation='relu')(conv5)
conv5= layers.BatchNormalization()(conv5)
conv5=         layers.Conv1D(filters=512,         kernel_size=11,         padding='same',
activation='relu')(conv5)
conv5= layers.BatchNormalization()(conv5)


#Decoder
up6= layers.Dropout(rate=0.2)(conv5)
up6= layers.Conv1DTranspose(filters=256,kernel_size=2,strides=2)(up6)
#merge6=attention_gate(conv4,up6) # se agrega el attention gate
merge6= layers.concatenate([conv4,up6],axis=2)
conv6             =             layers.Conv1D(filters=256,             kernel_size=11,
padding='same',activation='relu')(merge6)
conv6 = layers.BatchNormalization()(conv6)
conv6             =             layers.Conv1D(filters=256,             kernel_size=11,
padding='same',activation='relu')(conv6)
conv6 = layers.BatchNormalization()(conv6)
conv6             =             layers.Conv1D(filters=256,             kernel_size=11,
padding='same',activation='relu')(conv6)
conv6 = layers.BatchNormalization()(conv6)

up7= layers.Dropout(rate=0.2)(conv6)
up7= layers.Conv1DTranspose(filters=128,kernel_size=2,strides=2)(up7)
merge7=layers.concatenate([conv3,up7],axis=2)
conv7             =             layers.Conv1D(filters=128,             kernel_size=11,
padding='same',activation='relu')(merge7)
conv7 = layers.BatchNormalization()(conv7)
conv7             =             layers.Conv1D(filters=128,             kernel_size=11,
padding='same',activation='relu')(conv7)
conv7 = layers.BatchNormalization()(conv7)
conv7             =             layers.Conv1D(filters=128,             kernel_size=11,
padding='same',activation='relu')(conv7)
conv7 = layers.BatchNormalization()(conv7)

up8= layers.Dropout(rate=0.2)(conv7)
up8= layers.Conv1DTranspose(filters=64,kernel_size=2,strides=2)(up8)
merge8=layers.concatenate([conv2,up8],axis=2)
conv8             =              layers.Conv1D(filters=64,             kernel_size=11,
padding='same',activation='relu')(merge8)
conv8 = layers.BatchNormalization()(conv8)
conv8             =              layers.Conv1D(filters=64,             kernel_size=11,
padding='same',activation='relu')(conv8)
conv8 = layers.BatchNormalization()(conv8)
conv8             =              layers.Conv1D(filters=64,             kernel_size=11,
padding='same',activation='relu')(conv8)
conv8 = layers.BatchNormalization()(conv8)

up9= layers.Dropout(rate=0.2)(conv8)
up9= layers.Conv1DTranspose(filters=32,kernel_size=2,strides=2)(up9)
merge9=layers.concatenate([conv1,up9],axis=2)
```

```
conv9              =              layers.Conv1D(filters=32,              kernel_size=11,
padding='same',activation='relu')(merge9)
conv9 = layers.BatchNormalization()(conv9)
conv9              =              layers.Conv1D(filters=32,              kernel_size=11,
padding='same',activation='relu')(conv9)
conv9 = layers.BatchNormalization()(conv9)
conv9              =              layers.Conv1D(filters=32,              kernel_size=11,
padding='same',activation='relu')(conv9)
conv9 = layers.BatchNormalization()(conv9)

#Capa de salida
outputs= layers.Conv1D(num_classes,1,activation='softmax')(conv9)

#Crear el modelo
model=Model(inputs=inputs,outputs=outputs)
```