# Una red convolucional para la clasificación de las fases de sueño

1st Isaac Fernández-Varela
*CITIC*
*Universidade da Coruña*
A Coruña, España
isaac.fvarela@udc.es

2nd Elena Hernández-Pereira
*CITIC*
*Universidade da Coruña*
A Coruña, España
elena.hernandez@udc.es

3rd Diego Alvarez-Estevez
*Sleep Center & Clinical Neurophysiology*
*Haaglanden Medisch Centrum*
The Hague, The Netherlands
diego.alvarez@udc.es

4th Vicente Moret-Bonillo
*CITIC*
*Universidade da Coruña*
A Coruña, España
vicente.moret@udc.es

***Resumen*—La clasificación de fases de sueño es una tarea crucial en el contexto de los estudios de sueño que incluye el análisis de varias señales simultáneamente y, por tanto, es tediosa y compleja. Incluso para un experto entrenado puede costar varias horas anotar las señales registradas durante el sueño de una única noche. Para resolver este problema se han desarrollado métodos automáticos, la mayor parte basándose en características inherentes al conjunto de datos utilizado. En este trabajo evitamos esa imparcialidad resolviendo el problema con un modelo de *deep learning* que puede aprender las características relevantes de las señales del paciente sin nuestra intervención. En concreto, proponemos un *ensemble* de 5 redes convolucionales que obtiene un índice kappa de $0,83$ clasificando 500 registros polisomnográficos.**

***Palabras clave*—red convolucional, fases de sueño, clasificación**

## I. Introducción

Los trastornos del sueño afectan a una parte mayoritaria de la población. Como ejemplo, el 20% de los adultos españoles sufren insomnio, y entre el 12% y el 15% somnolencia diurna [1, 2]. Dormir bien es esencial para tener buena salud y las consecuencias de la falta de sueño son bien conocidas [3]. Para el diagnóstico de los trastornos del sueño es útil la identificación de las diferentes fases que atraviesa el sueño del paciente. Con este objetivo, la técnica más utilizada es el polisomnograma (PSG) que registra las señales biomédicas del paciente, entre las que se encuentran señales pneumológicas, electrofisiológicas e información contextual. Este análisis es caro, incómodo para el paciente y de difícil interpretación. Una manera habitual de simplificar esta interpretación es con el uso del hipnograma, la representación ordenada de la evolución de las fases de sueño.

El estándar de oro para la construcción del hipnograma es la guía de la *American Academy of Sleep Medicine* (AASM) [4] para la identificación de fases del sueño y sus eventos asociados, los despertares, los movimientos y los eventos cardíacos y respiratorios. Dicha guía identifica cinco fases de sueño: Despierto (*Awake*, W), movimientos oculares rápidos (*Rapid Eye Movements*, REM), y tres fases de sueño lento o no REM (N1, N2 y N3). Un hipnograma construido correctamente facilita encontrar problemas y diagnosticar trastornos del sueño, permitiendo enfocar el tiempo en la terapia. Su construcción implica analizar una gran cantidad de información y conocimiento [5]. Además, pese a las guías el acuerdo entre expertos es usualmente inferior al 90%. Por ejemplo, Stepnowsky et al. [6] estudiaron el acuerdo entre dos expertos obteniendo índices kappa entre 0,48 y 0,89. De manera similar, Wang et al. [7] obtuvieron índices entre 0,72 y 0,85. A mayores, el acuerdo también es menor para fases concretas, siendo la fase N1 la que obtiene los peores resultados.

Por todo ello, la automatización de la clasificación de fases de sueño es una tarea necesaria. La mayor parte de los métodos a día de hoy siguen una aproximación de dos pasos. Primero, extraen características específicas para este problema, muchas veces dependientes de los datos utilizados. Después, construyen un vector de características con el que se entrena algún clasificador y predicen las fases de sueño. Algunos autores utilizan un único canal de una señal y otros utilizan múltiples canales, construyendo un vector de varios elementos. Las características extraídas pueden pertenecer tanto al dominio del tiempo como al de la frecuencia. Aunque algunos trabajos utilizan una única señal, en este caso siempre el electroencefalograma (EEG), otros utilizan varias, incluyendo electrooculograma (EOG) o electromiograma (EMG), para adaptarse a las guías de la AASM.

Entre los métodos que utilizan extracción de características para su posterior clasificación encontramos los siguientes: Fraiwan et al. [8], utilizan un *random forest* para la clasificación de características del dominio del tiempo-frecuencia y las características de entropía de Reny; Liang et al. [9], obtienen la entropía en múltiples escalas y características autoregresivas analizándolas con un discriminante linear; Has-

* Esta investigación ha sido financiada en parte por la Xunta de Galicia (ED431G/01) y la Unión Europea a través del fondo ERDF.

san and Bhuiyan [10], utilizan una única señal con transformaciones *wavelet* para la extracción de características y un *random forest* para la clasificación. Sharma et al. [11], también comparan varios clasificadores, utilizando filtros iterativos para analizar un único canal de EEG; Koley and Dey [12], entrenan una *support vector machine (SVM)* con características de frecuencia, de tiempo y no lineares extraídas de un único canal de EEG; Lajnef et al. [13], utilizan varias señales y múltiples SVM para crear un árbol de decisión; Huang et al. [14], estudian la densidad espectral de potencia de 2 canales de EEG para clasificar las características de frecuencia utilizando una modificación de una SVM; Finalmente, Günes et al. [15], también analizan la densidad espectral de potencia pero clasificando con un algoritmo de *nearest neighbords*.

Solucionar el problema de clasificación de fases de sueño con extracción de características provoca sesgos por el diseño de características basadas en un único conjunto de datos. Por ello, las propuestas anteriores no generalizaban bien, concretamente dada la naturaleza de los registros de PSG, que presentan variaciones debido al paciente junto con las que provoca el hardware o el método de adquisición utilizado.

Una alternativa para resolver este problema es utilizar métodos que puedan aprender de los datos, evitando el sesgo humano. En este sentido, la apuesta natural es el *deep learning* ya que ha demostrado mejoras frente a métodos tradiciones en múltiples campos en general y en el diagnóstico médico en particular [16, 17].

Ya existen trabajos que exploran distintos modelos de *deep learning*: Längkvist et al. [18], utilizan redes *deep belief* para aprender una representación probabilística de señales preprocesadas de PSG; Tsinalis et al. [19], extraen características de una señal EEG y después utilizan redes convolucionales para la clasificación. Los mismos autores en otro trabajo [20] utilizan una pila de *sparse autoencoders*; Supratak et al. [21], utilizan una red convolucional con una red recurrente bidireccional para clasificar directamente a partir de las señales; Biswal et al. [22], comparan una red recurrente contra otros modelos, pero entrenados con características en vez de con la propia señal; Finalmente, Sors et al. [23] también utilizan una red convolucional sobre un único canal de EEG.

En este trabajo se utiliza *deep learning* para clasificar las fases de sueño a través de una red convolucional que puede aprender las características relevantes de cada fase. Siguiendo las guías de la AASM utilizamos múltiples señales; en particular, dos canales de EEG, un canal de EMG y ambos canales de EOG (derecho e izquierdo). Además, las señales se filtran previamente, para reducir el ruido y eliminar artefactos.

## II. Materiales

El desarrollo y análisis del modelo que se presenta se realiza utilizando registros reales de pacientes. Dichos registros pertenecen al *Sleep Heart Health Study* (SHHS) [24], una base de datos que ofrece la *Case Western University* que proviene de un estudio entre varios centros dirigido por el *National Heart Lung and Blood Institute* para determinar las consecuencias cardiovasculares de los trastornos de sueño asociados a la respiración.

Cada registro incluye las anotaciones de distintos eventos, realizadas por expertos siguiendo las reglas de la AASM [25]. Todos los registros se anonimizaron y anotaron a ciegas. El montaje utilizado para la adquisición de las señales incluye entre otras señales dos derivaciones de EEG (C4A2 y C4A1), EOG derecho e izquierdo, un EMG submental, electrocardiograma(ECG). El EEG, EOG y EMG están muestreados a 125 Hz mientras que los EOG están a 50 Hz. Las señales se filtraron durante su adquisición con un filtro paso alto a 0,15 Hz.

Usamos tres conjuntos de datos distintos para entrenar, validar y testear nuestro modelo. El conjunto de entrenamiento incluye 400 registros, el de validación 100 y el de test 500. La duración de los registros de entrenamiento se iguala (se incluyen 7 horas aleatorias por registro) para facilitar la codificación del algoritmo de entrenamiento del modelo. De esta manera, tenemos $288,000$ ejemplos para entrenar, $119,121$ para la validación y $606,981$ para el test. Los registros se escogieron de manera aleatoria incluyendo aquellos con mucho ruido o muchos artefactos.

Las distribuciones de las diferentes clases, tanto para el conjunto de datos entero como para cada registro individual se muestran en la Tabla I. Esta tabla demuestra el desbalanceo de los conjuntos de datos, siendo la fase W predominante (aproximadamente el 38% de los casos), aunque la proporción es similar para la fase N2 (aproximadamente el 36%). Por el contrario, la clase N1 solo está representada en un 3%. También es interesante destacar que algunos registros no tienen ninguno de los casos para alguna clase y lo mucho que varían las proporciones entre ellos. Por ejemplo, en el conjunto de test, mientras un registro contiene un 7,10% para la clase N2, otro contiene un 83,43%. De hecho, estos son los dos principales problemas cuando se clasifican fases de sueño: 1) el gran desbalanceo de las clases y 2) las diferencias entre registros individuales.

## III. Método

### A. Filtrado de las señales

Las señales son procesadas para eliminar ruido y artefactos comunes. Ambas operaciones son pasos habituales en trabajos previos utilizados antes de la extracción de características.

El primero de los dos filtros utilizados para eliminar ruido es un filtro *Notch* centrado en 60 Hz para eliminar la interferencia que causa la línea de corriente. Este filtro se aplica a las señales con un muestreo superior a 60 Hz, EEG y EMG. El segundo elimina las frecuencias no relacionadas con movimientos musculares del EMG, utilizando un filtro paso alto a 15 Hz.

En cuanto a los artefactos, la mayor parte ocurren durante períodos muy concretos y cortos de tiempo, haciendo que sea incluso difícil reconocerlos. De cualquier manera, los artefactos ECG, causados por las interferencias del latido del corazón, son comunes y constantes a lo largo de toda la señal. Eliminamos estos artefactos usando un filtro adaptativo. Para ello, primero obtenemos la serie de latidos utilizando un

Tabla I
Distribución de las distintas clases en los conjuntos de entrenamiento, validación y test.

| | | W | N1 | N2 | N3 | REM | Total |
|---|---|---|---|---|---|---|---|
| *Conjunto de entrenamiento* | Total | 187.513 | 17.283 | 172.451 | 44.454 | 62.168 | 483.869 |
| | Proporción | 38,75 % | 3,57 % | 35,64 % | 9,19 % | 12,85 % | 100 % |
| | Min en un registro | 8,20 % | 0,00 % | 12,59 % | 0,00 % | 0,00 % | |
| | Max en un registro | 71,61 % | 13,75 % | 68,65 % | 33,43 % | 26,58 % | |
| *Conjunto de validación* | Total | 43.742 | 3.963 | 43.510 | 12.900 | 15.006 | 119.121 |
| | Proporción | 36,72 % | 3,33 % | 36,53 % | 10,83 % | 12,60 % | 100 % |
| | Min en un registro | 11,21 % | 0,29 % | 12,38 % | 0,00 % | 0,00 % | |
| | Max en un registro | 76,79 % | 17,08 % | 60,09 % | 30,16 % | 23,68 % | |
| *Conjunto de test* | Total | 231.707 | 19.769 | 217.246 | 61.281 | 76.978 | 606.981 |
| | Proporción | 37,77 % | 3,26 % | 35,96 % | 10,25 % | 12,75 % | 100 % |
| | Min en un registro | 7,75 % | 0,00 % | 7,10 % | 0,00 % | 0,00 % | |
| | Max en un registro | 76,53 % | 16,93 % | 83,43 % | 43,82 % | 31,11 % | |

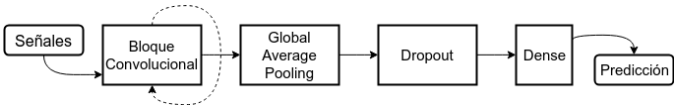


Figura 1. Red convolucional propuesta.

algoritmo estándar de detección de ondas QRS [26]. Después estudiamos la calidad de la señal de ECG para saber que intervalos podemos incluir en la construcción del filtro adaptativo. Por último, durante los intervalos con suficiente calidad, aplicamos y actualizamos el filtro adaptativo para eliminar los artefactos. Este proceso se describe en profundidad en Fernández-Varela et al. [27].

### B. Red convolucional

La clasificación de las fases de sueño se realiza habitualmente con ventanas de 30 s, llamadas epochs. Analizando múltiples características de cada epoch, los expertos clínicos pueden decidir cual es la fase de sueño correspondiente a dicho epoch.

Una red convolucional [28] es una red *deep-forward* que soluciona las limitaciones del perceptrón multicapa con una arquitectura que comparte pesos. Básicamente, aplica una operación de convolución sobre la entrada, reduciendo el número de parámetros. Por eso, permite construir redes más profundas que facilitan reconocer características más complejas. La red propuesta se representa en la Figura 1.

La red convolucional recibe como entrada el conjunto de señales (2 canales de EEG, EMG y ambos EOG). Cada entrada es un epoch: ventana de 30 segundos. Debido a la diferencia de muestreo entre las señales, tal y como se comentó en la Sección II, se realiza una operación de *upsampling* a 125 Hz. Se descarta un *downsample* a 50 Hz porque perderíamos las frecuencias altas del EEG que clínicamente contienen información interesante para clasificar las fases de sueño. De la misma forma, se descarta una operación de *padding* porque no sería fácilmente extensible a otros conjuntos de datos con señales adquiridas con otras frecuencias de muestreo. De esta manera, cada entrada de la red convolucional es una matriz de $3750 \times 5$. Cada señal se normaliza con media 0 y desviación 1, obteniendo la media y desviación a partir de todas las señales correspondientes del conjunto de datos de entrenamiento. Usando otras normalizaciones de menor granularidad las redes probadas inicialmente no convergían. El bloque convolucional que se muestra en la Figura 1 es una sucesión de cuatro capas incluyendo una convolución 1D que preserva el tamaño de la entrada (con *padding*), una capa de *batch normalization* [29] para la regularización, una activación ReLu [30] y un *average pool* que reduce el tamaño de la entrada por un factor de 2. Usando una convolución 1D evitamos imponer una estructura espacial que desconocemos entre las distintas señales. Este bloque se repitió $n$ veces, siendo $n$ un hiperparámetro cuyo valor fue seleccionado durante la experimentación. Todas las capas se configuraron con el mismo tamaño de kernel pero el número de filtros para la capa $i$ es dos veces el número de filtros de la capa $i-1$. La selección de $n$, el tamaño de kernel y el número inicial de filtros se explica en la siguiente sección, junto a otros hiperparámetros.

La salida del último bloque convolucional, tras ajustar la dimensión con un *global pooling* y aplicarle *dropout* para mejorar la regularización, se utiliza como entrada en una capa densa con activación *softmax*. Esta capa devuelve la probabilidad de cada clase para la entrada. Como es habitual, se selecciona la clase con mayor probabilidad como decisión de clasificación.

Para entrenar la red se utiliza el optimizador Adam [31] con 64 elementos por *batch*. Este tamaño de *batch* está condicionado por el hardware disponible para la ejecución. Del optimizador se configura el hiperparámetro ratio de aprendizaje, manteniendo ambas betas con los valores por defecto. El entrenamiento se termina utilizando *early stopping* monitorizando la pérdida en el conjunto de validación con una paciencia de 10 epochs. Debido al desbalanceo de las clases se utiliza *cross entropy* ponderada, obteniendo los pesos del conjunto de entrenamiento.

### C. Optimización de hiperparámetros

Una correcta elección de los hiperparámetros puede significar el éxito de un modelo de*deep learning*. La dificultad a

la hora de seleccionar los mejores hiperparámetros no es sólo obtener el mejor rendimiento sino en conseguirlo minimizando el coste al hacerlo, pudiendo ser este coste económico o computacional.

En este trabajo se utiliza un *Tree-structured Parzen Estimator* (TPE) que se ha mostrado superior frente a otros métodos [32, 33]. El TPE es un aproximación secuencial de optimización basada en modelos (SMBO). Los métodos SMBO construyen secuencialmente modelos para aproximar el rendimiento de una selección de hiperparámetros basándose en resultados históricos y así escoger nuevos hiperparámetros que se comprueban con el modelo. Particularmente, TPE modela dos distribuciones $P(x|y)$ y $P(y)$ donde $x$ representa los hiperparámetros e $y$ el rendimiento asociado, y optimiza la mejora esperada (*expected improvement*, EI) siguiendo la ecuación:

$$EI_{y^*}(x) = \int_{-\infty}^{y^*} (y^* - y)\frac{P(x|y)P(y)}{P(x)}$$

donde $y^*$ es algún cuantil $\gamma$ de los valores observados $y$ tal que $p(y < y^*) = \gamma$.

Utilizamos TPE para optimizar los siguientes hiperparámetros relacionados con la red convolucional: el número de bloques convolucionales, el tamaño del kernel de cada convolución 1D y el número de filtros del primer bloque. A mayores, existe una relación entre el número inicial de filtros y el número de bloques convolucionales. Dadas nuestras restricciones computacionales no añadimos bloques que tuviesen más de 1024 filtros.También se utiliza TPE para el ratio de aprendizaje del optimizador. Las distribuciones usadas para cada uno de estos hiperparámetros se resumen en la Tabla II.

Tabla II
Distribuciones para los distintos hiperparámetros

| Hiperparámetro | Distribución |
|---|---|
| Bloques convolucionales | Uniforme entre 1 y 10 |
| Tamaño del kernel | Uniforme entre 3 y 50 |
| Filtros iniciales | Elección entre 8, 16, 32 o 64 |
| Ratio de aprendizaje | Log-uniforme entre -10 y -1 |

Para reducir el tiempo computacional de selección de los hiperparámetros utilizamos un subconjunto del conjunto de entrenamiento para entrenar, validar y hacer el test de los distintos modelos. Este subconjunto contiene 250 registros de los que 20 se utilizan para validación durante el entrenamiento y 50 para el test de cada modelo. En total probamos 50 modelos utilizando el índice kappa para seleccionar el mejor.

### D. Medidas de rendimiento

El rendimiento de los modelos se evalúa con las siguientes medidas:

- **Precisión,** la fracción entre el número de verdaderos positivos y el número de predicciones positivas.
- **Sensitividad**, la fracción entre el número de verdaderos positivos y el número elementos de la clase.

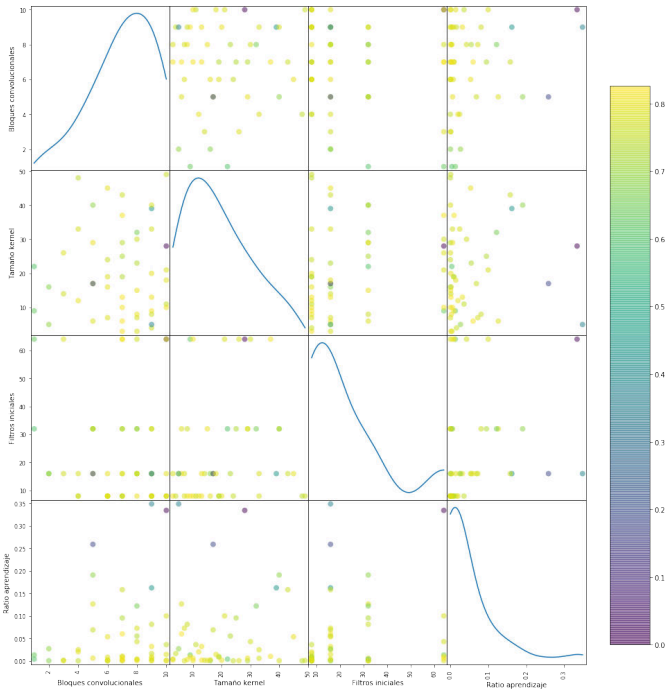


Figura 2. Gráfico de dispersión para los diferentes conjuntos de hiperparámetros probados. El color del punto representa el índice kappa para el modelo con dichos hiperparámetros. La diagonal representa la distribución de los valores asignados para ese hiperparámetro concreto.

- ***F1 score***, la media armónica de precisión y sensitividad.
- **Kappa**, la medida del acuerdo entre dos clasificadores que tiene en cuenta la posibilidad de acuerdo casual. El acuerdo perfecto obtiene un valor de 1 y solo por casualidad un valor 0.

## IV. Resultados

Antes de ver los resultados conseguidos podemos visualizar el rendimiento que obtuvieron los distintos modelos probados para la búsqueda de hiperparámetros en la Figura 2. Destaca la necesidad de un ratio de aprendizaje bajo para que el entrenamiento converja adecuadamente.

Para mejorar los resultados obtenidos por nuestro modelo utilizamos un *ensemble*. Varios modelos clasifican las fases de sueño de manera individual y el resultado final es la fase más repetida. En este caso, hemos escogido los 5 mejores modelos obtenidos durante la selección de hiperparámetros. Los valores para los hiperparámetros de cada uno de ellos se presentan en la Tabla II.

Los resultados obtenidos por el *ensemble* utilizando el conjunto de test se muestran en la Tabla IV. La mejor clasificación se obtiene para la clase W, con valores cercanos a $0,95$ para la precisión, sensitividad y *F1 score*; después las clases N2, N3 y REM presentan resultados similares, especialmente si nos fijamos en la *F1 score*, aunque con menor sensitividad para la clase N3 y, por tanto, también mayor precisión. Por último, los resultados no son los esperados para la clasificación de la fase N1, no llegándose a un *F1 score* de $0,3$. Típicamente, N1 es la clase más difícil de clasificar incluso para los expertos.

Tabla III
Hiperparámetros para los 5 mejores modelos

| Parámetro | Modelo 1 | Modelo 2 | Modelo 3 | Modelo 4 | Modelo 5 |
|---|---|---|---|---|---|
| Bloque convolucionales | 7 | 9 | 7 | 7 | 7 |
| Tamaño del kernel | 6 | 9 | 13 | 3 | 10 |
| Filtros iniciales | 16 | 8 | 8 | 8 | 64 |
| Ratio de aprendizaje | $5,99 \times 10^{-2}$ | $9,00 \times 10^{-3}$ | $1,45 \times 10^{-3}$ | $1,91 \times 10^{-3}$ | $5,49 \times 10^{-3}$ |

Tabla IV
Medidas de rendimiento para la clasificación del conjunto de test utilizando el ensemble de los 5 mejores modelos.

| **Fase** | Precisión | Sensitividad | *F1 score* |
|---|---|---|---|
| W | 0,94 | 0,96 | 0,95 |
| N1 | 0,39 | 0,21 | 0,27 |
| N2 | 0,87 | 0,89 | 0,88 |
| N3 | 0,92 | 0,77 | 0,84 |
| REM | 0,82 | 0,90 | 0,86 |
| **Average** | 0,78 | 0,75 | 0,76 |

La matriz de confusión obtenida con el *ensemble* se muestra en la Figura 3, en la que se puede comprobar que la mayor parte de las fases N1 son clasificadas como otras fases, especialmente N2. Además, aunque en una proporción mucho menor, cuando se comete un error clasificando una clase distinta a N2, también se tiende a clasificar como N2.

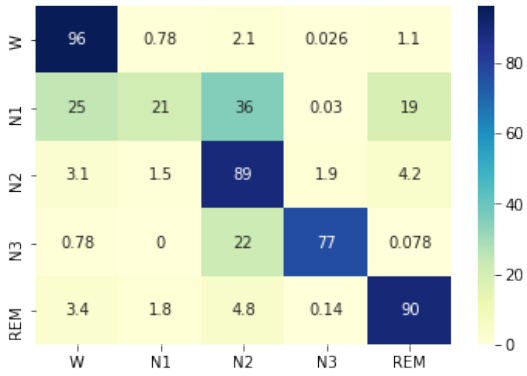


Figura 3. Matriz de confusión para la clasificación del conjunto de test utilizando el *ensemble* de los 5 mejores modelos.

## V. Discusión y Conclusiones

En este trabajo presentamos un *ensemble* de redes convolucionales para la clasificación de fases de sueño. Es una tarea que consume mucho tiempo y crítica a la hora de diagnosticar trastornos del sueño. La mayor parte de los métodos automáticos de clasificación de fases de sueño se basan en características diseñadas para un conjunto de datos concreto y no suelen, por tanto, adaptarse bien a otros conjuntos de datos. Para solucionar este problema usamos una red convolucional que puede aprender las características relevantes de las señales involucradas en la tarea por su cuenta.

Un aspecto importante para el éxito o el fracaso de los modelos convolucionales es la elección correcta de los hiperparámetros. En nuestro caso concreto trabajamos con 4 hiperparámetros, optimizando su selección con un *tree-structured parzen estimator* y evaluando 50 modelos distintos.

El *ensemble* propuesto, a partir de las 5 mejores configuraciones de hiperparámetros, obtiene una precisión, sensitividad y un *F1 score* medias de $0,78$, $0,75$ y $0,76$ respectivamente, con un índice kappa de $0,83$. Aunque globalmente los resultados son aceptables, nuestra solución ha mostrado problemas para clasificar la fase N1. Además, cuando la clasificación es incorrecta suele ser hacia la clase N2.

La comparación frente a trabajos similares es difícil por la falta de estándares, tanto en los conjuntos de datos como en el proceso de evaluación. En la Tabla V mostramos los resultados de trabajos anteriores, limitándonos a los que ofrecen valores para cada clase. Como se puede ver, obtenemos el índice kappa más alto, aunque no los mejores *F1 score*. Salvo para la clase W, algún trabajo presenta mejor *F1 score*. De cualquier manera, los valores que obtenemos son competitivos, con la excepción de la clase N1, aunque queda claro en la comparación que dicha clasificación es la más difícil. Frente al único trabajo que presenta resultados con un conjunto de datos similar [23], obtenemos mejor índice Kappa y *F1 score* para la clase W, con valores casi idénticos para N2, N3 y REM aunque bastante más bajos para N1.

Los resultados son prometedores y el método escogido debería ser adaptable a otros conjuntos de datos, especialmente si además se puede adaptar el entrenamiento. Además, entrenando simultánemante con varios conjuntos de datos la red debería generalizar mejor, evitando especializarse en registros concretos.

Para mejorar estos resultados es necesario explicar como se hace la clasificación. Además, sería interesante introducir memoria en el modelo, posiblemente en forma de red recurrente, porque en ciertas condiciones, la clasificación de un epoch depende de los epochs anteriores.

## Bibliografía

[1] M. M. Ohayon and T. Sagales, "Prevalence of insomnia and sleep characteristics in the general population of spain." *Sleep medicine*, vol. 11, no. 10, pp. 1010–8, dec 2010.

[2] J. Marin *et al.*, "Prevalence of sleep apnoea syndrome in the spanish adult population," *International Journal of Epidemiology*, vol. 26, no. 2, pp. 381–386, apr 1997.

[3] H. R. Colten and B. M. Altevogt, *Sleep Disorders and Sleep Deprivation*. Washington, D.C.: National Academies Press, sep 2006, vol. 6, no. 9.

Tabla V
Comparación de resultados de trabajos previos para la clasificación de fases del sueño.

| Trabajo | Conjunto de datos | Kappa | F1 score | | | | |
|---|---|---|---|---|---|---|---|
| | | | W | N1 | N2 | N3 | REM |
| Biswal et al. [22] | Massachusetts General Hospital, 1000 registros | 0,77 | 0,81 | **0,70** | 0,77 | 0,83 | **0,92** |
| Längkvist et al. [18] | St Vicent's University Hospital, 25 registros | 0,63 | 0,73 | 0,44 | 0,65 | **0,86** | 0,80 |
| Sors et al. [23] | SHHS, 1730 registros | 0,81 | 0,91 | 0,43 | 0,88 | 0,85 | 0,85 |
| Supratak et al. [21] | MASS dataset, 62 registros | 0,80 | 0,87 | 0,60 | **0,90** | 0,82 | 0,89 |
| Supratak et al. [21] | SleepEDF, 20 registros | 0,76 | 0,85 | 0,47 | 0,86 | 0,85 | 0,82 |
| Tsinalis et al. [19] | SleepEDF, 39 registros | 0,71 | 0,72 | 0,47 | 0,85 | 0,84 | 0,81 |
| Tsinalis et al. [20] | SleepEDF, 39 registros | 0,66 | 0,67 | 0,44 | 0,81 | 0,85 | 0,76 |
| This work | SHHS, 500 registros | **0,83** | **0,95** | 0,27 | 0,88 | 0,84 | 0,86 |


[4] R. B. Berry *et al.*, "AASM Scoring Manual Updates for 2017 (Version 2.4)." *Journal of clinical sleep medicine : JCSM : official publication of the American Academy of Sleep Medicine*, vol. 13, no. 5, pp. 665–666, may 2017.

[5] Á. Fernández-Leal *et al.*, "A knowledge model for the development of a framework for hypnogram construction," *Knowledge-Based Systems*, vol. 118, pp. 140–151, 2017.

[6] C. Stepnowsky *et al.*, "Scoring accuracy of automated sleep staging from a bipolar electroocular recording compared to manual scoring by multiple raters." *Sleep medicine*, vol. 14, no. 11, pp. 1199–207, nov 2013.

[7] Y. Wang *et al.*, "Evaluation of an automated single-channel sleep staging algorithm." *Nature and science of sleep*, vol. 7, pp. 101–11, 2015.

[8] L. Fraiwan *et al.*, "Automated sleep stage identification system based on time–frequency analysis of a single EEG channel and random forest classifier," *Computer Methods and Programs in Biomedicine*, vol. 108, no. 1, pp. 10–19, oct 2012.

[9] J. Liang *et al.*, "Predicting seizures from electroencephalography recordings: A knowledge transfer strategy," in *Proceedings - 2016 IEEE International Conference on Healthcare Informatics, ICHI 2016*. IEEE, oct 2016, pp. 184–191.

[10] A. R. Hassan and M. I. H. Bhuiyan, "A decision support system for automatic sleep staging from EEG signals using tunable Q-factor wavelet transform and spectral features," *Journal of Neuroscience Methods*, vol. 271, pp. 107–118, sep 2016.

[11] R. Sharma, R. B. Pachori, and A. Upadhyay, "Automatic sleep stages classification based on iterative filtering of electroencephalogram signals," *Neural Computing and Applications*, vol. 28, no. 10, pp. 2959–2978, oct 2017.

[12] B. Koley and D. Dey, "An ensemble system for automatic sleep stage classification using single channel EEG signal," *Computers in Biology and Medicine*, vol. 42, no. 12, pp. 1186–1195, 2012.

[13] T. Lajnef *et al.*, "Learning machines and sleeping brains: Automatic sleep stage classification using decision-tree multi-class support vector machines," *Journal of Neuroscience Methods*, vol. 250, pp. 94–105, jul 2015.

[14] C.-S. Huang *et al.*, "Knowledge-based identification of sleep stages based on two forehead electroencephalogram channels," *Frontiers in Neuroscience*, vol. 8, p. 263, sep 2014.

[15] S. Günes, K. Polat, and S. Yosunkaya, "Efficient sleep stage recognition system based on EEG signal using k-means clustering based feature weighting," *Expert Systems with Applications*, vol. 37, no. 12, pp. 7922–7928, dec 2010.

[16] A. Esteva *et al.*, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, feb 2017.

[17] V. Gulshan *et al.*, "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *JAMA*, vol. 316, no. 22, p. 2402, dec 2016.

[18] M. Längkvist *et al.*, "Sleep Stage Classification Using Unsupervised Feature Learning," *Advances in Artificial Neural Systems*, vol. 2012, pp. 1–9, 2012.

[19] O. Tsinalis *et al.*, "Automatic sleep stage scoring with single-channel eeg using convolutional neural networks," oct 2016.

[20] O. Tsinalis, P. M. Matthews, and Y. Guo, "Automatic sleep stage scoring using time-frequency analysis and stacked sparse autoencoders," *Annals of Biomedical Engineering*, vol. 44, no. 5, pp. 1587–1597, may 2016.

[21] A. Supratak *et al.*, "Deepsleepnet: A model for automatic sleep stage scoring based on raw single-channel eeg," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 11, pp. 1998–2008, nov 2017.

[22] S. Biswal *et al.*, "Sleepnet: Automated sleep staging system via deep learning," jul 2017.

[23] A. Sors *et al.*, "A convolutional neural network for sleep stage scoring from raw single-channel EEG," *Biomedical Signal Processing and Control*, vol. 42, pp. 107–114, apr 2018.

[24] S. F. Quan *et al.*, "The sleep heart health study: Design, rationale, and methods," *Sleep*, vol. 20, no. 12, pp. 1077–1085, dec 1997.

[25] M. H. Bonnet *et al.*, "EEG arousals: scoring rules and examples: a preliminary report from the Sleep Disorders Atlas Task Force of the American Sleep Disorders Association." *Sleep*, vol. 15, no. 2, pp. 173–184, apr 1992.

[26] V. Afonso *et al.*, "Ecg beat detection using filter banks," *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 2, pp. 192–202, 1999.

[27] I. Fernández-Varela *et al.*, "A simple and robust method for the automatic scoring of EEG arousals in polysomnographic recordings," *Computers in Biology and Medicine*, vol. 87, pp. 77–86, aug 2017.

[28] Y. Le Cun *et al.*, "Handwritten digit recognition: applications of neural network chips and automatic learning," *IEEE Communications Magazine*, vol. 27, no. 11, pp. 41–46, nov 1989.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015.

[30] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," *Proceedings of the 27th International Conference on Machine Learning*, no. 3, pp. 807–814, 2010.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," dec 2014.

[32] J. Bergstra *et al.*, "Algorithms for hyper-parameter optimization," in *NIPS*, 2011.

[33] J. Bergstra, D. Yamins, and D. D. Cox, "Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms," in *Proc. of the 12th python in science conf*, 2013.