Predicción de probabilidad de tener los ojos abiertos en los siguientes intantes de tiempo a partir de información registrada en señales electroencefalograficas

Proyecto Fundamentos de Deep learning

Susana Mejía Echeverry

1. Contexto

Extraer información a partir de electroencefalografía (EEG) resulta valioso en campos como la bioingeniería, medicina y neurociencia. Además, saber si la persona tiene los ojos abiertos o cerrados permite diferenciar estados de conciencia. El EEG durante los períodos de ojos abiertos y cerrados muestra patrones distintos de actividad cerebral. Durante los ojos abiertos, la actividad alfa disminuye, mientras que durante los ojos cerrados, la actividad alfa aumenta.

En este proyecto se han utilizado 14 señales cerebrales (recolectadas a partir de la disposición de electrodos de superficie en la cabeza, tal como se muestra en la Figura 1) de un sujeto que se encuentra abriendo y cerrando los ojos de manera aleatoria durante 117 segundos, para predecir la probabilidad de que tenga los ojos abiertos o cerrados en los instantes posteriores.

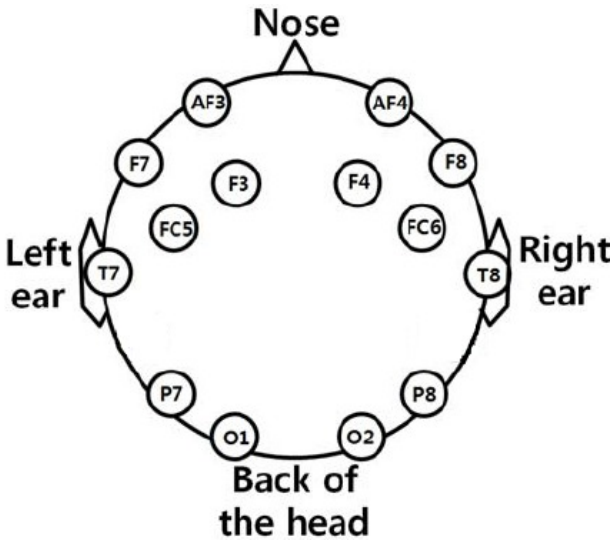

Figura 1: Disposición de electrodos de superficie para registro de señales cerebrales.

2. Estructura de los notebooks

Para cumplir con el objetivo de predicción se han elaborado cuatro notebooks. Cada uno de ellos se lista y describe en la Tabla 1. El número indica el orden en que deben ser ejecutados.

Tabla 1: Descripción de la estructura de los notebooks.

Número	Notebook	Descripción
1	Contexto	Explicación simplificada de la toma de los datos, su etiquetado y el objetivo de predicción para uso de redes neuronales recurrentes
2	Exploración de datos	Revisión del dataset, eliminación de datos atípicos y filtrado
3	Preprocesado	Creación de estructuras de entrada (X) y salida (Y) del modelo, normalización y feature engineering
4	Arquitectura de línea base	Diseño de red neuronal, iteraciones y resultados

3. Descripción de solución

En la Figura 2 se presentan los pasos realizados para la elaboración del proyecto, en la que se destacan cuatro etapas principales: exploración de los datos, preprocesado, decisión de la arquitectura y toma de decisiones según los resultados. Las líneas azules remarcan los pasos iterativos realizados cuando la respuesta del modelo presentaba un bajo valor de ajuste con los datos de entrenamiento y validación.

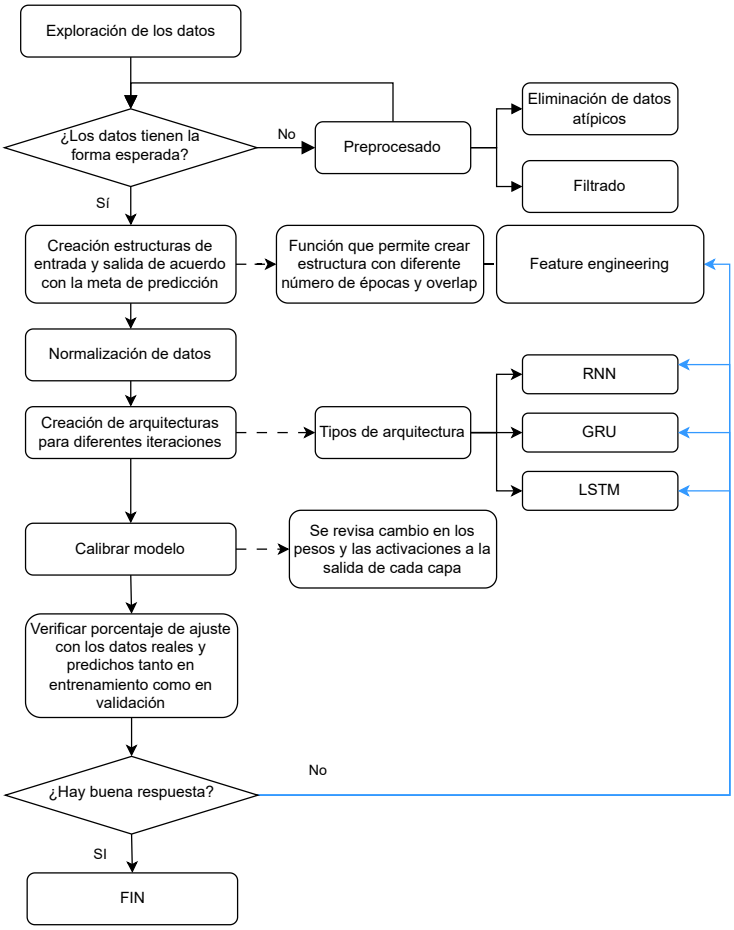

Figura 2: Diagrama de flujo de la metodología implementada.

Considerando que se trata de predicción en el tiempo, se emplea una red neuronal recurrente (RNN); además, para hacer varias pruebas, se implementan arquitecturas de Gated recurrent unit (GRU) y Long short term memory (LSTM). La cantidad de parámetros y capas para cada arquitectura se presenta en la Tabla 2. Independiente del tipo de arquitectura, se utilizó una capa densa de salida con una neurona y activación *softmax* con el fin de obtener a la salida del modelo un valor entre 0 y 1, que pueda ser asumido como un porcentaje.

Tabla 2: Características de las tres arquitecturas consideradas para las pruebas.

Arquitectura	Parámetros	Capas	Capa de entrada	Capa de salida
RNN	632,091	8	RNN con 350 neuronas	Densa con 1 neurona
GRU	445,601	6	GRU con 200 neuronas	Densa con 1 neurona
LSTM	445,651	5	LSTM con 200 neuronas	Densa con 1 neurona

Con el objetivo de tener datos adecuados para ingresar a la red y obtener mejores resultados en la predicción,

la información de cada señal fue preprocesada de la siguiente manera:

- Se eliminaron los datos atípicos de acuerdo con los resultados de media y desviación estándar del conjunto de datos para cada señal.
- Una vez comprobada la forma de onda esperada según la literatura, se filtraron los datos para eliminar el rizado proveniente de los procesos de captación.
- Las iteraciones estaban demostrando saturación de las neuronas, por lo que se normalizaron los datos, dividiendo por el valor máximo del dataset, para tener valores entre 0 y 1 y así disminuir la saturación de los pesos de la red.

Una vez preprocesados los datos, se crean las estructuras presentadas en la Figura 3 para separar la información en la entrada (X) y salida (Y) con los cuales fue calibrado el modelo. El conjunto de datos de entrada se trata de un arreglo tridimensional donde se dividen los datos de las señales por épocas permitiendo overlapping. Este diseño se consideró como estrategia para aumentar el número de datos bajo la hipótesis de mejorar el rendimiento del modelo, reducir el sesgo, aumentar la variabilidad y mejorar la adaptabilidad de la red. Por su parte, se ha creado un vector con los porcentajes de cada época como datos de salida de acuerdo con el objetivo de modelado. Tal como se muestra en la imagen, independiente de la cantidad de épocas y la separación de los datos, la longitud del vector de salida debe ser equivalente a la longitud del número de épocas de los datos de entrada.

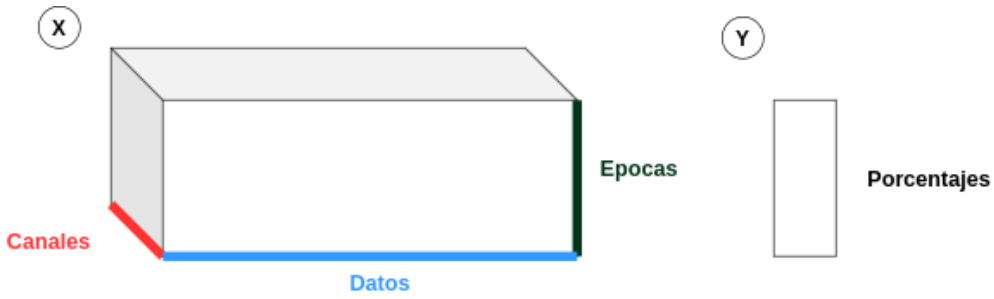

Figura 3: Estructuras de entrada (X) y salida (Y) para el entrenamiento del modelo.

Además de ingresar los datos crudos a la red, se realizó una extracción de características para aumentar el número de pruebas, tener otra perspectiva del rendimiento de la red y tener información representativa de cada conjunto de datos. Para esto, se calculó el promedio, la desviación estándar, el valor máximo y el valor mínimo de cada partición de los datos en cada época manteniendo las estructuras de la Figura 3.

4. Descripción de iteraciones

Las iteraciones realizadas se detallan en las líneas azules de la Figura 2. Cada nuevo ensayo se hizo una vez se comprobó el bajo ajuste en la predicción de los datos de respuesta de entrenamiento y validación.

4.1. Iteraciones cambiando la arquitectura

Al ver que las neuronas se saturaban y no proporcionaban un valor flotante en el vector de salida, se decidió por aumentar la complejidad de la red: agregar mayor cantidad de neuronas y más capas densas a la salida de la RNN. Además, se iteró sobre diferentes arquitecturas como LSTM y GRU ya que con este tipo de redes se disminuyen las desventajas de las redes neuronales recurrentes.

Si bien los resultados de ajuste cambiaban conforme se modificaba la complejidad o el tipo de red, este tipo de iteración no proporcionó cambios significativos en el nivel de predicción del modelo.

4.2. Iteraciones cambiando el tamaño de los datos

Como se mencionó anteriormente, una buena cantidad de datos favorece la predicción y evita el sobreajuste. El dataset no tenía muchos datos, por lo que se preparó una función para poder variar en cada prueba el tamaño

de las secciones de las señales y el valor de overlapping de manera que se encontrara las dimensiones óptimas de entrada al modelo.

En las primeras iteraciones no se utilizó overlapping para comprobar si bastaba con una estructura de entrada con dimensiones pequeñas, una de ellas fue de (48, 14, 298). Los resultados demostraron que 48 épocas no eran suficientes para obtener buen porcentaje de ajuste entre los datos reales y predichos. En respuesta, se realizaron pruebas con solapamiento de datos. Fueron utilizados diferentes tamaños de entrada con overlapping pero se dispuso un tamaño de (143, 200, 100) para triplicar el número de épocas respecto a arreglo sin solapamiento.

Estos cambios mejoraron el comportamiento de la minimización de la función de coste, ya que la perdida (loss) se fue reduciendo a en cada época del entrenamiento.

4.3. Iteraciones cambiando los valores de entrada

Una vez se demostró que los datos crudos no mostraban mejoras en el valor de ajuste, se extrajeron características de los datos para darle mayor información a la red. Se calculó el promedio, la desviación estándar, el valor máximo y mínimo con el objetivo de agregar información que dé cuenta de la distribución de los datos.

5. Resutados

A pesar de la cantidad de iteraciones realizadas y tras aumentar la cantidad de información usando overlapping, el hecho de tener pocos datos afectó en el nivel de predicción del modelo. Incluso, como se muestra en las Figuras 4 y 5 para el comportamiento de los pesos y las activaciones respectivamente, se observa que las capas no se activan mucho conforme avanza el entrenamiento y esto se de reflejado en la variación de los pesos a la salida de cada capa.

Para mejorar esto, los pesos fueron inicializados en la primera capa. Sin embargo, aunque hubo pequeños cambios, no fue significativo.

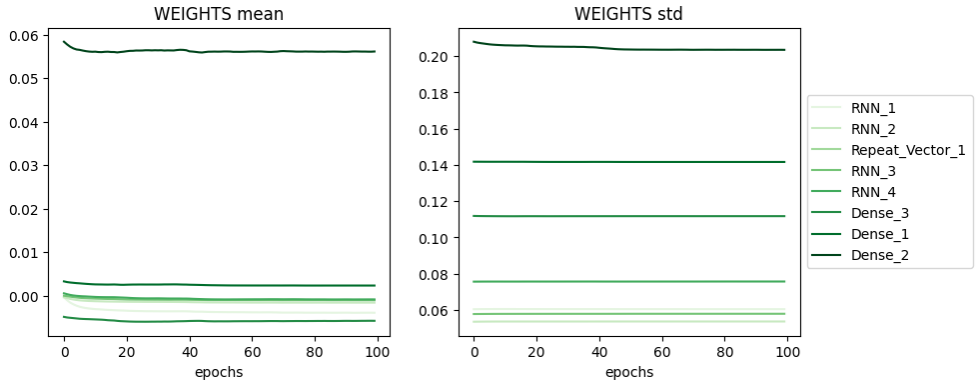

Figura 4: Cambio de los pesos conforme avanza el entrenamiento

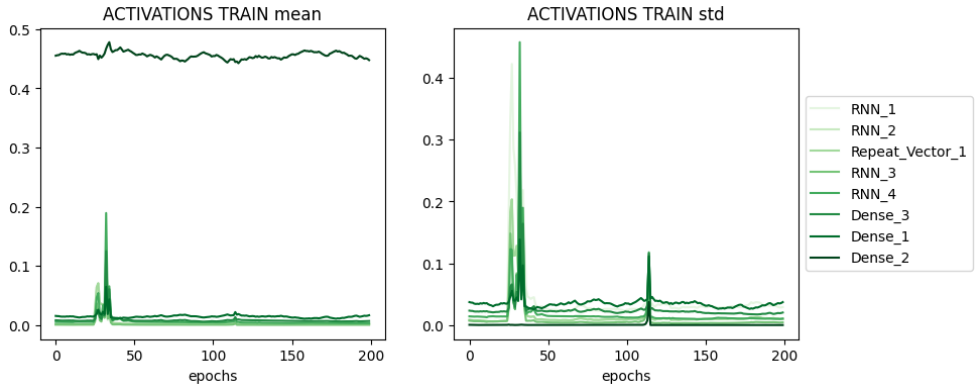

Figura 5: Activaciones a la salida de cada capa conforme avanza el entrenamiento.

A pesar de los inconvenientes tenidos producto de la poca activación de las capas, el porcentaje de ajuste para el entrenamiento y validación para dos casos diferentes, uno ingresando a la red los datos crudos y otro con la estructura con las características extraídas, se presentan en la Tabla 3.

Tabla 3: Porcentaje de ajuste en entrenamiento y validación para dos datos de entrada diferentes.

Prueba	Ajuste entrenamiento	Ajuste validación
Datos crudos	34 %	36 %
Características extraídas	35 %	45 %

Estos porcentajes pueden mejorar teniendo un dataset más poblado, mayor dificultad en la red y manejando diferentes tipos de inicialización de los pesos. Además, evitar tener el modelo como caja negra y sacar información del comportamiento de los pesos y activaciones, favorece la toma de decisiones para obtener mejores respuestas.