Universidad Nacional Autónoma de México

Facultad de ingeniería

TESIS

Redes neuronales artificiales aplicadas a la clasificación de litofacies en registros geofísicos de pozo

Para obtener el titulo de

Ingeniero Geofísico

Por

Arturo Ruiz Sánchez

Director de tesis:

Dr. Iván Vladimir Meza Ruiz

Ciudad de México. México. 2023.

Agradecimientos

*To professor **Matteo Ravasi**, who guided me through the development of this project while being an intern in his research group. Thank you for believing in me, for showing me how amazing doing research is and for giving me the opportunity to live a life-changing experience. These words are not enough to describe how grateful I am.*

*Al **Dr. Iván Vladimir Meza Ruiz**, por haberme brindado su apoyo durante el proceso de titulación, por sus aportaciones a la realización de este trabajo y su excelente disposición en cada momento.*

*A mis sinodales, el **Dr. Mauricio Nava Flores**, la **Dra. Iza Canales García**, el **Mtro. Jorge Guízar Alfaro** y el **Dr. Luis Miguel De la Cruz Salas**, por sus importantes comentarios y sugerencias sobre la estructura y contenido de este trabajo.*

*A los **amigos** que hice durante estos 5 años de experiencia universitaria, por compartir tantas anécdotas y momentos juntos, son personas a las que admiro mucho y siempre me impulsaron a ser mejor. Por un futuro lleno de más anécdotas y éxitos.*

*A mis **familiares**, porque a lo largo de mi vida siempre han estado presentes en cada momento de éxito y de dificultad. Por las tantas despedidas y por esperarme permanentemente con los brazos abiertos para compartir juntos un domingo más. Desde que nací creyeron en mí y nunca han dejado de estar a mi lado, por lo que este es mi regalo para ustedes.*

*A mi hermanito **Alex**. El ser tu hermano es mi motivación más grande para ser una persona de bien. Gracias por enseñarme a luchar por mis sueños tal y como tu lo haces día con día. Siempre me esforzaré por ser un ejemplo para tí, así como lo eres tú para mí. Por una vida juntos, te quiero hermano.*

*A ti **mamá**, por tus noches de desvelo, por tus ojeras y lágrimas, por hacerte la fuerte cada que nos vamos de casa mi hermano y yo. Por haber tenido esa valentía y coraje de ser mamá, esposa, trabajadora y estudiante al mismo tiempo. Porque siempre estas de pie, de sol a sol, ante las adversidades y sin quejarte, en la parada del taxi con dos niños o en el aeropuerto esperando a que regresen convertidos en adultos. Sé que siempre seremos tus niños, pero tu siempre serás mi orgullo.*

*A ti **papá**, por dedicar tu vida a cumplir los sueños de tus hijos. Porque aún faltando los recursos, siempre buscaste la manera de que tus hijos tengan la mejor educación y sean felices. Pero aún más importante, gracias por secarte esas lágrimas cuando caminabas cansado por el malecón con los zapatos rotos después de un día de trabajar y estudiar al mismo tiempo, porque hoy eres el ingeniero que siempre he querido ser, pero sobre todo, el papá que algún día seré con tus nietos.*

Resumen

Se le define como litofacies a la subdivisión de una unidad estratigráfica que se caracteriza únicamente por su textura, mineralogía, tamaño de grano y ambiente geológico donde se deposita. La correcta identificación de la presencia de estos cuerpos sedimentarios en los registros geofísicos de pozo es de suma importancia para la exploración de hidrocarburos, ya que pueden indicar la presencia de unidades sedimentarias de alto valor para la caracterización de un yacimiento, por ejemplo, unidades generadoras y/o almacenadoras.

Existen distintas metodologías para la identificación manual de litofacies en registros geofísicos de pozo, las cuales requieren de analistas/intérpretes capacitados con conocimientos en geología y petrofísica, restringidos por la capacidad de procesamiento humana para realizar dicha tarea.

En este trabajo, exploramos el uso de redes neuronales basadas en mecanismos de recurrencia para la clasificación de litofacies a partir de registros geofísicos de pozo, de tal manera que esta tecnología pueda servir como apoyo a los intérpretes humanos para agilizar el proceso de clasificación de litofacies ayudados por la capacidad de procesamiento de una computadora.

Las redes neuronales completamente conectadas son conocidas por desempeñarse peor en esta tarea que los ensambles de algoritmos clásicos de aprendizaje de máquina, por lo cual utilizamos una red neuronal bidireccional de gran memoria a corto plazo, mejor conocida como LSTM Bidireccional, tal que sea posible tomar en cuenta la relación espacial entre litofacies.

Las distintas variantes de modelos fueron entrenadas utilizando un conjunto de datos sintéticos cuyo diseño se explica en este trabajo, para posteriormente probar los modelos entrenados en un conjunto de datos reales correspondientes a la región del Mar del Norte en Noruega, que pertenecen al Norwegian Petroleum Dictatorate y fueron compartidos para el concurso "FORCE 2020 Machine Learning Contest", cuyo propósito fue motivar a los concursantes a diseñar algoritmos supervisados de aprendizaje de máquina que fueran capaces de clasificar litofacies a partir de registros geofísicos de pozo.

La evaluación de los resultados de este trabajo se realizó a partir de la comparación de métricas de precisión, matrices de confusión y "FORCE score", una métrica propuesta específicamente para evaluar el desempeño de los algoritmos desarrollados para la competencia antes mencionada.

Abstract

Lithofacies are defined as the subdivisions of a stratigraphic unit that is uniquely based on its texture,mineralogy, grain size, and the geological environment where they were produced. The accurate identification of the presence of these sedimentary bodies in well logs is of high importance for hydrocarbons exploration, given that they are able to indicate the presence of high-value sedimentary units for reservoir characterization, e.g. generator and reservoir units.

Different methodologies exist for manual identification of lithofacies in well logs, which require of analists/interpreters with geological and petrophysical acumen, restrained by the human processing capabilities to perform this task.

In this work, we explore the use of recurrence-based neural networks for lithofacies classification from well logs, such that this technology can be an useful resource for human interpreters in order to speed up their lithofacies classification workflow, aided by the processing power of a computer.

Fully-connected neural networks are known for performing worse on this task than the ensembles of classic machine learning algorithms, this is why we chose to use a bi-directional long short-term memory unit neural network, better known as Bidirectional LSTM, which takes into account the spatial relationship betweeen lithofacies.

The different model variants were trained using a synthetic dataset, which design is further explained, and then they were tested on a real-world dataset of the Northern Sea region in Norway, that belongs to the Norwegian Petroleum Dictatorate and was shared for the "FORCE 2020 Machine Learning Contest", which goal was to encourage contestants to design supervised machine learning algorithms capable of classifying lithofacies from well logs.

Model evaluation of the results was performed by comparing accuracy metrics, confusion matrices and "FORCE score", a metric proposed specifically to evaluate the performance of the algorithms designes for the aforementioned contest.

Índice general

Capítulo 1

Introducción

Durante varias décadas, la innovación y los avances tecnológicos desarrollados para modernizar a la industria petrolera se centraron en la creación de herramientas físicas modernas y sofisticadas para llevar a cabo los procesos que componen a la cadena de valor en esta industria de manera eficaz y eficiente. A la par, tanto en centros de investigación especializados como en la academia, se ha realizado un esfuerzo muy importante para enriquecer el conocimiento teórico sobre ramas científicas e ingenieriles como la geología, la geofísica, la petrofísica, la ingeniería de yacimientos, entre otras.

A raíz de la revolución tecnológica de finales del siglo pasado y principios del actual, se empezó a trazar un camino distinto en el desarrollo tecnológico de la industria, el cual nos lleva a la revolución del *software* como catalizador principal de la innovación; tanto en la industria petrolera como en el resto de actividades económicas.

Los medios de comunicación y diversos autores, empezaron a hablar acerca de una nueva revolución tecnológica a mediados de la década de los 2010's, cuya pieza central son los algoritmos de inteligencia artificial y el supercómputo. Este nuevo paradigma tecnológico propone la utilización de cantidades masivas de datos para enseñar a algoritmos complejos a realizar tareas de manera similar a los seres humanos.

Tanto la geofísica, como la industria petrolera, se han reinventado constantemente a partir de cada una de estas revoluciones tecnológicas y esta última no ha sido la excepción, ya que la Inteligencia Artificial ha abierto la brecha en la exploración de métodos que agilizen a procesos que consumen grandes cantidades de tiempo y esfuerzo humano. Ejemplos de ellos en la industria petrolera destacan, la identificación de fallas en secciones sísmicas (An et al., 2021), la delimitación de cuerpos salinos (Consolvo et al., 2021) e incluso la interpretación de registros geofísicos de pozo (Wu et al., 2018).

La motivación de este trabajo se centra en uno de los procesos mencionados anteriormente: la interpretación de registros geofísicos de pozo, en específico la identificación de litofacies a partir de ellos utilizando algoritmos de inteligencia artificial. La razón de esto es que este proceso consume grandes cantidades de tiempo y energía para ser realizados por intérpretes humanos, a su vez de un vasto conocimiento en temas como la petrofísica. Sin embargo, el objetivo no es diseñar un algoritmo capaz de sustituir a un intérprete humano, si no el de utilizar modelos de redes neuronales preexistentes y comparar su funcionamiento para que puedan ser utilizados por un especialista para apoyar y agilizar su labor de interpretación de registros geofísicos de pozo.

Para llevar a cabo este objetivo, mediremos el rendimiento de algoritmos de tres algoritmos de inteligencia artificial: Multilayer Perceptron (MLP), Long Short-Term Memory Unit (LSTM)

y Bidirectional Long Short-Term Memory Unit (BiLSTM), para identificar litofacies a partir de registros geofísicos de pozo generados de manera sintética o bien, correspondientes a pozos reales de la región del Mar del Norte en Noruega. De esta manera, obtenemos una medida base del rendimiento de cada uno de estos algoritmos y podemos comparar su eficiencia.

El contenido de este trabajo ha sido dividido en cinco secciones principales:

- **Registros Geofísicos de Pozo**: Describe teóricamente el principio de funcionamiento de los registros geofísicos de pozo, así como conceptos fundamentales que ayudan a entender su importancia en la exploración petrolera.
- **Aprendizaje Automático**: Establece las bases matemáticas para entender el funcionamiento de los algoritmos de inteligencia artificial, así como su inspiración en la biología.
- **Metodología**: Explica el proceso de creación de un conjunto de datos sintéticos utilizados para entrenar algoritmos de inteligencia artificial, así como las distintas arquitecturas de modelos propuestas.
- **Resultados**: Realiza una comparación del rendimiento de cada uno de los modelos probados en ambos conjuntos de datos, para medir la eficacia de cada uno de ellos.
- **Conclusiones**: Discusión sobre los resultados obtenidos por los distintos modelos y comentarios finales.

Capítulo 2

Estado del arte

El problema de clasificación de litofacies a partir de registros geofísicos de pozo ha evolucionado en cuanto al enfoque utilizado para tratar de resolverlo. Si bien existen diversos métodos de interpretación conocidos; tales como la utilización de gráficas cruzadas, lógica difusa o análisis de cluster, la clasificación de litofacies se ha estudiado desde hace un par de décadas como un problema que puede ser resuelto con un enfoque meramente estadístico.

(Eidsvik et al., 2004) propuso un enfoque utilizando modelos de cadenas ocultas de Markov para estimar atributos geológicos a partir de registros de pozo, sin embargo, este método fue desarrollado específicamente para el conjunto de datos con el que contaban los autores. Posteriormente, (Dubois et al., 2007) trató de combinar el análisis probabilístico realizado anteriormente con un enfoque basado en datos, siendo uno de los primeros trabajos que trataron de realizar una comparación entre estas dos metodologías al medir el rendimiento de distintos modelos pertenecientes a cada una de ellas.

A partir de la masificación de herramientas de cómputo capaces de entrenar efectivamente algoritmos de aprendizaje automático, se empezó a popularizar el uso de estos algoritmos en la exploración del problema de clasificación de litofacies. Ejemplo de ello son los múltiples esfuerzos por utilizar algoritmos *clásicos* de aprendizaje automático, como Random Forest (Y. Kim et al., 2018), o algoritmos un poco más populares en la actualidad, como XGboost (L. Zhang y Zhan, 2017).

De manera reciente, se ha optado por utilizar redes neuronales profundas de distintos tipos para resolver este problema. (Tschannen et al., 2017) utilizó redes neuronales convolucionales que se basan en el principio de correlación cruzada para clasificar litofacies a partir de registros geofísicos de pozo modelados como si fueran secuencias de una sola dimensión. Utilizando otro enfoque, (Grana et al., 2020) comparó el rendimiento de redes neuronales recurrentes y redes de gran memoria a corto plazo (LSTM) con métodos Monte Carlo en este problema. De esta manera, las LSTM fueron utilizadas para tomar en cuenta la correlación espacial entre las propiedades de entrada que son los registros de pozo, tal y como se realiza en este trabajo, aunque en este caso exploramos una variante de las LSTM's; la LSTM Bidireccional, para comparar su rendimiento clasificando litofacies con el de un perceptón multicapa (MLP).

Capítulo 3

Registros geofísicos de Pozo

3.1. Definición de los registros de pozo

Los registros geofísicos de pozo representan mediciones con respecto a la profundidad de ciertas características físicas de las formaciones rocosas a partir de algún dispositivo que viaje a través del pozo (Serra, 1984), a este dispositivo comúnmente se le conoce como *sonda* y tiene distintas configuraciones y componentes dependiendo del tipo de registro que se requiera tomar, que a su vez está condicionado acorde al objetivo de exploración correspondiente.

Las propiedades medidas por los registros geofísicos de pozo representan condiciones *in situ* de la roca, a diferencia del análisis de recortes o de núcleos de perforación, cuyas muestras se analizan en condiciones atmosféricas y no de yacimiento. Por lo tanto, una manera efectiva y relativamente económica de monitorear las condiciones del yacimiento para su correcta explotación es haciendo uso de estos registros.

Algunos registros requieren que la herramienta tenga un contacto directo con las paredes del pozo, por otra parte, existen registros cuya herramienta se encuentra *centrada* en el pozo, es decir, que no tiene contacto con las paredes del mismo; esto depende del fundamento físico de la medición que se está efectuando y/o de las condiciones del agujero. En cualquiera de estos dos escenarios y en general para los registros geofísicos de pozo, estos no miden parámetros que puedan determinar directamente la existencia de hidrocarburos en la formación de estudio o la producción de ellos, en contraste, a partir de propiedades medidas como resistividad, velocidad sísmica, radiación gamma, etc., es posible derivar parámetros que indican, en un cierto rango de certidumbre, la presencia o efecto de hidrocarburos (Bjorlykke, 2015).

3.2. Objetivo de los registros geofísicos de pozo

Hoy en día, los registros de pozo cumplen distintos objetivos dependiendo del área de aplicación, ya que su uso no es exclusivo en la industria de los hidrocarburos, su aplicación es tan variada que incluso se encuentran casos de uso para geotermia (Rudman, 1978), hidrogeología (Farrag et al., 2019), entre otras áreas. Sin embargo, el objetivo general de los registros geofísicos de pozo es el de proveer un registro continuo a través del pozo que permita observar los cambios graduales o abruptos en las propiedades físicas de un estrato de roca a otro, de manera que sea posible identificar propiedades que caracterizen al objeto de estudio en cuestión.

3.3. Importancia de los registros geofísicos de pozo para la exploración petrolera

Los registros geofísicos de pozo juegan un papel muy importante en el ciclo de vida de un yacimiento, el cual esta definido por 5 fases distintas (Satter y Iqbal, 2016), comprendidas por: Exploración, Descubrimiento, Evaluación y Delimitación, y Producción y Abandono. Podría decirse que toman un rol relevante en todas las fases del ciclo de vida del yacimiento a excepción del abandono, ya que los registros pueden utilizarse para determinar la ubicación de un pozo productor, delimitar la extensión del yacimiento, ayudar en la decisión de la estrategia de perforación a seguir, monitorear la producción del pozo, entre otras tareas importantes.

De igual manera, el rol tradicional de los registros de pozo se ha centrado en la evaluación de formaciones y en la evaluación de terminaciones de pozo (Ellis, 1987). La evaluación de formaciones se apoya de los registros geofísicos de pozo para identificar la presencia de hidrocarburos en las formaciones que tienen contacto con el pozo, y aún más importante, para determinar la profundidad en la que éstos se encuentran. En este contexto, la propiedad indicadora más importante es la porosidad, definida como el porcentaje del volumen total de la roca que puede almacenar fluidos, la cual permite cuantificar la fracción de hidrocarburos presente en la matriz de la roca.

Por otra parte, en el contexto de la evaluación de terminaciones, los registros geofísicos son utilizados para diversas cuestiones, tales como la determinación de la calidad de la cementación, corrosión de la tubería, mediciones de presión, etc.(Ellis, 1987)

3.4. Conceptos fundamentales

El objetivo principal de este trabajo es el de analizar la efectividad de redes neuronales basadas en mecanismos de recurrencia para clasificar litofacies en registros geofísicos de pozo. En el contexto de los registros de pozo, es importante definir algunos conceptos fundamentales para comprender mejor este objetivo, de manera que en la sección de Resultados podamos avocarnos a analizar exclusivamente el desempeño de los modelos utilizados para clasificar litofacies teniendo claro qué es aquello que nuestros modelos clasifican.

3.4.1. Estrato

Un **estrato** es una capa tabular o lenticular de rocas sedimentarias que tiene una unicidad litológica, textural o estructural que lo distingue claramente de los estratos adyacentes (Boggs, 2006). Las superficies superior e inferior de un estrato representan los planos del mismo, y pueden o no tener una relación geométrica paralela (Figura 3.1), lo anterior depende de las condiciones de deposición de los sedimentos que conforman al estrato, dichos planos se generan debido a eventos de no-depósito o a superficies de erosión.

Figura 3.1: Secuencia de estratos con una relación geométrica paralela. Tomado de https://commons.wikimedia.org/wiki/File:Quebrada_de_Cafayate,_Salta_(Argentina).jpg

3.4.2. Formación

En estratigrafía, la unidad de clasificación fundamental es la **formación**. Se define como un cuerpo de roca que se identifica por sus características litológicas y su posición estratigráfica, puede tener, o no, una geometría tabular y se puede reconocer en el subsuelo o en la superficie (N.A.C.O.S.M., 2005).

Como unidad fundamental de clasificación, el concepto de formación es utilizado para describir la geología de una región (Figura 3.2), ya que presenta una litología aproximadamente homogénea o una heterogeneidad característica que pueda distinguirla de otras unidades rocosas de la región y que por lo tanto sea representativa de dicha región.

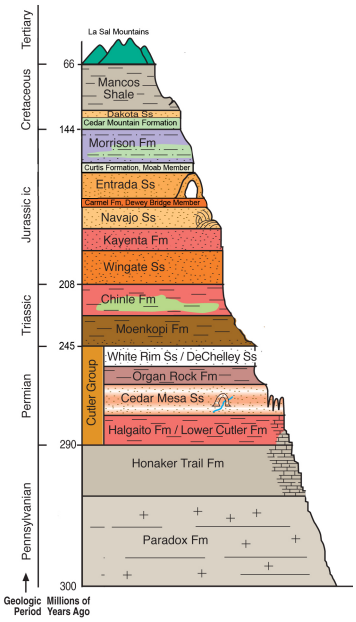


Figura 3.2: Ejemplo de columna estratigráfica dividida por formaciones estratigráficas pertenecientes a la región de Canyonlands, Utah. Tomado de https://cnha.org/visit/arches-national-park/things-to-know/geology/

La importancia de esablecer esta unidad de clasificación para la exploración petrolera radica en que gracias a ella es posible tanto establecer objetivos de exploración dado que se conocen de antemano las características petrofísicas de una formación, así como correlacionar unidades sísmicas o tendencias en los registros geofísicos de pozo con formaciones conocidas en la región de estudio.

3.4.3. Facies

Se define como **facies** a un cuerpo de rocas con características específicas; no confundir con **formación** (Subsección 3.4.2), dado que este concepto es propio de estratigrafía, mientras que facies corresponde a un concepto de sedimentología.

Puede referirse a un solo **estrato** (Subsección 3.4.1) o a una secuencia de estratos, los cuales poseen características en común, sean ciertas litologías identificables o condiciones de deposición similares, que permitan identificar ambientes geológicos antiguos. De igual manera, pueden caracterizarse por contener especies de fósiles distintivos, en dado caso se les especifica como biofacies. De manera cualitativa, las facies pueden identificarse y/o describirse por medio de mecanismos como la textura de la roca, estructuras sedimentarias, color, etc (Reading, 2011).

3.4.4. Litofacies

Este trabajo se aboca a la exploración de nuevas metodologías para clasificar litofacies en registros de pozo, por lo que es necesario definir este concepto. En la sección anterior definimos el concepto de **facies** (Subsección 3.4.3), por lo que ahora será más fácil comprender el de **litofacies**.

En un contexto descriptivo, litofacies es el término utilizado para referirse a cuerpos de rocas sedimentarias que pueden describirse en términos de sus procesos deposicionales y litología (Miall, 2015). A diferencia de las biofacies, para definir una litofacies no tomamos en cuenta el aspecto de los procesos biológicos involucrados para la constitución del cuerpo de roca.

Para simplificar esta definición, tomemos a facies como un concepto general que puede especificarse acorde a los parámetros utilizados para su descripción, modificando así el prefijo que se le antepone a la palabra. En este caso, si sólo tomamos en cuenta los aspectos litológicos como la composición, tamaño de grano, estructuras sedimentarias, etc., estaremos hablando de una **litofacies**.

3.5. Tipos de registros geofísicos de pozo

Si bien hemos mencionado en repetidas ocasiones a los registros geofísicos de pozo, es preciso definir los tipos de registros existentes dado que se utilizaron distintos tipos para los experimentos que se realizaron (véase Capítulo 6).

Debido a que el objetivo de este trabajo no es el de realizar un análisis extensivo de la teoría y las aplicaciones correspondientes a todos los tipos de registros existentes, nos limitaremos a definir aquellos registros que se encuentran presentes en el conjunto de datos de prueba (véase Sección 6.2) y que el autor considera son fundamentales para ejemplificar los distintos principios de funcionamiento de estas herramientas.

3.5.1. Registro de rayos gamma

El registro de rayos gamma mide la radioactividad natural de las formaciones y se utiliza principalmente para identificación de litologías y para establecer correlaciones, pero una de sus aplicaciones más importantes es la de determinar el volumen de arcilla, V_{Sh}, presente en la formación.

En una carta de registros, la curva del registro de rayos gamma se presenta en el primer carril, cuya escala normalmente tiene un rango entre 0 y 150 unidades API (American Petroleum Institute), dicha unidad representa el valor de radioactividad de la formación con respecto a un estándar de radioactividad correspondiente a un bloque de concreto artificialmente radioactivo que se encuentra en The University of Houston, Texas y que cuenta con 200 unidades API de radioactividad (Schlumberger, 2022).

El principio de funcionamiento de la herramienta se basa en que las rocas exhiben un nivel natural de radioactividad dado que contienen una amplia variedad de minerales radioactivos, en rocas sedimentarias los de mayor abundancia son aquellos correspondientes a la serie del uranio (U), torio (Th) y potasio-40 (K-40) (Bassiouni, 1994), el decaimiento radioactivo emitido por estos elementos es medido por la herramienta. Estos minerales radioactivos comúnmente se encuentran en lutitas, mientras que las areniscas *limpias* y carbonatos tienen muy poca presencia de estos minerales y por lo tanto la respuesta del registro de rayos gamma para estas litologías tendrá una respuesta mucho menor comparada a las lutitas, como se puede observar en la Figura 3.3.

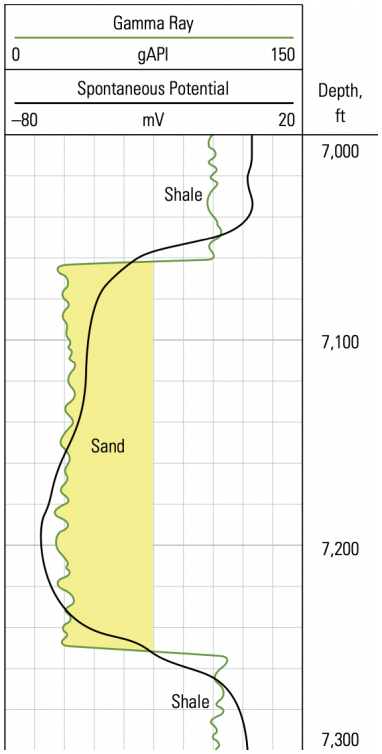


Figura 3.3: Ejemplo de un registro de rayos gamma (verde). Se puede observar el contraste en la respuesta del registro en la presencia de lutitas, teniendo un valor mayor en grados API, con respecto a areniscas limpias, las cuales presentan un valor menor en grados API. Por otra parte, el registro SP (negro), diferencia lutitas de areniscas limpias a partir de la medición del potencial espontáneo, a partir del cual es posible diferenciar formaciones permeables de no permeables (Varhaug, 2016).

La herramienta más sencilla de rayos gamma registra una cuenta total de radioactividad que se representa en una curva, por otra parte, en zonas donde se pueden encontrar feldespatos potásicos, micas, glauconitas y carbonatos en general, es posible correr una herramienta especial, llamada *Espectrometría de Rayos Gamma* que pueda registrar la contribución de cada elemento radioactivo (K, Th, U) en partes-por-millón (ppm) (Figura 3.4).

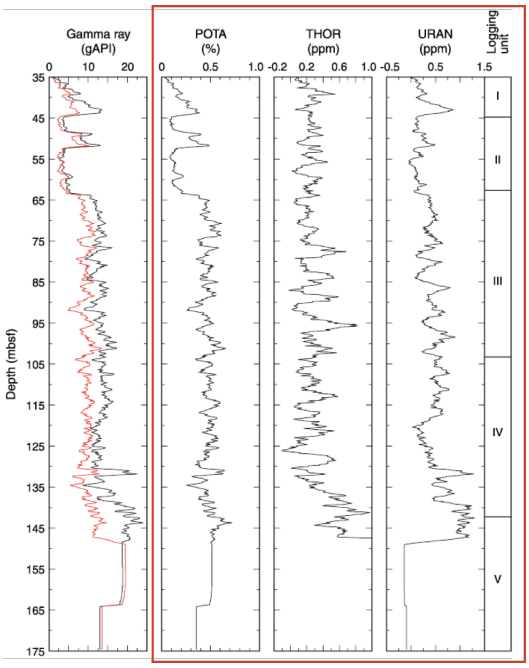


Figura 3.4: Registros de rayos gamma. En el primer carril a la izquierda se observa el registro de rayos gamma sencillo, mientras que encerrados en el recuadro rojo se observan los registros de espectrometría. Tomado de http://www-odp.tamu.edu/publications/200_IR/chap_04/c4_f83.html.

Una de las aplicaciones principales del registro de rayos gamma es el del cálculo del volumen de arcilla en una formación (V_{Sh}). Para calcularlo es necesario obtener el índice de arcilla (I_{Sh}), dado por la siguiente ecuación:

$$I_{Sh} = \frac{Gr - Gr_{min}}{Gr_{max} - Gr_{min}} \tag{3.1}$$

donde:

$$\begin{aligned} Gr &= \text{Medición del registro de rayos gamma a la profundidad seleccionada} \\ Gr_{min} &= \text{Valor mínimo en el registro de rayos gamma} \\ Gr_{max} &= \text{Valor máximo en el registro de rayos gamma} \end{aligned}$$

Para un registro de rayos gamma con respuesta lineal se asume que $I_{Sh} = VSh$. Comúnmente un registro de rayos gamma tiene una respuesta no lineal con base en el área geográfica donde se toma el registro o en la edad de la formación, es por ello que se han desarrollado relaciones para calcular V_{Sh} que toman en cuenta la edad de la formación para obtener valores más precisos (Asquith y Gibson, 1982), por ejemplo:

Larionov (1969), rocas Terciarias:

$$V_{Sh} = 0{,}083 * (2^{3{,}7 * I_{Sh}} - 1) \tag{3.2}$$

Steiber (1970), rocas del Cretácico Superior:

$$V_{Sh} = \frac{I_{Sh}}{3 - 2 * I_{Sh}} \tag{3.3}$$

Clavier (1971), rocas del Cretácico Inferior:

$$V_{Sh} = 1{,}7 * [3{,}38 * (I_{Sh} + 0{,}7)^2]^{0{,}5} \tag{3.4}$$

Larionov (1969), rocas más antiguas:

$$V_{Sh} = 0{,}33 * [2^{2*I_{Sh}} - 1] \tag{3.5}$$

3.5.2. Registro de potencial espontáneo (SP)

En la naturaleza ocurren potenciales electricos asociados a la actividad mineral de las rocas, cambios en las propiedades geológicas de las rocas, así como por la interaccion entre fluidos subterráneos (Bassiouni, 1994).Se le conoce como potencial espontáneo a aquel que se produce sin la necesidad de una fuente artificial de corriente. En el contexto de la exploración petrolera, es aquel que se genera gracias al contacto entre un fluido de pergoración conductor y la formación rocosa.

Una de las primeras mediciones utilizadas para la exploración de hidrocarburos fue la del potencial espontáneo (SP). El antecedente más famoso es el del primer registro geofísico del cual se tiene conocimiento, realizado por los hermanos Schlumberger y Henri Doll en Pechelbronn, Francia en 1927 (Ellis, 1987). En aquella ocasión, se midió la diferencia de potencial producida entre dos electrodos de potencial cuando no se emitía corriente, generando así un gradiente de potencial que variaba con la profundidad.

La herramienta utilizada para medir el potencial espontáneo (Figura 3.5) consiste en un circuito formado por un electrodo que viaja a profundidad a través del pozo, un electrodo fijo en superficie, un voltímetro, un circuito convertidor-reductor compuesto de baterías y un resistor (Bassiouni, 1994).

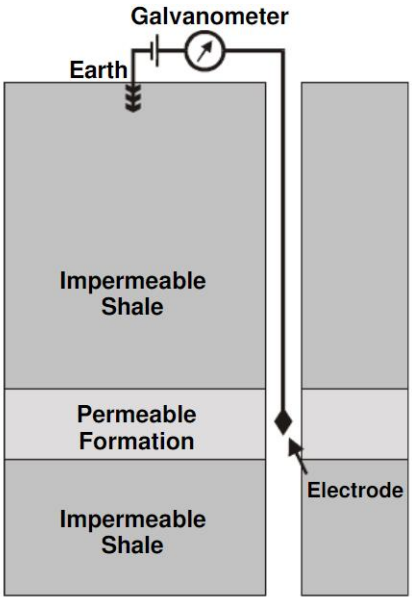


Figura 3.5: Esquema simplificado de una herramienta para medir potencial espontáneo en un pozo. Tomado de https://petrophysicsblogcom.wordpress.com/2018/09/07/sp-log/.

La herramienta de potencial espontáneo no mide directamente el valor de potencial si no la diferencia de potencial en milivoltios (mV) medida entre un electrodo fijo en superficie y uno móvil a profundidad en el pozo.

El potencial medido es de origen electroquímico y electrocinético, siendo la componente electroquímica la de mayor contribución al potencial espontáneo.(Asquith y Gibson, 1982). La Ecua-

ción 3.6 representa una relación que ejemplifica las componentes del potencial espontáneo:

$$SP = E_{eq} + E_{ec} \tag{3.6}$$

donde:

$$SP = \text{Potencial espontáneo}$$
$$E_{eq} = \text{Potencial electroquímico}$$
$$E_{ec} = \text{Potencial electrocinético}$$

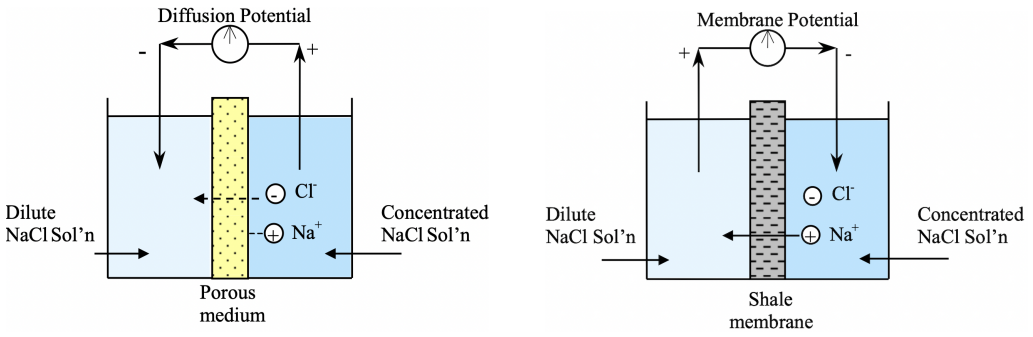
A su vez, el potencial electroquímico E_{eq} esta compuesto por el potencial de difusión E_d y el de membrana E_m, por lo que la Ecuación 3.6 se puede reescribir como:

$$SP = (E_d + E_m) + E_{ec} \tag{3.7}$$

donde:

$$SP = \text{Potencial espontáneo}$$
$$E_d = \text{Potencial electroquímico de difusión}$$
$$E_m = \text{Potencial electroquímico de membrana}$$
$$E_{ec} = \text{Potencial electrocinético}$$

El potencial de difusión ocurre cuando dos electrolitos Na^+ y Cl^- con actividad química distinta en una solución $NaCl$ se separan por un medio poroso (Figura 3.6a), de tal manera que los iones positivos y negativos atraviesan la membrana porosa para concentrarse en la solución de $NaCl$ más diluida y se carga negativamente, generando un potencial de difusion E_d.



(a) Potencial de difusión (b) Potencial de membrana

Figura 3.6: Representaciones esquemáticas de las distintas componentes del potencial electroquímico, donde (a) representa al potencial de difusión y (b) al de membrana. Tomado de https://faculty.ksu.edu.sa/sites/default/files/sp-lab.pdf.

El potencial de membrana E_m se genera cuando dos electrolitos Na^+ y Cl^- de concentraciones distintas se separan por un medio poco permeable (Figura 3.6b). En este caso, los iones tratan de ir de la solución concentrada a la diluida, pero al estar cargada negativamente la membrana, los iones de Cl^- ven restringido su paso, pero los iones positivos de Na^+ pueden adherirse a los poros libremente y se genera un potencial E_m en la membrana.

En cuanto a la componente electrocinética E_{ec} del potencial espontáneo, esta se produce cuando el lodo de perforación se filtra en la formación. En el enjarre de lodo se genera una diferencia

de potencial debido a la diferencia en concentración de iones entre el agua ligada a la arcilla y el agua libre.

Generalmente, el registro SP es utilizado para diferenciar zonas permeables de zonas no permeables, distinguiendo así a las lutitas impermeables de areniscas permeables, delimitando sus espesores. Usualmente se presenta en el primer carril de una carta de registros (Figura 3.3) y a partir de él es posible determinar la resistividad del agua de formación R_w.

3.5.3. Registros de resistividad

Este tipo de registros basan su principio de medición en dos propiedades físicas principales. La primera es la resistividad, la cual representa la intensidad con la que un material se opone al flujo de corriente electrica a través de él, y por otra parte, la conductividad, que es una propiedad inversa a la resistividad y mide la facilidad que proporciona un material al flujo de corriente a través de él (Bjorlykke, 2015).

Las formaciones porosas de roca que contienen agua salada, tendrán una baja resistividad en unidades de ohm-m ($\Omega \cdot m$), por otra parte, aquellas formaciones que albergen hidrocarburos en sus poros o que cuenten con una baja resistividad tendrán un valor alto de resistividad, como puede observarse en la Figura 3.7.

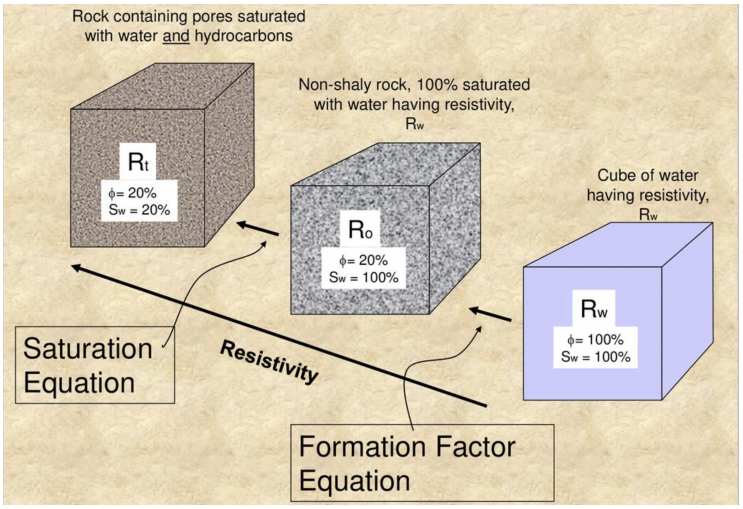


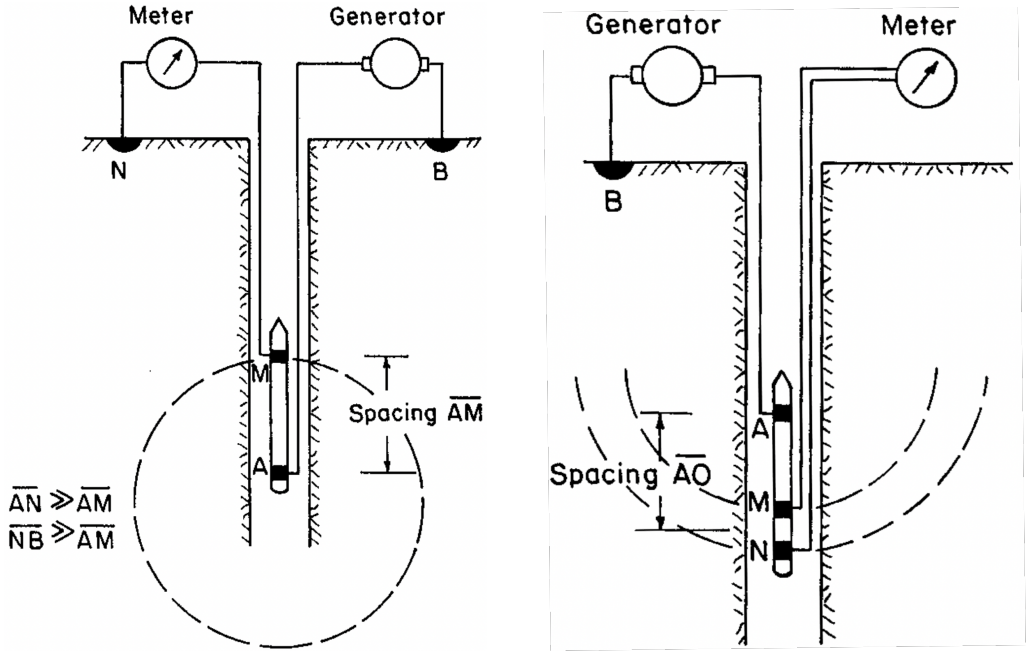
Figura 3.7: Tendencia de la resistividad eléctrica en formaciones y fluidos. Tomado de (Bjorlykke, 2015)

El uso más importante de los registros de resistividad es el de distinguir aquellas formaciones almacenadoras de hidrocarburos de aquellas que almacenan agua en su mayoría (Asquith y Gibson, 1982). La forma de *correr* estos registros consiste en producir una corriente en la formación adyacente a la herramienta y medir la respuesta de la formación a dicha corriente, para obtener información de las características conductivas de la roca e inferir la litología correspondiente y si se encuentra, o no, saturada de fluidos. Esta corriente se puede producir por medio de dos métodos principales, sean herramientas **electródicas** que emiten una corriente directa a través de la formación, o herramientas basadas en **inducción**, que inducen una corriente para medir la conductividad de la formación.

Las herramientas **electródicas** consisten en arreglos de electrodos que comúnmente se distribuyen en electrodos que van adheridos a la sonda a través del pozo y el resto que se fijan en la superficie, este tipo de herramientas requieren de un lodo conductor que se infiltre en la

formación para poder conducir la corriente eléctrica.

Se pueden tener distintos arreglos de herramientas electródicas que se usan dependiendo de los espesores de las formaciones que se desean analizar. Tal es el caso de los arreglos normales y laterales (Bassiouni, 1994). El arreglo normal (Figura 3.8a)consiste en cuatro electrodos, de los cuales un electrodo de corriente y otro de potencial se encuentran en la sonda que viaja a través del pozo y los dos restantes se encuentran en la superficie. Este tipo de arreglo se subdivide acorde al radio de investigación en *corto* y *largo*, siendo el arreglo normal largo para un radio de investigación máximo de $10ft$. En el caso del arreglo lateral (Figura 3.8b), se colocan ambos electrodos de potencial y uno de corriente en la sonda, mientras que un electrodo de corriente se mantiene en la superficie. Este tipo de herramienta alcanza una profundida de investigación de $19ft$.



(a) Potencial de difusión (b) Potencial de membrana

Figura 3.8: Tipos de arreglos electródicos en herramientas para medir resistividad en el pozo. El arreglo de tipo normal (a) tiene una profundidad de investigación menor al arreglo lateral (b). Tomado de (Bassiouni, 1994)

A diferencia de los registros electródicos que miden la resistividad, los registros de resistividad **inductivos** miden la conductividad de la formación, es decir, la facilidad con la que la corriente inducida viaja a través del espacio poroso de las rocas. Una herramienta de inducción (Figura 3.9) está conformadad por bobinas transmisoras que emiten corriente alterna de alta frecuencia a intensidad constante. El campo electromagnético que se genera induce corriente en la formación, esta corriente fluye de manera perpendicular a la herramienta hacia la formación y crean campos electromagnéticos secundarios cuya señal es recibida por la bobina receptora y la intensidad de esta señal es proporcional a la conductividad de la formación (Asquith y Gibson, 1982).

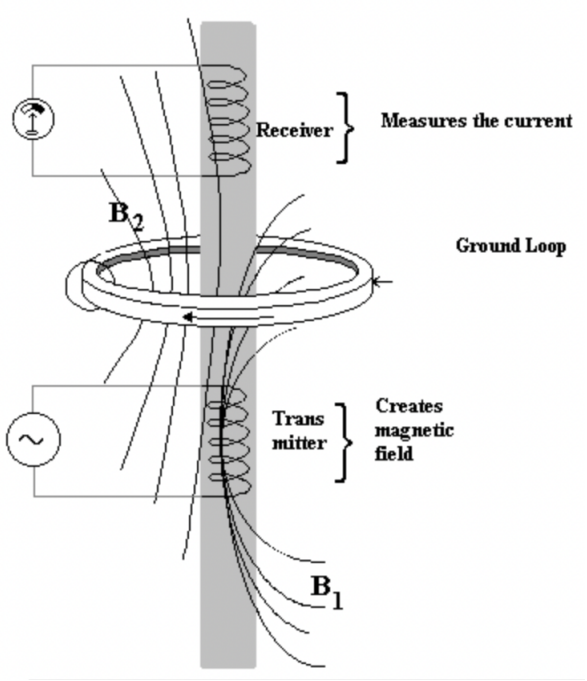


Figura 3.9: Descripción esquemática del principio de funcionamiento de una herramienta de inducción. Tomado de http://www.bridge7.com/grand/log/gen/resistivity/induction.html

Entre las herramientas de inducción más utilizadas se encuentra el registro de doble inducción, el cual registra dos mediciones de resistividad a distintas profundidades, una curva de inducción profunda (ILD) y otra curva de inducción media (ILM), aunque conmunmente este registro se *corre* en conjunto con algún registro de investigación somera, como el Laterolog somero.

3.5.4. Registros de porosidad

La porosidad es una propiedad muy importante para, en conjunto con otras propiedades, determinar el potencial de almacenamiento y explotación de hidrocarburos en una formación. A diferencia de otras propiedades como la resistividad, no es posible medir la porosidad directamente a partir de un registro geofísico de pozo, aunque determinarla a apartir de otras propiedades relacionadas ha sido objeto de estudio a lo largo de la historia en el desarrollo de herramientas de registros de pozo. Entre las herramientas principales para utilizadas para obtener porosidad indirectamente destacan el **registro de porosidad de neutrón**, **registro de densidad** y el **registro sónico**.

Registro de porosidad de neutrón

La herramienta de porosidad de neutrón (Figura 3.10) mide la concentración de iones de hidrógeno en la formación, en una formación limpia se utiliza para obtener el porcentaje de espacio poroso llenado por agua o aceite, por medio de la emisión contínua de neutrones de una fuente compuesta por americio y berilio hacia la formación.

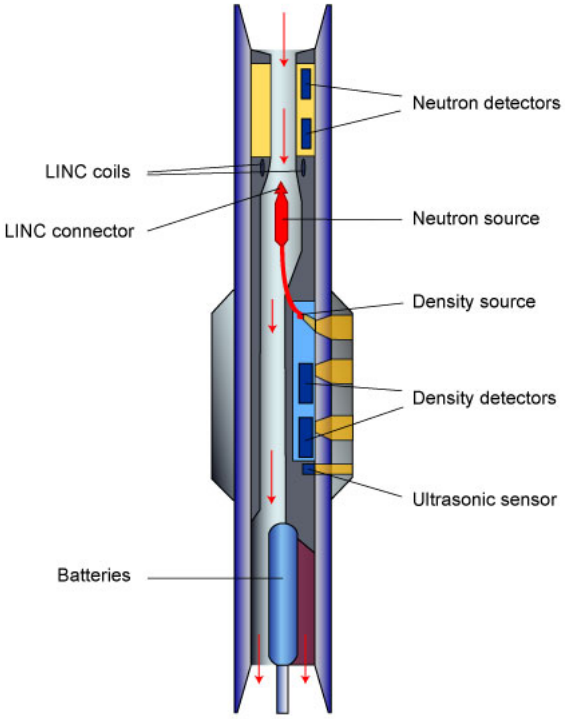


Figura 3.10: Herramienta utilizada para obtener el registro de porosidad de neutrón. Tomado de https://mlp.ldeo.columbia.edu/logdb/technology/schlumberger-lwd-tools/logging-while-drilling-adnvision-tool/

El principio de funcionamiento de la herramienta consiste en que los neutrones producidos por la fuente colisionan con los núcleos de los elementos presentes en la formación y se mide el nivel de decaimiento en la energía de los neutrones que son recibidos por los detectores de la herramienta.

El máximo decaimiento de energía resulta cuando chocan los neutrones con un núcleo de hidrógeno, por lo que al obtener un gran decaimiento de la energía del neutrón recibidp significa que en la formación existe una gran concentración de hidrógeno.

Se asume que los poros de la roca almacenan fluidos, por lo que el contenido de hidrógeno de la roca se asocia al contenido de fluidos en los poros y por ende, este nivel de hidrógeno se correlaciona a la porosidad de la formación.

Cuando se tiene presencia de gas en la formación, existe una menor concentración de hidrógeno en el espacio poroso, por lo que la lectura del decaimiento de energía en la herramienta será menor, a este fenómeno se le conoce como *efecto del gas* (Bjorlykke, 2015).

Registro de densidad

La herramienta de porosidad de neutrón mide el decaimiento de la energía de los neutrones gracias a la interacción con los elementos de la formación, similarmente, la herramienta de densidad mide la densidad de los electrones en la formación, gracias a la interacción de los rayos gamma producidos por una fuente de Cobalto-60 y Cesio-137 con los electrones, que produce una pérdida de energía en la partícula de rayos gamma.

A esta pérdida de energía debida a la colisión de partículas de rayos gamma con electrones se le conoce como *efecto Compton*. La reducción total de la energía en los rayos gamma es pro-

porcional a la densidad de electrones en la formación y la densidad de neutrones se relaciona directamente con la densidad aparente de la formación (*bulk density*).

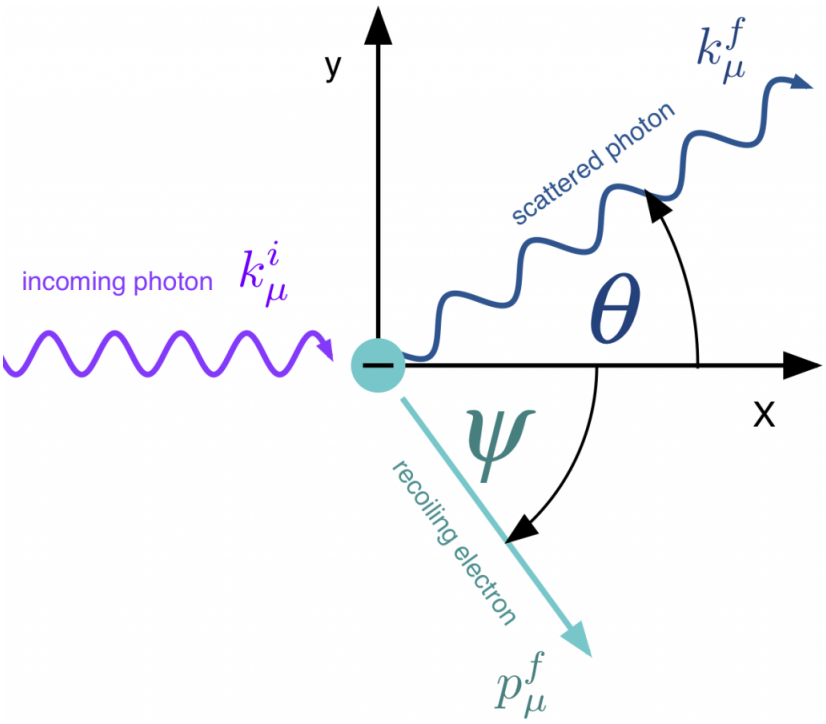


Figura 3.11: Efecto Compton (*Compton Scattering*), en el que la colisión de un fotón con un electrón propicia la pérdida de energía y cambio de trayectoria del fotón. Tomado de https://physicsopenlab.org/2016/02/04/compton-scattering-2/

La densidad aparente de la formación es función de la densidad de la matriz, porosidad y densidad del fluido en los poros y está dada por la Ecuación 3.8:

$$\phi_{den} = \frac{\rho_{ma} - \rho_b}{\rho_{ma} - \rho_f} \tag{3.8}$$

donde:

$$\begin{aligned} \phi_{den} &= \text{Porosidad densidad} \\ \rho_{ma} &= \text{Densidad de la matriz} \\ \rho_b &= \text{Densidad aparente de la formación} \\ \rho_f &= \text{Densidad del fluido} \end{aligned}$$

Registro sónico

El registro sónico mide Mide el intervalo de tiempo de tránsito de una onda compresional con respecto a una unidad de longitud en pies a través de la formación para obtener información indirecta sobre la porosidad de la misma.

Una herramienta compuesta por transmisores sónicos y receptores (Figura 3.12) emite pulsos acústicos que viajan a través de la roca y regresa a la unidad receptora de la herramienta, registrando la velocidad de onda compresional de la roca para poder estimar la presencia de fluidos en la roca y correlacionar litologías.

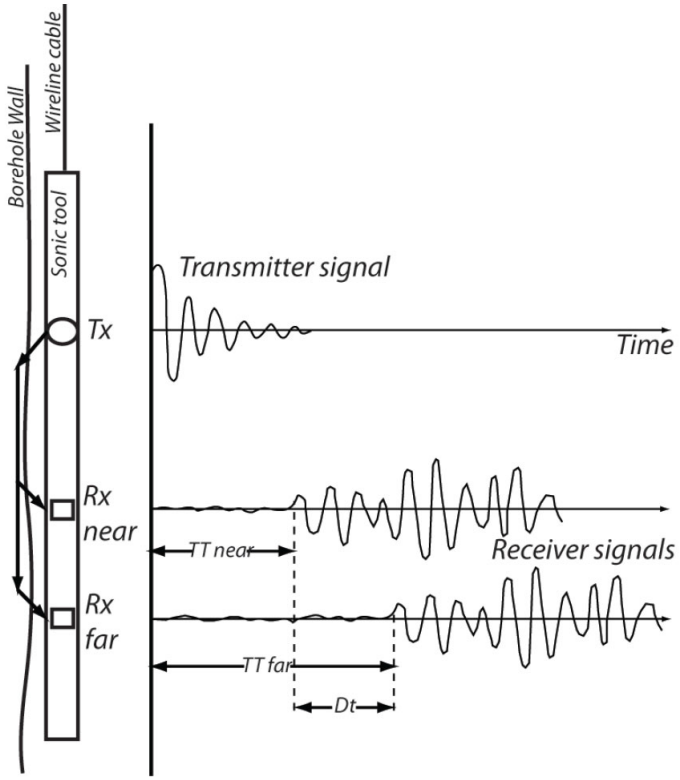


Figura 3.12: Herramienta utilizada para obtener el registro sónico, compuesta por un transmisor y dos receptores. Tomado de https://csegrecorder.com/articles/view/the-sound-of-sonic-a-historical-perspective-and-intro-to-acoustic-logging

El intervalo de tiempo de tránsito se mide normalmente en unidades de microsegundos (μ_s) sobre pies (ft) y es el recíproco de la velocidad de onda compresional. Este intervalo de tiempo depende de la litología y porosidad de la formación y está dado por la (Ecuación 3.11):

$$\phi_{son} = \frac{\Delta t_{log} - \Delta t_{ma}}{\Delta t_f - \Delta t_{ma}} \tag{3.9}$$

donde:

$$\begin{aligned} \phi_{son} &= \text{Porosidad sónica} \\ \Delta t_{log} &= \text{Intervalo de tiempo de tránsito de la formación} \\ \Delta t_{ma} &= \text{Intervalo de tiempo de tránsito de la matriz} \\ \Delta t_f &= \text{Intervalo de tiempo de tránsito del fluido} \end{aligned}$$

El intervalo de tiempo de tránsito de una formación se incrementa debido a la presencia de hidrocarburos en la formación (Asquith y Gibson, 1982), por lo que comúnmente se trata de corregir este efecto (*efecto de hidrocarburos*) utilizando la corrección empírica de Hilchie (1978):

$$\phi = \phi_{son} * 0{,}7 \ (gas) \tag{3.10}$$

$$\phi = \phi_{son} * 0{,}9 \ (aceite) \tag{3.11}$$

Capítulo 4

Aprendizaje automático

Desde finales de la década de los 2010´s y principios de la década actual, términos como **Inteligencia Artificial**, **Machine Learning**, **Redes Neuronales**, entre otros tantos, se han popularizado gracias a la atención mediática que se les ha brindado a los desarrollos del campo de la Inteligencia Artificial (IA) y ahora forman parte de las conversaciones que sostienen día con día científicos, ingenieros, empresarios, políticos y la sociedad en general. La mayor parte del tiempo, sin siquiera notarlo consumimos productos y contenido que han sido desarrollados y funcionan con base en una serie de algoritmos de IA y que se adaptan a nuestras necesidades siguiendo las recomendaciones que otros algoritmos hacen.

Constantemente surgen nuevas noticias sobre algoritmos que *aprendieron* a realizar tareas humanas, sean actividades sencillas como: ver, escuchar, hablar, etc., o bien, llevar a cabo tareas que rozan los límites de la capacidad humana, como vencer al campeón mundial del juego de Go (Silver et al., 2016), escribir automáticamente código de programación a partir de una entrada de texto (Li et al., 2022) o controlar magnéticamente un reactor de plasma de fusión nuclear (Degrave et al., 2022).

Curiosamente, son las tareas más complejas aquellas en las que la inteligencia artificial ha demostrado poder alcanzar un nivel experto, a la par de los humanos más capacitados o incluso superándolos con creces (como en el caso de (Silver et al., 2016)). Por otra parte, son las actividades humanas comúnes donde las *máquinas inteligentes* no han siquiera alcanzado al ser humano promedio; conducir un automóvil, reconocer objetos, conversar, describir una imagen, o simplemente caminar.

La razón principal del por qué es tan complicado de *entrenar* a un sistema inteligente para realizar actividades básicas humanas es que es muy difícil formalizar matemáticamente a través de reglas estas capacidades humanas que para nosotros resultan intuitivas e incluso automáticas, pero que requieren una profunda comprensión y conocimiento del mundo. Sin embargo, cuando se trata de problemas o fenómenos complejos, tenemos reglas y modelos matemáticos que aproximan el comportamiento de estos sistemas complejos; es decir, tenemos más herramientas matemáticas para enseñar a una máquina a identificar fallas geológicas en un cubo sísmico, que para enseñarla a escribir un cuento.

4.1. Aprendizaje de máquina

(Rzóska, 1953) realizó un estudio sobre la manera en que las ratas logran diferenciar trozos de comida envenenada de trozos comestibles. Dicho estudio menciona que cuando las ratas encuentran comida que les llame la atención, empiezan a comer bocados muy pequeños, y dependiendo del sabor de la comida y sus efectos fisiológicos determinan si seguir comiendo, o no. Si la comida

produce un efecto negativo, las ratas asocian esa comida en particular con dicho efecto negativo y determinan no volver a comerla.

Si analizamos este patrón de conducta en las ratas, podemos identificar que existe un mecanismo de aprendizaje en el que estos animales utilizan experiencias pasadas ingiriendo estos alimentos para obtener conocimiento que robustezca su capacidad de identificar comida *segura*, ya que estos animales infieren que si dicho alimento tuvo un efecto negativo en el pasado también lo tendrá en el futuro.

El ejemplo anterior es una muestra de un mecanismo de aprendizaje particular que permite a un sistema, en este caso una rata, utilizar el conocimiento obtenido en experiencias previas para tomar decisiones en el futuro. Una rata es un *agente*, incapaz de realizar actividades equiparables a las de un ser humano, y a su vez, en cuanto a capacidad de cómputo, un ser humano no es capaz de realizar con la misma eficiencia las tareas que realiza una máquina.

Cuando se piensa en algoritmos, inmediatamente imaginamos una serie de pasos estructurados en cierta manera para obtener una salida o producto; en computación, un algoritmo es un proceso compuesto por una serie de instrucciones definidas que pueden recibir entradas y que culminan en una salida. Si trasladamos un algoritmo a cualquier lenguaje de programación que sea *compilable* por una computadora podemos aprovechar su capacidad de cómputo para obtener una salida deseada de una manera considerablemente más rápida y eficiente que un ser humano.

La manera clásica de ejecutar un algoritmo es traduciéndolo a lenguaje de máquina en un programa finito, es decir, cuya estructura tiene principio y final fijos, de tal manera que no se pueden modificar en tiempo de ejecución. Este paradigma de programación es útil cuando conocemos una serie de reglas e instrucciones estáticas que permiten llevar a cabo la tarea de interés.

Regresando al ejemplo del estudio de (Rzóska, 1953), surge la pregunta: ¿Cómo programar el mecanismo de aprendizaje de una rata de manera que podamos simular el comportamiento de este agente?. Si tomamos en cuenta el paradigma de programación clásico tendríamos que programar una abstracción de la *memoria* del agente que vaya creciendo en cada iteración, o bien, programar un conjunto de reglas que vaya aumentando a medida que el agente observa nuevas situaciones. Cualquiera de esas dos opciones carecen de eficiencia computacional, ya que para N iteraciones el tamaño de la memoria también crecerá N veces y en el caso de la estrategia de programar manualmente un conjunto de reglas será necesario re-compilar y re-programar tantas veces sea necesario para robustecer el algoritmo.

Tomando en cuenta las limitaciones del paradigma de programación clásico, si se quiere simular este mecanismo de aprendizaje, es necesario programar a la computadora para que sea capaz de aprender de la experiencia y extraer patrones que permitan describir un fenómeno a partir de conceptos simples y así poder **generalizar**, de tal manera que sea posible reconstruir conceptos complejos a partir de estos conceptos más generales; a este paradigma computacional se le conoce como **aprendizaje de máquina** o **aprendizaje automático** (Goodfellow et al., 2016).

Acorde a (Shalev-Shwartz y Ben-David, 2014), existen dos aspectos que motivan a utilizar programas que aprenden a partir de la experiencia: la complejidad del problema y la necesidad de adaptabilidad.

Los mismos autores señalan que la complejidad de un problema suele deberse a dos factores fundamentales. El primero es la incapacidad de aterrizar dicho problema o fenómeno a un programa debido al poco conocimiento del mismo; esto ocurre comúnmente al momento de intentar

discretizar actividades humanas naturales, como reconocer voces humanas o entender una imagen, gracias a que nuestro conocimiento de cómo realizamos dichas actividades no está bien definido. Por otra parte, el otro factor de complejidad se debe a problemas que van más allá de las capacidades humanas, en los que para el ser humano es muy complicado o tardado reconocer patrones; como elaborar sistemas de recomendación o extraer información en bases de datos extensas.

Es necesario elaborar programas adaptables cuando los datos de entrada cambian a lo largo del tiempo y se requieren algoritmos lo suficientemente robustos para seguir funcionando sin importar estos cambios, como en el ejemplo anterior, en el que las situaciones cambian para el agente dependiendo de su interacción con el entorno. Otros ejemplos de aplicación de estos programas pueden ser el reconocimiento facial o la detección de e-mails *spam*.

4.2. Una definición de aprendizaje

El concepto de *aprender* tiene distintas acepciones dependiendo el contexto de estudio, para efectos de este trabajo nos basaremos en la definición de aprendizaje en el contexto de la computación. Una definición interesante es aquella propuesta por (Mitchell, 1997):

"*Se dice que un programa de computadora* ***aprende*** *de la experiencia E con respecto a cierto conjunto de tareas T y una medida de rendimento R, si su rendimiento en las tareas T, medido por R, mejora con la experiencia E*"

(Goodfellow et al., 2016) realiza un análisis de esta definición el cual es utilizado como base de esta sección. En él menciona que el aprendizaje, no es la tarea T del aprendizaje de máquina, si no un medio para obtener la capacidad de realizar dicha tarea. En el contexto de este trabajo en particular, utilizamos el aprendizaje de máquina para clasificar litofacies en registros geofísicos de pozo, siendo entonces la tarea T un problema de **clasificación**.

Clasificar, constituye una de las aplicaciones principales del aprendizaje de máquina. Se trata de aprender una función f que mapee un conjunto de entradas o **características** $x \in \mathcal{X}$ a un conjunto de salidas finito, no ordenado y mutuamente excluyente C conocido como **clases**, $\mathcal{Y} = \{1, 2, \cdots, y_C\}$. Si $x \in \mathbb{R}^n$, entonces $f : \mathbb{R}^n \to \{1, \cdots, C\}$, tal que:

$$y = f(\mathbf{x}) \tag{4.1}$$

La Ecuación 4.1 ayuda a definir un problema de clasificación, en el cual el programa asignará una categoría perteneciente a $\mathcal{Y}$ a cada uno de los vectores que pertenezcan a $\mathbf{x}$. En nuestro caso, para cada vector x_i, que representa al conjunto de valores de k registros de pozo pertenecientes al nivel de profundidad i, la función f o el **modelo** clasificador asignará una clase o **etiqueta** y_i asociada a una (y solo una) litofacies.

Siguiendo la definición de (Mitchell, 1997), para poder determinar si el programa de computadora está aprendiendo, es necesario determinar una medida de rendimiento R acorde a la tarea T, la cual nos proporcione un indicador cuantitativo del grado de aprendizaje efectivo del modelo. Existen distintas métricas para cuantificar el rendimiento de un algoritmo, entre las que destacan: exactitud, precisión, exhaustividad, puntaje-F1, entre otras. La que se utiliza comúnmente en problemas de clasificación debido a su simpleza es la exactitud, que no es más que el cociente entre el número de veces que el algoritmo asoció correctamente una etiqueta a una entrada y el número total de entradas.

Como se mencionó en la Sección 4.1, los algoritmos de aprendizaje de máquina parten de la

premisa de ser capaces de encontrar patrones generales en los datos de entrada para poder ser aplicados en entradas no vistas y obtener un rendimiento similar en ellas. A los datos de entrada utilizados para *entrenar* a un algoritmo se le conocen como **datos de entrenamiento**, mientras que a las entradas no vistas se le conocen como **datos de prueba**. Es por ello, que el rendimiento de los algoritmos de aprendizaje de máquina es medido en los datos de prueba, ya que de esta manera es posible determinar la capacidad de generalización de los mismos.

El último elemento por analizar es la experiencia E, esta depende del tipo de algoritmo de aprendizaje de máquina que se utiliza, cuya clasificación más común depende de si se cuenta con un conjunto de etiquetas para todas y cada una de las entradas.

Se dice que un algoritmo de aprendizaje de máquina es **supervisado** (Figura 4.1) si para todo elemento de $\mathcal{X}$ corresponde un elemento de $\mathcal{Y}$ y se puede estimar la probabilidad $p(y|\mathbf{x})$, es decir, que para cada entrada corresponde su respectiva etiqueta, por lo que es posible estimar una probabilidad condicional de la etiqueta predecida dada una entrada.

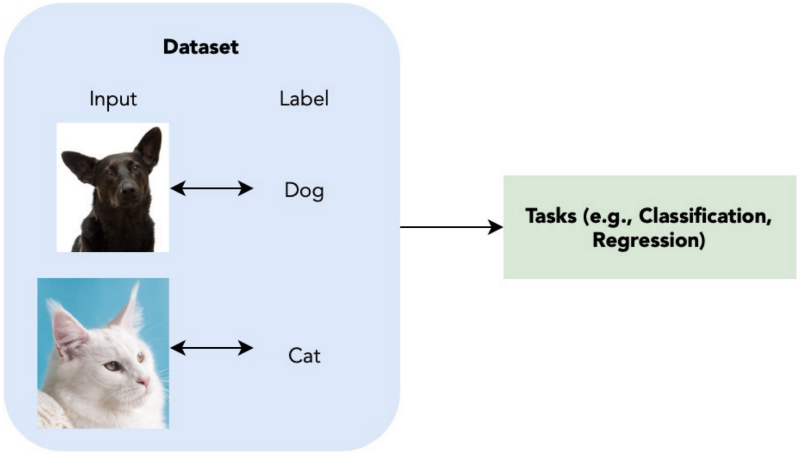


Figura 4.1: Conjunto de datos supervisados. Tomado de https://towardsdatascience.com/supervised-semi-supervised-unsupervised-and-self-supervised-learning-7fa79aa9247c

Cuando un algoritmo de aprendizaje de máquina es **no supervisado** (Figura 4.2) implica que no existe dicho conjunto $\mathcal{Y}$, por lo que el algoritmo se limita a aprender características propias del conjunto de datos para realizar tareas como *clustering*, en las que se agrupan entradas con características similares en un espacio n-dimensional a manera de cúmulos o *clusters*.

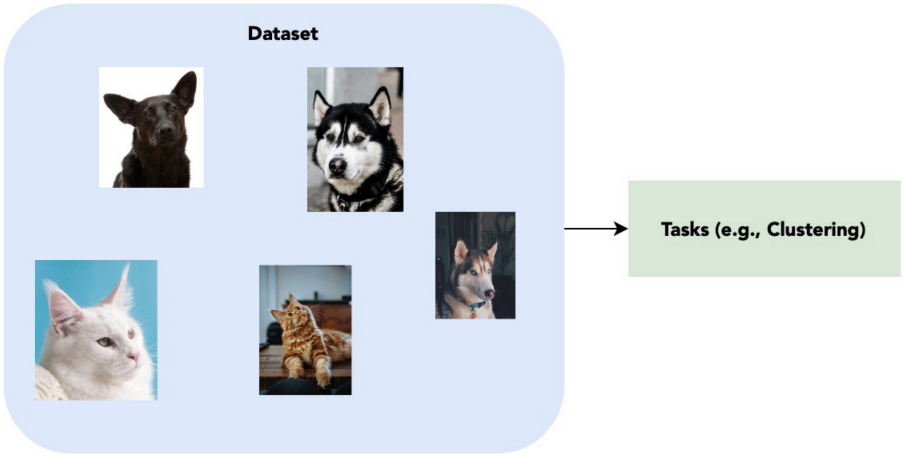


Figura 4.2: Conjunto de datos no supervisados. Tomado de https://towardsdatascience.com/supervised-semi-supervised-unsupervised-and-self-supervised-learning-7fa79aa9247c

4.3. Aprendiendo f

En la sección anterior se mencionó que para resolver un problema de clasifiación como el que nos atiene en este trabajo es necesario aprender una función f que mapee un conjunto de entradas con sus etiquetas correspondientes (Ecuación 4.1). Naturalmente surge la pregunta: ¿Cómo se aprende dicha función f?, cuya solución ha sido el tema de investigación más importante para los investigadores de aprendizaje automático desde mediados de la década pasada hasta hoy en día.

Para la comunidad científica en esta área ha sido de gran ayuda apoyarse en otras disciplinas para lograr descifrar la manera de hacer que una computadora aprenda; disciplinas como la biología y las neurociencias han estudiado desde hace más tiempo la naturaleza del aprendizaje humano y los mecanismos cerebrales involucrados en este fenómeno, los cuales construyeron los cimientos de la ciencia detrás del aprendizaje de máquina.

4.3.1. Modelo biológico

La unidad fundamental del sistema nervioso es la neurona. La estructura principal de estas células es conocida como *soma* y se conecta con otras células por medio del *axón* y las *dendritas*, dichas terminales de conexión se conocen como *sinapsis*. El axón es una estructura alargada que extiende el alcance de la neurona, mientras que las dendritas forman terminaciones que reciben impulsos eléctricos por parte de otras neuronas, formando así una red interconectada entre neuronas mejor conocida como **red neuronal biológica** (Figura 4.3).

El intercambio electroquímico entre células nerviosas propicia la activación de las mismas si la suma de los impulsos eléctricos es lo suficientemente potente para activar a la neurona, de tal manera que transmita la señal recibida a lo largo del axón y genere otras conexiones con otras neuronas. El comportamiento de las neuronas es binario, es decir, se activa o no; de esta

manera nuestro cerebro es capaz de realizar procesos complejos, a partir de una red de unidades biológicas que producen impulsos binarios.

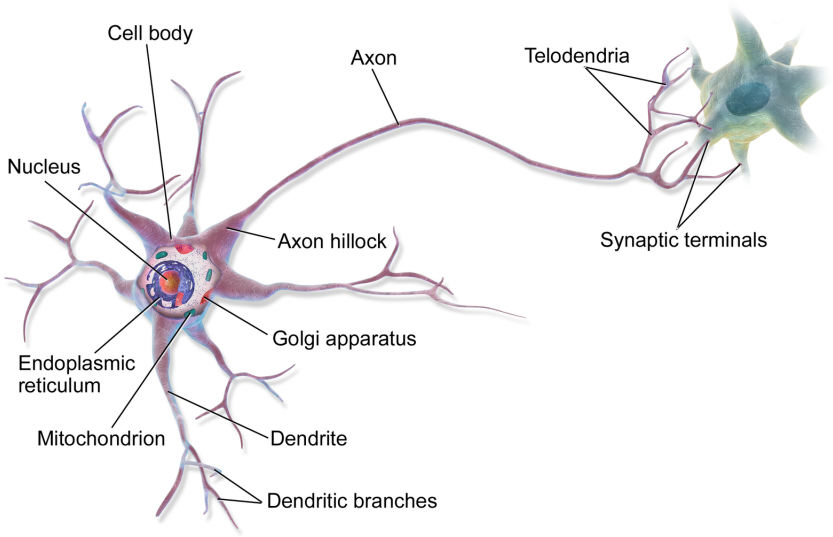


Figura 4.3: Modelo biológico de una neurona. Tomado de https://commons.wikimedia.org/wiki/File:Blausen_0657_MultipolarNeuron.png

4.3.2. Redes neuronales artificiales y el perceptrón de Rosenblatt

Como bien explica el modelo biológico de una red neuronal, el cerebro humano *aprende* mediante las activaciones entre las conexiones de neuronas. En la búsqueda de una metodología para trasladar los principios del aprendizaje humano al cómputo y así lograr replicarlo (Mcculloch y Pitts, 1943) propusieron un modelo lógico basado en dicho modelo biológico para obtener un sistema capaz de clasificar dos categorías de datos distintas y así dar pie al desarrollo de Redes Neuronales Artificiales.

A finales de la década de los 50's, (Rosenblatt, 1958) implementó dicho modelo para clasificar patrones linealmente separables en un sistema computacional llamado *Perceptrón*. El perceptrón se asemeja al funcionamiento de una neurona, dado que recibe un vector de entrada $\mathbf{x} \in \mathbb{R}^n$ y calcula una combinación lineal de estas entradas y un vector de *pesos* $\mathbf{w} \in \mathbb{R}^n$asociados a cada una de las entradas para producir una salida igual a 1 si la combinación resulta mayor a cierto *umbral* o -1 en caso contrario (Mitchell, 1997). La combinación lineal entre las entradas y los pesos está dada por:

$$s = \sum_{i=0}^{m} w_i x_i \tag{4.2}$$

donde m corresponde al número de elementos de entrada. Si expandimos la ecuación anterior:

$$s = w_0 x_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m \tag{4.3}$$

El término x_0 es una constante $x_0 = 1$, por lo que la Ecuación 4.3 puede reescribirse como:

$$s = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m \tag{4.4}$$

En el argot de aprendizaje de máquina al término w_0 se le conoce como *sesgo* o *bias* en inglés, y se denota como b. Así, la Ecuación 4.2 se escribe de manera compacta como:

$$s = b + \sum_{i=1}^{m} w_i x_i \tag{4.5}$$

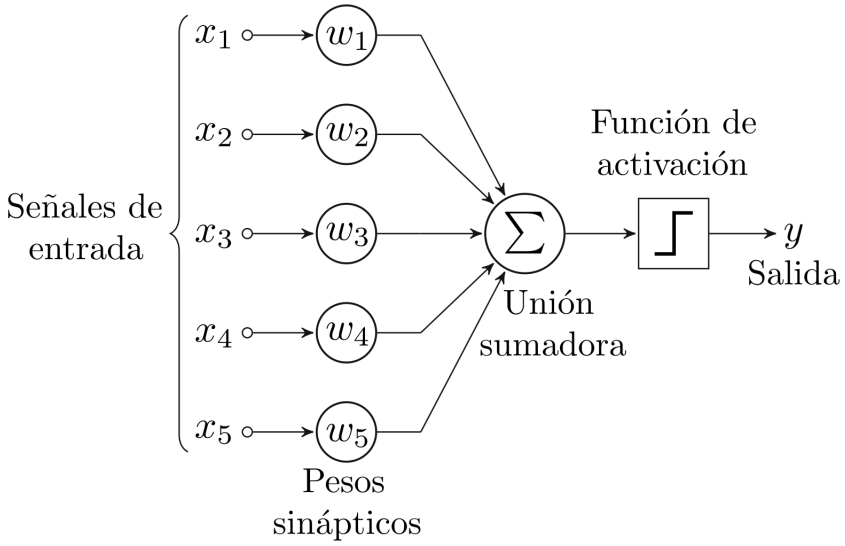


Figura 4.4: Representación gráfica del Perceptrón, utilizado para clasificar conjuntos de datos linealmente separables. Tomado de https://commons.wikimedia.org/wiki/File:Perceptr%C3%B3n_5_unidades.svg

Como se muestra en la Figura 4.4, la combinación lineal entre $\mathbf{x}$ y $\mathbf{w}$ que denotamos como s es la entrada a una *función de activación*. Dicha función es la encargada de binarizar la salida del perceptrón; recordemos que en el modelo biológico neuronal, una neurona solo tiene dos estados: activada o no, por lo tanto dicha función se encarga de *activar* al perceptrón y producir una salida. La salida y del perceptrón está dada por:

$$y = \varphi(s) \tag{4.6}$$

(Rosenblatt, 1958) propuso como función de activación φ a la función escalón o *Heaviside*, por lo tanto, para un umbral t:

$$y = \begin{cases} 1 & si\ s > t \\ -1 & para\ cualquier\ otro\ caso \end{cases} \tag{4.7}$$

Existen otros tipos de funciones de activación φ que son usadas para mantener el valor de la salida de la neurona restringido a un límite requerido. Algunos ejemplos son:

- Signo:

$$y = \begin{cases} -1 & si\ s < 0 \\ 0 & si\ s = 0 \\ 1 & si\ s > 0 \end{cases} \tag{4.8}$$

- Lineal:

$$y = s \tag{4.9}$$

- Sigmoide:

$$y = \frac{e^s}{1 + e^s} \tag{4.10}$$

- ReLu:

$$y = \begin{cases} 0 & si\ s < 0 \\ s & si\ s \geq 0 \end{cases} \tag{4.11}$$

Un perceptrón representa un *hiperplano*, es decir, una superficie de decisión n-dimensional cuya ecuación es $\vec{w} \cdot \vec{x} = 0$ (Mitchell, 1997). En $\mathbb{R}^2$ la superficie de decisión es una recta, mientras que en $\mathbb{R}^3$ será un plano. Aquellos conjuntos de entradas que pueden ser separados por el perceptrón se dice que son linealmente separables, a manera de ejemplo la Figura 4.5 muestra un conjunto de datos en $\mathbb{R}^2$ linealmente separables y la recta que los separa.

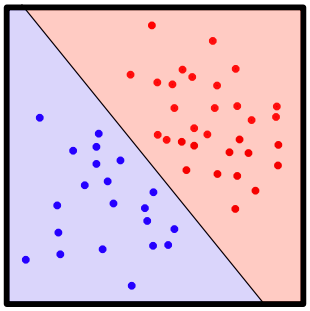


Figura 4.5: Conjunto de datos linealmente separables. La recta de separación es capaz de distinguir claramente dos categorías de datos. Tomado de https://commons.wikimedia.org/wiki/File:Linearly_separable_red-blue_cropped_.svg

4.3.3. Clasificación no lineal y perceptrón multicapa

El ejemplo más simple de clasificación corresponde a clasificadores lineales; aquellos que son utilizados únicamente para conjuntos de datos linealmente separables. En la mayor parte de los casos nos encontramos tratando de clasificar conjuntos de datos que no guardan una relación linealmente separable, por lo que un clasificador lineal es incapaz de categorizarlos utilizando un perceptrón de una sola capa. (Murphy, 2022) muestra un ejemplo muy ilustrativo y valioso para comprender a los clasificadores no lineales e introducir el concepto de perceptrón multicapa:

Sean $\mathbf{x} = (x_1, x_2)$ un vector de entradas y $\phi(\mathbf{x}) = (x_1^2, x_2^2)$ la versión transformada de dicho vector, esta transformación fue elegida convenientemente debido a que el conjunto de datos que se presenta Figura 4.6 no pueden ser separados por una recta pero son visiblemente separables por una región geométrica correspondiente a una circunferencia de radio R.

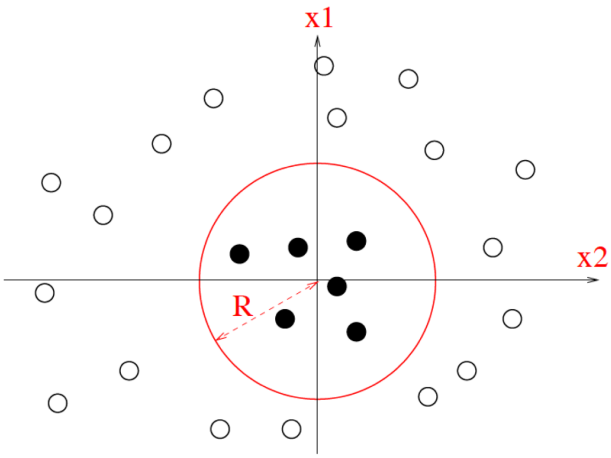


Figura 4.6: Conjunto de datos previo a la transformación. Se observa que no es posible estimar una superficie de separación lineal. Modificado de (Murphy, 2022)

Al transformar las entradas acorde a $\phi(\mathbf{x}) = (x_1^2, x_2^2)$ podemos trabajar con un clasificador lineal gracias a que la transformación permite separar linealmente los datos como se muestra en la Figura 4.7.

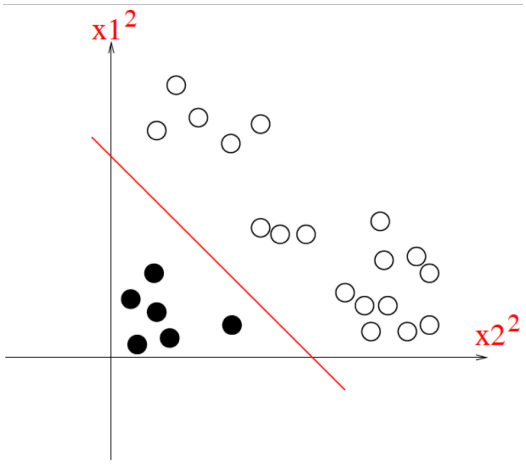


Figura 4.7: Conjunto de datos transformados. Es evidente que un clasificador lineal puede fácilmente categorizar los datos en dos clases distintas. Modificado de (Murphy, 2022)

Si bien (Murphy, 2022) nos presenta que es posible linearizar un problema si realizamos un pre-procesamiento efectivo de los datos, este es un ejemplo ideal ya que difícilmente encontremos una transformación tan directa o incluso puede no ser posible pre-procesar los datos para forzar el uso de un clasificador lineal. Se vuelve necesario utilizar funciones más complejas para estimar un hiperplano de separación.

El concepto del perceptrón simple se extendío al uso de perceptrones multicapa, es decir, perceptrones apilados u ordenados recursivamente para aproximar funciones más complejas. Podemos reescribir la Ecuación 4.6 en su forma vectorial:

$$f(\mathbf{x};\theta) = \mathbf{W}\mathbf{x} + \mathbf{b} \tag{4.12}$$

Donde θ corresponde a los **parámetros** de la función tal que $\theta = (\mathbf{W}, \mathbf{b})$. Si deseamos estimar una función no lineal, tendríamos que transformar $\mathbf{x}$ de tal manera que podamos obtener una función de clasificación lineal como se vió en la Subsección 4.3.3, por lo que la Ecuación 4.12 se reescribiría como:

$$f(\mathbf{x};\theta) = \mathbf{W}\phi(\mathbf{x}) + \mathbf{b} \tag{4.13}$$

Siguiendo la definición de (Murphy, 2022), podemos dotar a la función de transformación ϕ de sus propios parámetros $\boldsymbol{\theta'}$ y aprenderlos en lugar de tratar de definirla empíricamente. Por lo que:

$$f(\mathbf{x};\theta,\boldsymbol{\theta'}) = \mathbf{W}\phi(\mathbf{x};\boldsymbol{\theta'}) + \mathbf{b} \tag{4.14}$$

Este proceso se puede seguir realizando tantas veces como se desee para aproximar funciones más complejas Figura 4.8; si llevamos a cabo esta composición de funciones c veces tendremos dicho número de capas de perceptrones cuya última capa *oculta* es h_c, por lo que la salida del perceptrón multicapa utilizando una función de activación φ **diferenciable** está dada por:

$$y = \varphi(\mathbf{W}_{c+1}h_c + \mathbf{b}_{c+1}) \tag{4.15}$$

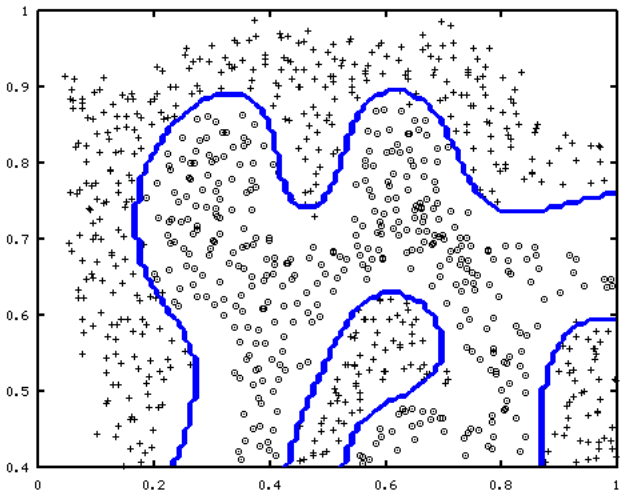


Figura 4.8: Una función compleja estimada por un perceptrón multicapa para datos no separables linealmente. Tomado de https://stackoverflow.com/questions/14440658/multi-layer-perceptron-finding-the-separating-curve

Para facilitar la interpretación de la Ecuación 4.15, la Figura 4.9 representa un perceptrón multicapa con dos capas *ocultas*, es decir, dos perceptrones apilados. A este tipo de arquitectura se le conoce como MLP (Multilayer Perceptron), red densa, red completamente conectada o red neuronal de propagación hacia adelante. El número de capas ocultas define la *profundidad* de la red; de ahí el nombre de **aprendizaje profundo**, que se le da a esta metodología para aproximar funciones complejas utilizando redes neuronales.

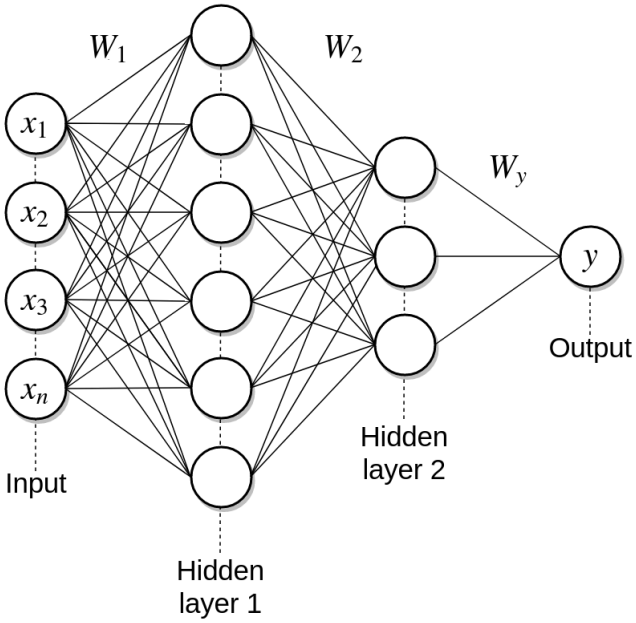


Figura 4.9: Perceptrón multicapa con dos capas ocultas. Tomado de https://www.researchgate.net/figure/A-fully-connected-neural-network-with-two-hidden-layers_fig4_323218981

4.3.4. Entrenamiento de una red neuronal

Si bien se ha mencionado que tanto un perceptrón simple, como un MLP son capaces de aproximar funciones que categorizan datos, el proceso para obtener los parámetros $\boldsymbol{\theta} = (\mathbf{W}, \mathbf{b})$ no

ha sido definido.

La Figura 4.8 muestra una función no lineal capaz de realizar una clasificación binaria, sin embargo, para ese mismo conjunto de datos existe una infinidad de funciones, con distintos parámetros $\boldsymbol{\theta}$, que pueden realizar la misma tarea. Entonces, ¿bajo qué criterio se determinan los parámetros que definirán a $f(\boldsymbol{x};\boldsymbol{\theta})$?. Para responder esta pregunta es necesario definir el concepto de **función de pérdida o coste**.

En términos generales, el objetivo de un MLP (y de cualquier algoritmo de aprendizaje automático) es estimar una función tal que minimize el número de salidas incorrectas del algoritmo. Dicho objetivo puede ser expresado de la siguiente manera (Aggarwal, 2018):

$$\min_{\mathbf{W}} L = \sum_{(\mathbf{x},y)} (y - \hat{y})^2 = \sum_{(\mathbf{x},y)} (y - \varphi(\mathbf{W}\mathbf{x} + \mathbf{b}))^2 \tag{4.16}$$

Donde a L se le conoce como función de pérdida o coste, la cual en el caso de la Ecuación 4.16 corresponde al error cuadrático medio y es muy símilar a la función a minimizar con el método de optimización de mínimos cuadrados. Sin embargo, esta función de pérdida se utiliza para problemas de regresión; para algoritmos de clasificación esta definición se extiende a una función de pérdida de entropía cruzada, dada por:

$$L = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \tag{4.17}$$

La Ecuación 4.17 es aplicable cuando sólo se tienen dos categorías (clases) de datos; para M clases se tiene:

$$L = -\sum_{c=1}^{M} y_c \log(\hat{y}_c) \tag{4.18}$$

En general, para un algoritmo de aprendizaje automático, la función de costo a minimizar es una expresión:

$$arg \min_{\theta} \sum L(f_\theta(\mathbf{x}), y) \tag{4.19}$$

Para resolver el problema de optimización dado por la Ecuación 4.19 la estrategia más común es inicializar los parámetros $\boldsymbol{\theta}$ aleatoriamente para posteriormente computar el valor de la función de pérdida y verificar la precisión del algoritmo. Si bien el razonamiento directo es ir modificando iterativamente en pequeñas magnitudes cada uno de los parámetros y decidir qué combinación tiene menor coste, dicha estrategia supone una labor titánica tanto para un humano como para una computadora, sobre todo si se trata de una red neuronal densa con miles (o millones) de parámetros.

Afortunadamente, se han desarrollado métodos iterativos para optimizar la función de costo de manera eficiente, en especial aquellos basados en el cálculo del gradiente de la función de costo con respecto a cada uno de los parámetros, siendo el más utilizado en aprendizaje automático el **descenso del gradiente**. El método de descenso del gradiente busca la combinación de parámetros más efectiva para llegar al mínimo óptimo de la función de pérdida apoyándose del concepto de derivada y por consiguiente del gradiente.

Derivar la función de pérdida es útil ya que por su definición nos dice cómo cambiar los parámetros $\boldsymbol{\theta}$ para obtener pequeñas mejoras en la imagen de la función (Goodfellow et al., 2016), a su vez, el gradiente es una generalización del concepto de derivada en un vector, y nos proporciona la dirección de cambio máximo de la función Figura 4.10.

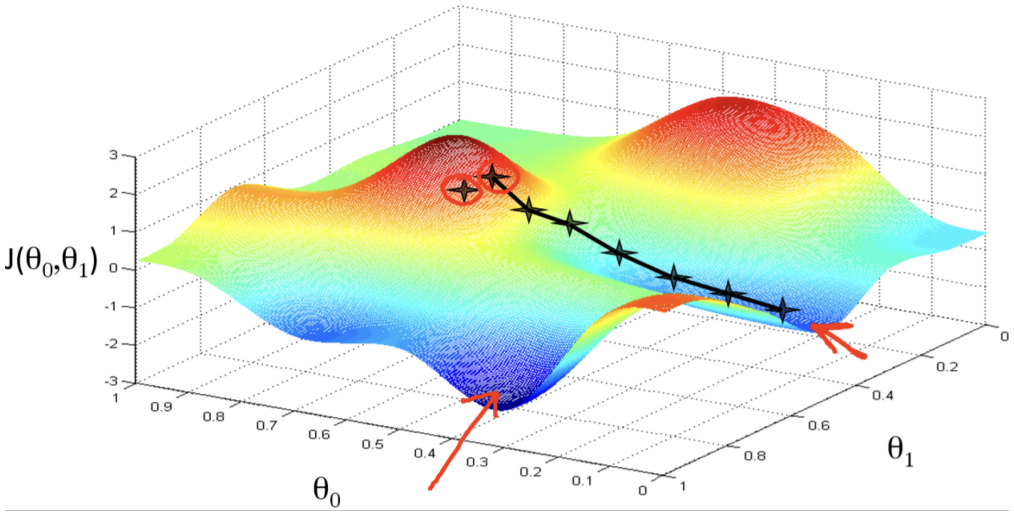


Figura 4.10: Método del descenso del gradiente para optimizar una función de pérdida J con respecto a sus parámetros θ_0 *y* θ_1. Tomado de https://medium.com/hackernoon/gradient-descent-aynk-7cbe95a778da

El gradiente de una función de pérdida $L(\boldsymbol{\theta})$ se denota como $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$, al ser $\boldsymbol{\theta}$ una representación de todos los parámetros de la red neuronal tal que $\boldsymbol{\theta} = (\theta_0, \theta_1, \cdots, \theta_n)$, entonces $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ estará dado por las derivadas parciales de L con respecto a cada uno de los parámetros $\boldsymbol{\theta}$.

De esta manera, la expresión del método del descenso del gradiente esta dada por:

$$\theta' = \theta - \alpha \nabla_\theta L(\theta) \tag{4.20}$$

Donde α es la **tasa de aprendizaje** un *hiper parámetro* que controla la magnitud de los pasos que toma el descenso del gradiente para converger a un mínimo.

Al calcular el gradiente de la función de costo, básicamente lo que estamos haciendo es decirle a la red neuronal qué parámetros son importantes y necesitan modificarse para poder llegar al mínimo de la función. Sin embargo, calcular el gradiente para una composición de tantas funciones no es una tarea trivial y mucho menos posible de calcular a mano, es incluso muy costosa para una computadora. Debido a esto, (Rumelhart et al., 1986) desarrolló el algoritmo de **propagación hacia atrás** (backpropagation) para calcular iterativamente el gradiente de una función de costo basándose en la regla de la cadena del cálculo

No nos detendremos a detallar el desarrollo de este algoritmo, basta con la comprensión de su principio general y por qué ha sido un factor importante para hacer del aprendizaje automático una alternativa viable para resolver problemas de clasificación. Para ello nos basaremos en la derivación realizada por (Goodfellow et al., 2016):

Se necesita calcular la derivada parcial de la función de pérdida L con respecto a los parámetros θ de la red neuronal, tal que $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \frac{\partial L}{\partial \theta}$. Si $\theta = (\mathbf{W}, \mathbf{b})$ entonces $\frac{\partial L}{\partial \theta} = (\frac{\partial L}{\partial \mathbf{W}}, \frac{\partial L}{\partial \mathbf{b}})$. Sea una red neuronal con C capas, la composición de funciones que representan a la última capa de la red está dada por la expresión:

$$E = L(\varphi(Z^C)) \tag{4.21}$$

Donde E es el error calculado por la función de pérdida L y $Z^C = \mathbf{W}^C X + \mathbf{b}^C$. Entonces, con base en la regla de la cadena del cálculo, las derivadas de la función de coste con respecto a los parámetros de la red en la última capa están dadas por:

$$\begin{aligned}\frac{\partial L}{\partial W^C} &= \frac{\partial L}{\partial \varphi^C}\frac{\partial \varphi^C}{\partial Z^C}\frac{\partial Z^C}{\partial W^C} \\ \frac{\partial L}{\partial b^C} &= \frac{\partial L}{\partial \varphi^C}\frac{\partial \varphi^C}{\partial Z^C}\frac{\partial Z^C}{\partial b^C}\end{aligned} \tag{4.22}$$

El término $\frac{\partial L}{\partial \varphi^C}\frac{\partial \varphi^C}{\partial Z^C}$ la variación del error en función de Z^C y representa el error provocado por las neuronas de la capa C, por lo tanto, podemos rescribir dicho término como δ^C y por ende, la Ecuación 4.22 se compacta de la siguiente forma:

$$\begin{aligned}\frac{\partial L}{\partial W^C} &= \delta^C \frac{\partial Z^C}{\partial W^C} \\ \frac{\partial L}{\partial b^C} &= \delta^C \frac{\partial Z^C}{\partial b^C}\end{aligned} \tag{4.23}$$

Las derivadas restantes son sencillas de calcular analíticamente. El vector de entrada a la capa C está dado por φ^{C-1}, por lo tanto $Z^C = \varphi^{C-1}W^C + b^C$. Realizando las derivadas correspondientes, la expresión Ecuación 4.23 se reduce a:

$$\begin{aligned}\frac{\partial L}{\partial W^C} &= \delta^C \ \varphi^{C-1} \\ \frac{\partial L}{\partial b^C} &= \delta^C\end{aligned} \tag{4.24}$$

Este proceso se repite para la capa $C-1$, sin embargo ya hemos ahorrado gran parte de los cálculos para dicha capa. Observemos la expresión del gradiente en la capa $C-1$:

$$\begin{aligned}\frac{\partial L}{\partial W^{C-1}} &= \frac{\partial L}{\partial \varphi^C}\frac{\partial \varphi^C}{\partial Z^C}\frac{\partial Z^C}{\partial \varphi^{C-1}}\frac{\partial \varphi^{C-1}}{\partial Z^{C-1}}\frac{\partial Z^{C-1}}{\partial W^{C-1}} \\ \frac{\partial L}{\partial b^{C-1}} &= \frac{\partial L}{\partial \varphi^C}\frac{\partial \varphi^C}{\partial Z^C}\frac{\partial Z^C}{\partial \varphi^{C-1}}\frac{\partial \varphi^{C-1}}{\partial Z^{C-1}}\frac{\partial Z^{C-1}}{\partial b^{C-1}}\end{aligned} \tag{4.25}$$

El par de ecuaciones anterior podrá parecer muy complicado de calcular, pero en realidad ya calculamos la mayor parte de las derivadas que se necesitan, la Ecuación 4.25 se reescribe como:

$$\begin{aligned}\frac{\partial L}{\partial W^{C-1}} &= \delta^C \frac{\partial Z^C}{\partial \varphi^{C-1}}\frac{\partial \varphi^{C-1}}{\partial Z^{C-1}}\varphi^{L-2} \\ \frac{\partial L}{\partial b^{C-1}} &= \delta^C \frac{\partial Z^C}{\partial \varphi^{C-1}}\frac{\partial \varphi^{C-1}}{\partial Z^{C-1}}\end{aligned} \tag{4.26}$$

Notamos que el cálculo del gradiente en la capa $C-1$ se ha simplificado en gran medida, y puede simplificarse aún más, ya que el término $\frac{\partial \varphi^{C-1}}{\partial Z^{C-1}}$ representa la derivada de la función de activación, por lo que dicha derivada una vez calculada para la función de activación utilizada será constante para cualquier capa. Dicho lo anterior, solo nos restaría calcular una sola derivada; si reproducimos este proceso recursivamente solo tendremos que calcular $C-2$ derivadas y habremos obtenido exitosamente el gradiente de la función de pérdida con respecto a los parámetros de la red para poder efectuar el método del descenso del gradiente. De esta manera, se es ahora capaz de entrenar completamente una red neuronal de manera efectiva.

El algoritmo de propagación hacia atrás era el eslabón que faltaba en el campo del aprendizaje automático para poder entrenar redes neuronales a gran escala, ya que las operaciones necesarias son considerablemente menos y la capacidad computacional necesaria para hacerlo es mucho menor.

4.4. Redes neuronales recurrentes y sus arquitecturas derivadas

Un perceptrón multicapa es un aproximador de funciones muy potente, sin embargo, la rigidez de dicha arquitectura sólo permite mapear una entrada x_i a una salida y_i. ¿Qué pasaría si los datos de entrada tienen alguna dependencia espacio-temporal?, es decir, que sean datos *secuenciales* dado que la entrada x_i (y por consiguiente su salida y_i) depende de todas las entradas anteriores. Este es un problema mucho más común de lo que parece, especialmente en tareas de reconocimiento de voz, traducción automática de textos e incluso en nuestro caso especial donde tenemos registros geofísicos de pozo, los cuales son datos secuenciales. Si bien un MLP será capaz de clasificar *entrada a entrada*, la salida del modelo carecerá de la naturaleza de nuestros datos de entrada, y por lo tanto, es muy probable que la salida tenga poco sentido acorde a los parámetros físico-matemáticos que caracterizan a la tarea en cuestión.

Analizemos esta problemática de la siguiente manera: Tenemos una serie de registros de un pozo en cuestión y queremos predecir el tipo de roca que se encuentra a cada nivel de profundidad. Sabemos que geológicamente existe una gran interdependencia entre cada cuerpo de roca presente en un yacimiento, por lo que nuestras predicciones deberían ser geológicamente consistentes. Es decir, sea una predicción de roca del modelo a una profundidad p, para tener un resultado geológicamente consistente es necesario que se tomen en cuenta las rocas que se predijeron desde $p^{(1)}$ hasta $p-1$, de lo contrario la salida en p podrá ser cualquier tipo de roca y eso no es necesariamente consistente. Por ejemplo, si las últimas n predicciones de roca corresponden a una lutita es probable que para el nivel de profundidad actual nos encontremos en una secuencia de lutitas y por lo tanto la probabilidad de encontrarnos con otra lutita en el siguiente nivel de profundidad gracias al **contexto** debe ser mayor que la de encontrarnos cualquier otra litología, por ejemplo, una halita.

Hacer frente a este problema implica que el modelo tenga algún tipo de *memoria* para recordar a todas las entradas anteriores y poder realizar una predicción con base a los estados previos. El concepto de memoria puede representarse en términos de aprendizaje automático como que la red neuronal debe tener algún tipo de estructura que le permita realizar conexiones de manera cíclica. Esta estructura se ha implementado para desarrollar una familia de arquitecturas de redes neuronales llamada: **Redes Neuronales Recurrentes** o **RNN's**.

Las RNN's basan su funcionamiento en el principio de compartir parámetros, ya que si los parámetros son independientes para x_i estaremos perdiendo la relación secuencial de los datos que mencionamos con anterioridad. Si compartimos los parámetros para cada entrada estaremos asegurando que cada elemento de salida es función de las anteriores.

(Goodfellow et al., 2016) describe a las redes neuronales recurrentes como sistemas dinámicos que reciben una señal externa x^t tal que:

$$\mathbf{s}^{(t)} = f(\mathbf{s}^{(t-1)}; \mathbf{x}^t; \boldsymbol{\theta}) \tag{4.27}$$

Es decir, el estado $\mathbf{s}$ en el tiempo t está referido al estado $\mathbf{s}$ en el tiempo $t-1$. Si para una capa oculta de una red neuronal recurrente cualquiera llamamos al estado $\mathbf{s}$ el estado oculto (de una capa oculta) $\mathbf{h}$, tendremos que reescribir la Ecuación 4.28 como:

$$\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}; \mathbf{h}^t; \boldsymbol{\theta}) \tag{4.28}$$

En una RNN, el estado oculto $\mathbf{h}^{(t)}$ representa un *resumen* de los aspectos más importantes de todas las entradas anteriores a t:

$$\mathbf{h}^{(t)} = g^{(t)}(\mathbf{x}^{(t)}, \mathbf{x}^{(t-1)}, \cdots, \mathbf{x}^{(1)}) \tag{4.29}$$

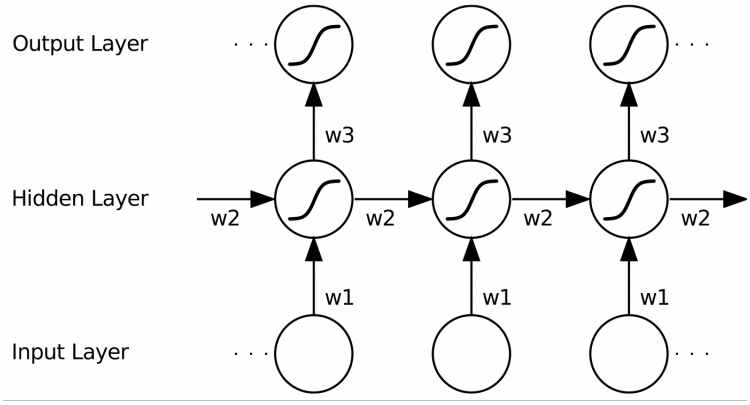


Figura 4.11: Arquitectura simple de una Red Neuronal Recurrente. En las capas ocultas se observa el principio de compartir los parámetros aprendidos en entradas anteriores para proveer del contexto previo antes de generar una salida. Tomado de (Graves, 2012)

Podemos observar la arquitectura básica de una red neuronal recurrente en la Figura 4.11. Como podemos observar, se comparten los parámetros en las capas ocultas para lograr mantener la interdependencia entre los parámetros aprendidos. Si analizamos la notación de la figura podemos identificar tres tipos de matrices de pesos presentes en una red neuronal recurrente. $\boldsymbol{W_1}$ representa los pesos aprendidos por la red para las conexiones entre la entrada y las capas ocultas, $\boldsymbol{W_2}$ las conexiones de capa oculta a capa oculta y por último $\boldsymbol{W_3}$ que mapea las capas ocultas a la capa de salida.

Con base en la definición anterior de los tipos de matrices de pesos en una RNN, podemos redefinir las ecuaciones de la propagación hacia adelante de una RNN (Goodfellow et al., 2016), tal que para cualquier estado desde $t = 1$ hasta $t = \tau$:

$$\begin{aligned} \mathbf{a}^{(t)} &= \mathbf{b} + \boldsymbol{W_2}\mathbf{h}^{(t-1)} + \boldsymbol{W_1}\mathbf{x}^{(t)}, \\ \mathbf{h}^{(t)} &= tanh(\mathbf{a}^{(t)}), \\ \mathbf{o}^{(t)} &= \mathbf{c} + \boldsymbol{W_3}\mathbf{h}^{(t)}, \\ \hat{\mathbf{y}}^{(t)} &= softmax(\mathbf{o}^{(t)}) \end{aligned} \tag{4.30}$$

Donde $\mathbf{b}$ y $\mathbf{c}$ son los vectores de sesgos correspondientes a las matrices de pesos $\boldsymbol{W_2}$ y $\boldsymbol{W_3}$ respectivamente y la función de activación utilizada es la función de tangente hiperbólico. Por último, para obtener una distribución de probabilidad se utiliza la función softmax sobre la salida $\mathbf{o}^{(t)}$ dada por la ecuación:

$$softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad para\ i = 1, 2, \ldots, K \tag{4.31}$$

La forma de entrenar una RNN es prácticamente la misma que para un MLP, ya que también se utiliza el método del descenso del gradiente y por consiguiente el algoritmo de propagación hacia atrás, aunque con una ligera variación para poder ser aplicado a las ecuaciones de una RNN; a esta variante se le conoce como *propagación hacia atrás a través del tiempo*, sin embargo no detallaremos su funcionamiento por cuestiones de practicidad.

4.4.1. Redes de gran memoria a corto plazo: LSTM's

A pesar de la gran utilidad de las RNN's para trabajar con datos secuenciales, tienen una problemática que ha dado pie al desarrollo de otras variantes de arquitecturas de redes neuronales recurrentes. Las RNN's con muy efectivas para secuencias cortas de texto, como oraciones cortas de un texto o cuadros cortos de videos, sin embargo, si se requiere trabajar con secuencias

más largas de datos, como un párrafo de texto o grabaciones medianamente largas de voz, carecen de la capacidad de proveer del contexto inicial para predecir las salidas finales de la secuencia.

El problema anteriormente descrito se conoce como **problema del gradiente desvanesciente** y no es exclusivo de las RNN's, de hecho, este es un problema general de las redes neuronales a medida que se hacen más profundas. La razón principal de esta problemática es que los gradientes correspondientes a las primeras dependencias de la red neuronal van perdiendo su *efecto* a medida que se procesan más dependencias, debido a que se realizan muchas multiplicaciones matriciales y para la red es muy complicado identificar la contribución de las dependencias iniciales. Afortunadamente, existen mecanismos como las Redes Neuronales De Gran Memoria a Corto Plazo o Long Short Term Memory Units (LSTM's) (Hochreiter y Schmidhuber, 1997) que fueron creados específicamente para trabajar con secuencias de datos más largas.

En las RNN's clásicas el mecanismo utilizado para conjuntar los parámetros aprendidos por las dependencias anteriores es una sola capa formada por una función activación de tangente hiperbólico (Ecuación 4.30). Por otra parte, las LSTM's están compuestas por cuatro capas de redes neuronales capaces de realizar esta misma tarea pero procurando recordar las dependencias antiguas.

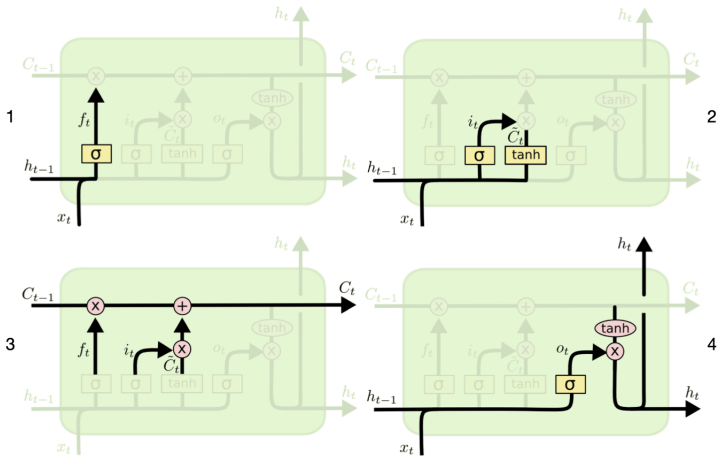


Figura 4.12: Descripción del funcionamiento de una LSTM. 1) Decidir la información a desechar, 2)Decidir la información a incorporar, 3) Actualizar la célula de estado, 4) Obtener la salida. Modificado de https://colah.github.io/posts/2015-08-Understanding-LSTMs/

(Olah, 2015) explica el funcionamiento general de una LSTM, el cual sirve como base del siguiente apartado:

Las LSTM's se valen del concepto de *célula de estado* C_t, el cual es una abstracción de una memoria a largo plazo, que cambia muy poco a lo largo del procesamiento de la secuencia y que sólo es alterada por la LSTM cuando existe información que merece ser recordada a largo plazo. La Figura 4.12 describe el proceso de aprendizaje de una LSTM, el cual puede ser resumido en 4 simples pasos:

1. **Decidir la información a desechar**: Una capa compuesta por una función sigmoide llamada *compuerta de olvido* es utilizada para decidir qué información remover de la célula

de estado. Está dada por la función:

$$f_t = \sigma(\boldsymbol{W_f} \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b_f}) \tag{4.32}$$

2. **Decidir la información a incorporar**: En esta etapa se decide la información que será añadida a la memoria a largo plazo. Está compuesta de dos partes, una *compuerta de entrada* i_t que decide cuáles son los valores que se actualizarán y por otra parte un vector de valores canditados $\tilde{C}_t$. Ambos están definidos respectivamente por las expresiones:

$$\begin{aligned} i_t &= \sigma(\boldsymbol{W_i} \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b_i}) \\ \tilde{C}_t &= tanh(\boldsymbol{W_C} \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b_C}) \end{aligned} \tag{4.33}$$

3. **Actualizar la célula de estado**: Como su nombre lo dice, consiste en simplemente utilizar los valores calculados anteriormente para actualizar el valor de la célula de estado, olvidando la información necesaria y añadiendo nuevos valores candidatos conforme a lo obtenido en el paso pasado. La célula de estado se actualiza como:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{4.34}$$

4. **Obtener la salida**: Por último, se obtiene el vector de estados ocultos h_t a partir de una capa sigmoide y una función de activación de tangente hiperbólico:

$$\begin{aligned} o_t &= \sigma(\boldsymbol{W_o} \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b_o}) \\ h_t &= o_t * tanh(C_t) \end{aligned} \tag{4.35}$$

De esta manera, se han equipado a las redes neuronales recurrentes de una *memoria*, esto es particularmente útil para nuestro problema de clasificación, ya que queremos que para cada predicción de litofacies la red neuronal que creemos sea capaz de tomar en cuenta a las entradas que se encuentran antes de la actual. Incluso, podemos dotar a una red LSTM de la capacidad de ver las entradas que se encuentran después de la actual; a esta variante se le conoce como LSTM bidireccional (Bi-LSTM).

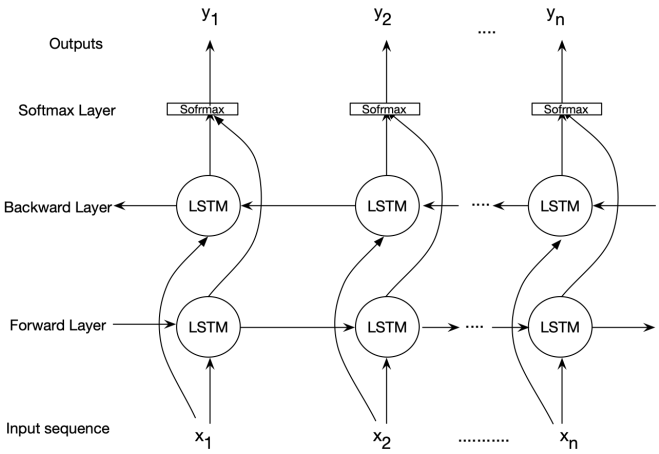


Figura 4.13: LSTM bidireccional, que consiste de dos capas de LSTM's que trabajan observando las entradas anteriores y futuras. Tomado de https://nedatavakoli.github.io/Modeling-and-Clustering-Genome-using-Bidirectional-LSTM/

No entraremos en detalle con respecto a las ecuaciones del BiLSTM, ya que básicamente se compone de dos capas de LSTM's, la primera en la dirección *hacia adelante* y la segunda en la

dirección *hacia atrás* Figura 4.13. Las ecuaciones siguen siendo las mismas, la única diferencia es que tendremos el doble de parámetros para entrenar; sin embargo, las ventajas si que son significativas, ya que le permitimos a la red neuronal que *vea* a las entradas a su alrededor y realice la mejor predicción con base en el pasado y el futuro.

Capítulo 5

Metodología

El enfoque elegido para cumplir el objetivo de clasificar litofacies a partir de registros de pozo de manera automatizada fue diseñar una red neuronal que reciba curvas de registros geofísicos de pozo y produzca una columna estratigráfica que represente las litofacies correspondientes a cada nivel de produndidad (Figura 5.1).

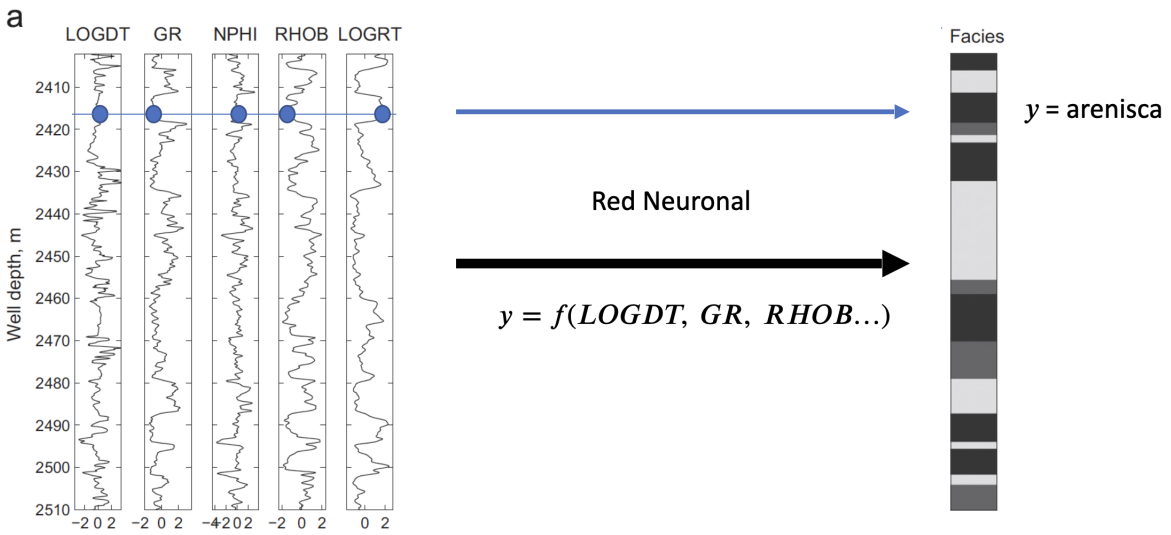


Figura 5.1: Descripción de la metodología básica. Se toman n registros geofísicos de pozo como entrada a una red neuronal que los procesa y obtiene la columna de litofacies correspondientes a cada nivel de profundidad. Traducido de imagen original creada por (Ravasi, 2021).

La metodología propuesta se divide en las siguientes etapas:

1. **Creación de un conjunto de datos sintéticos**: Crear datos sintéticos a partir de características petrofísicas básicas.

2. **Definición de los modelos de aprendizaje automático**: Desarrollar dos modelos de aprendizaje automático. El primero, un perceptrón multicapa para realizar una comparación básica de resultados. El segundo, un LSTM Bidireccional para explorar las capacidades de las RNN's en clasificación de registros geofísicos de pozo.

3. **Descripción de las métricas de evaluación**: Definir las métricas básicas de evaluación, así como una métrica propuesta especialmente para el conjunto de datos reales.

5.1. Creación de un conjunto de datos sintéticos

En geofísica, las limitantes principales para la adopción del aprendizaje automático para apoyar en la resolución de distintas tareas como la identificación de fallas geológicas en secciones sísmicas, la selección de primeros arribos de eventos en sismogramas o el procesamiento de datos sísmicos, por poner algunos ejemplos, se deben a distintos factores entre los que destacan dos en particular: la **privacidad** de los datos y la existencia de datos debidamente **etiquetados** para su uso en algoritmos de aprendizaje automático.

La cuestión de la privacidad de los datos es un tema ampliamente difundido en la comunidad de proyectos de código abierto en geofísica, ya que quienes se encargan de adquirir dichos datos son compañías con un interés comercial sobre ellos. Sin embargo, existen ciertos esfuerzos por parte de entes y organizaciones sin fines de lucro por democratizar el uso de datos no comerciales con fines de investigación (por ejemplo, https://www.bgs.ac.uk/geological-data/opengeoscience/).

Por otro lado, para poder entrenar algoritmos de aprendizaje automático (al menos en aprendizaje supervisado) es necesario tener conjuntos de datos debidamente etiquetados y disponibles en gran cantidad. Por ejemplo, para poder entrenar un algoritmo que remueva el ruido de una sección sísmica es necesario tener miles de secciones sísmicas ruidosas y su respectiva sección sin ruido. Esto es indudablemente difícil de obtener en un campo de estudio como la geofísica.

Si bien existen varias técnicas para evitar estos problemas, como desarrollar algoritmos de aprendizaje no supervisado, nosotros optamos por crear nuestro propio conjunto de datos sintéticos de registros geofísicos de pozo debidamente etiquetados con la litofacies correspondiente en cada nivel de profundidad. Aunque el escenario ideal sería entrenar nuestros algoritmos con datos reales para poder evaluar en datos de este mismo tipo, creemos que el desempeño de un algoritmo de redes neuronales recurrentes entrenado en datos sintéticos puede ser muy bueno a la hora de evaluar en datos reales si diseñamos dicho algoritmo correctamente. Además, otras ventajas de utilizar datos sintéticos es que tenemos una cantidad ilimitada de datos para entrenar y podemos tener un control absoluto sobre la naturaleza de los mismos.

Crear registros geofísicos de pozo sintéticos es una tarea lo suficientemente compleja como para realizar un trabajo de investigación por sí sólo. Diversos autores como (S. Kim et al., 2020), (D. Zhang et al., 2018) y (Akinnikawe et al., 2018) explorar diversos métodos utilizando Machine Learning para la creación de datos sintéticos a partir de redes neuronales recurrentes y árboles de decisión. Sin embargo, nosotros consideramos fuera del enfoque de este trabajo el utilizar técnicas de aprendizaje automático para la generación de datos sintéticos.

Por lo tanto, decidimos crear curvas de *pseudo registros* que fueran fáciles de generar a partir de parámetros petrofísicos muestreados utilizando el método de Cadenas de Markov Monte Carlo. Si bien, en primera instancia no estamos utilizando registros geofísicos propiamente, podemos utilizar estos pseudo registros para realizar pruebas de concepto de los algoritmos para verificar que efectivamente un LSTM Bidireccional es capaz de clasificar litofacies a partir de curvas que se asemejan a la *forma* de un registro de pozo, para finalmente entrenar un clasificador con registros geofísicos de pozo reales (Sección 6.2).

Para ello, elegimos cinco litologías a modelar en nuestro conjunto de datos sintéticos y que representan las **clases** que predecirán los modelos de aprendizaje automático:

- Creta
- Caliza

- Dolomía
- Arenisca saturada con agua S_w
- Arenisca saturada con hidrocarburos S_o

Los métodos de Cadena de Markov Monte Carlo son algoritmos que obtienen muestras a partir de una distribución de probabilidad. Si cada litofacies es una distribución de probabilidad multivariada, para generar las curvas de registros sintéticas, tenemos que utilizar cierto método de muestreo, en este caso muestreo Monte Carlo, que siga las propiedades de Markov para obtener muestras de dichas distribuciones. En primera instancia, tenemos que definir los parámetros petrofísicos a analizar y generar una distribución de probabilidad para cada litofacies a partir de:

$$X \sim \mathcal{N}(\mu,\ Cov) \tag{5.1}$$

Donde, X es una distribución de probabilidad normal, μ es el valor de la media de dicha probabilidad y Cov es una matriz de covarianza que controla la forma de dicha distribución.

A partir de los valores proporcionados en (Mavko et al., 2009) para media y desviación estandar de Densidad ρ, Impedancia Acústica (IA) y el cociente entre velocidad de onda P y velocidad de onda S (V_p/V_s) para cada una de las litofacies a modelar se pueden parametrizar distribuciones de probabilidad independientes como se muestra en la Tabla 5.1:

Litología	**Media** [Vp/Vs, IA, ρ]	**Desviación Estándar** [Vp/Vs, IA, ρ]
Creta	[1.67, 4.02, 1.85]	[0.06,0.89,0.13]
Caliza	[1.88, 1.43, 2.43]	[0.08,0.22,0.16]
Dolomía	[1.82,1.4,2.59]	[0.07, 0.23, 0.12]
S_o	[1.3, 7.57, 1.6]	[0.13, 0.67, 0.13]
S_w	[1.2, 6.7, 1.5]	[0.13, 0.67, 0.13]

Tabla 5.1: Parámetros de media y desviación estándar para cada uno de los parámetros petrofísicos utilizados en las cinco litologías. Tomado de (Mavko et al., 2009).

A su vez, las matrices de covarianza que parametrizan cada distribución generada están dadas por la ecuación:

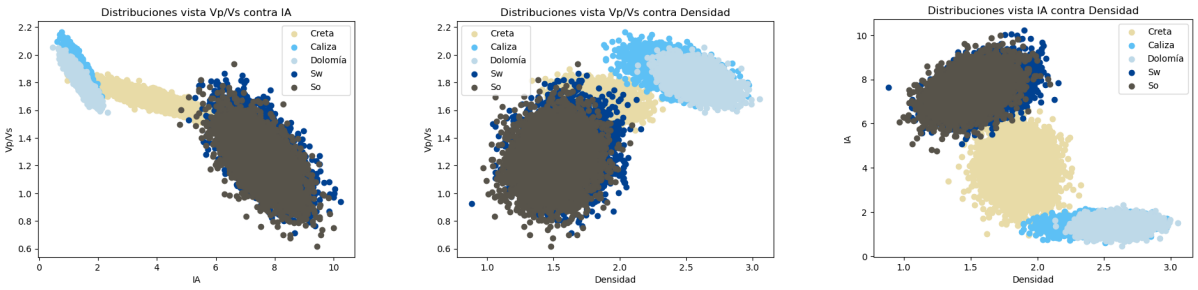
$$Cov = \begin{bmatrix} \sigma_{11} & 0 & 0 \\ 0 & \sigma_{22} & 0 \\ 0 & 0 & \sigma_{33} \end{bmatrix} \begin{bmatrix} 1 & \rho_{12} & \rho_{13} \\ \rho_{21} & 1 & \rho_{23} \\ \rho_{31} & \rho_{32} & 1 \end{bmatrix} \begin{bmatrix} \sigma_{11} & 0 & 0 \\ 0 & \sigma_{22} & 0 \\ 0 & 0 & \sigma_{33} \end{bmatrix} \tag{5.2}$$

Donde σ_{ii} corresponde a la desviación estandar de cada parámetro petrofísico utilizado y ρ_{ij} es un coeficiente de correlación que controla la forma de la distribución generada. Dichos coeficientes fueron propuestos de la siguiente manera: La forma de cada una de las distribuciones fueron

Litología	**Coeficientes de Correlación** [Vp/Vs, IA, ρ]
Creta	[-0.9,-0.3,-0.1]
Caliza	[-0.9,-0.5,0.13]
Dolomía	[-0.9,-0.5,0.13]
S_o	[-0.81,0.7,0.5]
S_w	[-0.7,0.65,-0.5]

Tabla 5.2: Coeficientes de correlación propuestos para cada propiedad en todas las litologías. Estos coeficientes de correlación son propuestos acorde a la forma deseada de la distribución.

propuestas con un propósito; necesitamos que la relación entre los datos sea lo suficientemente compleja para poder aprovechar la capacidad de las redes neuronales de clasificar datos con una superficie de separación no lineal. Si proponemos distribuciones fácilmente distinguibles y separables, puede que no sea necesario utilizar una red neuronal, pero esto tampoco se apega a la realidad en la que podemos tener una intricación entre litofacies compleja. Una vista de las distribuciones propuestas es proporcionada por la Figura 5.2.



(a) Vp/Vs vs IA (b) Vp/Vs vs Densidad (c) IA vs Densidad

Figura 5.2: Distribuciones generadas a partir de los parámetros descritos en Tabla 5.1 y Tabla 5.2.

Ahora procedemos a definir la estructura de cadena de Markov necesaria para poder realizar un muestreo Monte Carlo y generar nuestras curvas de pseudo registros. Para ello comenzamos proponiendo una matriz de transiciones entre litofacies que nos ayude a definir las probabilidades condicionales que forman la estructura de cadena de Markov, de tal manera que tendremos para cada litofacies probabilidades condicionales $p(F_j|F_i)$, es decir, la probabilidad de muestrear una litofacies F_j dado que la litofacies recién muestreada es F_i. La matriz de transiciones tiene la forma:

$$\mathbf{T} = \begin{bmatrix} p(Cr \to Cr) & p(Cr \to Ca) & p(Cr \to D) & p(Cr \to S_w) & p(Cr \to S_o) \\ p(Ca \to Cr) & p(Ca \to Ca) & p(Ca \to D) & p(Ca \to S_w) & p(Ca \to S_o) \\ p(D \to Cr) & p(D \to Ca) & p(D \to D) & p(D \to S_w) & p(D \to S_o) \\ p(S_w \to Cr) & p(S_w \to Ca) & p(S_w \to D) & p(S_w \to S_w) & p(S_w \to S_o) \\ p(S_o \to Cr) & p(S_o \to Ca) & p(S_o \to D) & p(S_o \to S_w) & p(S_o \to S_o) \end{bmatrix} \tag{5.3}$$

Cuyos valores propuestos son:

$$\mathbf{T} = \begin{bmatrix} 0{,}8 & 0{,}05 & 0{,}05 & 0{,}05 & 0{,}05 \\ 0{,}7 & 0{,}05 & 0{,}05 & 0{,}15 & 0{,}05 \\ 0{,}5 & 0{,}05 & 0{,}05 & 0{,}3 & 0{,}10 \\ 0{,}2 & 0{,}05 & 0{,}05 & 0{,}7 & 0 \\ 0{,}1 & 0{,}05 & 0{,}05 & 0{,}1 & 0{,}7 \end{bmatrix} \tag{5.4}$$

Nótese en la Ecuación 5.4 que la $p(S_w \to S_o)$ es 0. Esto debido a que sabemos que por diferencia de densidad no es posible encontrar en un yacimiento agua e inmediatamente después aceite, por lo tanto dicha probabilidad de transición es 0; esta es una restricción que queremos hacer notar durante el entrenamiento de nuestro modelo a través de los mecanismos propios de las redes neuronales recurrentes para aprender efectivamente aspectos característicos de una secuencia. En cuanto al resto de las probabilidades propuestas, no seguimos alguna regla en específico, simplemente quisimos mantener una fuerte presencia de los dos tipos de areniscas y de la creta; en cuanto a esta última aumentamos su presencia en los datos generados debido a su fuerte interacción con todas las distribuciones generadas, para que le sea más difícil al modelo diferenciar entre litofacies.

Para comenzar el proceso de muestreo para generar las curvas, definimos una matriz T_{cum} que

representa las probabilidades acumulativas de la matriz T (Ecuación 5.4), a su vez, para comenzar nuestro muestreo a partir de cadenas de Markov necesitamos un vector de probabilidades iniciales P_0 y su correspondiente vector de probabilidades acumulativas $P_{0_{cum}}$. Estos dos últimos vectores definen la probabilidad en el estado inicial del muestreo para cada una de las litofacies y están dados por:

$$P_0 = \begin{bmatrix} P_{0_{Cr}} \\ P_{0_{Ca}} \\ P_{0_D} \\ P_{0_{S_w}} \\ P_{0_{S_o}} \end{bmatrix} \tag{5.5}$$

$$P_{0_{Cum}} = \begin{bmatrix} P_{0_{Cr}} \\ P_{0_{Cr}} + P_{0-Ca} \\ P_{0_{Cr}} + P_{0_{Ca}} + P_{0_D} \\ P_{0_{Cr}} + P_{0_{Ca}} + P_{0_D} + P_{0_{S_w}} \\ 1 \end{bmatrix} \tag{5.6}$$

Una vez definidos estos vectores, procedemos a generar una secuencia de litofacies que llamaremos F como la que se muestra en la Figura 5.1. Esta nos servirá para generar las tres curvas de IA, V_p/V_s y ρ, ya que al tener la litofacies correspondiente a cada nivel de profundidad, simplemente obtendremos muestras aleatorias de las distribuciones generadas anteriormente para cada propiedad y litofacies correspondiente. Para ello nos apoyamos en el algoritmo 1:

Algoritmo 1 Algoritmo de generación de secuencias de facies

Sean: $P_{0cum} = P_c$; $T_{cum} = T_c$; $L_k = (Cr, Ca, D, S_w, S_o)$; F_i ; N
  $N \leftarrow 0$
  $x \leftarrow \mathcal{U}\ (0, 1)$
  $F_0 \leftarrow (L_{i-1} | P_{ci} > x)$
  $N \leftarrow 1$
  mientras $N < i$ **do**
    $x \leftarrow \mathcal{U}\ (0, 1)$
    $F_j \leftarrow (L_{j-1} | T_{c(i-1,j)} > x)$
    $N \leftarrow N + 1$
  fin mientras

El algoritmo anterior describe que, sea una sequencia de litofacies F con N número de elementos y un conjunto de litofacies $L_k = (Cr, Ca, D, S_w, S_o)$. Para determinar el primer elemento de la secuencia F_0 muestreamos un valor x de una distribución uniforme $\mathcal{U}$ y revisamos en el vector $P_{0_{Cum}}$ el valor con índice i que sea mayor a x. Por lo tanto, el valor de F_0 será la litofacies L_{i-1}.

Para determinar el resto de elementos de la secuencia F_j, repetiremos el proceso anterior iterativamente. Volveremos a muestrear x y revisaremos ahora en T_{cum} en la fila $i-1$ el valor con índice j que sea mayor a x. Entonces, el elemento F_j será L_{j-1}.

Una vez obtenida la secuencia de litofacies F y realizado el proceso posterior de generar las tres curvas de pseudo registros como se describió anteriormente, tendremos ya nuestro *pozo sintético* y las litofacies correspondientes a cada nivel de profundidad como se muestra en la Figura 5.3:

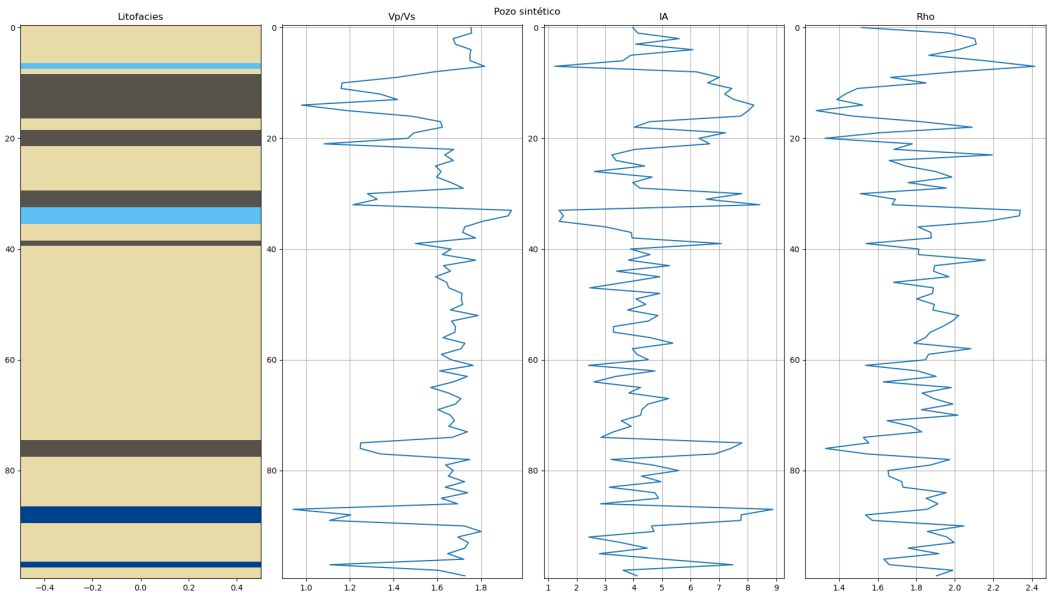


Figura 5.3: Terna de pseudo registros de pozo generados sintéticamente etiquetados con la litofacies correspondiente en cada nivel de profundidad.

5.2. Definición de los modelos de aprendizaje automático

La Figura 5.4 muestra el flujo de trabajo utilizado para la clasificación de litofacies. Empezamos con una entrada, correspondiente a las tres curvas de registros de pozo que se crean sintéticamente, la cual ingresa a la red neuronal en la capa de entrada. Posteriormente, la red neuronal procesa la entrada a partir de los parámetros aprendidos durante el entrenamiento y genera una predicción que representa a cada litofacies correspondiente a cada nivel de profundidad.

Aún no hemos definido al componente intermedio de este flujo de trabajo, es decir, la red neuronal que usaremos. En este caso se utilizaron dos arquitecturas distintas de redes neuronales para realizar predicciones de litofacies a partir de registros de pozo sintéticos: un MLP y un LSTM Bidireccional. La elección de estas dos arquitecturas se debe a la naturaleza del problema que queremos resolver; un LSTM Bidireccional nos provee de la capacidad de hacer que nuestro modelo aprenda en un tiempo T_i de las muestras en T_{i-1} y T_{i+1}, por otra parte, el MLP nos permite comparar el rendimiento del LSTM Bidireccional con un modelo más simple.

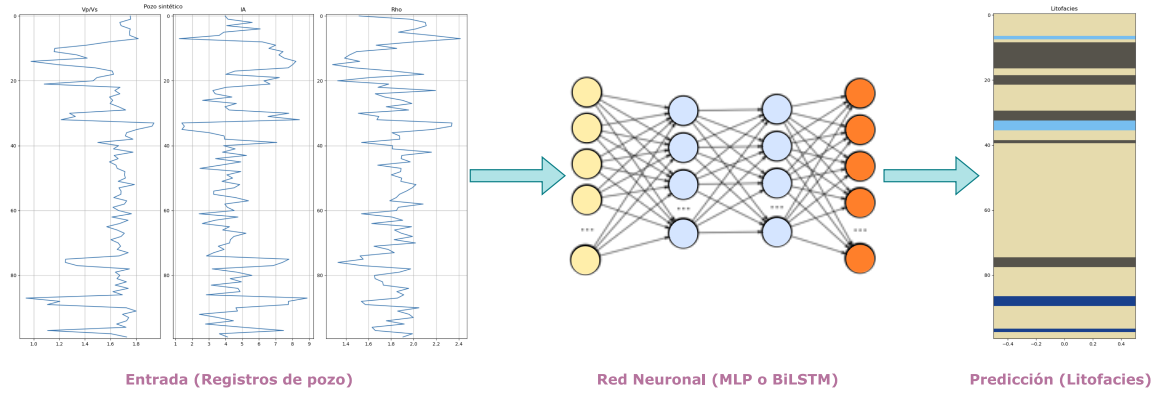


Figura 5.4: Representación conceptual del flujo de los datos.

En el MLP (Figura 5.5), el modelo consiste en una secuencia de capas lineales donde cada salida

de las capas ocultas pasa por una función de activación ReLu. Por otra parte, la salida de la última capa lineal es una matriz de tamaño $N_{muestras} \times N_{clases}$, y contendrá a las predicciones no normalizadas que genera el modelo, o bien, como se conoce en aprendizaje automático: los *logits*.

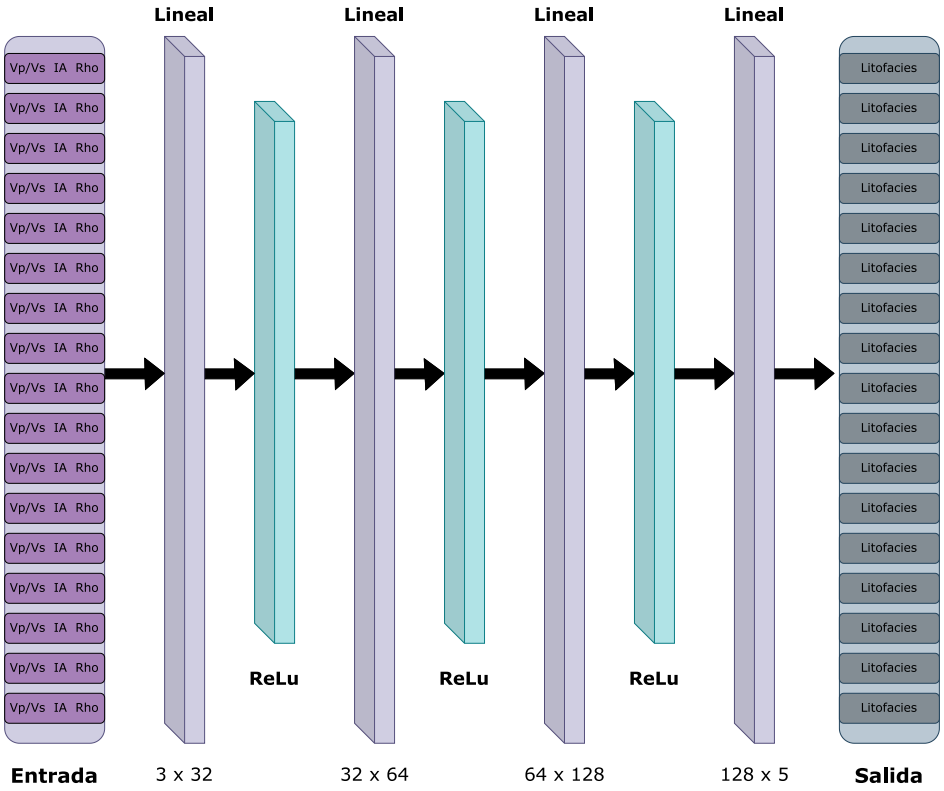


Figura 5.5: Arquitectura propuesta para generar un modelo MLP que servirá como punto de comparación con el modelo de BiLSTM.

La Figura 5.6 muestra la descripción de la arquitectura de LSTM Bidireccional propuesta. En este caso la entrada fue modificada de tal manera que se puedan generar *ventanas*, cuyo tamaño W_l es un hiperparámetro del modelo, esto con el objetivo de que multiples muestras tanto en T_{i-1} como en T_{i+1} puedan ser utilizadas para realizar predicciones en T_i.

El número de neuronas ocultas para cada capa LSTM es de 256, por lo tanto, la salida del BiLSTM es una matriz de tamaño W_l x 512, ya que al ser un LSTM Bidireccional se cuenta con dos capas LSTM de 256 neuronas cada una. Por último, la salida del BiLSTM será la entrada de una capa lineal que transforma la representación del BiLSTM a una matriz de tamaño W_l x N_{clases} con los *logits* obtenidos.

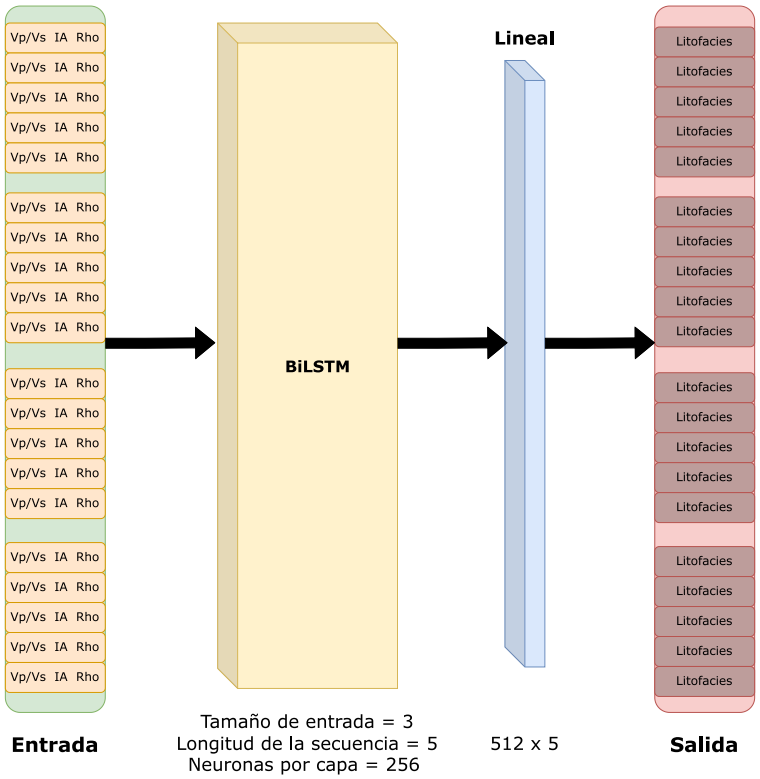


Figura 5.6: Arquitectura propuesta para un modelo de LSTM Bidireccional

5.3. Descripción de las métricas de evaluación

En aprendizaje automático existen dos fases importantes en el desarrollo de un modelo: **entrenamiento** e **inferencia**. Hemos mencionado el entrenamiento de una red neuronal como el método iterativo para optimizar una función de pérdida con respecto a los parámetros aprendidos por la red neuronal. En cuanto a la inferencia, esta es la parte del ciclo de vida en aprendizaje automático donde el algoritmo hace predicciones a partir de datos nunca antes vistos, utilizando los parámetros aprendidos durante el entrenamiento.

Para definir los elementos que se utilizaron para evaluar el desempeño de nuestros modelos en la fase de inferencia, es necesario definir la métrica utilizada durante entrenamiento. Como bien hemos mencionado, durante el entrenamiento buscamos optimizar una función de pérdida, por lo tanto, la métrica utilizada para evaluar el desempeño del entrenamiento es el valor de la función de pérdida en cada iteración.

La función de pérdida definida para entrenar tanto el MLP como las variantes de LSTM es

la función de entropía cruzada, la cual definimos anteriormente en la Ecuación 4.18. Por otra parte, para entrenar el LSTM Bidireccional propusimos una función de pérdida con un término de **regularización**. La regularización es una técnica utilizada en aprendizaje automático para reducir el error en la inferencia utilizando un conjunto de datos de prueba, por medio de la inducción de sesgo a cambio del aumento en la varianza (Goodfellow et al., 2016).

Entre los distintos tipos de estrategias de regularización, se encuentra la regularización L_2, la cual modifica un modelo que sobre-ajusta (es decir, que se desempeña bien en los datos de entrenamiento pero no en los de prueba) al añadir una penalización equivalente a la suma de los cuadrados de las magnitudes de sus parámetros. La Ecuación 5.7 describe este tipo de regularización:

$$L = Coste + \lambda \sum ||\mathbf{w}||^2 \tag{5.7}$$

Donde λ es el factor de penalización elegido. Básicamente la Ecuación 5.7 nos dice que el valor de la función de pérdida L será mayor cuando la suma $\sum ||\mathbf{w}||^2$ sea muy grande, es decir, cuando aumente la complejidad del modelo al tener más parámetros.

Las métricas que utilizamos para evaluar el desempeño de nuestro modelo durante la etapa de inferencia son:

- Exactitud
- Matriz de confusión
- Puntaje FORCE

La **exactitud** representa la fracción de predicciones que nuestro modelo realizó de manera correcta. Es la métrica de evaluación más sencilla y está definida por:

$$Exactitud = \frac{Predicciones\ correctas}{Total\ de\ predicciones} \tag{5.8}$$

Una **matriz de confusión** (Figura 5.7) es una herramienta de visualización en aprendizaje automático del rendimiento de un algoritmo en aprendizaje supervisado. Cada fila de la matriz representa las instancias de la clase verdadera, mientras que las columnas representan a las clases estimadas. El rendimiento de un *algoritmo perfecto* tendría valores de 1 a lo largo de la diagonal principal de la matriz de confusión y valores de 0 en el resto de la matriz.

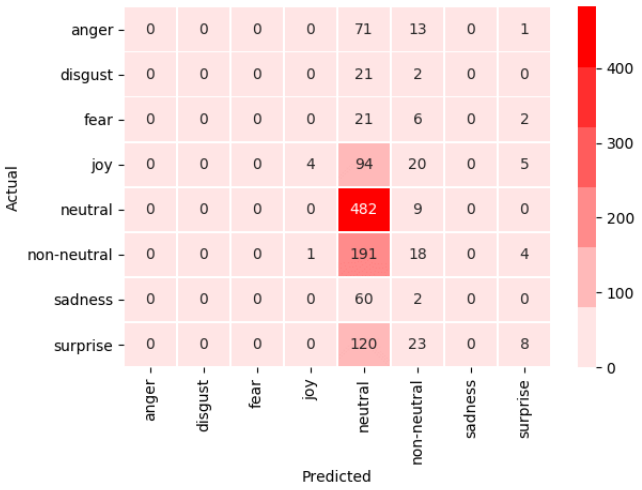


Figura 5.7: Ejemplo de una matriz de confusión. Las filas representan a las clases reales y las columnas a las clases predichas. Tomado de (Torres, 2018)

Por último, el **puntaje FORCE** es una métrica diseñada exclusivamente para medir el desempeño de los algoritmos desarrollados para ser utilizados en los datos de pozos proporcionados en la competencia "FORCE 2020 Machine Learning Contest with Wells and Seismic", creada por el Norwegian Petroleum Dictatorate (https://www.npd.no/en/force/Previous-events/machine-learning-contest-with-wells-and-seismic/).

Para determinar el algoritmo ganador en la competencia, la métrica más importante fue el puntaje FORCE, la cual fue propuesta por los organizadores para medir un grado de incorreción petrofísica de las predicciones de los modelos. Utiliza una matriz de penalización (Figura 5.8) que provee un grado de castigo al modelo por cada tipo de predicción equivocada de litofacies, de esta manera, si el modelo predice una litofacies de lutitas en lugar de margas, no se le penalizará tanto al modelo como si predijera lutitas en lugar de dolomía.

label \ prediction	Sandstone	Sandstone/shale	Shale	Marl	Dolomite	Limestone	Chalk	Halite	Anhydrite	Tuff	Coal	Crystalline Basement
Sandstone	0	2	3.5	3	3.75	3.5	3.5	4	4	2.5	3.875	3.25
Sandstone/Shale	2	0	2.375	2.75	4	3.75	3.75	3.875	4	3	3.75	3
Shale	3.5	2.375	0	2	3.5	3.5	3.75	4	4	2.75	3.25	3
Marl	3	2.75	2	0	2.5	2	2.25	4	4	3.375	3.75	3.25
Dolomite	3.75	4	3.5	2.5	0	2.625	2.875	3.75	3.25	3	4	3.625
Limestone	3.5	3.75	3.5	2	2.625	0	1.375	4	3.75	3.5	4	3.625
Chalk	3.5	3.75	3.75	2.25	2.875	1.375	0	4	3.75	3.125	4	3.75
Halite	4	3.875	4	4	3.75	4	4	0	2.75	3.75	3.75	4
Anhydrite	4	4	4	4	3.25	3.75	3.75	2.75	0	4	4	3.875
Tuff	2.5	3	2.75	3.375	3	3.5	3.125	3.75	4	0	2.5	3.25
Coal	3.875	3.75	3.25	3.75	4	4	4	3.75	4	2.5	0	4
Crystalline Basement	3.25	3	3	3.25	3.625	3.625	3.75	4	3.875	3.25	4	0

Figura 5.8: Matriz de penalización utilizada en la competencia para proveer un grado de castigo para cada modelo al realizar predicciones equivocadas. Tomado de https://xeek.ai/challenges/force-well-logs

Finalmente, el puntaje FORCE es calculado con la siguiente expresión:

$$S = -\frac{1}{N}\sum_{i=0}^{N} \boldsymbol{A}\hat{y}_i y_i \tag{5.9}$$

Donde N es el número de muestras, A es el puntaje de la predicción en la matriz de penalización, $\hat{y}_i$ es la litofacies predicha y y_i la verdadera. Por lo tanto, mientras más cercano a 0 esté el valor absoluto de S, se le considerará al modelo mejor que uno con valor absoluto de S mayor.

Capítulo 6

Resultados

En esta sección se muestran los resultados de los experimentos más significativos, dado que queremos mostrar aquellos que proporcionen una contribución sustancial al entendimiento y futura solución del problema. Si bien, se realizaron gran cantidad de pruebas, especialmente probando combinaciones diferentes de hiperparámetros como taza de aprendizaje y factor de regularización L_2, sólamente incluimos los resultados de aquella combinación de parámetros que produjo un mejor valor de exactitud, esto con el fin de mantener esta sección tan sucinta como sea posible.

Para la realización de dichos experimentos, tanto con datos reales, como con datos sintéticos, designamos tres tipos de modelos para la comparación de los resultados: MLP, LSTM simple y LSTM Bidireccional. Se decidió utilizar estas tres variantes de modelos, descritas en la Sección 5.2, con el fin de comparar un modelo sencillo (MLP), contra un modelo basado en redes recurrentes (LSTM) y con un modelo más complejo de redes recurrentes (BiLSTM).

6.1. Datos sintéticos

Estos experimentos se llevaron a cabo con el objetivo de medir el rendimiento de los modelos anteriormente descritos en un conjunto de datos menos complejo que uno real, más pequeño, con menos clases por clasificar y mayor variabilidad entre las mismas. La generación de este conjunto de datos sintéticos fue descrita en la Sección 5.1.

El conjunto de datos de entrenamiento consiste en ocho *pozos* con 2,000 muestras cada uno, es decir, un total de 16,000 datos de entrenamiento. Por otra parte, se utilizaron dos pozos más, ambos con 2,000 muestras cada uno, para validación y prueba, respectivamente. El conjunto de datos de validación fue utilizado para medir el desempeño del modelo durante el entrenamiento, mientras que el conjunto de datos de prueba fue utilizado para medir el desempeño final del modelo y obtener las métricas correspondientes.

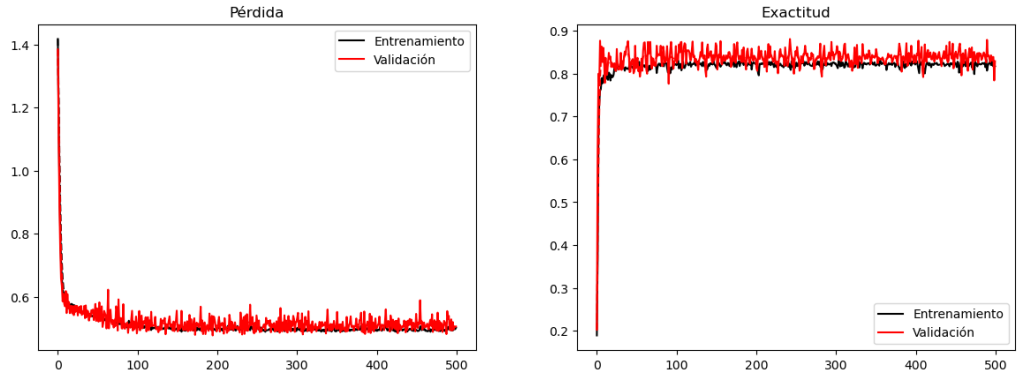
6.1.1. MLP

Se entrenó un MLP con los siguientes hiperparámetros:

Épocas	**Función de Pérdida**	**Taza de aprendizaje**	**Weight Decay**
500	Entropía Cruzada	0.001	0.01

Tabla 6.1: Combinación de hiperparámetros utilizada para entrenar un MLP con un conjunto de datos sintéticos.

Donde las **épocas** son el número de rondas de entrenamiento que se aplicaron al modelo, el tipo de **optimizador** es el algoritmo utilizado para actualizar los parámetros del modelo, y el **weight decay** es el factor que controla el grado de regularización L_2 aplicado a la función de pérdida.



(a) Función de pérdida (b) Exactitud

Figura 6.1: Curvas de pérdida (a) y exactitud (b) para el MLP entrenado en un conjunto de datos sintéticos.

La Figura 6.1 nos muestra en (a) el comportamiento de la función de pérdida con respecto a las épocas de entrenamiento, mientras que (b) muestra los valores de exactitud obtenidos. Podemos notar un comportamiento deseado en ambas gráficas, ya que tanto para el conjunto de datos de entrenamiento, como de validación, obtenemos un comportamiento casi igual. Sin embargo, hay que considerar que tanto los datos de entrenamiento, como de validación y prueba provienen de la misma distribución de datos, por lo que el comportamiento anteriormente descrito es el esperado.

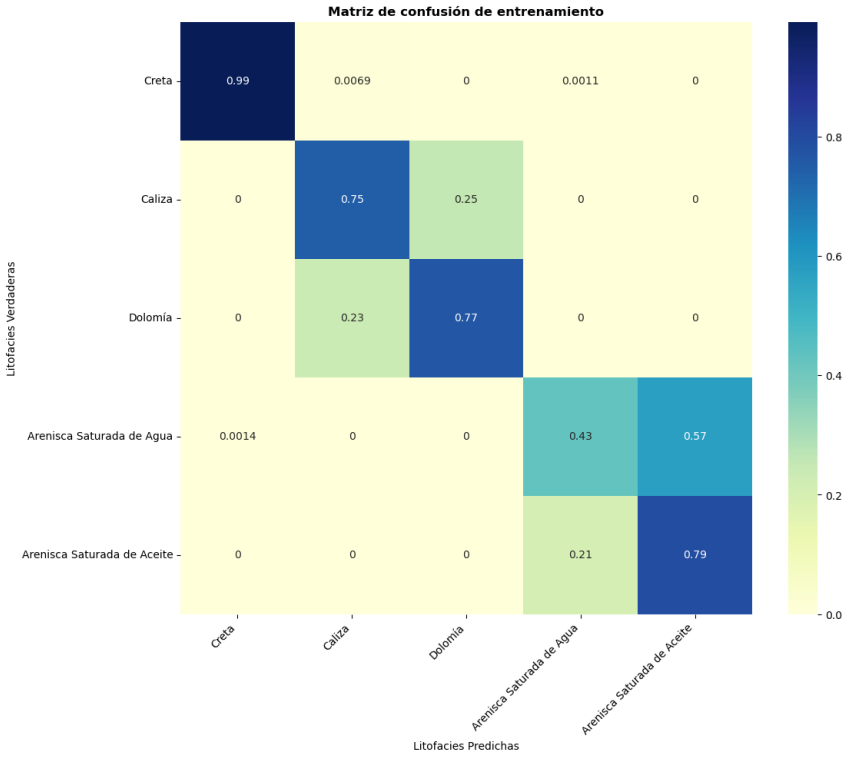


Figura 6.2: Matriz de confusión obtenida para los datos de entrenamiento por un MLP.

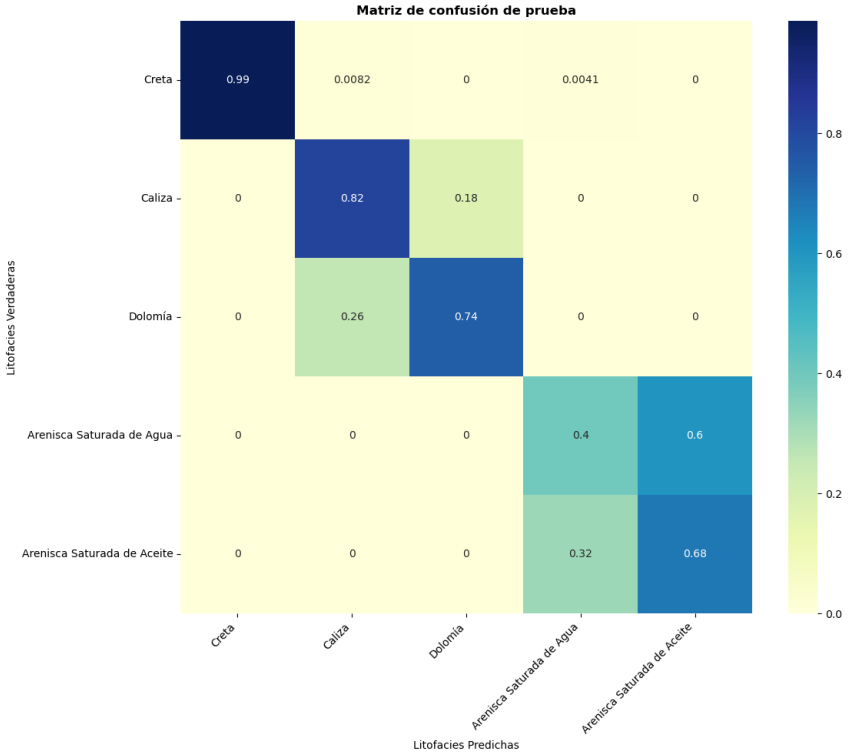


Figura 6.3: Matriz de confusión obtenida para los datos de prueba por un MLP.

La Figura 6.2 y la Figura 6.3 son las matrices de confusión obtenidas para el conjunto de datos de entrenamiento y de prueba, respectivamente. En ambas podemos notar que la clase *Creta* se predice correctamente en casi todos los casos, mientras que la *Arenisca Saturada con Agua*, se

confunde con *Arenisca Saturada con Aceite* en un 60 % de los casos aproximadamente. Si regresamos a la Figura 5.2 podemos notar que los datos se crearon de tal manera que a cualquier modelo le sea muy dificil diferenciar entre estas dos clases.

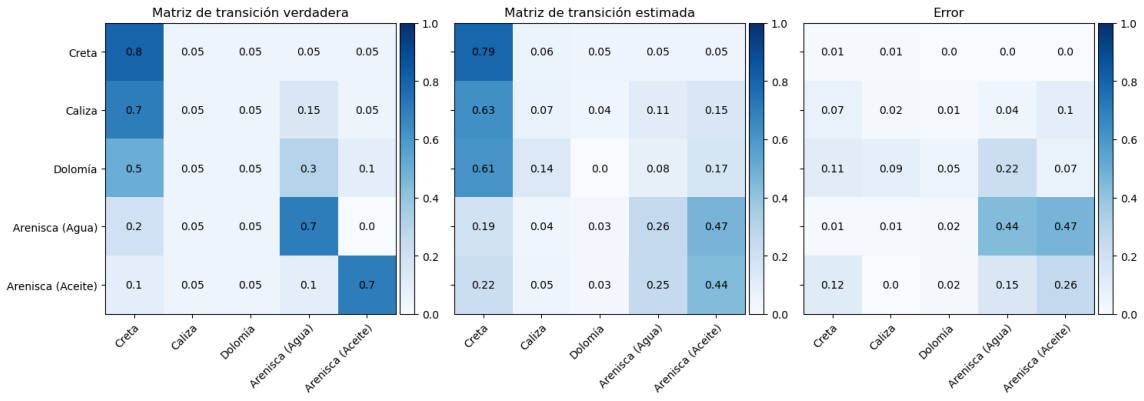


Figura 6.4: Comparación entre la matriz de transición verdadera del conjunto de datos sintéticos (Izquierda), la matriz de transición estimada a partir de las predicciones del MLP en el conjunto de datos de prueba (Centro) y la matriz del error calculado entra las dos matrices anteriores.

Se estimó una matriz de transición a partir de las predicciones que se hicieron sobre los datos de prueba (Figura 6.4, en medio). Recordemos que los datos de prueba fueron modelados a partir de una matriz de transición propuesta en la Ecuación 5.4, y se muestra en la Figura 6.4, a la izquierda. En la misma figura se incluyó un la diferencia absoluta calculada entre las dos matrices anteriormente mencionadas, donde podemos observar que el mayor error del modelo proviene en las transiciones entre areniscas.

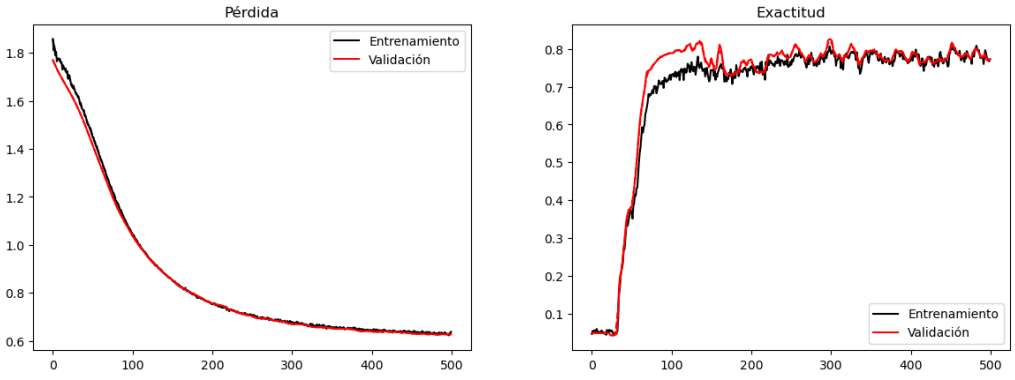
Es importante recalcar que se eligió presentar el error entre la matriz de transición verdadera y la estimada de la manera en que se muestra en la Figura 6.4 porque consideramos que presentar el valor de la norma de la matriz de error *enmascara* la contribución al error de cada transición especiífica. Al presentarlo en una matriz, podemos visualizar de una mejor manera la contribución particular de cada transición al error y de esta manera identificar la situación que le da mayor problema a nuestros modelos.

6.1.2. LSTM

Utilizamos los siguientes hiperparámetros para entrenar un LSTM sobre nuestro conjunto de datos sintéticos:

Épocas	**Función de Pérdida**	**Taza de aprendizaje**	**Weight Decay**
500	Entropía Cruzada	0.001	0.01

Tabla 6.2: Combinación de hiperparámetros utilizada para entrenar un LSTM con un conjunto de datos sintéticos.



(a) Función de pérdida (b) Exactitud

Figura 6.5: Curvas de pérdida (a) y exactitud (b) para el LSTM entrenado en un conjunto de datos sintéticos.

Observamos en la Figura 6.5, un comportamiento óptimo, como en el entrenamiento del MLP. Este es un comportamiento que esperamos de igual manera en el LSTM Bidireccional, ya que seguimos evaluando con un conjunto de datos sintéticos.

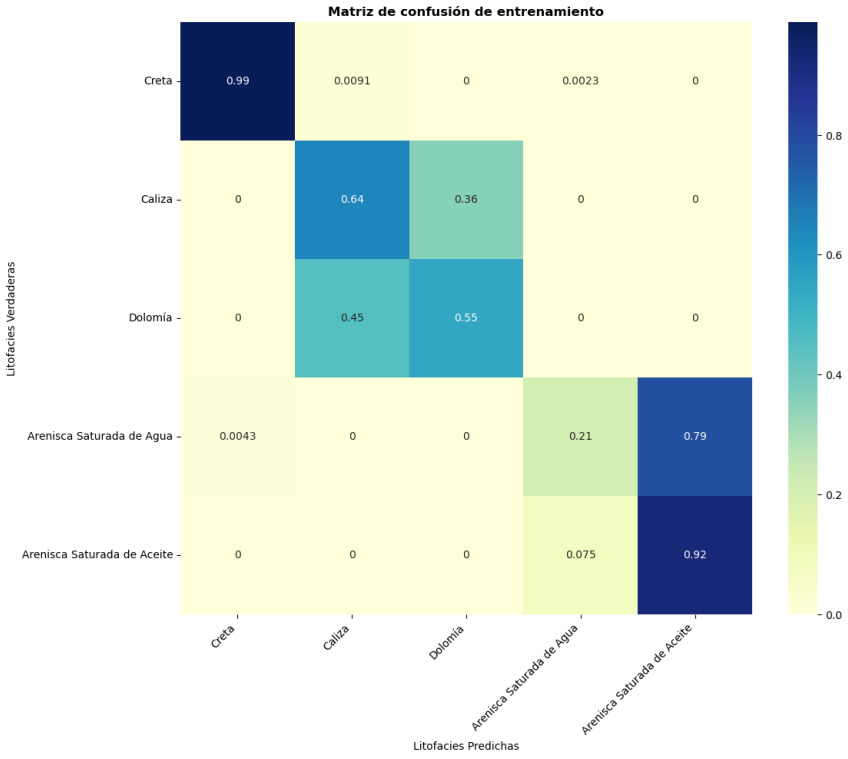


Figura 6.6: Matriz de confusión obtenida para los datos de entrenamiento por un LSTM.

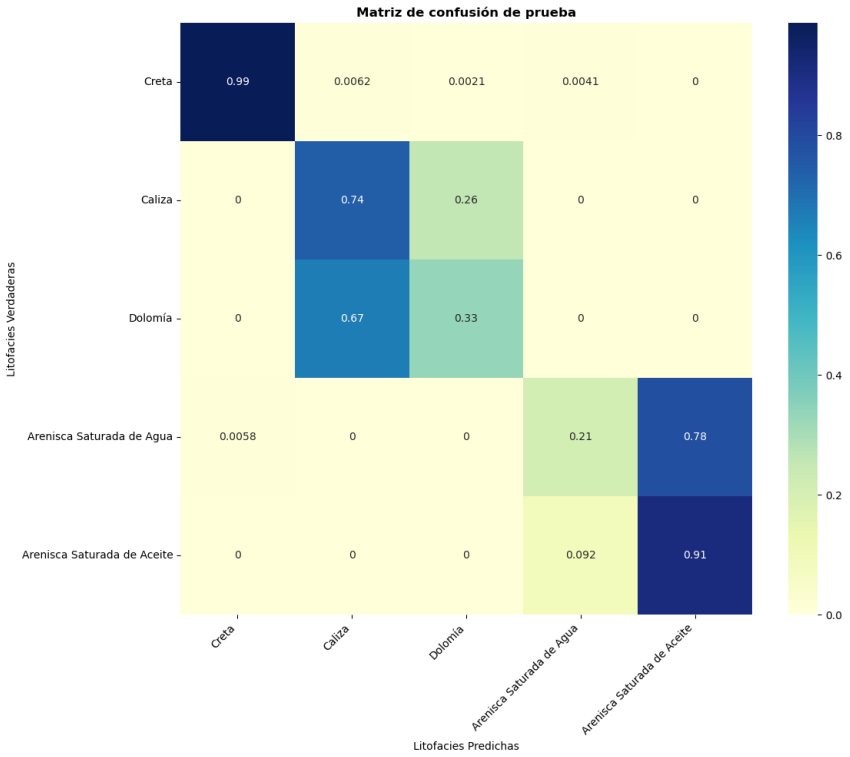


Figura 6.7: Matriz de confusión obtenida para los datos de prueba por un LSTM.

En cuanto a las matrices de confusión (Figura 6.6 y Figura 6.7), observamos que el LSTM simple tiene una capacidad muy limitada para distinguir correctamente entre *areniscas* y para identificar efectivamente a la *dolomía*.

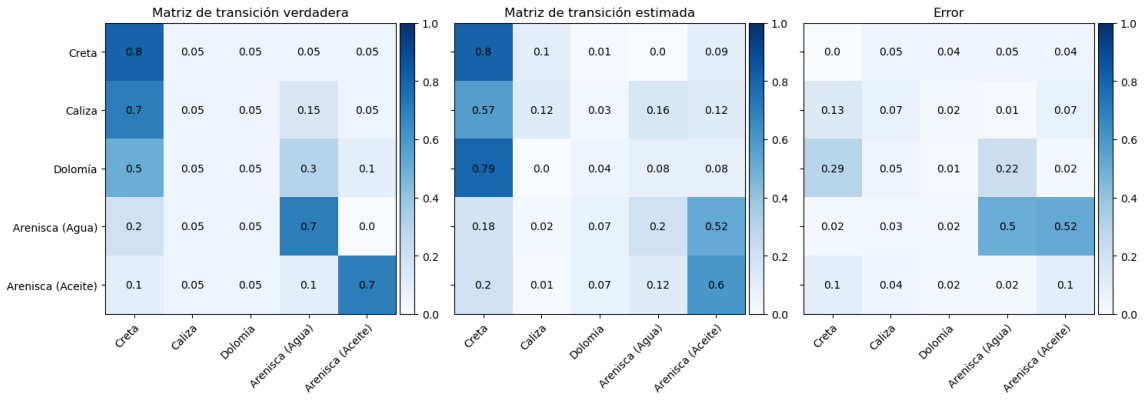


Figura 6.8: Comparación entre la matriz de transición verdadera del conjunto de datos sintéticos (Izquierda), la matriz de transición estimada a partir de las predicciones del LSTM en el conjunto de datos de prueba (Centro) y la matriz del error calculado entra las dos matrices anteriores.

La matriz de transición estimada para el LSTM simple (Figura 6.8) confirma lo mostrado anteriormente por las matrices de confusión. El error obtenido para captar correctamente las transiciones de la *arenisca saturada con agua* es mayor en un LSTM simple; esto puede deberse a que el proveer la información de las instancias anteriores en la secuencia de litofacies es insuficiente para realizar las predicciones correctamente.

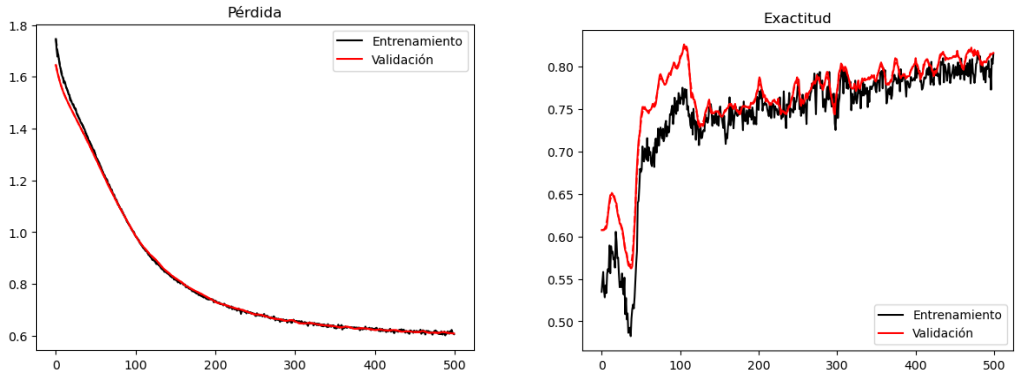
6.1.3. BiLSTM

Finalmente, en cuanto a los experimentos con datos sintéticos se refiere, entrenamos un LSTM Bidireccional con los siguientes hiperparámetros:

Épocas	**Función de Pérdida**	**Taza de aprendizaje**	**Weight Decay**
500	Entropía Cruzada	0.001	0.01

Tabla 6.3: Combinación de hiperparámetros utilizada para entrenar un BiLSTM con un conjunto de datos sintéticos.

El comportamiento de la función de pérdida es muy similar a los casos anteriores. Por otra parte, dada la complejidad añadida de un LSTM Bidireccional, es más complicada la convergencia a un valor de exactitud, como se muestra en la Figura 6.9 (b).



(a) Función de pérdida (b) Exactitud

Figura 6.9: Curvas de pérdida (a) y exactitud (b) para el BiLSTM entrenado en un conjunto de datos sintéticos.

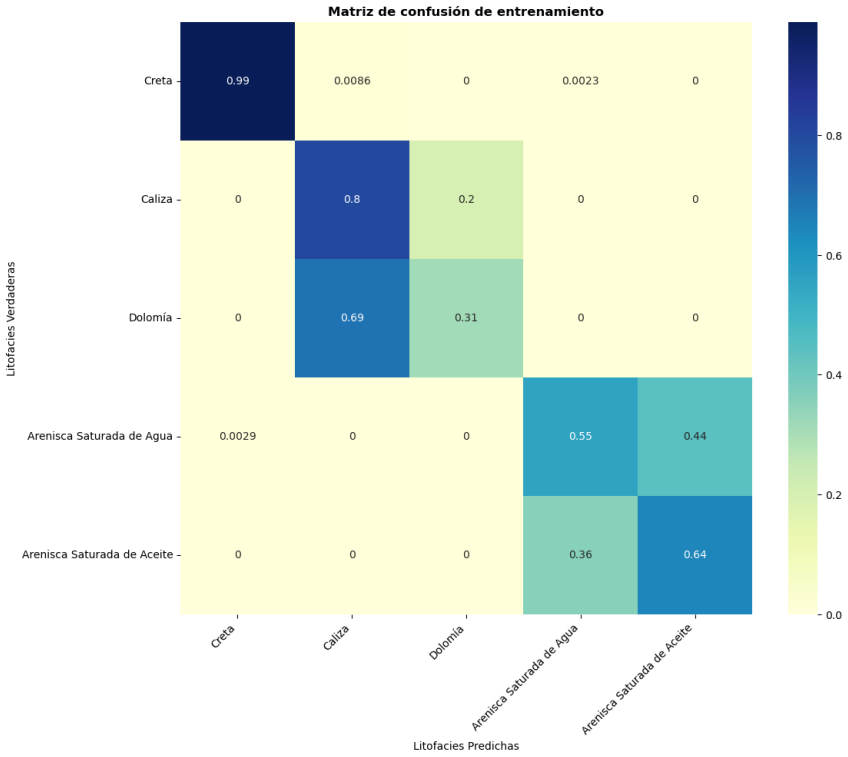


Figura 6.10: Matriz de confusión obtenida para los datos de entrenamiento por un BiLSTM.

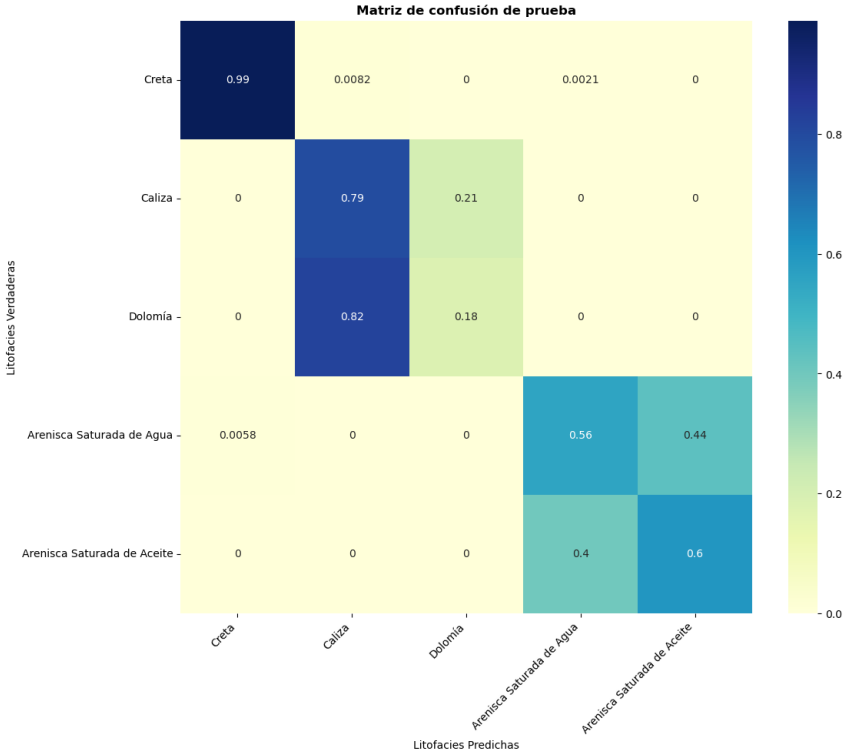


Figura 6.11: Matriz de confusión obtenida para los datos de prueba por un BiLSTM.

Es importante considerar, que los resultados mostrados por las matrices de confusión (Figura 6.10 y Figura 6.11) corresponden a la métrica principal que queremos mejorar, pero si nos limitamos a considerar sólo esta métrica estaremos omitiendo algunas de las capacidades añadidas

al utilizar modelos distintos, como veremos inmediatamente en la Figura 6.12. Si comparamos las matrices de confusión del LSTM simple con la del LSTM Bidireccional, podemos notar que ambas muestran la dificultad presentada en dichos modelos para distinguir entre *areniscas*.

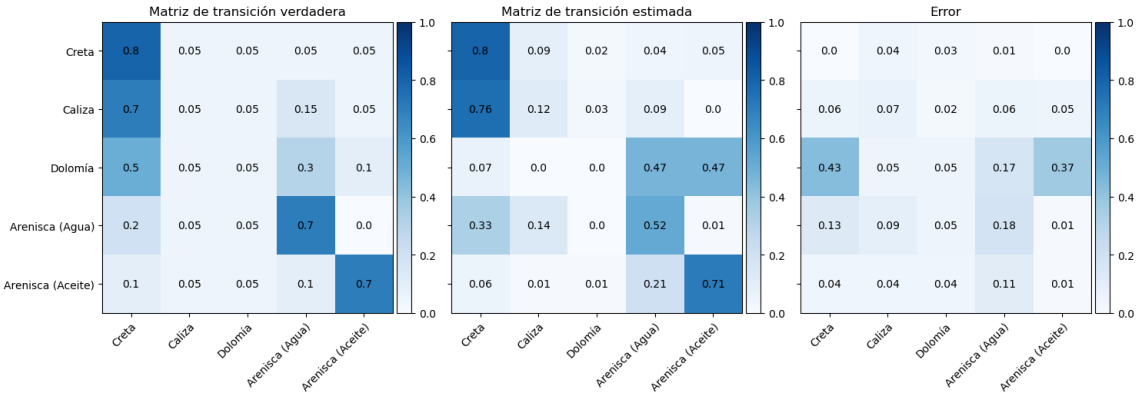


Figura 6.12: Comparación entre la matriz de transición verdadera del conjunto de datos sintéticos (Izquierda), la matriz de transición estimada a partir de las predicciones del BiLSTM en el conjunto de datos de prueba (Centro) y la matriz del error calculado entra las dos matrices anteriores.

Una cuestión muy importante para resaltar a partir de la Figura 6.12, es que si bien las matrices de confusión anteriores no muestran una mejoría en el rendimiento del modelo con respecto a los anteriores, la matriz de transición estimada refleja las capacidades que se añadieron al utilizar un LSTM Bidireccional. Notemos que la probabilidad de transición verdadera de *arenisca saturada con agua* a *arenisca saturada con aceite* fue establecida como 0 desde un inicio. Como se esperaba, un LSTM Bidireccional fue capaz de estimar de buena manera esta probabilidad de transición (1 %), al ahora contar con información más completa de la secuencia de litofacies.

6.1.4. Comparación de resultados

A continuación, mostramos los valores de exactitud para los modelos descritos anteriormente:

Modelo	**Exactitud**
MLP	0.816
LSTM	0.771
BiLSTM	0.823

Tabla 6.4: Comparación entre los resultados numéricos obtenidos sobre el conjunto de datos sintéticos.

A priori, el BiLSTM tiene el valor de exactitud más alto, aunque no es significativamente mayor a la exactitud obtenida por el MLP. Basarse exclusivamente en el valor de exactitud obtenido para juzgar el rendimiento de cada modelo puede orillarnos a descartar completamente a un modelo por su baja exactitud, o viceversa, y podríamos perder de vista las características propias del modelo que lo pueden hacer más útil en algún otro escenario.

Para visualizar de una manera gráfica el rendimiento de cada modelo, tomamos un mismo subconjunto del conjunto de datos de prueba y lo utilizamos para realizar inferencia en cada modelo, tal como se muestra a continuación:

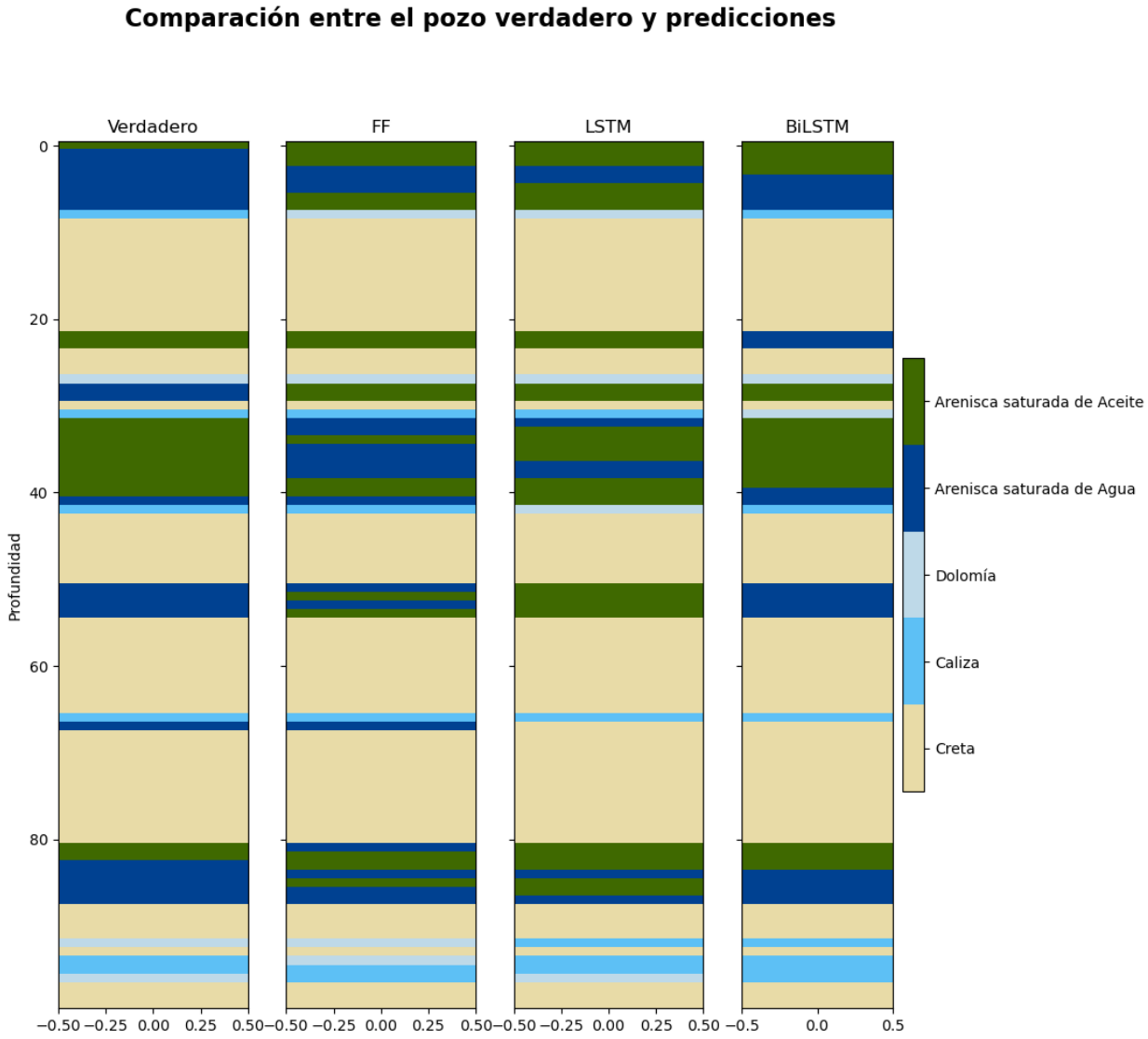


Figura 6.13: Comparación entre las columnas de litofacies verdadera, la obtenida por el MLP, la obtenida por el LSTM y el BiLSTM para el conjunto de datos sintéticos.

Podemos observar a partir de la figura anterior, que los tres métodos probados tienen problemas para distinguir entre los dos tipos de areniscas, como lo hemos venido mencionando con anterioridad, sin embargo, cada uno de ellos presenta su *indecisión* de una manera distinta.

Si nos enfocamos a las predicciones del MLP o *FF (Feed-Forward)*, observamos que a excepción de la *creta* le es difícil ser consistente en sus predicciones y propicia la alta alternancia entre clases. Por ejemplo, si nos enfocamos en la zona entre las profundidades de 30 a 40 unidades, podemos notar que le es complicado al modelo predecir correctamente un bloque de *arenisca saturada con aceite*, ya que la confunde constantemente con *arenisca saturada con agua*, propiciando una alternancia entre estas clases. Dicho lo anterior, la matriz de transición estimada para este modelo (Figura 6.4) cobra más sentido, ya que podemos notar que esta alta alternancia entre clases propicia el alto error en la transición estimada de *arenisca saturada con agua* a *arenisca saturada con aceite*.

Por otra parte, vemos que el LSTM trata de distinguir mejor entre areniscas, provocando que la alternancia vista para el modelo de FF no sea tan constante, sobre todo en la región de pro-

fundidad entre 50 y 54 unidades, donde simplemente se *decide* por una clase, aunque sea errónea.

Por último, el BiLSTM es el modelo que presenta un valor mayor de exactitud, y podemos ver en la comparación con el pozo verdadero que produce predicciones más consistentes que el par anterior de modelos. A su vez, es de suma importancia notar que el BiLSTM logra aprender de manera muy efectiva la nula transición de *arenisca saturada con agua* a *arenisca saturada con aceite*, ya que no presenta ese caso en la predicción, justificando así los resultados obtenidos en su matriz de transición estimada.

6.2. Datos de FORCE

Se utilizó un segundo conjunto de datos pre-procesados de entrenamiento y prueba, a partir de pozos reales en la región del Mar del Norte en Noruega, proporcionados por el Norwegian Petroleum Dictatorate con fines de investigación y se hicieron públicos a partir de la competencia internacional de algoritmos de aprendizaje automático para la clasificación de litofacies "FORCE 2020 Machine Learning Contest with Wells and Seismic". A partir de este nuevo conjunto de datos se entrenaron y probaron los tres tipos de modelos utilizados anteriormente con una combinación diferente de hiperparámetros.

El conjunto de datos de entrenamiento contiene 98 pozos, mientras que los de prueba cuentan con 10 pozos. Las curvas de registros utilizadas como entradas del modelo son:

- Caliper
- Resistividad Media
- Resistividad Profunda
- Densidad Aparente
- Rayos Gamma
- Porosidad de Neutrón
- Factor Fotoeléctrico
- Potencial Espontáneo
- Taza de Penetración
- Sónico de Onda Compresional
- Sónico de Onda de Corte

Este conjunto de entradas se encuentra debidamente etiquetado con la litofacies correspondiente a cada nivel de profundidad, correspondiente a una de las doce litofacies existentes en el conjunto de datos, las cuales son:

- Arenisca
- Arenisca Arcillosa
- Lutita
- Dolomía
- Toba

- Marga
- Creta
- Halita
- Carbón
- Caliza
- Anhidrita
- Basamento

La Figura 6.14 muestra un ejemplo de un pozo proveniente del conjunto de datos de entrenamiento:

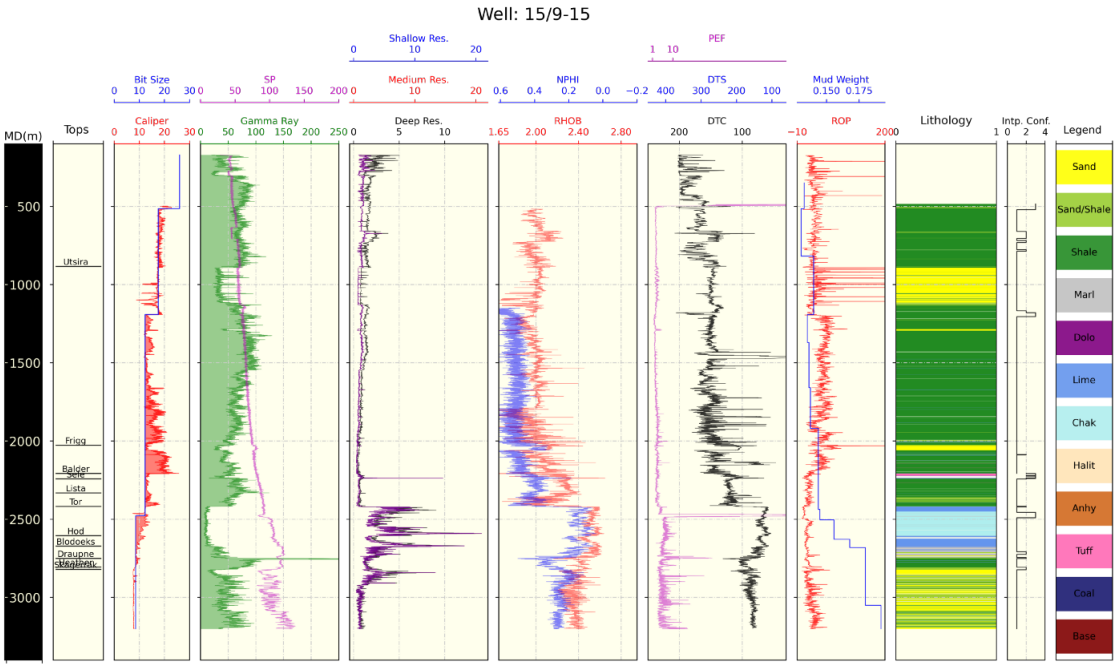


Figura 6.14: Ejemplo de un pozo correspondiente al conjunto de datos FORCE utilizado para entrenamiento. Tomado de https://towardsdatascience.com/lithology-prediction-using-deep-learning-force-2020-competition-dataset-part-1-1ba0264981b4

6.2.1. MLP

Entrenamos un MLP con los hiperparámetros siguientes:

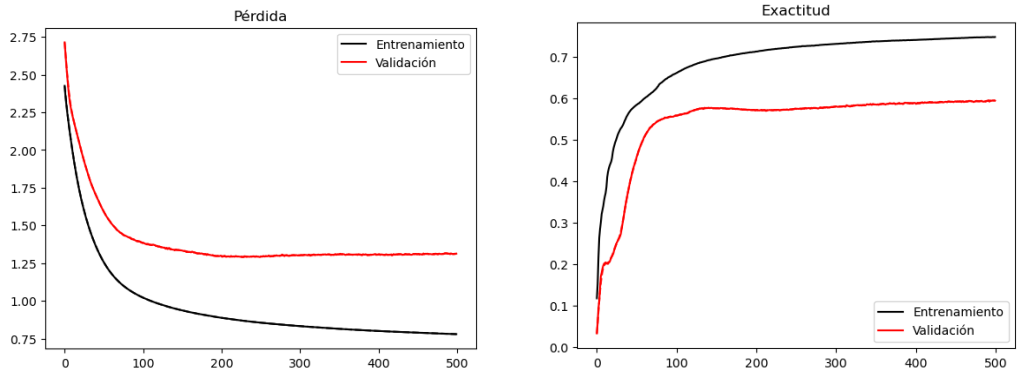
Épocas	Función de Pérdida	Taza de aprendizaje	Weight Decay
500	Entropía Cruzada	0.001	0.01

Tabla 6.5: Combinación de hiperparámetros utilizada para entrenar un MLP con el conjunto de datos FORCE.

A diferencia de los experimentos realizados con datos sintéticos, la curvas presentadas en la

Tabla 6.5 muestran una separación entre el comportamiento para datos de entrenamiento y datos de validación. Esto se debe a que no podemos asegurar que la distribución de los datos de entrenamiento y validación es la misma, ya que son datos reales y de pozos distintos, por otra parte, los datos sintéticos sí provienen de la misma distribución que nosotros propusimos.

A medida que la separación entre las curvas de entrenamiento y validación se hace más grande, esto implica que existe el problema de sobre-ajuste a los datos de entrenamiento o *overfitting*, en el cual, el rendimiento del modelo en datos de prueba disminuye considerablemente con respecto a los datos de entrenamiento. Si bien, existe un cierto grado de sobre-ajuste en este modelo este no es excesivo, por lo que se decidió no utilizar la técnica de regularización conocida como *dropout* en la que se eliminan aleatoriamente un número de neuronas de la red a partir de un valor de probabilidad para reducir la complejidad del modelo.



(a) Función de pérdida (b) Exactitud

Figura 6.15: Curvas de pérdida (a) y exactitud (b) para el MLP entrenado en el conjunto de datos FORCE.

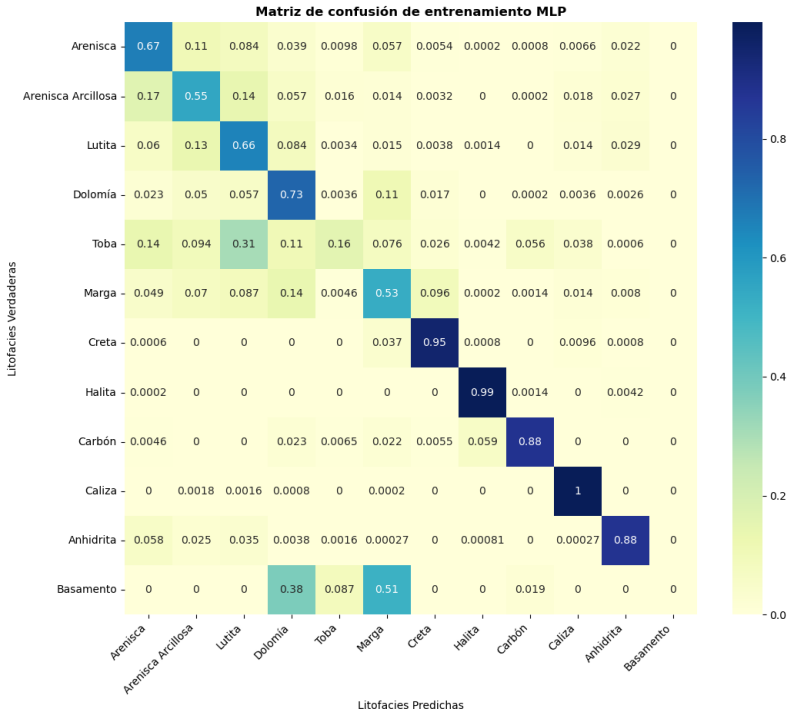


Figura 6.16: Matriz de confusión obtenida para los datos de entrenamiento por un MLP entrenado con el conjunto de datos FORCE.

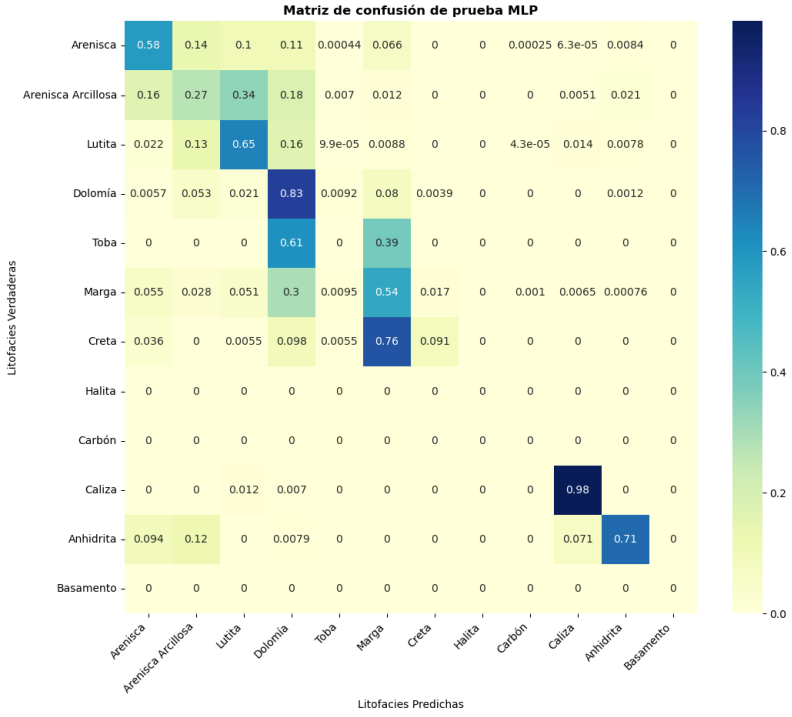


Figura 6.17: Matriz de confusión obtenida para los datos de prueba por un MLP entrenado con el conjunto de datos FORCE.

De igual manera, las matrices de confusión obtenidas (Figura 6.16 y Figura 6.17) reafirman nuestra hipótesis sobre la disminución en el rendimiento del modelo para los datos de prueba, ya que podemos observar que la matriz de confusión en datos de entrenamiento (Figura 6.16) tiene

a sus mayores valores de probabilidad en la diagonal principal, lo cual indica que en la mayor parte de los casos se predice correctamente la litofacies correspondiente.

Sin embargo, la matriz de confusión para datos de prueba no muestra el mismo comportamiento (Figura 6.17), ya que como podemos notar al modelo le es más complicado predecir correctamente las litofacies en datos de prueba.

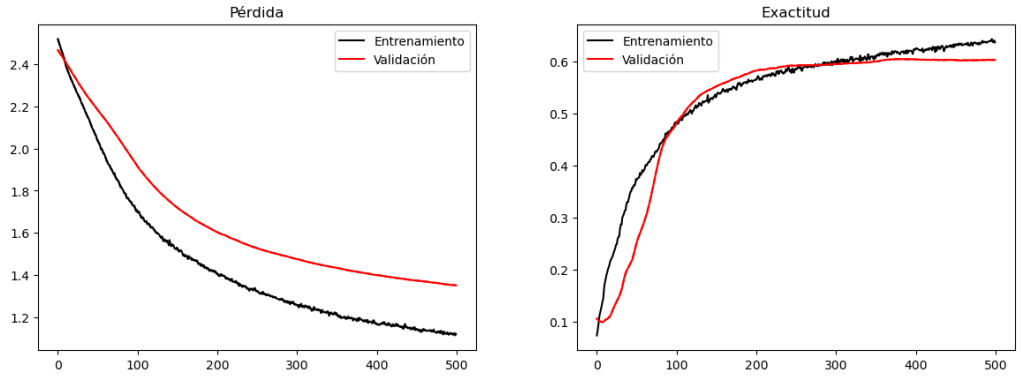
6.2.2. LSTM

Posteriormente entrenamos un modelo LSTM con los siguientes hiperparámetros:

Épocas	**Función de Pérdida**	**Taza de aprendizaje**	**Weight Decay**	**Dropout**
500	Entropía Cruzada	0.01	0.001	0.25

Tabla 6.6: Curvas de pérdida (a) y exactitud (b) para el LSTM entrenado en el conjunto de datos FORCE.

Tal y como lo mencionamos anteriormente, el hiperparámetro *dropout*, es utilizado para contrarrestar el efecto del sobre-ajuste del modelo al reducir la complejidad del mismo eliminando parámetros (neuronas) aleatoriamente.



(a) Función de pérdida (b) Exactitud

Figura 6.18: Curvas de pérdida (a) y exactitud (b) para el LSTM entrenado en el conjunto de datos FORCE.

Como podemos observar en la Figura 6.18, la regularización mediante *dropout* ayudó a reducir un poco los efectos del sobre-ajuste del modelo, aunque esto no quiere decir que los resultados sean mejores. Simplemente, los resultados en el conjunto de datos de prueba serán tan buenos o tan malos como los del conjunto de datos de entrenamiento.

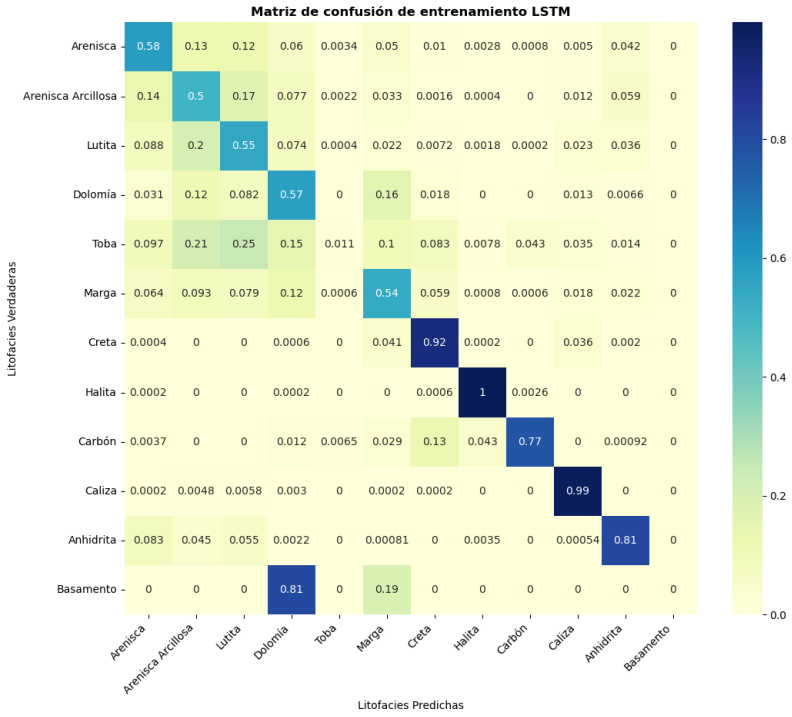


Figura 6.19: Matriz de confusión obtenida para los datos de entrenamiento por un LSTM entrenado con el conjunto de datos FORCE.

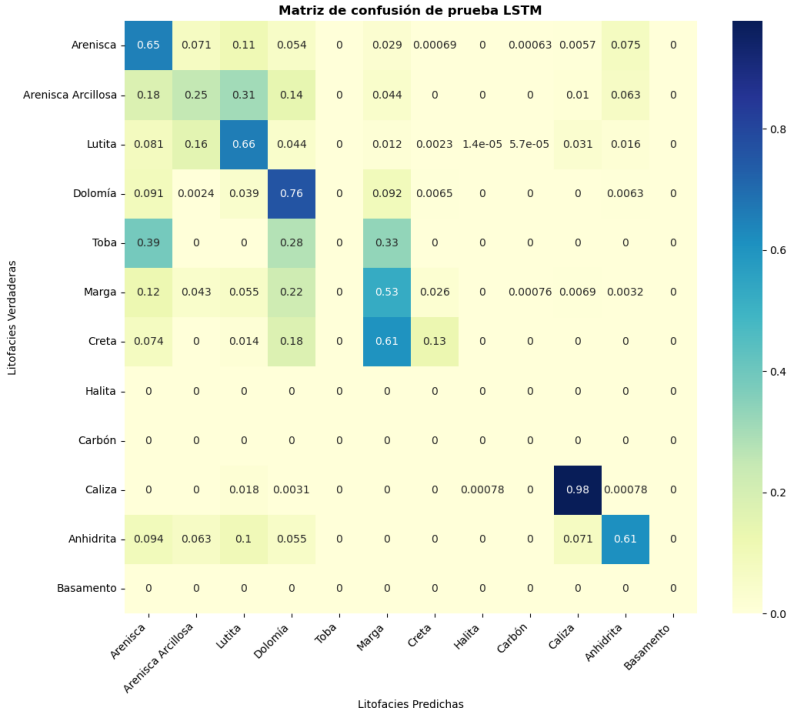


Figura 6.20: Matriz de confusión obtenida para los datos de prueba por un LSTM entrenado con el conjunto de datos FORCE.

Siguiendo la misma tendencia del MLP, las matrices de confusión obtenidas muestran un comportamiento dispar, aunque se haya tratado de emparejar el comportamiento en datos de prueba con los de entrenamiento, para el modelo es aún muy complicado encontrar la manera de predecir

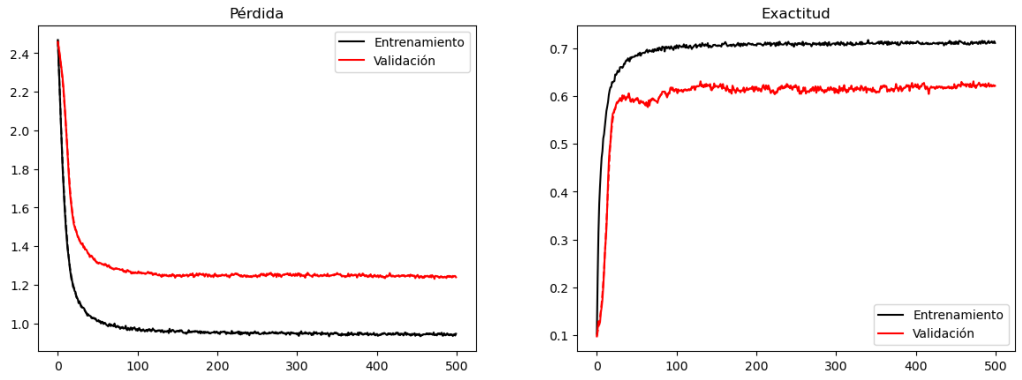
litofacies correctamente en distribuciones de datos distintas.

6.2.3. BiLSTM

Finalmente, entrenamos un LSTM Bidireccional con los siguientes hiperparámetros.

Épocas	Función de Pérdida	Taza de aprendizaje	Weight Decay	Dropout
500	Entropía Cruzada	0.001	0.01	0.25

Tabla 6.7: Curvas de pérdida (a) y exactitud (b) para el BiLSTM entrenado en el conjunto de datos FORCE.



(a) Función de pérdida (b) Exactitud

Figura 6.21: Curvas de pérdida (a) y exactitud (b) para el BiLSTM entrenado en el conjunto de datos FORCE.

Las curvas de pérdida y exactitud (Figura 6.21) muestran que el LSTM Bidireccional logró converger a un conjunto de parámetros óptimos en etapas tempranas del entrenamiento. Esto no es una característica *per se* del LSTM Bidireccional, pero puede deberse a que este modelo se encontró en alguna parte de su entrenamiento con un valor mínimo local en la superficie de pérdida.

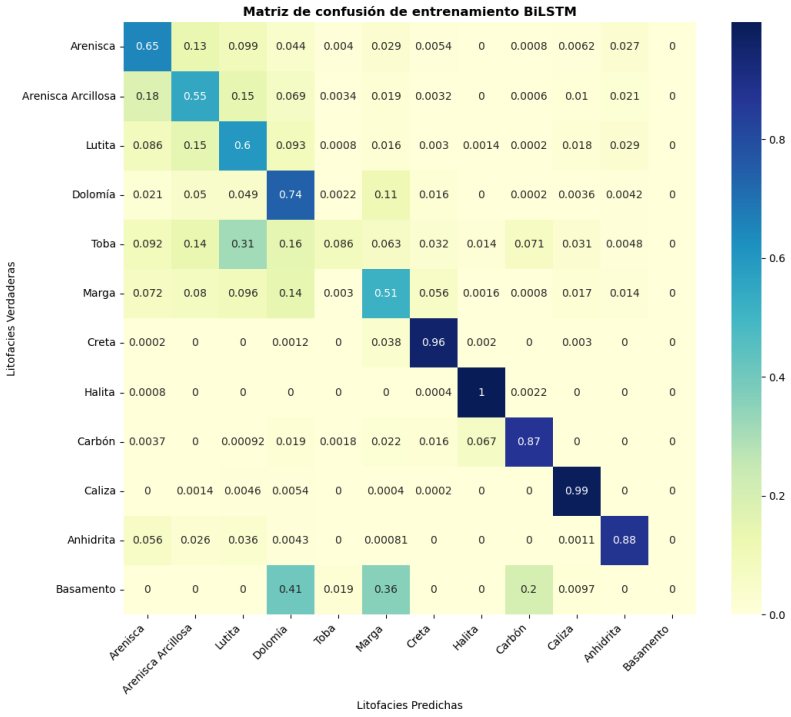


Figura 6.22: Matriz de confusión obtenida para los datos de entrenamiento por un BiLSTM entrenado con el conjunto de datos FORCE.

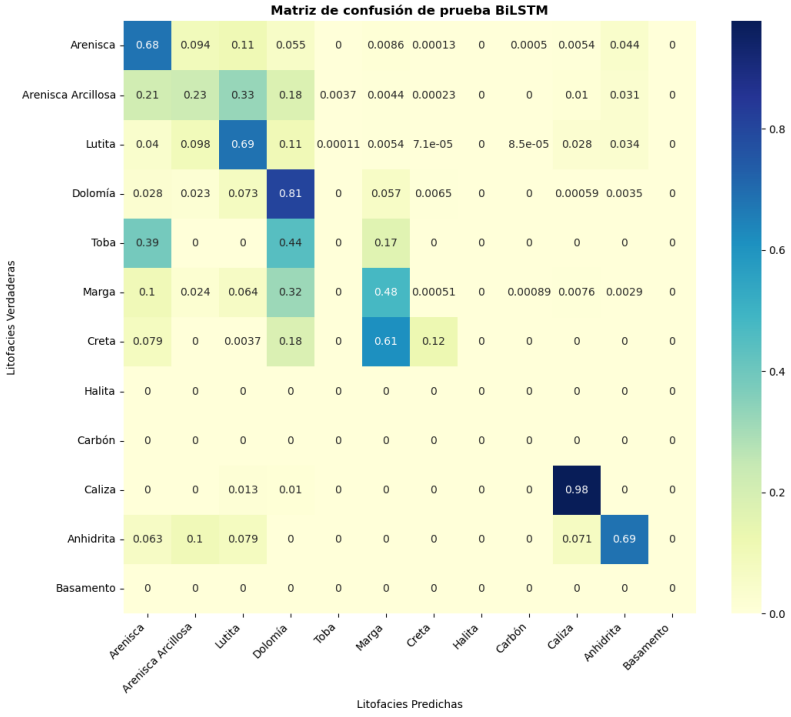


Figura 6.23: Matriz de confusión obtenida para los datos de prueba por un BiLSTM entrenado con el conjunto de datos FORCE.

Por último, las matrices de confusión obtenidas no muestran características significativamente distintas a las anteriores, a excepción de una ligera mejoría al predecir los primeros tres tipos de litofacies (*Arenisca, Arenisca Arcillosa* y *Lutita*), con respecto al LSTM.

6.2.4. Comparación de resultados

La descripción numérica del rendimiento de los modelos anteriores se puede resumir en la siguiente tabla:

Modelo	Exactitud	Puntaje FORCE
MLP	0.59	-0.98
LSTM	0.60	-1.07
BiLSTM	0.62	-0.96

Tabla 6.8: Comparación entre los resultados numéricos obtenidos sobre el conjunto de datos FORCE.

Notemos que al igual que en los datos generados sintéticamente, el LSTM Bidireccional tiene una ligera ventaja sobre los otros dos algoritmos. Tomemos en cuenta, que la métrica de Puntaje FORCE corresponde a un valor de penalización, por lo que mientra más cercano sea este valor a cero, se tendrá un mejor puntaje. Si bien, el modelo de LSTM Bidireccional que hemos propuesto tiene un mejor puntaje FORCE que los otros dos modelos probados, el modelo ganador de la competencia tuvo un puntaje FORCE de -0.46, por lo que conforme a ese parámetro aún es un modelo lejos de ser bueno.

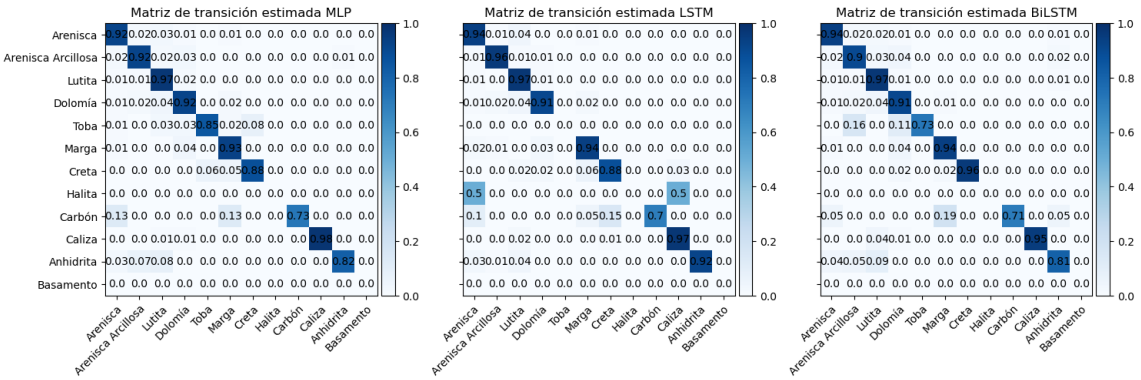


Figura 6.24: Matrices de transición estimadas en el conjunto de datos FORCE para cada uno de los modelos, MLP (Izquierda), LSTM (Centro) y BiLSTM(Derecha).

Al no contar con una matriz de transiciones verdaderas para este conjunto de datos, nos limitamos a presentar sólo las estimaciones de matrices de transición para cada modelo. Podemos observar que a grandes rasgos, los tres modelos convergen a una matriz de transición muy similar, a excepción de algunas clases como es el caso de las transiciones en la *Halita*.

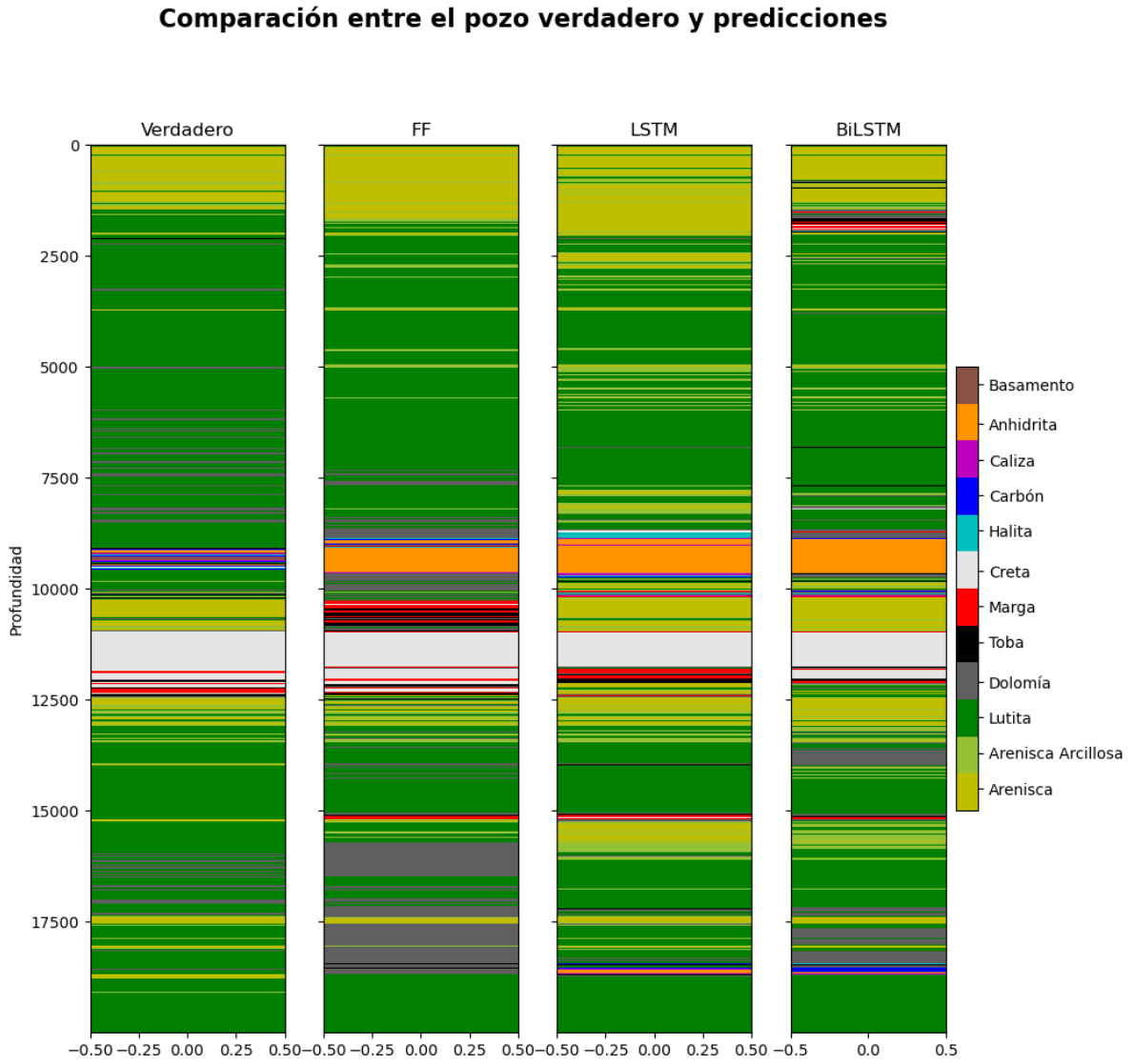


Figura 6.25: Comparación entre las columnas de litofacies verdadera, la obtenida por el MLP, la obtenida por el LSTM y el BiLSTM para el conjunto de datos FORCE.

Como comparación final, tenemos en la Figura 6.25 las predicciones obtenidas por los tres modelos en un subconjunto de datos de prueba y los valores verdaderos que les corresponden. Podemos notar una capacidad generalizada en los tres modelos entrenados de identificar columnas extensas de *lutita*, a su vez, una identificación satisfactoria de la zona donde se encuentra la *creta*. Sin embargo, es más que evidente la incapacidad de dichos modelos de identificar correctamente las intercalaciones de *dolomía* presentes en los datos verdaderos. A su vez, en regiones de intercalaciones complejas entre varias litofacies, como en los niveles de profundidad entre 9,000 y 10,000 unidades, los tres modelos son incapaces de reconocer estas intercalaciones y optan por predecir una sola litofacies (*anhidrita*).

En el modelo de LSTM Bidireccional, el cual tuvo un aparente mejor desempeño que los demás modelos, no se presentan características sobresalientes con respecto a los otros algoritmos. Su mejor rendimiento en cuanto a métricas puede deberse simplemente a que este modelo es capaz de predecir un poco mejor aquellas litofacies que son más importantes en el cálculo del puntaje FORCE, como la(*Arenisca*,la *Arenisca Arcillosa* y la *Lutita*).

Capítulo 7

Conclusiones

La aplicación de métodos de aprendizaje automático para explorar soluciones alternativas a problemas en distintos campos científicos ha brindado a los especialistas de diferentes áreas del conocimiento nuevas herramientas para atacar dichos problemas con un enfoque basado en la eficiencia y el aprovechamiento de las capacidades computacionales a las que se tiene acceso actualmente.

En cuanto al problema de aplicación explorado en este trabajo, pudimos constatar de primera mano que con los algoritmos particulares que se utilizaron y con la metodología propuesta, si bien obtenemos resultados de una manera considerablemente más eficiente con respecto a la interpretación de registros realizada por un intérprete humano, no lo es más en términos de confiabilidad.

Sin embargo, el objetivo de este estudio no es el de determinar si un algoritmo de aprendizaje automático puede realizar una tarea sobre un registro geofísico de pozo, de mejor manera que un intérprete humano. Contrariamente, lo que se desea es lograr entrenar algoritmos capaces de agilizar el trabajo del intérprete humano, a partir de una base para la interpretación de los registros, donde el humano sea capaz de utilizar la predicción del algoritmo para reforzar, robustecer y agilizar su propia interpretación.

El desempeño de los algoritmos propuestos en los experimentos realizados se ve limitado por diversas cuestiones. Hablando específicamente del desempeño en el conjunto de datos sintéticos creado, es necesario tomar en cuenta que la propia generación del conjunto de datos fue realizada de tal manera que obtuvieramos datos que fueran difíciles de clasificar para cualquiera de estos algoritmos. Esto con el propósito de explorar qué tan bien se puede clasificar un conjunto de datos complejo utilizando algoritmos muy sencillos.

Tanto el MLP propuesto para dichos experimentos como las variantes del LSTM pueden considerarse como algoritmos sencillos, ya que no se implementaron mecanismos especiales para tratar de adaptarse mejor a este conjunto de datos en particular. La motivación principal detrás de la proposición de estos algoritmos tan sencillos es que puedan ser implementados y reutilizados a manera de *plug-and-play* (conectar y usar) por intérpretes humanos con amplios conocimientos en geofísica pero poco conocimiento en temas de aprendizaje automático.

Al evaluar los resultados obtenidos en el conjunto de datos sintéticos se pudo comprobar el beneficio añadido de utilizar un algoritmos de redes neuronales recurrentes, específicamente el LSTM Bidireccional, para aprender secuencias de datos de manera más robusta. Esta capacidad se vio resaltada al momento de evaluar el error entre la matriz de transición estimada con la verdadera, donde se pudo observar que el LSTM Bidireccional fue capaz de aprender sin necesidad de añadir algún mecanismo extra la restricción en la transición entre las litofacies *arenisca*

saturada de agua a *arenisca saturada de aceite*. Recordemos que los valores de probabilidad en esta transición estimados por el MLP y el LSTM simple fueron de 0.47 y 0.52, respectivamente, mientras que el LSTM Bidireccional fue capaz de aproximarse casi exactamente al valor real de 0 al estimar una probabilidad de transición de 0.01.

En cuanto a los resultados obtenidos en el conjunto de datos reales, tomemos en cuenta que éste supone un reto mucho más complejo para estos algoritmos sencillos. Como referencia, en la competencia antes mencionada donde se probaron distintos algoritmos para clasificar este conjunto de datos, las propuestas con mejores puntajes corresponden a ensambles complejos de modelos, algoritmos basados en *gradient boosting*, redes neuronales con arquitecturas muy complejas, como los *Transformers*, entre otros.

Por nuestra parte, se buscó mantener la filosofía de utilizar sólo modelos sencillos como los utilizados en el conjunto de datos sintéticos, bajo la misma motivación de entrenar algoritmos fácilmente explicables y reutilizables. No obstante, como pudimos observar en los resultados presentados, para cualquiera de los modelos probados clasificar este conjunto de datos reales implica un reto muy complicado debido a la extrema complejidad de los datos y a la sencillez misma de los algoritmos.

Los resultados obtenidos sirven como base de comparación para futuros esfuerzos en el diseño de algoritmos que solucionen el problema de la clasificación de litofacies a partir de registros de pozo. Es importante resaltar de este trabajo que un LSTM Bidireccional sin modificaciones considerables a la arquitectura o al tratado de los datos, logró obtener un 62 % de exactitud en un conjunto de datos reales, mientras que los modelos ganadores de la competencia obtuvieron entre 75 y 80 por ciento de exactitud, después de realizar una extensa búsqueda de hiperparámetros, aumentaciones en los datos, cambios importantes en arquitecturas de redes neuronales y múltiples validaciones de modelos. El balance entre tiempo de desarrollo, eficacia y complejidad de un modelo de aprendizaje automático tiene que ser tomado en cuenta, sobre todo si se desarrolla con el objetivo de buscar la adopción de este tipo de métodos en campos de aplicación nuevos, como es el caso de la geofísica.

Referencias

Aggarwal, C. C. (2018). *Neural networks and deep learning*. Springer.

Akinnikawe, O., Lyne, S., y Roberts, J. (2018, 07). Synthetic Well Log Generation Using Machine Learning Techniques. , *Day 2 Tue, July 24, 2018*.

An, Y., Guo, J., Ye, Q., Childs, C., Walsh, J., y Dong, R. (2021). Deep convolutional neural network for automatic fault recognition from 3d seismic datasets. *Computers Geosciences*, *153*, 104776.

Asquith, G. B., y Gibson, C. R. (1982). *Basic well log analysis for geologists* (Vol. 3). American Association of Petroleum Geologists Tulsa.

Bassiouni, Z. (1994). *Theory, measurements, and interpretation of well logs* (Vol. 4). Society of Petroleum Engineers.

Bjorlykke, K. (2015). *Petroleum geoscience: From sedimentary environments to rock physics* (2nd ed.). Noruega: Springer.

Boggs, S. (2006). *Principles of sedimentology and stratigraphy* (4th ed.). Pearson Prentice Hall.

Consolvo, B., Consolvo, B., Docherty, P., y Uwaifo, J. (2021). Deep learning for salt body detection: A practical approach. *82nd EAGE Annual Conference Exhibition,*, *2021*(1), 1-5.

Degrave, J., Felici, F., Buchli, J., Neunert, M., Tracey, B., Carpanese, F., ... Riedmiller, M. (2022). Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, *602*(7897), 414-419. doi: 10.1038/s41586-021-04301-9

Dubois, M. K., Bohling, G. C., y Chakrabarti, S. (2007). Comparison of four approaches to a rock facies classification problem. *Computers Geosciences*, *33*(5), 599-617.

Eidsvik, J., Mukerji, T., y Switzer, P. (2004). Estimation of geological attributes from a well log: An application of hidden markov chains. *Mathematical Geology*.

Ellis, D. (1987). *Well logging for earth scientists* (2nd ed.). Springer.

Farrag, A., Ebraheem, M., Sawires, R., Ibrahim, H., y Khalil, A. (2019). Petrophysical and aquifer parameters estimation using geophysical well logging and hydrogeological data, wadi el-assiuoti, eastern desert, egypt. *Journal of African Earth Sciences*, *149*, 42-54.

Goodfellow, I., Bengio, Y., y Courville, A. (2016). *Deep learning*. MIT Press. (http://www.deeplearningbook.org)

Grana, D., Azevedo, L., y Liu, M. (2020). A comparison of deep machine learning and monte carlo methods for facies classification from seismic data. *Geophysics*.

Graves, A. (2012). *Supervised Sequence Labelling with Recurrent Neural Networks*. Berlin: Springer. Descargado de https://cds.cern.ch/record/1503877

Hochreiter, S., y Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Kim, S., Kim, K. H., Min, B., Lim, J., y Lee, K. (2020). Generation of synthetic density log data using deep learning algorithm at the golden field in alberta, canada. *Geofluids*, *2020*, 1-26.

Kim, Y., Hardisty, R., Torres, E., y Marfurt, K. J. (2018). Seismic-facies classification using random forest algorithm. , 2161-2165.

Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., ... Vinyals, O. (2022). Competition-level code generation with alphacode. Descargado de https://arxiv.org/abs/2203.07814

Mavko, G., Mukerji, T., y Dvorkin, J. (2009). *The rock physics handbook: Tools for seismic analysis of porous media* (2.ª ed.). Cambridge University Press.

Mcculloch, W., y Pitts, W. (1943). A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, *5*, 127–147.

Miall, A. (2015). *Stratigraphy: A modern synthesis* (1st ed.). Springer.

Mitchell, T. M. (1997). *Machine learning* (Vol. 1). McGraw-hill New York.

Murphy, K. P. (2022). *Probabilistic machine learning: An introduction*. MIT Press. Descargado de probml.ai

N.A.C.O.S.M. (2005, 11). North American Stratigraphic Code. *AAPG Bulletin*, *89*(11), 1547-1591.

Olah, C. (2015). *Understanding lstm networks* (Inf. Téc.). (https://colah.github.io/posts/2015-08-Understanding-LSTMs/)

Reading, H. (2011). *Sedimentary environments: Processes, facies and stratigraphy* (3rd ed.). Blackwell Publishing.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408.

Rudman, A. (1978). Analysis of geophysical logs from the hawaii geothermal project well. *Geothermal Resources Exploration in Hawaii*(6).

Rumelhart, D. E., Hinton, G. E., y Williams, R. J. (1986). Learning Representations by Back-propagating Errors. *Nature*, *323*, 533–536. Descargado de http://www.nature.com/articles/323533a0

Rzóska, J. (1953). Bait shyness, a study in rat behaviour. *The British Journal of Animal Behaviour*, *1*(4), 128-135.

Satter, A., y Iqbal, G. (2016). Reservoir life cycle and role of industry professionals. , 127-136.

Schlumberger. (2022). *Slb energy glossary: Api unit.* Descargado 2022-08-31, de https://glossary.slb.com/en/Terms/a/api_unit.aspx

Serra, O. (1984). *Fundamentals of well-log interpretation: 1: the acquisition of logging data.* Amsterdam: (Translated by Westaway, P. and H. Abbott): Developments in Petroleum Science, 15A, Elsevier Science Publishers.

Shalev-Shwartz, S., y Ben-David, S. (2014). *Understanding machine learning - from theory to algorithms.* Cambridge University Press.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., ... Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489.

Torres, J. (2018, 01). Emotionx-jtml: Detecting emotions with attention. , 56-60.

Tschannen, V., Delescluse, M., Rodriguez, M., y Keuper, J. (2017). Facies classification from well logs using an inception convolutional network. *ArXiv*, *abs/1706.00613*.

Varhaug, M. (2016). The defining series: Basic well log interpretation. *Schlumberger Oilfield Review*.

Wu, P.-Y., Jain, V., Kulkarni, M. S., y Abubakar, A. (2018). Machine learning–based method for automated well-log processing and interpretation. En *Seg technical program expanded abstracts 2018* (p. 2041-2045).

Zhang, D., Chen, Y., y Meng, J. (2018). Synthetic well logs generation via recurrent neural networks. *Petroleum Exploration and Development*, *45*(4), 629-639. Descargado de https://www.sciencedirect.com/science/article/pii/S1876380418300685

Zhang, L., y Zhan, C. (2017). Machine learning in rock facies classification: An application of xgboost. , 1371-1374.