Detección automática de fallas geológicas en datos sísmicos utilizando aprendizaje profundo

Tesis de Grado de Geofísica

Felipe Corbetta

Director Académico: Dr. Julián Luis Gómez
Co-Director: Dr. Emilio Camilión

Facultad de Ciencias Astronómicas y Geofísicas
Universidad Nacional de La Plata

Agradecimientos

No alcanzarían las páginas para agradecer, pero aqui van algunos:

A Julián Gómez, una persona que ejerce la docencia en todo ámbito de la vida...

A los jurados Gustavo Vergani y a Veronica Martinez que siempre estuvieron a mi disposición...

A mis padres y hermano, solo ellos y yo sabemos lo que pasamos...

A Maca quién fue mi sostén este último tiempo...

A Sofi por el diseño de las imágenes.... **(LinkedIn)**

Y a TODOS los que aportaron su granito de arena para que esto se pueda realizar...

GRACIAS!

Índice general

Resumen

En esta tesis se aplicó un algoritmo de segmentación para detectar fallas geológicas en datos sísmicos. El algoritmo pertenece a la familia del aprendizaje automático y utiliza una red neuronal de correlación en base a la arquitectura de codificación y decodificación, U-Net. Se trata de una arquitectura de redes neuronales profundas comúnmente usada en aplicaciones de segmentación de imágenes de distintos origenes (medicina, biología, sísmica, etc.). U-Net se adaptó para analizar datos sísmicos y detectar fallas geológicas. El algoritmo se entrenó con datos sintéticos para luego usar en la detección de fallas en datos reales. El objetivo de esta tesis es probar las técnicas de aprendizaje automatico en problemas geofísicos, modificar la red original y desglosarla para entender cómo funciona usando código libre (Python) y librerias como TensorFlow y Keras. En el capítulo 1 se determinaron los objetivos, el valor de las fallas para la sismica de reflexion y se hizo un raconto historico de métodos usados para detectarlas. Luego, en el capítulo 2, se describieron las operaciones básicas de estas redes y se profundizó en el concepto de U-Net. En los cápitulos siguientes 3 y 4 se mostró el proceso de entrenamiento en cubos sinteticos y detección en datos 3D reales. Por último, en el capítulo 5, se presentan conclusiónes y trabajos a futuro.

Los resultados de esta tesis de grado confirman el potencial de las técnicas de aprendizaje automático para ser utilizadas en la detección de fallas geológicas en datos de campo 3D al demostrar que el algoritmo tiene un nivel de generalizacion muy alto acelerando el proceso con respecto a anteriores métodos. Pudiendo tener tambien importantes implicancias para la industria de la exploración de recursos

naturales. La detección de fallas geológicas utilizando técnicas de aprendizaje automático, puede acelerar el proceso de exploración y reducir el riesgo de perforaciones fallidas. Como también puede ayudar a eliminar interpretaciones subjetivas del intérprete en fallas sutiles o poco definidas al ojo humano.

Introducción

1.1. Objetivos

Esta tesis tiene como objetivo principal investigar y aplicar una técnica de aprendizaje automático en el ámbito de la geofísica. En particular, se busca probar la eficacia de esta técnica en la resolución de un problema en particular que es la detección de fallas. Además, se pretende realizar modificaciones en una red neuronal, la de [Wu et al., 2019], para adaptarla a las necesidades particulares del usuario y analizar su funcionamiento en detalle. Para lograr esto, se empleará código libre y se desglosará la red neuronal para comprender su estructura y operación.

1.2. Motivación: Fallas geológicas

Las fallas geológicas son discontinuidades en las rocas producto de las ruptura de las mismas con desplazamiento de sus bloques, son importantes porque permiten inferir la evolución del entorno de tensión o deformación [Fossen, 2016]. También juegan un papel crítico en la exploración subsuelo con métodos sísmicos como por ejemplo exploración geofísica. Son vitales para comprender la migración de petróleo y gas y el sellado de yacimientos [Hardman and Booth, 1991]. Ademas son un factor importante en el almacenamiento o entrampamiento de dióxido de carbono. La detección y el seguimiento de posibles fallas en el subsuelo son cruciales para determinar la viabilidad, la integridad del sellado y la seguridad a largo plazo de un reservorio objetivo [Vilarrasa and Carrera, 2015]. En la exploración geotérmica, particularmente en los sistemas hidro-geotérmicos, los

sistemas de fallas conectados actúan como vías para la inyección, el flujo y la extracción de fluidos geotérmicos [Gan and Elsworth, 2014];[Gao et al., 2021].

Convencionalmente, la interpretación de fallas a partir de datos de reflexión sísmica es un proceso manual que depende fuertemente de la experiencia de los intérprete [Alcalde et al., 2017];[Hale, 2012]. La detección de fallas a partir de datos sísmicos de buena calidad es relativamente objetiva pero suele ser un tanto repetitiva y requiere de bastante tiempo al ser ejecutada manualmente. Por ello, la interpretación de un único conjunto de datos puede tardar semanas o hasta meses en completarse [Hale, 2012].

En cuanto a aprendizaje profundo, muchos estudios previos han desarrollado métodos considerando la detección de fallas como un problema de clasificación de imágenes [Di et al., 2018a]; [Pochet et al., 2018]; [Zhang et al., 2019]; [Cunha et al., 2020].

1.3. Detección automática de fallas

Para la interpretación sísmica, la detección automática de fallas basada en el aprendizaje automático puede lograr resultados notablemente superiores en tiempo de entrega en comparación con los métodos convencionales.

Recientemente, se han introducido métodos de redes neuronales convolucionales para detectar fallas con múltiples atributos sísmicos [Huang et al., 2017]; [Di et al., 2018a]; [Guitton, 2018] ;[Guo et al., 2018] ;[Zhao and Mukhopadhyay, 2018]. Por ejemplo, el método de [Wu et al., 2018] debe elegir una ventana o un cubo local para realizar la predicción de fallas, en cada muestra de la imagen la CNN que se usa necesita muchos

ejemplos para generalizar y solo lo hace de a parches pequeños.

Otros estudios tratan el reconocimiento de fallas como una tarea de segmentación de imágenes ([Wu et al., 2019]) y aplican una red de segmentación denominada U-Net [Ronneberger et al., 2015 La segmentación de imágenes divide una imagen en varias regiones. En el trabajo en el que se basa la tesis [Wu et al., 2019] entrenan una U-Net 3D usando datos sintéticos y luego analizan los resultados con datos sintéticos y reales.

1.4. Metodología

Se va a utilizar una red neuronal de correlación o *correlational neural network* (CNN), para realizar la detección de fallas en datos sísmicos 3D.

Para ello, simplifica el modelo original de la CNN siguiendo los lineamientos de [Wu et al., 2019], lo que ahorra significativamente el costo computacional sin detrimento de la capacidad de detección.

El problema de segmentación considerado sufre de lo que se conoce como desbalance de clases. Este desbalance ocurre debido a que los puntos asociados a fallas son una minoría respecto a la cantidad total de puntos en los cubos de entrenamiento. Para esto existen funciones de costo específicas.

Se entrena el modelo con cubos sísmicos sintéticos. El modelo es validado considerando las predicciones arrojadas para cubos sintéticos que no fueron utilizados en el proceso de optimización.

Es interesante destacar que si bien el modelo es entre-

nado únicamente con datos sísmicos sintéticos, el modelo optimizado genera resultados aceptables al ser aplicado a cubos sísmicos reales. En el capítulo 4 se aplica a datos de campo 3D.

En este proceso, se utiliza un algoritmo de AA que toma como entrada un cubo sísmico en amplitudes y genera como salida otro cubo sísmico del mismo tamaño, pero con marcadores de fallas o *fault scores* (MF) asociadas a cada una de las muestras del dato sísmico. Estos marcadores son números entre 0 y 1, donde valores más cercanos a 1 indican que esa zona es más probable de ser reconocida como una falla según los datos de entrenamiento utilizados. Es importante tener en cuenta que estos marcadores no representan probabilidades de falla.

Detección automátoca de fallas: su importancia y su historia

2.1. Sismica de reflexión e interpretación sísmica

La sismica de reflexion es útil ya que es el método que permite tener resolución del subsuelo tanto lateral como verticalmente; permite interpretar cómo fue la evolución de la geología a lo largo del tiempo y los elementos que la componen, la geometría de los estratos del subsuelo, su estratigrafía y deformación; determinar niveles de navegación en pozos horizontales (por ejemplo: Vaca Muerta Fig. 2.1) [Conexplo, IAPG, 2022], puede ser también un indicador directo de la presencia de hidrocarburos (por ejemplo: gas).

Figura 2.1: Pozo en Vaca Muerta

La interpretación sísmica se refiere a mapear toda la información posible asociada a la geología y eventualmente a la presencia de fluídos. Una de los rasgos que se mapea son las fallas. Las fallas actuan como conductos para los fluídos.

En los últimos años, el desarrollo del aprendizaje automático (AA) ha permitido la detección automática de fallas en un tiempo de entrega significativamente menor en comparación con los métodos tradicionales, la detección es solo el primer paso en el proceso de interpretación de fallas. Sin embargo, los ángulos de buzamiento más pronunciados generan problemas. Además, al igual que los atributos sísmicos regulares, la red neuronal es sensible al ruido sísmico, lo que puede distorsionar las predicciones de fallas. Por lo tanto, el método debe tratarse como una herramienta de interpretación adicional que necesita un control de calidad. El método puede generar implicaciones valiosas en la caracterización de yacimientos y evaluaciones de integridad de la roca de cubierta para la exploración de hidrocarburos y evaluaciones de sitios de almacenamiento de dioxido de carbono [Wrona et al., 2022].

2.2. DNN vs CNN

Dentro del AA se encuentran distintos tipos de redes neuronales. Las redes neuronales de tipo red neuronal densa o *dense neural network* (DNN) son un modelo comúnmente utilizado para la clasificación de atributos sísmicos en la industria del petróleo y gas. Sin embargo, las CNN han demostrado ser más eficientes en la detección de estructuras sísmicas, ya que son capaces de aprender características y patrones complejos a partir de los datos de entrada, sin depender de la interpretación física de los atributos. Esto

ha llevado a una mayor precisión en la clasificación de fallas y domos salinos [Di et al., 018a].

Hay dos razones por las cuales las CNN tienen una ventaja significativa sobre las DNN. Primero, la CNN genera automáticamente un grupo de *feature maps* o mapas de características y los optimiza mediante su proceso. Mientras que de otra forma en las DNN, se deberia elegir atributos sismicos. En segundo lugar, y lo que es más importante, la clasificación de CNN se basa en parches que incorporan patrones de reflexión sísmica local para construir la relación de mapeo entre las señales sísmicas y las estructuras objetivo [Di et al., 018a]

2.3. Elección de la red U-Net

Una CNN normal puede construir una relación de mapeo más precisa entre las señales sísmicas y las estructuras objetivo, y con menos parámetros. Además, este enfoque basado en parches puede ayudar a reducir la complejidad computacional de la red, ya que no requiere el análisis de todo el conjunto de datos sísmicos a la vez.

Una de las desventajas de las CNN es que al clasificar cada píxel (o muestra) de forma independiente la red puede asignar etiquetas similares a regiones contiguas de la imagen, lo que puede resultar en una redundancia en la clasificación final. Además, al no reducir el tamaño de la imagen de entrada, es decir, al aplicar convoluciones lo unico que hacemos es elevar el número de parámetros. Esto lleva, como deciamos anteriormente a un aumento en el número de parámetros necesarios para la red y también puede dificultar la interpretación de los resultados de la clasificación. Por lo que se opta utilizar un modelo denominado U-Net.

2.4. U-Net

Una U-Net es un tipo particular de CNN cuya arquitectura es específica para realizar segmentación de datos (imágenes) 2D o 3D (volúmenes). Fue desarrollada originalmente para la segmentación de imágenes médicas [Ronneberger et al., 2015], pero se ha utilizado con éxito en otras aplicaciones. Por ejemplo en geofísica para la detección de fallas o domos salinos [Di et al., 2018b];[Wu et al., 2019]. La arquitectura de U-Net consta de dos partes principales como se ve en la Fig.2.2.: la rama codificadora y la rama decodificadora. La parte codificadora se encarga de reducir el tamaño del volumen mediante la aplicación de decimación y convolución (cambiando el *stride* o paso), mientras que la parte decodificadora se encarga de aumentar de la imagen mediante la aplicación de capas de interpolación.

2.5. Profundización sobre la red usada en la Tesis

Una U-Net (Fig. 2.2) es un tipo particular de CNN que contiene una rama de codificadora y una rama decodificadora. A diferencia de las CNN estándar en las que la entrada fluye secuencialmente a través de capas de convolución, U-Net utiliza concatenaciones entre la rama codificadora y la rama decodificadora. Estas parejas de capas de convolución conectadas por conexiones de salto tienen "distancias"diferentes, es decir, una capa codificadora de nivel más alto tiene una conexión de salto con una capa decodificadora de nivel más alto, y una capa codificadora de nivel más bajo tiene una conexión de salto con una capa decodificadora de nivel más bajo, haciendo que la CNN sea visualmente una red neuronal en forma de U concatenada por conexiones de salto largas/cortas entre las ramas co-

dificadora/decodificadora. Notablemente, una conexión de salto conecta un par de capas codificadoras/decodificadoras en el mismo nivel de resolución. [Wu et al., 2019] demostraron que la arquitectura de la U-Net proporciona una mejor precisión de segmentación de fallas en datos sísmicos junto con una mejora en la velocidad en comparación con la detección basada en atributos sísmicos convencionales [Gao et al., 2022].

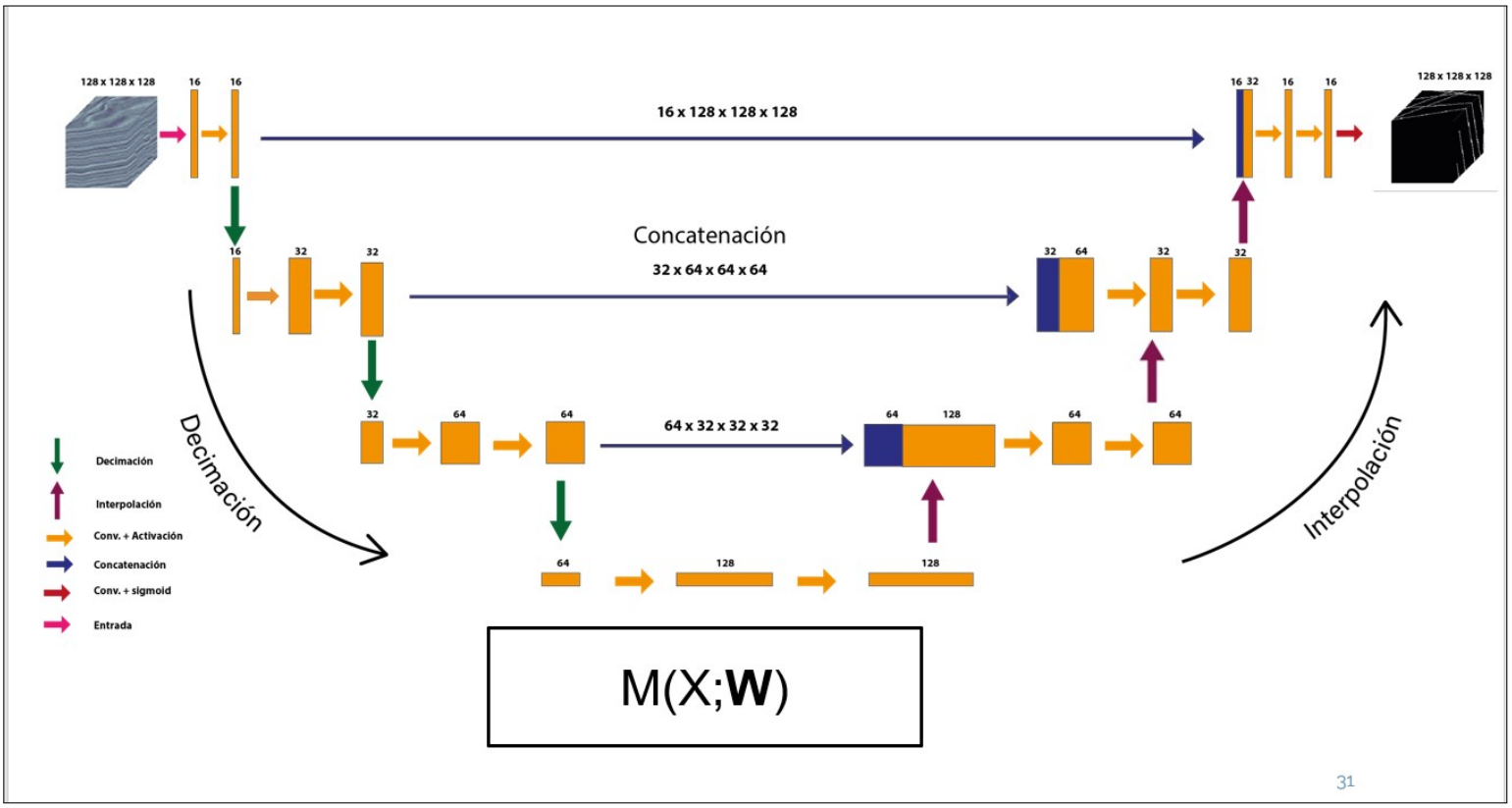

Figura 2.2: U-Net usada en la tesis

Operaciones básicas de una CNN y una U-Net

3.1. Aprendizaje profundo

El aprendizaje profundo o Deep Learning hace referencia a la cantidad de capas tenga el modelo entre la entrada y la salida, esa va a ser la "profundidad"de la que estamos hablando. Bien podria llamarse aprendizaje en capas o *layered representations learning*, aprendizaje jerárquico o *hierarchical representation learning* [Chollet, 2021].

Una de las características más importantes del aprendizaje profundo o *deep learning* (DL) es su capacidad para aprender características sin la necesidad de que el usuario especifique de qué manera deben ser aprendidas. Esto permite que las redes neuronales profundas sean utilizadas en una variedad de aplicaciones, como el reconocimiento de objetos y la generación de imágenes [Nielsen, 2015].

3.2. DNN, representación matemática

Las DNN se utilizan para procesar imágenes, audio y otros tipos de datos. Estas redes están compuestas por varias capas de procesamiento, cada una de las cuales tiene un conjunto de pesos y un vector *bias*. En términos matemáticos, podemos escribir la función de una DNN como una composición de funciones individuales. Cada capa i en la red se puede escribir como una función f_i que toma la entrada de la capa anterior y la transforma en una nueva re-

presentación mediante una combinación afín de los valores de entrada y un conjunto de pesos W_i:

$$h_i = f_i(h_{i-1}, W_i) \tag{3.1}$$

Donde h_i es la salida de la capa i, h_{i-1} es la entrada de la capa i y W_i son los pesos de la capa i. La salida final de la red se puede escribir como:

$$y = f_L(h_{L-1}, W_L) \tag{3.2}$$

Donde y es la salida final de la red, h_{L-1} es la salida de la penúltima capa y W_L son los pesos de la última capa.

Así, la función de una DNN Fig. 3.11 es una composición de funciones individuales que forman las capas de la red, y cada capa utiliza una función de activación no lineal para introducir no linealidades en la red y permitir la modelización de relaciones más complejas entre los datos de entrada y salida.

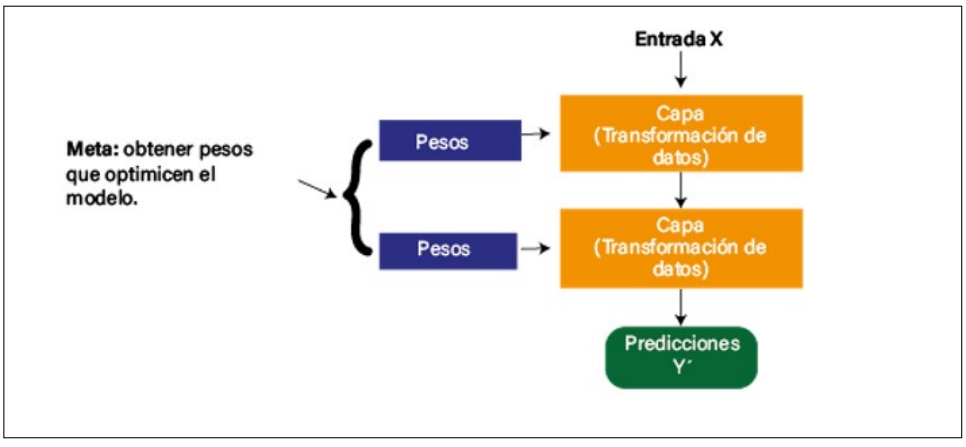

Figura 3.1: Diagrama de Flujo de una DNN

3.3. Operaciones basicas

Resumen de las operaciones de una CNN:

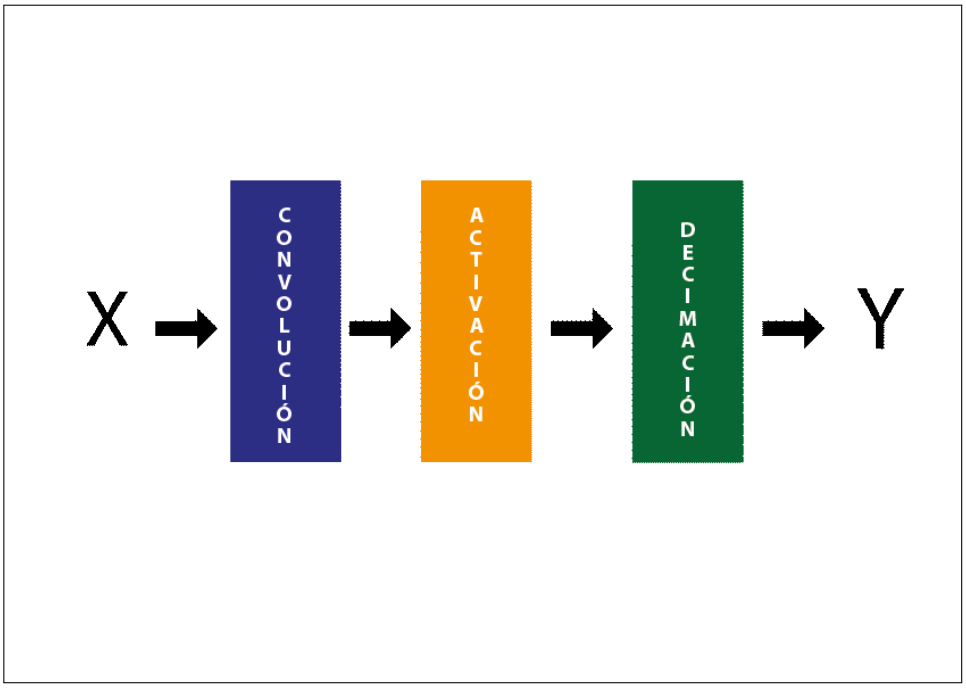

Figura 3.2: Operaciones CNN

Ejemplo

Se va a emplear un ejemplo sencillo de red CNN para mostrar los operadores que están involucrados en la red que se utiliza más adelante en la tesis.

En este ejemplo ilustrativo Fig. 3.3. Lo que se hizo fue usar un filtro Sobel $3x3$ [Padrón, 2022] sobre la imagen normalizada Fig 3.5 y luego se realizaron las operaciones básicas de una CNN como la convolución. A la imagen filtrada, que se suele llamar *feature map* en AA, se le aplicó la función activación ReLu (Rectified Linear Unit) y luego

se decimó.

Los mapas de caracteristicas o *feature maps* (FM) se generan a través de capas de convolución que aplican filtros para extraer características específicas de la imagen de entrada. Cada capa de convolución produce una serie de FM que se utilizan como entrada para la siguiente capa.

Mientras que en el ejemplo se aplica un solo filtro y se obtiene un solo FM para resaltar los bordes de la imagen, en problemas más complejos se requiere el uso de múltiples filtros y capas convolucionales para capturar características más sofisticadas y realizar tareas más desafiantes en el procesamiento de imágenes.

Figura 3.3: Imagen del ejemplo

Las sucesivas capas en las CNN en el aprendizaje profundo tienen filtros cuyos valores son nuestras incognitas.

En una CNN, los filtros utilizados en las primeras capas se asumen comúnmente como detectores de bordes, lo

cual se basa en el conocimiento previo de las características visuales presentes en las imágenes. Sin embargo, es importante tener en cuenta que los valores específicos de los filtros no se conocen de antemano en una CNN.

En lugar de eso, los filtros se tratan como incógnitas o pesos que se aprenden durante el proceso de entrenamiento del modelo. Este aprendizaje se lleva a cabo mediante la optimización de una función de costo adecuada. Durante el entrenamiento, se presenta un conjunto de imágenes etiquetadas al modelo y se procesan a través de las capas convolucionales de la red.

3.4. Desglose de operaciones

Operación Convolución:

La convolución 3D es similar a la correlación 2D Fig. 3.4, donde se utiliza un filtro para deslizar sobre el dato de entrada y multiplicar y sumar en cada subregión de la imagen. En el caso 3D el filtro se traslada en tres dimensiones.En la carrera aplicamos esa oprecación por ejemplo bajo el nombre de ´Filtro correlador´en la materia análisis de señales.

Una correlación 3D se realiza multiplicando el valor de cada punto en el volumen de entrada con el valor correspondiente en el filtro y sumando todos los productos. El resultado es un nuevo volumen con un valor en cada punto, que representa al producto del operador con el dato.

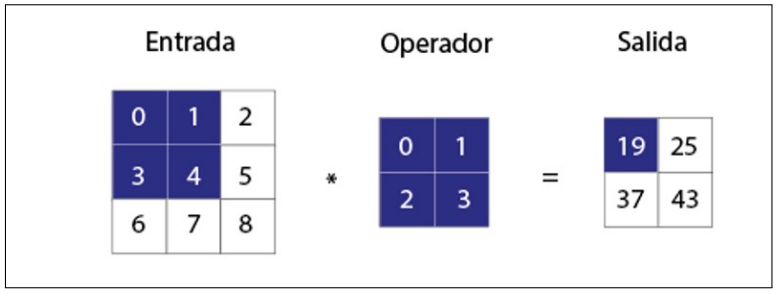

Figura 3.4: Ejemplo gráfico de convolución 2D

Figura 3.5: Aplicación de la convolución con un filtro Sobel 3x3. Der.: FM

Función ReLU (*Rectified Linear Unit*):

La función activación se introduce con el objetivo de la no linealización de red. Se aplica después de cada capa convolucional permitiendo que los valores de activación tomen cualquier valor real en lugar de limitarse a un rango. Esto hace que la CNN aprenda características no lineales (más complejas) de los datos de entrada, lo que mejora su capacidad para realizar tareas de clasificación y reconocimiento de patrones. La función ReLu es una de ellas Fig. 3.6:

$$ReLu(x) = max(x, 0) \tag{3.3}$$

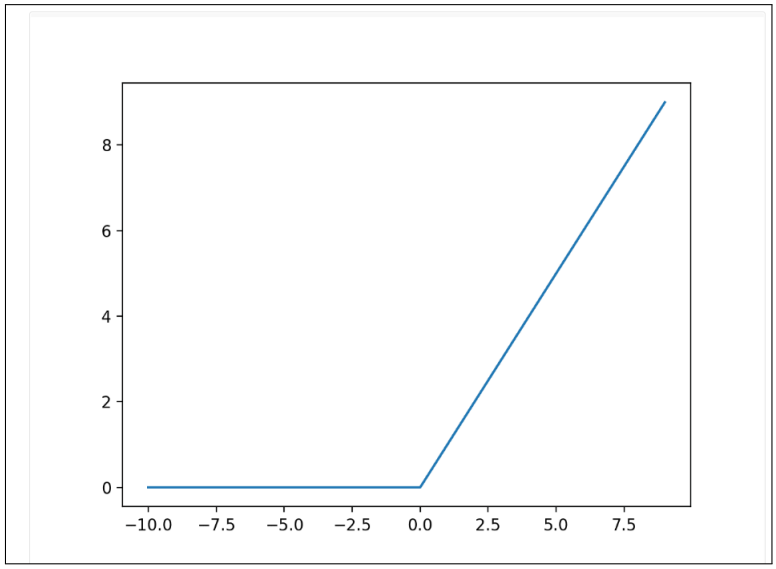

Figura 3.6: Funcion de activación ReLu

Es una función de activación popular en las redes neuronales por varias razones:

Es no-lineal, la función ReLU permite la introducción de no-linealidad en la red neuronal, lo que permite al modelo aprender características más complejas y representaciones

abstractas de los datos.

Es computacionalmente eficiente, la función ReLU es computacionalmente más eficiente que otras funciones de activación ya que solo requiere una comparación y una operación aritmética simple.

Resuelve el problema del gradiente desvaneciente: En las redes neuronales profundas, es común que el gradiente a través de las capas se desvanezca y haga difícil el entrenamiento del modelo. La función ReLU ayuda a resolver este problema ya que el gradiente en los valores positivos es 1, lo que permite que el entrenamiento de las capas superiores continúe.

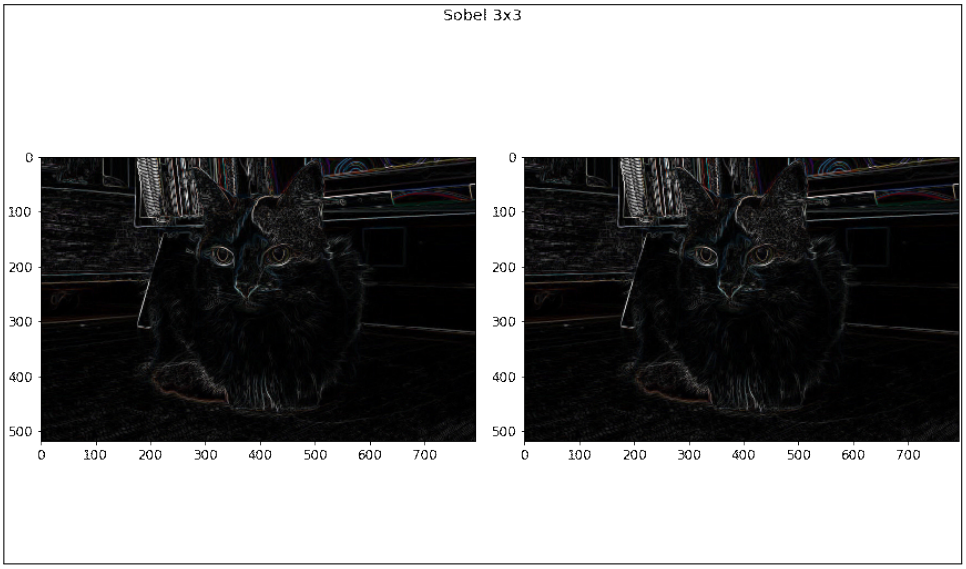

Figura 3.7: Aplicación de la función ReLu a la imagen

Operación decimación:

Para reducir el tamaño del dato y por lo tanto el número

de parámetros necesarios para resolver el problema sin perder informacion importante de los datos se decima. A esta decimación en particular Fig. 3.9, se le denomina *max pooling*: La operación se realiza mediante el uso de un parche que se desliza sobre la imagen de entrada, y se selecciona el valor máximo dentro de cada subregión del parche Fig. 3.8.

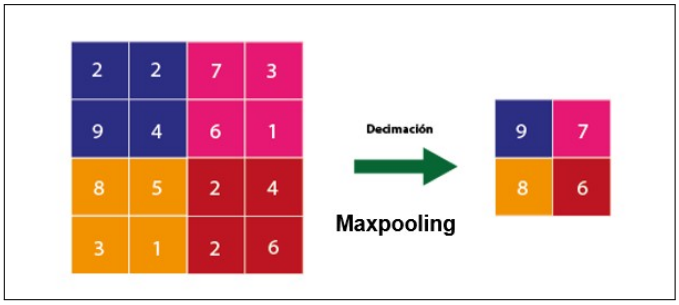

Figura 3.8: Operación decimación

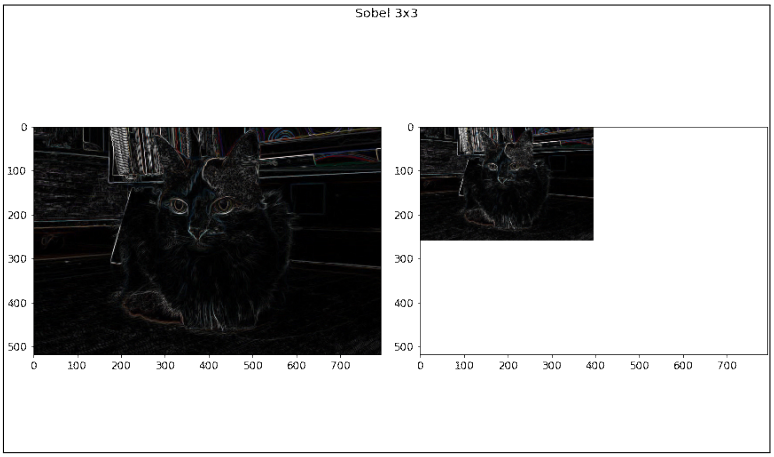

Figura 3.9: Decimación aplicada a la imagen

Función sigmoide

Teniendo en cuenta que hay varias funciones de activacion, en la capa final en esta tesis y en la mayoria de las CNN se usará la función sigmoide:

$$\sigma(z) = \frac{1}{1 + e^{(-z)}} \tag{3.4}$$

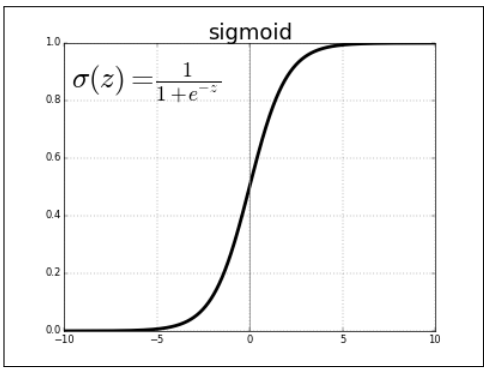

Figura 3.10: Función activación Sigmoide

La función sigmoide Fig 3.10 tiene varias propiedades útiles, como ser diferenciable en todas partes, lo que facilita el cálculo del gradiente durante el entrenamiento de la red neuronal mediante *backpropagation*, que es la forma en la que se ajustan los pesos (como más adelante se explicará). La función sigmoide se utiliza en la última capa de una red neuronal debido a que su rango de salida está en el intervalo de [0, 1]. Esto es beneficioso en problemas de clasificación binaria, ya que nos permite interpretar la salida como una probabilidad o en este caso un indicador de fallas (MF, por sus siglas en inglés).

3.5. Operaciones de una U-Net

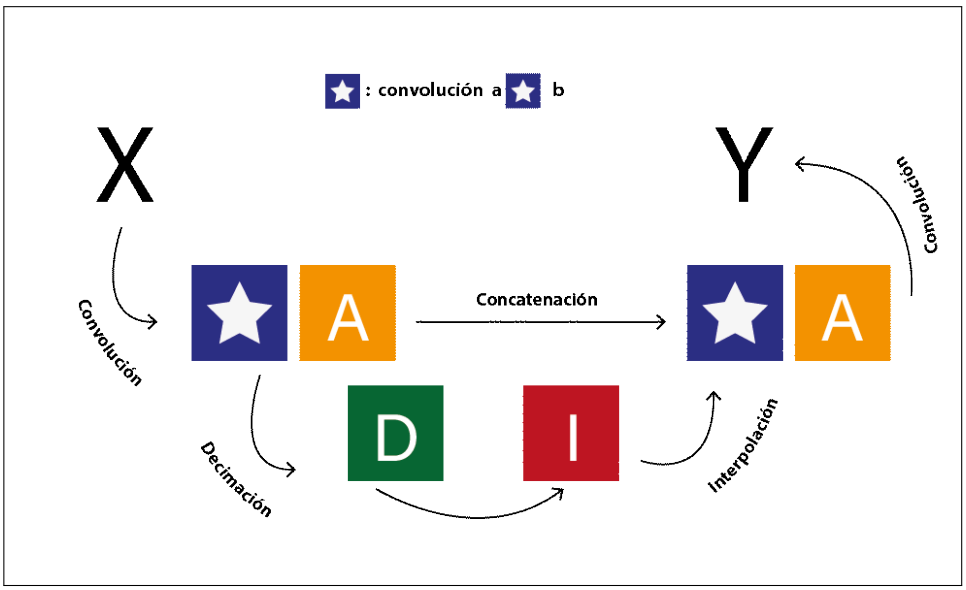

Figura 3.11: Esquema U-Net

La arquitectura de la U-Net se caracteriza por su diseño simétrico, con una estructura de codificadora-decodificadora, y por la concatenación de las características aprendidas en diferentes escalas, lo que permite una mejor segmentación de las imágenes, que ayuda a rellenar la falta de información, debido a la decimación y al aumento de la profundidad.

La parte codificadora está compuesta por varios bloques de capas convolucionales y *pooling* o decimación, cada uno de ellos con una determinada cantidad de canales que se van reduciendo a medida que se va profundizando en la red, permitiendo que la red vaya extrayendo características cada vez más complejas. La parte decodificadora, por otro lado, está compuesta por varios bloques de capas de

interpolación y concatenación. La interpolación aumenta el tamaño del volumen, permitiendo que la red recupere información descartada en la decimación durante la parte de la rama decodificadora. La concatenación se utiliza para combinar la información aprendida en la parte codificadora con la información de la capa actual en el decodificadora.

Decimación: Ver sección Fig. 3.8.

Interpolación:

La operación de interpolación o *UpSampling* es una técnica utilizada para aumentar el tamaño de una imagen o volumen tridimensional. Agregando nuevos puntos de datos entre los puntos existentes. Por ejemplo: si se tiene la secuencia $(1, 2, 3)$ y se pide que agregue una muestra entre muestras la secuencia interpolada seria de la siguiente manera $(1, 1, 2, 2, 3, 3)$.

Como se ve en la figura 3.12 lo que se hace en la interpolación es agregar el mismo valor que el anterior entre muestras.

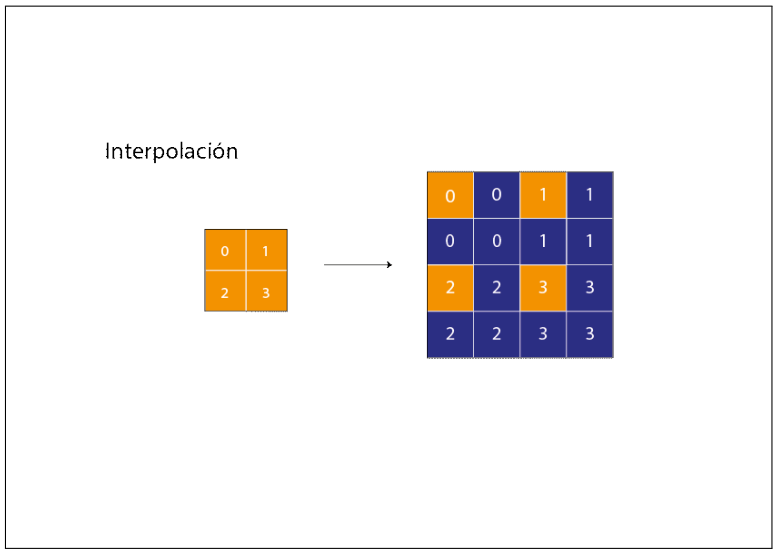

Figura 3.12: Interpolación

Concatenación:

La operación de concatenación (Fig. 3.13) se refiere al proceso de unir dos o más capas de información en una sola

capa. Se utiliza para combinar información de diferentes escalas en una sola capa de salida. Sirve para mejorar la precision de la segmentación, ya que cuando interpolamos a parte de realizar interpolación agrandando el tamaño de los FM, además podemos agregar información real. Lo que se busca resaltar con la concatenación es que no se realiza una interpolación grosera o arbitraria, sino que se utiliza información real existente en la rama codificadora.

En dos dimensiones se podría pensar de la siguiente manera:

$$M1 = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \quad M2 = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix} \tag{3.5}$$

Al concatenar podria hacerlo de dos maneras:

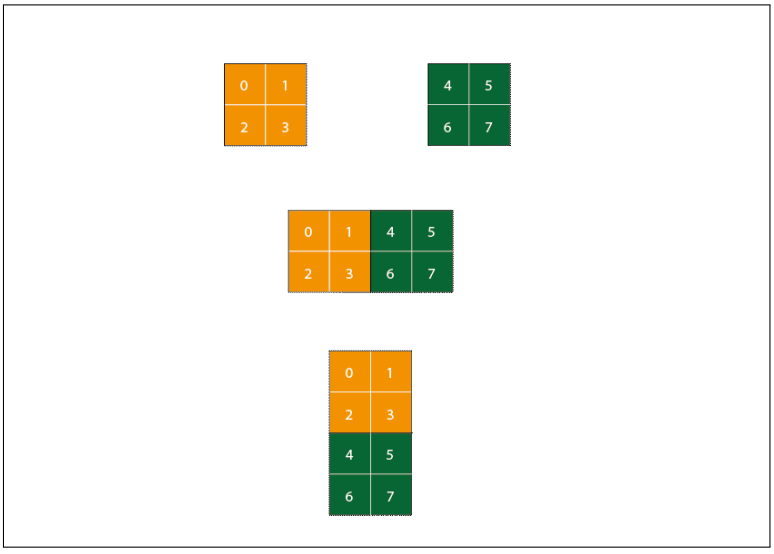

Figura 3.13: Concatenación

$$Concat(M1, M2) = \begin{pmatrix} 1 & 2 & 5 & 6 \\ 3 & 4 & 7 & 8 \end{pmatrix} \tag{3.6}$$

$$Concat(M1, M2) = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{pmatrix} \tag{3.7}$$

Convolución 1x1x1:

En el caso de la convolucion 1x1x1 lo que se hace es reducir el numero de canales. A diferencia de las convoluciones con filtros de mayor tamaño, la convolución 1x1x1 no realiza una operación de convolución en profundidad, sino que realiza una combinación lineal de los *píxeles* o muestas de entrada en cada posición espacial. Esto significa que la convolución 1x1x1 reduce la dimensión de los tensores en la dimensión de la profundidad, sin afectar el tamaño de las dimensiones espaciales.

Por ejemplo en la red que se usa en la tesis, la anteúltima capa de convolucion tiene $128x128x128x16$ canales donde el 16 es el número de filtros usados, entonces para volver a tener la misma dimension de entrada y de salida se usa la convolucion $1x1x1x16$, se reduce de esta manera el número de canales a $128x128x128$ sin modificar el tamaño del volumen original.

En este ejemplo (Fig. 3.14) le aplicamos al tensor T convolución 1x1x1 con el filtro f:

$$T = \left\{ \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}, \begin{pmatrix} 9 & 10 \\ 11 & 12 \end{pmatrix} \right\} \tag{3.8}$$

$$Conv\ 1x1x1\ (T) = \begin{pmatrix} 1+5+9 & 2+6+10 \\ 3+7+11 & 4+8+12 \end{pmatrix} \tag{3.9}$$

Siendo el filtro:

$$f = [1, 1, 1] \tag{3.10}$$

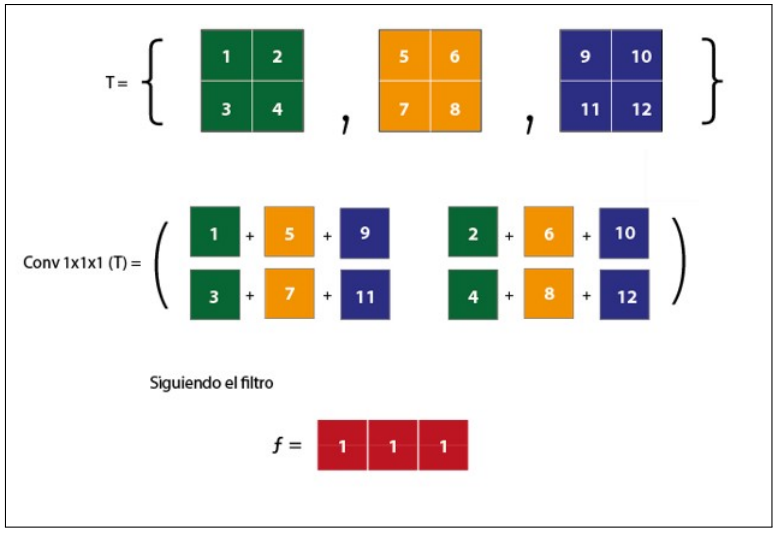

Figura 3.14: Convolución 1x1x1

Entrenamiento de la U-Net

4.1. Generador de datos

Este preprocesamiento incluye cargar los datos desde el sistema de archivos en un formato específico, estandarizarlos y aplicar una técnica de incremento artificial de datos. Todo esto se hace con el objetivo de preparar los datos de manera adecuada para que el modelo que el entrenamiento se realice de manera más eficiente.

Datos

Los datos con los que se cuenta, como se mencionó antenriormente, son 200 cubos sintéticos de entrenamiento, etiquetados con sus respectivas fallas, y 20 de validación. Los cuales se pueden visualizar de esta forma Fig. 4.1, Fig. 4.2:

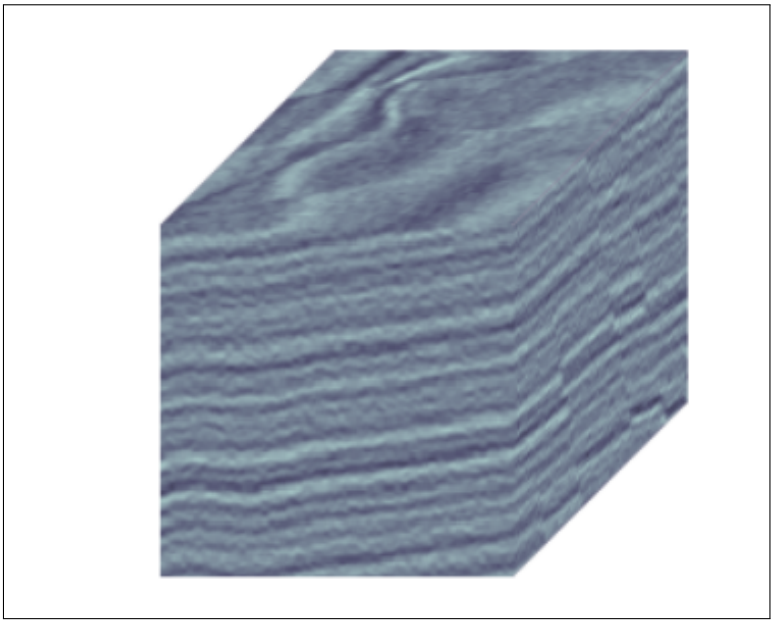

Figura 4.1: Cubo de entrenamiento.

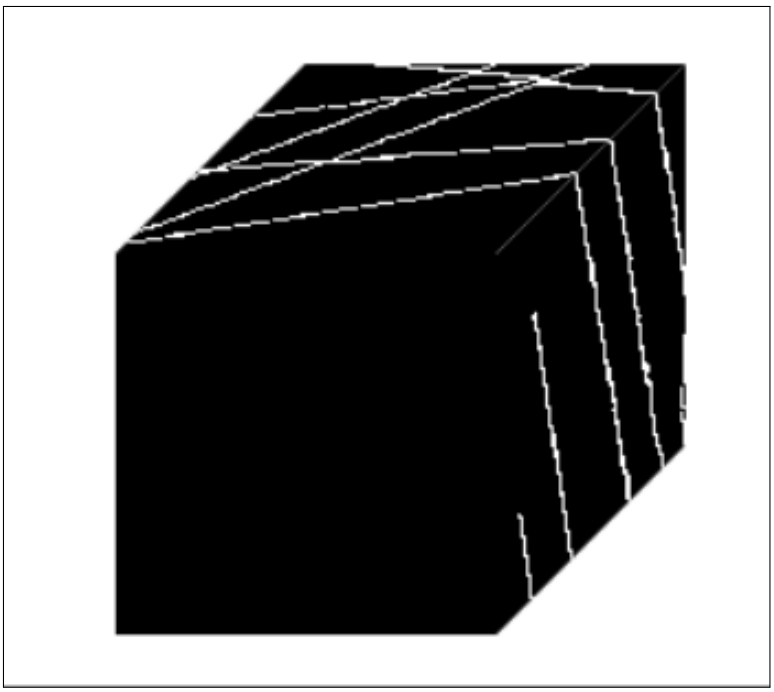

Figura 4.2: Cubo de entrenamiento etiquetado.

Incremento artificial de datos

Una técnica para aumentar la cantidad de datos sin incorporar nuevos se conoce como incremento artificial de datos o *data augmentation*. Esto se hace volteando el cubo, haciendoló se estará aumentando el número de ejemplos de entrenamiento de 200 a 400 . Por cada época se estarían usando 400 cubos. Con los pesos conseguidos producto de ese entrenamiento se evaluarán los 20 cubos de validación.

Desbalance de clases

En los problemas binarios como este, puede surgir lo que se llama un desbalance de clases. Como se ve en las figuras 4.3 y 4.4. Hay un gran porcentaje de muestras que no son fallas. , aproximadamente un **7 %** son fallas en el conjunto de entrenamiento, por lo que estaríamos partiendo de un *Binary Accuracy* o Precisión Binaria del **93 %**. Esto quiere decir que si se dijera que el cubo no tiene fallas estaríamos acertando en un **93 %** algo a tener en cuenta en las métricas usadas en el trabajo.

Además, es importante tener en cuenta que se utiliza la métrica de precisión binaria (*Binary Accuracy*) con un umbral de 0.4. Esto implica que las predicciones que caen en el rango de 0 a 1 serán clasificadas como 0 si son menores que 0.4, y como 1 si son iguales o mayores a 0.4. Esta elección de umbral se realiza para que la comparación entre las etiquetas y las predicciones sea significativa y coherente

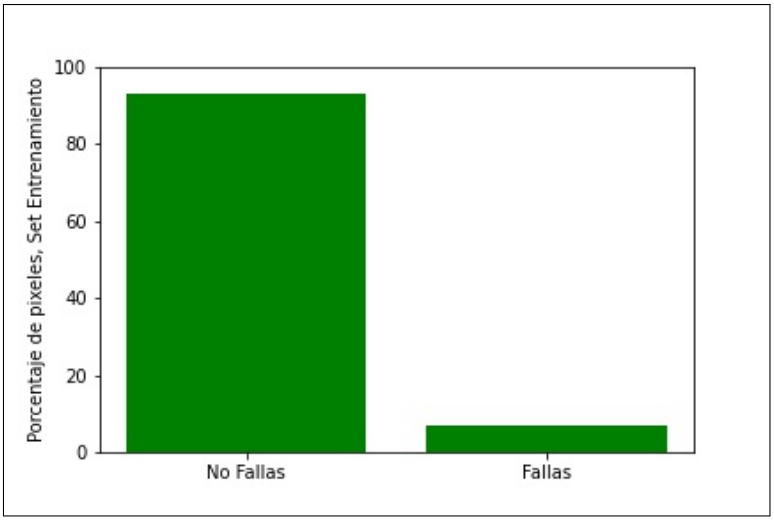

Figura 4.3: Desbalance de clases y media.

Esta figura (4.4) nos muestra el porcentaje de fallas para cada uno de los cubos de entrenamiento, siendo la línea horizontal naranja un promedio.

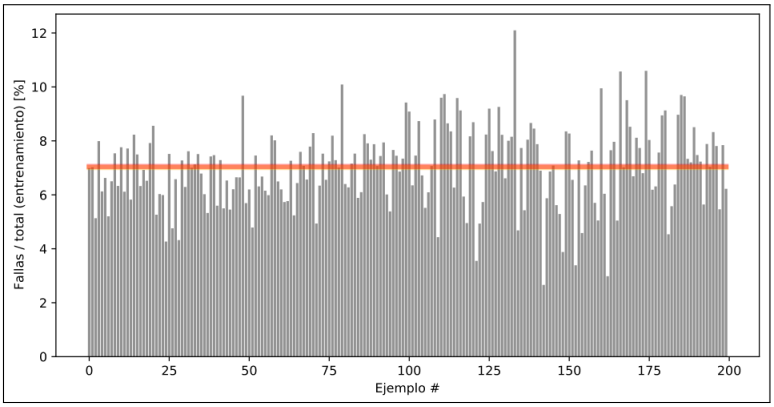

Figura 4.4: Desbalance de cada cubo.

4.2. Entrenamiento

La optimizacion es una herramienta importante de decision en la ciencia. Primero hay que definir algun objetivo, hacer una buena predución de las fallas, para esto se usa un modelo. El modelo es U-Net cuyas variables son los valores de los filtros. Luego, para saber cúan bueno es el modelo tengo que compararlo con las etiquetas, es decir, para controlar la salida de la red hay que saber cuán lejos está la salida de lo que se espera como salida. Esto lo hace la función de costo. Para llevar a acabo el objetivo hay que encontrar valores de los filtros que optimicen el modelo. El entrenamiento es un proceso de optimización, consiste en la derivación de la función de costo y en la actualización de los pesos mediante *backpropagation* o propagación hacia atrás usando un optimizador, en este caso Adam. El proceso se esquematiza en la Fig 4.5.

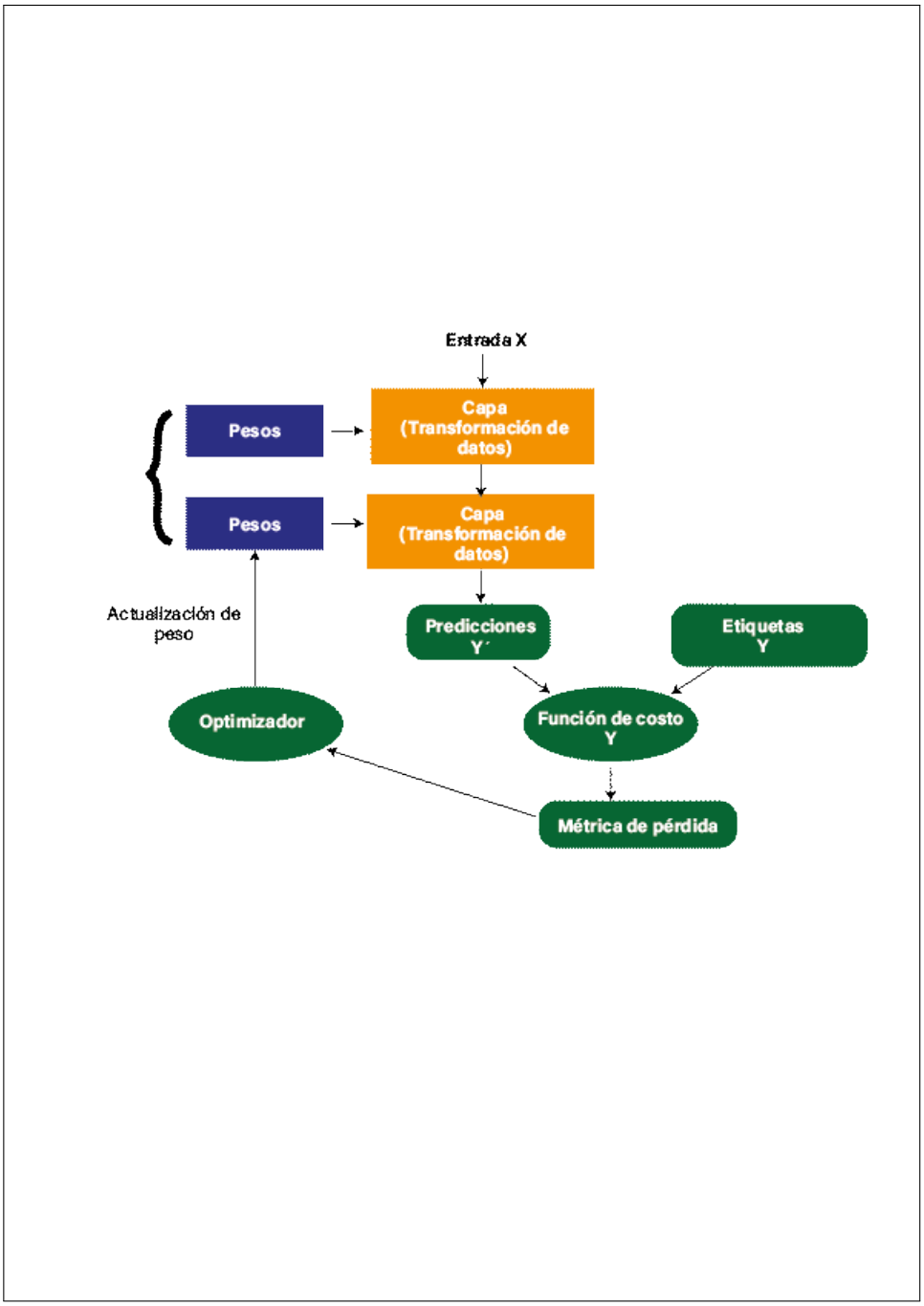

Figura 4.5: Esquema de entrenamiento.

Función de costo

La función de costo utilizada es *Smoothed Dice* (Eq. 4.1) la cual es una variante de la función de pérdida *Dice*, que se utiliza comúnmente en problemas de segmentación de imágenes médicas:

$$\ell(p, g) = 1 - \frac{2\sum_{i=1}^{N} p_i g_i + 1}{\sum_{i=1}^{N} p_i + \sum_{i=1}^{N} g_i + 1} \tag{4.1}$$

$p = M(X; W)$ es la predicción $0 \leqq p \leqq 1$ y $g = Y$ es el dato etiquetado

La función de costo *Dice* se calcula como el cociente entre la intersección y la unión de los conjuntos de muestras etiquetadas como positivas en la imagen predictiva y la imagen de referencia.

La función *Smoothed Dice* agrega una suavizado en el numerador y denominador para evitar problemas numéricos cuando hay muy pocos o ningúna muestra positiva en la imagen predictiva o de referencia. El número de Dice es un número entre 0 y 1 por lo que la función pérdida va a ir de 1 a 0 idealmente.

Optimizador

Se usa el optimizador Adam (Eq. 4.2) [Kingma and Ba, 2014] para optimizar los parámetros de la red con un *learning rate* de 0.0001. Es un optimizador que hace más eficiente la optimización estocástica y surge de la combinación de dos métodos AdaGrad [Duchi et al., 2011] y RMSProp [Hinton et al., 2012].

Los pesos se modificarán de la siguiente manera:

$$\theta_t \leftarrow \theta_{t-1} - \frac{\alpha * \widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon} \tag{4.2}$$

Donde $\widehat{m}_t$ es un promedio ponderado (pesado) de los gradientes de las iteraciones anteriores y la parte que corresponde al RMSprop es una especie de normalización de los gradientes.

La combinación de estos dos funciona muy bien y reduce el número de iteraciones (aproximadamente se ahorraron 5 épocas) considerablemente en la práctica.

Finalmente se realiza el entrenamiento por 20 épocas:

Se puede observar en las figuras 4.6 e 4.7 que a partir de la época 15 (eje horizontal) el resultado tanto en los datos de entrenamiento(naranja punteada) como en los de validación (negra sólida) se horizontalizan.Eso se logra gracias a que se encontró un valor para los pesos mediante *backpropagation* que mínimizan la función de costo propuestauna función de costo que resulta más eficiente para este problema:

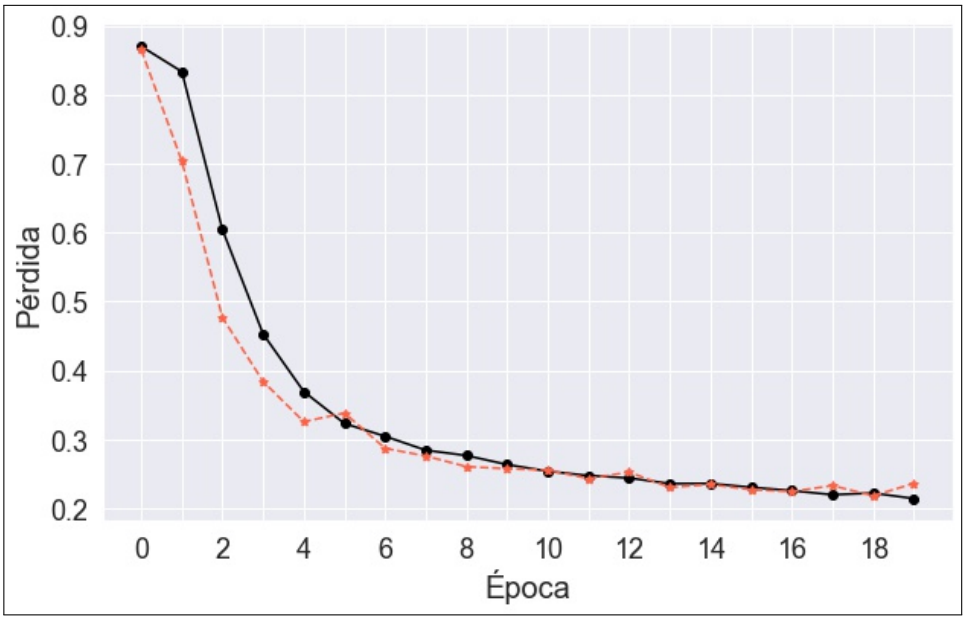

Figura 4.6: Pérdida

Se definión error medio cuadrático como :

$$EMC = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{4.3}$$

donde y_i es el valor observado, $\hat{y}_i$ es el valor estimado y n es el número total de cubos usados en el entrenamiento y la validación. Es bueno ver que tiene el mismo comportamiento que la función de costo

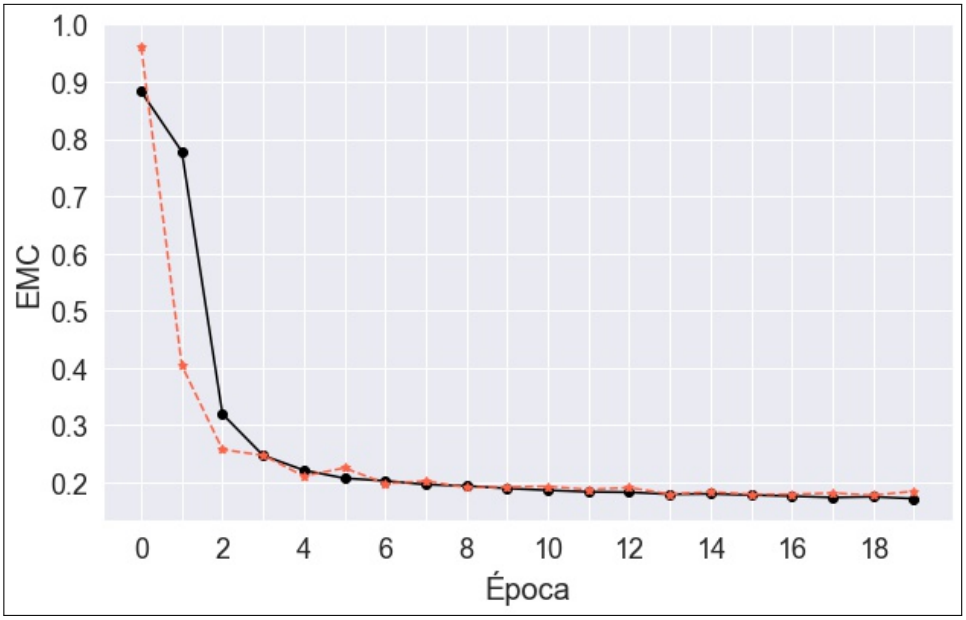

Figura 4.7: Error medio cuadrático

En esta figura (4.8) se muestra la precisión binaria. Si bien la precisión binaria es muy alta no hay que perder de vista lo dicho anteriormente, que se trata de un problema desbalanceado y que el comando usa un umbral o *trheshold* de 0.5. Por eso se aplicarán a modo de ejemplo otras metricas cómunes en problemas binarios como, especificidad, sensibilidad y F1.

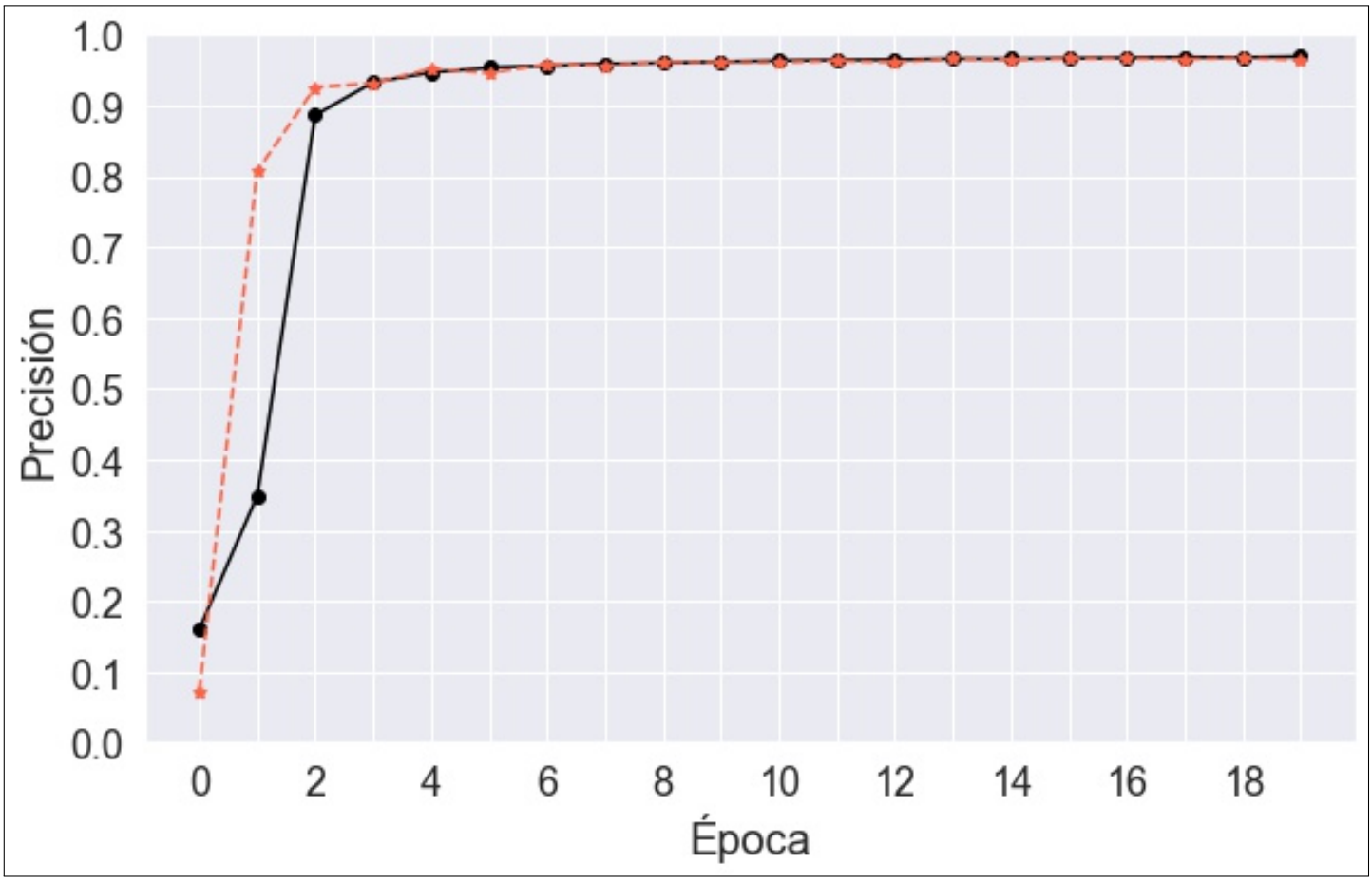

Figura 4.8: Precisión binaria

4.3. Aplicación a datos de validación

Los datos de validación son fundamentales porque son cubos que tienen la misma estructura pero no fueron usados para el entrenamiento. Muchas veces se produce un fénomeno que se lllama *Overfitting*, que tiene que ver con la poca generalización del modelo planteado. Se puede imaginar este fenómeno que para interpolar unas pocas muestras se usa polinomio de grado alto. Para asegurarnos que no ocurra, planteamos el análisis de los cubos de validacion y comparamos con los de entrenamiento, si tienen un comportamiento muy desigual significa que el nivel de generalización fue bajo.

Ejemplo de validación

Para esquematizar el proceso de validación, se toma un cubo en particular elegido aleatoriamente. Para el cubo se logra una Precisión Binaria de 0,98 y un EMC de 0,13. El resultado lo consideramos bueno como para una primera aproximación, que es el objetivo del código de la tesis.

A continuación se presentan dos figuras, la primera (4.9) corresponde al cubo completo, la prediccion y la etiqueta en ese orden. La segunda imagen (4.10) son cortes del cubo donde se puede observar con más detalle la diferencia entre la etiqueta y lo predicho.

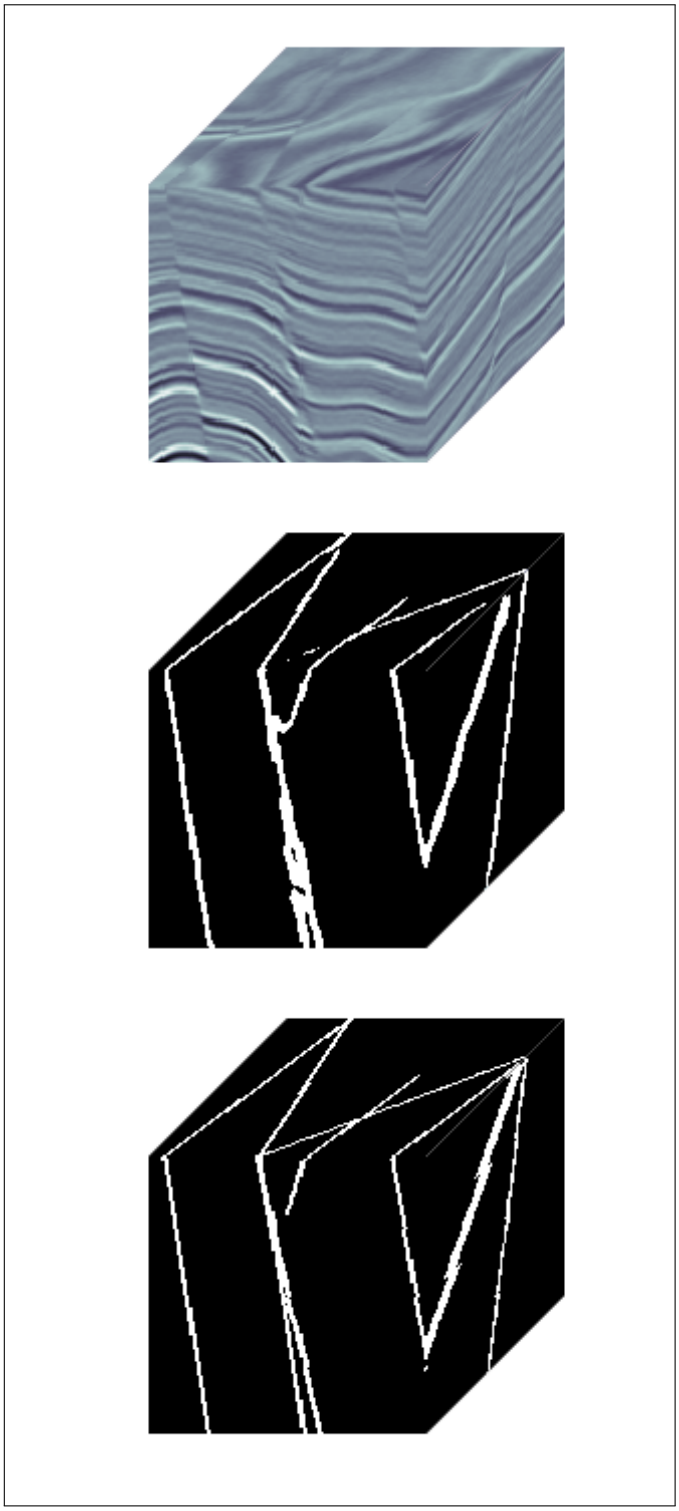

Figura 4.9: Cubo de validación, predicción y etiqueta.

c

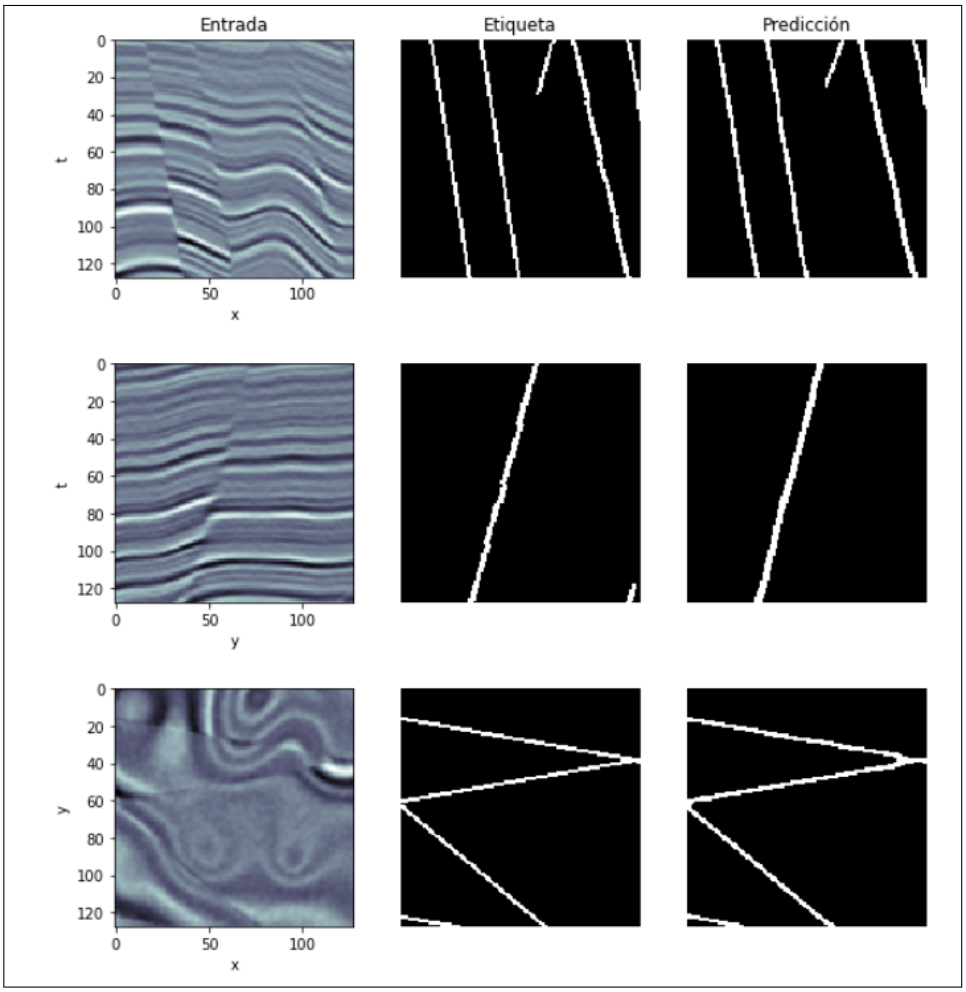

Figura 4.10: Inline, Crossline, Time Slice del cubo de validación. Real, etiquetas y predicción.

Métricas comunmente usadas en Clasificación Binaria:

Métrica	**Valor**	**Fórmula**	**Objetivo**
TPR (Sensibilidad)	0.56	$\frac{TP}{TP+FN}$	Evitar falsos negativos
Exactitud	0.55	$\frac{TP}{TP+FP}$	Evitar falsos positivos
F1-Score	0.55	$2 \times \frac{Precision \times Recall}{Precision+Recall}$	Media armónica de Exactitud y Sensibilidad

Cuadro 4.1: Principales métricas de evaluación para clasificación binaria.

La **sensibilidad** proporciona información sobre la capacidad del modelo para evitar falsos negativos, es decir, clasificar incorrectamente casos positivos como negativos.

La **exactitud** muestra la capacidad del modelo para evitar clasificar incorrectamente casos negativos como positivos. Es particularmente útil cuando el costo de los falsos positivos es alto.

La **puntuación F1** proporciona una evaluación equilibrada del modelo, teniendo en cuenta tanto los falsos positivos como los falsos negativos. Es útil cuando se desea encontrar un equilibrio entre la precisión y la sensibilidad.

Por ejemplo si estamos tratando un problema médico como diagnosticar cancer por ímagenes, vamos a preferir que nuestra red clasifique incorrectamente casos positivos a que pase lo contrario. Por lo tanto preferiremos que priorice la exactitud a la sensibilidad.

4.4. Conclusiones del entrenamiento y prueba en los datos de validación

En el entrenamiento, tanto los datos de entrenamiento como los datos de validación, arrojan una precision binaria muy alta en pocas iteraciones. El entrenamiento aproximadamente 2 horas (Core i7, Nvidia Geforce GTX 1070). La

predicción en los cubos de validación toma del orden de segundos. La cantidad de parámetros de nuestra versión de la U-Net tiene 1,4M de parámetros, simplificando la red de [Ronneberger et al., 2015] y la de [Wu et al., 2019].

Sin perder de vista que son cubos sínteticos y pequeños se podria concluir que el algoritmo es bueno en relación a tiempo de cómputo - resultados.

Aplicación en dato real

5.1. Aplicación en dato real

En esta etapa del proceso, es importante destacar que aplicar el modelo entrenado a datos sísmicos 3D reales implica una evaluación crítica y cuidadosa. Aunque el modelo haya demostrado un buen desempeño en los datos sintéticos y en los conjuntos de datos de entrenamiento y validación, los datos reales presentan desafíos adicionales y representan una prueba importante para evaluar la generalización del modelo, el cual es nuestro objetivo.

Al trabajar con datos sísmicos reales, es fundamental tener en cuenta las características específicas de los datos, como la calidad de la adquisición sísmica, el ruido inherente, las variaciones en las propiedades del subsuelo y las limitaciones de resolución espacial. Estos factores pueden influir en el rendimiento del modelo y es posible que se requieran ajustes adicionales para adaptarlo al nuevo conjunto de datos.

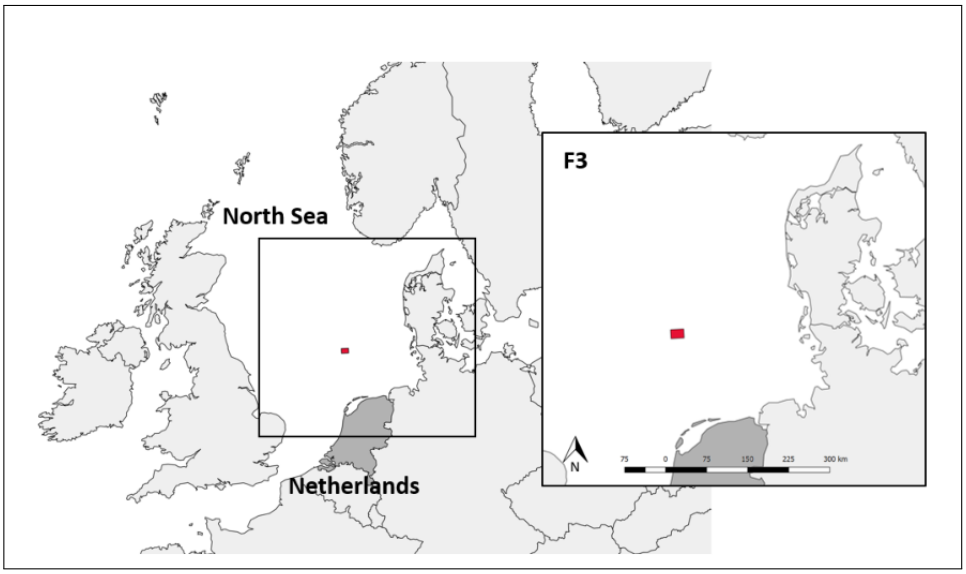

Figura 5.1: Ubicación de la toma de datos

Aprendizaje transferido

De la misma forma en que un Geofísico puede usar sus bases de Matemática, Física y Estadística para luego resolver problemas de otras disciplinas como las Ciencias de Datos o la inteligencia artificial (IA); las redes neuronales profundas que fueron entrenadas para resolver un problema pueden hacerlo para resolver problemas similares. En este caso una red que fue entrenada con ´dato sintético´, se aplica a ´dato real´.

El aprendizaje transferido o *Transfer Learning*, se refiere al conjunto de métodos que permiten transferir conocimientos adquiridos gracias a la resolución de problemas para resolver otros problemas.

5.2. Dato real a evaluar

El *dataset* consiste en $384km^2$ migrado en tiempo 3D, con 651 *inlines* y 951 *crosslines*, ubicado en el Mar del Norte (Fig. 5.1), *Netherlands offshore*. El rango de tiempo del *set* es de 1,848ms, muestreado a 4ms y un *bin* de 25m.

Descripción Geológica de la zona prospectada

La Cuenca Graben Central es el resultado de la apertura del Atlántico Norte y la división posterior del supercontinente Pangea. Esta región principal se caracteriza por un triple sistema de rift *Central Graben*, *Viking Graben* y *Moray Firth Basins* [Silva et al., 2019]

El bloque está cubierto por sísmica 3D que se adquirió para explorar en busca de petróleo y gas en los estratos del Jurásico Superior al Cretácico Inferior, que se encuentran debajo del intervalo seleccionado para este conjunto de demostración. Los $1200ms$ superiores del conjunto de demostración consisten en reflectores pertenecientes al Mioceno, Plioceno y Pleistoceno. El lecho sigmoidal a gran escala es evidente y consiste en los depósitos de un gran sistema fluvio deltaico que drenó gran parte de la región del Mar Báltico [Sørensen et al., 1997] ;[Overeem et al., 2001].

El paquete deltaico está compuesto de arena y lutita, con una porosidad generalmente alta $(20 - 33\,\%)$. Algunas vetas cementadas con carbonato están presentes. Se pueden observar varias características interesantes en este paquete. La más destacada es la estratificación sigmoidal a gran escala, con estructuras de *downlap*, *toplap*, *onlap* y truncamiento de alta calidad, como se enseña en los libros de texto. También se pueden observar *bright spots*, causados por bolsas de gas biogénico Fig. 5.2. No son poco comunes

en esta parte del Mar del Norte. Se pueden distinguir varias facies sísmicas: transparentes, caóticas, lineales y tejas.

Los registros de pozo muestran que las facies transparentes consisten en una litología bastante uniforme, que puede ser arena o lutita. Las facies caóticas probablemente representan depósitos deslizados. Se ha demostrado que las tejas en la base de los clinoformos consisten en turbiditas arenosas. [Silva et al., 2019]

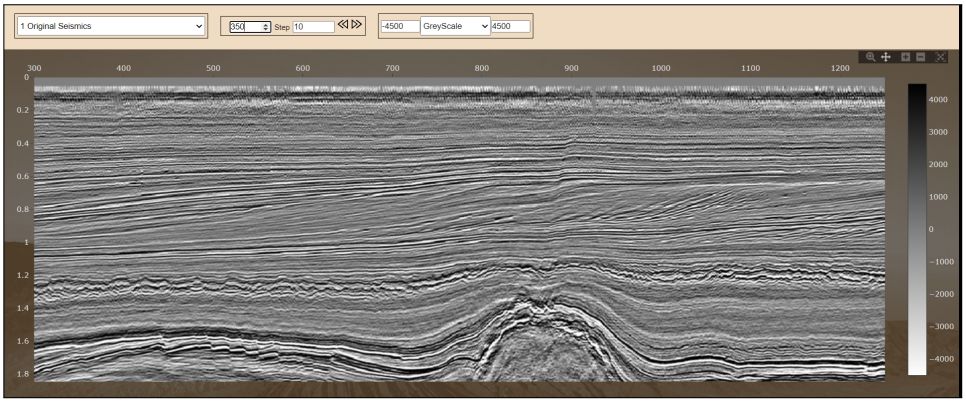

Figura 5.2: Sección Sísmica Inline 350

El dato sísmico 3D que efectivamente se va a usar es un subconjunto (384 [*inline*] x 514 [*crossline*] x 128 [vertical] muestras) extraído del *Netherlands off-shore F3 block seismic data*. Como puede observarse directamente desde el dato en amplitud Fig. 5.2, existen una serie de fallas orientadas en varias direcciones por eso es que son de nuestro interés [Wu et al., 2019].

5.3. Procesamiento y resultados del dato sísmico

Para procesar este dato sísmico 3D. Se toman los datos y los convierten en un arreglo de $384x514x128$; se subdivide en cubos de $128x128x128$, luego se define una ventana de Hamming en 3D para aplicarle a cada uno de los cubos con sus MF correspondientes. Quedando 20 cubos del tamaño mencionado anteriormente. Por último se eliminan los efectos de solapamiento al ser pegados obteniendo así el resultado final.

En las siguientes figuras (Fig. 5.3, Fig. 5.4, Fig. 5.5), se presentan cortes del subcubo F3. Estos cortes están superpuestos con los valores de los MF, que varían entre 0 y 1. Aunque en este ejemplo no se incluyen etiquetas, a simple vista se aprecia un resultado bastante satisfactorio. Los MF parecen proporcionar una buena representación de los datos, lo cual es prometedor para este caso real.

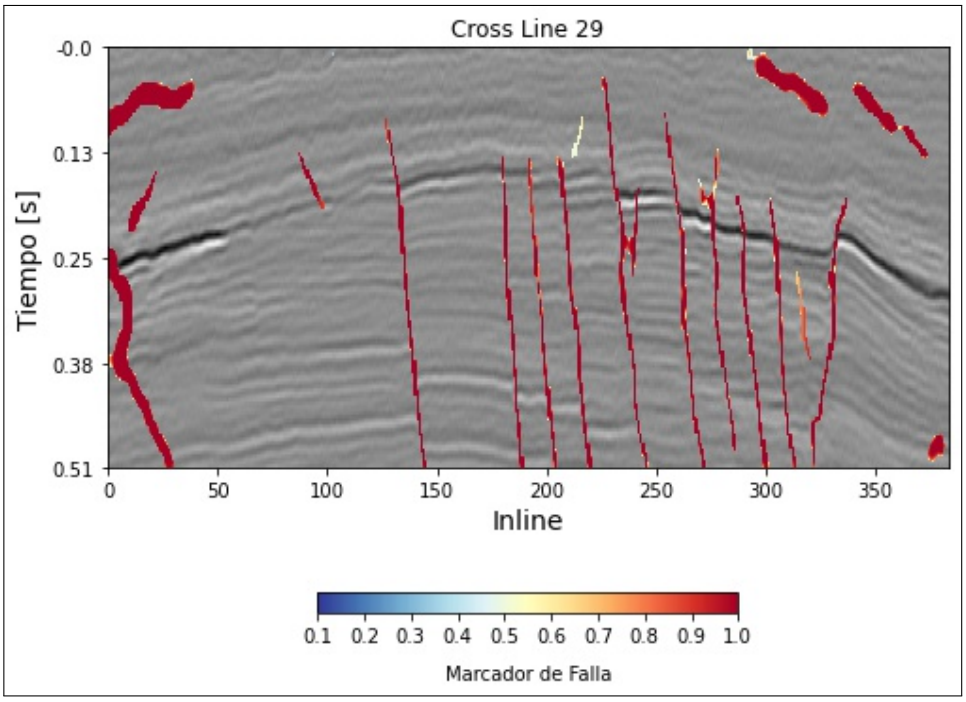

Figura 5.3: Crossline 29

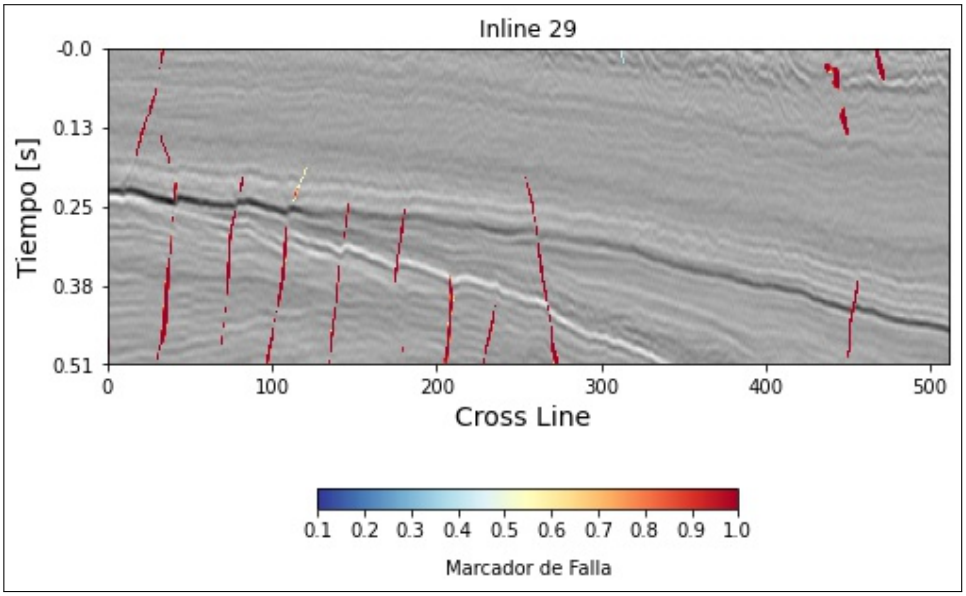

Figura 5.4: Inline 29

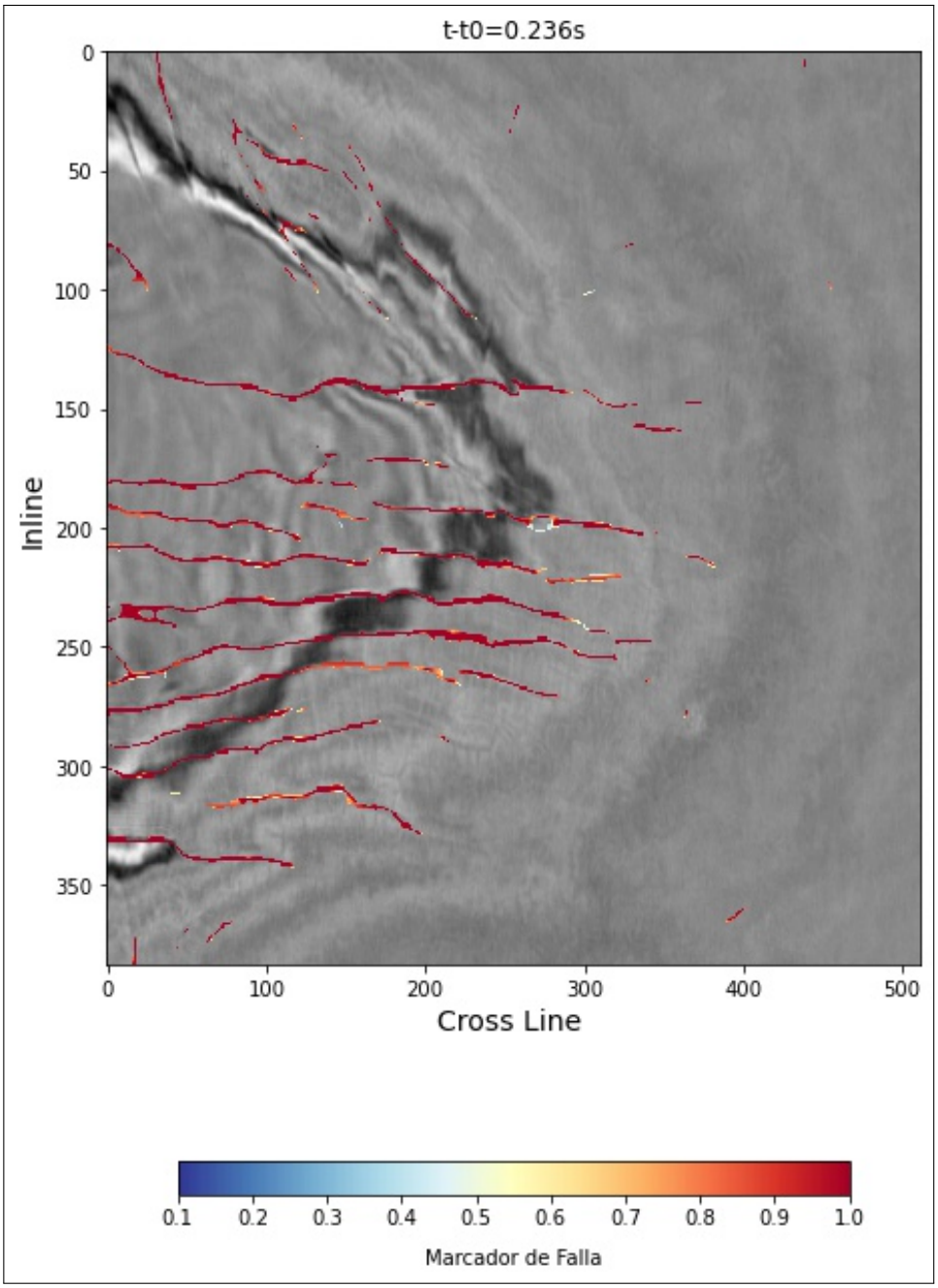

Figura 5.5: Time Slice 59=0.24s

Conclusiones y trabajos a futuro

6.1. Conclusiones

Se ha desarrollado y evaluado con éxito una CNN para la detección de fallas en imágenes sísmicas 3D.Abordando el problema como una tarea de segmentación binaria y simplificando la arquitectura U-Net para optimizar el uso de memoria de la GPU y el tiempo de cálculo sin comprometer el rendimiento.

Mediante el uso de la función de pérdida *smoothed Dice Loss* se ha logrado superar el sesgo presente en la distribución de muestras de "fallasz "no fallas", permitiendo que el modelo aprenda de manera efectiva a detectar fallas en los volúmenes sísmicos. Tambien se redujo el número de iteraciones necesarias para la convergencia con respecto a [Wu et al., 2019].

A pesar de haber entrenado el modelo únicamente con 200 pares de volúmenes sísmicos y de fallas sintéticas, generados mediante la introducción aleatoria de plegamientos, fallas y ruido, se ha logrado obtener resultados aceptables en la detección de fallas en volúmenes sísmicos 3D de campo adquiridos en diferentes estudios.

Es destacable que, aunque solo se agregaron fallas planas en los datos de entrenamiento, la red neuronal ha demostrado la capacidad de detectar fallas curvas en las imágenes sísmicas reales. Esto resalta la capacidad de generalización del modelo y su capacidad para adaptarse a diferentes geometrías de fallas más allá de las utilizadas en el entrenamiento.

Nuestro enfoque demuestra que las CNN son una herramienta efectiva para la detección de fallas en imágenes sísmicas 3D. Además, el uso de datos sintéticos en el entrenamiento ha indicado ser una estrategia viable para lograr un buen rendimiento en la detección de fallas en volúmenes sísmicos reales. Estos resultados prometen aplicaciones prácticas en la industria petrolera y la exploración de yacimientos, donde la detección precisa de fallas es de vital importancia.

6.2. Trabajos a futuro

Hay varias áreas de investigación y desarrollo que se pueden explorar en trabajos futuros para mejorar y expandir el método propuesto de detección de fallas en imágenes sísmicas. Algunas posibles direcciones incluyen:

Mejorar la precisión de la geometría de las fallas: Dado que el método actual puede presentar discrepancias en los ángulos de inclinación y extensiones de las fallas en comparación con las interpretaciones manuales, se puede investigar cómo refinar y ajustar los parámetros del modelo de red neuronal para obtener una representación más precisa de las geometrías de las fallas. [Lorentzen et al., 2022]

Diversificar el conjunto de datos de entrenamiento: Actualmente, el conjunto de datos de entrenamiento se compone principalmente de fallas planas de alta inclinación. Sería beneficioso ampliar y diversificar el conjunto de datos de entrenamiento para incluir diferentes tipos de fallas, como fallas curvas, escalonadas y otros patrones más complejos. Esto ayudaría a mejorar la capacidad del modelo para detectar y representar una variedad más amplia de

geometrías de fallas.

Reducción del ruido sísmico: Dado que el método propuesto es sensible al ruido sísmico, se pueden investigar técnicas de preprocesamiento y filtrado de datos para reducir la influencia del ruido en las predicciones de fallas. Esto podría implicar el uso de algoritmos de eliminación de ruido, técnicas de atenuación de ruido o el desarrollo de enfoques específicos para mejorar la robustez del modelo frente al ruido sísmico.

Comparación con atributos sísmicos: se puede comparar contra atributos como ant-tracking o la coherencia, tambien contra interpretaciones hechas de forma manual por interpretes experimentados y ver qué valor agregado puede aportar este nuevo atributo.

Bibliografía

[Alcalde et al., 2017] Alcalde, J., Bond, C. E., Johnson, G., Ellis, J. F., and Butler, R. W. (2017). Impact of seismic image quality on fault interpretation uncertainty. *GSA Today*.

[Chollet, 2021] Chollet, F. (2021). *Deep learning with Python*. Simon and Schuster.

[Conexplo, IAPG, 2022] Conexplo, IAPG (2022). *Crónicas de Exploración y Desarrollo de Hidrocarburos. 50 años de Conceptos e Historias en Primera Persona*. Conexplo, IAPG.

[Cunha et al., 2020] Cunha, A., Pochet, A., Lopes, H., and Gattass, M. (2020). Seismic fault detection in real data using transfer learning from a convolutional neural network pre-trained with synthetic seismic data. *Computers & Geosciences*, 135:104344.

[Di et al., 2018a] Di, H., Shafiq, M., and AlRegib, G. (2018a). Patch-level mlp classification for improved fault detection. In *SEG Technical Program Expanded Abstracts 2018*, pages 2211–2215. Society of Exploration Geophysicists.

[Di et al., 2018b] Di, H., Wang, Z., and AlRegib, G. (2018b). Why using cnn for seismic interpretation? an investigation. In *2018 SEG International Exposition and Annual Meeting*. OnePetro.

[Di et al., 018a] Di, H., Wang, Z., and AlRegib, G. (2018a). Seismic fault detection from post-stack amplitude by convolutional neural networks. In *80th EAGE Conference*

and Exhibition 2018, volume 2018, pages 1–5. EAGE Publications BV.

[Duchi et al., 2011] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

[Fossen, 2016] Fossen, H. (2016). *Structural geology*. Cambridge university press.

[Gan and Elsworth, 2014] Gan, Q. and Elsworth, D. (2014). Analysis of fluid injection-induced fault reactivation and seismic slip in geothermal reservoirs. *Journal of Geophysical Research: Solid Earth*, 119(4):3340–3353.

[Gao et al., 2021] Gao, K., Huang, L., and Cladouhos, T. (2021). Three-dimensional seismic characterization and imaging of the soda lake geothermal field. *Geothermics*, 90:101996.

[Gao et al., 2022] Gao, K., Huang, L., Zheng, Y., Lin, R., Hu, H., and Cladohous, T. (2022). Automatic fault detection on seismic images using a multiscale attention convolutional neural networkmultiscale attention fault detection. *Geophysics*, 87(1):N13–N29.

[Guitton, 2018] Guitton, A. (2018). 3d convolutional neural networks for fault interpretation. In *80th EAGE Conference and Exhibition 2018*, volume 2018, pages 1–5. EAGE Publications BV.

[Guo et al., 2018] Guo, B., Li, L., and Luo, Y. (2018). A new method for automatic seismic fault detection using convolutional neural network. In *2018 SEG International Exposition and Annual Meeting*. OnePetro.

[Hale, 2012] Hale, D. (2012). Fault surfaces and fault throws from 3d seismic images. In *2012 SEG Annual Meeting*. OnePetro.

[Hardman and Booth, 1991] Hardman, R. and Booth, J. (1991). The significance of normal faults in the exploration and production of north sea hydrocarbons. *Geological Society, London, Special Publications*, 56(1):1–13.

[Hinton et al., 2012] Hinton, G., Srivastava, N., and Swersky, K. (2012). Neural networks for machine learning. *Coursera, video lectures*, 264(1):2146–2153.

[Huang et al., 2017] Huang, L., Dong, X., and Clee, T. E. (2017). A scalable deep learning platform for identifying geologic features from seismic attributes. *The Leading Edge*, 36(3):249–256.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Lorentzen et al., 2022] Lorentzen, M. C., Bredesen, K., Smit, F. W., Hansen, T. H., Nielsen, L., and Mosegaard, K. (2022). Mapping cretaceous faults using a convolutional neural network. *Bulletin of the Geological Society of Denmark*, 71.

[Nielsen, 2015] Nielsen, M. A. (2015). *Neural networks and deep learning*, volume 25. Determination press San Francisco, CA, USA.

[Overeem et al., 2001] Overeem, I., Weltje, G. J., Bishop-Kay, C., and Kroonenberg, S. (2001). The late cenozoic eridanos delta system in the southern north sea basin: a climate signal in sediment supply? *Basin Research*, 13(3):293–312.

[Padrón, 2022] Padrón, A. (2022). *Detección de bordes en datos sísmicos con filtros Sobel*. PhD thesis, Universidad Nacional de La Plata.

[Pochet et al., 2018] Pochet, A., Diniz, P. H., Lopes, H., and Gattass, M. (2018). Seismic fault detection using convolutional neural networks trained on synthetic post-stacked amplitude maps. *IEEE Geoscience and Remote Sensing Letters*, 16(3):352–356.

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer.

[Silva et al., 2019] Silva, R. M., Baroni, L., Ferreira, R. S., Civitarese, D., Szwarcman, D., and Brazil, E. V. (2019). Netherlands dataset: A new public dataset for machine learning in seismic interpretation. *arXiv preprint arXiv:1904.00770*.

[Sørensen et al., 1997] Sørensen, J. C., Gregersen, U., Breiner, M., and Michelsen, O. (1997). High-frequency sequence stratigraphy of upper cenozoic deposits in the central and southeastern north sea areas. *Marine and Petroleum Geology*, 14(2):99–123.

[Vilarrasa and Carrera, 2015] Vilarrasa, V. and Carrera, J. (2015). Geologic carbon storage is unlikely to trigger large earthquakes and reactivate faults through which co2 could leak. *Proceedings of the National Academy of Sciences*, 112(19):5938–5943.

[Wrona et al., 2022] Wrona, T., Pan, I., Bell, R., Jackson, C. A.-L., Gawthorpe, R., Fossen, H., Osagiede, E., and Brune, S. (2022). Complex fault system revealed from 3-d seismic reflection data with deep learning and fault network analysis. *EGUsphere*, pages 1–22.

[Wu et al., 2019] Wu, X., Liang, L., Shi, Y., and Fomel, S. (2019). Faultseg3d: Using synthetic data sets to train an end-to-end convolutional neural network for 3d seismic fault segmentation. *Geophysics*, 84(3):IM35–IM45.

[Wu et al., 2018] Wu, X., Shi, Y., Fomel, S., and Liang, L. (2018). Convolutional neural networks for fault interpretation in seismic images. In *SEG International Exposition and Annual Meeting*, pages SEG–2018. SEG.

[Zhang et al., 2019] Zhang, Q., Yusifov, A., Joy, C., Shi, Y., and Wu, X. (2019). Faultnet: A deep cnn model for 3d automated fault picking. In *SEG International Exposition and Annual Meeting*. OnePetro.

[Zhao and Mukhopadhyay, 2018] Zhao, T. and Mukhopadhyay, P. (2018). A fault detection workflow using deep learning and image processing. In *2018 SEG international exposition and annual meeting*. OnePetro.