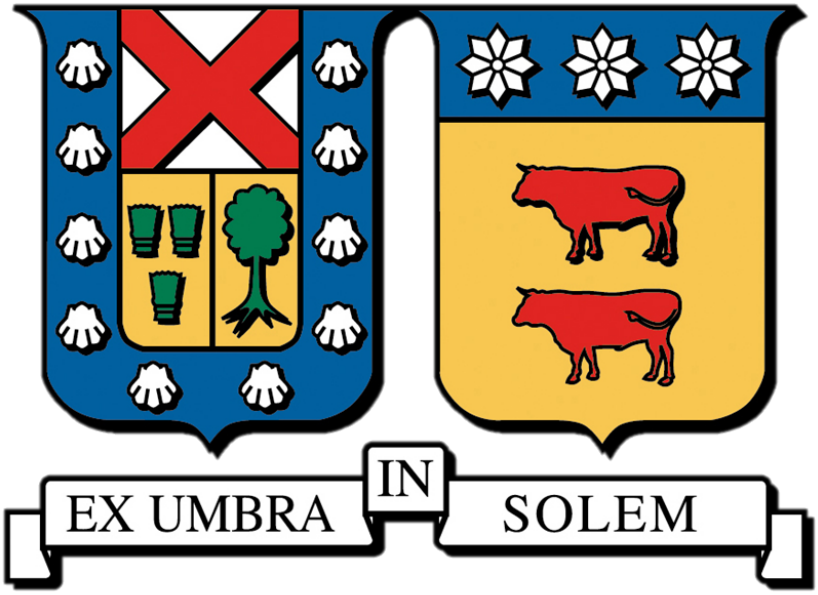
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE MATEMÁTICA
VALPARAÍSO-CHILE



Deep Learning Methods for a Fluid Inverse Problem

Tesis presentada por:

Cristóbal Álvarez Contreras

Como requisito parcial para optar al grado académico de Magíster en Ciencias, mención Matemática y al título profesional de Ingeniero Civil Matemático

Director:

Rodrigo Lecaros Lira

Co-director:

Eduardo Cerpa Jeria

Enero, 2024

Material de referencia, su uso no involucra responsabilidad del autor o de la Institución.

TÍTULO DE LA TESIS:
Deep Learning Methods for an Obstacle Reconstruction Inverse Problem

AUTOR: Cristóbal Álvarez Contreras

TRABAJO DE TESIS, presentado como requisito parcial para optar al grado académico de Magíster en Ciencias, mención Matemática y al título profesional de Ingeniero Civil Matemático de la Universidad Técnica Federico Santa María.

COMISIÓN EVALUADORA:

Integrantes	Firma
Integrante 1 Institución 1	_____
Integrante 2 Institución 2	_____
Integrante 3 Institución 3	_____

Valparaíso, Enero 2024.

Resumen

En este trabajo se aborda el estudio del problema inverso de recuperación de obstáculos en el contexto del sistema de Stokes mediante un enfoque basado en Deep Learning. Se desarrolló un método que consiste en un proceso en el cual se generan imágenes binarias de obstáculos y se calculan las soluciones correspondientes al problema directo. A partir de estas simulaciones, se construye un conjunto de datos que permite llevar a cabo mediciones del problema inverso para diversos obstáculos en una región de interés en la frontera exterior. Posteriormente, se emplean diversas redes neuronales convolucionales en combinación con técnicas de estimación de hiperparámetros basadas en métodos bayesianos, para obtener una aproximación precisa de la solución al problema inverso. Esto implica la reconstrucción de la imagen del obstáculo a partir de las mediciones del tensor de Cauchy, logrando niveles de error que permiten la recuperación de las geometrías asociadas a una familia particular de obstáculos en el sistema de Stokes.

Palabras clave: Problemas Inversos Geométricos, Aprendizaje Profundo, Sistema de Stokes, Reconstrucción numérica.

Abstract

This work addresses the study of the inverse problem of obstacle recovery in the context of the Stokes system through a Deep Learning-based approach. A method was developed, which consists of a process in which binary images of obstacles are generated, and the corresponding solutions to the direct problem are computed. From these simulations, a dataset is constructed, enabling the inverse problem measurements for various obstacles in a region of interest on the outer boundary. Subsequently, various convolutional neural networks are employed in combination with hyperparameter estimation techniques based on Bayesian methods to obtain an accurate approximation of the inverse problem solution. This involves the reconstruction of the obstacle image from the measurements of the Cauchy tensor, achieving error levels that allow the recovery of geometries associated with a particular family of obstacles in the Stokes system.

Keywords: Geometrical Inverse Problems, Deep Learning, Stokes System, Numerical Reconstruction.

Agradecimientos

Hay personas que sin duda han aportado a seguir este camino para llegar hasta acá y poder sacar adelante esta carrera que significan mucho para mí. Mi Papá y Mamá fueron mi ejemplo, aquellos que durante todos estos años me dieron amor y apoyo incondicional. Les agradezco eternamente por haberme dado la vida y la oportunidad de estudiar, sé que ahora están desde el infinito apoyándonos a mi y mi hermano. Su recuerdo es mi fuerza y su amor es mi guía. Siempre estarán en mi corazón.

También agradecer a mi hermano quién es una parte importante desde que llegó, tenemos muchas historias juntos y espero que también logres tus metas en todo lo que realizas, siempre te voy a apoyar independiente de lo que pase, para que algún día todos esos sueños que tuvimos de niños sean reales. Por muy complejo que te parezca algo sigue perseverando, todo esto está hecho para seres humanos.

Una de las personas más importantes que tuve en el último tiempo es Catalina. He sido un afortunado de contar contigo. Me alegras los días con tu forma de ser, y nunca olvidaré tus consejos, tus historias y esos días de estudio que compartimos donde a la par me ayudaste a salir adelante en momentos muy complejos. Este logro puedes sentirlo también como algo tuyo, gracias por la motivación que me diste y por la confianza de contar conmigo, estaré siempre contigo y Simbita. Gracias por todo Catita.

A mis profesores Rodrigo y Eduardo que me vieron crecer dentro de la carrera, desde que estaba en primer año iba a la oficina de Eduardo para aprender lo que fuera y prestaba mucha atención en cada detalle y consejo. Su conocimiento me formó como un matemático riguroso y estoy muy contento de haber sido parte de su línea de investigación en Control y Problemas Inversos donde los profesores tuvieron un excelente recibimiento y son de los mejores que pude encontrar durante la carrera.

También agradecer a algunos compañeros que estuvieron presentes, a Benito un gran compañero que espero termines este camino, eres ejemplo de perseverancia. A Felipe con Nathaly que siempre estuvieron atentos y fueron parte de muchos momentos. También a Eduardo (compañero de varias batallas), Heavy, Javier que fueron un apoyo importante durante los primeros años de carrera.

Por último agradecer el financiamiento de la beca de Magister USM y la beca Financiamiento Basal ANID Centro Nacional de Inteligencia Artificial CENIA FB210017, los cuales me permitieron costear mis estudios y gastos durante este período.

Dedicado a Patricia y Guillermo. Dos estrellas en el infinito.

Índice general

Parte I

Introducción

1 Objetivos y Estructura

El objetivo de esta tesis es investigar la aplicación de técnicas de aprendizaje profundo en problemas inversos de recuperación de obstáculos en mecánica de fluidos. Existe una categoría de problemas inversos en mecánica de fluidos que buscan determinar la distribución de obstáculos dentro de un medio en función de las mediciones del flujo. Estos problemas son de gran importancia en aplicaciones industriales, como la detección de obstrucciones en tuberías, la localización de objetos submarinos y la caracterización de la estructura de la corteza terrestre.

En este trabajo, abordamos este desafío utilizando un enfoque basado en Aprendizaje Profundo, específicamente mediante el uso de Redes Neuronales Convolucionales (CNN). Nuestra estrategia combina la generación de datos de obstáculos a través de imágenes binarias de alta resolución (144×144 píxeles) con la obtención de mediciones de esfuerzos en la frontera exterior del dominio mediante el tensor de Cauchy. Esta información se utiliza para entrenar una red neuronal capaz de aproximar la inversa del operador de Poincaré-Steklov y, por lo tanto, reconstruir la geometría del obstáculo en cuestión.

Esta tesis propone la utilización de técnicas de Aprendizaje Profundo para abordar estos problemas inversos. El aprendizaje profundo ha demostrado ser efectivo en diversas tareas de procesamiento de datos, incluyendo la clasificación de imágenes, el procesamiento del lenguaje natural y el análisis de series temporales. Sin embargo, su aplicación en problemas inversos en mecánica de fluidos aún es limitada. Una ventaja importante de estas técnicas es su capacidad para aprender patrones de manera rápida a partir de los datos, una capacidad que actualmente falta en los algoritmos de reconstrucción de obstáculos. Las tareas relacionadas con la inferencia ofrecen la ventaja de realizar predicciones para datos no vistos por una red neuronal durante el entrenamiento y poseen la velocidad computacional necesaria para realizar diversos análisis.

La estructura de esta tesis se divide en varias secciones clave. En la Sección 1, se proporcionarán fundamentos teóricos sobre problemas inversos geométricos, donde se presentará el sistema de Stokes y se discutirán algunos teoremas relacionados con la existencia y unicidad del problema inverso. En la Sección 2, se ofrecerá una breve introducción al aprendizaje profundo, con un enfoque especial en las redes neuronales convolucionales y las técnicas utilizadas a lo largo de este trabajo, como Backpropagation, capas deconvolucionales, conexiones skip

y técnicas de estimación de hiperparámetros bayesianas. La Sección 3 se centrará en la generación de geometrías aleatorias, representadas como imágenes binarias en MATLAB, y en la creación de datos para las mediciones del problema inverso utilizando FreeFEM++. Aquí se explicará la construcción del conjunto de datos. En la Sección 4, se detallará la arquitectura de la red neuronal utilizada, junto con consideraciones importantes durante el proceso de entrenamiento. Además, se evaluará el rendimiento de la red mediante diversos análisis estadísticos. Finalmente, en la Sección 5, se ampliará la solución propuesta y se presentarán algunas conclusiones derivadas de este trabajo de investigación.

2 Preliminares

2.1. Problemas Inversos

Los problemas inversos son desafíos matemáticos que surgen cuando el objetivo es recuperar información interna u oculta a partir de observaciones externas o datos ruidosos. Un ejemplo ilustre de un problema inverso es la tomografía, donde se busca la reconstrucción de una imagen médica a partir de datos capturados mediante rayos X. Estos problemas tienen aplicaciones en diversas áreas del conocimiento, incluyendo la medicina y la geofísica. En el contexto de esta tesis, nos enfocaremos en la teoría de problemas inversos en ecuaciones en derivadas parciales (EDP). En términos generales, un problema inverso puede ser formulado como

$$Ax = y \tag{2.1}$$

Donde A es un operador continuo no necesariamente lineal que actua desde un subconjunto X de un Espacio de Banach en un subconjunto Y de otro Espacio de Banach, el objetivo es que dado $y \in Y$, ser capaces de encontrar $x \in X$ tal que se cumpla la ecuación 2.1. Se estudia la capacidad de poder resolver 2.1 cuando A^{-1} no existe mediante la teoría de Tikhnov [10]. Para estudiar esta teoría es necesario comprender la noción de problema **bien puesto** en el sentido de Hadamard

Definición 2.1. Diremos que la ecuación 2.1 representa un problema bien puesto en el sentido de Hadamard si se cumplen las siguientes condiciones

1. **(Unicidad)** Para todo $y \in Y$, existe un único $x \in X$ que satisface 2.1.
2. **(Existencia)** Para todo $y \in Y$, existe por lo menos un $x \in X$ que satisface 2.1.
3. **(Estabilidad)** Sean $(x, y), (x^*, y^*)$ pares de soluciones de 2.1. Si $\|y - y^*\|_Y \to 0$ entonces $\|x - x^*\|_X \to 0$.

Estas tres condiciones en conjunto determinan si un problema inverso está **bien puesto** en el sentido de Hadamard. En otras palabras, un problema inverso se considera bien formulado si cumple con estas tres condiciones fundamentales, lo que garantiza la existencia, unicidad y estabilidad de la solución. Si alguna de estas condiciones no se cumple, el problema inverso

se considera **mal puesto**, lo que significa que su solución puede ser ambigua, inexistente o extremadamente sensible a errores en los datos observados, lo que hace que su resolución sea problemática o imposible. En general, cuando se tratan problemas inversos de EDP se considera X e Y como subconjuntos abiertos de los espacios clásicos $C^k(\Omega)$, $H^k(\Omega)$ o subespacios finito dimensionales.
Un ejemplo de problema inverso es el Problema de Calderón [6], el cual consiste en que dado un dominio acotado $\Omega \subset \mathbb{R}^n$ con frontera suave y $\gamma(x)$ una función acotada y positiva que representa la conductividad de Ω, sobre la frontera $\partial\Omega$ se realiza una perturbación $f \in H^{1/2}(\partial\Omega)$ y se induce un potencial $u \in H^1(\Omega)$ que resuelve el problema de Dirichlet

$$\begin{cases} \operatorname{div}(\gamma \nabla u) = 0, & \text{en } \Omega \\ u = f, & \text{sobre } \partial\Omega \end{cases} \tag{2.2}$$

Sobre este sistema se define el operador Dirichlet-Neumann dado por

$$\Lambda_\gamma(f) = \gamma \frac{\partial u}{\partial n}\Big|_{\partial\Omega}$$

El problema inverso de Calderón consiste en recuperar γ a partir de Λ_γ. Si se asume que existe $\varepsilon_0 > 0$ tal que $\gamma \geq \varepsilon_0$ y además que γ es medible, es posible demostrar que existe una única solución $u \in H^1(\Omega)$ al problema directo 2.2, cuando $\partial\Omega$ es Lipschitz. Más aún, hay estabilidad de u respecto a f en las normas de sus espacios correspondientes. Por lo tanto, el problema directo está bien puesto. Para el estudio del problema inverso, existen diferentes teoremas que aseguran que el problema está bien puesto bajo ciertas condiciones, a continuación damos algunos teoremas que dan nociones de unicidad y estabilidad

Teorema 2.1. (Unicidad) Sea $\Omega \subset \mathbb{R}^n$ un dominio acotado de clase C^2 y además, sean $\gamma_1, \gamma_2 \in C^2(\bar{\Omega})$ funciones positivas si $n \geq 3$ y en $L^\infty(\Omega)$ si $n = 2$. Se tiene que

$$\Lambda_{\gamma_1} = \Lambda_{\gamma_2} \implies \gamma_1 = \gamma_2 \quad \text{en } \Omega.$$

Teorema 2.2. (Estabilidad) Sea $\Omega \subset \mathbb{R}^n$ un dominio C^∞ acotado, $n \geq 3$, y sean $\gamma_1, \gamma_2 \in H^{s+2}(\Omega)$ funciones positivas con $s > \frac{n}{2}$ que satisfacen

$$\frac{1}{M} \leq \gamma_j \leq M, \quad \|\gamma_j\|_{H^{s+2}(\Omega)} \leq M, \quad j = 1, 2$$

Existen constantes $C = C(\Omega, n, M, s) > 0$ y $\sigma = \sigma(n, s) \in]0, 1[$ tal que

$$\|\gamma_1 - \gamma_2\|_{L^\infty(\Omega} \leq \omega(\|\Lambda_{\gamma_1} - \Lambda_{\gamma_2}\|_{\mathcal{L}(H^{1/2}(\partial\Omega), H^{-1/2}(\partial\Omega))})$$

En donde ω es un módulo de continuidad que cumple

$$\omega(t) \leq C|\log t|^{-\sigma}, \quad t \in]0, 1/e[$$

Estos resultados consideran el caso de tener a disposición todos los datos, es decir, conocemos todas las mediciones en la frontera, lo cual en la realidad no ocurre frecuentemente y motiva el estudio de los problemas inversos con datos parciales. Otro aspecto importante es que numéricamente seamos capaces de realizar la reconstrucción con algoritmos convergentes.

2.2. Mecánica de fluidos

En esta sección nos basaremos en [4]. Un fluido consiste en una gran cantidad de moléculas en movimiento sin una forma precisa en reposo (a diferencia de un sólido). Un enfoque inicial para estudiar un fluido podría implicar escribir las ecuaciones de movimiento para cada una de las partículas considerando sus interacciones (colisiones, caracterizadas por el camino libre medio, pero también interacciones a larga distancia). Un enfoque de este tipo, conocido como enfoque estadístico, condujo a la teoría cinética de los fluidos y a la mecánica estadística en general. Este enfoque se remonta al trabajo de Maxwell a mediados del siglo XIX.

En un gran número de situaciones físicas, si la densidad media del fluido estudiado no es demasiado baja (es decir, si las longitudes características del problema son grandes en comparación con el camino libre medio de las partículas), entonces el fluido se puede considerar como un medio continuo. Esto significa que el movimiento de las partículas se puede considerar como un todo y no de manera independiente para cada partícula. Por lo tanto, podemos definir cantidades macroscópicas que caracterizan el sistema: densidad, velocidad, y así sucesivamente.

Este enfoque simplificado nos permite describir el comportamiento de los fluidos en una amplia gama de situaciones físicas, lo que resulta fundamental en la mecánica de fluidos y otras disciplinas relacionadas. La siguiente sección explorará en detalle los principios y las ecuaciones que rigen el comportamiento de los fluidos en estado continuo, proporcionando una base sólida para comprender los fenómenos y aplicaciones posteriores en este campo.

La hipótesis del medio continuo nos permite considerar el movimiento de las moléculas de fluido como un elemento y no de forma individual. Consideremos $\omega \subset \mathbb{R}^n$ el volumen de espacio que ocupa el fluido en un instante de tiempo inicial. El flujo se puede definir por una sucesión de funciones biyectivas $(\varphi_t)_t$ definidas en ω tal que para cualquier $\Omega_0 \subset \omega$, el conjunto $\Omega_t = \varphi_t(\Omega_0)$ contiene exactamente, en tiempo t, las mismas moléculas de fluido que estaban presentes en Ω_0 en un instante inicial. Tal familia de conjuntos $(\Omega_t)_t$ es llamada elemento de fluido. En el caso que Ω_0 es un singleton, esto es, $\Omega_0 = \{x_0\}$, lo llamaremos partícula del fluido.

Las ecuaciones que modelan el movimiento de un fluido se pueden deducir a partir de leyes físicas como la conservación de la masa, conservación del momentum y la conservación de energía. Para realizar una deducción consideremos $(\Omega_t)_t$ un elemento de fluido arbitrario.

2.2.1. Conservación de la Masa

La propiedad de conservación de la masa establece que para cualquier volumen inicial Ω_0, la masa total de la materia contenida en $\Omega_t = \varphi_t(\Omega_0)$ es la misma que la contenida en Ω_0. Usando la noción de densidad, representada por $\rho = \rho(t, x)$, podemos expresar esto como

$$\frac{d}{dt}\int_{\Omega_t} \rho \, dx = 0 \tag{2.3}$$

Para desarrollar esta expresión usamos el Teorema de Tranporte que nos muestra como derivar una cantidad integrada sobre un elemento de fluido.

Teorema 2.3. Para toda función $f \in C^1$ con respecto a $(t, x) \in \mathbb{R} \times \mathbb{R}^3$, tendremos que

$$\frac{d}{dt}\int_{\Omega_t} f(t,x) \, dx = \int_{\Omega_t} \left(\frac{\partial f}{\partial t} + \operatorname{div}(fv) \right) dx, \quad \forall t \in \mathbb{R}$$

Donde $v(t,x) = \dfrac{\partial x}{\partial s}(s,t,x)\Big|_{s=t}$.

En el caso que f sea un campo vectorial, el teorema de transporte se escribe como

$$\frac{d}{dt}\int_{\Omega_t} f(t,x) \, dx = \int_{\Omega_t} \left(\frac{\partial f}{\partial t} + \operatorname{div}(f \otimes v) \right) dx, \quad \forall t \in \mathbb{R}$$

Utilizando esto en 2.3 tendremos que

$$\int_{\Omega_t} \left(\frac{\partial \rho}{\partial t} + \operatorname{div}(\rho v) \right) dx = 0 \quad \forall t \in \mathbb{R} \tag{2.4}$$

Asumiendo la suficiente regularidad para ρ y v, además de la arbitrariedad del elemento de fluido, es posible deducir la ecuación de conservación de masa (también conocida como ecuación de continuidad o balance de masas)

$$\frac{\partial \rho}{\partial t} + \operatorname{div}(\rho v) = 0 \tag{2.5}$$

2.2.2. Conservación del momentum lineal

Aplicaremos la segunda ley de Newton para expresar la evolución del momento lineal para un elemento de fluido. El momentum lineal total contenido en un elemento de fluido es la cantidad

$$\int_{\Omega_t} \rho v \, dx$$

La segunda ley de Newton establece la tasa de cambio del momentum total es igual a la suma de las fuerzas externas aplicadas al fluido. En general, existen dos tipos de fuerzas

- Fuerzas de cuerpo:

  $$\int_{\Omega_t} \rho f \, dx$$

  Donde en muchas situaciones, la densidad de masa de fuerzas f (que representa las fuerzas externas experimentadas por el fluido) es reducida a la aceleración debido a la gravedad.

- Fuerzas de superficie:

  $$\int_{\partial\Omega_t} \tau(\nu) \, dy$$

  Donde ν representa el vector normal unitario exterior a la superficie $\partial\Omega_t$. Estas fuerzas se ejercen sobre el elemento fluido investigado.

Por lo tanto, usando el Teorema de Transporte para el caso vectorial tendremos que

$$\int_{\Omega_t} \left(\frac{\partial \rho v}{\partial t} + \operatorname{div}(\rho v \otimes v) \right) dx = \int_{\Omega_t} \rho f \, dx + \int_{\partial\Omega_t} \tau(\nu) \, dy \tag{2.6}$$

Esto para cualquier elemento de fluido $(\Omega_t)_t$.

2.2.3. Conservación de la energía

Comenzamos recordando la primera ley de la termodinámica, que establece que la tasa de cambio de la energía total de un sistema es la suma de la potencia de las fuerzas mecánicas y la tasa de intercambio de calor con el entorno.
La energia total por unidad de volumen viene dada por la cantidad $\rho E = \rho\big(e + \frac{1}{2}|v|^2\big)$, donde e es la energía interna específica relacionada al estado termodinámico del fluido y $\frac{1}{2}\rho|v|^2$ representa la energía cinética por unidad de volumen.

La potencia de las fuerzas mecánicas puede ser descompuesta en un término asociado a las fuerzas de cuerpo y otro a las fuerzas aplicadas en la superficie:

$$\dot{W} = \int_{\Omega_t} \rho f v \, dx + \int_{\partial\Omega_t} \tau(\nu) v \, dy \tag{2.7}$$

Si suponemos que no existe una fuente de calor interna (por radiación, por ejemplo) dentro del volumen; entonces, la tasa de entrada de calor en el volumen Ω_t toma la siguiente forma:

$$\dot{Q} = -\int_{\partial\Omega_t} \Phi(t, y, \nu) \, dy$$

En esta suposición, consideramos que el término Φ para la transferencia de calor a través de la superficie de Ω_t depende únicamente de la posición, el tiempo y la normal exterior de Ω_t en el punto considerado. Con esto el balance de energía queda de la siguiente forma

$$\frac{d}{dt}\int_{\Omega_t} \rho E \, dx = \int_{\Omega_t} \rho f v \, dx + \int_{\partial\Omega_t} \tau(\nu) v \, dy - \int_{\partial\Omega_t} \Phi(t, y, \nu) \, dy$$

Aplicando el Teorema del Transporte se obtiene la siguiente ecuación de conservación

$$\int_{\Omega_t} \left(\frac{\partial \rho E}{\partial t} + \operatorname{div}(\rho E v) \right) dx = \int_{\Omega_t} \rho f v \, dx + \int_{\partial\Omega_t} \tau(\nu) v \, dy - \int_{\partial\Omega_t} \Phi(t, y, \nu) \, dy \tag{2.8}$$

Ahora, utilizaremos el siguiente Teorema que muestra la forma general de los esfuerzos τ estipulando que el esfuerzo es lineal con respecto al vector normal a la superficie del elemento de volumen considerado.

Teorema 2.4. (Esfuerzo de Cauchy) Asumamos que ρ, v y f son regulares. Además, el mapeo $(t, y) \mapsto \tau(t, y, \nu)$ es continuo para ν fijo. Entonces, existe una función tensor-valuada $(t, y) \mapsto \sigma(t, y)$ tal que para todo $(t, y), \nu$ tenemos que

$$\tau(t, y, \nu) = \sigma(t, y) \cdot \nu$$

Observación 2.1. Es posible demostrar un resultado similar para el término que contiene la transferencia de calor Φ, en efecto, si ρ, E, v y f son funciones regulares y para todo ν la función $(t, y) \mapsto \Phi(t, y, \nu)$ es continua, entonces existe un campo vectorial continuo φ tal que

$$\Phi(t, y, \nu) = \varphi(t, y) \cdot \nu$$

Con esto y el Teorema de la Divergencia, se puede asegurar que

$$\int_{\partial\Omega_t} \tau(\nu)v \, dy = \int_{\partial\Omega_t} (\sigma(t,y)\cdot\nu)v \, dy = \int_{\partial\Omega_t} \nu\cdot(\sigma^T\cdot v) \, dy = \int_{\Omega_t} \operatorname{div}(\sigma^T\cdot v) \, dx \quad (2.9)$$

También, es posible demostrar que para ρ, E, v y f regulares que existe un campo vectorial continuo φ tal que $\Phi(t,y,\nu) = \varphi(t,y)\cdot\nu$. De igual forma, aplicando el teorema de la divergencia se puede concluir que la ecuación de evolución de la energía es

$$\int_{\Omega_t}\left(\frac{\partial\rho E}{\partial t} + \operatorname{div}(\rho E v)\right) dx = \int_{\Omega_t} \rho f v \, dx + \int_{\Omega_t} \operatorname{div}(\sigma^T\cdot v) \, dx - \int_{\Omega_t} \operatorname{div}(\varphi) \, dx \quad (2.10)$$

Como esto ocurre para cualquier elemento de fluido obtenemos la ecuación

$$\frac{\partial\rho E}{\partial t} + \operatorname{div}(\rho E v) - \operatorname{div}(\sigma^T\cdot v) + \operatorname{div}(\varphi) = \rho f\cdot v \quad (2.11)$$

Siguiendo el mismo procedimiento en la ecuación 2.6 se puede obtener la ecuación del momentum lineal

$$\frac{\partial\rho v}{\partial t} + \operatorname{div}(\rho v\otimes v) - \operatorname{div}\sigma = \rho f \quad (2.12)$$

A través de las ecuaciones de conservación del momento angular es posible probar que σ es simétrico. Utilizando las propiedades de la divergencia en 8.1 y 8.2 podemos escribir las ecuaciones del momentum lineal y de la energía como las siguientes:

- Ecuación del Momentum Lineal:

$$\rho\left(\frac{\partial v}{\partial t} + (v\cdot\nabla)v\right) - \operatorname{div}\sigma = \rho f \quad (2.13)$$

- Ecuación de Energía:

$$\rho\left(\frac{\partial E}{\partial t} + (v\cdot\nabla)E\right) - \operatorname{div}(\sigma\cdot v) + \operatorname{div}(\varphi) = \rho f\cdot v \quad (2.14)$$

En esta tesis estamos interesados en fluidos en reposo, en este sentido, un fluido en reposo es aquel donde los esfuerzos en un elemento de superficie actúan en la dirección opuesta a aquella normal a la superficie. Mas aún, el módulo de este esfuerzo es independiente de la dirección y es denotado por p, también conocido como presión hidrostática del fluido. Con

esto el esfuerzo en todos los puntos será $\tau(\nu) = -p\nu$, con $p \geq 0$. Esto quiere decir, que el tensor de esfuerzo puede ser escrito como

$$\sigma = -pI$$

Pero, cuando el fluido está en movimiento, separamos los efectos de la presión y del movimiento expresando el tensor de esfuerzos como

$$\sigma = \mathcal{T} - pI$$

Donde $\mathcal{T}$ es conocido como tensor de deformaciones, este elemento puede probarse que tiene algunas propiedades respecto al tensor $e(v) = \frac{1}{2}(Dv + Dv^T)$. Estas son las siguientes:

1. El tensor $\mathcal{T}$ es un flujo, depende solo del tensor $e(v)$.
2. La dependencia entre $\mathcal{T}$ y $e(v)$ es lineal.
3. La relación que conecta $\mathcal{T}$ y $e(v)$ es isotrópica.

Un fluido que satisface tales propiedades se conoce como *Fluido Newtoniano*. Con esto se puede probar el siguiente resultado

Proposición 2.1. Para un fluido Newtoniano, se tiene que

$$\mathcal{T} = 2\mu e(v) + \lambda(\operatorname{div} v)I$$

Donde $\lambda, \mu \in \mathbb{R}$

En el caso de un fluido incompresible, esto es, el volumen de cualquier elemento de fluido es constante en el tiempo. Se puede probar que tal condición equivale a que el campo de velocidades v es solenoidal. Es decir,

$$\operatorname{div} v = 0$$

Con esto, si utilizamos la ecuación de continuidad 2.5 y desarrollamos el operador diferencial tendremos que por condición de incompresibilidad

$$\frac{\partial \rho}{\partial t} + v \cdot \nabla \rho + \rho(\operatorname{div} v) = 0 \iff \frac{\partial \rho}{\partial t} + v \cdot \nabla \rho = 0$$

Esta última expresión es también conocida como la derivada material $\frac{D\rho}{Dt}$ por lo que

$$\frac{D\rho}{Dt} = 0$$

Lo cual equivale a que ρ es constante a lo largo de las curvas características de v.
Al no considerar los efectos de la temperatura y considerar la densidad homogénea, el sistema que hemos modelado es el siguiente

$$\begin{cases} \rho\left(\dfrac{\partial v}{\partial t} + (v \cdot \nabla)v\right) - \operatorname{div} \sigma = \rho f \\ \operatorname{div} v = 0 \end{cases} \tag{2.15}$$

En particular, la siguiente sección dará nociones de como agregar un obstáculo dentro de este sistema, el cual será reducido para este caso de estudio, a un sistema estacionario y linealizado.

2.3. Problema a estudiar: Obstáculo inmerso en un fluido viscoso

En este trabajo, nos centramos en el sistema de Stokes que se define en un dominio acotado $\Omega \subset \mathbb{R}^N$, $D \Subset \Omega$ y consideramos una perturbación en la frontera $\varphi \in H^{1/2}(\partial\Omega)^N$ que satisface la condición de compatibilidad

$$\int_{\partial\Omega} \varphi \cdot n \, ds = 0,$$

Consideramos $(v, p) \in H^1(\Omega \backslash \overline{D}) \times L^2(\Omega \backslash \overline{D})$ como la solución del sistema de Stokes:

$$\begin{cases} \operatorname{div} \sigma(v,p) = 0, & \Omega \backslash D \\ \operatorname{div} v = 0, & \Omega \backslash D \\ v = 0, & \Gamma_1 \cup \Gamma_2 \cup \partial D \\ v = \varphi, & \Gamma_0 \\ \sigma(v,p) \cdot n = 0, & \Gamma_N \end{cases} \tag{2.16}$$

Donde $\partial\Omega = \Gamma_0 \cup \Gamma_1 \cup \Gamma_2 \cup \Gamma_N$ y la partición de la frontera se realiza por abiertos (relativos a la topología de la frontera) disjuntos. Además, el tensor de esfuerzos σ se define como

$$\sigma(v,p) = -pI + 2\nu e(v)$$

con I como la matriz identidad de $N \times N$, $\nu > 0$ la viscosidad cinemática, y $e(v) = \frac{1}{2}(Dv + Dv^T)$ denominado tensor de deformaciones.
La pregunta central a resolver en este trabajo es si, a través de mediciones sobre $\Gamma_1 \cup \Gamma_2$, es posible reconstruir la forma del obstáculo D.

Una pieza clave que hace que este problema tenga solución es la propiedad de continuación única para el sistema de Stokes. Esta propiedad, relacionada con la ecuación de Laplace, es esencial para establecer la factibilidad del problema [8]. La estabilidad en el contexto del problema de Cauchy mal puesto es otro aspecto crucial en esta área de estudio. Este tema se aborda en varios artículos, y se han desarrollado resultados de estabilidad, como la desigualdad de tres bolas [11]. Además, se han aplicado enfoques de regularización, como la minimización de la funcional de Kohn-Vogelius, para abordar de manera efectiva el problema de Cauchy mal puesto.
Una vez que se establece la propiedad de continuación única, se deriva la propiedad de unicidad para el problema inverso de obstáculos. Esto es esencial para garantizar que las soluciones sean únicas y evita ambigüedades en la identificación de obstáculos. En este sentido, en [1]

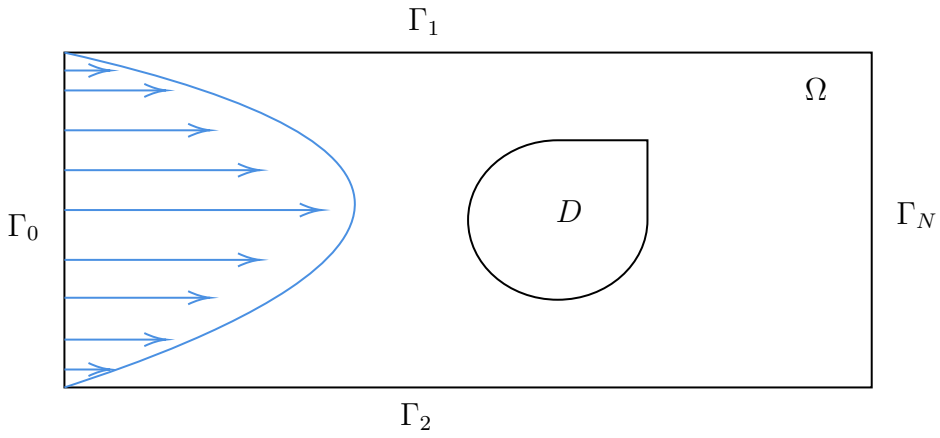


Figura 2.1: Configuración geométrica del sistema

se tiene un resultado de unicidad para el problema inverso sobre un conjunto de dominios admisibles

$$D_{ad} = \{D \Subset \Omega : D \text{ es abierto, Lipschitz y } \Omega\backslash\overline{D} \text{ es conexo}\}$$

Teorema 2.5. Sea $\Omega \subset \mathbb{R}^N$, $N = 2$ ó $N = 3$, un dominio Lipschitz acotado, y $\Gamma \subset \partial\Omega$ abierto no vacío. Sean $D_0, D_1 \in D_{ad}$ y $\varphi \in H^{3/2}(\partial\Omega)^N$ con $\varphi \neq 0$ que cumple la condición de compatibilidad. Para $\varepsilon = 0$ ó $\varepsilon = 1$, consideremos (v_j, p_j), $j = 0, 1$ soluciones de

$$\begin{cases} -\operatorname{div}(\sigma(v_j, p_j)) + \varepsilon \operatorname{div}(v \otimes v) = 0 & \text{en } \Omega\backslash\overline{D}_j \\ \operatorname{div}(v_j) = 0 & \text{en } \Omega\backslash\overline{D}_j \\ v_j(s) = \varphi(s) & \text{sobre } \partial\Omega \\ v_j(s) = 0 & \text{sobre } \partial D_j \end{cases} \tag{2.17}$$

Si (v_j, p_j) son tales que

$$\sigma(v_0, p_0) \cdot n = \sigma(v_1, p_1) \cdot n \quad \text{sobre } \Gamma$$

Entonces $D_0 = D_1$.

La literatura también ha explorado una variedad de métodos de identificación efectivos. Entre ellos se encuentran métodos de optimización basados en la parametrización del obstáculo, técnicas que utilizan derivadas de forma del obstáculo [1] en donde si consideramos σ_m una medición de la componente normal del tensor de esfuerzos en $\Gamma_m \subset \partial\Omega$ y a priori se sabe que D es un círculo, entonces para identificar D basta encontrar los parámetros (a, b) que

representen el centro del círculo y $r > 0$ que representa el radio de tal forma que $\overline{B(a,b,r)} \subset \Omega$ y minimizan el funcional

$$J(a,b,r) = \int_{\Gamma_m} |\sigma(v,p) \cdot n - \sigma_m|^2 \, ds$$

Donde debido a los resultado de identificabilidad se puede asegurar que el problema de minimización

$$\min_{(a,b,r)\in S} J(a,b,r) \quad , S = \{(a,b,r) \in \mathbb{R}^3 : \overline{B(a,b,r)} \subset \Omega\}$$

Tiene una única solución y en particular [2] da un algoritmo de reconstrucción para encontrar tales parámetros a través del método de Shape Differentiation con la restricción que D debe ser un obstáculo con frontera que posea parametrización suave.
Por otro lado, en [3] se da un enfoque exterior a la resolución del problema inverso que consiste en definir una sucesión decreciente de dominios que convergen al obstáculo en el sentido de Hausdorff, de manera más precisa, se tiene un enfoque iterativo basado en una combinación de dos técnicas: El método de Cuasi-reversibilidad para actualizar la solución del problema de Cauchy mal puesto fuera del obstáculo obtenido en la iteración anterior y un método de level set para actualizar el obstáculo con la ayuda de la solución obtenida en la iteración anterior.
La investigación previa proporciona una sólida base de conocimientos y metodologías para resolver el problema inverso de obstáculos en el sistema de Stokes. En este trabajo, buscamos construir sobre este fundamento y proponer un enfoque novedoso que aprovecha técnicas de aprendizaje profundo, específicamente redes neuronales convolucionales, para mejorar la precisión de las reconstrucciones de obstáculos a partir de mediciones en la frontera.

3 Introducción al Deep Learning

Para encontrar una aproximación de la inversa del operador de Poincaré Steklov se han utilizado diferentes enfoques basados en técnicas de optimización de formas. En este artículo trabajaremos en torno a un enfoque basado en Deep Learning donde a través de diferentes redes neuronales llegaremos a modelos que son capaces de aprender e identificar patrones en el conjunto de datos generado.
La unidad central sobre la cual se enmarca el Deep Learning son las redes neuronales artificiales, en nuestro caso utilizaremos redes neuronales convolucionales las cuales han mostrado ser eficientes en tareas de procesamiento de imágenes donde podemos destacar arquitecturas como VGG16 y AlexNet.

3.1. Contexto Histórico

La historia del aprendizaje profundo tiene sus raíces en la biología y la búsqueda por replicar la inteligencia humana en sistemas artificiales. En este contexto, la neurona biológica ha sido una fuente de inspiración fundamental. Una neurona es la unidad básica de procesamiento en el cerebro humano y está compuesta por un cuerpo celular, dendritas que reciben señales y un axon que transmite señales a otras neuronas a través de sinapsis. Esta estructura fue adaptada al campo de la inteligencia artificial dando lugar a la creación de redes neuronales artificiales.

Una red neuronal es un modelo matemático inspirado en la organización de las neuronas en el cerebro. Consiste en un conjunto de unidades de procesamiento interconectadas por nodos y organizadas en capas. Cada neurona artificial realiza una operación simple en sus entradas, seguidas de una función de activación no lineal. La información fluye a través de las capas, transformándose y refinándose en cada paso. La verdadera potencia de las redes neuronales radica en su capacidad para aprender patrones y representaciones de datos a través del ajuste de los pesos sinápticos que conectan las neuronas.

Formalmente, una red neuronal artificial se define a partir de la morfología que tienen sus capas, neuronas y dimensionalidad de la señal de entrada y salida con la siguiente noción

Definición 3.1. (Red neuronal) Sea $d \in \mathbb{N}$ una dimensión, $L \in \mathbb{N}$ el número de capas cada una con $l_i \in \mathbb{N}$ neuronas para cada $i \in \{1, \ldots, L\}$. Consideremos una dimensión de

salida $k = I_L$, matrices de pesos $(W_i)_{i=1}^L$, vectores de bias $(b_i)_{i=1}^L$ y finalmente una función de activación $\sigma : \mathbb{R} \to \mathbb{R}$. Dados los parámetros (d, L, l_i, W_i, b_i) antes mencionados y $\theta = (W_i, b_i)_{i=1}^L$, definimos la red neuronal $\mathcal{U} : \mathbb{R}^d \to \mathbb{R}^{I_L}$ como

$$\mathcal{U}(x; \theta) = (A_L \circ \sigma \circ A_{L-1} \circ \cdots \circ A_2 \circ \sigma \circ A_1)(x)$$

donde $A_i : \mathbb{R}^{l_{i-1}} \to \mathbb{R}^{l_i}$ son funciones lineales dadas por $A_i(x) = W_i x + b_i$ y σ es aplicada componente a componente.

El rango de funciones computables variando las dimensiones de los parámetros $\theta = (W_i, b_i)$ es llamado el **Espacio de las redes neuronales**, denotado por $\mathcal{N}$. Este espacio es de alto interés dado que cumple propiedades de aproximación universal que fueron primeramente enunciadas y demostradas por Cybenko [7], este establece una importante propiedad de las redes neuronales feedforward. Supongamos que tenemos una red neuronal feedforward con funciones de activación sigmoidales. Denotemos esta red como $\mathcal{U} \in \mathcal{N}$. El teorema establece lo siguiente:

Teorema 3.1. **(Cybenko)** Dado un dominio compacto $\Omega \subset \mathbb{R}^n$ y una función continua $f : \Omega \to \mathbb{R}$, para cualquier $\varepsilon > 0$, existe una red neuronal $\mathcal{U} \in \mathcal{N}$ con una arquitectura adecuada (número suficiente de neuronas en capas ocultas) tal que:

$$\sup_{x \in \Omega} |f(x) - \mathcal{U}(x)| < \epsilon$$

En otras palabras, $\mathcal{U}$ puede aproximar la función continua f con un error arbitrariamente pequeño en todo el dominio Ω. Dado que el teorema de aproximación universal, como el formulado por Cybenko, asegura solo la existencia de una red neuronal capaz de aproximar cualquier función continua dentro de un margen de error arbitrariamente pequeño, surge una pregunta fundamental: ¿Cómo encontramos efectivamente los parámetros de una red neuronal que logre esta aproximación? Aquí es donde el algoritmo de backpropagation desempeña un papel crucial. El backpropagation es un método eficiente para la estimación de parámetros óptimos en redes neuronales, basado en el principio de optimización del gradiente. Este algoritmo permite ajustar los pesos y sesgos de la red de manera que se minimice una función de costo, usualmente una medida del error entre las salidas de la red y las salidas deseadas.

Definición 3.2. (Backpropagation) Sea $\mathcal{U}$ una red neuronal con arquitectura y parámetros definidos como anteriormente, y sea $L(\mathcal{U}(x), y)$ una función de pérdida que mide el error entre la salida de la red $\mathcal{U}(x)$ y la salida deseada y. El algoritmo de backpropagation busca minimizar L a través de la actualización iterativa de los parámetros $\theta = (W_i, b_i)$ en la dirección opuesta al gradiente de la función de pérdida con respecto a estos parámetros. El proceso se realiza en dos pasos:

1. **Propagación hacia adelante:** Se calculan las activaciones de la red para una entrada dada, pasando la información a través de las capas de la red desde la entrada hasta la salida.

2. **Retropropagación del error:** Se calcula el gradiente de la función de pérdida con respecto a cada uno de los parámetros, propagando el error desde la salida hacia la entrada. Luego, se actualizan los parámetros utilizando un algoritmo de optimización, como el descenso del gradiente.

Este proceso se repite iterativamente, mejorando progresivamente los parámetros de la red para minimizar el error de salida.

En este contexto, backpropagation es más que un simple mecanismo de ajuste: es una herramienta esencial que permite a las redes neuronales alcanzar su potencial como aproximadores universales, tal como lo establece el teorema de Cybenko. A través de la aplicación iterativa del backpropagation, una red neuronal puede aprender a modelar complejas funciones y patrones, haciendo efectiva la promesa teórica de la aproximación universal en aplicaciones prácticas. Describamos en detalle ambas etapas del Backpropagation:

Propagación hacia adelante: Consideramos una red neuronal con L capas. La salida de la neurona j en la capa l se denota como $a_j^{(l)}$. Las activaciones se calculan como:

$$a_j^{(l)} = \sigma\left(\sum_k w_{jk}^{(l)} a_k^{(l-1)} + b_j^{(l)}\right)$$

donde σ es la función de activación, $w_{jk}^{(l)}$ es el peso entre la neurona k en la capa $l-1$ y la neurona j en la capa l, y $b_j^{(l)}$ es el sesgo para la neurona j en la capa l.

Retropropagación del error: El error en la capa de salida se calcula utilizando la función de pérdida L, comparando la salida obtenida $a^{(L)}$ y la salida deseada y. El error en la capa de salida es:

$$\delta^{(L)} = \nabla_a L \odot \sigma'(z^{(L)})$$

El error en las capas ocultas se retropropaga como:

$$\delta^{(l)} = \left((w^{(l+1)})^T \delta^{(l+1)}\right) \odot \sigma'(z^{(l)})$$

donde $(w^{(l+1)})^T$ es la transpuesta de la matriz de pesos entre la capa l y $l+1$, y $\odot$ representa el producto de Hadamard.

Los gradientes de la función de pérdida respecto a los pesos y sesgos se calculan como:

$$\frac{\partial L}{\partial w_{jk}^{(l)}} = a_k^{(l-1)} \delta_j^{(l)}$$

$$\frac{\partial L}{\partial b_j^{(l)}} = \delta_j^{(l)}$$

Estos gradientes se utilizan para actualizar los pesos y sesgos:

$$w_{jk}^{(l)} = w_{jk}^{(l)} - \eta \frac{\partial L}{\partial w_{jk}^{(l)}}$$

$$b_j^{(l)} = b_j^{(l)} - \eta \frac{\partial L}{\partial b_j^{(l)}}$$

donde η es la tasa de aprendizaje.

3.2. Redes Neuronales Convolucionales

El surgimiento de las redes neuronales convolucionales (Convolutional Neural Networks, CNNs) ha marcado un hito en la historia de la inteligencia artificial, ofreciendo avances significativos en tareas como el reconocimiento de imágenes, el análisis de vídeo y la interpretación de datos complejos y multidimensionales. Este capítulo tiene como objetivo brindar una exposición rigurosa del funcionamiento matemático de las CNNs, detallando la formulación matemática que respalda su arquitectura y explicando el algoritmo de retropropagación para su entrenamiento.

Una CNN se compone principalmente de una o más capas de convolución, seguidas generalmente por una función de activación no lineal. La operación de convolución se define como

$$f * g(t) = \int_{-\infty}^{+\infty} f(\tau)g(t-\tau)\, d\tau$$

Para funciones discretas, la convolución se puede escribir como

$$(f * g)[n] = \sum_{m=-\infty}^{+\infty} f[m]g[n-m]$$

En el contexto de una CNN, la operación de convolución se simplifica aún más debido a que disponemos de una cantidad finita de datos. Supongamos que tenemos una imagen I y un kernel K, ambos representados como matrices. En este caso, la convolución se calculará como

$$(I * K)[x, y] = \sum_{i=-a}^{a} \sum_{i=-b}^{b} I[x+i, y+j]K[i, j]$$

Otra propiedad que tienen las CNN son el uso de Capas de Pooling, el objetivo del pooling es reducir la dimensionalidad de la entrada. La operación de pooling más común es el max pooling que toma el máximo valor de un conjunto de elementos, matemáticamente se define para un filtro de tamaño s como

$$P(I)[x, y] = \max_{i=0,\ldots,s-1} \max_{j=0,\ldots,s-1} I[x+i, y+j]$$

3.3. Capas Deconvolucionales en Redes Neuronales

Las capas deconvolucionales, también conocidas como capas de transposición de convolución, constituyen un elemento esencial en la arquitectura de las Redes Neuronales Convolucionales (CNN) para aplicaciones de visión por computadora que implican la reconstrucción

de imágenes o segmentación semántica. Su función es realizar la operación inversa a la convolución, transformando características de representación condensada en reconstrucciones detalladas y de alta resolución.

3.3.1. Formalismo Matemático de la Deconvolución

La operación de deconvolución se define matemáticamente como sigue: para una entrada $X \in \mathbb{R}^{N\times N}$ y un filtro $W \in \mathbb{R}^{k\times k}$, la deconvolución se expresa mediante la relación

$$Y_{ij} = (W * X)_{ij} = \sum_{m=1}^{k}\sum_{n=1}^{k} W_{mn} X_{(i+m-1)(j+n-1)}, \tag{3.1}$$

donde $Y \in \mathbb{R}^{M\times M}$ con $M > N$, indicando así una expansión en la dimensión espacial de la entrada.

3.3.2. Implementación en Redes Neuronales

En el diseño de una CNN, las capas deconvolucionales se alternan comúnmente con funciones de activación no lineales, como ReLU, y en ocasiones con capas de normalización por lotes y dropout para la regularización. Se pueden añadir conexiones residuales, conocidas como skip connections, para facilitar la retroalimentación de la entrada original y mejorar el flujo de gradientes durante el entrenamiento.

La utilidad de las capas deconvolucionales es particularmente notoria en la tarea de reconstrucción de imágenes, donde la red progresivamente sintetiza la imagen a partir de los mapas de características latentes, atravesando múltiples niveles de representaciones jerárquicas. Según [12], estas capas son capaces de aprender descomposiciones convolucionales de manera no supervisada bajo una restricción de escasez, lo cual es fundamental para el desarrollo de características robustas y representativas para el análisis de imágenes.

3.4. Aprendizaje en Redes Neuronales

La característica esencial que se incorporará a la arquitectura de la red neuronal es la capacidad de utilizar conexiones de salto (skip connections), que consisten en establecer una conexión directa entre las capas de la red, permitiendo que la salida de una capa se agregue o concatene con la entrada original antes de pasar a las capas subsiguientes. Matemáticamente, esto se expresa como:

$$y = g(I + f)(x)$$

en donde g corresponderá a la función sigmoide y f es la salida de la red neuronal convolucional. Las ventajas inherentes a las skip connections incluyen la facilitación del entrenamiento profundo al mitigar la desaparición del gradiente, la mejora de la capacidad de aprendizaje al capturar características de alto y bajo nivel, la promoción de la generalización y la mitigación del problema de degradación de red en modelos profundos. Con la incorporación de estas conexiones, la salida del modelo de red resultará en una representación mejorada y más precisa de los datos de entrada, lo que contribuirá a mejorar su rendimiento en tareas específicas, como la resolución del problema inverso de recuperación de obstáculos en el sistema de Stokes.

Uno de los puntos importantes del Deep Learning es su capacidad de aprendizaje. En este sentido se hace la distinción entre lo que es aprendizaje y optimización pura puesto que difieren de algunas formas. En el aprendizaje de máquinas se considera una medida del rendimiento P que se define con respecto a un conjunto de test que no se manipula durante el entrenamiento. Por lo tanto, se realiza una optimización de P solo indirectamente minimizando una función de costo $J(\theta)$ que logre mejorar el valor de P, esto es diferente a la optimización pura donde se busca minimizar J como objetivo.
Típicamente, la función de costo puede ser escrita como un promedio sobre el conjunto de entrenamiento a través del valor esperado:

$$J(\theta) = \mathbb{E}_{(x,y)\sim\hat{p}_{data}} L(f(x;\theta), y) \tag{3.2}$$

donde L es la función de pérdida, $f(x;\theta)$ es el valor predicho cuando la entrada a la red neuronal es x y $\hat{p}_{data}$ es la distribución empirica. En el contexto de aprendizaje supervisado, y es la salida deseada (target).

En aprendizaje supervisado, la función de pérdida L mide la diferencia entre la predicción $f(x;\theta)$ y la etiqueta real. Dependiendo del problema se pueden escoger diferentes funciones de pérdida, por ejemplo para tareas de clasificación se suele utilizar la entropía cruzada, mientras que para problemas de regresión se suele usar el error cuadrático medio (MSE). Luego de esto se necesita un algoritmo de optimización para hallar los parámetros θ. Dentro del contex-

to de este artículo estaremos utilizando el algoritmo Adam (Adaptive Moment Estimation) es una extensión del Gradiente Descendiente estocástico (SGD) que utiliza técnicas para adaptar dinámicamente las tasas de aprendizaje durante el entrenamiento. En lugar de mantener una tasa de aprendizaje global constante, Adam mantiene una tasa de aprendizaje adaptativa para cada parámetros. Se introdujeron dos términos adicionales en el algoritmo conocidos como momentos.

Algorithm 1 Adam

Inicializar: $m_0 = 0, v_0 = 0, t = 0$
Hiperparámetros: α (tasa de aprendizaje), β_1, β_2 (factores de decaimiento), ϵ (evitar división por cero, usualmente $1e-8$)
for cada iteración t **do**
    $t \leftarrow t + 1$
    Calcular el gradiente $g_t = \nabla_\theta J(\theta)$
    Actualizar primer y segundo momentos:
    $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$
    $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$
    Corregir momentos para el sesgo inicial:
    $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$
    $\hat{v}_t = \frac{v_t}{1-\beta_2^t}$
    Actualizar los parámetros del modelo:
    $\theta = \theta - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$
end for

Con esto se estima θ, para prevenir el sobreajuste es común incorporar términos de regularización en la función de costo, en tal caso

$$J(\theta) = \mathbb{E}_{(x,y)\sim\hat{p}_{data}} L(f(x;\theta), y) + \lambda R(\theta)$$

Donde $R(\theta)$ puede ser una regularización L^2 o L^1 mientras que λ es un hiperparámetro que controla la importancia de la regularización. Otro aspecto importante que tendremos como regularización es el Dropout. La técnica de dropout es una estrategia de regularización utilizada durante el entrenamiento de redes neuronales. Su objetivo es prevenir el sobreajuste al introducir aleatoriamente la desactivación temporal de neuronas durante el proceso de entrenamiento.

Matemáticamente, durante el entrenamiento de una red neuronal, en cada paso de actualización de los pesos, se selecciona aleatoriamente un subconjunto de neuronas y se desactivan temporalmente. Esto equivale a eliminar sus conexiones y sus contribuciones al cálculo de sa-

lida de la red. En el siguiente paso de entrenamiento, se seleccionan diferentes neuronas para desactivar. Este proceso aleatorio de desactivación se repite en cada paso.

El dropout se puede expresar matemáticamente como sigue: Supongamos que tenemos una capa de neuronas con funciones de activación σ y entrada x. Durante el entrenamiento, para cada ejemplo de entrenamiento, aplicamos dropout independientemente a cada neurona con una probabilidad p, lo que significa que la neurona se apaga con probabilidad p. Esto se puede representar matemáticamente como:

$$y_i = \frac{1}{1-p} \cdot \sigma(w_i \cdot x + b_i) \cdot z_i$$

Donde:

- y_i es la salida de la neurona i.
- w_i y b_i son los pesos y sesgos de la neurona i.
- z_i es una variable aleatoria de Bernoulli con probabilidad $1 - p$.

El término $\frac{1}{1-p}$ se utiliza para compensar la desactivación para mantener la esperanza de la salida.

El backpropagation en capas deconvolucionales, también conocidas como capas de transposición convolucional o unpooling, es conceptualmente similar al backpropagation en capas convolucionales, pero con algunas diferencias clave en su implementación. Las capas deconvolucionales son comúnmente utilizadas en tareas como la segmentación semántica de imágenes o en redes generativas, donde se requiere aumentar las dimensiones espaciales de los mapas de características.

El propósito de una capa deconvolucional es aprender a mapear de una representación de menor resolución a una de mayor resolución. En términos de backpropagation, esto significa calcular los gradientes no solo con respecto a los pesos de los filtros deconvolucionales, sino también manejar adecuadamente las dimensiones aumentadas de los mapas de características.

Pasos del Backpropagation en Capas Deconvolucionales:

1. Propagación hacia adelante:

- Las capas deconvolucionales transforman sus entradas mediante filtros, aumentando las dimensiones espaciales de los mapas de características.
- Similar a las capas convolucionales, estas transformaciones son seguidas por funciones de activación no lineales.

2. Retropropagación del error:

- *Gradiente de la Función de Pérdida:* Se calcula el gradiente de la función de pérdida con respecto a la salida de la capa deconvolucional.
- *Cálculo del Gradiente de los Filtros:* Se determina cómo el error se propaga a través de los filtros deconvolucionales. Esto implica calcular cómo cada elemento del filtro contribuye al error en la salida.
- *Gradientes con Respecto a la Entrada:* A diferencia de las capas convolucionales, donde se reduce la dimensión, aquí se calcula cómo el error se propaga a través de la capa para aumentar las dimensiones espaciales. Esencialmente, se realiza un proceso similar a la convolución, pero aplicado al gradiente.
- *Actualización de Filtros:* Los pesos de los filtros deconvolucionales se actualizan en la dirección opuesta a su gradiente, similar a como se hace en las capas convolucionales.

El entrenamiento de redes neuronales profundas implica una gran cantidad de cálculos matriciales. Para acelerar estos cálculos, se utilizan unidades de procesamiento gráfico (GPU) y unidades de procesamiento tensorial (TPU) que están diseñadas específicamente para realizar operaciones matriciales de manera eficiente.

Además, el despliegue de modelos de aprendizaje profundo en aplicaciones del mundo real requiere optimización y eficiencia computacional. Esto se logra mediante el uso de marcos y bibliotecas de software, como TensorFlow y PyTorch, que están diseñados para aprovechar al máximo el hardware especializado y permiten la implementación y el despliegue eficiente de modelos de aprendizaje profundo en diversas plataformas.

3.5. Optimización Bayesiana

La adaptación de modelos de aprendizaje automático, en particular redes neuronales convolucionales, a problemas específicos va mucho más allá de simplemente elegir una arquitectura y entrenarla en un conjunto de datos. A menudo, el desempeño óptimo de una CNN se encuentra en los detalles más sutiles: sus hiperparámetros. Estos hiperparámetros, que incluyen tasas de aprendizaje, tasa de abandono (*dropout*), regularización, entre otros, pueden tener un impacto significativo en el rendimiento del modelo. Sin embargo, la tarea de encontrar el conjunto óptimo de hiperparámetros no es trivial y actualmente existen diferentes algoritmos de optimización que nos ayudan a resolver este tipo de tareas.

Para el caso de la red neuronal expuesta en este trabajo utilizamos el algoritmo *Tree-structured Parzen Estimator* (TPESampler) que es un método de optimización bayesiana para la búsqueda de hiperparámetros. La idea principal detrás de la optimización bayesiana es modelar la función de pérdida mediante un modelo probabilístico. Formalmente, dado un dominio $X_d \subset \mathbb{R}$ para el d-ésimo hiperparámetro y $X = \prod_{i=1}^{D} X_i$, el problema de optimización bayesiana consiste en encontrar $x \in X$ tal que:

$$x \in \text{argmin}_{x \in X} f(x)$$

donde f es la función objetivo. En este sentido, la optimización bayesiana realiza una búsqueda de manera iterativa para x usando una función de adquisición que balancea el grado de exploración (búsqueda en regiones no vistas) y explotación (búsqueda de buenas observaciones cercanas). Una elección común para la función de adquisición es el *Expected Improvement* (EI) dado por:

$$EI_{y^\star}[x|\mathcal{D}] = \int_{-\infty}^{y^\star} (y^\star - y) p(y|x, \mathcal{D})\, dy$$

Otra elección es el *Probability Improvement* (PI) dado por:

$$\mathbb{P}(y \leq y^\star|x, \mathcal{D}) = \int_{-\infty}^{y^\star} p(y|x, \mathcal{D})\, dy$$

donde $y = f(x) + \varepsilon$ es una observación de la función objetivo con ruido ε, $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$ es un conjunto de observaciones.

El algoritmo TPE es una variante de los métodos bayesianos que estructura el espacio de búsqueda como un árbol. Supone que el espacio de búsqueda se puede dividir en dos grupos: el mejor grupo, que contiene configuraciones de hiperparámetros que conducen a mejores resultados, y el peor grupo, que contiene configuraciones que conducen a peores resultados.

La división se realiza en función de un umbral de cuantil superior, γ, que se ajusta en cada iteración del algoritmo.

El algoritmo TPE opera de la siguiente manera:

1. Se recopilan observaciones de la función objetivo $f(x)$ en el espacio de hiperparámetros utilizando configuraciones seleccionadas aleatoriamente.

2. Se dividen las observaciones en el mejor grupo (aquellas con mejores resultados) y el peor grupo (aquellas con peores resultados) en función del valor objetivo del cuantil superior, y^{γ}.

3. Se ajusta una distribución probabilística a las observaciones en cada uno de los dos grupos utilizando técnicas estadísticas, como la estimación por núcleos o modelos probabilísticos específicos como Gaussian Mixture Models (GMM).

4. Se genera una nueva configuración de hiperparámetros seleccionando aleatoriamente una de las dos distribuciones (mejor o peor) y luego muestreando de esa distribución.

5. Se evalúa la nueva configuración en la función objetivo y se agrega al conjunto de observaciones.

6. Se repiten los pasos 2-5 hasta que se alcance un número predeterminado de iteraciones.

El algoritmo TPE se beneficia de su capacidad para adaptarse a la exploración y explotación de manera eficiente. Al centrarse en las regiones prometedoras del espacio de hiperparámetros, puede converger rápidamente hacia soluciones de alto rendimiento. Además, el ajuste de γ en cada iteración permite una adaptación continua a la información recopilada hasta ese momento.

En el contexto de este trabajo, el algoritmo TPE se utiliza para ajustar los hiperparámetros de la red neuronal convolucional (CNN) que se emplea en la reconstrucción de obstáculos en el sistema de Stokes. La elección adecuada de hiperparámetros puede ser crucial para el rendimiento de la CNN, y TPE ofrece una forma sistemática de abordar esta optimización.

A medida que avanzamos en este trabajo, exploraremos cómo el uso de algoritmos de optimización bayesiana como TPE puede mejorar la capacidad de adaptación de modelos de aprendizaje profundo a problemas específicos y cómo influye en la precisión de la reconstrucción de obstáculos en el contexto del sistema de Stokes. Además, consideraremos aspectos computacionales y prácticos relacionados con la implementación de estos algoritmos en un entorno de aprendizaje automático.

Parte II

Desarrollo del Tema

Antes de comenzar en específico con cada uno de los desarrollos de esta tesis es importante comprender la estructura de un proyecto de ciencia de datos en este contexto. Esto pues existen muchos pasos a considerar que provienen del modelado clásico y otros que son más nuevos respecto a lo realizado típicamente. El esquema de la Figura 3.1 ilustra la forma en que abordamos el problema y destacaremos cada una de las fases

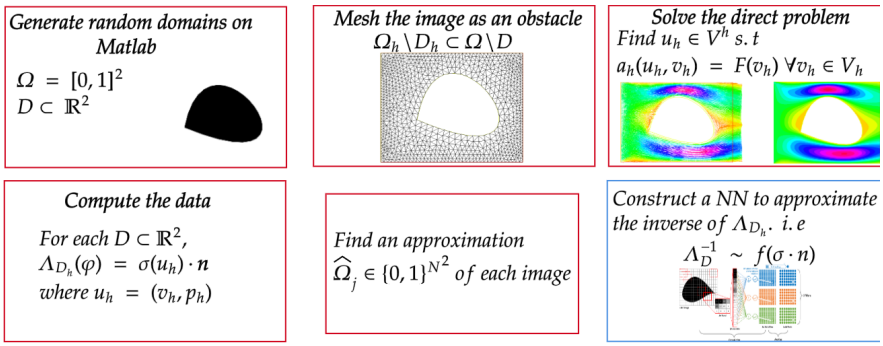


Figura 3.1: Esquema propuesto para ejecución de la investigación

1. **Generación de Dominios**: Dado que queremos aplicar algoritmos de Inteligencia Artificial aprovechamos la potencia que tienen las capas convolucionales, las cuales reciben como datos imágenes y por lo tanto nuestros obstáculos serán generados con un algoritmo y guardados como imágenes binarias utilizando MATLAB.

2. **Mallado de la Imagen como un Obstáculo**: Para calcular los datos se necesita crear un script capaz de procesar las imágenes y dejar las secciones en negro sin mallado, mientras que las secciones en blanco son malladas. Esto se realiza en FreeFem basados en el trabajo del lago Leman en [9] y generando así dominios $D_h \Subset \Omega_h$. Con Ω_h una discretización de $[0,1]^2$.

3. **Resolución del Problema Directo**: Utilizando el método de elementos finitos se resuelve el problema directo para cada una de las imágenes generadas, para esto se requiere hallar la formulación variacional del sistema de Stokes con obstáculos y con esto obtenemos (u_h, p_h) soluciones del sistema de stokes.

4. **Cálculo de los Datos**: Se aborda el cálculo de los datos, donde para cada $D \Subset \Omega$, se calcula $\Lambda_{D_h}(\varphi) = \sigma(u_h, p_h) \cdot n|_{\Gamma_1 \cup \Gamma_2}$.

5. **Aproximación de la imagen**: Se representa cada imagen del dominio $\Omega \setminus \overline{D}$ en una aproximación binaria $\hat{\Omega} \in \{0,1\}^{N^2}$.

6. **Construcción de una Red Neuronal**: Finalmente, se detalla la construcción de una red neuronal para aproximar la inversa de Λ_D, Se discute la arquitectura de la red, la selección de hiperparámetros, el entrenamiento y la validación de la red, y cómo se compara la salida del modelo con los datos reales.

El código de cada sección se puede encontrar en el repositorio de github asociado a la investigación.[1]

[1] https://github.com/Cristobal314/Deep-Learning-Methods-for-a-fluid-inverse-problem

4 Generación de datos

4.1. Obstáculos Aleatorios

Para la generación de datos de obstáculos se consideró un enfoque basado en [5] donde se realiza la aproximación de un funcional que representa la torsión para diferentes geometrías utilizando MATLAB. El algoritmo a utilizar tiene algunas modificaciones para satisfacer la hipótesis que el obstáculo debe estar contenido en el dominio sin tocar las paredes del cuadrado unitario ($D \Subset \Omega$). Para generar figuras arbitrarias se siguen los pasos:

Algorithm 2 Generación de Obstáculos mediante Puntos Interpolados

1: Escoger $r \in \mathbb{Z} \cap [3, 20]$ y $i \geq 1$.
2: Definir un umbral $\varepsilon > 0$.
3: Generar r puntos $p_1, p_2, \ldots, p_r$ tales que $p_i \sim U(0,1) \times U(0,1)$ y la distancia de cada p_i a $\partial\Omega$ es mayor que ε.
4: Interpolar los puntos $p_1, p_2, \ldots, p_r$ mediante splines para obtener la curva $\mathcal{C}$.
5: Definir $D = \{(x, y) \in \mathbb{R}^2 : (x, y)$ está encerrado por la curva $\mathcal{C}\}$

Esto se iteró para diferentes valores de r obteniendo cerca de 18.000 geometrías diferentes en donde todas satisfacen las restricciones del Teorema 2.5. Luego de esto, para generar figuras que tengan una frontera más suave vamos a considerar círculos y elipses, éstas figuras tienen la condición de tener una parametrización conocida que viene dada en el caso de un círculo por su radio r y su centro (h, k) con lo cual se define un círculo como

$$B((h,k),r) = \{(x,y) \in \mathbb{R}^2 : (x-h)^2 + (y-k)^2 \leq r^2\}$$

En el caso de una elipse se puede representar mediante su extensión en el eje x e y dadas por a, b respectivamente y su centro (h, k) con lo cual se define una elipse como

$$\mathcal{E}_{a,b}(h,k) = \left\{(x,y) \in \mathbb{R}^2 : \left(\frac{x-h}{a}\right)^2 + \left(\frac{y-k}{b}\right)^2 \leq 1\right\}$$

con esto en consideración el algoritmo para generar de forma aleatoria círculos viene dado por

Algorithm 3 Generación de Círculos Aleatorios

1: Generar $(h, k) \in \mathbb{R}^2$ tales que $(h, k) \sim U(0, 1) \times U(0, 1)$.
2: Calcular la proyección del centro en la frontera como:

$$\hat{h} = \min_{x \in \partial\Omega} \|x - h\|,$$
$$\hat{k} = \min_{x \in \partial\Omega} \|x - k\|.$$

3: Definir $r = \frac{1}{2} \min\{\hat{h}, \hat{k}\}$.
4: Guardar el círculo $B((h, k), r)$ asegurando que $B((h, k), r) \Subset [0, 1] \times [0, 1]$.

El caso de una elipse se trata de forma similar puesto que se debe asegurar que la extensión en cada eje no haga que la elipse toque la frontera, para esto el algoritmo que se sigue es el siguiente:

Algorithm 4 Generación de Elipses Aleatorias

1: Generar $(h, k) \in \mathbb{R}^2$ tales que $(h, k) \sim U(0, 1) \times U(0, 1)$.
2: Calcular la proyección del centro en la frontera como:

$$\hat{h} = \min_{x \in \partial\Omega} \|x - h\|,$$
$$\hat{k} = \min_{x \in \partial\Omega} \|x - k\|.$$

3: Definir los semiejes de la elipse como $a = \frac{1}{2}\hat{h}$ y $b = \frac{1}{2}\hat{k}$.
4: Guardar la elipse $\mathcal{E}_{a,b}(h, k)$ asegurando que $\mathcal{E}_{a,b}(h, k) \Subset [0, 1] \times [0, 1]$.

Siguiendo estos algoritmos se generaron 10.000 círculos y 10.000 elipses por lo que nuestro conjunto de datos de obstáculos queda determinado de la siguiente forma:

- 18.000 Polígonos con 3 lados hasta 6 lados.
- 10.000 elipses.
- 10.000 círculos.

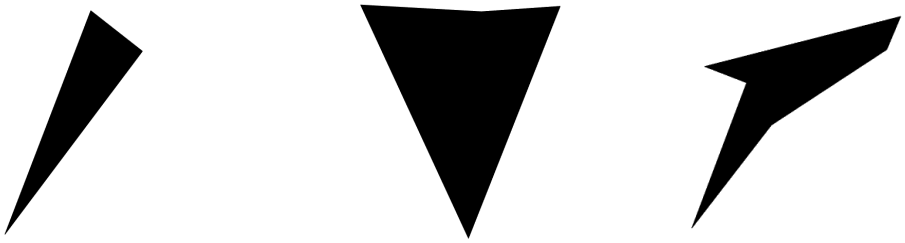
Estos obstáculos son representados mediante imágenes binarias que son una aproximación discreta de la indicatriz de cada conjunto, para esto consideremos $\Omega = [0, 1] \times [0, 1]$ y $D \Subset \Omega$

un obstáculo admisible. Definimos la función $f : \Omega \to \{0, 1\}$ que mapea cada píxel de la imagen a un valor binario de la siguiente manera

$$f(p) = \begin{cases} 1 & \text{si } p \in D \\ 0 & \text{si } p \notin D \end{cases}$$

Donde p es un píxel específico en el dominio Ω. Así, si la función f devuelve 1, indica que el píxel pertenece al interior del obstáculo, mientras que si devuelve 0, indica que el píxel está fuera del obstáculo. En la Figura 4.1 se pueden ver algunos obstáculos circulares de ejemplo mientras que en la Figura 4.2 se pueden observar 3 obstáculos aleatorios generados por el algoritmo.

Figura 4.1: Algunos obstáculos generados por el algoritmo de generación de círculos



(a) Obstáculo de 3 lados (b) Obstáculo de 4 lados (c) Obstáculo de 6 lados

Figura 4.2: Ejemplos de obstáculos generados por el algoritmo

4.2. Mediciones Sobre la Frontera

Cuando se tienen los obstáculos generados por MATLAB y guardados en una imagen binaria se procede a resolver el problema directo utilizando FreeFEM [9] donde las imágenes generadas en la sección anterior serán malladas como obstáculos dentro del cuadrado unitario para luego resolver una formulación débil del sistema de Stokes (2.16) con el método de elementos finitos (FEM) y calcular $\sigma \cdot n$ sobre Γ_1 y Γ_2. Estas mediciones son las salidas del operador Λ_D y por lo tanto podemos formular este elemento como el input de problema inverso mientras que el obstáculo será la salida del problema inverso.

4.2.1. Formulación débil del sistema de Stokes

A continuación, se deduce la formulación débil de (2.16). Consideremos $u = (u_1, u_2), v = (v_1, v_2)$ en V, donde V un espacio de Hilbert adecuado a determinar y $q \in W$. Si multiplicamos (2.16) por v e integrando en $\Omega \backslash \overline{D}$, obtenemos

$$\int_{\Omega \backslash \overline{D}} v \operatorname{div} \sigma(u,p)\, dx = 0$$

Para (u, p) suficientemente regulares, la relación $\operatorname{div} \sigma(u,p) = -\nabla p + \nu \Delta u$ es válida y por lo tanto, integrando por partes, se tiene

$$\int_{\Omega \backslash \overline{D}} \nu \nabla v : \nabla u - v \nabla p \, dx + \nu \int_{\partial(\Omega \backslash D)} v \nabla u \cdot \hat{n} \, ds = 0 \tag{4.1}$$

Desarrollando el término de borde, obtenemos

$$\int_{\partial(\Omega \backslash D)} v \nabla u \cdot \hat{n} \, ds = \int_{\Gamma_0} v \nabla u \cdot \hat{n} \, ds + \int_{\Gamma_1 \cup \Gamma_2 \cup \partial D} v \nabla u \cdot \hat{n} \, ds + \int_{\Gamma_N} v \nabla u \cdot \hat{n} \, ds \tag{4.2}$$

Luego, considerando $v = 0$ en $\partial D \cup \Gamma_1 \cup \Gamma_2$ y $v = \varphi$ en Γ_0, se tiene

$$\int_{\Omega \backslash \overline{D}} \nu \nabla v : \nabla u - v \nabla p \, dx + \nu \int_{\Gamma_N} v \nabla u \cdot \hat{n} \, ds + \nu \int_{\Gamma_0} \varphi \nabla u \cdot n \, ds = 0 \tag{4.3}$$

Notemos que

$$-\int_{\Omega \backslash \overline{D}} v \nabla p \, dx = \int_{\Omega \backslash \overline{D}} \nabla v : p\boldsymbol{I} \, dx - \int_{\Gamma_0} \varphi p \cdot \hat{n} \, ds - \int_{\Gamma_N} v p \cdot \hat{n} \, ds \tag{4.4}$$

Dado que $\nabla v : p\boldsymbol{I} = p \operatorname{div} v$, si consideramos que $\operatorname{div} v = 0$ en $\Omega \backslash \overline{D}$ y lo reemplazamos en (4.3), obtenemos que

$$\int_{\Omega\backslash\overline{D}} \nu \nabla v : \nabla u \, dx - \int_{\Gamma_0} \varphi p \cdot \hat{n} \, ds + \nu \int_{\Gamma_0} \varphi \nabla u \cdot n \, ds = 0 \tag{4.5}$$

Notar que no está presente el término de borde asociado a Γ_N dado que al combinar los resultados de (4.3) y (4.4) se elimina el término por la condición $\sigma(u,p) \cdot \hat{n} = 0$ en Γ_N. Además de esto, para considerar la presión dentro de la formulación variacional se puede considerar la condición de incompresibilidad

$$\operatorname{div} u = 0 \quad \text{en } \Omega \backslash D$$

Multiplicando por una función q e integrando se obtiene

$$\int_{\Omega\backslash D} q \operatorname{div} u \, dx = 0$$

Sumando este término a (4.5) se obtiene la formulación variacional del sistema en estudio:

Hallar $(u,p) \in H^1(\Omega\backslash\overline{D})^N \times L^2(\Omega\backslash\overline{D})$ con $\operatorname{div}(u) = 0$ tal que

$$\int_{\Omega\backslash\overline{D}} \nu \nabla v : \nabla u - q \operatorname{div} u \, dx - \int_{\Gamma_0} \varphi p \cdot \hat{n} \, ds + \nu \int_{\Gamma_0} \varphi \nabla u \cdot n \, ds = 0 \tag{4.6}$$

Para todo $(v,q) \in V \times W$, con

$$\begin{aligned} V &= \{v \in H^1(\Omega\backslash\overline{D})^N : \operatorname{div}(v) = 0, v = 0 \text{ sobre } \partial D \cup \Gamma, v = \varphi \text{ sobre } \Gamma_0\} \\ W &= L^2(\Omega\backslash\overline{D}) \end{aligned}$$

Donde $\Gamma = \Gamma_1 \cup \Gamma_2$.

El espacio V está definido para capturar la incompresibilidad y las condiciones de frontera de Dirichlet, mientras que W maneja la presión.
Con esto, a través del método de elementos finitos podemos determinar en FreeFEM++ los valores de $\sigma(u,p) \cdot n|_\Gamma$. En particular dado que se tratará un problema discreto dentro de FreeFEM++ se consideran 50 puntos en Γ_1 y 50 puntos en Γ_2, como a cada punto le corresponden dos componentes tendremos que la medición será un vector de 200 componentes y así formalizamos que queremos encontrar un mapeo de un vector en $\mathbb{R}^{200}$ en una matriz en $\mathcal{M}_{144\times144}$.

En específico, se consideran espacios de elementos finitos $V_h \subset V, W_h \subset W$ que tienen funciones base $(\phi_i)_{i=1}^{n_v}, (\theta_i)_{i=1}^{n_w}$ y con esto la formulación débil discreta vendrá dada por:
Hallar $(u_h, p_h) \in V_h \times W_h$ con $\operatorname{div}(u_h) = 0$ tal que

$$\int_{\Omega\setminus\overline{D}} \nu\nabla v_h : \nabla u_h - q_h \operatorname{div} u_h \, dx - \int_{\Gamma_0} \varphi_h p_h \cdot \hat{n} \, ds + \nu \int_{\Gamma_0} \varphi_h \nabla u_h \cdot n \, ds = 0 \quad (4.7)$$

Para todo $(v_h, q_h) \in V_h \times W_h$, con

$$\begin{aligned} V_h &= \operatorname{span}(\phi_i)_{i=1}^{n_v} \\ W_h &= \operatorname{span}(\theta_i)_{i=1}^{n_w} \end{aligned}$$

Continuando con la formulación débil discreta del sistema de Stokes, se procede al ensamblaje del sistema lineal. La solución aproximada u_h en V_h y p_h en W_h se expresan como combinaciones lineales de las funciones base:

$$u_h = \sum_{i=1}^{n_v} U_i \phi_i, \quad p_h = \sum_{i=1}^{n_w} P_i \theta_i,$$

donde U_i y P_i son los coeficientes a determinar. La formulación variacional discreta se convierte en un sistema lineal al reemplazar u_h y p_h en la ecuación integral y considerar cada función base en V_h y W_h como las funciones de prueba v_h y q_h, respectivamente.
El ensamblaje del sistema lineal implica calcular las siguientes matrices y vectores:

- **Matriz de Rigidez** A, donde cada elemento A_{ij} se calcula como

$$A_{ij} = \int_{\Omega\setminus\overline{D}} \nu\nabla\phi_i : \nabla\phi_j \, dx.$$

- **Matriz de Presión** B, donde cada elemento B_{ij} se calcula como

$$B_{ij} = \int_{\Omega\setminus\overline{D}} \theta_i \operatorname{div} \phi_j \, dx.$$

- **Vector de Carga** f, donde cada componente f_i se calcula considerando las condiciones de frontera y se define como

$$f_i = -\int_{\Gamma_0} \varphi_h \cdot (\theta_i \hat{n}) \, ds - \nu \int_{\Gamma_0} \varphi_h \cdot (\nabla\theta_i \cdot n) \, ds.$$

Con estas definiciones, el sistema lineal toma la forma

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} U \\ P \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}. \tag{4.8}$$

Para resolver este sistema lineal, se puede utilizar el **método de Crout**, que es una variante de la factorización LU. Este método descompone la matriz de coeficientes en dos matrices triangulares, L (triangular inferior) y U (triangular superior), donde U tiene unos en la diagonal. El sistema se resuelve en dos etapas: primero, se resuelve $Ly = f$ para y, y luego se resuelve $Ux = y$ para x. La factorización de Crout es particularmente útil cuando se trabaja con matrices grandes y dispersas, como es común en problemas de elementos finitos.

Es importante destacar que, para estabilizar correctamente el problema y asegurar la convergencia del método de elementos finitos, puede ser necesario añadir un término de estabilización a la formulación variacional. Esto se hace especialmente en el contexto de la incompresibilidad, donde la presión puede ser tratada de manera inadecuada. Un término común de estabilización es el término de penalización de la presión, que se añade al sistema lineal como sigue:

$$\varepsilon \int_{\Omega \setminus \overline{D}} p_h q_h \, dx,$$

donde ε es un parámetro de penalización pequeño. Este término ayuda a controlar las oscilaciones en la presión y mejora la estabilidad numérica del problema.

Una de las soluciones del problema discreto se pueden ver en la Figura 4.3 en donde se puede validar que el campo de velocidades rodea el obstáculo y logra seguir su camino por Γ_N. También, para observar la variedad de obstáculos que se pueden considerar en la Figura 4.4 se puede ver la velocidad y presión encontradas para el Sistema de Stokes en FreeFem donde nuevamente el campo de velocidades rodea el obstáculo para luego seguir su camino por Γ_N lo cual nos da una comprobación visual de que el modelado realizado para el problema directo respeta las leyes físicas impuestas en las condiciones de borde.

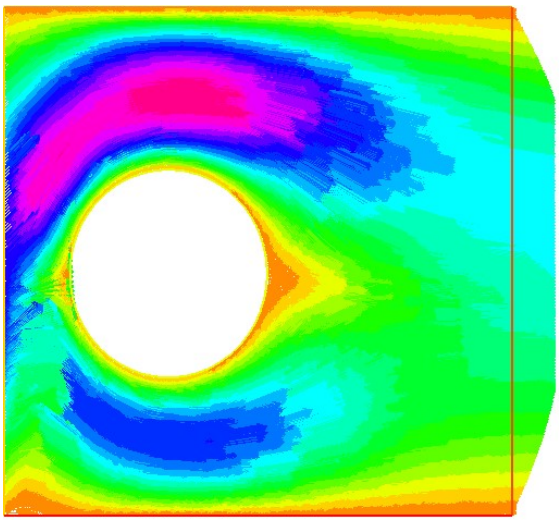
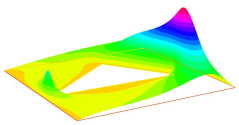
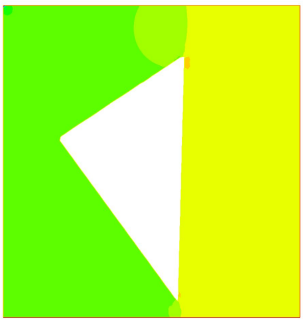


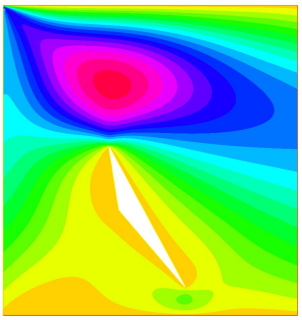
Figura 4.3: Campo de velocidades u asociado al sistema de Stokes en el caso de un obstáculo circular



(a) Vista superior de una componente de u con un obstáculo triangular.



(b) Presión p de un fluido conteniendo un triángulo.



(c) Perfil de velocidad de una componente de u para un obstáculo triangular.

Figura 4.4: Solución del sistema de Stokes utilizando FreeFEM

5 Enfoque basado en Deep Learning

5.1. Arquitectura Propuesta

La arquitectura propuesta se basa en una Convolutional Neural Network (CNN) que incluirá capas deconvolucionales y una conexión residual. La operación de deconvolución, denotada como Deconv, se utiliza para aumentar el tamaño de la entrada. Matemáticamente, la deconvolución toma una entrada $X \in \mathbb{R}^{N \times N}$ y la transforma en una salida $Y \in \mathbb{R}^{M \times M}$, donde $M > N$, aplicando un filtro W, donde W representa los pesos de la capa de deconvolución. Esta operación se realiza mediante la aplicación de filtros que se desplazan sobre la entrada y generan la salida ampliada.

Además de las capas deconvolucionales, la arquitectura incluirá capas intermedias de dropout y batch normalization. El dropout, es una técnica de regularización que se aplica a una capa con probabilidad p durante el entrenamiento.
Dado que el tamaño del dataset es grande y no permite la utilización directa de la memoria, el entrenamiento se llevará a cabo en una GPU NVIDIA A100 de forma distribuida por lotes (batches). Esto implica dividir el conjunto de datos en lotes más pequeños que pueden procesarse eficientemente en la GPU. Formalmente, si tenemos un conjunto de datos D con N ejemplos, se divide en lotes de tamaño B como $D = D_1, D_2, \ldots, D_{N/B}$, donde cada D_i contiene B ejemplos. Además, la arquitectura seguirá una estructura similar a la que se muestra en la descripción proporcionada. Esta estructura consta de capas lineales, funciones de activación ReLU, capas de dropout, capas deconvolucionales, capas de batch normalization y capas de salida. Una característica importante será la implementación de una skip connection, que permitirá retroalimentar la red con la entrada original antes de producir una salida. Se puede observar la arquitectura en la Figura 5.1 y en la Tabla 5.1. Es importante notar que la red neuronal construida tiene 11.4M parámetros lo cual representa una red de tamaño normal (e incluso pequeño) respecto a las actuales como Llama 2 (70B parámetros) y GPT3.5 (175B parámetros)

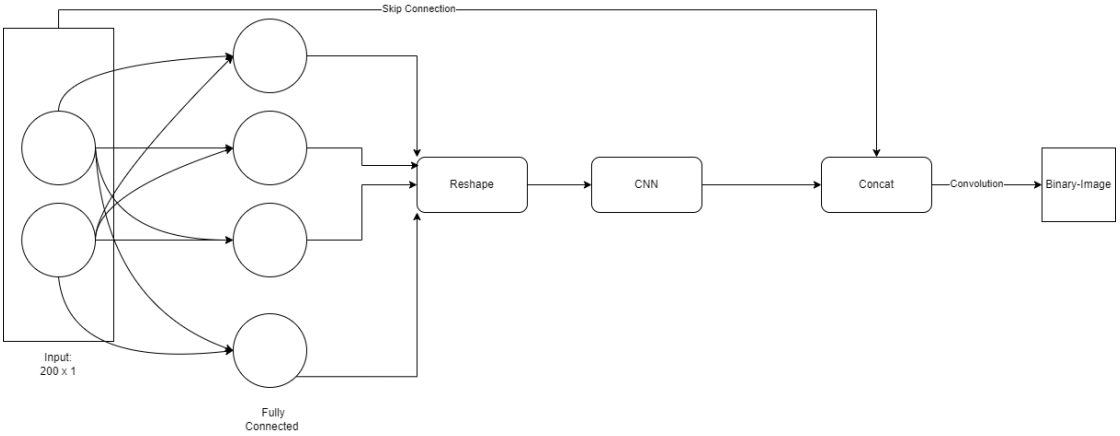


Figura 5.1: Arquitectura propuesta para la resolución del problema inverso. Los principales bloques será una red fully connected seguida de una red convolucional que será retroalimentada por una skip connection para obtener una imagen binaria representando el obstáculo.

Layer (type)	Output Shape	Param #
Linear-1	[-1, 20736]	4,167,936
ReLU-2	[-1, 20736]	0
Linear-3	[-1, 300]	60,300
ReLU-4	[-1, 300]	0
Dropout-5	[-1, 300]	0
Linear-6	[-1, 600]	180,600
ReLU-7	[-1, 600]	0
Dropout-8	[-1, 600]	0
Linear-9	[-1, 1200]	721,200
ReLU-10	[-1, 1200]	0
Dropout-11	[-1, 1200]	0
Linear-12	[-1, 5184]	6,225,984
ReLU-13	[-1, 5184]	0
Dropout-14	[-1, 5184]	0
Unflatten-15	[-1, 16, 18, 18]	0
ConvTranspose2d-16	[-1, 32, 36, 36]	4,640
BatchNorm2d-17	[-1, 32, 36, 36]	64
ReLU-18	[-1, 32, 36, 36]	0
Dropout-19	[-1, 32, 36, 36]	0
ConvTranspose2d-20	[-1, 32, 72, 72]	9,248
BatchNorm2d-21	[-1, 32, 72, 72]	64
ReLU-22	[-1, 32, 72, 72]	0
ConvTranspose2d-23	[-1, 64, 144, 144]	18,496
BatchNorm2d-24	[-1, 64, 144, 144]	128
ReLU-25	[-1, 64, 144, 144]	0
Dropout-26	[-1, 64, 144, 144]	0
ConvTranspose2d-27	[-1, 1, 144, 144]	1,601
Sigmoid-28	[-1, 1, 144, 144]	0
Conv2d-29	[-1, 1, 144, 144]	19
Sigmoid-30	[-1, 1, 144, 144]	0

Cuadro 5.1: Summary of the PyTorch Network Architecture

5.2. Entrenamiento

Para el entrenamiento de las redes neuronales se utilizaron diferentes funciones de pérdida, esto debido a que AE+CNN tendrá información sobre la codificación que debe realizar mientras que el enfoque basado netamente en CNN tendrá que comparar solo la similaridad entre los obstáculos. Además de esto, para obtener una resolución mayor se utilizará una regularización ℓ^2 dada por

$$R(\theta) = R((W,b)) = \|W\|_{\ell^2} = \sum_{i=1}^{L} |w_i|^2$$

Además de esto utilizaremos la entropía cruzada binaria (Binary Cross-Entropy Loss), esta es una función de pérdida comúnmente utilizada en la comparación de dos imágenes binarias en problemas de análisis de imágenes. Esta función de pérdida mide la discrepancia entre las distribuciones de píxeles de dos imágenes, evaluando cuán bien se ajusta la imagen predicha a la imagen de referencia. En el contexto de la comparación de imágenes binarias, la entropía cruzada binaria cuantifica la diferencia entre los valores de píxeles reales de la imagen de referencia y los valores predichos por la imagen generada. Es una función diferenciable que se puede utilizar como objetivo de optimización para mejorar la calidad de la imagen generada.

La fórmula de la entropía cruzada binaria para la comparación de imágenes binarias se expresa como:

$$BCELoss(I_{real}, I_{pred}) = -\frac{1}{N}\sum_{i=1}^{N} I_{real,i}\cdot\log(I_{pred,i}) + (1-I_{real})\cdot\log(1-I_{pred,i}) \quad (5.1)$$

Con esto la función de pérdida para CNN será la suma de el error entre las imágenes real y predicha con un regularizador ℓ^2:

$$L_{CNN}(x,y) = BCELoss(x,y) + \lambda R(\theta) = -\frac{1}{N}\sum_{i=1}^{N} x_i\cdot\log(y_i) + (1-x_i)\cdot\log(1-y_i) + \lambda\sum_{i=1}^{L}|w_i|^2 \quad (5.2)$$

Se realizaron 50 pruebas diferentes de learning rate, batch size y parámetro de regularización las cuales dieron como hiperparámetros óptimos los que se pueden ver en la Tabla 5.2, los cuales fueron estimados por el método de TPE mencionado en las secciones anteriores y utilizando 50 epochs para la evaluación del mejor modelo con una división de los datos en entrenamiento (80 %), Validación (10 %) y Test (10 %), éstos últimos serán usados en las secciones posteriores para analizar el modelo obtenido. Luego de esto, se procede a realizar un entre-

namiento profundo del modelo para hallar los parámetros a través de Backpropagation obteniendo una función de pérdida que evolucionó en el conjunto de entrenamiento de forma practicamente monótona con las iteraciones y que llegando a su epoch 349 nos entregó los mejores resultados ante lo cual se detiene el entrenamiento con la técnica de Early Stopping. Los resultados se pueden observar en los gráficos 5.2 y 5.3 generados a través de WandB. En particular el error del obstáculo está medido según el RMSE por lo que se puede concluir que el error promedio dado que el RMSE corresponde a un $1{,}88 \times 10^{-1}$ será aproximadamente $3{,}53 \times 10^{-2}$, los valores precisos de la función de pérdida durante el entrenamiento se pueden ver en la Tabla 5.3

Learning Rate	Batch Size	Regularization Parameter
10^{-3}	256	10^{-8}

Cuadro 5.2: Hiperparámetros estimados mediante TPE

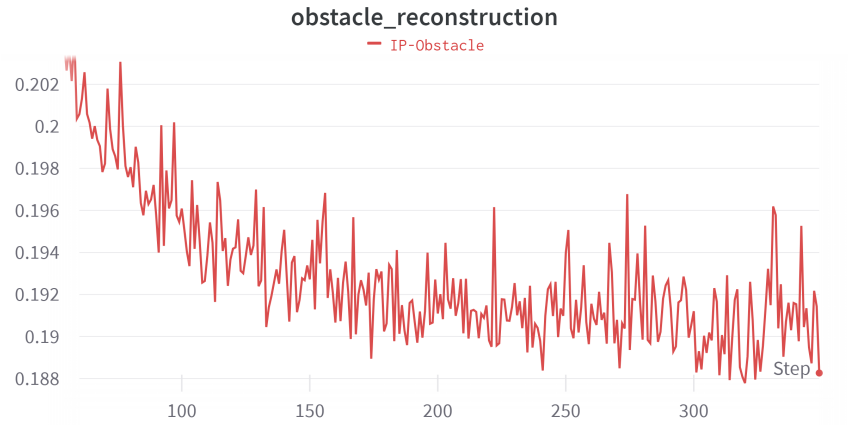


Figura 5.2: Evolución del error de reconstrucción del obstáculo medido por el RMSE

Train Loss	Validation Loss	Obstacle Reconstruction
$1{,}28 \times 10^{-1}$	$1{,}19 \times 10^{-1}$	$1{,}88 \times 10^{-1}$

Cuadro 5.3: Valores de función de pérdida en entrenamiento y validación

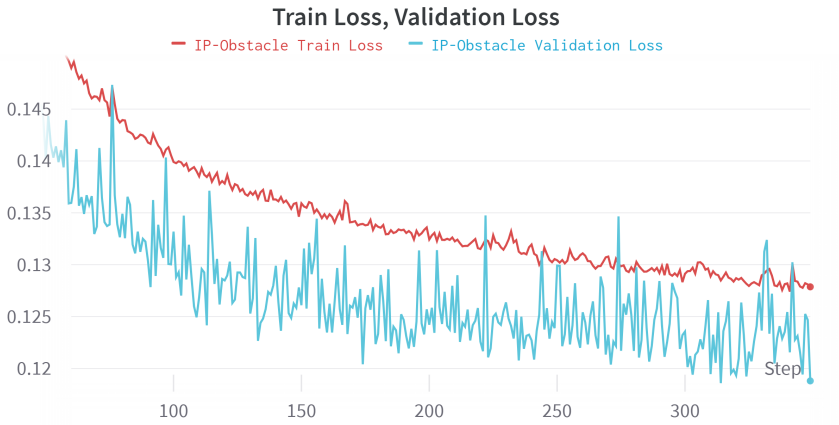


Figura 5.3: Evolución de la función de pérdida durante el entrenamiento, se puede observar un comportamiento más erróneo durante la etapa de validación dado que son geometrías no vistas por la red.

5.3. Validación Estadística

La validación del modelo es un paso crucial en el desarrollo y evaluación de cualquier sistema de aprendizaje automático. Nos permite entender la eficacia del modelo en datos no vistos y obtener intuiciones sobre dónde podría mejorar. En esta sección, presentaremos una validación exhaustiva del modelo propuesto utilizando una variedad de métricas de error. De ahora en adelante el análisis está basado exclusivamente en datos que no fueron vistos por la red neuronal durante el entrenamiento que provienen del conjunto de test. En las gráficas presentadas en la Figura 5.4, se ilustran los resultados de las reconstrucciones de objetos basadas en una muestra extraída del conjunto de prueba. Es fundamental señalar que las estructuras en color rojo representan los obstáculos reales, mientras que las tonalidades celestes denotan las aproximaciones realizadas por la red neuronal. Al analizar los datos, se evidencia una variabilidad en la precisión de la reconstrucción. Específicamente, la primera serie de imágenes muestra una tendencia ascendente del error, medido a través del MSE (Mean Square Error), a medida que la estructura del objeto real se simplifica. En contraste, la segunda serie, que presenta patrones más regulares, muestra errores considerablemente reducidos, sugiriendo una mayor eficiencia de la red al reconstruir dichos patrones. La tercera serie refleja estructuras más compactas y simples, con errores consistentemente bajos. Estos resultados reafirman la idea de que la complejidad y regularidad de la estructura del objeto influyen directamente en la capacidad de la red para aproximarse al obstáculo real. Estos hallazgos proporcionan valiosos insights sobre las fortalezas y áreas de mejora de la red en contextos específicos de reconstrucción.

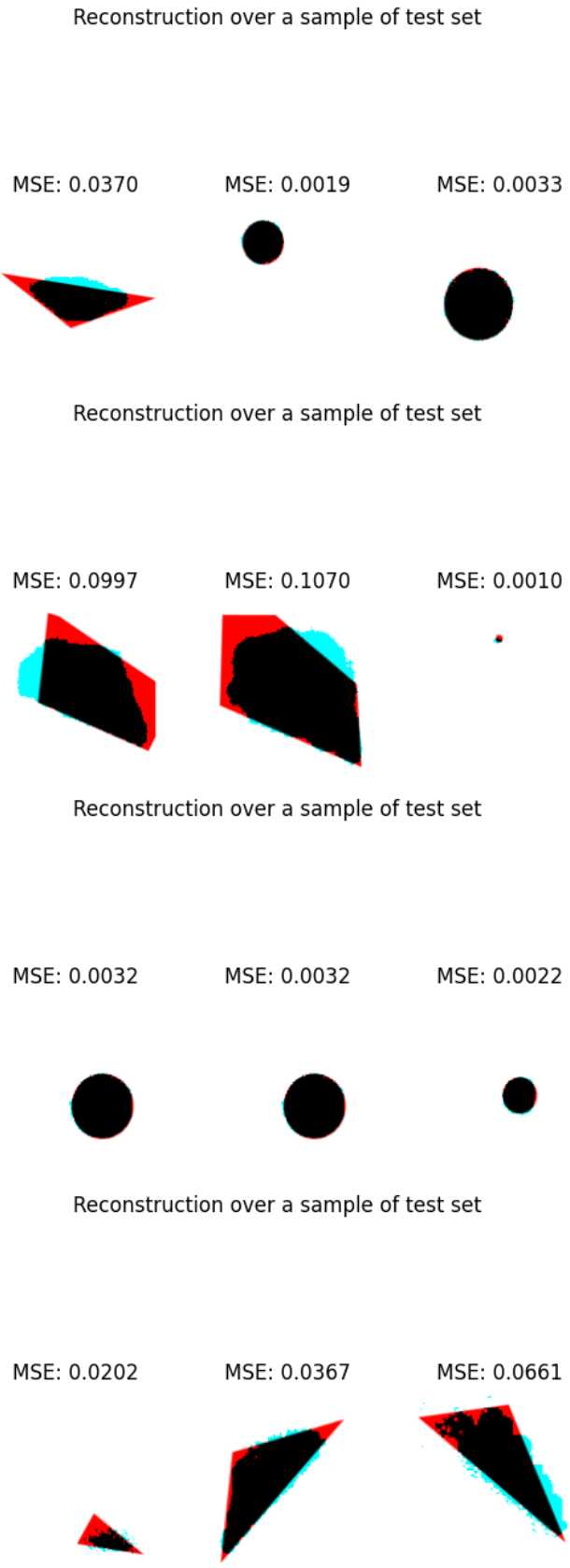


Figura 5.4: Reconstrucción sobre una muestra del conjunto de prueba.

5.3.1. Estudio del error

Al realizar un histograma con los errores medidos por el error cuadrático medio (MSE), en la Figura 5.5 se puede observar que la mayoría de las observaciones presentan un error menor a 0.05, lo que indica que el modelo, en general, está realizando predicciones bastante precisas para una amplia gama de datos. Esta concentración de errores bajos es un indicador positivo de la eficiencia del modelo, ya que implica que los errores son típicamente pequeños y, por lo tanto, las predicciones son cercanas al valor real.
Sin embargo, también se observa una cola decreciente que se extiende hacia valores de error más altos. Esto sugiere que hay ciertas observaciones para las cuales el modelo tiene dificultades en hacer predicciones precisas. Estas pueden ser situaciones atípicas o casos más complejos que desafían la capacidad predictiva del modelo.

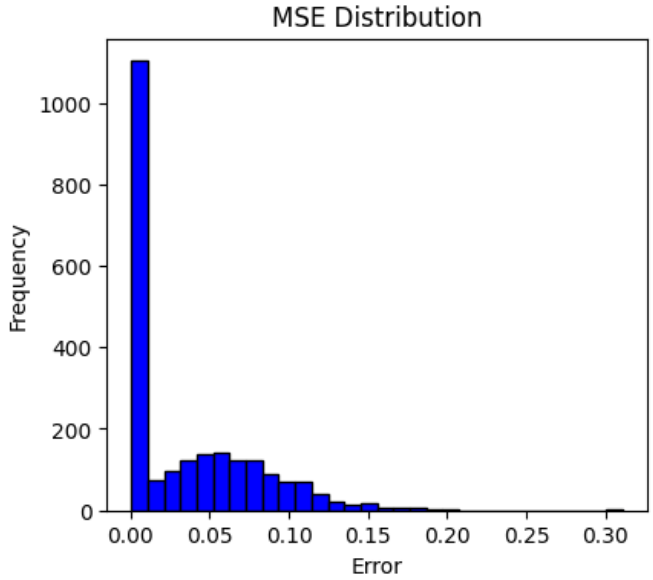


Figura 5.5: Distribución del error cuadrático medio

En este sentido, podemos estudiar las relaciones entre el área del obstáculo la cual puede ser aproximada a partir de la proporción de píxeles en negro con respecto al total de píxeles que en nuestro caso son $144 \times 144 = 20736$. Con esto podremos ver si existe algún rango de áreas para los cuales el modelo posee mayores complejidades en realizar la aproximación.
En la Figura 5.6, se realiza un gráfico del área del obstáculo y el error medido por el MSE para tal observación. Se observa que conforme el área del obstáculo se incrementa, el error también lo hace. Esta tendencia puede explicarse en parte porque las predicciones en áreas menores no son tan complejas. En estas zonas, gran parte de la imagen está dominada por

valores en blanco (representados por un 1 en la imagen binaria). Así, para minimizar el error, el modelo tiende a seleccionar la mayoría de los píxeles cercanos al valor 1. En contraste, las figuras con un área más amplia presentan mayor complejidad debido al incremento en valores negros (simbolizados con un 0 en la imagen). Sin embargo, para áreas superiores a 0.2, la correlación no es tan evidente y parece haber una mayor dispersión en los valores del error.

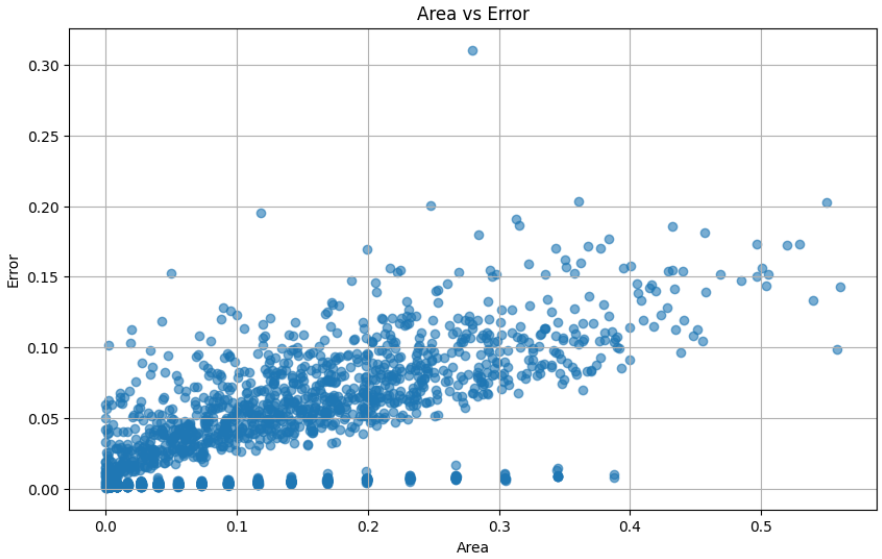


Figura 5.6: Scatter plot Area vs Error

Más aún, se puede observar que para áreas menores a 0.4 hay un grupo de observaciones que tienen errores cercanos a cero, mientras que la observación con error 0.30 corresponde más a un outlier que a algo representativo en la explicación de la capacidad predictiva del modelo. Si medimos la correlación de Pearson no alcanzamos a determinar cierta linealidad entre el mapeo área - error pues posee un coeficiente de 0.747 lo cual sugiere que ante la eliminación de outliers si podríamos establecer cierta linealidad y una relación más monótona entre área y error.

Estas observaciones sugieren una oportunidad de mejora en la metodología actual. En lugar de centrarnos en una métrica que intenta aproximar la imagen pixel por pixel, podría ser más beneficioso dirigir la función de pérdida hacia la localización precisa del obstáculo. Esto implica que la optimización no se enfocaría tanto en la reproducción exacta de cada pixel, sino en determinar con precisión dónde se encuentra el obstáculo dentro de la imagen. Una adaptación en esta dirección podría ofrecer resultados más robustos, especialmente en escenarios con obstáculos de mayor área. A pesar que las redes convolucionales se inspiran en el campo receptivo humano para establecer parches y una estructura más local en los datos, se podría ver una oportunidad de mejora en el hecho de mejorar la función de pérdida y por ejemplo

incluir que la imagen debe ser conexa o poseer otras propiedades geométricas más que solo ser similar a ciertas zonas de pixeles.

El gráfico de la Figura 5.7 sugiere cómo se distribuye el error basado en el tamaño de los obstáculos (medidos en área de píxeles), a través de violin plots los cuales grafican la distribución en conjunto con su respectivo boxplot. Se observa que, mientras que la red exhibe una eficacia considerable, evidenciada por una Entropía Cruzada Binaria de 0.12, presenta ciertas peculiaridades en su comportamiento. Específicamente, el error tiende a ser más bajo para áreas de obstáculos menores, incrementando ligeramente para tamaños intermedios, donde también se observa una mayor varianza en la predicción. Esta variabilidad sugiere que la red podría enfrentar desafíos al predecir obstáculos de tamaño medio con consistencia. A pesar de la robustez general del modelo, estos hallazgos indican áreas potenciales de mejora. Una exploración más profunda de las causas subyacentes de estas inconsistencias, junto con experimentos adicionales en técnicas avanzadas de modelado, podría mejorar significativamente el rendimiento del modelo en dominios problemáticos.

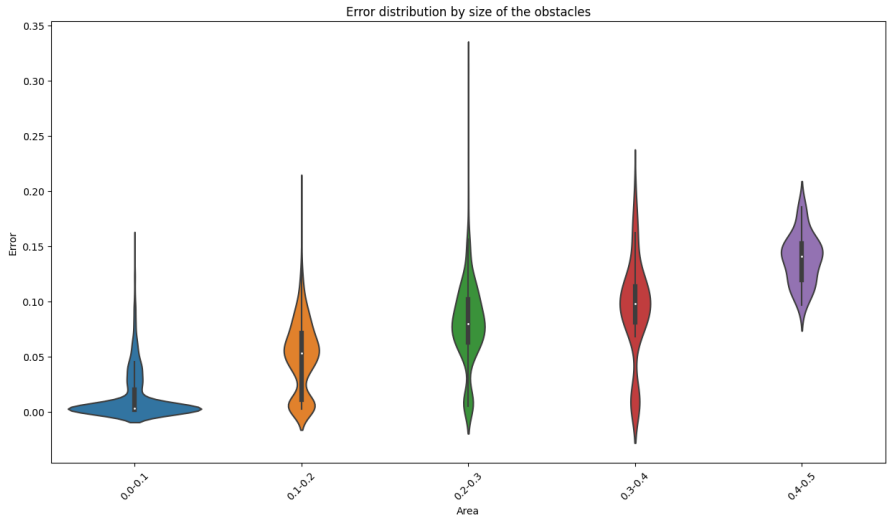


Figura 5.7: Violin Plot del error en función del tamaño del obstáculo

5.3.2. Robustez Ante el Ruido

Una gran capacidad de las redes neuronales artificiales es su capacidad de generalización. Esto significa que, una vez entrenado, el modelo puede realizar predicciones precisas en datos que no ha visto antes. Sin embargo, esta capacidad de generalización puede verse afectada por la presencia de ruido en los datos. En esta sección, analizamos cómo se comporta el modelo cuando se agregan diferentes niveles de ruido a las mediciones.

Para este análisis, se presentaron mediciones con diferentes niveles de ruido. Para cada medición y, se evaluó la precisión del modelo, comparando la forma original con la predicción del modelo. La adición de ruido ε generó un dato y_ε corrupto que viene dado por

$$y_\varepsilon = y + \nu \cdot \varepsilon \cdot \|y\|_2, \quad \varepsilon \sim \mathcal{N}(0,1)$$

El parámetro que se irá variando será ν, los resultados generados se pueden observar en las Figuras 5.8 y 5.9 donde se puede ver que para valores de ν lo suficientemente pequeños no existe una variabilidad muy alta entre las predicciones del modelo, mientras que para valores más grandes de ν (mayores a 0.05) existe mucha variabilidad en las predicciones.

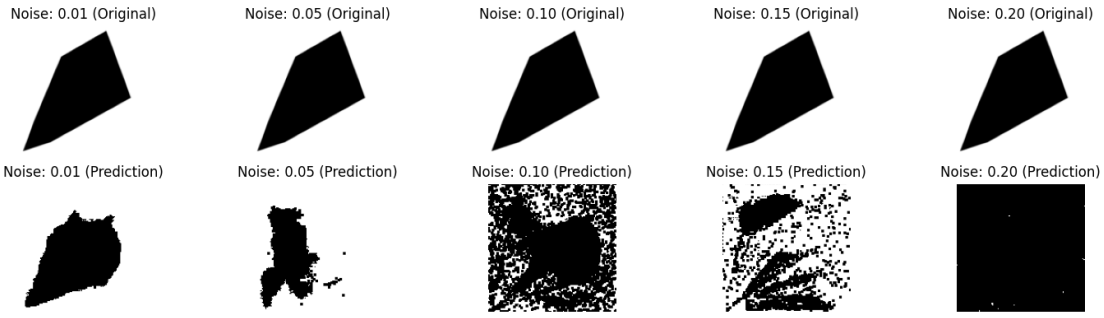


Figura 5.8: Comparación visual de la forma original y la predicción del modelo ante diferentes niveles de ruido ν.

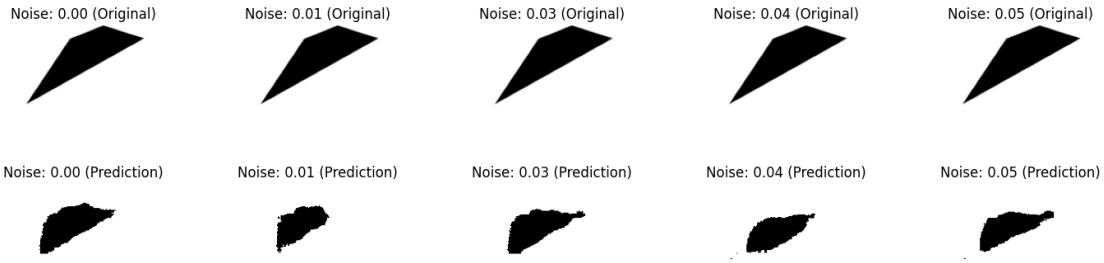


Figura 5.9: Comparación visual de la forma original y la predicción del modelo ante diferentes niveles de ruido ν.

Esta comparación visual la validamos con el Scatter y Violin plot realizado en las Figuras 5.10 y 5.11. En éstas figuras podemos ver que si ν es 0.01, las distribuciones del error no cambian con respecto a los de la Figura 5.7, mientras que para ruidos mayores la variabilidad aumenta. Los resultados subrayan la sensibilidad del modelo a perturbaciones en la entrada. Esta variabilidad en las predicciones sugiere que se deben explorar técnicas de preprocesamiento

o considerar arquitecturas de red más robustas que puedan manejar eficientemente el ruido en las imágenes, un ejemplo de ello sería generar más datos a través de la adición de ruido y entrenar a la red con ruido para ganar robustez.

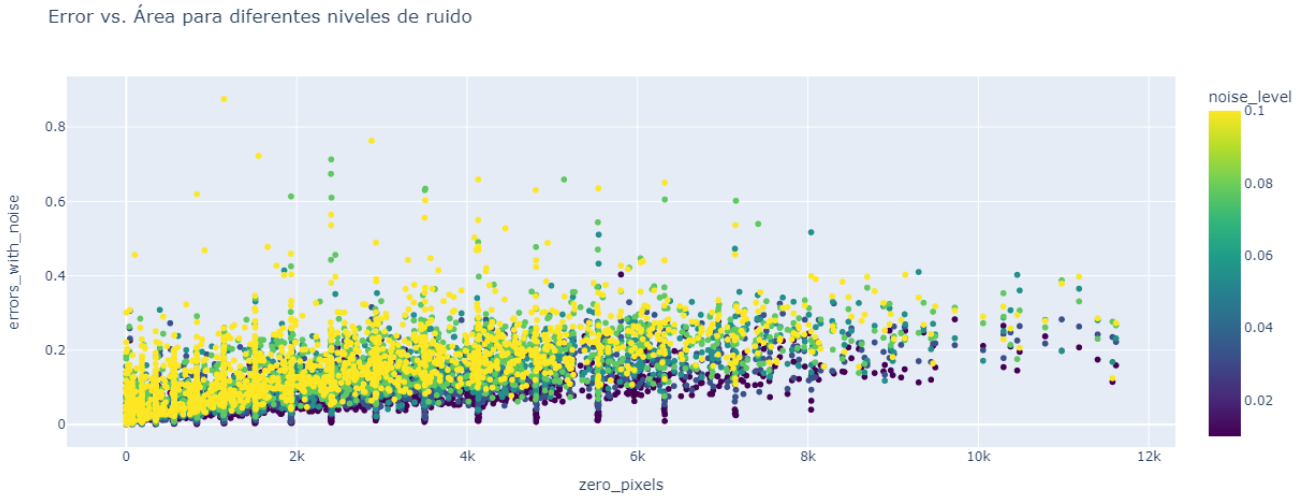


Figura 5.10: Scatter Plot del área vs Error para diferentes niveles de ruido.

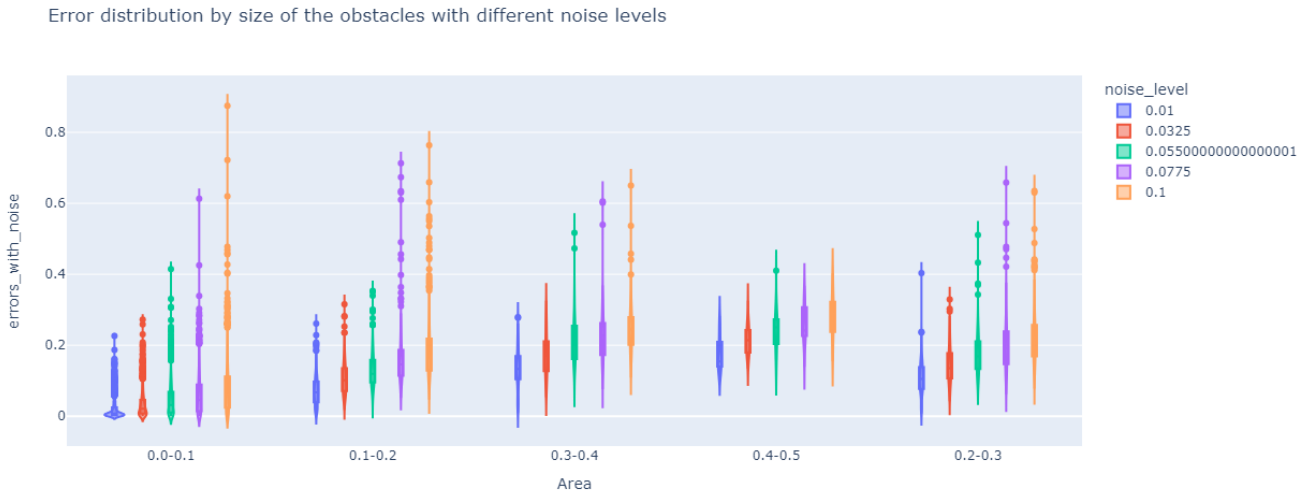


Figura 5.11: Violin Plot del área y error para diferentes niveles de ruido.

Parte III

Conclusiones

6 Limitaciones del enfoque

A pesar de los avances significativos presentados en esta tesis, es crucial reconocer las limitaciones inherentes al enfoque adoptado. Estas limitaciones no solo delinean el alcance actual de nuestra metodología, sino que también sugieren áreas de mejora y consideraciones para estudios futuros.

1. **Resolución Uniforme de Obstáculos**: La construcción de los modelos de aprendizaje profundo depende de una resolución uniforme de 144×144 para todos los obstáculos. Esta restricción de resolución podría limitar la capacidad de los modelos para capturar y aprender de las variaciones finas en los contornos de los obstáculos, especialmente cuando se trata de geometrías con características a escalas menores que las de la resolución fijada. Además, la necesidad de una resolución uniforme impone restricciones en la selección del conjunto de datos y en la representación de los obstáculos en el dominio de estudio.

2. **Identificación de Fronteras en Obstáculos Irregulares**: La metodología actual no garantiza una identificación precisa de las fronteras en obstáculos con irregularidades significativas. Mientras que los modelos de aprendizaje profundo han demostrado ser hábiles en la aproximación de las características generales de los obstáculos, la identificación exacta de fronteras irregulares o discontinuas sigue siendo un desafío. Esta limitación se evidencia especialmente en la reconstrucción de los detalles finos de la frontera, donde la red puede perder información crítica sobre la forma exacta del obstáculo

3. **Ausencia de Garantía Teórica**: No existe, hasta la fecha, un resultado teórico que garantice la identificación de obstáculos a través del uso de redes neuronales. La naturaleza empírica del aprendizaje profundo significa que, aunque los resultados prácticos pueden ser prometedores, la falta de una base teórica sólida puede plantear incertidumbres sobre la fiabilidad y generalización de los modelos. Este desafío es particularmente significativo en el campo de los problemas inversos, donde la complejidad del problema hace que la demostración de tales garantías sea una tarea formidable.

   Estas limitaciones resaltan la necesidad de continuar investigando y desarrollando el campo del aprendizaje profundo aplicado a problemas inversos en la mecánica de flui-

dos. Es imperativo explorar métodos que permitan superar las restricciones de resolución, mejorar la precisión en la identificación de fronteras y establecer fundamentos teóricos que respalden el uso de redes neuronales en este contexto.

6.1. Trabajo Futuro

Una dirección prometedora para futuras investigaciones en la reconstrucción de obstáculos inmersos en fluidos es la integración de principios físicos dentro de los marcos de aprendizaje profundo, un enfoque conocido como Physics Informed Machine Learning. Esta metodología capitaliza en la incorporación de leyes físicas conocidas, como las ecuaciones de conservación y las dinámicas de fluidos, directamente en la arquitectura de aprendizaje automático. Al informar a los modelos de aprendizaje profundo con estas leyes fundamentales, se espera mejorar su capacidad para generalizar a partir de datos limitados y mejorar su interpretabilidad. Physics Informed Machine Learning puede ser particularmente útil en situaciones donde los datos son escasos, costosos de obtener o ruidosos, condiciones comunes en aplicaciones de ingeniería y física. En el contexto de la mecánica de fluidos, los modelos de aprendizaje pueden ser informados con las ecuaciones de Navier-Stokes para garantizar que las predicciones respeten las propiedades de incompresibilidad y conservación de momento. Este enfoque tiene el potencial de mejorar significativamente la calidad de las reconstrucciones geométricas de obstáculos, facilitando la identificación de geometrías complejas y ofreciendo una mejor comprensión de los flujos de fluidos alrededor de ellos. La exploración de Physics Informed Machine Learning en futuros trabajos también debería centrarse en el desarrollo de algoritmos que puedan manejar la incertidumbre y proporcionar estimaciones de confianza en sus predicciones. Al combinar técnicas de Physics Informed Machine Learning con metodologías de cuantificación de incertidumbre, como el Bayesian Deep Learning, se puede proporcionar una medida de la confiabilidad de las predicciones del modelo, lo que es de suma importancia en aplicaciones críticas de ingeniería. A pesar de los avances significativos presentados en esta tesis, es crucial reconocer las limitaciones inherentes al enfoque adoptado. Estas limitaciones no solo delinean el alcance actual de nuestra metodología, sino que también sugieren áreas de mejora y consideraciones para estudios futuros.

7 Conclusión

En la culminación de este estudio, nos encontramos con hallazgos que no solo validan la aplicación de metodologías de aprendizaje profundo a problemas inversos en la mecánica de fluidos, sino que también amplían las capacidades conocidas de tales técnicas en la detección y reconstrucción de geometrías complejas. La investigación ha demostrado con éxito que las Redes Neuronales Convolucionales (CNN) pueden ser entrenadas para invertir el operador de Poincaré-Steklov, lo que facilita la identificación precisa de la geometría de los obstáculos inmersos en un fluido, incluso aquellos caracterizados por tener fronteras no suaves o irregulares.

Esta capacidad de tratar con discontinuidades y geometrías irregulares es particularmente notable, ya que tales características presentan desafíos significativos para los métodos convencionales. El aprendizaje profundo, como se ha implementado en este trabajo, demuestra una notable robustez frente a la complejidad geométrica, lo que sugiere un amplio campo de aplicaciones potenciales donde las irregularidades son la norma y no la excepción.

Más allá de las técnicas de generación y procesamiento de datos de alta resolución utilizadas para el entrenamiento del modelo, la contribución de este estudio radica en la demostración empírica de que es posible superar las limitaciones tradicionales asociadas con la recuperación de obstáculos. Aunque las redes neuronales requieren un volumen significativo de datos para un entrenamiento efectivo y su interpretación puede ser intrincadamente compleja, los resultados obtenidos indican que estos desafíos son superables y, de hecho, pueden ser atenuados a través de enfoques innovadores en la ciencia de datos.

Para el futuro, se anticipa que la extensión del conjunto de datos, junto con la optimización de la arquitectura de la red y los algoritmos de entrenamiento, aumentará la generalización y la precisión del modelo. Se reconoce la necesidad de estrategias más sofisticadas que permitan una interpretación más clara de los modelos generados por el aprendizaje profundo y que ofrezcan una cuantificación fiable de la incertidumbre de las predicciones.

Este trabajo no solo proporciona una contribución significativa al cuerpo de conocimiento en la mecánica de fluidos computacional y la ciencia de datos, sino que también establece un precedente para la exploración futura y la aplicación del aprendizaje profundo en la resolución de problemas inversos de Ecuaciones Diferenciales Parciales.

Parte IV

Apéndice

8 Propiedades de operadores diferenciales

Sea A un campo vectorial, f un campo escalar. Se tiene lo siguiente

$$\operatorname{div}(fA) = f(\operatorname{div} A) + (A \cdot \nabla)f \tag{8.1}$$

Definición 8.1. Sea $\Omega \subset \mathbb{R}^N$, v un campo vectorial regular en Ω entonces su gradiente se define como el campo

$$\nabla v = \left(\frac{\partial v_i}{\partial x_j}\right)_{1\leq i,j\leq N}$$

Definición 8.2. Sea $\Omega \subset \mathbb{R}^N$, $\sigma = (\sigma_{ij})_{1\leq i,j\leq N}$ un campo tensorial regular en Ω. Llamamos divergencia de σ al campo vectorial

$$\operatorname{div}\sigma = \left(\sum_{j=1}^{3} \frac{\partial \sigma_{ij}}{\partial x_j}\right)_{1\leq i\leq N}$$

Definición 8.3. (Producto Tensorial) Sean u, v dos campos vectoriales, definimos el producto tensorial como el tensor de segundo orden

$$u \otimes v = (u_i v_j)_{1\leq i,j\leq N}$$

De esto tenemos la siguiente fórmula para la divergencia de un producto tensorial

$$\operatorname{div}(u \otimes v) = (\operatorname{div} v)u + (v \cdot \nabla)u \tag{8.2}$$

Bibliografía

1. C. Alvarez, C. Conca, L. Friz, O. Kavian y J. Ortega. «Identification of immersed obstacles via boundary measurements». *Inverse Problems* 21/5, 2005, págs. 1531-1552.
2. C. Alvarez, C. Conca, R. Lecaros y J. Ortega. «On the identification of a rigid body immersed in a fluid: A numerical approach». *Engineering Analysis with Boundary Elements* 32, 2008, págs. 919-925.
3. L. Bourgeois y J. Dardé. «The exterior approach to solve the inverse obstacle problem for the Stokes system». *Inverse Problems & Imaging* 8, 2014, págs. 23-51.
4. F. Boyer y P. Fabrie. *Mathematical Tools for the Study of the Incompressible Navier-Stokes Equations and Related Models*. Vol. 183. 2012. ISBN: 978-1-4614-5974-3. DOI: 10.1007/978-1-4614-5975-0.
5. F. Calabró, S. Cuomo, F. Giampaolo, S. Izzo, C. Nitsch, F. Piccialli y C. Trombetti. «Deep Learning for the approximation of a shape functional». *arXiv*, 2021. URL: https://arxiv.org/abs/2110.02112.
6. A. Calderón. «On an inverse boundary value problem». *Seminar on Numerical Analysis and its Applications to Continuum Physics*, 1980, págs. 65-73.
7. G. Cybenko. «Approximation by superposition of a sigmoidal function». *Mathematics of Control, Signals and Systems*, 1989.
8. C. Fabre y G. Lebeaum. «Prolongement unique des solutions de Stokes». *Commun. Partial Differ. Eq.* 21, 1996, págs. 573-596.
9. F. Hecht. «New development in FreeFem++». *Journal of Numerical Mathematics* 20:3-4, 2012, págs. 251-266.
10. V. Isakov. *Inverse Problems of Partial Differential Equations*. Vol. 127. Springer, 2006.
11. C.-L. Lin, G. Uhlman y J.-N. Wang. «Optimal three–ball inequalities and quantitative uniqueness for the Stokes system». *Discrete Contin. Dyn. Syst.* 28/3, 2010, págs. 1273-1290.
12. M. D. Zeiler, D. Krishnan, G. W. Taylor y R. Fergus. «Deconvolutional networks». En: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010, págs. 2528-2535. DOI: 10.1109/CVPR.2010.5539957.