Universidad Michoacana de San Nicolás de Hidalgo

Instituto de Física y Matemáticas

ESTUDIO DE SIMULACIONES FÍSICAS USANDO MÉTODOS DE APRENDIZAJE AUTOMÁTICO

Tesis

que para obtener el título de
DR. EN CIENCIAS EN EL ÁREA DE FÍSICA

presenta
CARLOS EDUARDO LÓPEZ NÚÑEZ

Asesores
JOSÉ ANTONIO GONZÁLEZ CERVERA

ALFREDO RAYA MONTAÑO

Morelia, Michoacán, México Mayo del 2019

DEDICATORIA

A mi esposa Krystal, mi compañera de vida y quien me apoya en todo momento. Gracias por compartir tu vida conmigo.

Agradecimientos

Quiero agradecer a mis asesores de tesis, quienes siempre me ayudaron a resolver las dudas que surgieron durante el desarrollo de esta tesis, así como los consejos que me dieron. También agradezco a mis revisores de tesis, por otorgarme algo de su tiempo para leer y realizar sugerencias para mejorar el trabajo presentado. Por último, pero no menos importante, doy las gracias a mi familia, amigos y compañeros del instituto, por las convivencias y los buenos momentos que pasamos en el transcurso de mis estudios de doctorado.

Índice general

Resumen

En esta tesis se estudian simulaciones numéricas asociadas a diversos sistemas físicos haciendo uso de varios métodos de aprendizaje automático, específicamente, regresores lineales, redes neuronales artificiales y máquinas de soporte vectorial.

Los sistemas físicos que se estudian son los siguientes: Oscilaciones de Bloch en una cadena lineal, en una red cuadrada y en grafeno sujeto a distintas condiciones iniciales, brotes de rayos gama, obstrucciones dentro de un tubo por el que fluye un fluido y curvas de Bragg.
De las simulaciones correspondientes a cada sistema se extrae información que sirve para entrenar a los métodos de aprendizaje automático y con los cuales se estiman ciertos parámetros físicos, que son, la intensidad de un campo eléctrico, la separación interatómica, la densidad de un gas, el tamaño y posición de un objeto, la energía del sistema, en donde tales parámetros estimados dependen del sistema de estudio.

Adicionalmente, se explora el desempeño de un método de clasificación de patrones, al que se denominó como clasificación utilizando salidas multiclase.

Palabras clave: Regresores; Redes neuronales artificiales; Máquinas de soporte vectorial; Clasificación de patrones.

Abstract

In this thesis numerical simulations associated to several physical systems were studied, using different machine learning methods, specifically, linear regressors, artificial neural networks and support vector machines.

The physical systems studied include: Bloch oscillations in a linear chain, in a squared lattice and in graphene subject to different initial conditions, gamma ray bursts, obstructions inside a pipe in which a fluid is flowing and Bragg peaks.
From the simulations corresponding to each system, information that serves to train the machine learning methods is extracted and with which physical parameters are estimated, that are, an electric field intensity, the interatomic separation, the density of a gas, the size and location of an object, the system's energy, where the estimated parameters depend on the system under consideration.

Additionally the performance of a classification method, named as classification using multiclass outputs, was explored.

Capítulo 1

Introducción

Los avances en el campo de inteligencia artificial, particularmente los desarrollados haciendo uso de métodos de aprendizaje automático (*machine learning* en inglés), se han popularizado últimamente a tal grado que convivimos con ellos prácticamente todos los días. Los sistemas de sugerencias como los que usan los servicios de *streaming*, ya sea de música o de películas, los reconocedores faciales y de voz, los sistemas para pronósticos del clima, la conducción de vehículos autónomos, todos ellos tienen en común que utilizan métodos de aprendizaje automático para realizar sus tareas. Tal es el apogeo de estos métodos que, por ejemplo, los celulares más recientes traen procesadores fabricados para hacer uso de estas herramientas de manera eficiente.

Y así como estos métodos están presentes en nuestra vida cotidiana, también lo están dentro de las ciencias, en particular, en la Física. El objetivo principal de esta tesis es mostrar que los métodos de aprendizaje automático (AA) se pueden aplicar al estudio de distintos sistemas físicos, que van desde el estudio de fenómenos presentes en nuevos materiales bidimensionales como lo es el grafeno, hasta incluso aplicaciones industriales como es la detección de obstrucciones en conductos que transportan fluido. El problema a resolver consiste entonces, en estimar ciertas propiedades físicas al entrenar los métodos de AA con información obtenida a través de simulaciones numéricas, las cuales representan al sistema físico en cuestión. La razón de usar simulaciones numéricas, es que podemos idealizar dife-

rentes escenarios físicos, donde tales escenarios pueden ser muy difíciles de reproducir en un laboratorio, como por ejemplo, un brote de rayos gamma. También podría suceder, que la reproducción del escenario físico de interés sea muy costosa o incluso perjudicial para la salud, como el caso de la terapia de hadrones para tratamiento de pacientes con cáncer. En tal situación, si se trabaja directamente con los pacientes, se corre el riesgo de causar lesiones en tejidos del cuerpo o en órganos no deseados, comprometiendo al paciente. Por estos motivos resulta conveniente trabajar con simulaciones numéricas, donde por ejemplo, los parámetros estimados por los métodos de AA, podrían servir para calibrar adecuadamente el instrumento que genera el haz de partículas con el que se radia al paciente enfermo de cáncer y así evitar lesiones.

Los sistemas físicos que presentamos primero, corresponden a sólidos con estructuras cristalinas, donde los parámetros que se estiman son: la intensidad de un campo eléctrico, la distancia interatómica entre los elementos del cristal, el cociente de Poisson y la variación de la energía de *hopping*. Estos parámetros se encuentran presentes al estudiar oscilaciones de Bloch, fenómeno que se describe más adelante.
Posteriormente, al estudiar brotes de rayos gamma, estimamos la densidad del gas, el factor de Lorentz y la densidad de energía radiada, que corresponden a las cantidades físicas mas relevantes en estos sistemas, bajo las suposiciones que consideramos.
Al estudiar obstrucciones de flujo en tuberías, los métodos de AA predicen la morfología, tamaño y posición de la obstrucción.
Por último, analizamos curvas de Bragg, estimando la energía inicial de un haz de partículas para diferentes modelos considerados.

Se optó por estos métodos de AA para el análisis de los sistemas antes mencionados, debido a que tienen la propiedad de analizar una gran cantidad de datos y es lo que se pretende a largo plazo: poder analizar una gran cantidad de datos, asociados a sistemas físicos de interés. Por esta razón, se fue aumentando el número de simulaciones y la cantidad de datos procesados por los métodos en los diferentes sistemas expuestos. Comenzamos utilizando 100 simulaciones diferentes para entrenar los métodos de AA y terminamos usando hasta un total de 10000 simulaciones.

Actualmente uno de los temas de estudio que tiene apogeo, es el procesamiento de señales provenientes de ondas gravitacionales, debido a su reciente descubrimiento. La cantidad de datos almacenados de tales eventos, pueden llegar a ser de tal magnitud, que las herramientas de AA antes mencionadas son de gran utilidad en estos escenarios.
Se han utilizado los métodos de AA para caracterizar y estimar varias propiedades de sistemas físicos, los cuales pueden tener un índole teórico o práctico. Algunas de las aplicaciones de estos métodos pueden ser el análisis de sistemas con propiedades invariantes [1], estudiar dinámica de fluidos [2], predecir la morfología de galaxias [3], estudiar dinámica molecular [4], buscar partículas exóticas en física de altas energías [5], sólo por mencionar algunas.

Concretamente, los métodos de aprendizaje que se utilizaron fueron regresores lineales, redes neuronales artificiales con propagación hacia adelante, redes neuronales convolucionales y máquinas de soporte vectorial. Aunque el método con el que más se trabajó fueron las redes neuronales con propagación hacia adelante, los demás métodos se explican brevemente y también se explora su desempeño con un esquema de clasificación que se desarrolló, que aunque se menciona en la literatura, no es común que se utilice. En los sistemas descritos más adelante, también se estudian varios aspectos de este esquema, analizando algunas de sus ventajas así como aspectos negativos del mismo.
Este enfoque desarrollado es uno de los aportes principales de la tesis, ya que puede ayudar a disminuir el tiempo de cómputo empleado en el entrenamiento del método de AA, incluso mejorando el desempeño del mismo.

A pesar de que los algoritmos de AA utilizados no involucran demasiados parámetros libres, pueden llegar a estimar los parámetros físicos de interés con un error relativo de $\pm 1\,\%$, clasificando correctamente las simulaciones hasta un $98.6\,\%$ de las veces.

La tesis se presenta en el siguiente orden. En el segundo Capítulo se exponen los diferentes métodos de aprendizaje que utilizaron en el estudio de los sistemas físicos y adicionalmente se describe el esquema de clasificación que se utiliza en la mayoría de los sistemas expuestos. Después, en el tercer Capítulo, se estudian oscilaciones de Bloch partiendo de un sistema

sencillo descrito en una dimensión, hasta un sistema más complejo como lo es el grafeno y el cual se encuentra expuesto a distintas condiciones físicas iniciales. Este Capítulo es el que cuenta con más casos de estudio y para el análisis de las oscilaciones se utilizan regresores lineales, redes neuronales artificiales (RNAs) con propagación hacia adelante y redes neuronales convolucionales. Posteriormente, en el Capítulo número cuatro, se analizan brotes de rayos gamma estimando cantidades físicas asociadas a los mismos, presentando un esquema de refinamiento haciendo uso de redes neuronales artificiales. En el quinto Capítulo, se estudian bloqueos dentro de tuberías en dos dimensiones por las cuales fluye un fluido, donde primero se predice la morfología y posición del bloqueo haciendo uso de una red neuronal artificial y posteriormente se compara el desempeño entre una RNA y una máquina de soporte vectorial al predecir el tamaño y la posición de la obstrucción. El último de los sistemas estudiados se presenta en el Capítulo seis, donde se analizan curvas de Bragg, con la idea en mente de una futura aplicación médica.

Capítulo 2

Métodos de aprendizaje automático

Los métodos de AA como su nombre lo sugiere, aprenden en base a la información que se les suministra. Estos algoritmos tratan de generalizar y reconocer patrones, a partir de tal información suministrada en forma de ejemplos. Así como los seres humanos vamos aprendiendo en base al conocimiento y la experiencia adquirida durante nuestra vida, los algoritmos de AA aprenden en base a los ejemplos que se le muestran, sin ser explícitamente programados para este fin. En vez de programar el algoritmo con instrucciones paso a paso, este enfoque permite que la computadora aprenda de los datos, sin necesidad de que un programador le de instrucción por instrucción para realizar una tarea. En este sentido, los métodos de AA realizan un proceso de inducción de conocimiento, es decir, son métodos que permiten obtener conclusiones generales a partir de premisas que contienen casos particulares.
Por este motivo es que en la actualidad, son de gran utilidad este tipo de algoritmos, ya que entre más datos tenemos para entrenar al algoritmo, más aprenderá. De hecho, varios avances en estos métodos, se deben principalmente a que contamos con una gran cantidad de datos para entrenarlos y no tanto por la innovación en los algoritmos de aprendizaje.

El tipo de aprendizaje que se utiliza en los métodos de AA, puede ser clasificado de la siguiente manera:

- **Aprendizaje supervisado.** Los algoritmos que utilizan este aprendizaje, generan una función que establece una correspondencia entre las entradas y las salidas deseadas, donde la base del conocimiento está formada por ejemplos etiquetados.
- **Aprendizaje no supervisado.** En contraste con el aprendizaje supervisado, el aprendizaje no supervisado se usa cuando la información con la que se entrena al algoritmo, no cuenta con una etiqueta, por lo que el algoritmo busca características similares de los datos de entrada para poder agruparlos de acuerdo a la similitud entre estos.
- **Aprendizaje semi-supervisado.** Es una combinación de los dos aprendizajes anteriores, en donde se cuenta con datos etiquetados y sin etiquetar. Típicamente en estos algoritmos, se cuenta con poca información etiquetada y con mucha información sin etiquetar.
- **Aprendizaje por refuerzo.** Los algoritmos que usan este tipo de aprendizaje interactúan con su ambiente, realizando un proceso de ensayo y error, reforzando o recompensando aquellas acciones que reciben una respuesta positiva. Este aprendizaje permite determinar automáticamente el comportamiento ideal en un contexto específico, con el fin de maximizar su desempeño.

Estas son las categorías principales en las que se suele clasificar el tipo de aprendizaje, pero uno puede encontrar otro tipo de aprendizajes en la literatura.

En esta tesis solo se usaron métodos de AA con aprendizaje supervisado, ya que al ser nosotros los que introducimos las condiciones iniciales para generar las simulaciones físicas, contamos con información sobre los parámetros físicos del sistema que nos interesan y deseamos que nuestros métodos aprendan de acuerdo a estos parámetros. Por este motivo, contaremos con un conjunto de datos iniciales de los que tenemos información entre las características o variables independientes del sistema y de las variables dependientes de las que se necesita que el método aprenda. Normalmente este conjunto total de datos que se tiene inicialmente se divide en tres grupos diferentes: el conjunto de entrenamiento que sirve para entrenar el algoritmo, el conjunto de prueba que sirve para saber que tan bien funciona

el modelo al procesar datos ajenos al entrenamiento y el conjunto de validación, el cual sirve para probar que tan bien se minimiza la función de costo al variar parámetros del método de AA como la constante de aprendizaje o el número de parámetros dentro del modelo, así como también saber si es conveniente detener el aprendizaje para evitar sobreentrenarlo. En ocasiones, sólo se divide en un conjunto de entrenamiento y uno de prueba, excluyendo el conjunto de validación.
En la mayoría de los estudios expuestos en los próximos capítulos, se seleccionó aleatoriamente el 70 % de los datos correspondientes a las simulaciones de los sistemas físicos para el entrenamiento, 15 % como conjunto de prueba y 15 % como conjunto de validación. Este esquema es el que se suele usar cuando se quieren clasificar miles de datos. En caso de tener millones de datos se usa otro enfoque al momento de seleccionar estos conjuntos.

2.1. Regresores lineales y logísticos

El primer método de *machine learning* que estudiaremos es el que se conoce como regresor, el cual es entrenado haciendo uso de un aprendizaje supervisado, siendo de los métodos de AA más sencillos de implementar y puede ser muy útil para encontrar correlación entre los datos que se están analizando. Este tipo de aprendizaje requiere que contemos inicialmente con una serie de datos, mediciones u observaciones, en las que tenemos al menos un par de datos $\{x, y\}^p \in \Re$, donde el superíndice p indica la medición ($1 \leq p \leq P$, siendo P el total de mediciones) en las que se relaciona una variable independiente x con una dependiente y. Tales datos pueden provenir, por ejemplo, de una serie de tiempo en donde la variable independiente x representa el tiempo y la variable dependiente y puede ser la temperatura promedio de un cierto día p, o y puede ser el valor del peso mexicano frente al dólar, o pudiera ser la cantidad de dinero que se gasta en ese día una persona. En general podríamos tener $\{x_1, x_2, \ldots, x_m, y\}^p$ valores por medición p, siendo m variables independientes relacionadas a la variable dependiente y.

El objetivo es entonces encontrar una función $\hat{y}$, tal que al introducir un conjunto nuevo de valores $\{x_1, \ldots, x_m\}$ el método sea capaz de predecir

el valor de la variable dependiente y asociado a este conjunto. Para esto es necesario hacer uso de la colección de mediciones previamente adquiridas. Usualmente en el modelo más sencillo para un regresor se propone la función $\hat{y}$ como

$$\hat{y} = f(w_1 x_1 + w_2 x_2 + \ldots + w_m x_m + b), \tag{2.1}$$

donde los parámetros a ajustar del regresor son las variables reales $w_1, \ldots,$ w_m a los que se llaman pesos y un término adicional de sesgo b (*bias* en inglés). La función f es una función que se propone en base a prueba y error dependiendo de que tan buenas sean las predicciones una vez que se han ajustado los parámetros del regresor y de la naturaleza del problema. En caso de que esta función sea lineal respecto a los parámetros ($f(\sigma) = \sigma$, con $\sigma = \sum_i^m w_i x_i^p + b$), se le llama al método regresor lineal y en caso de que se use una función sigmoide ($f(\sigma) = 1/(1 + e^{-\sigma})$) se le llama regresor logístico.
Este modelo de regresor tiene entonces un total de $m + 1$ parámetros para ajustar, los cuales se adaptan usando la información disponible de las p mediciones. Lo que se suele hacer es minimizar una función que depende de estos parámetros, de tal manera que los resultados que genere el modelo se aproximen a los valores y conocidos. En el caso de un regresor lineal, la función a minimizar más usada es una de tipo error cuadrático medio:

$$C(\omega_1, \ldots, \omega_m, b) = \frac{1}{2P} \sum_{p=1}^{P} (y^p - \hat{y}^p)^2. \tag{2.2}$$

La función C es llamada función de costo y depende de los parámetros a través de $\hat{y}^p = \sum_i^m w_i x_i^p + b$. Esta función se aproxima a 0 cuando el valor producido por $\hat{y}$ es cercano al valor y que se conoce, lo que quiere decir que la estimación es buena. En cambio, si el valor de esta función es muy grande, quiere decir que nuestro modelo está prediciendo un valor $\hat{y}$ muy diferente del valor y objetivo (o *target* en inglés, razón por la que en algunas secciones posteriores se denota con la letra T) que conocemos.

Para encontrar entonces una regla para adaptar los pesos de manera que se minimice la función de costo se emplea una técnica de gradiente descendente, tal que para encontrar la dirección en la que decrece más

rápido la función de costo al variar los $m+1$ parámetros del modelo, se requiere que

$$\Delta\boldsymbol{\omega} = -\gamma\nabla\mathbf{C} = -\gamma\frac{\partial\mathbf{C}(\boldsymbol{\omega})}{\partial\boldsymbol{\omega}}, \tag{2.3}$$

donde $\boldsymbol{\omega}$ es un vector con cada componente correspondiendo a uno de los $m+1$ parámetros a adaptar, $\Delta\boldsymbol{\omega} = \boldsymbol{\omega}(s+1) - \boldsymbol{\omega}(s)$ siendo s el número de iteración en el entrenamiento y γ es la llamada constante de aprendizaje, el cual es otro parámetro libre del modelo. Al comenzar el entrenamiento se acostumbra inicializar los parámetros ($s=0$) aleatoriamente con valores cercanos a cero o con alguna distribución de probabilidad conveniente y la constante γ se selecciona en base a prueba y error. Entonces al derivar la función de costo con respecto a cada uno de los parámetros obtenemos que la derivada con respecto a los pesos es

$$\begin{aligned}
\frac{\partial C(\omega)}{\partial\omega_i} &= \frac{\partial}{\partial\omega_i}\frac{1}{2P}\sum_{p=1}^{P}(y^p-\hat{y}^p)^2 \\
&= \frac{2}{2P}\sum_{p=1}^{P}\Big(y^p-\hat{y}^p\Big)\Big(-\frac{\partial\hat{y}^p}{\partial\omega_i}\Big) \\
&= -\frac{1}{P}\sum_{p=1}^{P}\Big(y^p-\hat{y}^p\Big)\frac{\partial}{\partial\omega_i}\Big(\sum_{j'}^{m}w_{j'}(s)x^p_{j'}\Big) \\
&= -\frac{1}{P}\sum_{p=1}^{P}\Big(y^p-\hat{y}^p\Big)\delta_{ij'}x^p_{j'} \\
&= -\frac{1}{P}\sum_{p=1}^{P}\Big(y^p-\hat{y}^p\Big)x^p_i,
\end{aligned} \tag{2.4}$$

donde el índice i corre de 1 a m, mientras que la derivada con respecto al

término de sesgo es

$$\begin{aligned}
\frac{\partial C(\omega)}{\partial b} &= \frac{\partial}{\partial b}\frac{1}{2P}\sum_{p=1}^{P}(y^p - \hat{y}^p)^2 \\
&= \frac{2}{2P}\sum_{p=1}^{P}\Big(y^p - \hat{y}^p\Big)\Big(-\frac{\partial \hat{y}^p}{\partial b}\Big) \\
&= -\frac{1}{P}\sum_{p=1}^{P}\Big(y^p - \hat{y}^p\Big)\frac{\partial}{\partial b}(b) \\
&= -\frac{1}{P}\sum_{p=1}^{P}\Big(y^p - \hat{y}^p\Big). \qquad (2.5)
\end{aligned}$$

Al sustituir las derivadas de la función de costo y el valor de $\hat{y}^p$ en la Ec. (2.3) se llega a que la regla para adaptar los pesos y el término de sesgo es

$$\begin{aligned}
\omega_i(s+1) &= \omega_i(s) + \gamma\frac{1}{P}\sum_{p=1}^{P}\Big(y^p - (\sum_{j}^{m} w_j(s)x_j^p + b(s))\Big)x_i^p, \\
b(s+1) &= b(s) + \gamma\frac{1}{P}\sum_{p=1}^{P}\Big(y^p - (\sum_{j}^{m} w_j(s)x_j^p + b(s))\Big). \qquad (2.6)
\end{aligned}$$

La actualización de los pesos de acuerdo a la Ec. (2.6) se realiza después de haber utilizado cada una de las mediciones, a lo que se le llama entrenamiento fuera de línea (*offline* o *batch* en inglés). Se le nombra entrenamiento en línea o estocástico al actualizar los pesos después de usar cualquiera de las mediciones. Por último, si el conjunto de mediciones se divide en diferentes subconjuntos y se actualizan los pesos después de usar alguno de estos subconjuntos, al entrenamiento se le llama *minibatch* y es un tipo de entrenamiento que tiene características tanto del *offline* como del *online*. El tipo de entrenamiento requerido, depende de la cantidad de datos que se quiera procesar. Se usó un entrenamiento *offline* en la mayoría de sistemas analizados debido a que la cantidad de datos no era demasiada y al usar todo el conjunto de datos, la función de costo tiende a disminuir sin oscilar demasiado, es decir, sin estancarse en mínimos locales.

Los regresores también pueden ser utilizados como clasificadores de patrones. En tal escenario, la variable dependiente y tiene asociado dos posibles valores, los cuales representan si la variable independiente x pertenece a alguna de dos posibles etiquetas o clases. Por ejemplo, podría ser que estamos interesados en saber si una persona es apta para ofrecerle un préstamo, por lo que podríamos tener como datos de la persona la edad, su ingreso mensual, dependientes económicos, etc y en base a esta información, saber si es apto para ofrecerle el préstamo. Entonces, a la variable y se le puede asignar uno de los dos posibles valores en caso de ser apto, usualmente se le asigna el valor de 1 y 0 para diferenciar las dos clases.
En esta situación es conveniente usar los regresores logísticos, dado que la función sigmoide que se utiliza en este caso genera valores entre 0 y 1. Un valor generado por el regresor cercano a 1 representaría que la persona es apta para el préstamo en el ejemplo mencionado, mientras que un valor cercano a 0 representaría que no es apto para el préstamo. Además, también es conveniente usar una función de costo, que tiene la forma

$$C(\omega_1, \ldots, \omega_m, b) = -\frac{1}{P}\left(\sum_{p=1}^{P} y^p Log(\hat{y}^p) + (1 - y^p) Log(1 - \hat{y}^p)\right). \quad (2.7)$$

Esta función de costo se llama entropía cruzada, recordando que en el caso de un regresor logístico $\hat{y}^p = 1/(1 + e^{-\sigma})$. Aplicando un gradiente descendente a esta función de costo, se obtiene que la regla para adaptar los pesos en esta situación es idéntica a la Ec. (2.6). La razón de usar esta función de costo recae en que tiene una interpretación probabilística y de hecho, la manera formal de estudiar los regresores, es analizando la distribución de probabilidad de los pares de datos (x, y) que se tienen [6, 7], donde la colección de observaciones en realidad tiene una incertidumbre natural debido a la manera en que se midieron los datos o a que el instrumento de medición no esté bien calibrado, o la incertidumbre natural del mismo instrumento, entre otros motivos.

En caso de existir más de 2 clases en las que se pueden clasificar las variables independientes, lo que se puede hacer es utilizar un total de regresores igual al número de clases existentes utilizando un enfoque de clasificación de uno contra todos (one-vs-all u OVA) [8]. Utilizando este enfoque,

se tiene un total de y_c diferentes salidas generadas con el índice c corriendo desde 1 hasta el número total de clases. Una de las desventajas tomando este tratamiento es que si se necesita clasificar un número de clases muy grande, se incrementa el tiempo de entrenamiento al tener que adaptar una mayor cantidad de parámetros. Por este motivo, en los sistemas donde se usan los métodos de AA más adelante, se explora otro enfoque para clasificar los patrones de datos, el cual se describe con detalle en la última sección de este capítulo.

2.2. Redes neuronales artificiales

A las redes neuronales artificiales se les puede considerar como una colección de regresores, ya que los elementos computacionales o neuronas funcionan como regresores independientes que después de procesar la información la envían a otras neuronas.
Este tipo de método de AA como su nombre lo sugiere, fue desarrollado en analogía a la manera en la que se transfiere la información entre las neuronas biológicas de nuestros cerebros [9]. Dependiendo de la manera en la que están organizadas las neuronas y de como es el flujo de información entre ellas en estos modelos, es como se nombran las estructuras de redes diferenciándose principalmente dos: las redes neuronales con propagación hacia delante y las redes neuronales recurrentes (ver Figura 2.1).

El tipo de estructura neuronal que se utilizó en los sistemas de estudio es la que se denomina con propagación hacia adelante, en donde los elementos procesadores o neuronas están organizados en capas y en donde el flujo de información ocurre en una sola dirección: desde la capa de entrada, hasta la capa de salida, pasando por un número arbitrario de llamadas capas ocultas. Cada una de las neuronas en cada capa están conectadas con todas las neuronas de la capa subsecuente, a través de los pesos, los cuales tienen la interpretación de ser las conexiones biológicas entre las neuronas y que los podemos relacionar con una estimulación de exitación o inhibición hacia la neurona a la que fluye la información. Debido a que el flujo de información no puede ocurrir entre neuronas pertenecientes a la misma capa o a capas anteriores, es que recibe el nombre de propagación hacía

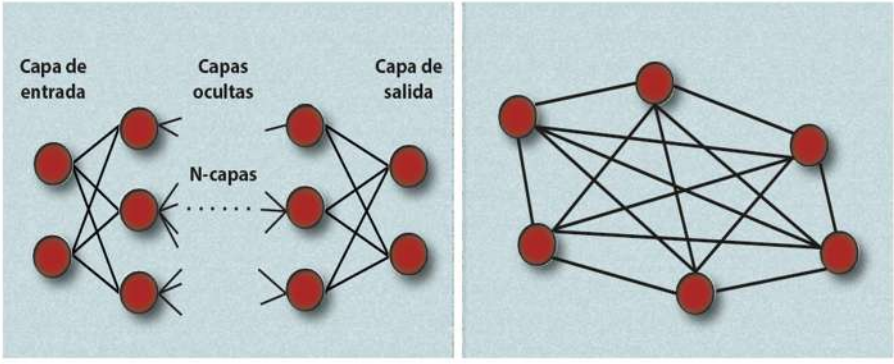

Figura 2.1: Esquema de los 2 tipos de estructuras de RNAs. Izquierda: Red neuronal con propagación hacia adelante. Derecha: red neuronal recurrente.

adelante, contrario al caso de las redes recurrentes, donde puede existir flujo de información entre cualquier neurona de la red e incluso puede suceder que la información producida por las neuronas es procesada nuevamente por las mismas antes de enviarla hacia otra neurona.
Estos dos tipos de estructura tienen sus ventajas y desventajas dependiendo del problema que se quiere solucionar, pero el que se decidió usar en esta tesis, se escogió debido a que se implementa fácilmente y se puede adecuar para que el tiempo de entrenamiento sea de horas, empleando una computadora de escitorio.

Como las neuronas están organizadas en capas, en las RNA con propagación hacia delante, los pesos entre cada capa son expresados por medio de matrices, siendo la matriz Ω^1 la que contiene los pesos entre la capa de entrada y la primer capa oculta, la matriz Ω^2 contiene los pesos entre la primer y segunda capa oculta y así sucesivamente hasta la matriz Ω^L que contiene los pesos entre la penúltima y la última capa, siendo L el total de capas de la red. Para propagar entonces la información por la red, primero se introduce la información disponible del problema, es decir, las variables independientes $\{x_1, \ldots, x_m\}^p$ que se tienen para cada medición p, por lo que la capa de entrada contiene un total de m ($m = N^0$ para notación más general) neuronas, las cuales envían la información a las N^1 neuronas

en la primer capa oculta a través de los pesos Ω^1. Explícitamente, esta matriz contiene los pesos $\omega^1_{11}, \ldots, \omega^1_{N^0 N^1}$. Adicionalmente, a cada neurona en la primer capa oculta se le asocia un término bias $b^1_1, \ldots, b^1_{N^1}$. Cada una de las neuronas en la primer capa oculta genera un valor a través de lo que se denomina una función de activación y que denotamos por f^1. Esta función de activación puede ser una función lineal o una función sigmoide como en el caso de los regresores, pero también se suelen emplear otro tipo de funciones como una tangente hiperbólica, una función lineal rectificada (RELU por sus siglas en inglés) o una función softmax y la selección de las mismas dependerá del problema que se está atacando. La primer neurona en la primer capa oculta genera el valor $f^1(\sum_{n^0=1}^{N^0} \omega^1_{n^0 1} x^p_{n^0} + b^1_1)$ y en general tenemos $f^1_{n^1} = f^1(\sigma^p_{n^1}) = f^1(\sum_{n^0=1}^{N^0} \omega^1_{n^0 n^1} x^p_{n^0} + b^1_{n^1})$ donde el subíndice n^1 indica la neurona en la primer capa oculta ($1 \leq n^1 \leq N^1$). Este proceso continua hasta la última capa L donde se produce el valor de salida $\hat{y}^p_{n^L} = f^L(\sigma^p_{n^L})$ para cada una de las salidas n^L.
Después de propagar la información, se adaptan los pesos de la RNA minimizando nuevamente una función de costo de error cuadrático medio, pero más general ya que involucra las múltiples salidas generadas por la red

$$C(\omega^1_{11}, \ldots, b^1_1, \ldots, \omega^L_{11}, \ldots) = \frac{1}{2P} \sum_{p=1}^{P} \sum_{n^L=1}^{N^L} (y^p_{n^L} - \hat{y}^p_{n^L})^2. \qquad (2.8)$$

Se puede notar que la función de costo es muy parecida a la Ec. (2.2) para el regresor, pero en este caso hay una suma entre el total de salidas N^L. Primero se encuentra cómo se adaptan los pesos y *bias* de la última capa (Ω^L, b^L) usando un gradiente descendente. Después, los pesos y *bias* de la penúltima capa y así sucesivamente hasta los pesos de la primer capa oculta. A este método se le conoce como aprendizaje de retropropagación (*backpropagation* en inglés), por el motivo que se encuentra primero como se adaptan los pesos de la última capa y al final los pesos de la primer capa.

La relación que se encuentra para adaptar los pesos en la última capa es

$$\begin{aligned} \omega^L_{n^{L-1}n^L}(s+1) &= \omega^L_{n^{L-1}n^L}(s) + \gamma \frac{1}{P} \sum_{p=1}^{P} (y^p_{n^L} - f^L_{n^L}) f'^L_{n^L} f^{L-1}_{n^{L-1}}, \\ b^L_{n^L}(s+1) &= b^L_{n^L}(s) + \gamma \frac{1}{P} \sum_{p=1}^{P} (y^p_{n^L} - f^L_{n^L}) f'^L_{n^L}, \end{aligned} \tag{2.9}$$

donde $f'^L_{n^L}$ denota la derivada de la función de activación n^L en la capa L con $1 \leq n^{L-1} \leq N^{L-1}$ y $1 \leq n^L \leq N^L$, s es el paso en el entrenamiento y γ es la constante de aprendizaje. En las funciones de activación va implícito el índice p ya que recordemos que los valores se producen al propagar la información de la medición p. Dependiendo de la función de activación que se use, la expresión se puede reducir, ya que en ocasiones la derivada queda en términos de la función de activación en sí (ver [10] para ejemplos con funciones de activación sigmoides y tangentes hiperbólicas). La regla para adaptar los pesos de la penúltima capa está dada por

$$\begin{aligned} \omega^{L-1}_{n^{L-2}n^{L-1}}(s+1) &= \omega^{L-1}_{n^{L-2}n^{L-1}}(s) + \gamma \frac{1}{P} \sum_{p=1}^{P} \sum_{n^L=1}^{N^L} (y^p_{n^L} - f^L_{n^L}) \bullet \\ &\quad f'^L_{n^L} \omega^L_{n^{L-1}n^L} f'^{L-1}_{n^{L-1}} f^{L-2}_{n^{L-2}}, \\ b^{L-1}_{n^{L-1}}(s+1) &= b^{L-1}_{n^{L-1}}(s) + \gamma \frac{1}{P} \sum_{p=1}^{P} \sum_{n^L=1}^{N^L} (y^p_{n^L} - f^L_{n^L}) \bullet \\ &\quad f'^L_{n^L} \omega^L_{n^{L-1}n^L} f'^{L-1}_{n^{L-1}}, \end{aligned} \tag{2.10}$$

en esta ocasión $1 \leq n^{L-2} \leq N^{L-2}$ y $1 \leq n^{L-1} \leq N^{L-1}$. De manera general,

las demás matrices de pesos se adaptan de acuerdo a

$$\begin{aligned}
\omega^{l}_{n^{l-1}n^{l}}(s+1) &= \omega^{l}_{n^{l-1}n^{l}}(s) + \gamma\frac{1}{P}\sum_{p=1}^{P}\sum_{n^{L}=1}^{N^{L}}(y^{p}_{n^{L}} - f^{L}_{n^{L}})\bullet \\
&\quad \prod_{\hat{l}=l+2}^{L}\left(\sum_{n^{\hat{l}-1}=1}^{N^{\hat{l}-1}}\omega^{\hat{l}}_{n^{\hat{l}-1}n^{\hat{l}}}f'^{\hat{l}-1}_{n^{\hat{l}-1}}\right)\omega^{l+1}_{n^{l}n^{l+1}}f'^{l}_{n^{l}}f^{l-1}_{n^{l-1}}, \\
b^{l}_{n^{l}}(s+1) &= b^{l}_{n^{l}}(s) + \gamma\frac{1}{P}\sum_{p=1}^{P}\sum_{n^{L}=1}^{N^{L}}(y^{p}_{n^{L}} - f^{L}_{n^{L}})\bullet \\
&\quad \prod_{\hat{l}=l+2}^{L}\left(\sum_{n^{\hat{l}-1}=1}^{N^{\hat{l}-1}}\omega^{\hat{l}}_{n^{\hat{l}-1}n^{\hat{l}}}f'^{\hat{l}-1}_{n^{\hat{l}-1}}\right)\omega^{l+1}_{n^{l}n^{l+1}}f'^{l}_{n^{l}}, \qquad (2.11)
\end{aligned}$$

donde $1 \leq l \leq L-2$ y además se define $f^{0}_{i} \equiv x^{p}_{i}$. El número de neuronas en cada capa oculta $(N^{1}, \ldots, N^{L-1})$ son parámetros libres del modelo y lo que está fijo únicamente para cada problema son las neuronas de entrada N^{0} y las neuronas de salida N^{L}, las cuales corresponden al número de variables independientes y dependientes respectivamente.
Estas reglas de actualización para los pesos son las más sencillas de implementar, ya que dejan de fuera otros parámetros que son convenientes utilizar, como lo es el parámetro de regularización o el término de momento [9, 11].

A las RNAs que tienen un gran número de capas ($L \gg 1$) se les denominan redes neuronales artificiales profundas. Estas herramientas al contar con un gran número de parámetros dentro de la red, el tipo de funciones que pueden aproximar con estas estructuras, son más complejas. Además, con el poder de cómputo de hoy en día, ya no es tan prohibitivo el uso de estas herramientas como lo era hace algunas décadas.
En los sistemas físicos donde se emplearon las RNAs, la mayoría de estructuras que se implementaron fueron estructuras con dos capas: una capa oculta y una de salida, por lo que en las ecuaciones anteriores $L = 2$, implicando únicamente el uso de las Ecs. (2.9) y (2.10). Esto con el fin de saber el desempeño que tienen las RNAs con estructuras simples en los sistemas físicos estudiados y aumentar el desempeño incluso más si se quisiera, únicamente aumentando el número de capas. La RNA está implementada para

incluir un número arbitrario de capas si se desea.

2.3. Redes neuronales convolucionales

Las redes neuronales convolucionales (RNCs) reciben este nombre, cuando dentro de la estructura de una red neuronal artificial con propagación hacia adelante, al menos una de sus capas cuenta con neuronas que realizan una convolución sobre los datos que se están procesando.
Para visualizar lo que es una convolución, podemos pensar en el siguiente ejemplo. Imaginemos que queremos rastrear la posición de un vehículo $x(t)$ mediante un sensor láser a diferentes instantes de tiempo t. Ahora, supongamos que las mediciones del sensor de alguna manera involucran algo de ruido, que como se mencionó anteriormente, es algo natural presente en cualquier medición. Si queremos obtener una medición menos ruidosa de la posición del vehículo, nos gustaría promediar varias mediciones, donde las mediciones mas recientes son mas relevantes, así que requerimos de un promedio pesado, dependiendo de qué tan reciente es la medición. Se puede hacer esto con una función $\omega(a)$, donde a es la edad de la medición. Si se aplica tal operación de promedios pesados en cada instante, se obtiene una nueva función que provee una estimación suavizada de la posición del vehículo:

$$s(t) = \int x(a)\omega(t-a)da, \tag{2.12}$$

donde la integral se realiza desde el tiempo en que se realizó la última medición (medición mas reciente) hasta la primer medición (medición mas antigua). A esta operación se le conoce como convolución y es una operación matemática sobre dos funciones de argumentos reales y se suele definir como $s(t) = (x * \omega)(t)$. En este ejemplo, ω necesita ser una densidad de probabilidad válida y además debe ser cero para argumentos negativos, ya que equivaldría a mediciones realizadas a tiempos en el futuro. Tales limitaciones son particulares del ejemplo, pero en general, la convolución se define para funciones en las que la integral anterior esta definida y puede ser usada para otros propósitos.
En el lenguaje de redes neuronales artificiales, el término $x(a)$ en la integral, corresponde a los datos de entrada, el término $\omega(t-a)$ se le conoce

como núcleo (*kernel* en inglés) y a la salida se le conoce como mapa de características. Como usualmente la colección de mediciones que se tienen son discretas, es decir, se tiene un número finito de mediciones a diferentes intervalos de tiempo, se usa la versión discreta de la convolución

$$s(t) = \sum_{a=-\infty}^{\infty} x(a)\omega(t-a). \tag{2.13}$$

En aplicaciones de AA, la entrada es usualmente un arreglo multidimensional (o tensor) de datos y el núcleo es un arreglo multidimensional de parámetros, los cuales se adaptan por el algoritmo de aprendizaje. Debido a que estos son conjuntos finitos, asumimos que son cero en todos lados, excepto en el conjunto finito de puntos en los que almacenamos los valores. Esto quiere decir, que en la práctica, se implementa la suma infinita como una suma sobre el número finito de elementos de los arreglos.
Por ejemplo, en el caso de procesado de imágenes en escala de grises, al tratarse de arreglos bidimensionales, la convolución que se utiliza también es en dos dimensiones:

$$s(i,j) = (x * \omega)(i,j) = \sum_{m}\sum_{n} x(m,n)\omega(i-m, j-n), \tag{2.14}$$

donde también el núcleo ω, en este caso llamado filtro, es un arreglo bidimensional. Se le llama filtro, debido a que al momento de realizar la convolución sobre los datos de entrada, filtra la información de acuerdo a los valores considerados en el filtro. Los datos $x(i,j)$ contienen la intensidad de negro de cada pixel que conforma la imagen, correspondiendo el índice i a la fila y el j a la columna.
Dependiendo de la manera en la que se seleccionan los filtros, es la aplicación que se le puede dar a la convolución. Por ejemplo, hay filtros particulares que se utilizan para detectar bordes dentro de una imagen. También hay filtros que se seleccionan para detectar objetos en particular dentro de una imagen.
En la Figura 2.2, se muestra un esquema donde se visualiza cómo se realiza la convolución, cuando se tiene una matriz de datos de tamaño 4×4 y un filtro 2×2. Dependiendo del tamaño del filtro y de cómo se recorre para realizar la multiplicación de matrices, es el tamaño de la salida o mapa de

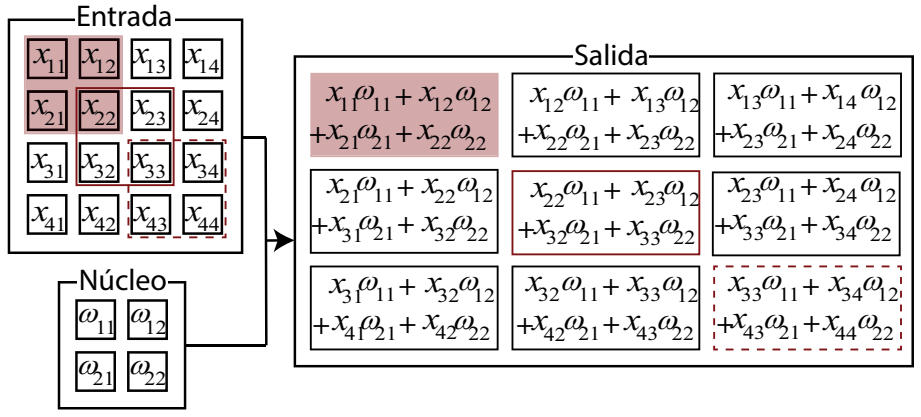

Figura 2.2: Esquema del mapa de características generado al realizar una convolución en los datos, utilizando un tamaño de paso igual a uno y sin rellenar.

características que se obtiene. Para obtener el primer valor de la salida, imaginamos que sobreponemos el filtro con los datos, correspondiendo el elemento de la primera fila y primera columna del filtro (ω_{11}), con el elemento de la primera fila y columna de la matriz de entrada (x_{11}), representados por un cuadrado rojo en el esquema. Posteriormente, se realiza una multiplicación por componentes con los valores que se sobreponen, donde se multiplican cada uno de los valores que se sobreponen y se suman para obtener el valor de la primera fila y primera columna de la salida (s_{11}), la cual está representada también por un cuadrado rojo en el esquema. Después, para obtener el segundo valor de la primera fila de salida (s_{12}), se especifica el tamaño de paso (*stride* en inglés) que hay que recorrer el filtro, para realizar nuevamente la multiplicación y suma de elementos que se sobreponen. Este tamaño de paso es un parámetro libre de la convolución. En el esquema de la Figura 2.2, el tamaño de paso es igual a uno, por lo que la salida s_{12}, se consigue recorriendo el filtro una columna hacia la derecha, donde ahora el elemento del filtro ω_{11}, se sobrepone con el elemento x_{12} de los datos de entrada y se realiza el procedimiento anterior de multiplicación y suma de los elementos. Se utiliza este procedimiento para obtener los valores de la primera fila de salida, hasta que el elemento de

la última columna del filtro (ω_{12}), se sobrepone con el último elemento de los datos (x_{14}), donde en este caso resultan un total de 3 elementos en la primera fila de salida.
Para obtener la segunda fila de salida, se recorre el filtro hacia abajo un total de filas igual al tamaño de paso especificado inicialmente, donde la salida s_{21}, se obtiene al sobreponer ω_{11} con x_{21} y realizar la multiplicación y suma de elementos correspondientes, siguiendo el procedimiento anterior para obtener los elementos restantes de esa fila. Se continua con tal procedimiento, hasta recorrer el filtro de tal manera que ω_{22} se sobrepone con el elemento x_{44}, obteniendo la salida del cuadrado con linea discontinua roja (s_{33}) en el esquema. Con este procedimiento, se obtiene una matriz de salida de tamaño 3x3.
En la Figura 2.3, se muestra un esquema similar, pero ahora considerando un tamaño de paso igual a dos, obteniendo en este caso una matriz de salida con dos filas y dos columnas.
Note que dependiendo del tamaño de paso y del filtro, puede suceder que al recorrer el filtro no todos los elementos se sobrepongan con la matriz de datos. Por ejemplo, si utilizamos un tamaño de paso igual a tres, al ir recorriendo el filtro sobre la primera fila para obtener las salidas, al tratar de obtener el segundo valor de esa fila, el elemento ω_{11} se sobrepone con el elemento x_{14}, pero ω_{12} no se sobrepone con ningún valor. Lo mismo sucede con el elemento ω_{22}.
Por este motivo, se suele introducir un parámetro de relleno (*padding* en inglés), de tal manera que se obtenga una convolución válida, es decir, que al recorrer el filtro a través de los datos, no queden elementos sin sobreponer como en el ejemplo anterior. Si este parámetro de relleno es igual a uno, se agrega una columna de ceros a la izquierda de la primera columna de los datos de entrada y a la derecha de la última columna de la misma; también se agrega una fila de ceros arriba de la primera fila de los datos de entrada y abajo de la última fila, obteniendo así una matriz de datos de tamaño 6×6. Si utilizamos un relleno igual a dos y tamaño de paso igual a tres, ahora sí se obtendría una convolución válida.
En general, para saber el tamaño de la salida en la convolución, se tiene que el número de filas (n_f) y columnas (n_c) es igual a

$$n_f = \text{piso}\left(\frac{h + 2r - f}{S} + 1\right); \quad n_c = \text{piso}\left(\frac{a + 2r - f}{S} + 1\right), \tag{2.15}$$

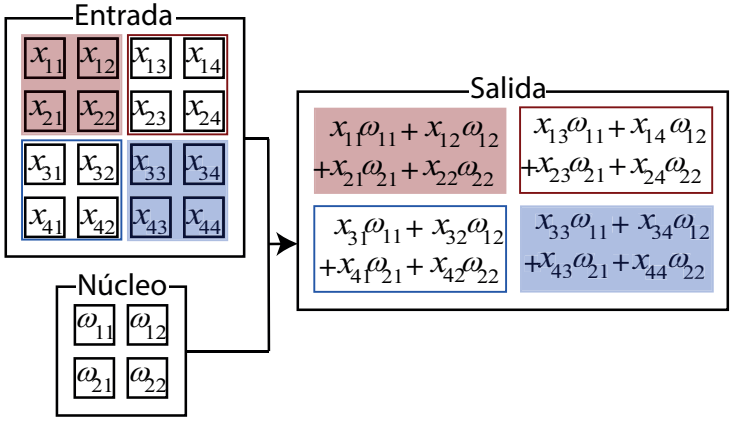

Figura 2.3: Esquema del mapa de características generado al realizar una convolución en los datos, utilizando un tamaño de paso igual a dos y sin rellenar.

siendo r el relleno, S el tamaño de paso, f el tamaño de filtro, h la altura de la imagen (número de filas en la matriz de entrada) y a el ancho de la imagen (número de columnas en la matriz de entrada). En el ejemplo anterior $h = a$, por lo que $n_f = n_c$.
Al usar la función piso, nos aseguramos de que los valores en donde se realiza la convolución, son aquellos donde no quedan valores sin sobreponer al recorrer el filtro a través de los datos de entrada. Esto es debido a que, cuando se trata con convoluciones no válidas, el número que resulta no es entero. En los esquemas mostrados, también podemos observar una de las propiedades de la convolución: la conectividad dispersa. Contrario a las redes neuronales completamente conectadas, en las redes convolucionales, las salidas generadas no dependen de todas las entradas, es decir, debido a que no todas las entradas se conectan con las salidas, existe un número menor de conexiones. Por lo tanto, el número de parámetros que se adaptan en el entrenamiento, es menor comparado con una red completamente conectada.

En una capa convolucional de una red neuronal artificial, típicamente existen tres etapas: la etapa de convolución, la etapa de detección y la etapa de agrupación (*pooling*). La etapa de convolución genera conjuntos lineales de

activación y consiste en el procedimiento previamente descrito. Posteriormente, en la etapa de detección, a las salidas generadas se les aplica una función de activación no lineal, como una función RELU o sigmoide. Por último, en la etapa de agrupación, se realiza una reducción de la cantidad de parámetros que se van a requerir en capas posteriores de la red, siendo la más común la agrupación máxima o *max pooling*. En esta etapa de agrupación máxima, se seleccionan subconjuntos de elementos de los valores obtenidos en la etapa de detección y de éstos conjuntos seleccionados, se toma el máximo de ellos. Además, esta etapa también cuenta con parámetros de tamaño de paso y tamaño de agrupación, éste último, análogo al tamaño del filtro.

El procedimiento para agrupamiento es parecido al que se utilizó al momento de realizar la convolución de los datos, donde vamos a ir seleccionando los elementos de izquierda a derecha y de arriba abajo, de acuerdo al tamaño de agrupación y de paso. Como ejemplo, tomamos un tamaño de agrupación y tamaño de paso igual a dos y consideramos el esquema de la Figura 2.3 como referencia para realizar una agrupación máxima, donde las salidas cambian con respecto a este esquema. Para obtener la primer salida, se consideran los datos dentro del tamaño de agrupación, donde el primer agrupamiento contiene los elementos $x_{11}, x_{12}, x_{21}, x_{22}$, que corresponden a los elementos del cuadrado rojo en la entrada. De estos elementos se selecciona el máximo de ellos, por lo que la salida $s_{11} = \max\{x_{11}, x_{12}, x_{21}, x_{22}\}$. Después, la segunda salida de la primera fila, se obtiene al recorrer el agrupamiento un tamaño de paso igual a dos y seleccionando nuevamente el máximo de estos elementos, en este caso corresponde al máximo de los elementos dentro del cuadrado con borde rojo en la entrada. Entonces, $s_{12} = \max\{x_{13}, x_{14}, x_{23}, x_{24}\}$. Similarmente, las salidas de la segunda fila son $s_{21} = \max\{x_{31}, x_{32}, x_{41}, x_{42}\}$ y $s_{22} = \max\{x_{33}, x_{34}, x_{43}, x_{44}\}$, que corresponde al máximo de los elementos en el cuadrado con borde azul en la entrada y en el cuadrado azul respectivamente.

Para obtener el tamaño de la salida, se puede usar también la Ec. (2.15), en donde en vez de usar el tamaño del filtro, se usa el tamaño de la agrupación y además se usa un relleno igual a cero.

En la etapa de agrupamiento, únicamente se reduce la cantidad de información y no existen parámetros para adaptar en esta etapa, por lo que al entrenar los pesos dentro de las capas convolucionales, se consideran sólo

los pesos presentes en las etapas 1 y 2 de la convolución.

Con lo descrito anteriormente, ya podemos incluir capas convolucionales dentro de una red neuronal artificial con propagación hacia adelante, entrenando también los pesos presentes en las capas convolucionales. La información expuesta para la convolución en esta sección, es la esencial para construir una capa convolucional, pero si se quiere profundizar en el tema, el lector puede consultar [12].

2.4. Máquinas de soporte vectorial

Enseguida pasaremos a describir otro método de aprendizaje automático, que a diferencia de las RNAs, no es un algoritmo inspirado en la naturaleza y se basa en ideas matemáticas más abstractas.

Las máquinas de soporte vectorial (MSV o *support vector machines*) fueron desarrolladas empleando ideas matemáticas sencillas y tienen un buen desempeño en aplicaciones prácticas, como son: clasificación de patrones, extracción de características y tareas de regresión, solo por mencionar algunas.

Se puede pensar en una MSV, como en un método lineal en un espacio multidimensional de características de mayor dimensión, relacionado con el espacio de entrada a través de funciones núcleo. Es decir, nuestros patrones de entrenamiento $\vec{x}^p$ los cuales tienen dimensión M, son mapeados al conjunto $\vec{\phi}(\vec{x}^p)$ de dimensión L, utilizando las funciones núcleo $\phi_1(\vec{x}^p), \ldots, \phi_L(\vec{x}^p)$. Debido a que en el espacio de características donde viven $\vec{\phi}$ se pueden separar linealmente los datos, este tipo de herramienta se analiza teóricamente de manera más simple, si se compara con las RNAs.
En este espacio de mayor dimensión, la MSV busca hiperplanos que equidisten de los ejemplos más cercanos de cada clase, donde inicialmente los patrones o ejemplos a clasificar, se encuentran etiquetados de acuerdo a la clase a la que pertenecen. Al encontrar tal hiperplano, obtenemos lo que se denomina un margen máximo a cada lado del hiperplano. Además, al momento de definir estos hiperplanos, se consideran los ejemplos de entrenamiento de cada clase que caen justo en la frontera de dichos márgenes y

a estos ejemplos, se les denomina vectores de soporte.

Entonces la MSV consiste en determinar la función de decisión óptima, resolviendo el problema de optimización [13, 14]

$$\min_{\vec{w},b,\vec{\xi}} \frac{1}{2}\vec{w}\cdot\vec{w} + \kappa \sum_{p=1}^{P} \xi^p \ , \tag{2.16}$$

sujeto a

$$y^p \left[\vec{w}\cdot\vec{\phi}(\vec{x}^p) + b\right] \geq 1 - \xi^p \ , \tag{2.17}$$

donde $p = 1, ..., P$ etiqueta cada uno de los patrones de entrenamiento, es decir, los vectores de datos $\vec{x}^p$ de entrada. En este esquema, los patrones de entrada están relacionados con una de dos posibles clases: $y_p = \{1, -1\}$. Los parámetros a optimizar en la función de decisión son $\vec{w} \in \Re^L$, el *bias* $b \in \Re$ y las funciones no negativas que permiten la separabilidad $\xi^p \geq 0$ (son conocidas como MSV de margen suave $L1$). Por último, los parámetros libres son las funciones de mapeo o núcleos $\vec{\phi} : \Re^M \to \Re^L$ y κ es el parámetro de penalización del término de error.
En lugar de resolver el problema definido por las Ecs. (2.16) y (2.17), se transforma en un problema sin restricciones usando multiplicadores de Lagrange no negativos y la solución óptima se encuentra resolviendo las condiciones de Karush-Kuhn-Tucker. También, en vez de usar directamente las funciones de mapeo $\vec{\phi}$, resolvemos el problema dual en el espacio de características con el *kernel* asociado con la función de mapeo $K(\vec{x}, \vec{y}) = \vec{\phi}(\vec{x}) \cdot \vec{\phi}(\vec{y})$.
En el enfoque descrito suponemos que los valores y^p asociados a los patrones $\vec{x}^p$ son valores binarios, donde cada valor, corresponde a una de dos posibles clases. Como en general, se pueden etiquetar los patrones en C clases diferentes ($C > 2$), el esquema anterior se aplica al descomponer el problema original en subproblemas de clasificación binaria.

Hasta aquí, la descripción ha sido muy superficial y sin entrar en detalles sobre el desarrollo de las ecuaciones presentes en esta sección, pero se puede consultar también [15] para detalles adicionales.

2.5. Clasificación mediante salidas multiclase

A continuación, se describe el enfoque de clasificación utilizado en la mayoría de aplicaciones presentes en la tesis, el cual sirve para clasificar directamente los patrones en un total de C clases diferentes y no se necesita la descomposición en subproblemas de clasificación binaria, como en el caso de la MSV. Este método de clasificación no es muy común en la literatura de aprendizaje automático, pero tiene algunas propiedades que lo hacen eficiente dependiendo del problema al que se enfrenta y de la precisión deseada.

Usualmente los métodos de aprendizaje se utilizan ya sea como regresores, para predecir algún valor real asociado al problema de estudio, o como clasificador, donde se clasifican los datos procesados de acuerdo a alguna propiedad que tienen. En el caso que son empleados como clasificadores, el enfoque que se les suele dar es el de usar una clasificación uno contra todos, que de acuerdo al número de clases en el que se van a clasificar las propiedades estudiadas, es el número de respuestas o salidas del algoritmo de AA utilizado.

Uno de los ejemplos más utilizados en el área de aprendizaje automático, es el de clasificar imágenes de dígitos escritos a mano de acuerdo al dígito que representa. Como cada persona tiene una manera de escribir diferente, cada uno de los dígitos escritos tendrá ligeras variaciones, por lo que para poder reconocer los dígitos se recomienda usar este tipo de herramientas de aprendizaje. En este ejemplo particular, usando la clasificación por k-índices mudos, se necesitan un total de diez clases, una para cada dígito a clasificar, por lo que se dota a nuestra herramienta de aprendizaje con diez salidas diferentes. Cada una de estas salidas nos servirá para relacionar los datos procesados con el grupo en el que se desea clasificar. Por ejemplo, si el dígito a clasificar es el cero, se le indica al algoritmo que el valor que debe producir debe ser un uno en la primer salida y cero en las demás, si el dígito es el uno, la segunda salida debe ser uno y cero en las demás y así sucesivamente. De esta forma, podemos asociar los resultados generados por el algoritmo de aprendizaje con la clase a la que pertenecen los patrones de datos.

Esta es una de muchas formas en las que podemos codificar los resultados del método de AA para asociarlos con la clase a la que pertenecen. Se puede

pensar en usar una codificación binaria para tratar de disminuir el número de neuronas de salida, como realicé en mi tesis de maestría [10] y poder clasificar un total de 2^n clases diferentes, siendo n el número de salidas del algoritmo utilizado. En el caso de la clasificación de dígitos, bastaría con un total de cuatro neuronas de salida para poder clasificar hasta 16 clases diferentes.

En general, se puede pensar en que el valor numérico generado por cada salida puede relacionarse con un total de C clases diferentes en las que se pueden catalogar los patrones de datos. En los problemas que se estudiaron, solo se consideraron funciones de salida tipo sigmoide, por lo que el valor generado por estas funciones está acotado al intervalo abierto (0,1). Debido a esto, si queremos representar C clases diferentes deben ser valores numéricos restringidos a este intervalo, por lo que necesitamos seleccionar también C valores diferentes para representar cada clase. Estos valores en el lenguaje de entrenamiento automático son los objetivos o *targets* que se proponen. Una manera de lograrlo es escogiendo los valores equidistantes entre sí, proponiendo:

$$T_c = \frac{(2c-1)}{2C}, \quad 1 \leq c \leq C, \tag{2.18}$$

siendo T_c el objetivo asociado a la clase c para un total de C clases. Retomando el ejemplo de clasificación de dígitos y empleando este enfoque, se tendría que el objetivo propuesto para una sola salida si la información procesada pertenece al dígito cero es $T_1 = 1/20$, en el caso del dígito uno sería $T_2 = 3/20$ y así sucesivamente siendo $T_{10} = 19/20$ el objetivo para el dígito nueve.

La razón de seleccionarlos de esta manera, es para poder definir también de forma genérica cuando se trate de una clasificación correcta. Cuando se trabaja con una clasificación de uno contra todos, la red no producirá exactamente los objetivos propuestos, por lo que también se debe especificar cual será la salida que se activa con el valor de uno. Se suele tomar como uno el valor mayor producido por las salidas del método de AA y las demás se consideran cero, para posteriormente codificar esos valores con la clase que se está prediciendo. En nuestro caso, se considera como una clasificación correcta, si la salida del algoritmo de aprendizaje se encuentra

dentro del intervalo

$$T_c - \frac{1}{2C} \leq O_c \leq T_c + \frac{1}{2C}, \quad 1 \leq c \leq C, \tag{2.19}$$

siendo O_c la salida relacionada a la clase c, en donde el correspondiente objetivo es T_c. De esta forma, cada objetivo está uniformemente distribuido en el intervalo (0,1), definiendo un margen tanto para la izquierda como para la derecha de cada valor T_c conforme a la Ec. (2.19) y así ya podemos asignarle una clase del total de C clases para cualquier valor dentro del rango de la función de activación utilizada.
El mismo enfoque de clasificación por salidas multiclase se puede realizar para otras funciones de activación, como la tangente hiperbólica, pero hay que adecuar las condiciones para el rango de esta función, el cual se encuentra en el intervalo (-1,1).

Usando este tipo de clasificación, lo que se realizó en los problemas físicos estudiados fue, generar un número de simulaciones N las cuales se dividieron en un total de C clases de tal manera que el residuo de N/C sea cero, o en otras palabras, que cada clase esté conformada por el mismo número de simulaciones o patrones de datos.
Como las simulaciones son generadas con valores especificados por el usuario, usualmente el parámetro que se desea estudiar se varía entre un intervalo acotado. Supongamos que representamos este parámetro que queremos estudiar con la letra griega κ y se generan N simulaciones entre $[\kappa_{min}, \kappa_{max}]$. De cada simulación se extrae información física del problema que servirá como datos de entrada para los métodos de AA. Posteriormente se les asignan los *targets* dependiendo del número de clases C en los que se van a dividir los patrones de datos extraídos, correspondiendo los primeros N/C patrones a la clase uno ($c = 1$), los siguientes N/C patrones a la clase dos ($c = 2$) y así sucesivamente. Adicionalmente, si los patrones están ordenados desde el valor mínimo κ_{min}, hasta el valor máximo κ_{max} de manera creciente, se puede definir un valor promedio del parámetro κ para cada patrón perteneciente a la clase c:

$$\hat{\kappa}_c = \kappa_{min} + \frac{(2c-1)\kappa_L}{2C}, \tag{2.20}$$

siendo $\kappa_L = \kappa_{max} - \kappa_{min}$ el intervalo donde se varía κ. Si consideramos como ejemplo los patrones pertenecientes a la clase uno, los valores que toma κ

están en el intervalo $[\kappa_{min},\kappa_{min} + \kappa_L/C]$, por lo que al valor promedio $\hat{\kappa}_1$ se le puede asociar un error relativo $\pm\kappa_L/(2C)$ respecto a la longitud total donde se varía el parámetro.

En el próximo Capítulo comenzamos a estudiar algunas de las propiedades que tiene este esquema de clasificación aplicado a diferentes métodos de AA, principalmente en redes neuronales artificiales.

Capítulo 3

Aplicación a oscilaciones de Bloch

Entender el movimiento de los electrones en estructuras cristalinas es uno de los temas tradicionales de la física del estado sólido, para lo cuál se han desarrollado un gran número de técnicas. La dualidad onda-partícula de los electrones en un cristal juega un rol esencial. Particularmente, en la presencia de un campo eléctrico externo, los electrones son acelerados y su longitud de onda se acorta de tal manera que cuando es dos veces la longitud de separación de la red, sufren difracción de Bragg en los espacios recíprocos y reales, es decir, en el espacio de coordenadas y en el de momentos. Por esta razón, los electrones describen un movimiento periódico, cuyo periodo depende completamente de la periodicidad del cristal a lo largo de la dirección del campo y de la fuerza del campo mismo aplicado. Este movimiento periódico ha sido nombrado como oscilaciones de Bloch (OB) [16, 17] y su descripción es conocida desde hace casi un siglo por el trabajo de Bloch. En los sólidos reales, el tunelaje entre bandas y la dispersión de electrones ultrarápidos evitan que las OB sean observadas. Aunque las OB no son observadas en sólidos reales, representan un ejemplo esencial de la influencia de un arreglo periódico (y un campo de fuerza externo) en el movimiento cuántico de los portadores de carga. Verdaderas OB son directamente observadas bajo una variedad de condiciones experimentales en mallas semiconductoras de alta pureza [18, 19, 20, 21, 22, 23, 24, 25, 26, 27] así como también en otros sistemas con propiedades similares a cristales

granulados, incluyendo sistemas atómicos [28, 29], dieléctricos [30, 31, 32], arreglos de guías de onda de plasmones [33] y supermallas bicapa de grafeno [34, 35]. Por lo tanto, hay una relevancia obvia en el estudio de tales sistemas que van más allá de los sólidos. El objetivo final a estudiar en este capítulo es realizar análisis de OB en grafeno sujeto a diferentes condiciones iniciales impuestas.

Aunque la estructura de banda del grafeno ha sido conocida por 70 años del trabajo seminal de Wallace [36], fue hasta poco después del primer aislamiento de sus membranas [37, 38, 39] que la ciencia de materiales se sometió a una continua revolución hacia la era de los materiales bidimensionales de Dirac-Weyl [40]. Ya que los cristales planos bidimensionales son inestables contra fluctuaciones térmicas de acuerdo al teorema de Mermin-Wagner [41], el estudio temprano de estos cristales era considerado solo por conveniencia académica y por esto tardó mucho en reavivarse el interés por los materiales bidimensionales.
Nuevas propiedades de excitaciones colectivas de estos materiales, de comportamiento similar a los fermiones ultrarelativistas, permiten establecer una directa conexión con la física fundamental en colisionadores de partículas. Sin embargo, algunos efectos tradicionales en la física de estado sólido siguen siendo relevantes para explorar en materiales como el grafeno, tal como el fenómeno de OB antes mencionado.

En este capítulo analizamos el problema inverso de OB: dada una curva periódica *experimental*, que representa el movimiento espacial real de electrones en un sólido sujeto a un campo eléctrico externo como función del tiempo y asumiendo que las OB dirigen la dinámica, determinamos diferentes parámetros asociados a la oscilación como puede ser la fuerza del campo eléctrico aplicado, o equivalentemente, el espaciado de la malla para un campo eléctrico fijo o también la tensión ejercida sobre la malla en caso de existir una tensión sobre la misma.
Para poder estudiar estas oscilaciones en sistemas físicos complejos, lo conveniente es estudiar primero el caso ideal, para posteriormente complicarlo hasta llegar a lo que realmente nos interesa, que en nuestro caso es estudiar las OB en grafeno. Por este motivo, primero estudiamos las OB en una cadena lineal monoatómica, posteriormente en una malla cuadrada bi-

dimensional y por último en grafeno, donde adicionalmente se estudia el escenario donde el material es expuesto a deformaciones.

A pesar de que existe una gran cantidad de métodos para analizar problemas inversos y curvas periódicas, en los trabajos expuestos a continuación construimos algoritmos eficientes de aprendizaje automático. Considerando que los métodos de aprendizaje automático son una fuente consistente y confiable para identificar y clasificar patrones en general, decidimos usar principalmente redes neuronales artificiales para el estudio de problemas inversos, los cuales se pueden caracterizar usando una configuración de RNA adecuada. Cuando se encuentra la configuración adecuada, las RNAs pueden producir predicciones precisas para señales que nunca han sido presentadas a la red.

A continuación se presentan los diferentes sistemas físicos en donde se presentan las OB y los cuales son analizados mediante el uso de los métodos de aprendizaje automático, en donde se utiliza el enfoque de salida multiclase descrito en el capítulo anterior.

3.1. Oscilaciones de Bloch en una cadena lineal

Esta primera sección se basa en el artículo titulado *Bloch oscillations: Inverse problem* [42], el cual fue publicado en colaboración con los Drs. José A. González, Saúl Hernandez y Alfredo Raya. En este primer estudio, usamos una cadena lineal monoatómica como nuestro ejemplo de trabajo y consideramos una aproximación de amarre fuerte para primeros (1V) y segundos (2V) vecinos.

3.1.1. Oscilaciones de Bloch: Enfoque semiclásico

Comenzamos nuestra discusión con el Hamiltoniano de amarre fuerte de una cadena lineal monoatómica con espaciado a entre elementos de la

cadena. En la aproximación de 1V, tenemos

$$\begin{aligned} H_n\psi(\mathbf{k}) &= -t\psi_{n+1}(\mathbf{k}) - t\psi_{n-1}(\mathbf{k}) + \epsilon_0\psi_n(\mathbf{k}) \\ &\equiv \varepsilon^{(n)}(\mathbf{k})\psi_n(\mathbf{k}), \end{aligned} \tag{3.1}$$

donde t es el parámetro de *hopping*. Del teorema de Bloch, es inmediato encontrar que para este caso, la relación de dispersión de energía-momento es

$$\varepsilon^{(n)} = \epsilon_0 - \epsilon^{(n)}(\mathbf{k}), \tag{3.2}$$

donde

$$\epsilon^{(n)}(\mathbf{k}) = \omega(1 - \cos(|\mathbf{k}|a)), \tag{3.3}$$

ϵ_0 es la energía en el sitio y $\omega = 2t$. Similarmente, para la aproximación de 2V,

$$\begin{aligned} H_n\psi(\mathbf{k}) &= -t_2\psi_{n+2}(\mathbf{k}) - t_2\psi_{n-2}(\mathbf{k}) - t_1\psi_{n+1}(\mathbf{k}) - t_1\psi_{n-1}(\mathbf{k}) + \epsilon_0\psi_n(\mathbf{k}) \\ &\equiv \varepsilon^{(nn)}(\mathbf{k})\psi_n(\mathbf{k}). \end{aligned} \tag{3.4}$$

En este caso t_2 y t_1 son los *hopping* para el vecino correspondiente. Entonces, del teorema de Bloch tenemos que la relación de dispersión es

$$\varepsilon^{(nn)} = \epsilon_0 - \epsilon^{(nn)}(\mathbf{k}), \tag{3.5}$$

pero en este caso

$$\epsilon^{(nn)}(\mathbf{k}) = \omega(1 - \cos(|\mathbf{k}|a) - \omega'\cos(2|\mathbf{k}|a)) \tag{3.6}$$

y $\omega = 2t_1$ mientras que $\omega' = t_2/t_1$.

Sabemos también las ecuaciones de movimiento semi-clásicas para un electrón moviéndose en un campo eléctrico externo $\mathbf{E}$ orientado paralelo a la cadena lineal, las cuales son

$$\frac{d\mathbf{k}}{dt} = -e\mathbf{E}, \tag{3.7}$$

$$\frac{d\mathbf{r}}{dt} = \frac{1}{\hbar}\frac{\partial}{\partial\mathbf{k}}\epsilon(k), \tag{3.8}$$

en donde en lugar de $\epsilon(\mathbf{k})$ podemos poner $\epsilon^{(n)}(\mathbf{k})$ o $\epsilon^{(nn)}(\mathbf{k})$ de las Ecs. (3.3) o (3.6) respectivamente. Podemos entonces integrar las ecuaciones de movimiento y obtener las velocidades y trayectorias para la intensidad de un campo externo dado. Considerando la cadena orientada a lo largo del eje x y un campo eléctrico uniforme $\mathbf{E} = E\hat{e}_x$, integramos la Ec. (3.7) asumiendo la condición inicial $k(0) = 0$. Entonces

$$k(t) = -\frac{eE}{\hbar}t, \tag{3.9}$$

donde k es el momento de los electrones en la cadena lineal. Por otro lado, la velocidad del electrón esta dada por la Ec. (3.8), que para 1V es

$$\begin{aligned} v_{(n)}(k(t)) &= \frac{\omega a}{\hbar}\text{sen}(k(t)a), \\ &= -\frac{\omega a}{\hbar}\text{sen}\left(\frac{eEa}{\hbar}t\right). \end{aligned} \tag{3.10}$$

Finalmente, obtenemos el perfil de la OB simplemente integrando la Ec. (3.3), lo que nos lleva a

$$\begin{aligned} x_{(n)}(t) &= \frac{\omega}{eE}\cos\left(\frac{eE}{\hbar}at\right), \\ &= \frac{\omega}{eE}\cos(\omega_E t), \end{aligned} \tag{3.11}$$

con $\omega_E = eEa/\hbar$. Análogamente para 2V, obtenemos que

$$v_{(nn)}(k(t)) = -\frac{\omega a}{\hbar}(\text{sen}(\omega_E t) + 2\omega'\text{sen}(2\omega_E t)), \tag{3.12}$$

$$x_{(nn)}(k(t)) = \frac{\omega}{eE}(\cos(\omega_E t) + 2\omega'\cos^2(\omega_E t)). \tag{3.13}$$

Las Ecs. (3.11) y (3.13) están muy de acuerdo con las observaciones experimentales de OB. En la próxima subsección introducimos las redes neuronales artificiales como la herramienta para analizar estas oscilaciones.

3.1.2. Consideraciones de la RNA

A continuación, describimos el enfoque que usamos para clasificar las OB mediante la implementación de una RNA y después establecemos el

problema inverso de OB para probar el desempeño de la red usando este enfoque. Por simplicidad, describimos el método cuando consideramos solo la interacción con 1V. Para la generación de las señales se fija el parámetro a y la magnitud del campo eléctrico E se varía de 0.01 a 1 en pasos $\Delta E = 0.01$, es decir $E^p = p\Delta E$ con $p = 1, ..., 100$ en unidades donde $\hbar = e = 1$.
Escogemos un conjunto de puntos discretos en el tiempo t_i y un valor dado de magnitud del campo eléctrico E^p. Evaluando la Ec. (3.11) con a y t_i dados para cada E^p producen un conjunto discreto de valores para la posición x_i^p, donde el índice superior p refleja la dependencia de la posición del electrón en el campo eléctrico. Del total de trayectorias generadas se toma 70 % de ellas como conjunto de entrenamiento y el 30 % restante como conjunto de validación.

Usamos entonces una RNA para clasificar las OB en términos de la intensidad del campo eléctrico. Como un primer enfoque, podemos pensar en introducir los valores x_i^p como entradas para la RNA y propagarlas a través de ella, obteniendo como resultado un valor que interpretamos como el correspondiente campo eléctrico E^p. Para tratar de disminuir la información que se le suministra a la red y así requerir menos tiempo de cómputo, ya que se está tratando con señales muy simples, en vez de usar los valores x_i^p como entrada, es posible calcular una transformada discreta rápida de Fourier (DFFT por sus siglas en inglés) de la trayectoria y solo usar el valor máximo de la amplitud y su correspondiente frecuencia. Este enfoque reduce el número de entradas a únicamente dos.

Se usa una estructura de red con propagación hacia adelante, completamente conectada, que cuenta con una capa de entrada, una capa oculta y una capa de salida. Las funciones de activación que se usaron son sigmoides en la capa oculta y lineales en la capa de salida. Una vez que se propaga la información hacia la capa de salida, obtenemos los resultados $\tilde{E}^p$ predichos por la red, los cuales se usan junto con las valores E^p involucrados en la creación de las señales para minimizar una función de costo de error cuadrático medio a través de la Ec. (2.8) usando $E_k^p = y_{n^L}^p$ y $\tilde{E}_k^p = \hat{y}_{n^L}^p$ para $L = 2$ con $N^L = 1$ para el caso en el que se estima un parámetro al considerar la aproximación de 1V y $N^L = 2$ para el caso en el que se

estiman dos parámetro al considerar la aproximación de 2V. La función de costo es minimizada usando un algoritmo de retropropagación *offline*, utilizando la regla de adaptación para los pesos y *bias* de las Ecs. (2.9) y (2.10). Dada la trayectoria de la OB para un valor de E, los parámetros internos de la RNA son ajustados de tal manera que el valor predicho $\tilde{E}^p$ se aproxima tanto como sea posible a E^p.
Después, usamos el conjunto de trayectorias de validación para rastrear el comportamiento del proceso de entrenamiento. Si el error de validación comienza a incrementar o si el error alcanza algún valor de corte dado, entonces el proceso de entrenamiento se interrumpe y proseguimos a la fase de predicción.
Una vez el entrenamiento está completo, usamos nuevas señales para probar la precisión de la RNA mediante el conjunto de prueba, el cual cuenta con el mismo número de señales que el conjunto de validación y es creado usando valores aleatorios de E en el intervalo dado. En este escenario, se plantea el problema inverso de oscilaciones de Bloch.

A continuación se presentan los detalles técnicos sobre el preprocesamiento de los datos que son utilizados.

3.1.3. Procesado de datos

Las señales fueron discretizadas considerando el teorema de Muestreo [43]. Este teorema establece que la frecuencia de muestreo debe ser de al menos dos veces la máxima frecuencia de la señal. En nuestro caso,

$$\omega_E = Ea, \tag{3.14}$$

y también usamos

$$\Delta t = \frac{1}{2Ea}. \tag{3.15}$$

En las tres situaciones que se presentan enseguida, se utiliza esta discretización.

Oscilaciones de Bloch para interacciones de 1V. E variable (a constante)

Cien señales de OB fueron generadas usando la Ec. (3.11), variando solo E ($0.01 \leq E \leq 1$) con pasos de ΔE=0.01. Todos los demás parámetros permanecieron fijos (a=1.0, ω=0.5). El número total de t_i es igual a 700, para asegurar que la señal creada con la frecuencia más baja realice por lo menos un periodo (Figura 3.1).

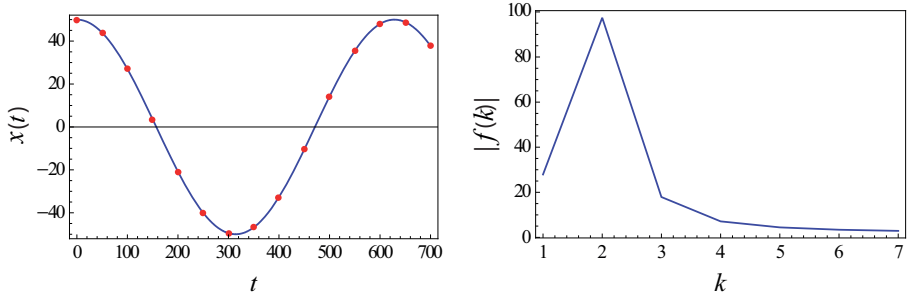

Figura 3.1: Izquierda: Trayectoria para el escenario de 1V correspondiente al más pequeño ω_E ($a = 1.0$, $w = 0.5$, $E = .01$). Los puntos muestran las posiciones donde se calcula la DFFT. Derecha: Valor absoluto de la DFFT. Las frecuencia seleccionada y su amplitud son $k^1 = 2$ y $|f(2)|$, donde el superíndice indica a que señal corresponde la frecuencia seleccionada.

Se calcula una DFFT $f(k)$ para cada trayectoria y el valor máximo de la amplitud y su frecuencia correspondiente de la señal fueron usadas como entradas para la red. Se calcula como

$$f(k) = \sum_{j=0}^{N} x^p(t_j) \mathrm{Exp}\left(\frac{-2\pi i k j}{N}\right), \tag{3.16}$$

donde los índices j y k están relacionados con los valores discretos de tiempo y de frecuencia. En el caso del tiempo, podemos cambiar el índice i por j para evitar confusiones con el número complejo i que aparece en la Ec. (3.16). La relación entre k y la frecuencia está dada por $\nu_k = \frac{k}{N} 2\omega_E$. Usamos el superíndice p para denotar la energía asociada con la señal.

La amplitud y la frecuencia son escaladas antes de usarlas como entradas para la RNA usando la siguiente transformación

$$\tilde{X} = \frac{X - \frac{1}{P}\sum X}{\max(X) - \min(X)}, \tag{3.17}$$

donde los valores máximos y mínimos de X son calculados sobre todos los elementos del conjunto de entrenamiento. Sustituyendo k^p y $|f(k^p)|$ en lugar de X, obtenemos las dos entradas requeridas para la red.

La RNA clasifica los datos de entrada entre diez posibles valores o clases del campo eléctrico, usando el esquema de clasificación del capítulo anterior con $E_{max} = 1$ y $E_{min} = 0.01$ y utilizando la Ec. (2.18) para definir los objetivos para diez clases diferentes ($C = 10$). Ya que la neurona de salida usa una función de activación lineal, los valores fuera del intervalo [0,1] son asociados con la primer o última clase respectivamente.
Una vez definidos los datos de entrada y los objetivos se usa un número total de pasos de $S = 10000$ y el parámetro de aprendizaje seleccionado es $\gamma = 0.0054$. Los valores de γ seleccionados para el entrenamiento, se encontraron manualmente y funcionan bien, pero con una búsqueda automatizada tal vez se puedan mejorar los resultados. El proceso de entrenamiento es repetido cinco veces para verificar el desempeño y evitar inicializaciones extrañas de los pesos y por último los pesos obtenidos del entrenamiento son usados para probar el desempeño para cada conjunto utilizando la Ec. (2.19).
Una vez que la red está completamente entrenada utilizando el conjunto de entrenamiento, usamos un conjunto de trayectorias generadas aleatoriamente con valores aleatorios de E entre [0,1] para verificar la capacidad de predicción de la RNA. El promedio sobre cinco diferentes conjuntos de predicción y la eficiencia de la RNA para todos los conjuntos se muestra en la Tabla 3.1.

Oscilaciones de Bloch para interacciones de 1V. a variable (E constante)

Repetimos exactamente el mismo procedimiento al descrito anteriormente, pero ahora fijando el valor del campo eléctrico $E = 1$ y permitiendo

Conjuntos	Entrenamiento	Validación	Prueba
E variable			
PPCC (%)	100.0	100.0	96.6
a variable			
PPCC (%)	92.8	90.0	95.3

Tabla 3.1: Porcentaje de patrones clasificados correctamente (PPCC) por la RNA para las señales generadas variando E entre [0,1] con a constante (arriba) y variando a entre [0.5,1] con E constante (abajo). Presentamos los resultados para los 70 patrones de entrenamiento, 30 de validación y el promedio de 5 conjuntos de predicción de 30 patrones cada uno.

variar al parámetro a entre 0.505 y 1. Cien trayectorias fueron generadas usando pasos de Δa=0.005.
Como antes, los pesos iniciales fueron generados aleatoriamente entre [-1,1] y el total de pasos en el entrenamiento fue de $S = 10000$ con un parámetro de aprendizaje $\gamma = 0.0055$. La eficiencia de la RNA para este escenario también es presentada en la Tabla 3.1.

Oscilaciones de Bloch para interacciones entre 2V

Cuando son tomadas en consideración las interacciones entre 2V, las señales son generadas usando la Ec. (3.13), variando simultáneamente a y E con los demás parámetros constantes (ω=1/2,ω'=1/4). En este caso, la red tiene dos neuronas en la capa de salida asociadas con los valores de a y E. El rango de parámetros estudiados en esta situación es $0.533 \leq a \leq 1$, con $\Delta E = 0.066$. Entonces, 225 señales fueron generadas (Figura 3.2), cada una de ellas con una duración temporal de 400 unidades.

Seis valores son guardados, tres de estos valores corresponden a las frecuencias (k_1,k_2,k_3) y los otros tres corresponden a sus amplitudes ($|f(k_1)|$, $|f(k_2)|$, $|f(k_3)|$). Como anteriormente, la Ec. (3.17) se usa para escalar los valores de las amplitudes y frecuencias. En lugar de usar 10 clases, ahora usamos 3 por cada parámetro, lo que da lugar a un total de 9 combinaciones de clases, donde usamos la Ec. (2.18) para definir los objetivos para cada

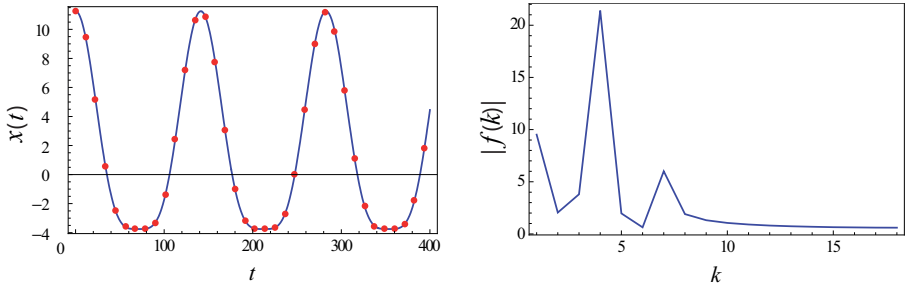

Figura 3.2: Izquierda: Trayectoria para 2V generada usando $w = 0.5, w' = 0.25$, $a = 0.66$, $E = 0.066$ correspondiente al patrón $p = 76$. Los puntos muestran las posiciones donde se calcula la DFFT. Derecha: Valor absoluto de la DFFT. Las frecuencias seleccionadas son $k_1^{76} = 1$, $k_2^{76} = 4$, $k_3^{76} = 7$ y $|f(2)|$, $|f(4)|$, $|f(7)|$ respectivamente, donde el superíndice indica la señal a la que pertenecen las frecuencias seleccionadas y el subíndice indica a cual de las tres frecuencias seleccionadas corresponde.

parámetro usando los índices $c = cl_a$ para a y $c = cl_E$ para E, siendo $C = 3$ el total de clases en cada parámetro. El número de clases fue seleccionado de tal manera que el tiempo de entrenamiento y predicción de la RNA sea similar al caso anterior.
Los parámetros para la fase de entrenamiento son pesos inicializados aleatoriamente entre [-1,1], $S = 20000$,$\gamma = 0.00022$ y las condiciones para decidir si la clasificación es correcta están dadas a través de la Ec. (2.19), usando una ecuación de esta forma para cada parámetro. La eficiencia de la red para este experimento se encuentra en la Tabla 3.2.

3.1.4. Conclusiones de la sección

En esta sección hemos llevado a cabo un análisis de OB en la cadena lineal monoatómica mediante un enfoque de amarre fuerte para el nivel de los primeros y segundos vecinos. Hemos desarrollado un procedimiento eficiente de aprendizaje automático usando una RNA para explorar el problema inverso de OB. Desarrollamos un proceso de entrenamiento para la RNA, tal que, de acuerdo a nuestras funciones de error, la precisión

Conjuntos	Entrenamiento		Validación		Prueba	
Salida	O_1	O_2	O_1	O_2	O_1	O_2
PPCC (%)	89.8	98.7	89.5	95.5	84.7	91.9

Tabla 3.2: Porcentaje de patrones clasificados correctamente (PPCC) por la RNA para las señales generadas con parámetros $\omega = 0.5$, $\omega' = 0.25$ y variando a entre [0.5,1] y E entre [0,1]. Presentamos los resultados para los 158 patrones de entrenamiento, 67 de validación y el promedio de 5 conjuntos de predicción de 67 patrones cada uno.

de las predicciones se optimiza para los parámetros internos de la RNA en una fase de validación. Después, la capacidad de predicción de la RNA se pone a prueba directamente de las curvas *experimentales*, obtenidas de integrar directamente las ecuaciones de movimiento (3.11) y (3.13) para cierta intensidad del campo eléctrico E y de la distancia interatómica a. Manteniendo a fijo, hemos sido capaces de predecir la intensidad del campo eléctrico para una muestra aleatoria de la trayectoria con 91 % de precisión para el nivel de los 1V. Se obtiene una precisión similar en las predicciones para E fijo, pero variando la distancia interatómica a. Cuando ambos parámetros variaron, en la aproximación para el nivel de 2V, nuestras predicciones son precisas hasta un 84 % para la distancia interatómica y de 93 % para la intensidad del campo eléctrico. Debido al desempeño de las RNA, estos hallazgos nos motivan a extender una estrategia similar para explorar el problema inverso de OB en sistemas más realistas, como los que se presentan en las siguientes secciones.

3.2. Oscilaciones de Bloch en una malla cuadrada

La presente sección esta basada en el artículo titulado *Bloch oscillations in two-dimensional crystals: Inverse problem* [44], el cual se realizó en conjunto con los Drs. Mauricio Carrillo, José A. González, Saúl Hernández y Alfredo Raya. Este trabajo analiza la extensión del estudio anterior a dos dimensiones, en donde además de estudiar un sistema un poco más com-

plejo, se usan un número de señales mayor y más datos de entrada para la RNA.

Los materiales 2D son en la actualidad una piedra angular de la física de estado sólido y la ciencia de materiales debido a su potencial tecnológico y su impacto en la investigación fundamental. Muchos de estos cristales 2D tienen la estructura cristalina de una malla cuadrada, que debido a su alta simetría, permite el estudio de fenómenos interesantes, como las oscilaciones de Bloch.

3.2.1. Oscilaciones de Bloch: enfoque semi-clásico

Empezamos nuestra discusión partiendo de un Hamiltoniano de amarre fuerte de una malla cuadrada monoatómica 2D de espaciado a. Considerando la aproximación de vecinos cercanos, tenemos

$$\begin{aligned} H\psi_{n,m}(k) &= -t\psi_{n+1,m}(k) - t\psi_{n-1,m}(k) \\ &\quad -t\psi_{n,m+1}(k) - t\psi_{n,m-1}(k) + \epsilon_0\psi_{n,m}(k) \\ &\equiv \epsilon^{n,m}(k)\psi_{n,m}(k), \end{aligned} \tag{3.18}$$

donde t es el parámetro de *hopping* y $k = k_1\hat{e}_x + k_2\hat{e}_y$ es el momento de los electrones en los cristales 2D. Del teorema de Bloch, es sencillo encontrar que la relación de dispersión energía-momento es:

$$\epsilon^{n,m}(k_1, k_2) = \epsilon_0 - \epsilon^{n,m}(k_1, k_2), \tag{3.19}$$

donde

$$\epsilon^{n,m}(k_1, k_2) = \omega(1 - cos(k_1 a) - cos(k_2 a)), \tag{3.20}$$

ϵ_0 es la energía en el sitio y $\omega = 2t$. Después, recordamos las ecuaciones semiclásicas de movimiento para un electrón moviéndose en un campo eléctrico externo $\mathbf{E}$ orientado paralelo a una dirección de la red cuadrada,

$$\frac{d\mathbf{k}}{dt} = -e\mathbf{E}\,, \tag{3.21}$$

$$\frac{d\mathbf{r}}{dt} = \frac{1}{\hbar}\frac{\partial}{\partial\mathbf{k}}\epsilon^{(n,m)}(k_1, k_2)\,. \tag{3.22}$$

Podemos sencillamente integrar las ecuaciones de movimiento y obtener las velocidades y trayectorias para un campo eléctrico dado. Considerando la red orientada a lo largo del plano $x - y$ y un campo eléctrico uniforme $\mathbf{E} = E_1\hat{e}_x + E_2\hat{e}_y$, integramos la Ec. (3.21) asumiendo la condición inicial $k_j(0) = 0$ con $j = 1, 2$. Entonces

$$k_j(t) = -\frac{eE_j}{\hbar}t. \tag{3.23}$$

Reescribiendo la Ec. (3.22), la velocidad del electrón está dada por:

$$\begin{aligned} v_j^{(n,m)}(k_j(t)) &= \frac{wa}{\hbar}\sin(k_j(t)a), \\ &= -\frac{wa}{\hbar}\sin\left(\frac{eE_j a}{\hbar}t\right), \end{aligned} \tag{3.24}$$

y la corriente eléctrica es simplemente $j_i = -ev_i$. Integrando las Ecs. (3.24) obtenemos el perfil de la OB para la posición de los electrones como función del tiempo:

$$\begin{aligned} x_j^{(n,m)}(t) &= \frac{w}{eE_j}\cos\left(\frac{eE_j}{\hbar}at\right), \\ &= \frac{w}{eE_j}\cos(\omega_{E_j}t), \end{aligned} \tag{3.25}$$

con $\omega_{E_j} = eE_j a/\hbar$. Las Ecs. (3.25) describen las trayectorias de los electrones expuestos al campo eléctrico E dentro de la red bidimensional.

Enseguida, describimos como las Ecs. (3.24) son simuladas y como son procesadas por una RNA para dar un resultado preciso.

3.2.2. Creación y procesamiento de señales

Para unos parámetros fijos a y t, las trayectorias descritas por las Ecs. (3.24) y (3.25) son funciones de la intensidad del campo eléctrico a lo largo de cada dirección espacial, los cuales se convierten en los únicos parámetros libres que caracterizan una trayectoria dada empleando nuestras consideraciones. Hemos entrenado una RNA que asocia las corrientes eléctricas de los electrones con sus correspondientes campos eléctricos. En otras palabras, la

RNA aprende a través de algunos ejemplos la relación entre las señales de corriente eléctrica en la red cuadrada y los campos eléctricos que generaron esas corrientes. Primero, describiremos como se generan las señales usadas para el entrenamiento y luego explicamos el proceso de clasificación.
Por simplicidad y sin pérdida de generalidad, todas las señales fueron creadas siguiendo las siguientes consideraciones:

- Los parámetros en las Ecs. (3.24) y (3.25) fueron usados con unidades adimensionales y usando $w_2 = w_2 = a = 0.5$.
- Las señales fueron generadas para un lapso $\tau = 200$.
- Integramos las señales considerando la posibilidad de un campo eléctrico positivo o negativo para E_1 y E_2 en tres diferentes rangos definidos por $E_{\min}$ y $E_{\max}$. Estos casos serán descritos con más detalle en la sección 3.2.2.

Una vez que se producen las señales, seleccionamos como entradas para la RNA valores para cada componente de la velocidad (v_1 y v_2) en cien diferentes instantes definidos por $t_i = i\Delta t$, con $\Delta t = \tau/100 = 2$ y $i = 0, 1, \ldots, 99$. Esto significa que la RNA analizará una señal V que consiste de doscientos valores:

$$V = \{v_1(t_1), v_2(t_1), \ldots, v_1(t_n), v_2(t_n)\}. \tag{3.26}$$

En la Figura 3.3 mostramos un ejemplo de las velocidades en una OB y sus correspondientes valores, donde las trayectorias fueron evaluadas con $E_1 = -0.22$ y $E_2 = 0.14$ usando la Ec. (3.24).

Como el objetivo es clasificar el campo eléctrico en 2D, imponemos que la RNA con propagación hacia adelante tenga dos salidas $\tilde{E}_1$ y $\tilde{E}_2$. Note la diferencia entre $\tilde{E}_i$, como el valor predicho y E_i como el valor físico. Considerando una capa oculta con 27 neuronas, la ecuación que define el valor predicho dada una señal de entrada V, está definida por:

$$\tilde{E}_j = F\left(\sum_{h=1}^{27} \tilde{\sigma}_{hj} F\left(\sum_{i=1}^{200} \sigma_{ih} V_i\right)\right), \tag{3.27}$$

donde $j = 1, 2$. F es la función de activación para las capas oculta y de salida; en este caso se uso la función logística sigmoidea estándar; σ_{ih} y $\tilde{\sigma}_{hj}$

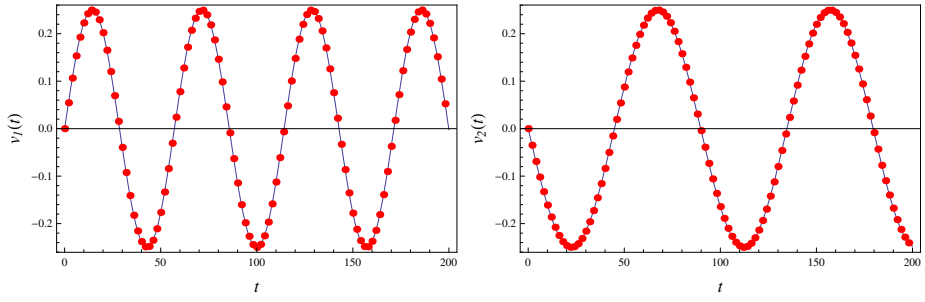

Figura 3.3: Velocidades de los electrones oscilando simuladas usando la Ec. (3.24). Los puntos muestran los valores usados como datos de entrada para la RNA. Izquierda: velocidad $v_1(t)$ para $E_1 = -0.22$. Derecha: velocidad $v_2(t)$ para $E_2 = 0.14$.

son los pesos entre la capa de entrada y oculta y entre la capa oculta y la de salida respectivamente. La estructura de la RNA está ilustrada en la Figura 3.4.

Escenarios para el campo eléctrico

La precisión de la RNA depende de la frecuencia de las señales, el campo eléctrico y los puntos de muestreo. En esta sección, analizamos como se comporta el desempeño de la RNA en tres distintos escenarios. Usando 625 señales con todos los parámetros fijos, excepto por el campo eléctrico que cambia de acuerdo al escenario:

1. Entre $[E_{\min} = -0.5, E_{\max} = 0.46]$ separados en pasos $\Delta E = 0.04$.
2. Entre $[E_{\min} = -1, E_{\max} = 0.92]$ con $\Delta E_j = 0.08$.
3. Entre $[E_{\min} = -0.25, E_{\max} = 0.23]$ con $\Delta E_j = 0.02$.

Considerando que la función de activación F usada en la Ec. (3.27) es una función sigmoide, la salida de la red estará en el rango $[0, 1]$. Las salidas de la RNA pueden ser divididas en clases que representan los intervalos en los objetivos para E_1 y E_2. Esto quiere decir que entre más clases tiene una salida, se requiere mas precisión para una clasificación correcta. Para

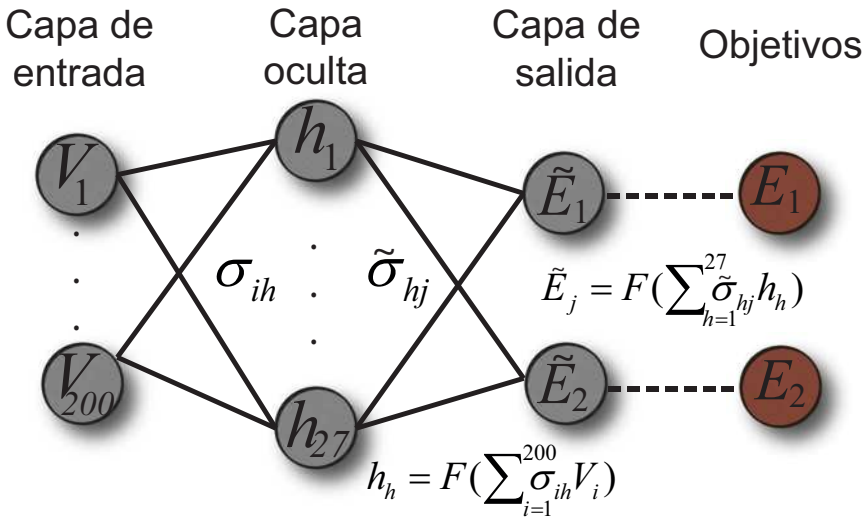

Figura 3.4: Estructura de la RNA desarrollada para la clasificación de las OB. La capa de entrada consiste en 200 neuronas de acuerdo a los valores extraídos de las señales correspondientes a las OB. La capa de entrada está conectada a una capa oculta con 27 neuronas con los pesos σ_{ih}. Cada neurona oculta calcula un valor usando la función de activación sigmoidea F. Posteriormente, estos valores son enviados a las dos neuronas en la capa de salida mediante los pesos $\tilde{\sigma}_{hj}$ donde se utiliza también una función sigmoidea. Finalmente, se mide la diferencia entre las salidas de la RNA y los objetivos propuestos asociados al campo eléctrico usado en la OB. Con esta diferencia se construye la función de costo y se minimiza.

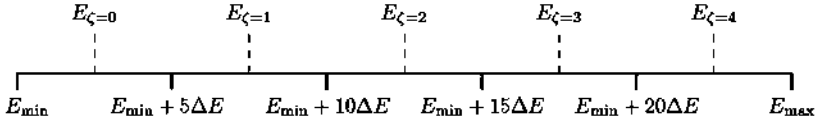

Figura 3.5: Representación esquemática de la definición de clases para los rangos del campo eléctrico. También, el centro de cada región E_ζ es equivalente a la clase objetivo $\hat{E}_\zeta$, que será usada durante la fase de entrenamiento. La representación muestra sólo una dirección del campo eléctrico, pero está hecho de la misma manera para la otra dirección.

este caso, hemos decidido dividir cada salida en 5 clases. Para aclarar, desarrollamos el caso (1) donde $\Delta E_j = 1/25$ y $E_{\min} = -0.5$ y $E_{\max} = 0.46$. Utilizando el enfoque de salidas multiclase para la selección de las clases usando cada E_j, se presenta un esquema en la Figura 3.5 de la selección de clases de acuerdo al rango del campo eléctrico en donde se encuentra la señal. En este esquema se utiliza la etiqueta cero para la primer clase, por lo que al especificar los objetivos usando la Ec. (2.18) se utiliza $\zeta = c - 1$ donde E_ζ etiqueta cada clase para cualquier señal E_j.

3.2.3. Resultados

En todos los resultados presentados a continuación, hemos entrenado cinco diferentes redes usando distintos pesos iniciales y reportando sólo el mejor de todos los casos descritos en la sección pasada.
Para el primer caso (1) la RNA ha clasificado correctamente con un récord perfecto en el conjunto de entrenamiento, mientras que con el conjunto de prueba alcanzó 82 % y 91 % de eficiencia en $\tilde{E}_1$ y $\tilde{E}_2$ respectivamente, como se observa en la Tabla 3.3. En este caso, los rangos extremos para E son más anchos y de acuerdo a la Ec. (3.25), la frecuencia máxima de las señales es dos veces más que en el caso (1). Entonces, las señales tienen una mayor frecuencia, así que en principio deberíamos requerir una cantidad mayor de puntos o un intervalo temporal más pequeño para caracterizar apropiadamente estas señales, antes de ser introducidas en la RNA. Consideramos

PPCC (%)	Entrenamiento		Validación		Prueba	
Salida	$\tilde{E}_1$	$\tilde{E}_2$	$\tilde{E}_1$	$\tilde{E}_2$	$\tilde{E}_1$	$\tilde{E}_2$
Caso (i)	100	99	90	86	82	91
Caso (ii)	99	99	90	85	42	40
Caso (iii)	100	100	98	97	96	93

Tabla 3.3: Porcentaje de patrones clasificados correctamente por la RNA para las señales creadas variando: Caso (i); E_1 entre $[-0.5, 0.46]$ y E_2 entre $[-0.5, 0.46]$, Caso (ii); E_1 entre $[-1, 0.92]$ y E_2 entre $[-1, 0.92]$, Caso (iii); E_1 entre $[-0.25, 0.23]$ y E_2 entre $[-0.25, 0.23]$. Usamos 438 patrones de entrenamiento, 187 de validación y el promedio de 5 conjuntos de prueba de 187 patrones cada uno.

que la baja eficiencia de la red se debe a este hecho.

Los resultados para el caso número (2), donde los valores extremos de E_j estuvieron en el rango $[-1, 0.92]$, son menos precisos, porque las salidas solo alcanzaron una precisión de $\tilde{E}_1 = 42\,\%$ y $\tilde{E}_2 = 40\,\%$ en el conjunto de prueba como se puede observar en la Tabla 3.3. Finalmente, para el caso (3) entre el intervalo $[E_{\min} = -0.25, E_{\max} = 0.23]$ y $\Delta E = 1/50$, las curvas generadas tienen una menor frecuencia que en el caso (1), por lo que los puntos muestreados tienen mayor información acerca de la señal, permitiendo a la RNA superar los casos previos, alcanzando una eficiencia de 96 % y 93 % para E_1 y E_2 respectivamente como se muestra en la Tabla 3.3. Un ejemplo de la predicción usando las señales de OB con un campo eléctrico compuesto por $E_1 = -0.2046$ y $E_2 = 0.1969$, se puede ver en la Figura 3.6. En este caso la RNA estima, que las señales fueron generadas con valores de E_1 y E_2 perteneciendo a las clases $E_{\zeta=0}$ y $E_{\zeta=4}$ respectivamente, lo cual corresponde a una clasificación correcta.

3.2.4. Observaciones finales

Hemos desarrollado un método empleando un enfoque de RNA para analizar oscilaciones de Bloch en una malla 2D cuadrada de átomos con una aproximación de amarre fuerte considerando la influencia de los vecinos

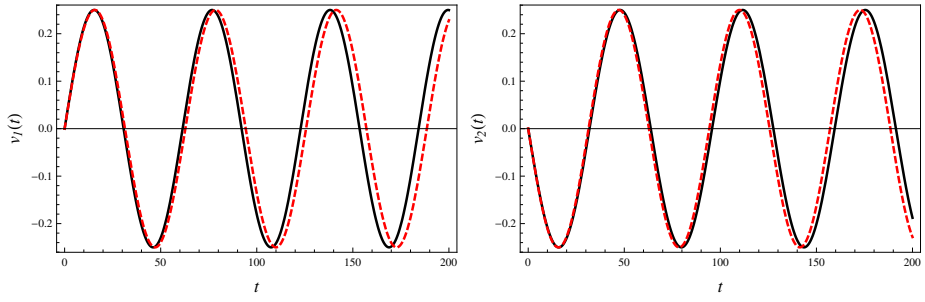

Figura 3.6: Gráfica de $v_1(t)$ y $v_2(t)$, para las oscilaciones de los electrones generadas con valores aleatorios de E_1 y E_2 entre [-0.25,0.23]. Las líneas continuas representan la OB generada usando el valor real del campo eléctrico E_1 y E_2, mientras que las lineas discontinuas son las señales de OB generadas usando el valor central de los campos eléctricos de las clases predichas $\tilde{E}_1$ y $\tilde{E}_2$, como se describió en la sección 3.2.2. A la izquierda esta graficada $v_1(t)$ para el valor real del campo eléctrico $E_1 = -0.2046$ y el valor central de la clase 0, es decir, $E_{\zeta=0} \equiv E_1 = -0.20$. A la derecha esta graficada $v_2(t)$ para $E_2 = 0.1969$ y el valor central $E_{\zeta=4} \equiv E_2 = 0.2$.

cercanos. La RNA considerada usa las velocidades (corriente eléctrica) de las oscilaciones como señales de entrada y estima la correspondiente fuerza del campo eléctrico proyectada a lo largo de cada dirección espacial. Para el propósito de este trabajo, se estudian tres escenarios diferentes donde el máximo y el mínimo de los campos eléctricos se encuentran restringidos. Los rangos extremos del campo eléctrico determinan la frecuencia del electrón y a su vez, el número de puntos muestreados por ciclo, lo cual impacta en el desempeño de las RNAs. Las RNAs fueron entrenadas y validadas con 625 señales entre estos rangos, mientras que fueron probadas para señales con campos eléctricos aleatorios en los mismos intervalos. En el mejor caso, para baja frecuencia, la RNA alcanza por lo menos 93 % de precisión en cada salida en el conjunto de prueba. Como se mencionó anteriormente, esto es debido al intervalo menor del campo eléctrico, en donde las curvas generadas oscilan menos y entonces la curvas son descritas mejor. Por otro lado, para el intervalo mayor del campo eléctrico las predicciones son menos precisas, porque las curvas requieren más puntos

para ser descritas adecuadamente.

De nuestro trabajo anterior [42] y los resultados aquí expuestos, es fácil ver que este enfoque tiene un buen potencial y nos motiva a explorar sistemas más complejos, en donde se presentan las OB, como es el caso del grafeno que se estudia a continuación.

3.3. Oscilaciones de Bloch en grafeno

El problema descrito en esta sección está basado en el artículo titulado *Bloch oscillations in graphene from an artificial neural network study* [45], el cual fue desarrollado con los Drs. Mauricio Carrillo, José A. González, Saúl Hernández y Alfredo Raya. Comparado con los trabajos anteriores de este capítulo, además de que las ecuaciones relacionadas a las OB en este caso son más complejas, nuevamente el estudio realizado utiliza un mayor número de simulaciones.

El grafeno posee un gran número de propiedades excepcionales, que van desde tremendamente altas conductividades eléctricas y térmicas, transparencia de membranas y encima de eso, rigidez y flexibilidad [46, 47, 48, 49]. Por lo tanto, la manipulación de propiedades eléctricas a través de medios mecánicos ha dado lugar al campo de *straintronics* [50] en grafeno y otros materiales (ver [51] para una revisión reciente). Debido a todas estas propiedades interesantes es que el estudio de grafeno últimamente ha tenido un auge.

3.3.1. Oscilaciones de Bloch en grafeno: Enfoque semiclásico

La estructura cristalina del grafeno (y otros materiales de Dirac-Weyl) consiste en un arreglo de átomos de carbono (u otros) con el grosor de un átomo apretujados en una red en forma de panal de abeja como se muestra en la Figura 3.7. En el espacio real, el arreglo hexagonal es mejor descrito en términos de dos subredes triangulares con vectores primitivos

$$\mathbf{a}_1 = \frac{\sqrt{3}}{2}a\hat{e}_x + \frac{3}{2}a\hat{e}_y\ , \qquad \mathbf{a}_2 = -\frac{\sqrt{3}}{2}a\hat{e}_x + \frac{3}{2}a\hat{e}_y\ , \tag{3.28}$$

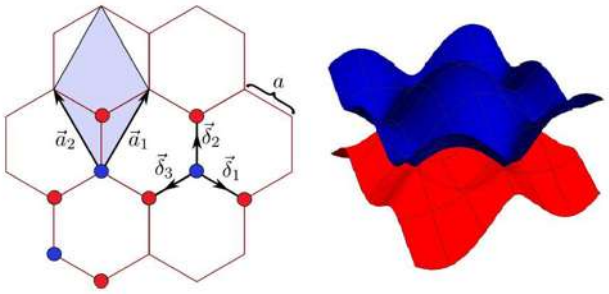

Figura 3.7: Izquierda: estructura cristalina del grafeno. Vectores primitivos $\mathbf{a}_{1,2}$ y vectores conectando cada átomo a su vecino más cercano $\boldsymbol{\delta}_{1,2,3}$ son mostrados. Derecha: Relación Energía-momento en la primer zona de Brillouin. Se encuentra un comportamiento lineal cerca de los llamados puntos de Dirac.

donde $\hat{e}_{x,y}$ son los vectores unitarios a lo largo de la membrana de grafeno en el plano, con una distancia interatómica $a \simeq 1.42$ Å. Cada átomo de una subred dada, está conectado a sus vecinos cercanos, los cuales a su vez pertenecen a la otra subred, a través de los vectores

$$\boldsymbol{\delta}_1 = \frac{\sqrt{3}}{2} a\hat{e}_x + \frac{a}{2}\hat{e}_y \ , \qquad \boldsymbol{\delta}_2 = -\frac{\sqrt{3}}{2} a\hat{e}_x + \frac{a}{2}\hat{e}_y \ , \qquad \boldsymbol{\delta}_3 = -a\hat{e}_y \ . \quad (3.29)$$

Correspondientemente, el Hamiltoniano de amarre fuerte es de la forma

$$H(\mathbf{k}) = \begin{pmatrix} 0 & \tau\varepsilon(\mathbf{k}) \\ \tau\varepsilon^*(\mathbf{k}) & 0 \end{pmatrix} \ , \quad (3.30)$$

donde $\mathbf{k} = (k_x, k_y)$ es el vector de momento del electrón en el cristal, τ es el parámetro de *hopping*

$$\varepsilon(\mathbf{k}) = \sum_{\boldsymbol{\delta}} e^{i\mathbf{k}\cdot\boldsymbol{\delta}} = 2ie^{\frac{i}{2}k_x a} \sin\left(\frac{\sqrt{3}}{2} a k_x\right) + e^{-iak_x} \ . \quad (3.31)$$

Entonces, la relación de dispersión de energía-momento es $\epsilon(\mathbf{k}) = \pm\tau|\varepsilon(\mathbf{k})|$, la cual puede escribirse convenientemente como

$$\varepsilon(\mathbf{k}) = \pm\sqrt{5 + 4\cos\left(\frac{3}{2} a k_x\right)\cos\left(\frac{\sqrt{3}}{2} a k_y\right) - 4\sin^2\left(\frac{\sqrt{3}}{2} a k_y\right)} \ . \quad (3.32)$$

Como explicamos anteriormente, muchas características de las oscilaciones de Bloch (OB) en grafeno y otros semimetales de Dirac-Weyl con una red hexagonal subyacente, como la que se muestra en la Figura 3.7, pueden ser derivadas de esta relación de dispersión.
Con un enfoque semiclásico, empezamos considerando un campo elétrico estático y uniforme $\mathbf{E} = E_x\hat{e}_x + E_y\hat{e}_y$ orientado a lo largo del plano de grafeno. Las OB son descritas de acuerdo a las ecuaciones de movimiento

$$\frac{d\mathbf{k}}{dt} = -e\mathbf{E}\,, \tag{3.33}$$

$$\frac{d\mathbf{r}}{dt} = \frac{\partial\epsilon(\mathbf{k})}{\partial\mathbf{k}}\,, \tag{3.34}$$

donde e es la carga de la cuasi-partícula, $\mathbf{r} = (x, y)$ es el vector de posición, $\epsilon(\mathbf{k})$ es la relación de dispersión para la red tipo panal de abeja descrita por la Ec. (3.32) y asumimos $\hbar = 1$. Combinando las Ecs. (3.32) y (3.34), obtenemos directamente las velocidades semi-clásicas a lo largo de cada dirección

$$\begin{aligned} v_x &= -\frac{\sqrt{3}\tau a\left[\cos\left(\frac{3}{2}ak_x\right)\sin\left(\frac{\sqrt{3}}{2}ak_y\right) + 2\sin\left(\frac{\sqrt{3}}{2}ak_y\right)\cos\left(\frac{\sqrt{3}}{2}ak_y\right)\right]}{\varepsilon(\mathbf{k})}, \\ v_y &= -\frac{3a\tau\cos\left(\frac{\sqrt{3}}{2}ak_y\right)\sin\left(\frac{3}{2}ak_x\right)}{\varepsilon(\mathbf{k})}. \end{aligned} \tag{3.35}$$

Además, integrando la Ec. (3.33) tenemos

$$k_x(t) = k_x(0) - eE_xt\,, \qquad k_y(t) = k_y(0) - eE_yt\,, \tag{3.36}$$

donde $k_{x,y}(0)$ son las componentes del vector de onda inicial, es decir, $\mathbf{k}_0 = (k_x(0), k_y(0))$. En nuestra discusión, establecemos $\mathbf{k}_0$ considerando tres escenarios diferentes con la intención de compararlos con los resultados mostrados en [52]:

I. El campo eléctrico $\mathbf{E} = E_y\hat{e}_y$ y $\mathbf{k}_0 = (0, (\pi/6)(2/\sqrt{3}a))$.

II. El campo eléctrico $\mathbf{E} = E_x\hat{e}_x$ y $\mathbf{k}_0 = (0, (\pi/4)(2/\sqrt{3}a))$.

III. El campo eléctrico $\mathbf{E} = E_x\hat{e}_x + E_y\hat{e}_y$ y $\mathbf{k}_0 = 0$.

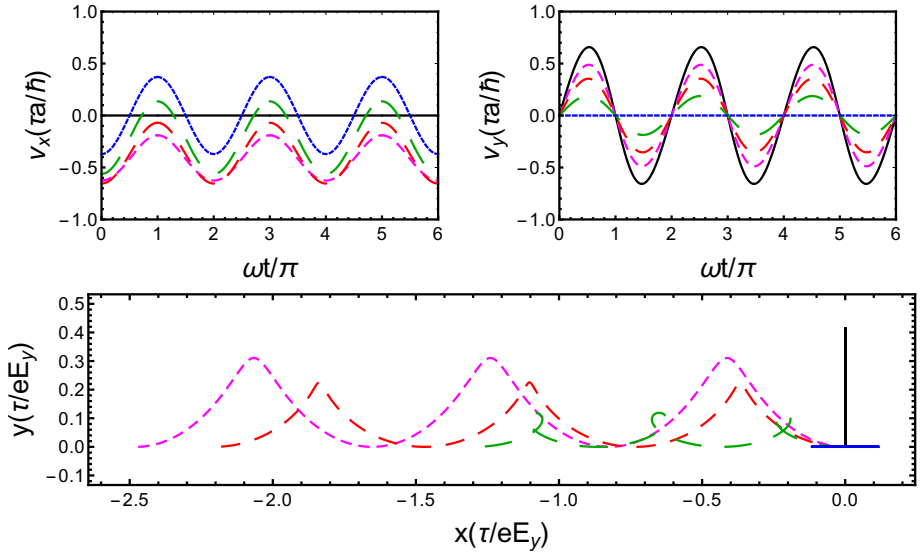

Figura 3.8: La dependencia temporal de v_x, v_y y las trayectorias de el electrón en la hoja de grafeno para diferentes k_x a $\frac{3}{2}ak_y = 0$ (línea continua obscura), $\frac{3}{2}ak_y = \pi/4$ (línea magenta discontinua), $\frac{3}{2}ak_y = \pi/3$ (línea roja discontinua), $\frac{3}{2}ak_y = 5\pi/12$ (línea verde punteada) y $\frac{3}{2}ak_y = \pi/2$ (línea azul continua). El campo eléctrico $\mathbf{E}$ está a lo largo de la dirección y.

En cualquier caso, insertando $k_{x,y}(t)$ de la Ec. (3.36) en las velocidades semi-clásicas (3.35) e integrando con respecto al tiempo, obtenemos las trayectorias para las OB en grafeno. Estas trayectorias ya no están expresadas en una forma cerrada y las integrales involucradas deben realizarse numéricamente. Expresiones similares fueron discutidas en [52], aunque los errores tipográficos en ese trabajo fueron corregidos en nuestro trabajo (ver Figuras 3.8 y 3.9). Generamos curvas correspondientes a las velocidades semi-clásicas para configurar la RNA, como se discute a continuación.

3.3.2. Creación de señales y procesado de datos

Usando la Ec. (3.35), hemos simulado OB variando el campo eléctrico aplicado a la estructura cristalina, considerando $\hbar = e = 1$ y $\tau = a = 1/2$. Entonces, el resultado es una serie de tiempo para ambas componentes de

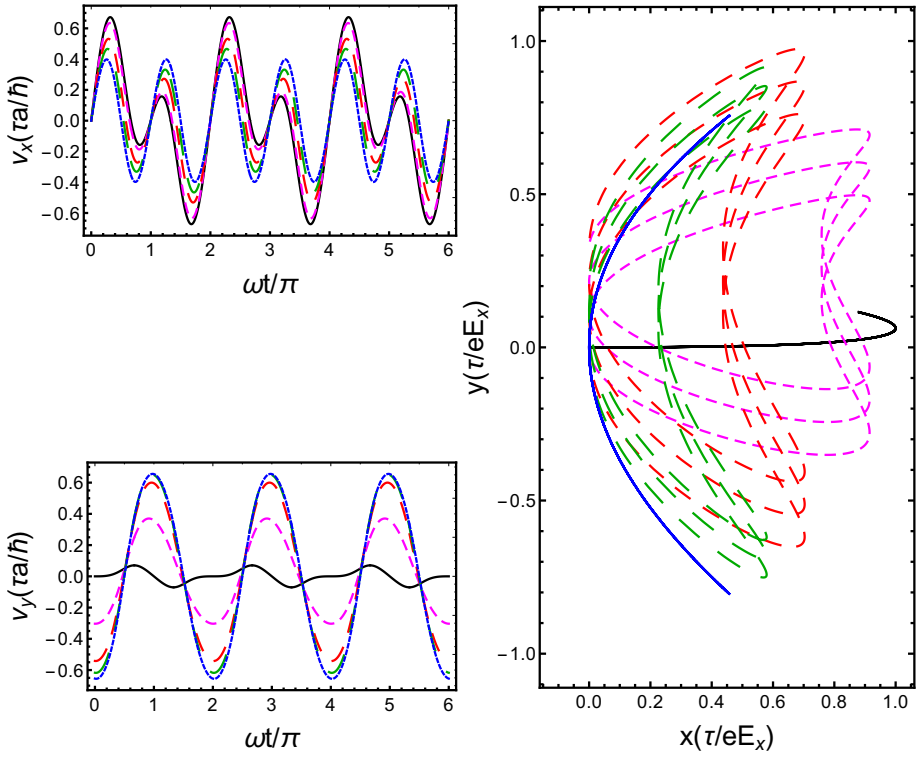

Figura 3.9: La dependencia temporal de v_x, v_y y las trayectorias de el electrón en la hoja de grafeno para diferentes k_y a $\frac{\sqrt{3}}{2}ak_x = 0$ (línea continua obscura), $\frac{\sqrt{3}}{2}ak_x = \pi/6$ (línea magenta discontinua), $\frac{\sqrt{3}}{2}ak_x = \pi/3$ (línea roja discontinua), $\frac{\sqrt{3}}{2}ak_x = 5\pi/12$ (línea verde punteada) y $\frac{\sqrt{3}}{2}ak_x = \pi/2$ (línea azul continua). El campo eléctrico **E** está a lo largo de la dirección x.

la velocidad del electrón (v_x and v_y). El lapso de las señales depende de su frecuencia, ya que para frecuencias más altas, se requiere un mayor muestreo para describir las OB apropiadamente, en términos de la precisión y de los recursos computacionales disponibles, como analizamos previamente en [44]. Con esto en mente, ambas series de tiempo (v_x y v_y) han sido discretizadas en 100 valores cada una y sirven como datos de entrada para la RNA. Esto significa, que en general, un vector de entrada tiene 200 elementos descritos por:

$$\mathbf{I} = (v_x(t_0), v_y(t_0), \ldots, v_x(t_{99}), v_y(t_{99}))\,. \tag{3.37}$$

En cada escenario, la RNA ha sido entrenada con un algoritmo de aprendizaje supervizado, lo cual quiere decir que necesitamos especificar los objetivos correspondientes, o en otras palabras, el campo eléctrico empleado para crear las señales. Similarmente al trabajo realizado en [44], la RNA trabaja como un clasificador. Esto significa que las salidas de la RNA están asociadas con clases, definiendo diferentes rangos del campo eléctrico usados para crear las OB. En otras palabras, cada vector de entrada es un patrón a clasificar en un rango del campo eléctrico, que pertenece a una clase especifica. A continuación, los rangos del campo eléctrico de estas clases son especificados para cada caso de estudio.

Una vez que los datos de entrada y los objetivos para cada señal han sido especificados, la RNA es entrenada usando esta información. Los patrones de entrenamiento y validación fueron seleccionados aleatoriamente del conjunto total de señales, donde el número de señales en cada conjunto depende del caso de estudio, como está especificado en la Tabla 3.4. Adicionalmente, evaluamos el desempeño de la RNA usando un conjunto de prueba con el mismo número de señales que el conjunto de validación, en donde los patrones fueron construidos usando valores aleatorios de el campo eléctrico dentro de las clases predefinidas. Dado que la estructura de la RNA depende del caso, discutimos cada escenario por separado.

Consideraciones de la RNA y definición de objetivos

Usamos una RNA con propagación hacia adelante entrenada con un algoritmo supervisado tipo *backpropagation* [9]. El algoritmo minimiza una

	Caso I	Caso II	Caso III
Conjunto de entrenamiento	350	350	1750
Conjunto de validación	150	150	750
Conjunto de prueba	150	150	750
Neuronas ocultas	25	45	20
Neuronas de salida	1	1	2
Parámetro de aprendizaje	10^{-2}	$7\text{x}10^{-3}$	$5\text{x}10^{-3}$
Iteraciones	$5\text{x}10^{4}$	$5\text{x}10^{4}$	10^{4}

Tabla 3.4: Número de patrones en cada conjunto y los parámetros usados en la RNA para cada caso de estudio.

función de error cuadrático medio usando una técnica de gradiente descendiente. Al inicio del proceso de aprendizaje, los pesos son inicializados aleatoriamente entre [-1,1] y es seleccionado el número de iteraciones para el entrenamiento tomando en cuenta que si el error de validación aumenta, el entrenamiento termina. El error de validación se obtiene también usando una función de error cuadrático medio pero calculada sobre los patrones del conjunto de validación.

Caso I.

Como se refiere en la Tabla 3.4, en este caso 500 simulaciones fueron creadas con el propósito de entrenar la RNA y validar su entrenamiento. Estas señales fueron generadas variando los valores de E_y desde el valor

$$E_{y_{min}} = -\pi/(4\sqrt{3}) + \pi/(4\sqrt{3} * 500), \tag{3.38}$$

hasta el máximo valor de

$$E_{y_{max}} = \pi/(4\sqrt{3}) - \pi/(4\sqrt{3} * 500), \tag{3.39}$$

con pasos ΔE_y de

$$\Delta E_y = \pi/(2\sqrt{3} * 500). \tag{3.40}$$

Dado que el valor de E_x es igual a 0, todas las series de tiempo respecto a v_x son las mismas para este escenario. Sin embargo, hemos escogido incluirlas como entrada para la RNA, pensando en general para el caso (III).

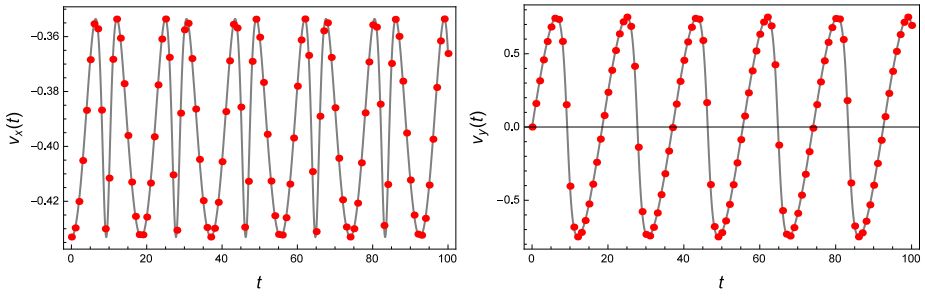

Figura 3.10: Muestra de las velocidades de los electrones oscilando generados usando las consideraciones para el caso I. Los puntos muestran los valores usados como entradas para la RNA. Izquierda: velocidad $v_x(t)$ para $E_x = 0$. Derecha: velocidad $v_y(t)$ para $E_y = E_{y_{min}}$.

Un ejemplo de la OB para este caso se muestra en la Figura 3.10.
Con estas consideraciones, construimos una RNA con una sola salida con diferentes rangos de valores de E_y, separando el total de señales en subgrupos o clases. Para este caso, consideramos 100 diferentes clases para E_y. Dado que hemos generado 500 señales de ejemplo, cada clase consta de cinco señales, en términos de los conjuntos de entrenamiento y de validación. Se definen los objetivos mediante el esquema de salidas multiclase, así como el valor promedio para el parámetro predicho.

Una vez que el entrenamiento está completo, procedemos a crear un conjunto de prueba con la misma cantidad de patrones que el conjunto de validación, considerando valores aleatorios para E_y entre $[-\pi/(4\sqrt{3}), \pi/(4\sqrt{3})]$ y el desempeño de la red es medido dependiendo del número de patrones que son clasificados correctamente.

Caso II.

El análisis de señales es análogo al caso previo, pero ahora $E_y = 0$ y E_x se define entre el intervalo desde

$$E_{x_{min}} = -\pi/(4\sqrt{3}) + \pi/(4\sqrt{3} * 500), \tag{3.41}$$

hasta

$$E_{x_{max}} = \pi/(4\sqrt{3}) - \pi/(4\sqrt{3} * 500), \tag{3.42}$$

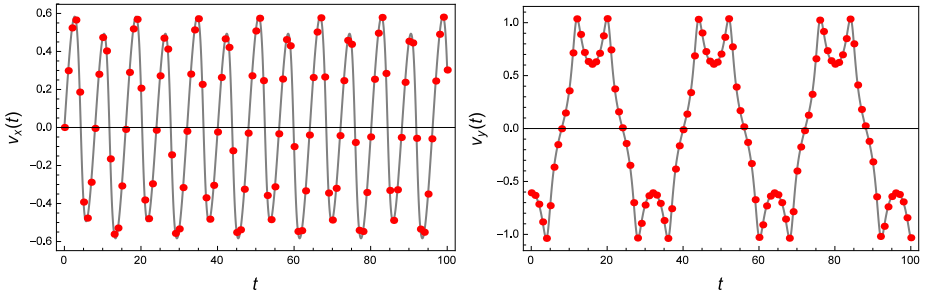

Figura 3.11: Muestra de las velocidades de los electrones oscilando, generados usando las consideraciones para el caso II. Los puntos muestran los valores usados como entradas para la RNA. Izquierda: velocidad $v_x(t)$ para $E_x = E_{x_{min}}$. Derecha: velocidad $v_y(t)$ para $E_y = 0$.

en pasos de

$$\Delta E_x = \pi/(2\sqrt{3} * 500). \tag{3.43}$$

Una muestra de las series de tiempo para v_x y v_y en este caso se presenta en la Figuras 3.11. La definición de los objetivos y el valor promedio del campo para un total de cien clases son los mismos que en el caso I, de acuerdo a las Ecs. (2.18) y (2.20).

Caso III.

Este es el escenario más general, donde ambas componentes del campo eléctrico fueron generadas en pasos de

$$\Delta E_x = \Delta E_y = \pi/(2\sqrt{3} * 50) \tag{3.44}$$

en el intervalo

$$E_x, E_y \in [-\pi/(4\sqrt{3}) + \pi/(4\sqrt{3} * 50), \pi/(4\sqrt{3}) - \pi/(4\sqrt{3} * 50)]. \tag{3.45}$$

Con estas ecuaciones, 2500 OB fueron generadas, definiendo 50 distintos valores para cada componente del campo eléctrico. En este caso la red tiene 20 neuronas en la capa oculta y dos neuronas de salida. Las dos salidas están relacionadas a cada componente del campo eléctrico: la primera se relaciona a E_x y la segunda a E_y.

El procedimiento para extraer los datos de las señales creadas es el mismo que en los primeros dos casos (ver Figura 3.12). Identificamos un rango de valores para el campo eléctrico con una clase de acuerdo a la Ec. (2.20), teniendo 10 clases para cada componente del campo eléctrico y por lo tanto diez posibles objetivos para cada salida, especificados a través de la Ec. (2.18). En este caso, el error asociado a cada componente del campo eléctrico predicho es de $\pm 5\,\%$.

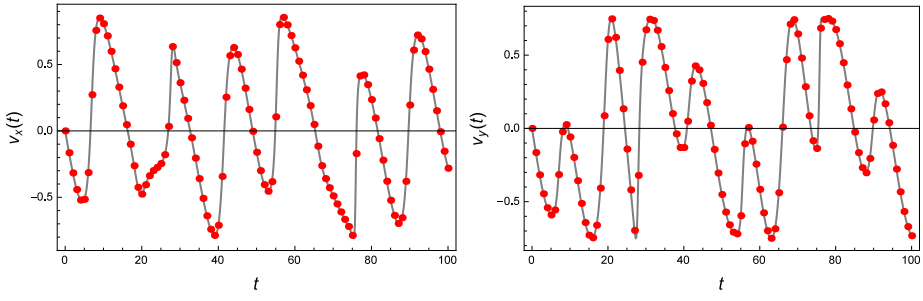

Figura 3.12: Muestra de las velocidades de los electrones oscilando generadas usando las consideraciones para el caso III. Los puntos muestran los valores usados como entradas para la RNA. Izquierda: velocidad $v_x(t)$ para $E_x = -\pi/(4\sqrt{3}) + \pi/(4\sqrt{3} * 50)$. Derecha: velocidad $v_y(t)$ para $E_y = -\pi/(4\sqrt{3}) + \pi/(4\sqrt{3} * 50)$.

Finalmente para cada patrón, se especifica que las salidas son clasificadas correctamente por la RNA si la condición de la Ec. (2.19) se cumple para cada salida respectivamente. Debajo se encuentran los resultados del entrenamiento para cada caso.

3.3.3. Resultados

Como se fija en la Ec. (3.37), en los tres casos la RNA consiste de una capa de entrada con 200 neuronas y una capa de salida como fue definida en cada uno de los casos. Sin embargo, el número de neuronas ocultas fue seleccionado basados en una configuración de la RNA que llevó al mejor desempeño en la misma. Esto es, la RNA con el mínimo error en el conjunto de validación durante la fase de aprendizaje. Todas las neuronas en

las capas oculta y de salida tienen una función de activación sigmoide.
La selección de la mejor RNA fue hecha variando el número de neuronas de cinco a cincuenta en pasos de cinco. Igualmente, el parámetro de aprendizaje fue escogido explorando valores entre el rango $[10^{-3}, 10^{-2}]$ en pasos de 10^{-3}. Las especificaciones de la RNA respecto a cada caso de estudio se encuentran en la Tabla 3.4. El desempeño de la mejor RNA para cada caso está reportado en la Tabla 3.5 con el porcentaje de patrones clasificados correctamente (PPCC) paras sus correspondientes conjuntos.

	Entrenamiento		Validación		Prueba	
Caso I						
Salida	O_1		O_1		O_1	
PPCC (%)	94.0		71.3		84.6	
Caso II						
Salida	O_1		O_1		O_1	
PPCC (%)	97.7		56.0		82.6	
Caso III						
Salida	O_1	O_2	O_1	O_2	O_1	O_2
PPCC (%)	97.7	99.0	92.6	92.5	91.3	87.8

Tabla 3.5: PPCC por la RNA para los conjuntos de entrenamiento, validación y prueba en cada caso.

En el caso I, la mejor estructura de RNA fue obtenida con 25 neuronas ocultas. Con esta RNA, un 94 % y 71.3 % de los patrones fue clasificado correctamente del conjunto de entrenamiento y validación después de 50000 iteraciones con una constante de aprendizaje $\gamma = 1 \times 10^{-2}$. Mientras tanto en el conjunto de prueba la RNA logró un PPCC = 84.6 %.
En el segundo escenario, la RNA seleccionada tiene 45 neuronas ocultas y logró un 97.7 % y 56 % de patrones clasificados correctamente para los conjuntos de entrenamiento y validación respectivamente. En el conjunto de predicción la RNA disminuyó su desempeño un 2 % respecto al caso I.
Finalmente, para el caso III, después del proceso de selección con 10000 iteraciones en la fase de entrenamiento y con una contante de aprendizaje $\gamma = 5 \times 10^{-3}$, la RNA con el mejor desempeño fue la que tenía 20 neuronas

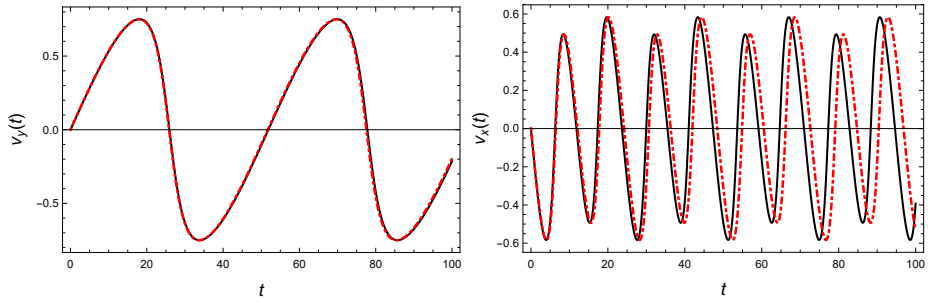

Figura 3.13: Izquierda: velocidad de un electrón usando las condiciones para el caso I con un valor aleatorio E_y (línea continua) y una velocidad del electrón usando el valor E_y predicho por la RNA (línea discontinua) con un error de $\pm 0.5\,\%$. Derecha: velocidad de un electrón usando las condiciones para el caso II con un valor aleatorio E_x (línea continua) y una velocidad del electrón usando el valor E_x predicho por la RNA (línea discontinua) con un error de $\pm 1\,\%$.

ocultas. La RNA obtuvo para el conjunto de entrenamiento un PPCC de $97.7\,\%$ y $99\,\%$ para las salidas 1 y 2 respectivamente. Mientras tanto en el conjunto de validación la RNA clasificó correctamente el $92.6\,\%$ y $92.5\,\%$ de patrones correspondientes a las clases de E_x y E_y respectivamente. Finalmente, probando la RNA con patrones completamente desconocidos, se obtuvo un $91.3\,\%$ de patrones clasificados correctamente para la primer salida y un $87.8\,\%$ para la segunda salida.

Se puede ver que para los casos I y II, el desempeño de la RNA no es tan bueno como en el caso III, pero esto es porque los primeros casos tienen más clases y el error asociado a la clase del campo eléctrico predicho es menor. Mientras tanto, en el último caso, aunque el PPCC es mayor, la incertidumbre asociada al objetivo del campo eléctrico predicho también es mayor. Por esta razón, si es considerado un error asociado mayor para la salida al clasificar la clase c como una clasificación correcta, usando la Ec. (2.19) con un error del doble obtenemos que la condición a satisfacer ahora es

$$T_c - \frac{1}{100} \leq O_c \leq T_c + \frac{1}{100}, \quad 1 \leq c \leq 100, \tag{3.46}$$

donde la salida es asociada a cada componente del campo eléctrico depen-

diendo del caso, el PPCC incrementa pero teniendo un error más grande (ahora de ±1 %) en el campo eléctrico predicho para los casos I y II, como puede verse en la Figura 3.13. El desempeño de la RNA tomando estas consideraciones se muestra en la Tabla 3.6.

	Entrenamiento	Validación	Prueba
Caso I			
Salida	O_1	O_1	O_1
PPCC (%)	100.0	98.0	99.3
Caso II			
Salida	O_1	O_1	O_1
PPCC (%)	100.0	84.6	94.6

Tabla 3.6: PPCC por la RNA considerando un error mayor en el campo eléctrico predicho para los casos I y II.

Una muestra de la velocidad de un electrón construida usando la componente del campo eléctrico predicha por la RNA para el caso III se presenta en las Figuras 3.14 y 3.15. En estas figuras, podemos ver que algunas veces las velocidades predichas son correctas (Figura 3.14), pero debido al error de ±5 % en el campo eléctrico predicho, puede pasar que la velocidad que se predice difiere de la real incluso cuando fue clasificada correctamente (Figura 3.15).

3.3.4. Observaciones finales

La RNA desarrollada clasifica señales numéricas correspondientes a OB en tres situaciones dependiendo del campo eléctrico involucrado.
En el primer caso, la RNA entrenada clasifica señales correctamente correspondientes a OB donde $E_y \in [-\pi/(4\sqrt{3}), \pi/(4\sqrt{3})]$ y $E_x = 0$ para un momento inicial k_0 constante. Un 84.6 % de las señales son clasificadas correctamente con un error de ±0.5 % para el valor de E_y predicho. Cuando es considerado un error de ±1 %, el porcentaje de patrones clasificados correctamente aumenta a 99.3 %.

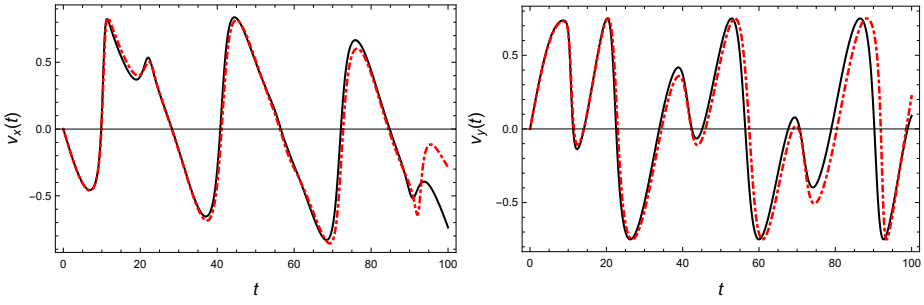

Figura 3.14: Velocidades de un electrón usando las condiciones para el caso III con un valor aleatorio E_x y E_y (líneas continuas) y las velocidades del electrón usando los valores E_x y E_y predichos por la RNA (líneas discontinuas) con alta precisión.

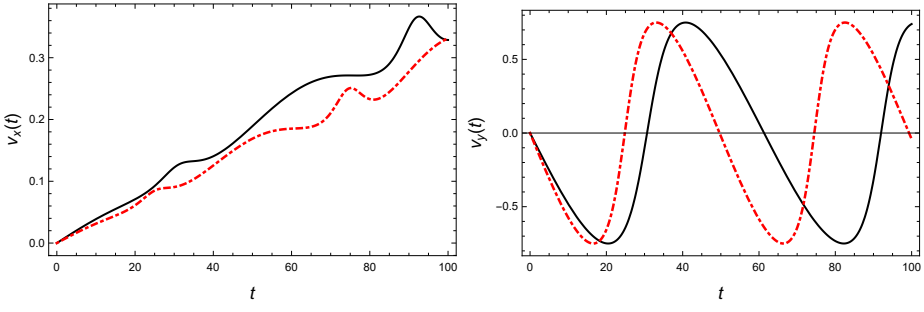

Figura 3.15: Velocidades de un electrón usando las condiciones para el caso III con un valor aleatorio E_x y E_y (líneas continuas) y las velocidades del electrón usando los valores E_x y E_y predichos por la RNA (líneas discontinuas) con baja precisión.

Para el segundo caso, la RNA entrenada clasifica señales correspondientes a OB donde $E_x \in [-\pi/(4\sqrt{3}), \pi/(4\sqrt{3})]$ y $E_y = 0$ con un momento inicial constante. Un 82.6 % de las señales son clasificadas correctamente con un error de $\pm 0.5\,\%$ para el valor de E_x predicho. Cuando es considerado un error de $\pm 1\,\%$, el porcentaje de clasificaciones correctas aumenta a 94.6 %.

En el último caso, la RNA entrenada clasifica señales correspondientes a OB donde ambas componentes del campo eléctrico están entre el intervalo $[-\pi/(4\sqrt{3}), \pi/(4\sqrt{3})]$ para un momento inicial fijo k_0. La RNA clasifica correctamente 91.3 % la componente E_x de las señales con un error de $\pm 5\,\%$, mientras que la componente E_y es clasificada correctamente 87.8 % también con un error de $\pm 5\,\%$.

Como una extensión natural del trabajo se puede considerar la influencia de tensiones en grafeno en las oscilaciones de Bloch desde una perspectiva de redes neuronales artificiales como una guía para ver observables experimentales, lo cual se analiza en la próxima sección.

3.4. Oscilaciones de Bloch en grafeno deformado

El estudio presentado a continuación se basa en el artículo titulado *Study of simulated Bloch oscillations in strained graphene using neural networks* [53], el cual realicé en colaboración de mis asesores de tesis, los Drs. José A. González y Alfredo Raya. Aquí extendemos y generalizamos los hallazgos encontrados previamente al estudiar OB en grafeno, pero ahora en el caso en el que el grafeno se encuentra bajo una deformación uniaxial. Comparado con el caso prístino, la primer diferencia natural que aparece al estar expuesto a una deformación, es el cambio en el periodo de las oscilaciones. Además, se ha observado que las amplitudes de trayectorias cerradas cambian de tal manera que nuevos patrones autointersectantes aparecen [54].
En esta sección abordamos el problema inverso de OB en grafeno uniaxialmente deformado bajo la extensión uniforme de la membrana.

3.4.1. Oscilaciones de Bloch en grafeno deformado

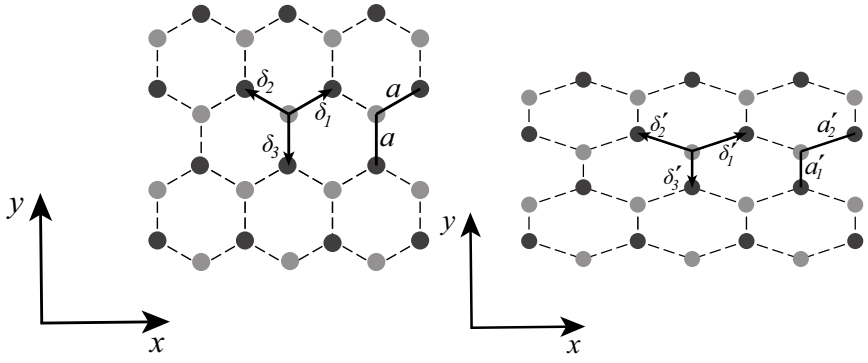

Figura 3.16: Estructura cristalina del grafeno. *Izquierda:* Caso prístino. *Derecha:* Bajo tensión uniforme uniaxial.

Consideramos la conexión entre elasticidad y la descripción de amarre fuerte del grafeno siguiendo de cerca la referencia [55]. Considerando el arreglo de panal de abeja del grafeno como la superposición de dos subredes triangulares, la posición de los átomos en la subred deformada A (ver Figura 3.16) puede escribirse como $\mathbf{x}' = (I + \epsilon)\mathbf{x}$, donde I es la matriz identidad y ϵ es el tensor de tensión, independiente de las coordenadas. Entonces, el Hamitoniano de amarre fuerte para los vecinos cercanos se expresa como

$$H = -\sum_{\mathbf{x}',n} t_n a^\dagger_{\mathbf{x}'} b_{\mathbf{x}'+\boldsymbol{\delta}'_n} + h.c., \tag{3.47}$$

donde $\mathbf{x}'$ corre sobre todos los puntos de la subred A y $\boldsymbol{\delta}'_n$ son los vectores conectando cada punto con sus vecinos cercanos. Aquí $a^\dagger_{\mathbf{x}'}$ y $b_{\mathbf{x}'+\boldsymbol{\delta}'_n}$ son los operadores de creación y aniquilación para los portadores de carga en las subredes A y B respectivamente en los sitios correspondientes $\mathbf{x}'$ y $\mathbf{x}' + \boldsymbol{\delta}'_n$. Notemos que los parámetros de *hopping* t_n en el Hamiltoniano (Ec. 3.47), son considerados independientes de las coordenadas, suposición que es válida sólo para el caso de una tensión uniforme. Usando una expansión de Fourier para los operadores de creación y aniquilación en el espacio de momentos, el Hamiltoniano de amarre fuerte tiene una forma muy similar

al caso ideal

$$H = -\sum_{\mathbf{k},n} t_n e^{-i\mathbf{k}\cdot(I+\epsilon)\cdot\delta_n} a^\dagger_{\mathbf{k}} b_{\mathbf{h}} + h.c., \tag{3.48}$$

con la diferencia de que los parámetros de *hopping* ahora son dependientes de la posición. La relación de dispersión se obtiene fácilmente como

$$\varepsilon(\mathbf{k}) = \pm \left| \sum_n t_n e^{-i\mathbf{k}^*\cdot\delta_n} \right|, \tag{3.49}$$

con $\mathbf{k}^* = (I+\epsilon)\cdot\mathbf{k}$. A orden lineal en el tensor de tensiones, escribimos

$$t_n = t_0 \left(1 - \frac{\beta}{a^2}\boldsymbol{\delta}_n \cdot \epsilon\cdot\right)\boldsymbol{\delta}_n, \tag{3.50}$$

donde t_0 es el parámetro de *hopping* para grafeno prístino y $\beta \simeq 3$ es la variación de la energía de *hopping* debido a la deformación de la red. Entonces, usando

$$\boldsymbol{\delta}_1 = \frac{a}{2}(\sqrt{3},1), \quad \boldsymbol{\delta}_2 = \frac{a}{2}(-\sqrt{3},1), \quad \boldsymbol{\delta}_3 = a(0,-1), \tag{3.51}$$

con a la distancia interatómica en una muestra ideal, explícitamente tenemos

$$\varepsilon(\mathbf{k}) = \pm t_0\sqrt{3 + f(\mathbf{k}^*) - \beta(3\mathrm{Tr}(\epsilon) + f_\epsilon(\mathbf{k}^*) + \beta^2 f_{\epsilon^2}(\mathbf{k}^*))}, \tag{3.52}$$

donde

$$f(\mathbf{k}^*) = 2\cos\left(\sqrt{3}k_x^* a\right) + 4\cos\left(\frac{\sqrt{3}k_x^* a}{2}\right)\cos\left(\frac{3k_y^* a}{2}\right), \tag{3.53}$$

mientras $f_\epsilon(\mathbf{k}^*)$ y $f_{\epsilon^2}(\mathbf{k}^*)$ representan modificaciones del espectro a primer y segundo orden en β, respectivamente. Para el análisis en este artículo, consideramos $\beta = 0$ y usamos la relación de dispersión simplificada

$$\varepsilon(\mathbf{k}) = \pm t_0\sqrt{3 + f(\mathbf{k}^*)} \tag{3.54}$$

y dejamos la relación de dispersión completa para un trabajo futuro.

Para el análisis de OB consideramos la ecuación de movimiento semiclásica

$$\frac{d\mathbf{k}}{dt} = -e\mathbf{E}, \tag{3.55}$$

donde $\mathbf{E}$ representa un campo eléctrico estático y uniforme y e es la carga fundamental. Después de integrar, obtenemos la expresión $\mathbf{k}(t) = \mathbf{k}(0) - e\mathbf{E}t$ y la sustituimos dentro de la relación de dispersión (Ec. 3.54)

$$\frac{d\mathbf{r}}{dt} = \frac{\partial\varepsilon(\mathbf{k})}{\partial\mathbf{k}}\,. \tag{3.56}$$

Integrando esta ecuación, podemos obtener la posición de los portadores de carga a un tiempo dado t. Consideramos un tensor de tensión de la forma

$$\epsilon = \begin{pmatrix} \epsilon_{xx} & 0 \\ 0 & -\nu\epsilon_{xx} \end{pmatrix}\,, \tag{3.57}$$

con cociente de Poisson ν. Abajo detallamos el procedimiento para simular las OB en grafeno deformado para este marco de trabajo.

3.4.2. Oscilaciones de Bloch simuladas

Una vez que tenemos las ecuaciones que describen la posición de los portadores de carga como función del tiempo, necesitamos especificar algunas condiciones iniciales como el momento inicial ($k_x(0)$, $k_y(0)$) o el campo eléctrico externo (E_x, E_y) y simular las trayectorias de los portadores para un lapso fijo, sólo variando la tensión. Adicionalmente, consideramos que $\hbar = a = \tau = 1$ y $e = -1$ para un intervalo de tiempo de $T = 4\pi$ unidades. Note que con estas suposiciones, las cantidades analizadas no tienen unidades físicas.

Dos casos son estudiados de acuerdo a los parámetros que son variados cuando las oscilaciones son generadas numéricamente:

1. El único parámetro que varía es ϵ_{xx} con $\epsilon_{xy} = \epsilon_{yx} = 0$, $\nu = 0.16$ y los otros parámetros están fijos en tres subcasos

   a) $E_x = 1$, $E_y = 0$, $k_x(0) = 0$, $k_y(0) = \pi/\sqrt{3}$.
   b) $E_x = 0$, $E_y = 1$, $k_x(0) = \pi/\sqrt{3}$, $k_y(0) = 0$.

c) $E_x \neq 0$, $E_y \neq 0$, $k_x(0) = k_y(0) = 0$.

2. Los parámetros que varían son ϵ_{xx} y ν con $E_x \neq 0$, $E_y \neq 0$, $k_x(0) = k_y(0) = 0$ y $\epsilon_{xy} = \epsilon_{yx} = 0$.

En el primer caso, se usan N valores diferentes para ϵ_{xx} y están equidistantes en el intervalo [0, 0.25] y etiquetadas en C_ϵ clases o grupos, de tal forma que cada clase tiene el mismo número de simulaciones, es decir, mód $(N/C_\epsilon) = 0$ debe satisfacerse. La RNA realizará una clasificación usando como datos de entrada para el entrenamiento, las componentes $x(t)$ y $y(t)$ de la posición del portador de carga. Interpretamos cada clase predicha con un valor promedio para ϵ_{xx} con un error relativo de $\pm\epsilon_L/2C_\epsilon$ respecto a la longitud total del intervalo $\epsilon_L = 0.25$. Análogo a la Ec. (2.20), definimos

$$\hat{\epsilon}_m = (2m-1)\frac{\epsilon_L}{2C_\epsilon}, \quad 1 \leq m \leq C_\epsilon, \tag{3.58}$$

como el valor promedio de ϵ_{xx} predicho asociado con la clase m, donde C_ϵ es el número total de clases.

El segundo caso es similar al primero, pero ahora también el parámetro ν varía en el intervalo [0, 0.2], seleccionando N' valores diferentes equidistantes entre sí y agrupándolos en C_ν clases. En este escenario, el número total de patrones generados es $N \times N'$ y la RNA clasifica simultáneamente ambos parámetros: ϵ_{xx} se asocia a una clase del total de C_ϵ y ν a una clase de C_ν clases. El valor promedio de ν predicho tiene una expresión similar a la de la Ec. (3.58).

$$\hat{\nu}_n = (2n-1)\frac{\nu_L}{2C_\nu}, \quad 1 \leq n \leq C_\nu, \tag{3.59}$$

con $\nu_L = 0.2$ la longitud total del intervalo donde ν varía y un error asociado de $\pm\nu_L/2C_\nu$.
En las Ecs. (3.58) y (3.59) observamos que al incrementar el número de clases, el error asociado con cada predicción es más pequeño. Vale la pena mencionar que mientras que el error en la predicción decrece, también la eficiencia en la clasificación decrece, como ilustramos en la próxima sección.

3.4.3. Consideraciones de la RNA

Usamos una RNA con alimentación hacia adelante para clasificar patrones, tal que la red estime los parámetros ϵ y ν que generaron las simulaciones numéricas. Los datos de entrada usados se obtienen de dos series temporales: la posición $x(t)$ y $y(t)$ de la carga en una OB simulada para el lapso $0 \leq t \leq \tau$ (donde τ es la duración total de la oscilación), sujetas a diferentes condiciones impuestas y donde la posición se obtiene integrando numéricamente la Ec. (3.56). Dividimos la simulación en cincuenta pasos de tiempo, tal que $t_i = i\Delta t$, con $0 \leq i \leq 49$ y $\Delta t = \tau/50$. El vector de entrada para cada patrón p se construye como

$$I^p = \{x(t_0), y(t_0), \ldots, x(t_{49}), y(t_{49})\},\ 1 \leq p \leq N_p, \tag{3.60}$$

con N_p el número total de patrones. Como se mencionó en la subsección previa, $N_p = N$ en el primer caso y $N_p = N \times N'$ en el segundo. Del total de patrones, 70 % de ellos son seleccionados aleatoriamente para entrenar la red y el restante 30 % conforman el conjunto de validación.

Para probar el desempeño de la red, el mismo número de señales que en el conjunto de validación se simulan usando valores aleatorios de las variables ϵ_{xx} y ν, asegurando que las nuevas simulaciones se encuentren dentro del rango considerado. La selección del conjunto de entrenamiento y validación se realiza una vez que los valores objetivos para los patrones son especificados, lo cual se hace mediante la Ec. (2.18) para cada salida tomando C_ϵ y C_ν.

La RNA fue programada en FORTRAN 90 y entrenada con un algoritmo de aprendizaje supervisado *offline* retroalimentado, diseñado para minimizar una función de costo tipo error cuadrático medio [9, 11]. La generación de las OB simuladas, el preprocesado y la visualización de los resultados, fueron realizados en Wolfram Mathematica. La estructura de la red cuenta con una capa de entrada con cien neuronas que reciben los datos extraídos de cada patrón, una capa oculta con un número variable de neuronas por determinar de acuerdo al desempeño de la RNA y una capa de salida con una neurona en el primer caso y dos neuronas en el segundo. Una de las neuronas produce un valor que es relacionado a ϵ_{xx} y el otro a ν. Por esta razón, la estructura de la capa de salida depende del caso.

Con los datos de entrada, los objetivos propuestos y la estructura de la red listos, es necesario entrenar la red con un número adecuado de iteraciones y evaluar el desempeño de la red contando el número de patrones clasificados correctamente. Usamos la condición de la Ec. (2.19) en cada salida para encontrar el porcentaje de patrones clasificados correctamente.

En la próxima subsección presentamos los resultados obtenidos después de que se entrenaron las RNAs, considerando variaciones en algunos de los parámetros de la red, por ejemplo, el número de neuronas ocultas, la constante de aprendizaje y/o el número de clases en las que fueron divididas las simulaciones.

3.4.4. Resultados

Los resultados presentados en esta sección son aquellos generados por la red que obtuvo el menor costo al final del entrenamiento de todos los parámetros de la red considerados, donde el número de neuronas ocultas fue igual a 2^j para $2 \leq j \leq 6$ y la constante de aprendizaje fue igual a 3^{-l} para $3 \leq l \leq 7$, con j y l enteros. Entonces, se tienen un total de 25 combinaciones de parámetros explorados con un código paralelizado usando MPI.
Usamos un total de 2×10^4 iteraciones en el aprendizaje, el número de simulaciones generadas es de 10^3 y dependiendo del caso, el número de clases utilizadas es diferente.

Predicción de ϵ_{xx}

El primer caso incluye simulaciones de OB con las condiciones iniciales mencionadas en la Sección 3.4.2 y fueron clasificadas en $C_\epsilon = 50, 100$ y 200 clases.
Una muestra de los datos seleccionados como entradas para la red de acuerdo a la Ec. (3.60) para los subcasos a), b) y c) se encuentran en las Figuras 3.17, 3.18 y 3.19 respectivamente.

Ya que hemos seleccionado la red con los parámetros que producen el menor costo durante el entrenamiento, el desempeño de la red es evaluado calculando el porcentaje de patrones correctamente clasificados (PPCC) en

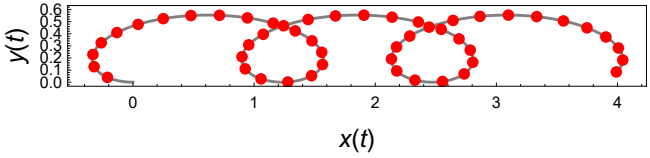

Figura 3.17: Muestra de la trayectoria de un electrón cuando $\epsilon_{xx} = 0.20$ durante el intervalo de tiempo $0 \leq t \leq 4\pi$ y las condiciones iniciales consideradas para el caso 1a. Los puntos rojos representan las coordenadas usadas como entradas para la RNA.

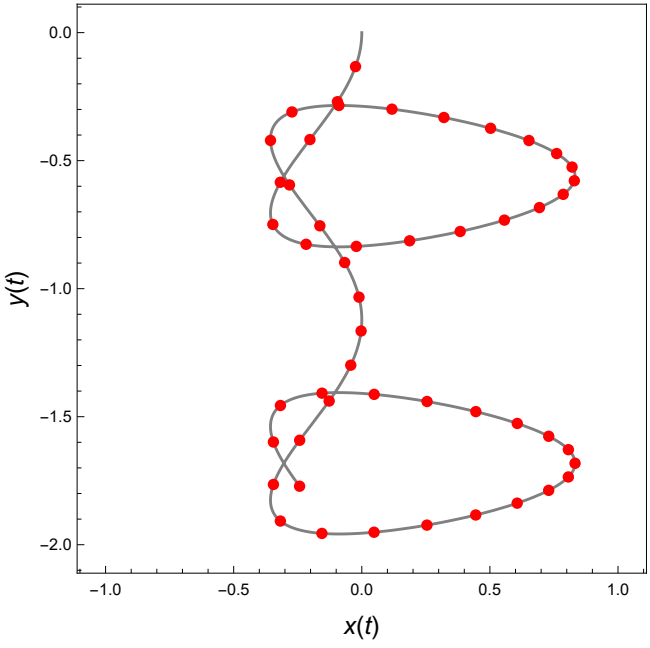

Figura 3.18: Muestra de la trayectoria de un electrón cuando $\epsilon_{xx} = 0.20$ durante el intervalo de tiempo $0 \leq t \leq 4\pi$ y las condiciones iniciales consideradas para el caso 1b. Los puntos rojos representan las coordenadas usadas como entradas para la RNA.

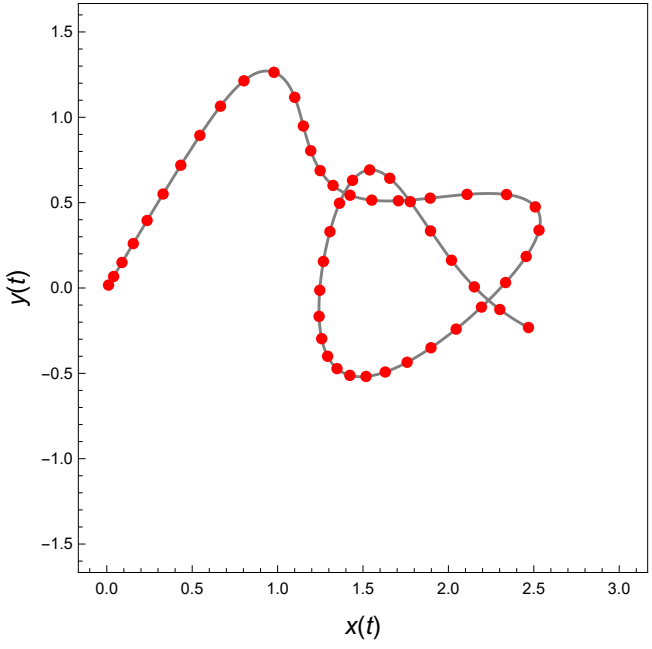

Figura 3.19: Muestra de la trayectoria de un electrón cuando $E_x = -0.33$, $E_y = -0.79$ y $\epsilon_{xx} = 0.20$ durante el intervalo de tiempo $0 \leq t \leq 4\pi$ y las condiciones iniciales consideradas para el caso 1c. Los puntos rojos representan las coordenadas usadas como entradas para la RNA.

cada caso y los resultados se encuentran en la Tabla 3.7. Podemos observar que para cada subcaso, entre más clases son consideradas, más bajo es el PPCC como es esperado. También observamos, por un lado, que el mejor desempeño se obtiene en el caso 1a) y por otro lado, el caso 1c) (donde ambas componentes del campo eléctrico son diferentes a cero) tiene el menor PPCC. Como mencionamos anteriormente, aunque el PPCC es menor cuando el número de clases incrementa, el error asociado al valor promedio predicho es menor como se establece en la Ec. (3.58). Por lo tanto, el error asociado con el valor promedio de $\hat{\epsilon}$ es de $\pm 1\,\%$ de ϵ_L cuando las simulaciones son divididas en cincuenta clases, mientras que el error asociado es $\pm 0.25\,\%$ de ϵ_L cuando son seleccionadas doscientas clases. Dependiendo en la precisión requerida, podemos escoger el número de clases en el que serán divididos los patrones.

PPCC(%) para	Entrenamiento	Validación	Prueba
Caso 1a			
$C_\epsilon = 50$	90.1	90.0	88.6
$C_\epsilon = 100$	82.5	77.6	86.0
$C_\epsilon = 200$	82.1	77.0	79.0
Caso 1b			
$C_\epsilon = 50$	88.1	86.6	90.3
$C_\epsilon = 100$	80.2	75.6	79.6
$C_\epsilon = 200$	68.4	65.0	68.3
Caso 1c			
$C_\epsilon = 50$	81.8	86.6	81.6
$C_\epsilon = 100$	66.7	65.3	69.0
$C_\epsilon = 200$	59.5	52.6	54.6

Tabla 3.7: PPCC obtenidos por la RNA para los conjuntos de entrenamiento, validación y prueba en el caso 1. La salida generada por la red predice el valor de ϵ_{xx}.

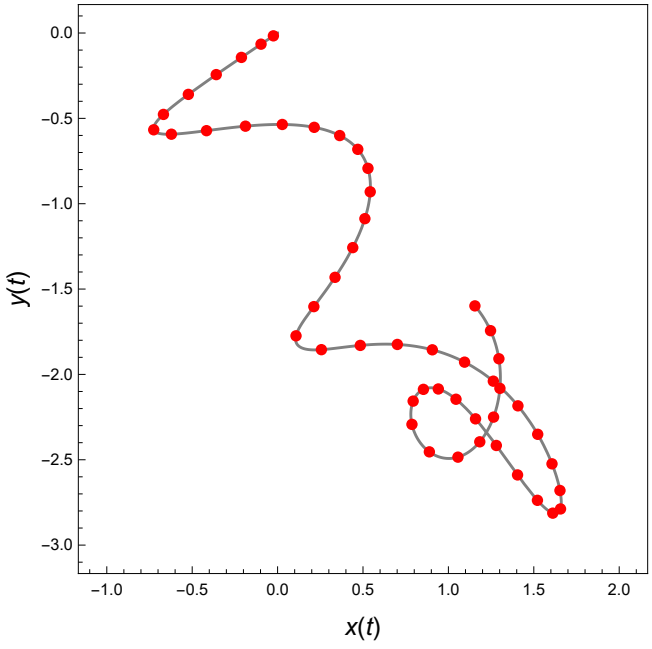

Figura 3.20: Muestra de la trayectoria de un electrón cuando $E_x = 0.76$, $E_y = 0.78$, $\epsilon_{xx} = 0.20$, y $\nu = 0.16$ durante el intervalo de tiempo $0 \leq t \leq 4\pi$ y las condiciones iniciales consideradas para el caso dos. Los puntos rojos representan las coordenadas usadas como entradas para la RNA.

Predicción de ϵ_{xx} y ν

Para el segundo caso de estudio, creamos la misma cantidad de simulaciones que el caso previo pero seleccionando $N = 50$ y $N' = 20$, variando ϵ_{xx} y ν respectivamente y considerando $C_\epsilon = 5$ y 10 clases para ϵ_{xx} y $C_\nu = 2$ y 4 clases para ν. E_x y E_y fueron seleccionados aleatoriamente en el intervalo [-1,1], presentando una muestra de la trayectoria para el lapso considerado y las condiciones iniciales en la Figura 3.20. El PPCC para este caso se muestra en la Tabla 3.8.

El comportamiento del PPCC es similar al caso 1, obteniendo más patrones clasificados correctamente cuando se considera un número menor de clases. Como el número de simulaciones es el mismo que en el caso

Caso 2				
PPCC(%) para	Salida	Entrenamiento	Validación	Prueba
$C_\epsilon = 5$	O_1	99.8	100.0	96.3
$C_\nu = 2$	O_2	99.0	98.0	97.3
$C_\epsilon = 5$	O_1	99.4	98.0	97.6
$C_\nu = 4$	O_2	91.4	91.3	88.3
$C_\epsilon = 10$	O_1	95.1	96.0	92.6
$C_\nu = 2$	O_2	92.7	92.1	94.0
$C_\epsilon = 10$	O_1	98.5	98.0	84.3
$C_\nu = 4$	O_2	83.1	83.6	87.0

Tabla 3.8: PPCC obtenido por la RNA para los conjuntos de entrenamiento, validación y prueba en el caso 2. La salida O_1 predice el valor de ϵ_{xx} y la salida O_2 el de ν. Ambas predicciones fueron divididas en C_ϵ y C_ν clases respectivamente.

anterior, pero con dos parámetros que varían, el número total de clases para cada parámetro es menor. Como consecuencia, los errores asociados a los valores predichos de ϵ_{xx} son $\pm 5\,\%$ y $\pm 10\,\%$ de ϵ_L para 10 y 5 clases respectivamente y un error para el valor predicho de ν de $\pm 12.5\,\%$ y $\pm 25\,\%$ de ν_L para 4 y 2 clases respectivamente.

3.4.5. Conclusiones

En esta sección hemos considerado el problema inverso de OB en grafeno deformado. Consideramos la situación de una tensión por tracción uniaxial y uniforme, descrita por un tensor diagonal independiente de las coordenadas en la correspondiente descripción de amarre fuerte del grafeno. De la relación de dispersión resultante, en el límite cuando la variación de la energía de *hopping* debida al desplazamiento de los átomos de carbono desaparece, tal que dicha relación se describe en la Ec. (3.54) con un aproximación semiclásica, simulamos OB para diferentes orientaciones del campo eléctrico y variando el cociente de Poisson, dejando el resto de parámetros del modelo fijos. Alimentando la RNA con 700 señales de entrenamiento, fuimos capaces de clasificar las componentes del tensor de tensiones para

300 señales nuevas generadas aleatoriamente.
Para el caso 1, cuando la componente E_y del campo eléctrico externo aplicado al grafeno es cero, la red tiene una precisión en sus predicciones para ϵ_{xx} que va desde 79.0 % a 88.6 % con un error de ± 0.25 % y ± 1 % respectivamente. Análogamente, cuando $E_x = 0$ la red tiene una precisión en sus predicciones para ϵ_{xx} que va desde 68.3 % a 90.3 % con un error de ± 0.25 % y ± 1 % respectivamente.
En la situación en que el campo eléctrico externo tiene una orientación aleatoria, la precisión al predecir ϵ_{xx} va de 54.6 % a 81.6 % con un error de ± 0.25 % y ± 1 % respectivamente.
En el caso 2, la red predice simultáneamente ϵ_{xx} y ν para una orientación de campo eléctrico aleatoria, obteniendo una precisión de 96.3 % con un error de ± 10 % prediciendo ϵ_{xx} y una precisión de 97.3 % con un error de ± 25 % prediciendo ν. Esta situación tiene la mayor precisión, pero también el mayor error asociado a cada parámetro. Para el escenario con el menor error asociado, la red tiene una precisión de 84.3 % con un error de ± 5 % prediciendo ϵ_{xx} y una precisión de 87.0 % con un error de ± 12.5 % prediciendo ν. Estos resultados pueden ser mejorados incrementando el número de simulaciones que alimentan la RNA, pero también incrementando el tiempo computacional usado durante el entrenamiento de la red.

Estos resultados alentadores nos motivan a perseguir un estudio más completo de la relación de dispersión completa [54]. Este es trabajo en proceso y algunos resultados preliminares se encuentran a continuación.

3.5. OB en grafeno deformado ($\beta \neq 0$)

En esta sección se presentan los primeros resultados obtenidos al analizar las oscilaciones de Bloch en grafeno sujeto a deformaciones, pero ahora considerando el parámetro β en la relación de dispersión de la Ec. (3.52) de la sección anterior, distinto de cero. Además, se pretende comparar el desempeño del enfoque de salidas multiclase, con el enfoque OVA, al utilizar regresores, redes neuronales artificiales y redes neuronales convolucionales. También, se hace uso de una idea parecida al de los mapas de colores, que se expone en el Capítulo 5, para localizar los patrones en donde los métodos realizan un mayor número de clasificaciones incorrectas.

El primer paso fue entonces, generar simulaciones fijando todos los parámetros y variando únicamente β. El procedimiento es análogo al realizado en la sección anterior, donde obtenemos la posición de un electrón de prueba en un intervalo de tiempo $T = 4\pi$ unidades e integrando la Ec. (3.56), utilizando la relación de dispersión completa y un tensor de tensión de la forma Ec. (3.57). De esta manera, generamos 1000 simulaciones al fijar $\hbar = a = \tau = E_y = 1$, $E_x = 0$, $k_y(0) = 0$, $k_x(0) = \pi$, $\epsilon_{xx} = 0.5$, $\nu = 0.16$ y utilizando $\beta = (1/1000)p$ con $p = 1, \ldots, 1000$. Entonces, la

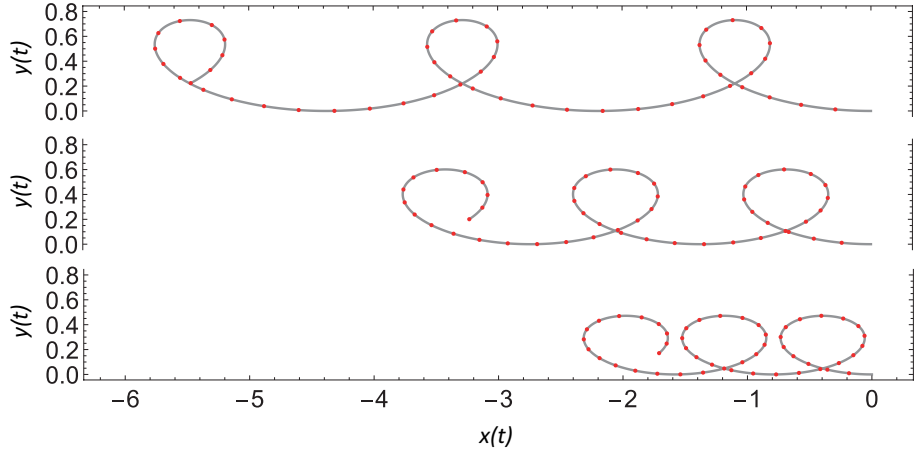

Figura 3.21: Trayectoria de un electrón de prueba, simulada numéricamente para distintos valores del parámetro β y utilizando las condiciones iniciales mencionadas en el texto. La curva continua es la solución numérica, mientras que los puntos indican los valores que se toman para formar los vectores de datos. Arriba: Trayectoria simulada al considerar $\beta = 0.001$. Centro: Trayectoria simulada al considerar $\beta = 0.5$. Abajo: Trayectoria simulada al considerar $\beta = 1$.

primer señal simulada corresponde a la simulación generada con un valor $\beta_{min} = 0.001$, mientras que la última se genera con $\beta_{max} = 1$. Se discretiza posteriormente cada señal simulada que se obtiene, seleccionando el valor de la posición en 50 instantes de tiempo diferentes, para obtener el vector

de datos

$$\vec{x}^p = \{r_x(t_1), r_y(t_1), \ldots, r_x(t_i), r_y(t_i), \ldots, r_x(t_{50}), r_y(t_{50})\}^p, \tag{3.61}$$

donde se usan los valores $t_i = (T/50)i$ para $i = 1, ..., 50$ y donde el superíndice etiqueta cada uno de los patrones. Por lo tanto, cada vector de entrada o patrón de datos a clasificar, cuenta con un total de 100 valores. La integración numérica de las ecuaciones, la discretización de la trayectoria para formar los vectores de datos y la visualización de los resultados, se realizó en Wolfram Mathematica. En la Figura 3.21, se muestran algunas de las trayectorias generadas al variar el valor de β, así como los puntos que se toman para formar los vectores de datos para. Estos vectores de datos, después nos sirven como datos de entrada para entrenar los métodos de *machine learning.*
Antes de usar estos vectores como datos de entrada para el entrenamiento, primero realizamos una transformación sobre estos y les asignamos sus correspondientes objetivos, los cuales dependen del número de clases consideradas y del enfoque a utilizar. El procedimiento de transformación, la selección de objetivos y los entrenamientos de los métodos de AA, se implementaron en Python.

Las transformaciones de los datos consideradas, son las siguientes:

$$\begin{aligned} \hat{x}_j^p &= \frac{x_j^p - \min(x_j)}{\max(x_j) - \min(x_j)}; \\ \hat{x}_j^p &= \frac{x_j^p - \bar{x}_j}{\max(x_j) - \min(x_j)}; \\ \hat{x}_j^p &= \frac{x_j^p - \bar{x}_j}{\mathrm{std}(x_j)}; \end{aligned} \tag{3.62}$$

en donde $1 \leq j \leq 100$, $\max(x_j)$ y $\min(x_j)$ es el máximo y el mínimo de la componente j de todos los patrones respectivamente, $\bar{x}_j = \sum_p x_J^p/P$ y $\mathrm{std}(x_j) = \sqrt{\sum_p (x_j^p - \bar{x}_j)^2/P}$. Al transformar los datos empleando la primer ecuación se le conoce como normalización min-max (la cual denotaremos con $\hat{I}_1$), a la transformación de la ecuación de en medio se le da el nombre de normalización promedio ($\hat{I}_2$) y a la última ecuación se le conoce como

normalización estándar ($\hat{I}_3$). Estos datos transformados son los que introducimos como datos de entrada para entrenar los métodos de aprendizaje automático.
Una vez que tenemos los datos de entrada, seleccionamos los objetivos de acuerdo al enfoque que se emplea. Una salida con los objetivos de la forma de la Ec. (2.18) si se utiliza el enfoque de salidas multiclase y un número de salidas igual al número de clases consideradas si se utiliza el enfoque OVA. Posteriormente, se dividen los patrones en los respectivos conjuntos de entrenamiento y prueba, tomando aleatoriamente el 80 % de los patrones generados para formar el conjunto de entrenamiento y el restante se toma como conjunto de prueba.

Los entrenamientos que se utilizaron en los tres métodos analizados, fueron del tipo supervisado y usando *minibatches*, con un tamaño de *batch* igual a 32. Para el regresor y la RNA se consideraron $2\text{x}10^4$ iteraciones, mientras que en la RNC se utilizaron $2\text{x}10^3$. Las funciones de activación que se utilizaron en el regresor y en la RNA fueron sigmoides, con una función de costo de entropía cruzada cuando se utiliza el enfoque OVA, mientras que se usa una función de costo del tipo error cuadrático medio al utilizar el enfoque multiclase. En el caso de la RNC, la función de activación en la capa convolucional fue una sigmoide y en la capa de salida una función de activación lineal, utilizando una función de costo de error cuadrático medio.
En todos los métodos se exploraron 5 valores diferentes del parámetro de aprendizaje, en donde $\gamma = 1/\exp(l)$, considerando $l = 0, ..., 4$.
El número de clases tomado en cuenta para ambos enfoques fue de 50, 20 y 10 clases, por lo que el error asociado al parámetro estimado, en este caso β, es de $\pm 1\,\%$, $\pm 2.5\,\%$ y $\pm 5\,\%$ respectivamente.

Lo que se hizo primero entonces, fue entrenar un regresor con todas las condiciones antes mencionadas y los mejores resultados dentro del espacio de parámetros explorados, se muestran en la Tabla 3.9. En esta tabla observamos, que cuando se consideran 50 clases, el mejor resultado en el conjunto de prueba se obtiene con el enfoque multiclase, donde se alcanza un $29.5\,\%$ de patrones clasificados correctamente. Cuando son 20 clases, el mejor desempeño se logra al considerar el enfoque multiclase, donde se

obtiene un 60.0 % de patrones clasificados correctamente en el conjunto de prueba. Finalmente, al considerar 10 clases con el enfoque OVA, obtenemos un PPCC de 55.5 en el conjunto de prueba, mientras que con el enfoque multiclase se alcanza 80.0 en el mismo conjunto.
Podemos observar que en todas las situaciones, el enfoque mediante salidas multiclase supera al enfoque OVA, cuando entrenamos un regresor. Además, el tiempo de cómputo empleado en el entrenamiento por el enfoque multiclase, es menor que el OVA, sin importar el número de clases considerado.
Posterior al entrenamiento del regresor, se procedió a visualizar la informa-

PPCC por un regresor logístico (%)					
Enfoque	Clases	Normalización	γ	Entrenamiento	Prueba
OVA	50	$\hat{I}_3$	1	10.1	7.5
	20	$\hat{I}_3$	0.367	25.0	25.0
	10	$\hat{I}_1$	1	54.5	55.5
Multiclase	50	$\hat{I}_2$	0.018	27.6	29.5
	20	$\hat{I}_3$	0.018	52.8	60.0
	10	$\hat{I}_2$	0.135	76.0	80.0

Tabla 3.9: PPCC por un regresor logístico en el conjunto de entrenamiento y en el de prueba. Se consideran 50, 20 y 10 clases, utilizando un enfoque OVA y un enfoque de salidas multiclase. Los resultados corresponden a la normalización y constante de aprendizaje que tuvieron el mejor desempeño.

ción de acuerdo a los patrones que son clasificados correctamente, a través del uso de barras. Esto se hace para localizar regiones en donde el método falla al clasificar los patrones. La manera de visualizar los datos, se idealizó de manera similar a los mapas de colores al considerar la clasificación de dos parámetros simultáneamente, como se hace más adelante en el Capítulo 5, pero aquí nos limitamos a una dimensión, ya que sólo estamos estimando un parámetro. Por este motivo, los datos los visualizamos a través de barras, dependiendo del número de aciertos por clase en el que el regresor clasificó correctamente.
En la Fig. 3.22, podemos observar las regiones donde se clasifica correc-

tamente al usar ambos enfoques, usando los parámetros expuestos en la tabla anterior. Al considerar 10 clases en las que se dividió el parámetro β, lo más que se puede lograr, es conseguir cien aciertos por clase, lo cual se logra en las clases 1, 4 y 10 cuando se usa el enfoque OVA y en las clases 2 y 9 al usar el multiclase. A pesar de que con el enfoque OVA obtenemos un

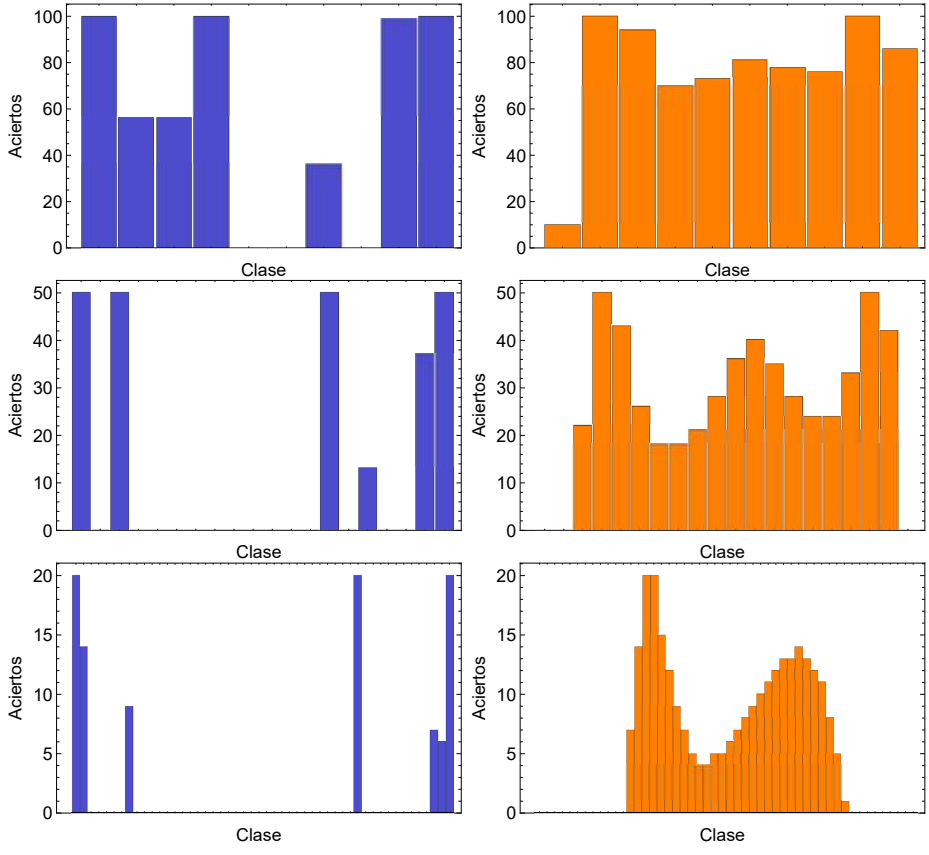

Figura 3.22: Aciertos obtenidos por el regresor, al clasificar los patrones considerando un enfoque OVA (barras azules) y uno de salidas multiclase (barras naranjas). Arriba: 10 clases. Centro: 20 clases. Abajo:50 clases

mayor número de clases con el total de aciertos posibles, en el multiclase se distribuyen mejor los aciertos y se puede observar que en el enfoque OVA

hay más regiones con pocos aciertos, comparado con el multiclase. Al incrementar el número de clases, las regiones en donde hay más clasificaciones incorrectas también aumenta en ambos enfoques. En el enfoque OVA, las clases con mayor número de aciertos son las de los extremos, es decir, el regresor predice buenos resultados para valores de β cercanos a uno y a cero. Por otro lado, al usar el enfoque multiclase, el regresor obtiene las peores estimaciones para los valores de β mencionados. De estos resultados, podemos decir que el comportamiento de las regiones con aciertos para cada enfoque, al parecer es independiente tanto de la manera en que se toman los conjuntos de prueba y entrenamiento, así como de la normalización que se utiliza.

Una vez que tenemos el desempeño que tuvo el regresor logístico, pasa-

PPCC por una RNA (%)					
Enfoque	Clases	Normalización	γ	Entrenamiento	Prueba
OVA	50	$\hat{I}_2$	1	44.6	34.5
	20	$\hat{I}_2$	1	91.1	89.0
	10	$\hat{I}_2$	0.367	96.2	94.5
Multiclase	50	$\hat{I}_2$	1	21.5	28.9
	20	$\hat{I}_1$	0.135	47.1	56.4
	10	$\hat{I}_2$	0.135	71.8	73.0

Tabla 3.10: PPCC por una RNA en el conjunto de entrenamiento y en el de prueba. Se consideran 50, 20 y 10 clases, utilizando un enfoque OVA y un enfoque de salidas multiclase. Los resultados corresponden a la normalización y constante de aprendizaje que tuvieron el mejor desempeño.

mos a analizar los resultados que se obtienen al entrenar una red neuronal artificial. Para esto, primero se consideró una estructura de red con una sola capa oculta, con un número de pesos similar al regresor. Esto quiere decir, que para el enfoque OVA necesitamos una red con aproximadamente $100*C+C$ pesos (con C clases), mientras que para el enfoque multiclase se requieren 101. Por este motivo, al usar el enfoque OVA y considerar 10, 20 y 50 clases, se usa una estructura con 9, 17 y 33 neuronas ocultas, respectivamente. Mientras que al emplear una clasificación por salidas multiclase,

sólo se utiliza una neurona oculta. Con estas consideraciones se procedió

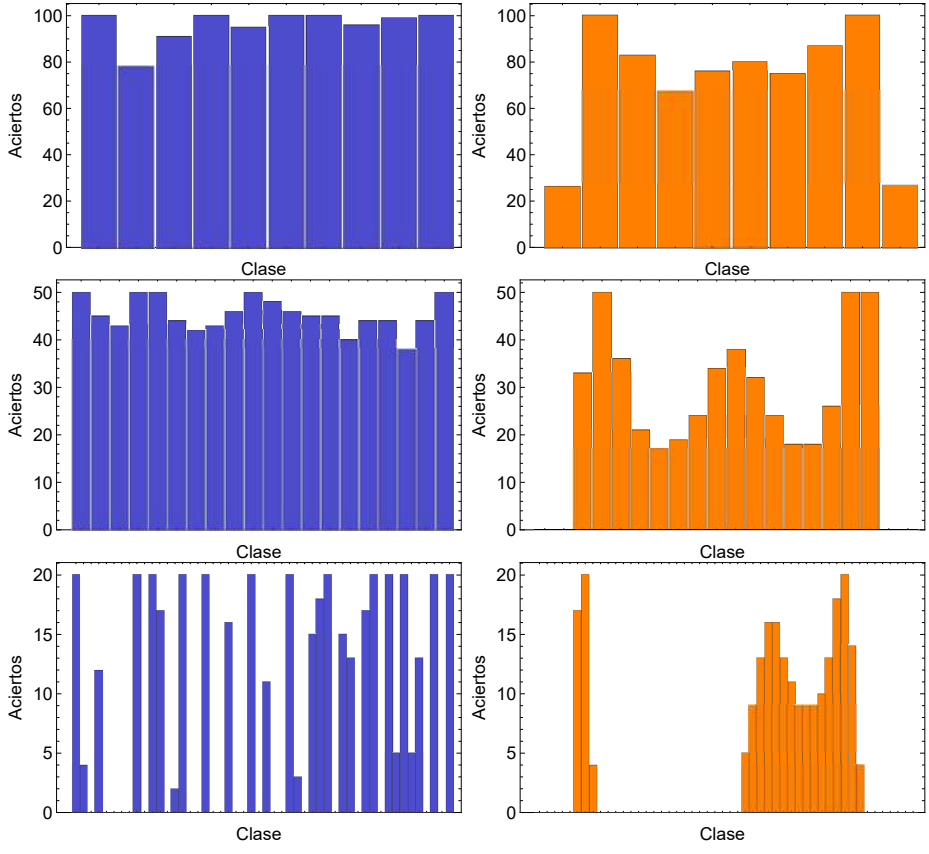

Figura 3.23: Aciertos obtenidos por la RNA, al clasificar los patrones considerando un enfoque OVA (barras azules) y uno de salidas multiclase (barras naranjas). Arriba: 10 clases. Centro: 20 clases. Abajo:50 clases

con los entrenamientos de las RNAs, donde los resultados de los mejores desempeños, se encuentran presentes en la Tabla 3.10. Aquí se observa, que con el enfoque OVA la red neuronal tiene un mejor desempeño al clasificar correctamente los patrones, comparado con el enfoque multiclase. Además, comparado con el regresor logístico, el enfoque multiclase tiene un desempeño ligeramente inferior, mientras que con el enfoque OVA el desempeño

es mucho mejor. En esta situación, dependiendo de la precisión deseada, si conviene emplear tiempo adicional en el entrenamiento al usar el enfoque OVA.
Una vez más, procedemos a visualizar los datos anteriores, usando un esquema de barras como el descrito anteriormente. En la Figura 3.23, se observan las regiones donde hay un mayor número de aciertos al clasificar los patrones por clases. El comportamiento de los aciertos para cada enfoque, es parecido al que se obtiene al usar el regresor. La región de aciertos para el caso multiclase es casi idéntico al del regresor, donde el desempeño para las clases de los extremos es inferior, es decir, para valores de β cercanos a 0 y a 1. Difiere un poco el comportamiento, debido a que en este escenario, el desempeño es menor. En el caso del enfoque OVA, las regiones de aciertos no se pueden comparar tan fácilmente, ya que la brecha de aciertos entre ambos enfoques es enorme. Podemos suponer que es un comportamiento análogo a cuando se usa el regresor, porque también tiene buenas predicciones para los valores de β cercanos a 0 y 1, además de que en ambos, hay múltiples regiones con pocos aciertos separadas de regiones con muchos aciertos, aspecto que no está presente con el enfoque multiclase, que a lo más, tiene 3 regiones principales con pocos aciertos.
Hasta este punto también podemos decir, que al parecer, el comportamiento de las regiones de aciertos para cada enfoque, es independiente del método de aprendizaje utilizado. Falta entonces, discutir los resultados obtenidos con la red neuronal convolucional para poder afirmar estas suposiciones.

Para entrenar la red neuronal convolucional (RNC), hicimos uso de la librería Tensorflow. Una vez que implementamos la red, utilizamos los vectores de datos disponibles, pero vistos como matrices, en lugar de como vectores. Así podemos entrenar la red convolucional, como si estuviera clasificando imágenes en escala de grises, donde cada vector de datos $\vec{x}^p$, lo expresamos por la matriz X^p, cuyas componentes están dadas por

$$X^p_{ij} = x^p_{j+10(i-1)}, \tag{3.63}$$

para $i, j \leq 10$. De esta manera, cada patrón de datos lo podemos visualizar como una imagen, con aspectos parecidos a los que se ven en la Fig 3.24. Los 10 valores de la primera fila (de arriba hacia abajo), corresponden a las componentes de la posición para los primeros 5 valores en el tiempo con-

siderados. La segunda fila, corresponde a las componentes de la posición para los siguientes 5 valores de tiempo y así sucesivamente, siendo la última fila las componentes de la posición para los últimos 5 tiempos. En la figura podemos observar imágenes muy similares entre sí, donde podemos ver un diseño para la representación de los datos, muy parecido para diferentes valores del parámetro que se quiere estimar, pero con pequeñas diferencias en la intensidad de color. A pesar de que estamos mostrando estos datos antes de normalizar, a la hora de programar esta parte, resulta más sencillo si primero normalizamos los datos y después transformamos los vectores a matrices por medio de la Ec. 3.63. La estructura de red implementada,

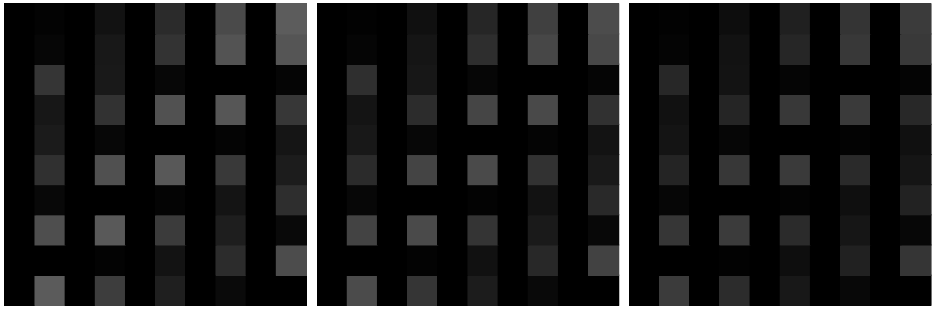

Figura 3.24: Representación de los datos como imágenes en escala de grises. La representación de los datos, corresponden a los patrones generados con valores de $\beta = 0.001$ (izquierda), $\beta = 0.5$ (centro) y $\beta = 1$ (derecha).

consiste de una capa convolucional y una capa completamente conectada. Utilizamos una capa convolucional con 5 filtros de tamaño 3×3, un tamaño de paso igual a 1, relleno igual a 0 y una función de activación sigmoide. Posteriormente en el agrupamiento máximo, se usa un tamaño de paso y agrupamiento de 2.
Con este procedimiento, obtenemos un tensor de salida de tamaño $4\times4\times5$. Entonces para poder usar una capa completamente conectada de la manera usual, necesitamos un vector de datos, por lo que ocupamos primero transformar el tensor en un vector. De esta manera, obtenemos un vector con 80 elementos, los cuales se conectan con una neurona de salida en el caso de usar el enfoque multiclase o en C neuronas de salida al usar el enfoque OVA. Estas neuronas de salida no cuentan con función de activación, por

lo que las salidas son funciones lineales respecto a los pesos en esta última capa.

El tamaño y la cantidad de filtros que se escogieron, en principio, era para que el número de parámetros dentro de la red fuera equivalente al de los métodos anteriores y realizar una comparación más justa. Pero el entrenamiento requirió de un mayor tiempo computacional, por lo que al final se optó por utilizar un menor número de iteraciones al entrenar la RNC. No sabemos si esto se debe a que la implementación no esta bien optimizada al usar Tensorflow. Entonces, para poder realizar una mejor comparación, la RNC debería de ser implementada sin usar Tensorflow.
En la Tabla 3.11, podemos observar el desempeño que tuvo el método al clasificar los patrones usando cada uno de los enfoques. En este caso, el enfoque de salidas multiclase es el que tiene un mejor desempeño comparado con el OVA. Este comportamiento también se encuentra presente al usar el regresor. Además, sorprendentemente al usar el enfoque multiclase, el PPCC no disminuye tanto como lo hace el regresor y la RNA. La diferencia del PPCC cuando se consideran 10 clases y cuando se consideran 50, solo es de 6 %, mientras que cuando se usa el regresor o la RNA es de más de 40 %. En cuanto al enfoque OVA, los resultados obtenidos son mejores que el regresor, pero peores que la RNA. Una vez más, visualizamos los

PPCC por una RNC (%)					
Enfoque	Clases	Normalización	γ	Entrenamiento	Prueba
OVA	50	$\hat{I}_2$	0.018	26.2	21.0
	20	$\hat{I}_2$	0.049	57.5	48.5
	10	$\hat{I}_2$	0.049	89.6	93.0
Multiclase	50	$\hat{I}_1$	0.018	91.7	92.0
	20	$\hat{I}_2$	0.018	97.5	97.5
	10	$\hat{I}_1$	0.018	96.6	98.0

Tabla 3.11: PPCC por una RNC en el conjunto de entrenamiento y en el de prueba. Se consideran 50, 20 y 10 clases, utilizando un enfoque OVA y un enfoque de salidas multiclase. Los resultados corresponden a la normalización y constante de aprendizaje que tuvieron el mejor desempeño.

datos en forma de barras para saber las regiones donde la RNC clasifica correctamente los patrones, como se puede ver en la Figura 3.25. Aquí se puede observar lo que ya suponíamos: el comportamiento de las regiones donde existe una mayor cantidad de clasificaciones correctas depende del enfoque usado y es independiente del método de aprendizaje empleado. Con estos resultados nos percatamos de que al usar el enfoque de salidas

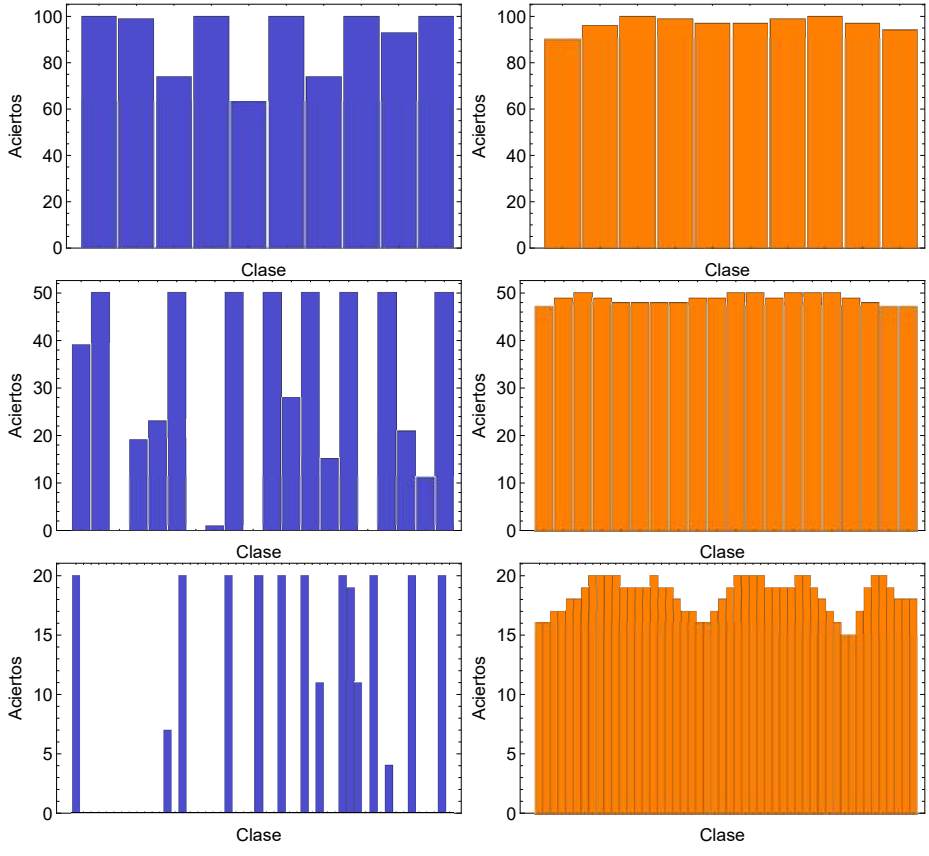

Figura 3.25: Aciertos obtenidos por la RNC, al clasificar los patrones considerando un enfoque OVA (barras azules) y uno de salidas multiclase (barras naranjas). Arriba: 10 clases. Centro: 20 clases. Abajo:50 clases

multiclase, podemos obtener mejores resultados en la mayoría de los ca-

sos estudiados y además se requiere de un menor tiempo de cómputo en los entrenamientos. En la Tabla 3.12, se encuentra la duración del entrenamiento para cada uno de los métodos y para cada enfoque. Al usar el enfoque de salidas multiclase, únicamente se presenta un resultado, ya que en esta situación el número de parámetros es el mismo para cada método, sin importar el número de clases considerado. En el caso del regresor y de la RNA, cuando se usa el enfoque OVA, la duración del entrenamiento incrementa al aumentar el número de clases, como era de esperarse, ya que se aumenta el número de pesos que se entrenan. Para estos métodos, el enfoque multiclase es el que menor tiempo de entrenamiento requiere.
Por otro lado, la duración del entrenamiento cuando se usa la RNC, en promedio es el mismo, sin importar el enfoque empleado. De manera general, la RNC requiere mas tiempo de entrenamiento, mientras que el regresor es el método que usa un menor tiempo de cómputo durante el entrenamiento. Pero a pesar de que la RNC requiere de un mayor tiempo de cómputo, es el método que obtiene el mejor desempeño al clasificar 50 clases diferentes, lo que se traduce en un error de $\pm 1\,\%$ al estimar β. Con estos resultados

Duración de entrenamiento (s)				
Enfoque	Clases	Regresor	RNA	RNC
OVA	50	73	110	85
	20	50	77	84
	10	34	57	82
Multiclase	50	22	39	86
	20			
	10			

Tabla 3.12: Tiempo de cómputo en segundos, requerido durante el entrenamiento para cada uno de los métodos de aprendizaje automático.

podemos decir, que el enfoque de salidas multiclase desarrollado y que se emplea en la mayoría de sistemas físicos analizados en la tesis, es de gran utilidad. Al usar este enfoque, requerimos un menor tiempo de cómputo en el entrenamiento, resultando ventajoso, ya que podemos incluso mejorar el desempeño de los métodos de aprendizaje automático. Adicionalmente, nos puede ayudar que tengamos regiones de clasificaciones correctas loca-

lizadas, como es el caso en el que se usa el enfoque de salidas multiclase, para tratar de aplicar técnicas adicionales y aumentar el desempeño de los métodos.
Existe un gran margen de mejora en los resultados presentados, ya que los parámetros examinados se buscaron de manera manual, por lo que una exploración automática utilizando un código en paralelo para los métodos, ayudaría en la búsqueda de parámetros óptimos. Además, la situación física estudiada involucra sólo una componente del campo eléctrico distinta de cero y no se simularon tantas señales como en otras secciones de la tesis, por lo que en este sentido, también se puede mejorar.

En cada una de las secciones de este capítulo, pudimos estimar algunos parámetros físicos relacionados con las OB como lo es la intensidad del campo eléctrico externo involucrado o las componentes de la matriz de esfuerzos aplicadas al material. En los escenarios estudiados, se aumentó tanto el número de señales usadas, como los datos de entrada utilizados por los métodos de AA para el entrenamiento. Así mismo, vimos cómo el enfoque de clasificación desarrollado tiene un buen desempeño dependiendo de que tanta precisión se desea al estimar tales parámetros físicos.
Además, con algunas simples modificaciones en los métodos de aprendizaje automático, fuimos capaces de clasificar las simulaciones incluso cuando estas se complicaron, dejando claro a lo que se quiere llegar con esta tesis: es sencillo aplicar y modificar las herramientas de AA para clasificar las simulaciones o patrones en prácticamente cualquier problema.

En el próximo capítulo, se emplea el enfoque de clasificación usando salidas multiclase con una técnica adicional para disminuir el error en los parámetros que se están estimando, contrario a algunos de los casos estudiados en este capítulo, donde se aumentó el número de patrones clasificados correctamente, pero a cambio de disminuir la precisión en el parámetro físico de interés.

Capítulo 4

Aplicación a brotes de rayos gamma

Este Capítulo está basado en el artículo titulado *Classifying initial conditions of long GRBs modeled with Radiation Relativistic Hydrodynamics* [56], el cual realicé en conjunto con los Drs. José A. González, Francisco S. Guzmán y Francisco J. Rivera. Adicional al uso del método de salidas multiclase utilizado hasta el momento en la RNA para analizar las señales, se desarrolla un método de refinamiento para incrementar la precisión en los parámetros físicos estimados. Las simulaciones físicas realizadas en este capítulo fueron generadas por el Dr. Francisco J. Rivera

Hay evidencia de que los brotes de rayos gamma (GRBs por sus siglas en inglés) de larga duración son producidos cuando estrellas masivas colapsan, específicamente supernovas con masa mayor a $10M_{\odot}$ [57, 58, 59, 60, 61]. Esta evidencia viene de dos distintas observaciones: (1) los GRBs han sido asociados espectroscópicamente con supernovas de tipo Ic [58]; (2) los GRBs son encontrados en regiones de formación estelar [62, 63, 64]. La conexión entre evidencia observacional y la teoría es importante para explicar los datos y validar modelos. Relacionado a GRBs de larga duración (LGRBs), hay una amplia variedad de modelos propuestos que consideran cualquier número de ingredientes y procesos que toman lugar durante un evento relativista de alta energía [61].
Modelar GRBs, como cualquier otro evento astrofísico, involucra actual-

mente un número de procesos basados en cada vez más complicados conjuntos de ecuaciones, con un número creciente de parámetros que enriquecen los modelos y los escenarios que originaron las observaciones. Hasta ahora, los modelos han sido construidos para inferir propiedades de las fuentes de GRBs y el medio circundante. Estos modelos son en general numéricamente resueltos bajo la aproximación de hidrodinámica ideal, magnetohidrodinámica o hidrodinámica radiativa [65, 66, 67, 68, 69, 70, 71, 72] y en general definen un problema inverso.
Entre los modelos de GRBs hay dos problemas inversos importantes. Primero, el problema del modelo, que consiste en que dado un flujo observado de una curva de luz (CL), uno tiene que determinar el modelo y proceso responsable de los datos observacionales. Segundo, el problema de la causa, que consiste en dada una CL y dado un modelo trabajando bajo algunas suposiciones (un régimen de velocidades, un régimen de densidades, estimación de la masa del progenitor, etc.), uno espera encontrar las condiciones que causaron los datos observacionales.
En este trabajo nos enfocamos en el segundo problema. El modelo puede ser tan completo como para considerar varios procesos de dispersión y elaborar escenarios para los progenitores [71], o incluso un modelo completo 3D que trabaja en varias frecuencias asociadas a diversos procesos que ocurren durante las diferentes fases de evolución de un jet que es candidato para una fuente de GRB [73].

En nuestro caso, asumimos un modelo idealizado en el que CLs observadas de un LGRB es debido a un sólo choque propagándose a velocidad relativista con su gas acoplado al campo de radiación [72]. Este simple modelo soluciona las ecuaciones unidimensionales de hidrodinámica relativista radiativa y la fuente de emisión es caracterizada por condiciones similares a aquellas usadas en las simulaciones de jets relativistas. Además, en nuestro escenario idealizado, sólo consideramos la opacidad asociada a la radiación de Bremsstrahlung con dispersión de Thomson.
El problema importante una vez que hemos escogido el modelo es encontrar las condiciones iniciales que dieron lugar al brote, porque contienen la información acerca de la fuente, de los procesos hidrodinámicos y las condiciones ópticas del material del medio donde el jet se propaga.

La manera tradicional en la que los choques relativistas y jets se fijan

inicialmente, consideran la definición de los parámetros del haz y aquellos del medio circundante. En nuestro caso, ya que el modelo es unidimensional, establecemos que el jet sea un choque relativista con condiciones iniciales similares a aquellas de un choque de tubo, pero involucrando no solo las tres variables hidrodinámicas (densidad, velocidad y energía o presión), pero también la energía y flujo de la radiación involucradas. Esto significa que hay diez parámetros definiendo las condiciones iniciales, cinco para el haz y cinco para los alrededores.

Aunque el modelo es unidimensional y aparentemente muy simple, tiene que tratar con la solución numérica de un conjunto de EDPs no triviales para cada conjunto de condiciones iniciales. Cada solución representa una simulación costosa y en el caso ideal, si uno quiere contrastar el modelo contra una CL observacional de un dado LGRB, el número de simulaciones requerido para explorar el espacio de parámetros debe ser reducido al mínimo. Aquí es donde los métodos de clasificación son de ayuda. En este artículo presentamos el uso de redes neuronales artificiales [74, 75, 76], aplicado a la clasificación de las condiciones iniciales [77, 42] de nuestro modelo de LGRB basado en el choque relativista unidimensional con radiación descrito abajo.

En la proxima sección se presentan los métodos numéricos usados para simular la evolución del jet produciendo un GRB de acuerdo a nuestro modelo y los métodos relacionados a la RNA. Después se presentan los resultados de la clasificación con CLs observacionales y generadas numéricamente, para finalmente presentar las conclusiones.

4.1. Métodos numéricos

4.1.1. Descripción del modelo físico

La evolución de un choque relativista en nuestro escenario, se asume que está gobernado por el sistema hidrodinámico relativista radiativo, el cual es un conjunto de EDPs para la densidad del gas ρ, la presión hidrodinámica P, la velocidad del v o equivalentemente su factor de Lorentz W, la densidad de la energía radiada E_r, la presión de la energía radiada P_r y el flujo

radiado F_r [72]. El sistema de ecuaciones de evolución tiene que ser cerrado con dos ecuaciones de estado, para la hidrodinámica y la radiación. Para la hidrodinámica usamos la ecuación de estado de Taub-Mathews (TM) la cual emula un gas ideal con un índice adiabático asintóticamente igual a 5/3 en las regiones frías, mientras que en las regiones calientes se acerca a 4/3 [78]. Para la radiación usamos la relación de clausura de mínima entropía, porque recupera los dos regímenes de transferencia radiativa, esto es, ópticamente grueso y delgado [79, 80, 81].

Estas ecuaciones son resueltas para condiciones iniciales dadas en dos regiones inicialmente separadas, similar a aquellas del problema de un tubo de choque para la propagación de un choque relativista, pero en este caso una de las regiones estará estratificada. Como se espera que la radiación venga de regiones en las que la profundidad óptica va de valores altos a bajos [82, 83], la primer región corresponde a un estado con variables constantes y juega el rol de la región en donde la energía del brote viene y es inicializada con alta velocidad. Usamos el subíndice b para etiquetar las cantidades en la primer región. La segunda región corresponde al medio circundante el cual asumimos que está en reposo con densidad ρ_m y perfiles de presión P_m que decrecen con el radio [78, 84]. Usamos la etiqueta m para distinguir cantidades en el medio circundante

$$\rho_m = \rho_0 \left(\frac{x_0}{r}\right)^2, \quad P_m = P_0 \left(\frac{x_0}{r}\right)^2, \tag{4.1}$$

donde los parámetros de tal medio son $\rho_0 = n_0 m_p c^2$ con $n_0 = 1\text{cm}^{-3}$ y $P_0 = \rho_0 c^2 10^{-10}$, donde m_p es la masa del protón, x_0 es la locación de la interfaz entre el haz y el medio al tiempo inicial, ρ_0 es la densidad en la interfaz, n_0 es la densidad de protones, c la velocidad de la luz y r es la coordenada radial a lo largo de la propagación del jet, que determina el perfil de densidad del medio circundante.

En todas las simulaciones de este estudio asumimos que el sistema está localmente en equilibrio térmico y que es ópticamente grueso. Entonces, las condiciones iniciales están caracterizadas por un conjunto de cinco variables en el haz $[\rho_b, P_b, W_b, E_{r,b}, F_{r,b}]$ y cinco en el medio circundante $[\rho_m, P_m, W_m, E_{r,m}, F_{r,m}]$. Note que ignoramos P_r en ambos lados porque

estamos usando la relación de clausura M1. También fijamos $W_m = 0$ porque el medio se asume que está en reposo inicialmente.

Para la evolución del sistema, resolvemos el sistema hidrodinámico relativista radiativo (RRH por sus sigla en inglés). El conjunto de ecuaciones de evolución, su correspondencia con la relación de clausura y los métodos numéricos usados para evolucionar el sistema pueden ser encontrados explícitamente en [72].

Finalmente, la luminosidad del proceso se calcula como $L = dE_r/dt$, donde E_r es la densidad de energía radiada medida por un observador situado lejos de la interfaz que separa los dos estados iniciales durante la evolución del proceso. Después, calculamos el flujo radiado dividiendo la luminosidad entre la distancia luminosa d_L para una fuente emisora estática e isotrópica. Entonces el flujo radiado está dado por $F_r = L/4\pi d_L^2$ [85]. Esta cantidad define el proceso de la CL. Por lo tanto, para cada combinación de condiciones iniciales, uno resuelve el problema directo y al final construye una CL numérica que en el caso ideal se ajustará a los datos observacionales de una CL observada. Sin embargo, ajustar una CL observacional requiere explorar el espacio de parámetros con nueve dimensiones (recordar que $W_m = 0$), con cada uno de sus puntos correspondiendo a un conjunto de parámetros de condiciones iniciales. Cada combinación involucra la solución numérica de un sistema complejo de EDPs que consume tiempo. Minimizar el número de simulaciones es el objetivo de usar métodos eficientes para clasificar condiciones iniciales, que ayudarán a localizar las regiones apropiadas del espacio de parámetros que es más posible que contenga los parámetros que se ajustan a una CL dada.

El problema, por lo tanto, involucrará la construcción de un esquema de clasificación en un espacio dimensional alto. Sin embargo, hemos encontrado que los parámetros más influyentes en las CLs son la densidad, velocidad y densidad de energía radiada del haz $[\rho_b, W_b, E_{r,b}]$ y este conjunto de parámetros usaremos para ilustrar nuestro análisis. Tres ejemplos de CLs se muestran en la Figura 4.1, para diferentes combinaciones de valores iniciales de estas cantidades físicas del haz inyectado. Se puede ver que las señales tienen una protuberancia con un resplandor corto. Sin embargo, son

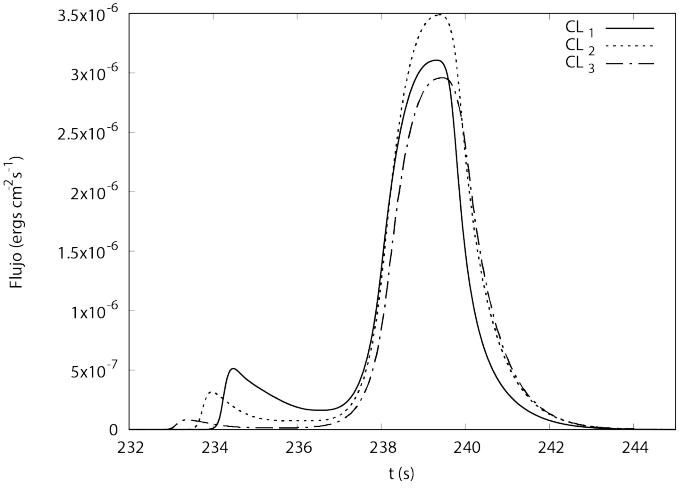

Figura 4.1: Muestra de señales generadas con un conjunto de parámetros iniciales diferentes. Los parámetros usados en estas curvas son: (CL_1) $\rho_b = 0.28\rho_0,\ W_b = 316.23,\ E_{r,b} = 1.66 \times 10^9 \mathrm{erg/cm}^3$, ($\mathrm{CL}_2$) $\rho_b = 0.55\rho_0,\ W_b = 500,\ E_{r,b} = 1.66 \times 10^8 \mathrm{erg/cm}^3$, ($\mathrm{CL}_3$) $\rho_b = 1.11\rho_0,\ W_b = 790.57,\ E_{r,b} = 1.66 \times 10^7 \mathrm{erg/cm}^3$.

considerablemente diferentes en el sentido que la amplitud y el ancho de cada una de ellas cambia cuando los parámetros iniciales cambian también. Algunas señales tienen un pre-resplandor, el cual destaca el tiempo cuando la densidad de energía radiada del haz incrementa.

Sería sencillo encontrar las condiciones iniciales apropiadas para ajustar una CL observacional, siempre que sea posible construir un catálogo de CLs producidas por muchas combinaciones de estos tres parámetros que escogimos. No obstante, la construcción de tal catálogo sería extremadamente lenta sin una selección educada de la región del espacio de parámetros para comenzar. Este es el punto donde una bien educada locación de parámetros no arbitraria pudiera reducir el número de simulaciones para llevar a cabo. El método que proponemos, consiste en construir un educado conjunto de soluciones numéricas del sistema RRH con sus respectivas CLs y después entrenar, validar y probar diferentes RNAs capaces de clasificar los parámetros físicos apropiados para una CL observacional dada.

Lo que se espera ganar en el proceso es la reducción del número de simulaciones requeridas para localizar las condiciones iniciales que se ajustan mejor a una CL dada. En la siguiente sección explicaremos nuestra implementación del método de RNA.

4.1.2. Descripción de la red neuronal artificial

Con el fin de clasificar los datos iniciales, primero fijamos los rangos posibles de valores para los tres parámetros físicos involucrados en la producción de CLs, es decir, $[\rho_b, W_b, E_{r,b}]$. Los rangos escogidos para cada uno de ellos son $\rho_b \in [0.277, 1.61]\rho_0$, $W_b \in [300, 900]$ y $E_{r,b} \in [8.29 \times 10^9, 6.71 \times 10^{11}] \left(\frac{M_\odot}{M}\right)^2 \mathrm{erg/cm^3}$, donde M es la masa del progenitor asociado al LGRB.

Para cada uno de estos rangos, seleccionamos n_p valores equidistantes de cada parámetro y generamos n_p^3 simulaciones numéricas correspondientes a cada combinación de posibles valores. Las CLs obtenidas en las simulaciones son usadas para entrenar la RNA. Cada CL proveerá datos como los mostrados en la Figura 4.1, los cuales son discretizados en 220 intervalos de tiempo, con la idea de que la ventana de tiempo contenga los datos de la CL de la simulación. El valor del flujo radiado se introduce como entrada para la RNA. La red propaga hacia delante los datos de entrada a través de una capa oculta y finalmente a una capa de salida que consiste de tres neuronas, cada una de ella normalizada para tener un valor entre cero y uno.
Asociamos cada una de estas salidas con una clase de n_c posibles para cada una de los parámetros con $n_c \leq n_p$. Básicamente, si la primer salida de la RNA tiene un valor entre cero y $1/n_c$, la red está prediciendo que el valor de ρ_b usado para generar esa CL tiene un valor en la primer clase de su rango. Para ilustrar esto, si $n_c = 3$ y las tres salidas de la RNA son $(0.4, 0.1, 0.8)$, quiere decir que las etiquetas de las clases correspondientes son (2,1,3). Los valores físicos asociados a estas etiquetas están entonces entre los valores $\rho_b = (0.94 \pm 0.22)\rho_0, W_b = 400 \pm 100$ y $E_{r,b} = (5.61 \times 10^{11} \pm 1.11 \times 10^{11}) \left(M_\odot/M\right)^2 \mathrm{erg/cm^3}$, donde la incertidumbre asociada se calcula como la mitad de la longitud de la caja.

Para entrenar la RNA con propagación hacia delante, usamos un algoritmo retroalimentado fuera de línea que minimiza una función de costo del tipo de la Ec. (2.8) para un total de 3 salidas.

Dividimos las n_p^3 simulaciones en los conjuntos de entrenamiento, validación y prueba: el conjunto de entrenamiento contiene el sesenta por ciento de las simulaciones, el conjunto de validación contiene veinte por ciento de las simulaciones y por último el conjunto de prueba, el cual contiene el veinte por ciento restante de las simulaciones.

Con el fin de reducir el tamaño de la incertidumbre de la predicción de los parámetros, una vez que la red ha seleccionado un intervalo dado para los tres parámetros físicos, realizamos otro conjunto de simulaciones contenidas dentro de tal caja seleccionada. Esto significa que producimos otro conjunto de n_p^3 simulaciones entre la caja seleccionada de condiciones iniciales y repetimos el proceso de clasificación, pero note que esta vez en una caja más pequeña en el espacio de parámetros. Este proceso de refinamiento se ilustra en la Figura 4.2 para tres niveles de refinamiento. Cada vez que repetimos este proceso, la incertidumbre de las condiciones iniciales decrece. En los resultados presentados en la próxima sección aplicamos este proceso tres veces.

4.2. Resultados

Para encontrar un desempeño óptimo de la RNA, exploramos diferentes topologías, cambiando la constante de aprendizaje y el número de neuronas ocultas. Entrenamos la red durante 2×10^4 iteraciones para todas las diferentes configuraciones. Escogimos dos diferentes constantes de aprendizaje ($\gamma = 1 \times 10^{-3}$ y $\gamma = 2 \times 10^{-3}$). Combinamos cada una de ellos con 5, 10, 20 y 40 neuronas ocultas, una a la vez. Llevamos a cabo dos experimentos para probar la eficiencia de cada una de las topologías mencionadas arriba. En el primero, dividimos los rangos de la densidad de masa en reposo, factor de Lorentz y densidad de energía radiada en $n_p = 12$ valores equidistantes para cada parámetro, creando un total de $12^3 = 1728$ CLs. Con estos 12 valores, analizamos 3, 6 y 12 clases para cada parámetro. En la Tabla 4.1 mostramos los porcentajes de clasificaciones correctas para cada escenario

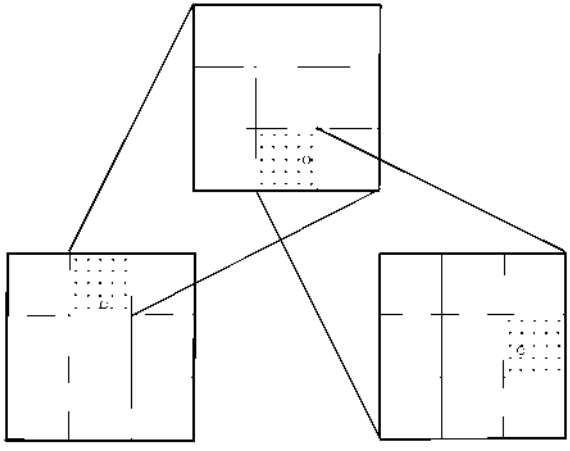

Figura 4.2: Muestreo del espacio de parámetros con refinamiento mostrando el plano bidimensional de parámetros $\rho_b - W_b$. La clasificación se realiza entre veintisiete cubos en realidad. Sin embargo, mostramos aquí la proyección en dos dimensiones con $n_c^2 = 9$ cajas cuadradas. Cada punto representa una simulación con un conjunto dado de condiciones iniciales $(\rho_b, W_b, E_{r,b})$. En este ejemplo hay tres niveles de refinamiento de las muestras. La predicción de la RNA es la que ajusta mejor los parámetros en la caja con coordenadas $(2, 3)$ en el primer refinamiento (fondo a la izquierda), la caja $(2, 1)$ en el segundo refinamiento (centro superior) y la caja $(3, 2)$ en el tercer refinamiento (fondo a la derecha).

con dos distintos valores de la constante de aprendizaje.

Podemos ver que si el número de neuronas incrementa, la precisión de la predicción incrementa, pero también el costo computacional como se puede ver en la Tabla 4.3. Este comportamiento ocurre en general para todos los escenarios independientemente del número de clases. Para los mismos parámetros de la red, si usamos un número más pequeño de clases, la precisión de la predicción incrementa pero también la incertidumbre de los parámetros estimados relacionados con el tamaño del intervalo usado para clasificar.

En el segundo experimento, dividimos los rangos de los parámetros físicos, en $n_p = 16$ valores equidistantes de cada parámetro, generando un total de $16^3 = 4096$ CLs. Con estos 16 valores formamos 4, 8 y 16 clases para cada parámetro. Las predicciones para este experimento se muestran en la Tabla 4.2. Con en el experimento previo, la precisión de la red incrementa cuando aumentamos el número de neuronas ocultas. Vale la pena notar que esta mejora en la precisión se alcanza en ambos experimentos con un número de neuronas ocultas entre 20 y 40. También mostramos el tiempo computacional requerido para la clasificación en la Tabla 4.3.

Comparando estos dos experimentos, podemos ver que las predicciones en el segundo experimento son menos precisas que en el primero. También, en las Tablas 4.1 y 4.2, podemos notar que el número de clasificaciones correctas es más grande para el experimento uno que para el dos. Sin embargo, en el primero (segundo), el tamaño del intervalo usado para clasificar es mayor (menor), lo que implica que las incertidumbres de los parámetros son mayores (menores).

Después de que se entrena la red, evaluamos su desempeño usando el conjunto de predicción. Los datos correspondientes a este conjunto se propagan a través de la red usando los pesos entrenados y encontramos que la red clasifica correctamente los datos para cada salida por lo menos 84.8 % de las veces, como se puede observar en la primera fila de la Tabla 4.5. A fin de cuantificar la calidad de nuestras predicciones, también preparamos dos pruebas basadas en las CL numéricas y observacionales.

		Clases								
		3			6			12		
γ	H	ρ_b	W_b	$E_{r,b}$	ρ_b	W_b	$E_{r,b}$	ρ_b	W_b	$E_{r,b}$
0.001	5	80.6	77.1	96.2	63.4	62.6	85.7	55.3	35.3	60.2
	10	98.2	84.3	96.2	72.4	66.9	89.2	78.8	45.2	68.9
	20	97.3	88.9	96.8	71.8	79.1	95.9	74.2	55.3	79.4
	40	91.8	88.6	97.3	76.8	77.6	97.3	82.0	50.1	86.3
0.002	5	84.6	86.3	95.6	63.1	52.4	78.2	54.2	32.7	61.1
	10	98.2	84.0	95.0	78.8	70.7	91.0	73.6	49.5	75.6
	20	96.2	86.9	96.2	81.4	78.8	96.5	85.5	53.9	88.6
	40	97.6	87.5	99.4	80.2	84.0	97.9	90.1	55.9	87.8

Tabla 4.1: Porcentaje de clasificaciones correctas para cada parámetro físico, considerando dos valores de la constante de aprendizaje γ, el número de neuronas ocultas H y el número de clases para el experimento uno, donde los parámetros físicos fueron divididos en 12 valores equidistantes.

Como mencionamos en la sección previa, para reducir la incertidumbre asociada a los parámetros estimados, realizamos una segunda clasificación asumiendo una caja más pequeña de valores de parámetros físicos centrados en la clase predicha. Después un nuevo conjunto de n_p^3 CLs son preparadas para la RNA, donde los resultados son presentados en las Tablas 4.4 y 4.5 para diferente número de neuronas ocultas y constante de aprendizaje.

4.2.1. Resultados con CLs numéricamente generadas

Producimos numéricamente dos CLs con la única condición de que los parámetros a generar tales CLs queden dentro de la región conteniendo los parámetros usados para entrenar la red. Los datos específicos usados están presentes en la Tabla 4.6. En ambas pruebas, fijamos la masa del progenitor en $10M_\odot$.

Recordamos que la caja de parámetros tiene valores de $(\rho_p, W_b, E_{r,b}) \in [0.277, 1.61]\rho_0 \times [300, 900] \times [8, 29 \times 10^9, 6.71 \times 10^{11}](M_\odot/M)^2$. Asumiendo $n_p = 9$ y $n_c = 3$, las condiciones iniciales para el caso 1 en la Tabla 4.6

		Clases								
		4			8			16		
γ	H	ρ_b	W_b	$E_{r,b}$	ρ_b	W_b	$E_{r,b}$	ρ_b	W_b	$E_{r,b}$
0.001	5	79.8	81.7	93.5	56.8	58.6	83.4	49.0	30.9	65.2
	10	82.1	86.4	98.0	61.4	62.1	84.0	68.1	40.7	73.6
	20	85.8	86.3	96.0	69.1	69.2	88.2	67.5	46.2	74.0
	40	85.6	86.9	97.6	71.9	70.8	91.0	76.3	52.5	83.1
0.002	5	81.3	84.5	94.6	70.6	58.7	73.7	56.4	34.5	51.7
	10	85.1	85.3	95.1	65.8	64.1	79.5	67.0	36.9	77.6
	20	85.7	87.4	98.9	76.2	68.2	91.8	79.2	46.3	77.9
	40	81.0	87.6	97.9	72.5	64.1	89.6	78.6	49.5	83.5

Tabla 4.2: Porcentaje de clasificaciones correctas para cada parámetro físico para dos valores de la constante de aprendizaje γ, el número de neuronas ocultas H y el número de clases para el segundo experimento, donde los parámetros físicos fueron divididos en 16 valores equidistantes.

	Tiempo computacional (s)	
H	$n_p = 12$	$n_p = 16$
5	8167	19540
10	16400	40117
20	32395	81946
40	74282	174554

Tabla 4.3: Tiempo computacional medido en segundos para cada estructura de la red, cuando $n_p = 12$ y 16.

H	ρ_b %	W_b %	$E_{r,b}$ %	Tiempo computacional (s)
1	36.5	33.7	37.2	700
2	48.9	33.7	75.1	1411
3	88.9	33.7	98.6	2189
4	95.1	64.1	99.3	2843
5	97.2	72.4	98.6	3536
6	97.9	70.0	100	4384
7	95.1	73.1	97.9	4877
8	97.9	76.5	100	5802
9	97.9	73.1	100	6513
10	96.5	78.6	100	6899
20	98.6	80.6	100	14169
40	97.2	82.7	100	28325
60	97.9	84.8	100	42971
80	97.2	84.1	100	56713
100	97.9	84.8	100	70324

Tabla 4.4: Porcentaje de clasificaciones correctas para cada parámetro físico usando una constante de aprendizaje $\gamma = 0.001$ y variando el número de neuronas ocultas H para $n_c = 3$ y $n_p = 9$, incluyendo el tiempo computacional medido en segundos usado por la red para el primer refinamiento.

γ	H	ρ_b	W_b	$E_{r,b}$
0.0025	5	95.1	84.8	98.6
	20	98.6	85.5	100.0
	60	97.9	91.7	100.0
0.00075	5	97.2	72.4	98.6
	20	96.5	77.2	99.3
	60	97.2	82.0	100.0

Tabla 4.5: Porcentaje de clasificaciones correctas para cada parámetro físico variando la constante de aprendizaje γ y considerando H neuronas ocultas para $n_c = 3$ y $n_p = 9$.

Prueba	ρ_b (g/cm^3)	W_b	$E_{r,b}\ \left(\frac{M_\odot}{M}\right)^2$ (erg/cm^3)
1	$1.54\rho_0$	790.5	6.63×10^{11}
2	$1.58\rho_0$	301.5	1.66×10^{11}

Tabla 4.6: Parámetros iniciales para dos condiciones iniciales particulares cuyas, CLs se usan para determinar la precisión del esquema de clasificación.

pertenecerían a la caja (3,3,3) dentro del primer refinamiento y las cajas (3,2,3)-(2,3,3) en el segundo y tercer refinamiento. Entonces, el clasificador perfecto debería encontrar que la CL fue generada con parámetros dentro de la secuencia de cajas (3,3,3)-(3,2,3)-(2,3,3) para la prueba 1. Por otro lado, para la prueba 2 la secuencia debería de ser (3,1,1)-(3,1,3)-(3,1,1).

La RNA predijo las siguientes secuencias de cajas (3,3,3)-(2,1,3)-(1,3,3) y (3,1,1)-(3,1,3)-(3,1,1) para las pruebas 1 y 2 respectivamente. Presentamos en la Tabla 4.7, los valores explícitos de los parámetros y sus incertidumbres, definidos como la mitad de la longitud de la clase en el tercer refinamiento.

Prueba	ρ_b (g/cm^3)	W_b	$E_{r,b}\ \left(\frac{M_\odot}{M}\right)^2$ (erg/cm^3)
1	$1.34 \pm 0.024\rho_0$	755.55 ± 11.11	$6.59 \pm 0.123 \times 10^{11}$
2	$1.59 \pm 0.024\rho_0$	311.11 ± 11.11	$1.68 \pm 0.123 \times 10^{11}$

Tabla 4.7: Predicciones obtenidas con la RNA después de tres niveles de refinamiento para cada prueba.

Para ilustrar escogimos la prueba 1 como ejemplo, mostrando que la RNA falla en clasificar en el segundo refinamiento, donde la densidad falla por un considerable 15 %. La prueba 2 es un ejemplo de una clasificación correcta, donde la densidad muestra un error menor a 1 %. En la Figura 4.3 comparamos la CL original con la predicha por la RNA al clasificar los parámetros de estas dos pruebas.

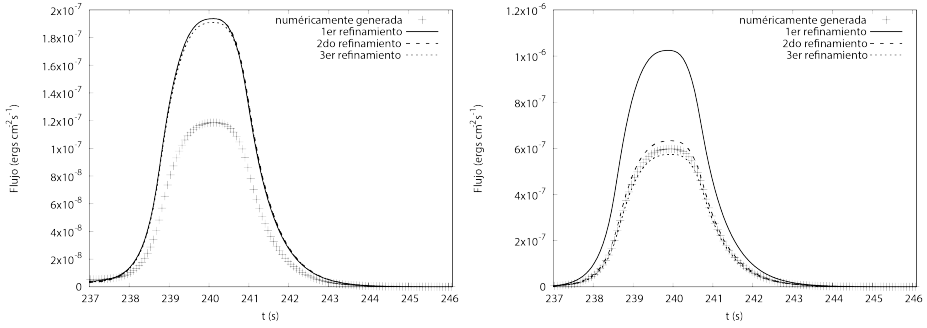

Figura 4.3: La CL generada numéricamente para la prueba 1 (izquierda) y la prueba 2 (derecha), comparada con la predicción de la RNA. El primer caso ilustra uno de los escenarios donde la RNA falla en clasificar las condiciones iniciales en el segundo refinamiento. La segunda prueba ilustra un caso exitoso de clasificación.

4.2.2. Resultados con GRBs observados

Antes de clasificar los parámetros, preparamos los datos de las CLs como series temporales que alimentarán nuestra RNA. Para esto, tomamos datos observacionales, como aquellos en la Figura 4.4 y los convertimos a unidades de código. Ajustamos los datos con un polinomio de alto orden que usamos para generar una serie de tiempo en un dominio temporal discreto con resolución uniforme Δt usados para las curvas de luz que generamos numéricamente en la subsección previa. Cuando este proceso está hecho, introducimos los datos en la RNA entrenada y se procede a la clasificación.

Como inicializamos los pesos de las RNAs con números aleatorios, las predicciones de la red pueden cambiar cuando los pesos iniciales son diferentes; esto implica que el entrenamiento depende de los pesos iniciales. A fin de superar la dependencia de la aleatoriedad, entrenamos diez redes con diferentes pesos iniciales. Para saber cual predicción fue la mejor, las diez redes fueron entrenadas y las coordenadas que fueron predichas más veces y aquellas que ajustan mejor en un refinamiento dado, fueron seleccionadas para el próximo refinamiento. Usamos tres refinamientos y las predicciones para cada una de las curvas de luz observadas son mostradas en la Tabla

4.8. Los GRBs son: GRB051111, GRB060206, GRB060904B,GRB070318 y GRB080413B. Estos GRBs de larga duración fueron tomados en una banda de energía de $15 - 150$ KeV de [86].

En la Figura 4.4 mostramos los ajustes para estos cinco LGRBs. Los parámetros ρ_b, W_b, $E_{r,b}$ usados para producir estas gráficas son aquellos del centro de las cajas seleccionadas por la RNA. En el primer caso, mostramos cómo en el primer refinamiento el ajuste es mejor que los refinamientos subsecuentes. Este es un método para bloquear más refinamientos. Las siguientes tres curvas muestran un ajuste razonable con el tercer refinamiento. Finalmente el ajuste de la quinta curva es una falla en la clasificación.

4.3. Comentarios finales

Hemos presentado una herramienta que clasifica las curvas de luz generadas por un sólo jet relativista unidimensional producido por diferentes valores de la densidad de masa en reposo, factor de Lorentz y energía radiada del haz, basados en un modelo hidrodinámico relativista radiativo.

Como se mostró en [72], el modelo usado para generar los jets es uno simple, pero de ayuda, ajustando la amplitud del flujo de LGRB de curvas de luz. Aunque el modelo no contiene todos los ingredientes de los casos más sofisticados, ayuda a ilustrar la utilidad de las RNAs como clasificador de condiciones iniciales en este especifico escenario astrofísico.

Probamos nuestro método usando CLs numéricamente generadas y obtenemos un 84.8 % de precisión en la clasificación como se muestra en la Tabla 4.5. Mostramos la prueba 1 como caso donde la RNA falla en clasificar apropiadamente. También aplicamos el método a CLs observacionales, donde el método fue correcto en cuatro de cinco casos analizados. Finalmente, debido a que la incertidumbre en los parámetros depende del refinamiento del espacio de parámetros, se espera que al incrementar el número de refinamientos, la calidad del ajuste mejore.

Utilizando una sola opacidad asociada con un sólo mecanismo de emisión, en nuestro caso, radiación de Bremsstrahlung y dispersión de Thom-

Refinamiento	$\rho_b(\mathrm{g/cm^3})$	W_b	$E_{r,b}(\mathrm{erg/cm^3})$	$M(M_\odot)$
GRB051111				
$(3,3,3)$	1.39 ± 0.22	800 ± 100	5.61 ± 1.11	25
$(1,3,3)$	1.24 ± 0.073	866.6 ± 33.33	6.34 ± 0.368	25
$(1,2,3)$	1.19 ± 0.024	866.6 ± 11.11	6.59 ± 0.123	25
GRB060206				
$(3,1,3)$	1.39 ± 0.22	400 ± 100	5.61 ± 1.11	12.5
$(3,3,1)$	1.54 ± 0.073	466.6 ± 33.33	4.87 ± 0.368	12.5
$(3,1,3)$	1.59 ± 0.024	444.4 ± 11.11	5.11 ± 0.123	12.5
GRB060904B				
$(3,1,3)$	1.39 ± 0.22	400 ± 100	5.61 ± 1.11	15
$(3,3,1)$	1.54 ± 0.073	466.6 ± 33.33	4.87 ± 0.368	15
$(3,1,3)$	1.59 ± 0.024	444.4 ± 11.11	5.11 ± 0.123	15
GRB070318				
$(3,3,3)$	1.39 ± 0.22	800 ± 100	5.61 ± 1.11	20
$(1,3,3)$	1.24 ± 0.073	866.6 ± 33.33	6.34 ± 0.368	20
$(1,3,3)$	1.19 ± 0.024	888.8 ± 11.11	6.59 ± 0.123	20
GRB080413B				
$(3,1,1)$	1.39 ± 0.22	400 ± 100	3.4 ± 1.11	10
$(3,1,3)$	1.54 ± 0.073	333.3 ± 33.33	1.92 ± 0.368	10
$(1,2,1)$	1.49 ± 0.024	333.3 ± 11.11	1.68 ± 0.123	10

Tabla 4.8: Parámetros del jet inicial. Con el fin de que los valores de la densidad de masa en reposo y la densidad de energía radiada sean correctamente escalados, deben ser multiplicados por ρ_0 y $\left(\frac{M_\odot}{M}\right)^2 \times 10^{11}$ respectivamente.

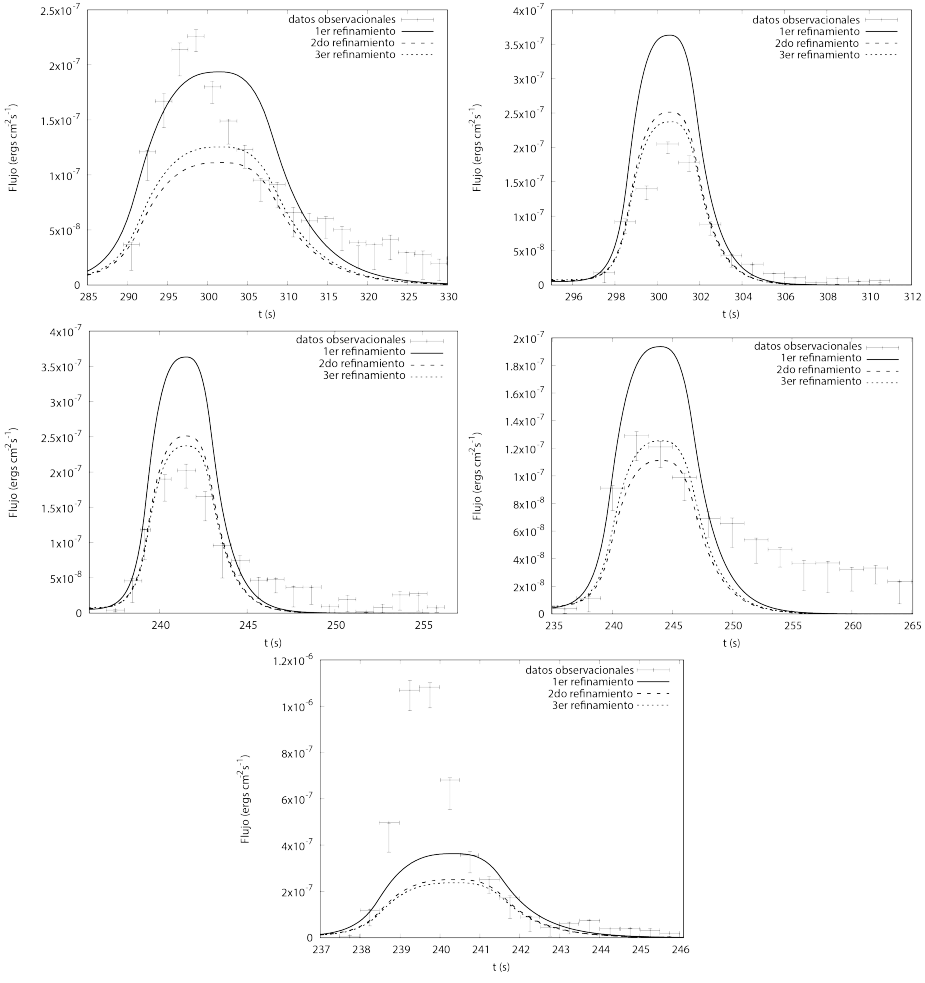

Figura 4.4: Los datos de las curvas de luz de cinco LGRBs, de izquierda a derecha y de arriba a abajo: GRB051111, GRB060206, GRB060904B, GRB070318, and GRB080413B respectivamente. Los datos representados por puntos con barras de error fueron tomados de [86]. Las tres líneas de cada gráfica corresponden a cada predicción en el primer, segundo y tercer refinamiento.

son, no pueden explicar los dos regímenes de la CL. Un escenario más realista incluiría el proceso inverso de Compton actuando en un régimen de energía más alto y el canal de Thomson en el régimen de baja energía. Esto puede mejorarse usando un modelo multibanda o multifrecuencia que puede tomar en consideración dos o más proceso con el mismo código (e.g. [73]), un caso que implementaremos en el futuro.

En conclusión, nuestro método provee una manera sencilla para rastrear los parámetros con una precisión dada. Problemas astrofísicos similares pueden requerir la solución de ecuaciones de evolución en tres dimensiones, que requieren un considerable poder de cómputo y entonces la exploración del espacio de parámetros sería muy costoso. En tal escenario, nuestro enfoque puede mostrar su fuerza al usar una exploración sistemática, la cual se espera que sea más eficiente que usar la fuerza bruta y por lo tanto una exploración costosa. Estamos conscientes de las limitaciones de nuestro método, algunas de ellas incluyen la necesidad de producir un considerable conjunto de simulaciones que eventualmente pueden ser computacionalmente prohibitivos y las posibles fallas en zonas del espacio de parámetros donde el problema podría parecer degenerado para la red. Sin embargo, este tipo de enfoques son esenciales al solucionar problemas inversos involucrando modelos relacionados a ecuaciones diferenciales parciales.

Con el enfoque de refinamiento presentado en este capítulo, tenemos una herramienta adicional para el método de clasificación por salidas multiclase, con el cual se puede disminuir el error asociado a los parámetros que se estiman, pero a cambio de utilizar un mayor número de simulaciones para entrenar la RNA.
En el próximo capítulo se utilizan métodos de AA en sistemas físicos con posibles aplicaciones industriales, donde también se desarrolla una herramienta que nos ayuda a identificar en el espacio de parámetros a estimar, donde es más común que los métodos de AA clasifiquen incorrectamente los patrones.

Capítulo 5

Aplicación en Física de fluidos

Estudiar las obstrucciones en tuberías transportando fluido es de gran importancia, ya que es un problema de gran impacto en la sociedad moderna al estar presente en la vida cotidiana de las personas. Esto puede verse en las redes de tuberías urbanas, salud pública [90] o en la ingeniería industrial [91]. Las tuberías son la forma más usual de transportar fluidos en las industria energética [92], química [93], manufacturera [94, 95] y de agua [96], así como en edificios y en el manejo de aguas residuales [97, 98, 99] entre muchas otras.
El flujo constante del fluido es crucial tanto en la industría, como incluso en nuestro organismo, ya que por ejemplo, cuando existen bloqueos de conductos biológicos como las arterias, pueden causar lesiones graves o incluso la muerte. Por esto, es importante la detección de objetos obstruyendo el flujo, así como saber la forma de tal obstrucción para prevenir perdidas económicas o accidentes.
Para abordar este problema, investigaciones como las presentes en [100, 101, 102] han propuesto diferentes metodologías para identificar obstáculos, fugas o defectos dentro de tuberías urbanas e industriales.

En este capítulo, utilizamos los métodos de AA para afrontar estas problemáticas al tratar de predecir la morfología, posición y tamaño de una obstrucción dentro de una tubería por la que se transporta un fluido,

mediante la simulación del fluido en dos dimensiones sujeto a diferentes condiciones iniciales.

5.1. Morfología de una obstrucción

Esta sección está basada en el artículo titulado *Recognition of an obstacle in a flow using artificial neural networks* [87] el cual fue publicado en colaboración con los Drs. Mauricio Carrillo, José A. González y Silvano U. Que. A diferencia de los demás capítulos, en esta ocasión la red neuronal artificial que se implementó no se usó como clasificador, por lo que el esquema de salidas multiclase no es requerido. El objetivo de este trabajo es reconocer la forma y localización de un obstáculo obstruyendo una tubería, cuyas dimensiones fueron seleccionadas considerando tuberías que transportan fluidos para uso industrial y de sistemas urbanos. Con este fin, se entrenaron RNAs usando como datos de entrada, información física del flujo, el cual fue simulado por el Dr. Silvano Ulices Que utilizando un código numérico 2D lattice-Boltzmann [88, 89].

Nuestro enfoque toma en consideración diferentes escenarios: cambios en el diámetro y localización del obstáculo, viscosidad y velocidad inicial del flujo, obteniendo información física relevante tal como la velocidad, vorticidad o presión dinámica del fluido a lo largo del dominio numérico. En particular se considera la componente x del campo de velocidad del fluido (v_x) o la presión dinámica (q) como la información fundamental para ser analizada por la RNA. Se seleccionó una proporción entre el ancho de la tubería y el obstáculo inmerso desde 1/80 hasta casi el valor de 1, donde el flujo es interrumpido por obstrucciones que pueden ir desde pequeños bloqueos, hasta la obstrucción completa de la tubería.

En la próxima subsección se presenta una descripción detallada del problema de bloqueo en las simulaciones con el método de lattice-Boltzmann.

5.1.1. Simulaciones numéricas

La simulación del flujo en dos dimensiones alrededor del obstáculo fue elaborada implementando un código numérico basado en el método de lattice Boltzmann (MLB o LBM por sus siglas en inglés) al igual que en [103].

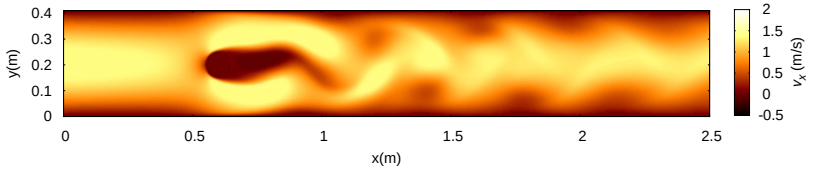

Figura 5.1: Magnitud de v_x para el flujo alrededor de un obstaculo cilíndrico con un flujo entrante de Poisseuille de $v_c = 1.5$m/s. El obstáculo está fijo en $x = 0.6$m y el fluido fluye en dirección x positiva a lo largo del tubo. Note que los vórtices formados después del obstáculo, son llevados a lo largo de la dirección del flujo, generando la característica calle de vórtices de Karman.

El MLB es muy popular porque es fácil de implementar y tiene una gran capacidad para llevar a cabo simulaciones computacionales en una amplia variedad de problemas físicos [104, 105, 106], principalmente aplicado en la dinámica de fluidos [107, 108].

Para los casos de estudio consideramos un obstáculo cilíndrico inmerso en un medio infinito (flujo libre), con un flujo moviéndose en la dirección positiva x. El obstáculo cilíndrico es perpendicular al plano $x - y$, por lo que es representado por un círculo en la simulación bidimensional. Se imponen condiciones de frontera en la salida del flujo lo suficientemente lejos para que los parámetros característicos del flujo no se vean afectados por los cálculos internos [109]. Para las paredes sólidas del tubo y el obstáculo se emplea un condición de frontera de rebote completo [88]. La condición de frontera representando el flujo entrante fue configurada con un perfil de velocidad dado como entrada al modelo numérico, tal que el flujo después del obstáculo presenta un patrón relacionado al perfil de velocidad de entrada, el diámetro y la localización del obstáculo.

Definimos β como el valor de la proporción entre el diámetro del obstáculo cilíndrico y el ancho de la tubería. El diámetro del obstáculo se varía para valores de β que van desde $\beta = 0.0122$, representando bloqueos pequeños, hasta valores cercanos a $\beta = 1$, representando obstrucciones de casi el diámetro del tubo. Para el dominio de la simulación, usamos una malla

de $165 \times 1,000$. Aunque el código del MLB tiene unidades adimensionales, el sistema se adaptó para tener dimensiones físicas siguiendo la siguiente referencia [110]. Se escogieron las unidades físicas tales que el dominio numérico corresponde a una longitud total de $L_y = 0.41$m en la dirección vertical y $L_x = 2.5$m en la horizontal. Consideramos un flujo entrante de Poiseuille en un régimen estacionario con una densidad de $\rho = 10^3 \text{kg/m}^3$, una viscocidad cinemática de $\nu = 10^{-3}\text{m}^2/\text{s}$, como se usó en [103]. La localización de todos los obstáculos estudiados está en $x = 0.6$m del tubo y su posición sobre el eje y será descrita en la próxima subsección.

Se llevaron a cabo una gran variedad de simulaciones numéricas, considerando como parámetros libres el perfil de velocidad entrante, el diámetro del obstáculo (cambiando el valor de β) y la posición del mismo con respecto al eje y. Un ejemplo de una simulación numérica se muestra en la Figura 5.1, con un obstáculo de tamño $\beta = 0.244$ y un flujo de Poiseuille con una velocidad característica de $v_c = 1.5$m/s. Las simulaciones numéricas se detuvieron hasta que el sistema alcanzó una estabilidad neutral, que ocurre antes de completar 30,000 iteraciones o un tiempo físico aproximado de 16 segundos.

Enseguida se describe la metodología empleada para utilizar los patrones de flujo como datos de entrada para entrenar la RNA.

5.1.2. Metodología

Queremos estimar el tamaño y la posición de un obstáculo midiendo v_x y q en alguna parte a lo largo del tubo, que en este caso se supone que la medición ocurre después de donde se encuentra el obstáculo. Para esto, podemos proponer el caso más sencillo: considerar un sólo sensor y aplicar una regresión lineal (RL) entre el diámetro del obstáculo y la velocidad del flujo en la posición del sensor. Aunque la RL es tan buena como la RNA en este caso, el análisis en otros escenarios muestra que la RNA supera al RL y es por esto que únicamente se presentan los resultados de la RNA. Además, la intención es dar un primer paso para proveer una metodología capaz de estimar múltiples y más complejas morfologías con formas asimétricas. Con este fin, definimos una *región objetivo* alrededor del área donde se encuentran los obstáculos. En esta región, el tamaño

y la posición del obstáculo son estimados en términos de la proporción entre el obstáculo y el fluido de alrededor; esto será explicado en detalle más adelante. También, se colocaron diferentes sensores a través del tubo en distintos sitios de medición a lo largo del eje x, una representación esquemática de esto se muestra en la Figura 5.2. Los sitios de medición están localizados en x =1.75m, 2.10m, 2.45m, a los cuales nos referimos como A, B y C respectivamente.

Con esto en mente, las RNAs fueron escogidas porque son flexibles en términos de los datos de entrada y son de fácil implementación, superando a los regresores lineales. En trabajos futuros nos gustaría aumentar la complejidad del problema, por lo que tal vez otros métodos de aprendizaje automático más sofisticados sean requeridos.

A continuación se describen los casos de estudio con las estructuras de RNA empleadas, así como los datos de entrada y objetivos utilizados, especificando como se escogen los conjuntos de entrenamiento, validación y prueba.

Casos de estudio

La habilidad de las RNA para predecir el tamaño y posición de los obstáculos se prueba en tres diferentes casos:

1. Se ejecutaron 80 simulaciones numéricas distintas para un obstáculo cilíndrico bidimensional inmerso en el flujo, localizado a la mitad del diámetro del tubo en el eje y. Simulamos obstáculos con diámetros variando de $d = 0.005$m a $d = 0.395$m en pasos de 0.005m, es decir, valores de $\beta = 0.012$ a $\beta = 0.964$ en pasos de 0.0122 y adicionalmente se considera un objeto pequeño de 0.001m. Para este caso, seleccionamos como conjunto de predicción o prueba, a aquellos obstáculos con diámetros que van de $d = 0.02$m a $d = 0.395$m en intervalos de $\Delta d = 0.025$m. El conjunto de validación consiste de obstáculos que van de $d = 0.025$m a $d = 0.375$m también con $\Delta d = 0.025$m. Los 49 obstáculos restantes se usan como conjunto de entrenamiento. Este caso a su vez, es dividido en tres subcasos:

   a) El perfil v_x o q para $t = 16$ segundos al final de la evolución numérica se considera como vector de entrada. Por simplicidad,

Parámetros de los casos estudiados					
Caso	Obstáculos	Posiciones en y	Velocidades	Espacio	Tiempo
1a	80	1	1	83	1
1b	80	1	1	1	300
1c	80	1	1	3	300
2a	4	43	1	83	1
2b	4	43	1	1	300
2c	4	43	1	3	300
3	11	1	12	10	1

Tabla 5.1: Parámetros usados en las simulaciones para los diferentes casos estudiados. La columna llamada Obstáculos representa el número de diámetros diferentes para los obstáculos. La columna Posiciones en y contiene el número de diferentes posiciones del centro del obstáculo a través del eje y. La columna definida como Velocidades, tiene el número de diferentes velocidades de flujo entrante usados en el caso 3. La columna Espacio está relacionada al número de sensores equidistantes implementados en el sitio de medición. Por último, la columna Tiempo corresponde al número de pasos en el tiempo extraídos en la evolución numérica llevada a cabo en cada sensor numérico considerado.

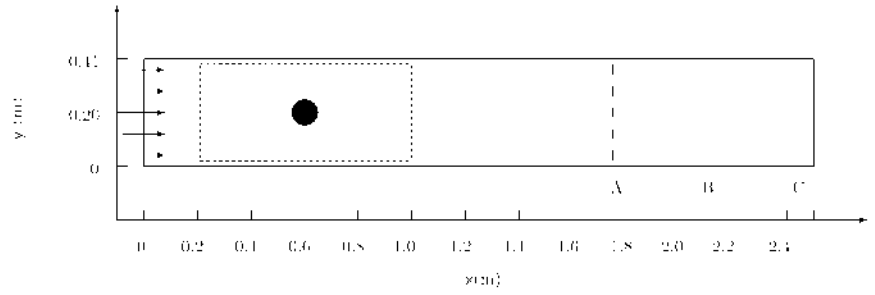

Figura 5.2: Representación esquemática de un tubo en 2D, el obstáculo cilíndrico y diferentes sitios de medición a lo largo del tubo localizados en A = 1.75m, B = 2.10m y C = 2.45m, marcados con líneas punteadas. La región rectangular punteada ilustra el área usada como objetivo para la RNA.

en términos de la estructura de la RNA, la información se extrae de solo 83 de los 165 nodos de la malla numérica. Este enfoque se utiliza para estudiar si la información física extraída, tal como v_x o q, disponible a un tiempo y distancias fijos, es suficiente para dar una estimación apropiada del tamaño del obstáculo.

b) La evolución de v_x o q para 300 pasos de tiempo, de $t = 0$s a $t = 16$s medidos en un sólo sensor en el centro del tubo ($y = 0.210$m), se considera como entrada para la RNA, usando la simetría del conducto. Con esto, inspeccionamos el límite de las predicciones considerando el menor número de sensores posibles, con la ventaja que puede tomar varias mediciones para un lapso fijo.

c) El mismo procedimiento que en el caso $1b$, pero adicionalmente agregamos 2 sensores más equidistantes entre sí, es decir, tenemos tres sensores localizados en $y = 0.105$m, 0.210m y 0.315m. Con este caso, estudiamos si al incrementar el número de senso-

Parámetros para el caso 3		
β	Velocidad (m/s)	Sensor (Localización)
0.0122	0.15	11 (0.025m)
0.0976	0.30	27 (0.065m)
0.1952	0.45	43 (0.105m)
0.2928	0.60	59 (0.145m)
0.3904	0.75	75 (0.185m)
0.4880	0.90	91 (0.225m)
0.5856	1.05	107 (0.265m)
0.6832	1.20	123 (0.305m)
0.7808	1.35	147 (0.365m)
0.8784	1.5	163 (0.405m)
0.9638	1.65	
	1.8	

Tabla 5.2: Datos de los diámetros, velocidades de flujo entrante y localización de los sensores usados en el caso 3. La última columna define la localización de los diez sensores sobre el eje y en la malla del LBM. En paréntesis se muestra sus valores equivalentes en unidades físicas. Los valores para v_x y q fueron extraídos de éstos sensores numéricos en $x = 0$m y en $x = 2.10$m, siendo este útimo valor el correspondiente al sitio de medición B.

res, incrementa la precisión en las predicciones.

2. En contraste con el caso 1, consideramos solo tres tamaños diferentes de obstáculos con valores de $\beta = 0.122$, 0.244 y 0.488. Esto es, $d = 0.05\text{m}, 0.1\text{m}$ y 0.2m. En cada simulación, se cambió la posición del obstáculo de entre 43 posiciones diferentes sobre el eje y, tal que el obstáculo puede estar cerca del centro del tubo o cerca de sus paredes, generando un total de 129 simulaciones. Para cada tamaño del obstáculo se escogieron 22 simulaciones para el entrenamiento y la validación y las restantes 21 simulaciones fueron usadas como conjunto de predicción. Para probar la capacidad de la RNA al realizar una estimación para un tamaño desconocido, se incluyeron en el conjun-

to de predicción 21 simulaciones espaciadas equidistantemente sobre el eje y para un obstáculo de diámetro $\beta = 0.366$. Este escenario, fue divido también en tres diferentes subcasos siguiendo la misma descripción que en 1a, 1b y 1c.

3. En este caso, se analiza una situación similar al caso 1 donde el obstáculo está localizado en $y = 0.210$m y $x = 0.6$m, con la diferencia que se ha cambiado el flujo entrante con una velocidad característica de $v_c = 1.5$m/s, usado en los casos 1 y 2, a diferentes valores que van de $v_c = 0.15$m/s a 1.8m/s, en pasos de $\Delta v_c = 0.15$m/s. Este análisis se realiza para examinar una extensión del caso $1a$, con diferentes velocidades de flujo entrante y once distintos tamaños del obstáculo como se muestra en la Tabla 5.2. Adicionalmente, también se explora el comportamiento de la red agregando más información acerca del fluido entrante y menos sensores que en el caso 1. Por simplicidad, consideramos diez valores equidistantes a lo largo del eje y de v_x o q antes del obstáculo, en $x = 0$m de acuerdo al esquema en la Figuras 5.2 y otros diez valores equidistantes de v_x o q en el sitio de medición B, análogo a lo hecho anteriormente en el caso $1a$. Los datos de entrada para las RNA consisten entonces en 20 valores para cada una de las 132 simulaciones producidas. A fin de explorar el desempeño de la red y su dependencia en la elección de las muestras para los conjuntos de entrenamiento, validación y prueba, se seleccionan aleatoriamente los patrones. Los conjuntos de validación y prueba contienen 20 patrones cada uno, mientras que el conjunto de entrenamiento es de 92.

Todos los casos de estudio se encuentran resumidos en las Tablas 5.1 y 5.2.

Región objetivo

Como usamos RNAs entrenadas con un entrenamiento supervisado, se necesita proveer los objetivos, relacionados al tamaño real del obstáculo, forma y localización. Para esto, definimos una región dentro del tubo conteniendo el obstáculo y sus alrededores inmediatos como la región objetivo. Sin embargo, si consideramos cada uno de los nodos de la simulación del

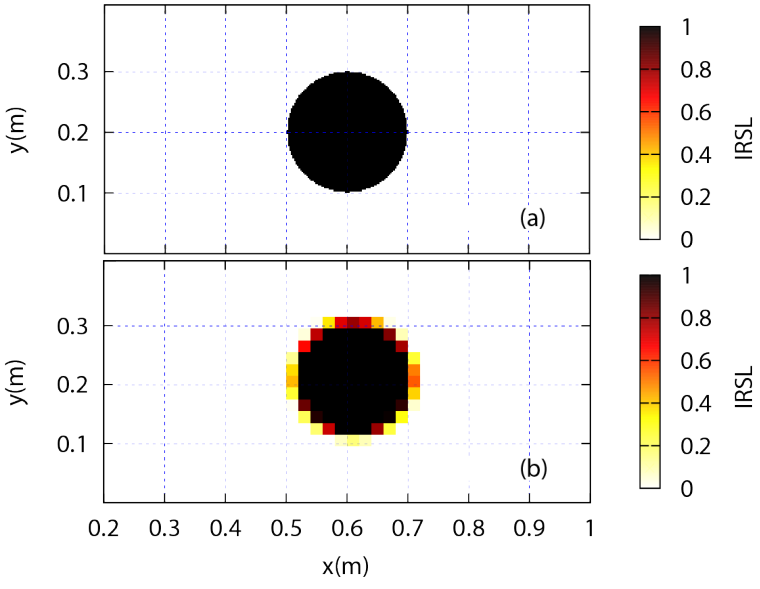

Figura 5.3: En la gráfica (a) están representados los elementos de sólido/fluido en la simulación de la malla del MLB, mientras que en (b) está la transformación de la simulación a la malla objetivo. El número menor de celdas usadas en la malla objetivo en comparación de la malla del MLB causa pérdida en la resolución de la malla. La caja de colores representa la razón entre el sólido y líquido, donde un valor de 0 indica una celda que esta compuesta únicamente de elementos de fluido y un valor de 1 representa una celda ocupada completamente de elementos sólidos. Las líneas de la malla mostradas en las figuras son únicamente para referencia y no representan el tamaño de la celda objetivo.

MLB directamente como los objetivos para la RNA, como se ilustra en la Figura 5.2, tendría un gran número de salidas, haciéndolo computacionalmente costoso. Entonces, como primer enfoque, proponemos considerar una región objetivo que consiste en una malla fija de 40x20 celdas, donde cada celda consiste de 64 nodos de la malla numérica. De ahora en adelante nos referiremos a la región objetivo como la *malla objetivo*.

Adicionalmente, asignamos un valor numérico a cada celda, dependiendo de la relación de nodos que representan elementos sólidos o de fluido en la malla numérica. Esto quiere decir que la proporción de celdas objetivo ocupadas por el obstáculo se representa por el número de nodos sólidos divididos por el número total de nodos de la malla numérica contenidas en esa celda. En otras palabras, hemos definido un índice de ocupación para cada celda de la malla objetivo, donde la relación sólido/líquido de las celdas tiene un valor de 1 si la celda contiene únicamente elementos sólidos y un valor de 0 implica que sólo hay elementos fluidos en la celda. De ahora en adelante, llamamos a este índice el índice de razón sólido/líquido (IRSL). Un ejemplo de la transformación de la simulación del MLB de la malla numérica a la malla objetivo para un obstáculo de tamaño igual a 0.2m se presenta en la Figura 5.3, donde los tonos obscuros hacen referencia a las celdas casi ocupadas por el obstáculo (alto IRSL), mientras que las celdas con tonos más claros indican una mayor proporción de elementos de fluido (bajo IRSL).

Estructura de la red neuronal

Recordemos que para el caso $1a$, sólo la mitad de los nodos espaciales en la dirección y de la malla son seleccionadas como entradas para la RNA, reduciendo el número de puntos donde los campos vectoriales son medidos, de 165 a 83, simplificando los datos de entrada y la estructura de la red, requiriendo así un número menor de cálculos durante el entrenamiento. Entonces, el vector de entrada consiste en 83 valores, que se traducen en 83 neuronas de entrada requeridas por la red, relacionadas con el perfil del fluido en los sitios de medición correspondientes. Por ejemplo, si consideramos los valores de v_x en los sensores en cualquier sitio de medición, el patrón de entrada es de la forma:

$$I = \{v_{x_1}, v_{x_3}, \ldots, v_{x_i}, \ldots, v_{x_{165}}\}. \tag{5.1}$$

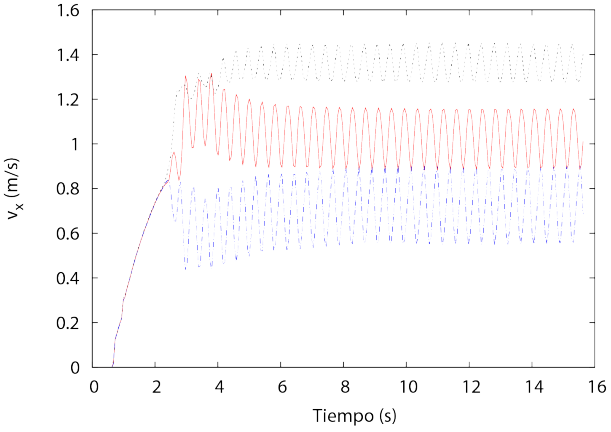

Figura 5.4: Series de tiempo de la componente x de la velocidad del flujo en $y = 0.105$m (línea contínua), 0.210m (línea discontínua) y 0.315m (línea discontínua punteada) en el sitio de medición B, para un obstáculo $\beta = 0.488$ centrado en $y = 0.21$m.

donde $i = 1, 3, ..., 165$ indica el *downsampling* hecho de 165 a 83 para los nodos en la malla numérica del LBM. Para el caso $1b$, el vector de entrada se define por la serie de tiempo de v_x o q en $y = 0.21$m durante 300 pasos de tiempo representados por el índice t. Por ejemplo para v_x:

$$I = \{v_{x_{85},t_1}, v_{x_{85},t_2}, \ldots, v_{x_{85},t_{300}}\}. \tag{5.2}$$

donde $v_{x_{85},t}$ etiqueta el valor de v_x en el nodo 85 de la simulación donde $y = 0.210$m, es decir, en medio del tubo a cierto tiempo t. Para el caso $1c$, las entradas de la RNA consisten en los valores de los 300 pasos en el tiempo para v_x o q en las posiciones: $y = 0.105$m, 0.210m y 0.315m en los sitios de medición considerados. En la Figura 5.4 se presenta la evolución de las series de tiempo para v_x en el sitio de medición B. Las entradas correspondientes a este escenario son:

$$I = \{v_{x_{43},t_1}, v_{x_{85},t_1}, v_{x_{127},t_1}, ..., v_{x_{43},t_{300}}, ..., v_{x_{85},t_{300}}, v_{x_{127},t_{300}}\}. \tag{5.3}$$

En el segundo escenario, las entradas son las mismas que las descritas en las Ecs. (5.1-5.3). Para el caso 3, recordemos que diez valores en $x = 0$m y $x = 2.1$m (sitio de medición B) son considerados como patrones de entrada. Por ejemplo, siguiendo la Tabla 5.2, un patrón de entrada usando v_x se define como:

$$I = \{v_{x_{11}}(x = 0\text{m}), v_{x_{27}}(x = 0\text{m}), ..., v_{x_{163}}(x = 0\text{m}), \\ ..., v_{x_{11}}(x = 2.1\text{m}), ..., v_{x_{163}}(x = 2.1\text{m})\}. \quad (5.4)$$

El mismo conjunto de Ecs. (5.1-5.4) son aplicadas cuando la presión dinámica q es usada en vez de v_x como datos de entrada para la red.
La estructura de la RNA puede diferir en cada caso, es decir, el número de neuronas ocultas y de entrada, pero el número de salidas es constante para todos los casos, ya que la meta es la misma: aproximar la forma de un obstáculo bloqueando el flujo. Recordemos que la región donde el obstáculo está localizado, la describimos con 40x20 celdas, lo que quiere decir que el objetivo y la predicción tienen 800 elementos. En forma vectorizada, para un patrón de entrada I, el resultado calculado por la RNA es $\vec{O} = [O_1, O_2, \ldots, O_{800}]$, y el k-ésimo elemento de este vector se calcula como

$$O_k = F_2\bigg(\sum_{j=1}^{J} \sigma_{jk} F_1\bigg(\sum_{i=1}^{I} \tilde{\sigma}_{ij}\tilde{I}_i + \tilde{\sigma}_{0j}\bigg) + \sigma_{0k}\bigg), \quad (5.5)$$

donde $1 \leq k \leq 800$ hace referencia a la celda de la malla objetivo; $\tilde{I}_i$ es el i-ésimo elemento del vector de entrada y J es el número de neuronas ocultas. F_1 y F_2 son las funciones de activación para las capas oculta y de salida respectivamente; $\tilde{\sigma}_{jk}$ y $\tilde{\sigma}_{0j}$ son los pesos y términos *bias* entre la capa de entrada y salida; σ_{jk} y σ_{0k} son los pesos y términos *bias* entre la capa oculta y de salida.

La implementación de las RNA fue desarrollada desde cero usando Fortran 90. En vez de utilizar códigos de fuente abierta, decidimos usar nuestra propia implementación para tener completo control de los parámetros dentro de la red, buscando diferentes estructuras y parámetros en el proceso de aprendizaje. Por un lado, la selección de la estructura de la RNA fue hecha considerando que se debe mantener lo más simple posible, a fin de mantener ventaja computacional y lo suficiente compleja para adaptarse bien a

patrones desconocidos. Encontramos que las RNAs con una capa de entrada como la definida en las Ecs. (5.1-5.4), una capa oculta con 20 neuronas y una capa de salida con 800 neuronas, fue lo suficientemente compleja para obtener buenos resultados, sin pérdida en el desempeño para todos los casos de estudio. Todas las RNAs usadas tienen funciones de activación sigmoides en las capa oculta y de salida. Por otro lado, las RNAs fueron entrenadas usando un algoritmo retroalimentado [76] para minimizar una función de error cuadrático medio, usando una constante de aprendizaje γ=0.001, con un máximo de 15,000 iteraciones en el entrenamiento y usando una técnica de validación cruzada como criterio de interrupción. Para mayor claridad, todos los resultados mostrados aquí corresponden al conjunto de predicción. Para más detalles acerca del entrenamiento supervisado y del algoritmo retroalimentado, el lector puede consultar [9, 11].

5.1.3. Resultados

Con el objetivo de estimar la precisión de RNA en las predicciones para cada uno de los casos analizados en el conjunto de prueba, empleamos el coeficiente R^2. El cálculo de R^2 fue realizado sobre la malla objetivo y de predicción, considerando el IRSL real y el predicho. R^2 se define de la siguiente manera:

$$R^2 = 1 - \frac{\sum_{i=1}(O_i - \langle T\rangle)^2}{\sum_{i=1}^{800}(T_i - \langle T\rangle)^2}, \tag{5.6}$$

donde T_i y O_i son el i-ésimo objetivo y salida de la RNA respectivamente y $\langle T\rangle$ es el promedio del IRSL para todos los vectores objetivo:

$$\langle T\rangle = \frac{1}{800}\sum_{i=1}^{800} T_i. \tag{5.7}$$

Esto quiere decir que el rango del coeficiente R^2 es $(-\infty, 1]$, donde un valor $R^2 = 1$ implica un emparejamiento perfecto término a término entre el objetivo y la salida de la RNA, mientras que $R^2 \to 0$ indica que la predicción se acerca a $\langle T\rangle$.

Recordemos que las mediciones del fluido consisten en una instantánea del perfil v_x o q del fluido a un tiempo cuando el sistema alcanza una estabilidad neutral. Para el caso descrito en $1\,a$, presentamos los resultados

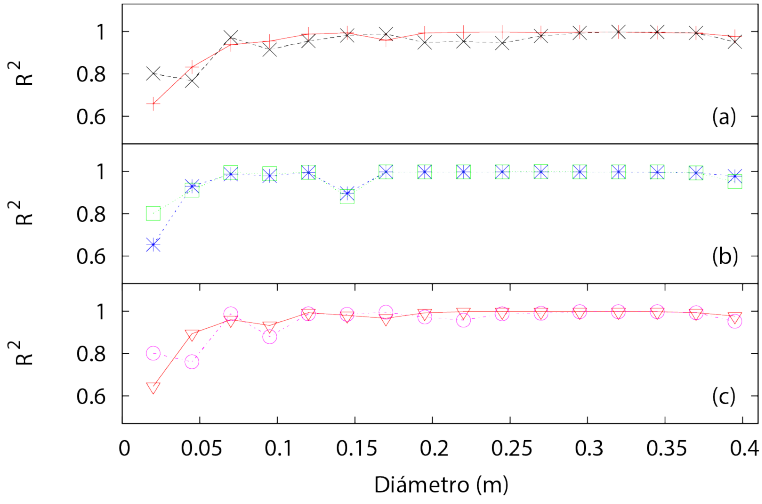

Figura 5.5: Coeficiente R^2 para predicciones producidas por la RNA para obstáculos localizados en el centro del tubo con diferentes diámetros. La RNA se entrena con datos extraídos con un sensor localizado sobre la línea en B y probado con mediciones en A, B y C. En (a), la curva roja (+) muestra los resultados usando v_x y q es representado por la curva negra ($\times$) en el sitio A. En (b), los datos de entrada fueron extraídos en el sitio de medición B y las curvas azules ($*$) y verdes ($\square$) representan R^2 usando los perfiles v_x y q respectivamente. En (c), las curvas rojas ($\triangledown$) y magenta ($\bigcirc$) corresponden a los resultados para v_x y q en el sitio de medición C respectivamente. Todos los coeficientes R^2 se encuentran arriba de 0.6, que puede ser interpretado como una buena correlación entre el objetivo y la predicción. Note como los resultados son independientes del sitio de medición.

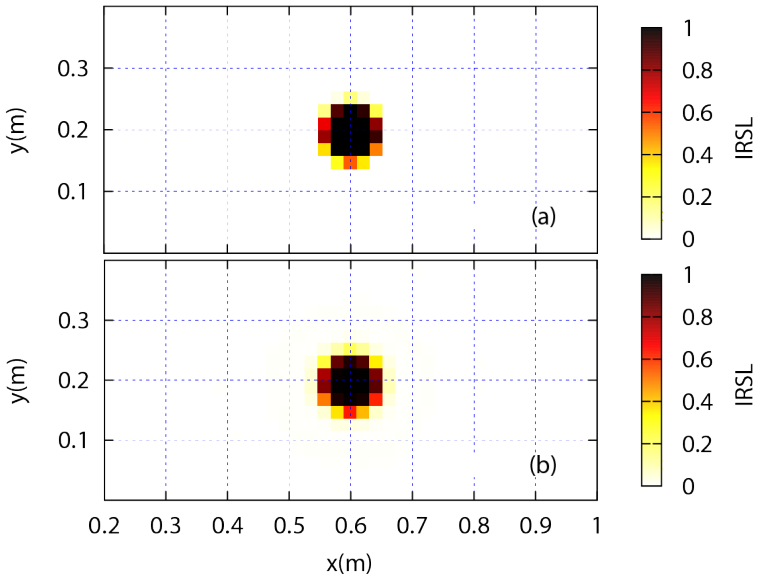

Figura 5.6: Objetivo (a) y predicción de la RNA (b) considerando el perfil v_x del fluido generado por un obstáculo en el centro del tubo con $\beta = 0.4758$. La caja de color representa el IRSL. La diferencia entre el obstáculo objetivo y el predicho es casi imperceptible a simple vista, el coeficiente R^2 ha alcanzado un valor de 0.979.

midiendo en los detectores localizados no sólo en el sitio B, si no también en A y C como se muestra en la Figura 5.5. En este caso se estudia como se comporta la RNA si sólo es entrenada con información del sitio B y probada en A, B y C para los mismos obstáculos.

Para mediciones hechas en el sitio B, q y v_x muestran valores R^2 muy cercanos a 1 para obstáculos con diámetros mayores que 0.05m, esto es, para $\beta > 0.25$, ver Figura 5.6 para ejemplo, donde la malla objetivo y de predicción están graficadas para un obstáculo con $\beta = 0.4758$, alcanzando una predicción de $R^2 = 0.979$. Sin embargo, para obstáculos pequeños, la precisión disminuye. Por ejemplo, con $\beta = 0.0488$ se obtiene $R^2 = 0.654$ cuando se considera v_x y $R^2 = 0.802$ cuando se usa q.

Estimaciones en los sitios de medición A y C para los mismos diáme-

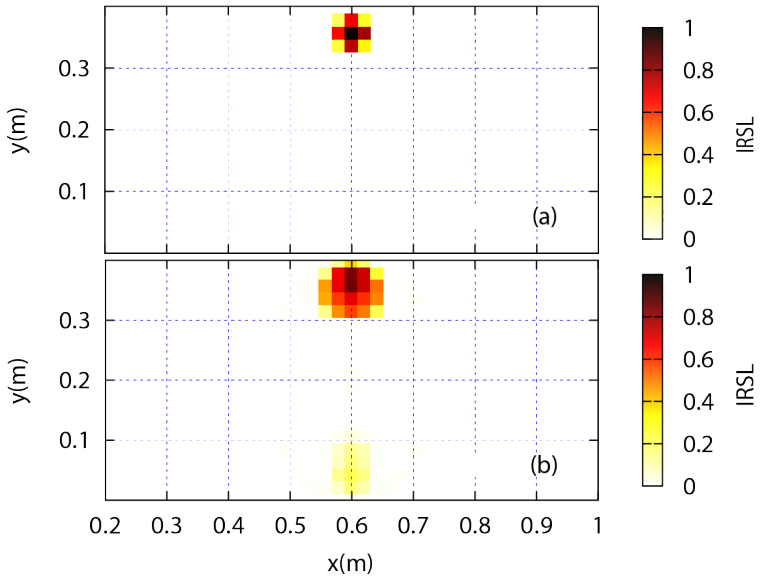

Figura 5.7: Objetivo (a) y predicción de la RNA (b) considerando el perfil v_x del fluido generado por un obstáculo en $y = 0.35$m con $\beta = 0.122$. Contrastando con la Figura 5.6, se observa como la RNA se confunde e indica un bajo IRSL en el fondo del tubo, obteniendo una predicción con $R^2 = 0.09$.

tros, muestran un comportamiento similar con pequeñas variaciones en la precisión; esto era de esperarse, ya que los perfiles cambian en tiempo y espacio. Por ejemplo, para la estimación para el obstáculo con $\beta = 0.0122$, R^2 decrece aproximadamente 16 %. Esto podría implicar que incluso al medir lejos del obstáculo se puede obtener una buena estimación del tamaño del mismo. Considerando los resultados obtenidos para este caso, no es posible establecer para cual de las variables físicas (v_x o q) la RNA muestra un mejor desempeño.

La Figura 5.8 muestra los resultados con un sólo sensor en $y = 0.21$m, donde se obtienen valores de R^2 arriba de 0.8 considerando v_x. Esta precisión decrece cuando q es considerada como entrada, obteniendo la peor predicción con $R^2 = 0.231$. Para el caso con tres sensores, los resultados

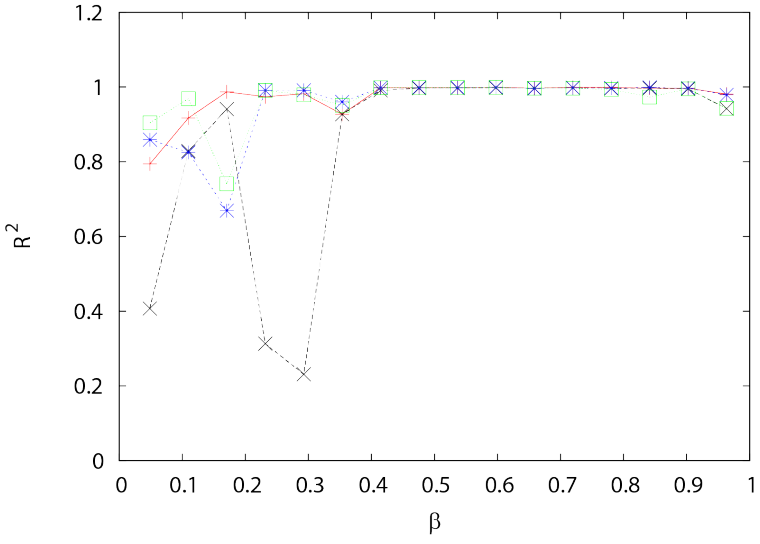

Figura 5.8: Predicciones de la RNA analizando R^2 para distintos tamaños de obstáculos y centrados en el tubo. Tomando la serie de tiempo en un sólo sensor para v_x $(+)$ y q $(\times)$ en $y = 0.205$m y tres detectores en $y = 0.105$m, 0.210m y 0.315m de v_x $(*)$ y q $(\square)$ en el sitio de medición B. Note que al considerar la serie de tiempo de tres detectores, hay valores de R^2 cercanos a 1 para diámetros mayores a $\beta = 0.20$, mientras que con uno sólo, esto ocurre para $\beta > 0.25$.

mejoran considerablemente, manteniendo un comportamiento similar para ambas variables físicas, obteniéndose la peor predicción con $R^2 = 0.669$ para $\beta = 0.183$. Además, en el caso $1c$, los valores de $R^2 > 0.9$ para $\beta > 0.25$.

Siguiendo el enfoque que en el caso 2a, donde las RNA son entrenadas con los perfiles de v_x o q medidos en el sitio B, con la intención de obtener no sólo una estimación del tamaño de la obstrucción, si no también su localización sobre el eje y. Como se muestra en la Figura 5.9, los resultados tienen un coeficiente R^2 para los obstáculos más grandes ($\beta \geq 0.488$) cercanos a 1. Sin embargo, en relación con el obstáculo con $\beta = 0.366$ para el que la RNA no fue entrenada, el peor resultado disminuyó hasta un valor $R^2 = 0.356$ para v_x y $R^2 = 0.243$ para q. Note que ambas estimaciones son

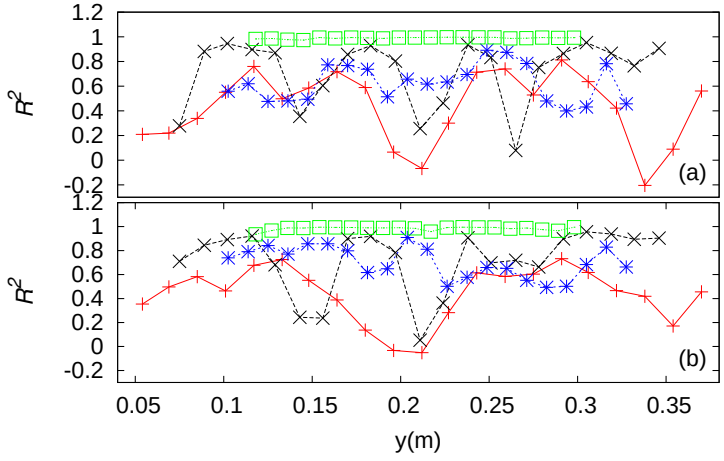

Figura 5.9: Análisis de la predicción para obstáculos en diferentes posiciones sobre el eje y, entrenando y probando la RNA con perfiles de v_x (a) y q (b) para detectores en el sitio de medición B. La curva roja (+) muestra los resultados obtenidos para el obstáculo con $\beta = 0.122$, mientras que las curvas negra ($\times$), azul ($*$) y verde ($\square$) muestran los resultados con $\beta = 0.244$, 0.366 y 0.488 respectivamente. Los valores de R^2 son similares para ambos enfoques, donde el obstáculo más pequeño tiene la peor predicción en el centro y cerca de las paredes del tubo.

hechas con obstáculos cerca de las paredes de la tubería, mientras que para el coeficiente R^2 con el obstáculo de diámetro $\beta = 0.244$, tiene su valor más bajo cuando está localizado en el centro del tubo, con $R^2 = 0.66$ y $R^2 = -0.033$ usando los perfiles de v_x y q respectivamente.

Respecto al caso 2b, donde la RNA se entrena con la serie de tiempo de un sólo sensor en el centro del tubo, la RNA es incapaz de aprender el comportamiento del flujo para la mayoría de obstáculos cerca de los bordes, así como para los pequeños, como se vio también en el caso 2a. Esto es evidente de la Figura 5.10, donde se obtienen valores negativos para los obstáculos pequeños $\beta = 0.122$, con un valor mínimo $R^2 = -0.312$

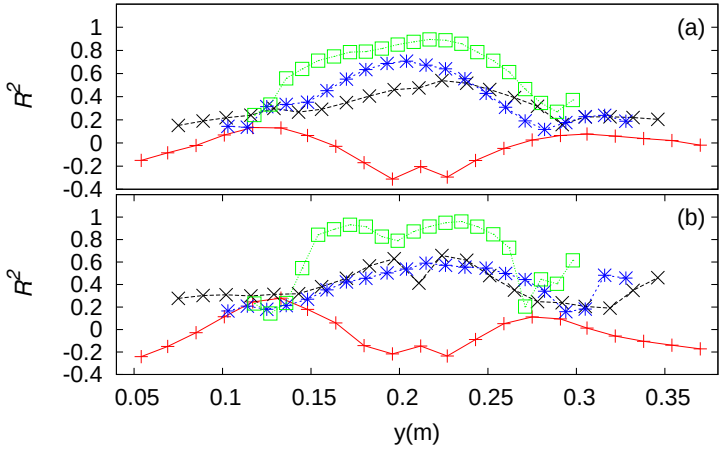

Figura 5.10: Análisis para la predicción de obstáculos en diferentes posiciones en el eje y. Las RNA son entrenadas con la serie de tiempo de v_x (a) y q (b) en el centro del tubo en el sitio de medición B. Las curvas roja (+), negra ($\times$), azul ($*$) y verde ($\square$) muestran los resultados para $\beta = 0.122$, 0.244, 0.366 y 0.488 respectivamente. Es evidente la dificultad de la RNA para dar una predicción adecuada acerca de la localización del obstáculo con un sensor, esto es más evidente cerca de los bordes del tubo.

cuando el obstáculo está en el centro del tubo. Para obstrucciones más grandes ($\beta \geq 0.244$) la RNA es mas precisa, excepto cuando están cerca de las paredes.

En el caso 2c, donde la serie de tiempo fue generada con tres detectores equidistantes en $y = 0.105$m, 0.210m y 0.315m en el sitio de medición B, las predicciones se muestran en la Figura 5.11 obteniendo valores similares R^2 para q y v_x. En este escenario, las predicciones para los obstáculos más pequeños ($\beta = 0.122$) muestran una mejora comparados con los resultados del caso 2b (Figura 5.10). Para obstáculos con $\beta = 0.366$, todos los valores de R^2 se encuentran arriba de 0.4 usando v_x o q, mientras que obstáculos con $\beta = 0.244$ y $\beta = 0.488$ la malla de predicción muestra un buen ajuste

con respecto a la malla objetivo en casi todas las posiciones, con R^2 arriba de 0.8. Sin embargo, los obstáculos con $\beta = 0.122$ y localizados en el centro del tubo tienen un valor $R^2 = -0.102$ usando el perfil de q.

En general, para el caso 2, el peor ajuste se obtiene para los obstáculos más pequeños. Esta confusión de la RNA no debe sorprender, ya que las fronteras del tubo perturban más el flujo del fluido después de donde se encuentra el obstáculo cuando está localizado cerca de la pared que en aquellos escenarios donde está localizado en el centro. La capa de frontera se afecta por la naturaleza viscosa del flujo y si el obstáculo es muy chico, el efecto causa que el flujo sea contrarrestado por fuerzas viscosas, lo cual implica que cuando se mide lejos del obstáculo (por ejemplo en el sitio de medición B), el flujo prácticamente se comporta como si no hubiera obstáculo. Además, los objetos pequeños a pesar de su distancia a las fronteras del tubo, no afectan significativamente el patrón de flujo que es generado detrás de él, por lo que la RNA no puede caracterizarlo correctamente. Como se puede observar en el caso 2b, un sólo sensor no es suficiente para obtener buenas estimaciones y también se reduce la precisión con respecto al caso 2a. Al incrementar a tres sensores en el caso 2c, se obtiene una mejora considerable en términos de los valores R^2.

Los resultados de los casos 1 y 2 muestran que al incrementar el número de puntos de medición en tiempo y espacio, se mejora considerablemente la predicción del tamaño del obstáculo por la RNA. Por ejemplo, para el caso $1a$ donde la RNA usa 83 valores en el espacio y sólo uno en el tiempo, produce una precisión equivalente al caso $1c$, donde tres mediciones en el espacio y 300 en el tiempo son usadas. A una conclusión similar se llega al comparar los casos 2a y 2c. En otras palabras, la RNA para el caso $1a$ se desempeña mejor que en el caso $1c$, ya que $1a$ requiere menos mediciones en el tiempo. Sin embargo, en un sentido práctico, tener un sitio de medición con sólo 3 sensores es más deseable que un enfoque donde se construyen sitios de medición con hasta 83 sensores.

En lo referente al caso 3, los coeficientes R^2 obtenidos para cada escenario se muestran en la Tabla 5.3. Aquí se observa que el problema persiste para obstáculos pequeños ($\beta = 0.0122$), con $R^2 = -23.753$ cuando se usa v_x o incluso peor cuando se usa q obteniendo $R^2 = -246.954$.

Sin embargo, los resultados tienen la segunda velocidad de flujo entrante más baja con $v_c = 0.3$m/s. Para entender esto, se pueden comparar los

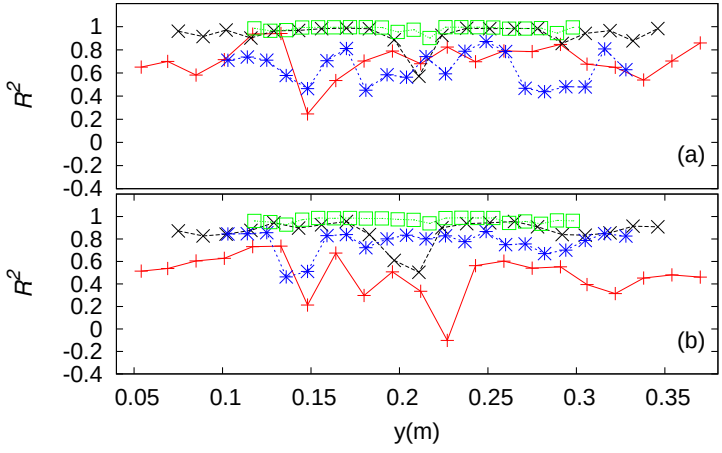

Figura 5.11: Resultados de R^2 obtenidos para obstáculos en diferentes posiciones en el eje y. Las RNA son entrenadas con la serie de tiempo de v_x (a) y q (b) de tres sensores en $y = 0.105$m, 0.210m y 0.315m en el sitio de medición B. Las curvas roja ($+$), negra ($\times$), azul ($*$) y verde ($\square$) muestran los resultados para obstáculos con $\beta = 0.122$, 0.244, 0.366 y 0.488 respectivamente. Una vez más, los peores ajustes para el coeficiente fueron obtenidos con las obstrucciones más pequeñas, con $R^2 = -0.102$ usando q como datos de entrada. A pesar de esto, es remarcable la mejora para los otros tres tamaños de obstáculo comparado con el caso donde se usó un sensor.

resultados obtenidos en el caso $1a$, donde la v_c asociada es siempre igual a 1.5m/s. Esto quiere decir que la disminución de la precisión se debe no solo con la dificultad de la RNA para caracterizar el patrón de flujo para obstáculos pequeños, sino también a que el campo de velocidad del flujo alrededor del obstáculo es pequeño. Es decir, en el caso 3 encontramos que al tener obstáculos pequeños y velocidades del flujo entrante pequeñas, la razón de fuerzas inerciales a viscosas dentro del fluido es muy baja. Note la mejora de los resultados para el valor de $\beta = 0.0122$ con $v_c = 1.35$m/s, obteniendo $R^2 = -0.645$ para v_x o $R^2 = -0.680$ para q. Lo mismo sucede para el obstáculo con $\beta = 0.0976$, mejorando de $R^2 = 0.250$ a 0.975 y de $R^2 = -4.518$ a 0.975 para v_x y q respectivamente cuando la velocidad de flujo entrante incrementa de $v_c = 0.15$m/s a $v_c = 0.75$m/s para ambas variables. Esto indica que la RNA aprende mejor cuando incrementamos la velocidad del flujo entrante. En general, la precisión de las predicciones mejora cuando v_x se usa en vez de q. Vale la pena mencionar, que para la mayoría de las predicciones, los valores de R^2 se encuentran muy cerca a 1, con un desempeño similar comparado con el caso $1a$. Esto muestra la flexibilidad de la RNA para trabajar con diferentes velocidades iniciales de flujo entrante y usando diez sensores en ambos sitios de medición.

Hemos probado que la RNA alcanza un buen desempeño, sin importar si seleccionamos los conjuntos de entrenamiento, validación y prueba de manera ordenada como en los casos 1 y 2 o aleatoriamente como en el caso 3. La relevancia de este último caso, es que la RNA se entrena no sólo con diámetros diferentes, si no también con diferentes velocidades de flujo entrante, implicando más complejidad con respecto a la información de entrada, resultando en una clara mejora en el reconocimiento de las formas de los obstáculos. A pesar de que para $\beta = 0.0122$ se obtiene resultados malos, la RNA fue capaz de obtener $R^2 > 0.96$ para $\beta \geq 0.0976$ cuando se usa como entrada v_x, excepto en el caso particular de $\beta = 0.0976$ con una velocidad entrante baja de $v_c = 0.15$m/s, que es diez veces menor que la v_c usada en el caso $1a$. Resultados similares se obtienen cuando se usa q, con valores $R^2 > 0.915$ en general. Note que comparando con el caso 1, los mejores resultados se obtuvieron para $\beta > 0.25$ con $R^2 > 0.9$ para v_x y q.

Hasta ahora, realizamos predicciones de tamaños y formas de obstáculos, pero podemos probar también la habilidad de la RNA entrenada para estimar la forma y tamaño de un obstáculo diferente a los usados en el

β	v_c (m/s)	R^2 para v_x	R^2 para q
0.0122	0.3	-23.753	-246.954
0.0122	1.35	-0.645	-0.680
0.0976	0.15	0.250	-4.518
0.0976	0.75	0.966	0.919
0.0976	1.8	0.975	0.975
0.1952	1.5	0.961	-0.016
0.2928	0.45	0.989	0.936
0.2928	1.65	0.989	0.944
0.3904	0.45	0.998	0.978
0.3904	1.2	0.991	0.997
0.4880	0.9	0.999	0.988
0.4880	1.05	0.998	0.994
0.4880	1.5	0.999	0.986
0.5856	1.05	0.998	0.997
0.5856	1.2	0.997	0.997
0.7808	0.75	0.999	0.996
0.7808	1.2	0.999	0.999
0.8784	1.35	0.999	0.992
0.9638	0.6	0.999	0.966
0.9638	0.9	0.999	0.992

Tabla 5.3: R^2 para el conjunto de prueba, considerando diez valores del flujo entrante del perfil de v_x o q antes del obstáculo y diez valores en B. Excluyendo los tres obstáculos más pequeños, los demás resultados son prometedores con valores cercanos a 1.

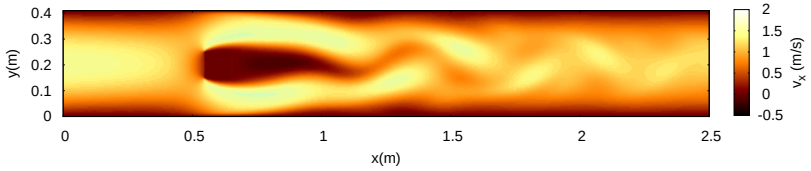

Figura 5.12: Magnitud de v_x, para una simulación de un flujo alrededor de un obstáculo cuadrado con un flujo entrante de $v_c = 1.5$m/s. El obstáculo está localizado en 0.6m sobre el eje x. Así como con el obstáculo cilíndrico, los vórtices formados después del obstáculo cuadrado son llevados a lo largo de la dirección del flujo, generando también una calle de vórtices de Karman, pero con diferente frecuencia.

entrenamiento. Para analizar esta situación, elaboramos un experimento final, introduciendo un obstáculo cuadrado de 0.1m de lado con las mismas condiciones a las presentadas en el caso 1, ver la Figura 5.12 para referencia. Por simplicidad, sólo se presenta el caso para las series de tiempo con tres sensores en el sitio de medición B (como en el caso $1c$). A pesar de que la forma predicha por la RNA es parecida a las que se usaron como entrenamiento, se obtiene un tamaño parecido al objetivo cuadrado, ver la Figura 5.13, con un sorprendente valor $R^2 = 0.93$. A continuación se presentan las conclusiones de esta sección.

5.1.4. Conclusiones de la sección

Se construyeron y entrenaron una variedad de RNAs con la capacidad de estimar el tamaño y posición de un obstáculo dentro de un tubo con un ancho específico de 0.41m, $1/80 \leqslant \beta < 1$ y un rango de flujos entrantes con velocidades características de 0.15m/s a 1.8m/s. Las RNAs usan como entrada el perfil de v_x o q a una cierta distancia del obstáculo. Analizamos distintos casos variando el diámetro (caso 1), la posición del obstáculo con respecto al eje y (caso 2) y la velocidad del fluido entrante (caso 3).

Basados en las especificaciones usadas en este trabajo, de los resultados obtenidos para el caso 1, la RNA es muy capaz de generar estimaciones acerca del tamaño del obstáculo cuando los datos suministrados son muy

similares a los que se usan en la fase de entrenamiento, con resultados de $R^2 > 0.9$ para valores de $\beta > 0.25$. En el caso 2 se hizo algo similar, considerando un obstáculo para el que la RNA no fue entrenada con información acerca del perfil o las series de tiempo de v_x o q. Incluso cuando hubo disminución en la precisión, la RNA fue capaz de estimar no sólo la forma, sino también la localización de tal obstáculo con $R^2 > 0.6$ en general. Finalmente, en el caso 3, usamos diferentes velocidades del flujo entrante para cada bloqueo considerado, encontrando que para velocidades y obstáculos pequeños, las predicciones de la RNA tienen poca precisión, mientras que para velocidades y tamaños más grandes, la precisión mejora considerablemente, alcanzando valores de $R^2 > 0.9$ en general para $\beta \geq 0.0976$.

Encontramos que la red se desempeña mejor en dos situaciones. Primero, cuando el número de sensores en un sitio de medición es mayor como se presenta en el caso $1a$. Segundo, cuando las series de tiempo con tres sensores son consideradas como en el caso $1c$. No obstante, pensamos que en un sentido práctico, es más conveniente extraer datos en un lapso de tiempo con menos sensores. En este contexto, los resultados de entrenar la RNA con v_x o q son similares.

Finalmente, de todos los casos revisados, los mejores resultados se obtienen para el caso 3, dado que la RNA puede dar predicciones con diferentes velocidades entrantes del fluido, mientras que en los casos 1 y 2 fueron entrenadas sólo con una velocidad del fluido entrante. Además, el caso 3 tiene un número menor de obstáculos en el conjunto de entrenamiento.

Tomando en cuenta que en la literatura uno puede encontrar trabajos como [111], donde usan un análisis modal para identificar un bloqueo identificando su localización, espesor y profundidad; o el estudio hecho por [112] donde se utiliza un análisis de reflexión transitoria presión-onda para caracterizar el bloqueo dentro del tubo, nos gustaría extender nuestra investigación para desarrollar herramientas numéricas capaces de reconocer la forma y profundidad (su posición alrededor del eje y) de tales obstrucciones.

En la próxima sección, realizamos una comparación entre una RNA y una MSV en donde se usan los métodos de aprendizaje automático para clasificar el tamaño y la posición de una obstrucción dentro de una tubería,

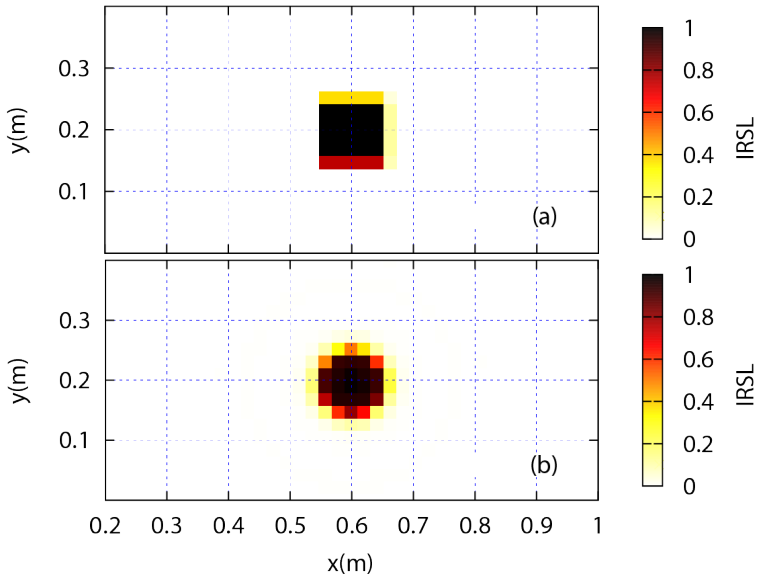

Figura 5.13: Predicción para un obstáculo cuadrado en el centro del tubo, considerando como entrada para la RNA las series de tiempo de q para tres sensores diferentes en B. Como es de esperarse, la RNA detecta un obstáculo con forma parecida a los usados en el entrenamiento. Sin embargo, tiene un tamaño y posición comparable al objetivo propuesto.

muy similar a lo que se realizó en esta sección pero en vez de usar una malla objetivo se utiliza un enfoque diferente.

5.2. Posición y tamaño de una obstrucción

La sección presente se apoya en el artículo titulado *Detection of the size and location of a blockage in a flow using machine learning methods*. El artículo aún se encuentra en desarrollo junto con los Drs. José A. González y Silvano U. Que y le faltan algunos resultados finales.

En este trabajo usamos dos métodos de AA para atacar el problema de

detectar un bloqueo en un conducto. Para realizar esto, usamos redes neuronales artificiales y máquinas de soporte vectorial para caracterizar patrones de flujo. Las RNAs han sido aplicadas en problemas de dinámica de fluidos como herramientas más rápidas para simulaciones de fluidos y para predicción de turbulencia [1, 113, 114] o clasificación de defectos en estructuras tubulares usando imágenes [115], pero principalmente han sido aplicadas en la identificación de fases de flujo [116, 117]. Las MSVs también han sido usadas en este tipo de problemas, como puede ser en la detección del fenómeno de la cavitación en las válvulas de mariposa [118], en el análisis de datos en los sistemas de distribución de agua [119] y recientemente en el diagnóstico de bloqueos en tuberías de agua [120]. La exploración de parámetros de la MSV empleada en el estudio, fue realizada por el Dr. José A. González.

El objetivo de este estudio es localizar y detectar el tamaño de una obstrucción bloqueando parcialmente o completamente el flujo en un conducto bidimensional, al clasificar los patrones de flujo. Estos patrones fueron obtenidos por un código numérico de lattice Boltzmann en 2D [121, 88] implementado por el Dr. Silvano Ulises Que, para la simulación del flujo alrededor de un obstáculo.

Este análisis toma en consideración dos diferentes escenarios: primero, cambiando el tamaño del obstáculo y su posición a lo largo del conducto, obteniendo información física relevante tal como la velocidad, vorticidad o la presión del flujo en el dominio numérico. De esta manera, se considera la relevancia de escenarios en los que pueden existir obstrucciones que varían en tamaño desde obstáculos muy pequeños que apenas perturban el flujo, hasta bloqueos completos que impiden la continuidad del flujo. Para representar este escenario, usamos la razón entre el ancho del conducto y el tamaño de la obstrucción, que va desde 1/100 hasta casi un valor de 1. En particular, hemos considerado la componente x del campo de velocidades del flujo (v_x) como la información fundamental que alimenta a la RNA y a la MSV para su entrenamiento. Enseguida se presenta una descripción detallada del problema del bloqueo en las simulaciones usando el código numérico desarrollado con el método de lattice Boltzman.

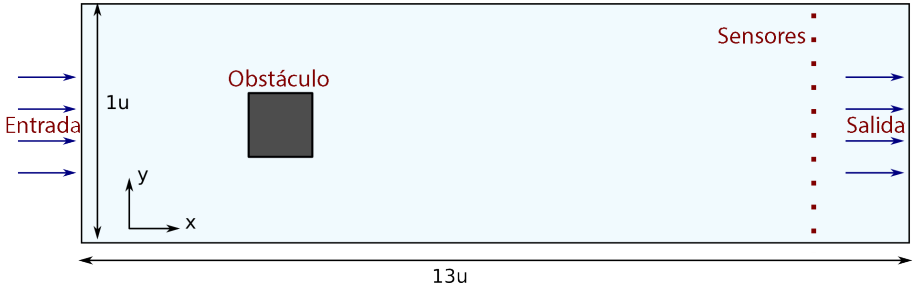

Figura 5.14: Representación esquemática de un conducto bidimensional. El obstáculo cuadrado está localizado en el centro del eje y y el sitio de medición se encuentra en $x = 12u$, donde la información física es extraída por sensores representados con puntos rojos.

5.2.1. Simulaciones numéricas

Las simulaciones numéricas fueron llevadas a cabo con el fin de proveer al método de aprendizaje automático con la información necesaria para su proceso de aprendizaje, realizadas con el MLB. Este método es relativamente nuevo para la simulación de dinámica de fluidos, el cual goza de una popularidad en crecimiento debido a la simplicidad al aplicarlo y una alta habilidad para emular una gran variedad de fenómenos [105, 122]. Adicionalmente, el MLB ha probado tener un gran potencial en numerosas aplicaciones, principalmente en simulaciones de dinámica de fluidos [107, 108] y es particularmente ventajoso para la simulación de flujo en medios porosos [123], flujos multifase y multicomponentes [121, 124] y hemodinámica [125].

Para los casos de estudio, consideramos un flujo en un conducto de dos dimensiones con un obstáculo fijo inmerso en el mismo, con el flujo moviéndose en la dirección x positiva. El obstáculo se encuentra fijo perpendicular al plano $x - y$, así que es representado por un cuadro en la simulación bidimensional, como se muestra en la Figura 5.14. La condición de frontera en la salida del flujo se impone lo suficientemente lejos para que los parámetros característicos del flujo no sean afectados por los cálculos internos realizados por el algoritmo numérico, usando el procedimiento descrito en [126]. Para las paredes sólidas representando el tubo y el

obstáculo bidimensionales, se usa una condición de frontera de tipo medio rebote [121, 88]. A su vez, para lidiar con el flujo entrante (caracterizado por un flujo de Poiseuille), se utilizó el tratamiento descrito en [127] para el caso en el que la velocidad de los elementos de la frontera es paralela a la dirección de entrada del flujo.

Para el dominio numérico de las simulaciones se usó una malla de 200×2600 nodos. Los estudios se realizaron con unidades adimensionales, así que el sistema se puede adaptar a cualquier dimensión física relevante, considerando que el código desarrollado simula la dinámica de un fluido newtoniano incompresible no relativista. Así, se seleccionó un dominio numérico correspondiente a una longitud de $L_y = 1$ en la dirección vertical y de $L_x = 13$ en la dirección horizontal como se muestra en la Figura 5.14. Se consideró un fluido entrante de Poiseuille en un régimen estacionario con una velocidad máxima de $v_{max} = 20$, una densidad de $\rho = 1$ y una viscosidad cinemática de $\nu = 10^{-1}$, por lo que el fluido entrante de Poiseuille está caracterizado por un número de Reynolds de 200. La localización de todos los obstáculos estudiados varía a través del dominio numérico, así que para el análisis de los casos estudiados, definimos α como la distancia a la que se encuentra cada obstáculo con respecto al detector donde se extrae la información física relevante y que se localiza en $x = 12u$, como se muestra en la Figura 5.14. Se varía α con valores que van desde $\alpha = 1u$ hasta $\alpha = 11u$ en pasos de $0.1u$, donde u representa una unidad de longitud cuyo valor es equivalente al diámetro del conducto. Adicionalmente, definimos β como el valor de la razón entre el tamaño del obstáculo cuadrado y el ancho de la tubería. El tamaño del obstáculo cambia de valores de β que van de $\beta = 0.01$, representando bloqueos pequeños, hasta valores de $\beta = 1$, representando obstrucciones del tamaño total del diámetro del conducto.

Se realizaron un total de 10^4 simulaciones, tomando en cuenta el rango de los parámetros libres α y β. Cada una de las simulaciones numéricas fue detenida después de completar $9\text{x}10^4$ iteraciones, que representa el tiempo transcurrido en el que el fluido entrante ha cruzado la posición del detector, lo cual ocurre antes de completar el total de las iteraciones.

En la próxima sección se describe la metodología seguida en los casos de estudio, así como el procesamiento de los datos que se usan para entrenar los métodos de AA para clasificar los patrones de flujo generados por las simulaciones numéricas.

5.2.2. Metodología

Para poder estimar la posición y tamaño del objeto cuadrado bloqueando el fluido en el conducto bidimensional, empleamos dos diferentes métodos de aprendizaje automático para identificar los bloqueos, considerando básicamente dos situaciones físicas diferentes. Los algoritmos se alimentaron para el entrenamiento con la información extraída de un sensor localizado en $x = 12u$. Este sensor extrae información de la componente x de la velocidad del flujo (v_x) a través de la evolución temporal en los 300 pasos de tiempo. A esta información de aquí en adelante se le conocerá como patrones de flujo o simplemente patrones y servirán para que los métodos de aprendizaje asocien tal información con una obstrucción con ciertos valores de α y β. Seleccionamos el sensor primeramente considerando el hecho que tiene que estimar un gran número de posibles posiciones del obstáculo a lo largo de la tubería, tomando en cuenta obstáculos que están muy cerca al detector, comparado con otros que se encuentran considerablemente lejos. Los estudios llevados a cabo mostraron que, era suficiente considerar 300 pasos de tiempo para la evolución temporal del flujo, extrayendo información de un sólo sensor centrado en $y = 0.5$.

A continuación se detallan los métodos de aprendizaje automático utilizados y los datos empleados para entrenarlos, así como los casos de estudio.

Métodos de aprendizaje automático

Como ya se mencionó anteriormente, se usaron dos métodos diferentes de AA para analizar las simulaciones generadas por el MLB: una red neuronal artificial y una máquina de soporte vectorial.

Ambos métodos se escogieron teniendo en mente que nuestra meta es predecir la razón entre el tamaño de la obstrucción y el ancho del conducto (β) y la posición del bloqueo a lo largo del conducto (α). Podemos usar

fácilmente un regresor lineal para estimar cada una de estas cantidades, pero los resultados no son tan buenos como los obtenidos usando una RNA o una MSV para clasificar las simulaciones de acuerdo a los parámetros α y β usados. En nuestros casos de estudio, consideramos en las simulaciones cien posiciones diferentes del obstáculo con cien tamaños diferentes, así que en vez de predecir directamente el valor α o β, dividimos las simulaciones en grupos o clases y las predicciones se realizan sobre la clase a la que pertenece cada patrón. Así que para cada valor que se predice, ya sea α o β, se puede asociar un valor promedio de acuerdo a la Ec. (2.20) para cada valor, dependiendo del número total de clases en las que se dividió cada parámetro. Una vez que se fija el número de clases para α y β, se especifican los objetivos para cada método de AA y se dividen las simulaciones en dos conjuntos: el conjunto de entrenamiento y el conjunto de prueba. 80 % de las simulaciones en cada caso de estudio constituyen el conjunto de entrenamiento y el 20 % restante se usa como conjunto de prueba. El conjunto de prueba se selecciona homogéneamente, de tal manera que, sin importar el número de clases considerado de cada clase, tomamos la misma cantidad de simulaciones para el conjunto de prueba. Esto se logra seleccionando las simulaciones para valores de α y β que satisfacen:

$$\text{prueba} = \{\alpha_i, \beta_j | i = 5(l-1) + 100(j-1) + \text{mod}(j, 5)\},$$

para $1 \leq l \leq 20$ y $1 \leq j \leq 100$, donde $\text{mod}(j, 5)$ es el residuo de $j/5$.

Los parámetros y consideraciones para la RNA y la MSV se discuten en las próximas subsecciones.

Consideraciones de la RNA

Una RNA con propagación hacia delante con un entrenamiento supervisado retroalimentado [11] fue desarrollada en Python para clasificar las simulaciones dependiendo del tamaño y la posición de la obstrucción considerada, donde se minimiza una función de costo de error cuadrático medio [9] y donde el conjunto de entrenamiento se divide en *mini batches*.
Consideramos un perfil normalizado v_x de la evolución del flujo usando una normalización min-max como datos de entrada para cada simulación, así que caracterizamos cada simulación con 300 valores: $\hat{v}_x(t) =$

$\{\hat{v}_x(t_0), \hat{v}_x(t_1), \ldots, \hat{v}_x(t_{299})\}$, donde $\hat{v}_x(t_0) = (v_x(t_0) - min_{t_0})/(max_{t_0} - min_{t_0})$ siendo min_{t_0} y max_{t_0} el mínimo y el máximo v_x de todos los patrones al tiempo $t = t_0$. Expresiones similares se obtienen para cada tiempo t. La estructura de la red consiste entonces de una capa de entrada con 300 neuronas que reciben la velocidad normalizada $\hat{v}_x$ de cada simulación, cuatro capas ocultas y una de salida con funciones de activación sigmoides. Dependiendo del caso de estudio, usamos tres estructuras diferentes: treinta neuronas en cada capa oculta y una neurona de salida para los casos 1 y 2, sesenta neuronas en cada capa oculta y una neurona de salida para los casos 3 a) y 3 b) y sesenta neuronas en cada capa oculta y dos neuronas de salida cuando se predicen simultáneamente el tamaño y la posición del obstáculo.

Los objetivos para α y β se definen de acuerdo a la Ec. (2.18) para cada valor. En el caso en el que la RNA estima los valores α y β simultáneamente, la red tiene dos salidas, por lo que la primer salida se asocia con α y la segunda con β.

Con estas condiciones para la estructura de la red y la definición de objetivos, se utiliza el enfoque de clasificación descrito en el Capítulo dos, utilizando un máximo de 20 clases. Debido a que la MSV usa una clasificación uno contra todos, con la finalidad de comparar ambos métodos cuando la red tiene 2 salidas, es necesario multiplicar el porcentaje de patrones clasificados correctamente de cada parámetro, para encontrar la probabilidad de clasificar simultáneamente ambos parámetros correctamente al usar la RNA. Esto es por que cada combinación de parámetros se considera como una clase cuando se utiliza la MSV y ya se está prediciendo entonces una correcta clasificación de ambos parámetros simultáneamente.

Una vez que se han especificado los datos de entrada con sus respectivos objetivos y que han sido separados en los conjuntos de entrenamiento y prueba, procedemos con el entrenamiento. Para el entrenamiento, se usaron $2{\times}10^4$ épocas, donde una época es una completa presentación del conjunto de datos de entrenamiento y se consideró un tamaño de *mini batch* de 32 patrones. Analizamos múltiples estructuras de redes con diferente número de clases y con una constante de aprendizaje $\gamma = 8$. Las estructuras, nor-

malizaciones y parámetros de la red presentados, son aquellos en donde el desempeño de la red fue mejor, de entre los parámetros considerados. Vale la pena mencionar que el código de la RNA aún no se encuentra paralelizado, así que la exploración de parámetros se realizó de manera manual. Aunque los resultados mostrados son buenos, se pueden mejorar con una exploración automática de parámetros empleando un código paralelizado. A continuación se encuentran los detalles del otro método de AA utilizado.

Consideraciones de la MSV

El núcleo particular que utilizamos para este estudio en la MSV, fue un núcleo con función de base radial definido por

$$K(\vec{x}_i, \vec{x}_j) = e^{-\gamma||\vec{x}_i - \vec{x}_j||^2} \ , \tag{5.8}$$

con $\gamma \geq 0$. También, como estamos tratando con un problema de clases múltiples (m clases), usamos un enfoque OVA, clasificando $m(m-1)/2$ problemas de dos clases y utilizando una estrategia de votación. En el caso en el que dos clases tengan la misma votación, la clase que aparece primero en el arreglo es la que se escoge.
En vez de usar nuestra propia implementación de la MSV, usamos la librería pública libsum [13]. Entonces, las entradas $\vec{x}_i$ son los valores de la velocidad medidas en el detector.
Quedan pendientes algunos detalles técnicos de la MSV, ya que aún faltan algunos resultados para poder resumir los parámetros que se utilizan.

5.2.3. Casos de estudio y procesamiento de datos

Realizamos primero dos casos sencillos como prueba para los métodos de aprendizaje automático, con el objetivo de estimar de manera separada el tamaño y la posición del obstáculo. Después, elaboramos dos casos más complejos, siendo el primero de ellos una situación idealista con un obstáculo en el centro del diámetro del conducto y el segundo una situación más realista donde el bloqueo se genera sobre las paredes del conducto.

Con la finalidad de probar los métodos de aprendizaje en los cuatro casos propuestos, fue necesario procesar los patrones obtenidos de las simulaciones numéricas de la manera más adecuada. Por este motivo, probamos

el desempeño de los métodos, al considerar los valores de α y β divididos en $C_\alpha = 5$, 10, 20 clases y en C_β=5, 10, 20 clases respectivamente.

Como nuestra meta es clasificar los patrones generados por las simulaciones de acuerdo a cada clase en las que fueron divididas, el primer paso fue clasificar cada parámetro independientemente, con el fin de conocer el desempeño de los métodos de AA en cada situación, antes de proceder con la clasificación de ambos parámetros simultáneamente. El comportamiento de cada método de AA que empleamos, se pone a prueba en los siguientes casos de estudio:

1. Generamos 100 simulaciones numéricas para un obstáculo cuadrado bloqueando el flujo en el conducto, localizado en el centro del eje y, es decir, en $y = 0.5$, variando únicamente el parámetro α desde $\alpha = 1u \equiv \alpha_1$ hasta $\alpha = 11u \equiv \alpha_{100}$ en pasos de $0.1u$. Estimamos la posición de la obstrucción a lo largo del conducto para un valor fijo de $\beta = 0.5 \equiv \beta_{50}$. Por lo tanto, los entrenamientos para este caso fueron realizados considerando las simulaciones para β constante y la posición de la obstrucción α variable, entrenando los métodos para clasificar los patrones de acuerdo a la clase C_α a la que pertenecen.

2. Similarmente al caso anterior, generamos un total de 100 simulaciones numéricas, variando ahora los valores de β desde $\beta = 0.01 \equiv \beta_1$ hasta $\beta = 1 \equiv \beta_{100}$ en pasos de 0.01 para estimar el tamaño del obstáculo para un valor fijo de $\alpha = 6 \equiv \alpha_{50}$. Entrenamos los métodos de AA con patrones que corresponden a simulaciones con una posición del obstáculo constante α, considerando β variable para clasificarlas en la clase C_β a la que pertenece cada patrón.

3. Para el escenario idealista, con un obstáculo en el centro del diámetro, generamos 10^4 simulaciones numéricas diferentes para un obstáculo cuadrado bloqueando el flujo en el conducto, localizado en el centro del eje y. Las simulaciones se realizaron cambiando ambos valores α y β para estimar la posición y tamaño. Por lo tanto, este caso es una generalización de los casos anteriores, donde se entrenan los métodos de ML para clasificar α y β en las clases C_α y C_β respectivamente. Ejemplos de las simulaciones numéricas obtenidas para el valor de v_x a lo largo del flujo se muestran en la Figura 5.15.

Con el fin de analizar en detalle las capacidades de los métodos de aprendizaje automático empleados, se elaboraron los siguientes subcasos:

a) Considerando todas las simulaciones, alimentamos los métodos de AA para predecir sólo su posición. Así que el entrenamiento para este subcaso se realizó considerando los patrones resultantes al variar los valores de los parámetros β y α para clasificarlos de acuerdo a la clase C_α a la que pertenecen.

b) Similar al subcaso previo, pero se predice independientemente el tamaño del obstáculo. Entonces, entrenamos los métodos para clasificar los patrones de acuerdo a la clase C_β a la que pertenecen.

c) Finalmente, considerando todas las simulaciones, se estima simultáneamente la posición y el tamaño del obstáculo. Para esto, los entrenamientos de los métodos de AA se realizaron considerando los patrones para los parámetros α y β variables, como en los subcasos previos, pero clasificándolos de acuerdo a las clases C_α y C_β.

4. En contraste con el caso anterior, se considera un escenario más real donde la obstrucción se encuentra en las paredes, como se muestra en la Figura5.16, en vez de estar en el centro del conducto. Entonces, simulamos dos obstáculos rectangulares, colocándolos en la misma posición a lo largo del eje x, pero en cada una de las paredes del conducto, uno en $y = 1$ y el otro en $y = 0$. Para estas simulaciones, se usaron los mismos valores que en el caso 3 para los parámetros α y β con el objetivo de estimar simultáneamente la posición y el tamaño del obstáculo. En la Figura 5.17 se muestran ejemplos numéricos obtenidos para el valor v_x a lo largo del flujo cuando el bloqueo es causado por obstáculos en las paredes del conducto.

Los casos 1 y 2 se realizaron con 100 simulaciones numéricas diferentes, en los que 20, 10 y 5 patrones fueron considerados para cada clase, resultando en 5, 10 y 20 clases respectivamente. Tomamos el 20 % de esas simulaciones como conjunto de predicción. Similarmente para los casos 3 y 4 los cuales se componen de un total de 10,000 patrones distintos, tomamos

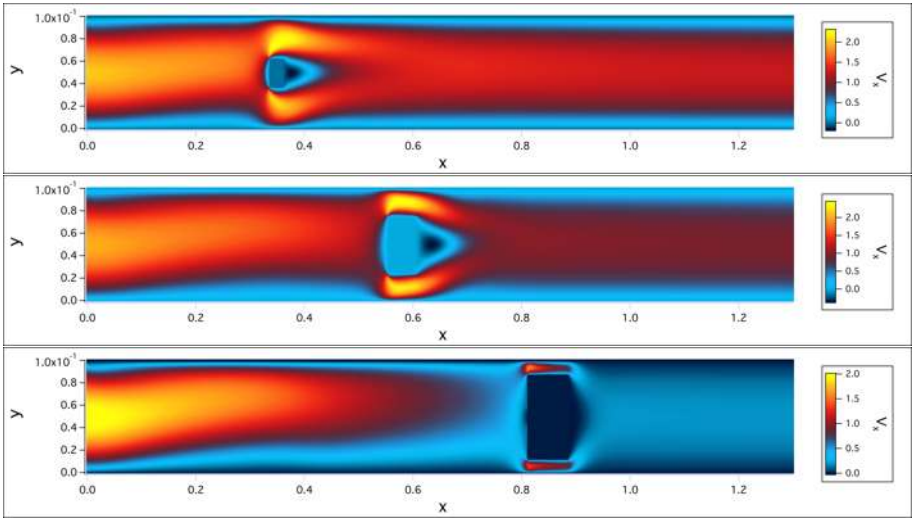

Figura 5.15: Valor v_x para el flujo alrededor de un obstáculo cuadrado con un flujo entrante de Poisseuille de $v_c = 20$. El tamaño y la posición del obstáculo cambia para cada simulación realizada, variando desde obstáculos muy pequeños hasta bloqueos completos y desde posiciones muy lejanas hasta muy cercanas al detector respectivamente. Arriba se muestra el patrón de flujo generado por una obstrucción pequeña a una distancia lejana del sensor, mientras que abajo se muestra el patrón generado por una obstrucción grande cercana al sensor.

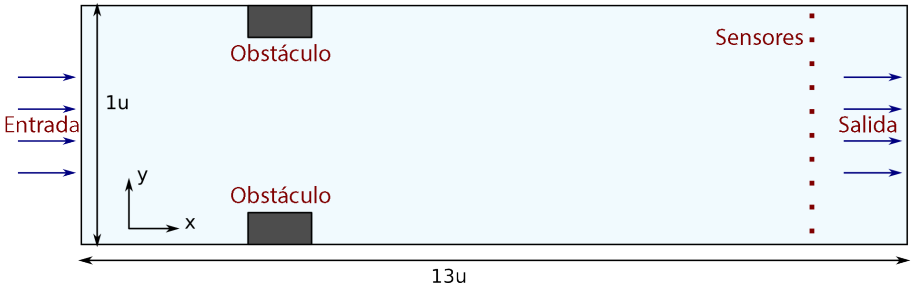

Figura 5.16: Representación esquemática del conducto bidimensional. Los obstáculos cuadrados se encuentran en las paredes y el sitio de medición está localizado en $x = 12u$, donde la información física se extrae por los sensores representados con puntos rojos.

400, 100 y 25 patrones para cada clase, resultando en 25, 100, y 400 clases respectivamente. Nuevamente usamos solo el 20 % de los patrones como conjunto de prueba.

Después de seleccionar los conjuntos de entrenamiento y de prueba para cada caso, estamos listos para entrenar los métodos de aprendizaje automático. En la próxima sección se compara la eficiencia de cada método al clasificar las simulaciones en cada uno de los casos de estudio.

5.2.4. Resultados

Una vez realizados los entrenamientos para los casos de estudio, es necesario analizar el comportamiento de ambos métodos de AA. Para esto, consideramos las clasificaciones correctas que realiza cada uno de los métodos si las salidas generadas por ellos son consistentes con los valores α y β involucrados en la generación de las simulaciones. Evaluamos el desempeño de los métodos de AA para cada caso, utilizando el porcentaje de patrones clasificados correctamente (PPCC).

Para las estimaciones asociadas al caso 1, donde los patrones son clasificados de acuerdo al valor α empleado en las simulaciones, mostrando el PPCC para β_{50} constante en la Tabla 5.4. Aquí, se puede ver que en general, cuando se consideran más clases para la clasificación de los patrones, el PPCC decrece. Sin embargo, el valor promedio predicho para cada clase

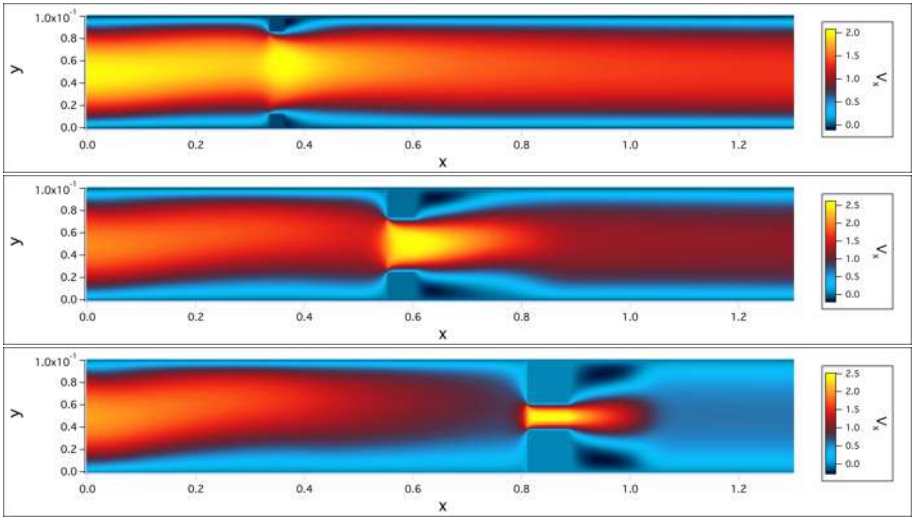

Figura 5.17: Magnitud de v_x para el flujo alrededor de dos obstáculos cuadrados fijos a las paredes del conducto con un flujo entrante de Poisseuille con $v_c = 20$. El tamaño y la posición del obstáculo cambia para cada simulación realizada, variando desde obstáculos muy pequeños hasta bloqueos completos y desde posiciones muy lejanas hasta muy cercanas al detector respectivamente. Arriba se muestra el patrón de flujo generado por una obstrucción pequeña a una distancia lejana del sensor, mientras que abajo se muestra el patrón generado por una obstrucción grande cercana al sensor.

caso 1				
PPCC (%)	RNA		MSV	
para	Entrenamiento	Prueba	Entrenamiento	Prueba
β_{50}				
$C_\alpha = 5$	98.75	100.00	95.00	100.00
$C_\alpha = 10$	95.00	85.00	91.25	100.00
$C_\alpha = 20$	95.00	80.0	82.50	95.00

Tabla 5.4: PPCC por la RNA y la MSV para los conjuntos de entrenamiento y prueba tomando en cuenta cien simulaciones. La salida está relacionada al valor de α, el cual fue dividido en C_α =5, 10 y 20 clases respectivamente para β_{50}.

tiene un error asociado menor, de acuerdo a la Ec. (2.20).

También se observa que la RNA obtiene un PPCC mayor en el conjunto de entrenamiento, comparado con el conjunto de prueba excepto para $C_\alpha = 5$, mientras que la MSV obtiene un PPCC más alto en el de conjunto de prueba, que en el de entrenamiento. La RNA pasa de clasificar todos los patrones correctamente cuando se consideran 5 clases, a clasificar incorrectamente 4 patrones cuando se consideran 20 clases. En el caso de la MSV, clasifica correctamente todos los patrones para 5 y 10 clases y falla en clasificar correctamente un patrón cuando son 20 clases. En esta situación la RNA tiene un mejor desempeño en el conjunto de entrenamiento comparada con la MSV, pero la MSV tiene mejor desempeño en el conjunto de prueba que la RNA.

Los resultados para el caso 2 relacionado con la estimación del tamaño del obstáculo, se presentan en la Tabla 5.5. Podemos observar un comportamiento análogo que el caso previo, aunque ligeramente peor, donde la RNA tiene mejor desempeño en el conjunto de entrenamiento que en el de prueba, mientras que la MSV tiene mejor desempeño en el conjunto de prueba que en el de entrenamiento, excepto cuando se consideran 5 clases. También en este caso, la RNA tiene un mejor desempeño en el conjunto de entrenamiento comparado con la MSV, pero la MSV tiene mejor desempeño en el conjunto de prueba que la RNA. Ambos métodos clasifican correctamente 95 % de los patrones en el conjunto de prueba, pero el PPCC decrece a 75 y

caso 2				
PPCC (%)	RNA		MSV	
para	Entrenamiento	Prueba	Entrenamiento	Prueba
α_{50}				
$C_\beta = 5$	100.00	95.00	96.25	95.00
$C_\beta = 10$	96.25	80.00	91.25	95.00
$C_\beta = 20$	92.50	75.00	72.50	85.00

Tabla 5.5: PPCC por la RNA y la MSV para los conjuntos de entrenamiento y prueba tomando en cuenta cien simulaciones. La salida está relacionada al valor de β, el cual fue dividido en C_β =5, 10 y 20 clases respectivamente para α_{50}.

85 para la RNA y la MSV respectivamente cuando se consideran 20 clases. Esto significa que cinco y tres patrones son clasificados incorrectamente por la RNA y la MSV respectivamente.

De los casos 1 y 2, es claro que cuando se estima el tamaño o la posición del obstáculo de manera separada, la MSV muestra mejores resultados que la RNA.

En el caso 3, la clasificación de α y β se realiza usando todas las simulaciones numéricas en vez de sólo usar cien de ellas, como en los casos 1 y 2. El desempeño de ambos métodos para todos los subcasos considerados se muestran en las Tablas 5.6 y 5.7. Aquí se observa una vez más la tendencia a disminuir del PPCC en los tres subcasos cuando son consideradas más clases.

En la Tabla 5.6 presentamos los resultados para la estimación independiente de la posición del obstáculo relacionado al caso 3a y la estimación del tamaño relacionado al subcaso 3b. Aquí se puede observar que en la mayoría de los resultados para ambos métodos se obtiene un mejor desempeño cuando estiman β, que cuando predicen α. Se puede ver como cuando se usa la MSV se obtiene un PPCC de al menos 95 en el conjunto de prueba, sin importar el número de clases considerado en ambos subcasos, mientras que la RNA alcanza al menos un PPCC de 90. Esto implica que de los 2000 patrones de prueba, no más de 100 fueron clasificados incorrectamente por

PPCC (%)	RNA		MSV	
para	Entrenamiento	Prueba	Entrenamiento	Prueba
caso 3 (a)				
$C_\alpha = 5$	99.16	97.95	98.61	99.10
$C_\alpha = 10$	98.06	96.60	97.13	97.75
$C_\alpha = 20$	91.64	90.25	94.51	95.60
caso 3 (b)				
$C_\beta = 5$	99.98	99.70	99.99	99.90
$C_\beta = 10$	100.00	99.85	99.88	99.70
$C_\beta = 20$	91.37	90.70	95.60	96.05

Tabla 5.6: PPCC por la RNA y la MSV para los conjuntos de entrenamiento y de prueba tomando 10^4 simulaciones. Arriba: la salida está relacionada a α, el cual se dividió en C_α =5, 10 y 20 clases respectivamente. Abajo: la salida está relacionada a β, el cual se dividió en C_β =5, 10 y 20 clases respectivamente.

la MSV y no más de 200 en el caso de la RNA.

Para el subcaso 3c, estimamos simultáneamente la posición y el tamaño del obstáculo, mostrando los resultados en la Tabla 5.7. El PPCC más bajo en el caso de la RNA comparado con los casos 3a y 3b, es debido a que se multiplica el PPCC de cada una de las salidas cuando se estiman ambos parámetros simultáneamente para poder compararlos con la MSV. Se puede ver también que se obtienen resultados similares a los subcasos previos, cuando se consideran 5 o 10 clases, pero cuando se usan 20 clases, el PPCC en el conjunto de prueba decrece considerablemente a un valor de 85.35 para la RNA y 81.50 para la MSV, que representa un total de 297 y 370 patrones clasificados incorrectamente respectivamente.

Con el fin de comprobar el comportamiento de los métodos utilizados y localizar en cuales clases se clasifican correctamente un mayor o un menor número de patrones, se generaron mapas de colores para los casos donde se clasifican simultáneamente α y β utilizando el conjunto completo de simulaciones.
El resultado de este análisis para el caso 3c utilizando la RNA se presenta

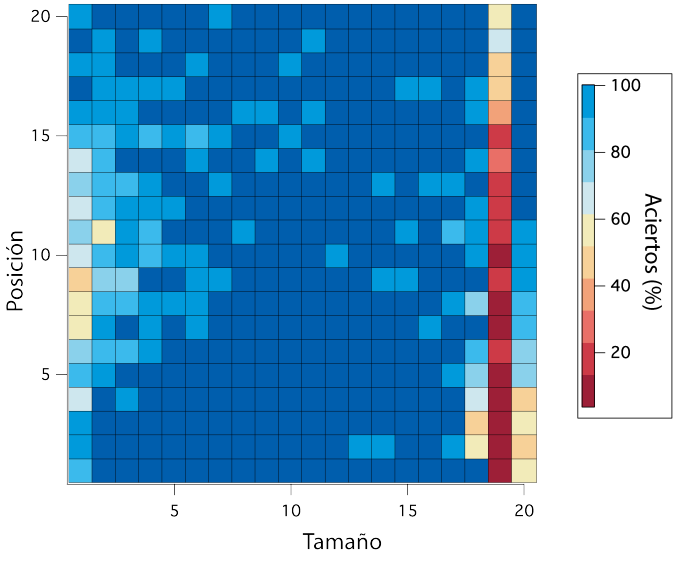

Figura 5.18: Porcentaje de patrones clasificados correctamente por la RNA para cada una de las clases cuando se consideran 400 clases en total en el caso 3c. El incremento en el valor de la posición indica la clase a la que corresponden patrones generados usando las posiciones del obstáculo más cercanas al sensor. El incremento en el valor del tamaño, indica una clase a la que corresponden patrones generados usando los tamaños del obstáculo más grandes. Entonces, la casilla inferior izquierda representa una clase donde los patrones fueron generados con los valores de α y β más pequeños, mientras que la casilla superior derecha representa una clase donde los patrones fueron generados con los valores de α y β más grandes.

caso 3 (c)				
PPCC (%)	RNA		MSV	
para	Entrenamiento	Prueba	Entrenamiento	Prueba
$C_\alpha = C_\beta = 5$	98.58	97.30	98.43	98.60
$C_\alpha = C_\beta = 10$	96.66	94.66	94.39	95.55
$C_\alpha = C_\beta = 20$	89.70	85.35	83.30	81.50

Tabla 5.7: PPCC por la RNA y la MSV para los conjuntos de entrenamiento y de prueba, clasificando α y β simultáneamente y tomando 10^4 simulaciones. Las salidas generadas estan relacionadas a α y β los cuales fueron divididos en C_α y C_β clases respectivamente.

en la Figura 5.18, donde los resultados mostrados son los obtenidos al considerar 20 clases para cada parámetro, por lo que cada clase consta de 25 patrones. Se observa un comportamiento extraño para el número de patrones clasificados correctamente, correspondientes a todas las posiciones y a la clase 19 en tamaño, que corresponden a los valores $\beta_{91}, \beta_{92}, \beta_{93}, \beta_{94}$ y β_{95}. Falta el mapa de colores de la MSV para saber si este problema sólo ocurre con la RNA o si se presenta en ambos métodos.

Finalmente, en el caso 4, donde consideramos un caso más realista con los obstáculos sobre las paredes del conducto, se presentan los resultados obtenidos para la estimación simultánea del tamaño y la posición del bloqueo en la Tabla 5.8. Similar al caso 3c, se obtiene un buen desempeño en ambos métodos para el conjunto de prueba cuando se consideran 5 y 10 clases, obteniendo un PPCC de por lo menos 95. También se ve una caída considerable en el PPCC por la RNA cuando se consideran 20 clases, logrando 86.24. Faltan los resultados para la MSV, pero se esperan resultados similares al caso 3c, disminuyendo el PPCC aún más que la RNA. Análogamente al caso 3c, se muestra el mapa de colores del total de simulaciones para la RNA cuando se consideran ambos parámetros divididos en 20 clases en la Figura 5.19. Nuevamente, se observa el comportamiento extraño presente en el caso previo, aunque menos marcado y en donde también falta el mapa de colores para la MSV para poder conocer si exhibe un comportamiento similar a la RNA en esta situación.

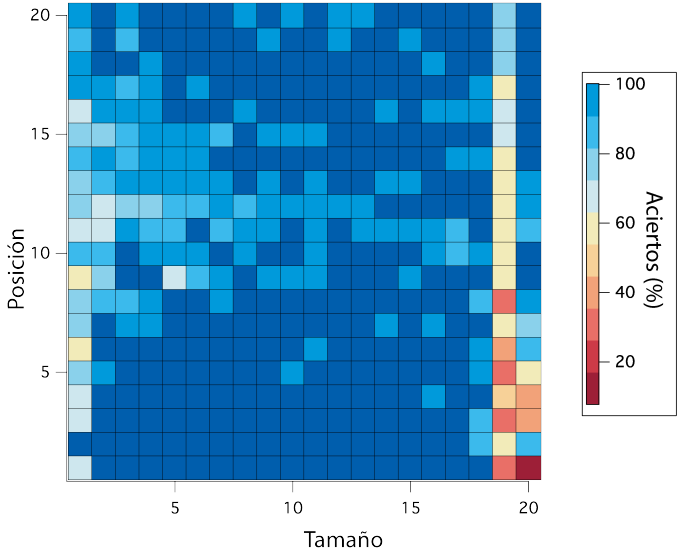

Figura 5.19: Porcentaje de patrones clasificados correctamente por la RNA para cada una de las clases cuando se consideran 400 clases en total en el caso 4. El incremento en el valor de la posición indica la clase a la que corresponden patrones generados usando las posiciones del obstáculo más cercanas al sensor. El incremento en el valor del tamaño indica una clase a la que corresponden patrones generados usando los tamaños del obstáculo más grandes. Entonces, la casilla inferior izquierda representa una clase donde los patrones fueron generados con los valores de α y β más pequeños, mientras que la casilla superior derecha representa una clase donde los patrones fueron generados con los valores de α y β más grandes.

caso 4				
PPCC (%)	RNA		MSV	
para	Entrenamiento	Prueba	Entrenamiento	Prueba
$C_\alpha = C_\beta = 5$	98.68	96.85	98.25	98.10
$C_\alpha = C_\beta = 10$	97.18	95.36	94.80	95.45
$C_\alpha = C_\beta = 20$	90.36	86.24		

Tabla 5.8: PPCC por la RNA y la MSV para los conjuntos de entrenamiento y de prueba, clasificando α y β simultáneamente y tomando 10^4 simulaciones para el caso realista, donde el bloqueo crece desde las paredes del conducto. Las salidas generadas están relacionadas a α y β los cuales fueron divididos en C_α y C_β clases respectivamente.

Enseguida se encuentran las conclusiones a las que llegamos al comparar los métodos de AA en los casos de estudio elaborados.

5.2.5. Conclusiones de la sección

En esta sección se desarrolló un estudio enfocado en la detección de bloqueos en tubos. Para este fin, implementamos dos métodos de aprendizaje automático para estimar la posición y el tamaño de la obstrucción presente en un conducto bidimensional. Para realizar estas estimaciones, entrenamos RNAs y MSVs con datos obtenidos de simulaciones numéricas elaboradas con un MLB para el flujo alrededor de un obstáculo cuadrado.

De los resultados obtenidos en la sección 5.2.4, se puede decir que en general, obtenemos un buen desempeño por ambos métodos en prácticamente todos los casos estudiados, sobresaliendo en la mayoría la MSV.
En el caso 1, estimamos independientemente la posición del obstáculo manteniendo su tamaño constante, obteniendo un PPCC de al menos 80.0 al usar la RNA y de 95.0 al usar la MSV. De manera similar, en el caso 2, estimamos independientemente el tamaño del obstáculo manteniendo su posición constante, obteniendo un PPCC de al menos 75.0 al usar la RNA y de 85.0 al usar la MSV. Esto muestra que es posible detectar la presencia de un bloqueo, al reconocer por ambos métodos los cambios en los patrones de flujo generados después del obstáculo, siendo capaces de estimar su

posición o su tamaño.
Sin embargo, para una aplicación práctica, uno quisiera poder estimar la posición y el tamaño del obstáculo a la vez. Por ello en el caso 3, estudiamos esta situación, utilizando el total de simulaciones generadas, primero estimando el tamaño y la posición independientemente como en los primeros 2 casos, antes de pasar a la situación general, donde se estiman los dos parámetros simultáneamente. En el caso 3a, predecimos independientemente el valor de la posición del obstáculo, obteniendo un PPCC en el conjunto de prueba de al menos 90.25 al usar la RNA y de 95.60 al usar la MSV, mientras que en el caso 3b predecimos independientemente el tamaño del obstáculo, obteniendo un PPCC en el conjunto de prueba de al menos 90.70 al usar la RNA y de 96.05 al usar la MSV. Para el caso 3c, donde simultáneamente predecimos la posición y el tamaño del obstáculo, el PPCC cuando consideramos 20 clases obtenido por la RNA es de 85.35 y por la MSV es de 81.50, estimando la posición con un error de ± 0.25u y el tamaño con un error de ± 0.025u. En esta ocasión, la RNA supera a la MSV para los resultados donde el error asociado al parámetro estimado es menor.
En la situación realista, que corresponde al caso 4, estimamos simultáneamente la posición y el tamaño del obstáculo, similar a lo realizado en el caso 3c, obteniendo un PPCC por la RNA de 86.24 y estimando la posición con un error de ± 0.25u y el tamaño con un error de ± 0.025u. Faltan los resultados pertinentes a la MSV cuando se consideran 20 clases, pero viendo la tendencia en los resultados, esperamos que el PPCC en esta situación sea aún mas bajo que en el caso 3c, por lo que tal vez en esta situación la RNA también supere a la MSV.
Finalmente, para poder tener conclusiones sobre los mapas de colores realizados en cuanto a las fallas que se observan en las clasificaciones realizadas por la RNA, faltan los resultados correspondientes a la MSV. Pensamos que el comportamiento presente en donde realiza más fallas la RNA, es debido a la manera en que se toma el conjunto de entrenamiento, pero esto solo es una suposición. Faltarían algunas pruebas adicionales para tener idea de lo que esta sucediendo en las clases donde clasifica incorrectamente.
En general, el desempeño de la MSV sobrepasa a la de la RNA, pero la RNA es mejor en las situaciones donde se estiman simultáneamente la posición y el tamaño del obstáculo con los errores asociados más pequeños.

A futuro esperamos hacer un análisis similar al presentado aquí, considerando conductos y obstrucciones con mayor complejidad, tratando de simular sistemas realistas en tres dimensiones.

En este capítulo se implementaron métodos de aprendizaje automático en sistemas físicos en los que pueden existir una aplicación industrial. Tal es el caso de bloqueos en tuberías que transportan fluido, contrario a los capítulos anteriores, donde los sistemas estudiados son más teóricos. Además, se muestra que no es necesario utilizar un enfoque de clasificación para utilizar los métodos de AA. Tal es el caso en el que se estima la morfología de la obstrucción en donde se utilizó un enfoque diferente.
Adicionalmente, desarrollamos una manera de visualizar los datos por medio de mapas de colores para saber en donde están fallando los métodos al clasificar los patrones, ya sea utilizando el enfoque de clasificación por salidas multiclase o utilizando un enfoque de clasificación uno contra todos como se mostró al clasificar el tamaño y la posición del obstáculo en un conducto.
En el próximo capítulo también se estudia un sistema físico al que se le puede dar una aplicación, pero en esta ocasión se trata de una aplicación en medicina.

Capítulo 6

Aplicación a curvas de Bragg

Este capítulo está basado en el artículo titulado *Characterizing Bragg peaks through an artificial neural network study* el cual esta pendiente de publicación y fue elaborado en colaboración con los Drs. José A. González y Alfredo Raya y con la MC Izamar Gutiérrez. Las simulaciones de las curvas que son utilizadas para entrenar la red neuronal artificial fueron realizadas por la MC Izamar Gutiérrez.

De entre las diferentes técnicas de radiación para tratamiento de cáncer, la terapia de hadrones exhibe algunas características remarcables que capturan el interés de una comunidad multidisciplinaria dedicada a enfrentar este problema. La fortaleza de esta terapia se basa en las propiedades de los hadrones, los cuales pueden penetrar la piel sin difusión y alcanzar el máximo depósito de energía justo antes de detenerse. Esta característica permite localizar la región precisa para ser irradiada sin comprometer el tejido de alrededor, como es el caso de la radiación convencional con rayos γ. En años recientes, la aceleración de haces de protones para este fin se ha vuelto una modalidad importante en el tratamiento de cáncer.

Cuando una partícula cargada entra en un medio absorbente, sufre de una pérdida continua de energía por procesos de ionización y excitación. Si la energía inicial E_0 de la partícula es mayor que la ionización y excitación promedio de los átomos en el medio, tal partícula pierde totalmente su energía después de viajar una distancia límite.

Por lo tanto, la física de la terapia de hadrones está basada en la noción de pérdida de energía (potencia de frenado) de un protón entrando en un tejido. Dentro de la aproximación de ralentizado continua (ARC), la regla de Bragg-Kleenman provee una relación determinada empíricamente por una ley de potencias entre la máxima distancia de penetración R_{ARC} y la energía del haz incidente E_0 [128].

Para el cálculo de la dosis, una estrategia favorita sigue siendo consultar tablas de localización y perfiles de curvas de Bragg usualmente extraídas directamente de datos [129, 130, 131, 132, 133, 134, 135]. Otros métodos incluyen el cálculo iterativo de dosis de protones entrando a un medio [136, 137] y el más eficiente, es usando estrategias con el método de Monte Carlo [130, 138, 139].

En este estudio, analizamos diferentes curvas de Bragg y localizamos los correspondientes picos de Bragg a través del uso de RNAs para tres anchos gaussianos considerados usualmente: (a) un ancho independiente de la energía; (b) un ancho con un comportamiento de ley de potencia en la energía inicial inspirado por la regla de Bragg-Kleenman y (c) un modelo fenomenológico capturando algunas características de dispersión real de protones en un medio [140]. Usando las simulaciones generadas correspondiente a cada caso, se emplea un enfoque por RNAs para analizarlas, ya que como hemos visto en los capítulos anteriores, las RNAs se pueden adaptar para analizar datos provenientes de prácticamente cualquier sistema y este caso no es la excepción.

En la próxima sección se deriva brevemente la forma más simple para la potencia de frenado en la terapia de hadrones y se describe como se simulan las curvas de Bragg para diferentes energías iniciales E_0 de los haces de protones.

6.1. Antecedente teórico

La Física de la terapia de protones está basada en la noción de pérdida de energía de un haz de partículas cargadas entrando en el tejido del paciente. Dentro de la ARC, la regla de Bragg-Kleeman estima el rango

máximo de penetración R_{ARC} de los protones con energía inicial E_0 en un medio homogéneo a través de la relación de ley de potencias

$$R_{\mathrm{ARC}} = AE_0^p. \tag{6.1}$$

Para los protones terapéuticos y asumiendo que el medio homogéneo es agua, se ha estimado que $p = 1.77$ y $A = 0.0022$ cm/MeV$^{\mathrm{p}}$ [141].
En esta aproximación, sólo se toma en cuenta la energía de transferencia de la partícula al medio homogéneo, en este caso agua, por amortiguamiento continuo del movimiento de la partícula y las fluctuaciones estadísticas no se toman en consideración. Sin embargo, la Ec. (6.1) no toma en cuenta la energía residual $E(z)$ o la potencia de frenado $S(z) = -dE/dz$ en la posición z dentro del tejido. Así, la integración completa de la Ec. (6.1) provee información acerca de $E(z)$ y $S(z)$ en cada posición z $(0 \leq z \leq R_{\mathrm{ARC}})$.

Al remplazar $R_{\mathrm{ARC}} \to z - R_{\mathrm{ARC}}$ y $E_0 \to E(z)$ en la Ec. (6.1) [140], se obtiene la que tal vez es la aproximación más simple a $E(z)$ y $S(z)$ como función de la posición z, pero aún así captura la física involucrada:

$$E(z) = A^{-1/p}(z - R_{\mathrm{ARC}})^{1/p} \;\Rightarrow\; -\frac{dE}{dz} = \frac{A^{-1/p}}{p}(R_{\mathrm{ARC}} - z)^{1/p-1}. \tag{6.2}$$

Además, las fluctuaciones estadísticas pueden ser fácilmente incorporadas al convolucionar el poder de frenado con un paquete de onda de ancho τ como [140]:

$$S(z,\tau) = \int_{-\infty}^{R_{\mathrm{ARC}}} S(u)\frac{1}{\tau\sqrt{\pi}}\exp\left(-\frac{(u-z)^2}{\tau^2}\right)du, \tag{6.3}$$

donde en nuestras consideraciones, $S(u)$ es el resultado en la Ec. (6.2). El ancho τ se considera de las siguientes tres maneras o casos de estudio:

1. $\tau = 0.5$,
2. $\tau = 0.01R_{\mathrm{ARC}}$,
3. $\tau = \sqrt{\tau_{\mathrm{s}}^2 + \tau_{\mathrm{i}}^2 + \tau_{\mathrm{h}}^2}$

utilizando,

$$\tau_{\mathrm{s}} = \tau_{\mathrm{s}}(R_{\mathrm{ARC}})\frac{\exp(zQ_{\mathrm{z}})-1}{\exp(Q_{\mathrm{z}}R_{\mathrm{ARC}})-1}; \qquad Q_z = 2.887\ \mathrm{MeV}^{-1};$$
$$\tau_{\mathrm{s}}(R_{\mathrm{ARC}}) = \sqrt{2}\cdot 0.012703276\cdot R_{\mathrm{ARC}}^{0.9358}; \qquad \tau_{\mathrm{i}} = 0.01R_{\mathrm{ARC}};$$
$$\tau_{\mathrm{h}} = 0.554111 - 0.000585437(E_0 - E_{\mathrm{res}}); \quad E_{\mathrm{res}} = 20.12\ \mathrm{MeV}.$$

En la Figura 6.1, mostramos distintas curvas de Bragg para cada modelo del parámetro de anchura considerado. Estas curvas describen la posición donde el haz deposita totalmente la energía; esto es, dada la energía inicial E_0, es posible conocer la posición exacta donde el tumor está localizado, con la distancia máxima de penetración R_{ARC}. Sin embargo, en un caso médico real, se conoce donde se encuentra el tumor y el problema consiste en determinar la energía inicial necesaria para penetrar el tejido e irradiar el tumor en el punto exacto. Para este propósito, se generaron mil curvas numéricamente para cada uno de los casos de la energía dependiendo del ancho τ con variaciones de energía de $\Delta = 0.25$ MeV, para energías iniciales $50 \leq E_0 < 300$ MeV.
Después de tener el catálogo de curvas simuladas para cada caso considerado, se utilizaron para entrenar la RNA y poder clasificar la energía inicial utilizada en la generación de las curvas.

La descripción y las consideraciones tomadas en cuenta para la implementación de la RNA se encuentran a continuación.

6.2. Consideraciones de la RNA

Para analizar las simulaciones, se implementó una red neuronal artificial en FORTRAN con un algoritmo de retropropagación para minimizar una función de costo de error cuadrático medio, usando una estructura que consiste de una capa de entrada, una capa oculta y una capa de salida. La red es entrenada para clasificar los patrones de entrada simulados de acuerdo a la energía inicial E_0 del haz: etiquetamos las simulaciones en un total de C clases, donde cada clase tiene el mismo número de simulaciones o patrones.

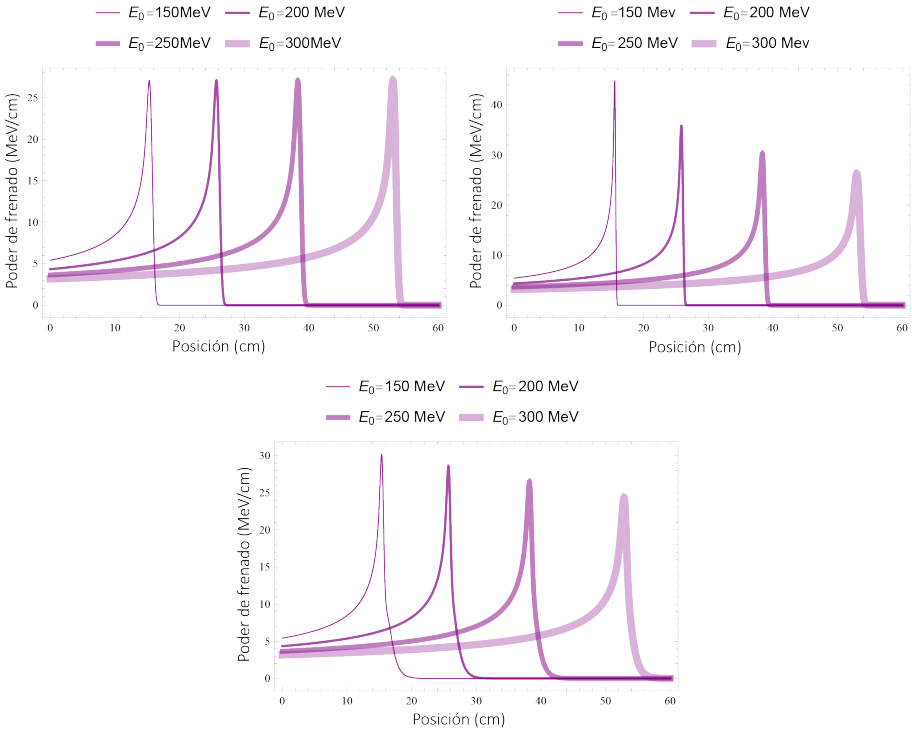

Figura 6.1: Comparación de los picos de Bragg para cada caso de τ, considerando $150 \leq E_0 \leq 300$ MeV. *Panel superior izquierdo:* ancho constante; *Panel superior derecho:* ancho de ley de potencias; *Panel inferior:* ancho fenomenológico.

La capa de entrada contiene cien neuronas, las cuales reciben una versión simplificada de las curvas originales, donde el patrón de entrada para cada simulación incluye $I = \{E_0(z = 0.6i)\}$, para $1 \leq i \leq 100$. Seleccionamos sólo el diez por ciento del número total de puntos en cada curva para reducir el tiempo computacional del entrenamiento. Además, los patrones de entrada son escalados de acuerdo al máximo valor en los datos de entrada $\hat{I} = I/\max(I)$.

Consideramos un número de diferente de neuronas en la capa oculta, los cuales son iguales a 2^j, con $2 \leq j \leq 6$ y la estructura seleccionada es aquella que obtiene el mejor desempeño después de entrenarse.

Finalmente, la capa de salida cuenta con solo una neurona y el valor que genera es asociado con la clase a la que pertenece el patrón procesado. Las neuronas en las capas oculta y de salida están dotadas con funciones de activación sigmoides, por lo que los objetivos propuestos para las C diferentes clases satisfacen la Ec. (2.18). Para este estudio, decidimos usar cien clases diferentes para las mil simulaciones, asociando 10 simulaciones a cada clase, por lo que $C = 100$. Considerando que E_0 tiene valores entre 50 y 300 MeV, asociamos los primeros diez valores ${E_0}^1 = 50$, ${E_0}^2 = 50.25$, ..., ${E_0}^{10} = 52.25$ a la primer clase, los siguientes diez valores son asociados con la segunda clase y así sucesivamente, siendo los últimos diez valores ${E_0}^{991} = 297.5$, ..., ${E_0}^{1000} = 299.75$ asociados a la clase 100, donde el superíndice denota la curva o patrón asociado a esa energía.

Una vez que se entrena la red, definimos que los patrones son clasificados correctamente, si los valores de salida O generados por la red satisfacen la Ec.(2.19).
Como estamos considerando conjuntos de diez simulaciones por clase y están ordenados del mínimo al máximo valor de E_0, a cada clase se le puede asociar un promedio de acuerdo a la Ec. (2.20):

$$\hat{E}_c = E_{min} + (2c - 1)E_L/2C \pm E_L/2C, \quad 1 \leq c \leq 100, \tag{6.4}$$

donde $E_L = E_{max} - E_{min} = E_0^{1000} - E_0^1$ y $\hat{E}_c$ es el valor promedio de E_0 para la clase c. El tercer término del lado derecho es el error con respecto a la longitud total E_L, que corresponde a un error físico de $\pm 0.5\,\%$ o ± 1.25

MeV. De la Ec. (6.4), se puede observar que al incrementar el número de clases, el error asociado al valor E_0 decrece, pero el precio a pagar es que el desempeño de la red disminuye. Vale la pena mencionar que cuando la red realiza una clasificación incorrecta, la clase correcta usualmente es un vecino cercano de la clase predicha. Con esto en mente, es posible considerar errores asociados más grandes, por ejemplo $\pm 1.0\,\%$ de E_L (± 2.5 MeV) en vez de $\pm 0.5\,\%$, aumentando el porcentaje de clasificaciones correctas por la red.

Una vez propuestos los objetivos se divide el total de simulaciones en los conjuntos de entrenamiento, validación y predicción. Estos conjuntos están compuestos por patrones seleccionados aleatoriamente: el conjunto de entrenamiento contiene el $70\,\%$ de las simulaciones, mientras que los conjuntos de validación y predicción contienen $15\,\%$ cada uno.

En la próxima sección se muestran los resultados de entrenar la red con todas estas consideraciones tomadas en cuenta.

6.3. Resultados

El entrenamiento de la red fue realizado para un total de 2×10^4 iteraciones en cada caso y monitoreando la condición de que si el costo en el conjunto de validación incrementa, el entrenamiento se detiene para evitar un sobre entrenamiento. Se exploraron 25 combinaciones de parámetros de la red, seleccionando una constante de aprendizaje igual a 3^{-l} para $3 \leq l \leq 7$, con la cantidad de neuronas ocultas consideradas. Este análisis se hizo usando un código paralelizado en MPI.

Finalmente, tomamos aleatoriamente una de las simulaciones en el conjunto de prueba y se compara en cada caso, la energía asociada a la curva simulada de Bragg y la curva de Bragg generada usando la energía estimada por la RNA. En la Figura 6.2 presentamos estas comparaciones, en donde las curvas estimadas son muy parecidas a las simulaciones reales. El caso 2 muestra que la curva estimada tiene el pico más desplazado comparado con el de simulación, lo cual está de acuerdo con los resultados de la Tabla 6.1. Una vez concluido el entrenamiento, seleccionamos la RNA con la función de costo más pequeña, siendo la estructura con 64 neuronas

PPCC (%)			
Error	Entrenamiento	Validación	Prueba
Caso 1			
± 0.5 %	88.2	85.3	80.6
± 1.0 %	99.5	96.6	96.6
Caso 2			
± 0.5 %	54.2	47.3	40.6
± 1.0 %	79.5	74.0	71.3
± 1.5 %	93.0	86.6	90.0
Caso 3			
± 0.5 %	87.2	79.3	80.6
± 1.0 %	99.5	98.0	98.6

Tabla 6.1: Porcentaje de patrones clasificados correctamente por la RNA en cada conjunto para los tres casos considerados.

ocultas para el caso uno y 32 neuronas para los casos dos y tres con una constante de aprendizaje de 3^{-3} las mejores. En cada caso, probamos el desempeño de la red evaluando el porcentaje de patrones clasificados correctamente (PPCC), mostrando los resultados en la Tabla 6.1. La tabla muestra el PPCC con un error de $\pm 0.5\,\%$, incrementándose el error en pasos de $\pm 0.5\,\%$ hasta que al menos $90\,\%$ de los patrones en el conjunto de prueba son clasificados correctamente. Se puede observar aquí que cuando el error asociado al valor estimado E_0 es pequeño, el PPCC también es pequeño, donde este comportamiento está presente en todos los casos. Los casos 1 y 3 tienen un desempeño parecido, miestras que el caso 2 tiene el peor desempeño con un eror asociado de $\pm 1.5\,\%$, necesario para alcanzar $90\,\%$ de patrones clasificado correctamente en el conjunto de prueba.

6.4. Conclusiones del capítulo

En este capítulo desarrollamos una estrategia numérica para clasificar curvas de Bragg utilizando un enfoque con redes neuronales artificiales. Al simular el poder de frenado con tres modelos del ancho gaussiano obtenido

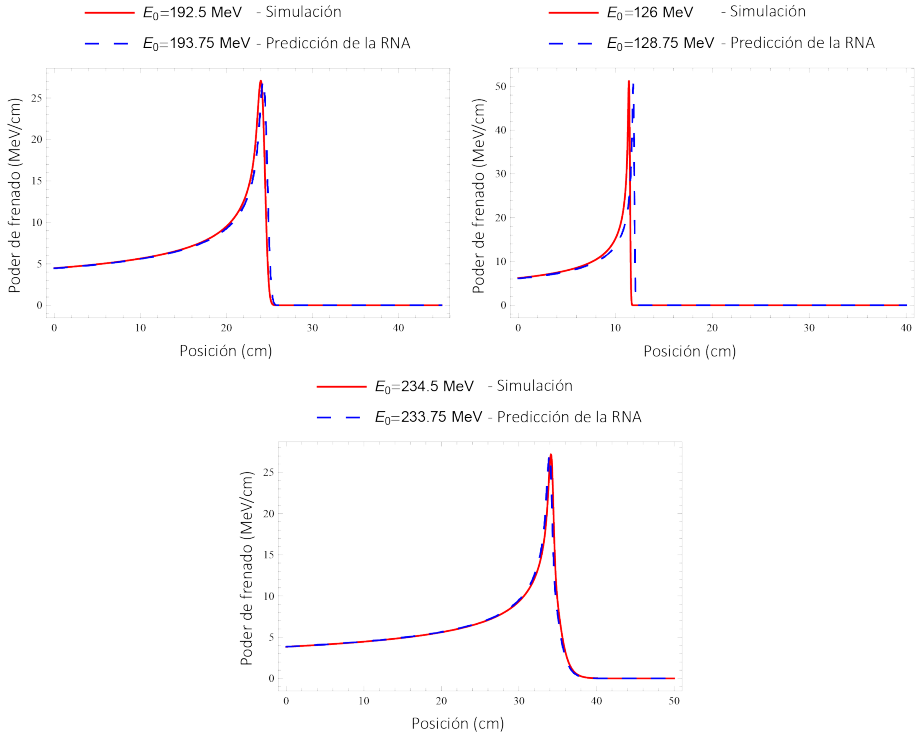

Figura 6.2: Comparación de las curvas de Bragg entre las simulaciones calculadas y las estimadas por la RNA. *Panel superior izquierdo:* ancho constante; *Panel superior derecho:* ancho con ley de potencias; *Panel inferior:* ancho fenomenológico.

dentro de la aproximación de ralentización continua más simple, entrenamos RNAs para clasificar curvas simuladas, obteniendo al menos un 90 % de patrones clasificados correctamente en el conjunto de prueba, con un error asociado de 1.5 % correspondiente a una energía de $\pm$ 3.75 MeV en el valor estimado de la energía inicial. En el mejor escenario, la RNA clasifica correctamente los patrones en el conjunto de prueba 98.6 % de las veces con un error de $\pm$ 2.5 MeV en el valor estimado de la energía inicial.

Estos resultados pueden ser mejorados si consideramos un perfil más refinado del poder de frenado e incluso extender al caso de un medio no homogéneo como en las aplicaciones realistas [140].

Pasaremos ahora al último capítulo donde se muestran las conclusiones de los métodos de AA, así como del enfoque de clasificación que se utilizó en el estudio de los sistemas físicos analizados hasta este momento.

Capítulo 7

Conclusiones

Con la descripción de los trabajos presentados en cada uno de los capítulos, se espera poder dejar claro que los métodos de aprendizaje automático tienen una gran flexibilidad para adaptarse a prácticamente cualquier problema. Se comenzó procesando datos extraídos de simulaciones muy sencillas, como en el caso del estudio de oscilaciones de Bloch en una cadena lineal monoatómica, en donde el método de AA que se entrenó únicamente recibe dos datos de entrada por patrón y clasificó un total de 100 señales o simulaciones, hasta clasificar un total de 10000 señales recibiendo 300 datos de entrada por simulación en el caso de estimar la posición y tamaño de un bloqueo en una tubería por la que fluye un fluido. Poco a poco en cada uno de los sistemas físicos estudiados, se incrementó tanto la cantidad de datos analizados, como la complejidad de las herramientas utilizadas. Y con complejidad, me refiero a métodos de AA que pueden ser entrenados en horas haciendo uso de poder de cómputo de una computadora de escritorio.
La investigación que realicé durante el doctorado en colaboración con profesores y colegas del instituto, dio como resultado, la publicación de seis artículos en revistas internacionales, un artículo al que nos falta encontrar una revista adecuada para enviarlo y un artículo pendiente de algunos resultados finales.

En ocasiones, al hacer más complejos los métodos de AA pueden surgir complicaciones, tal es el caso de las redes neuronales artificiales profundas,

que al contar con muchas capas ocultas pueden experimentar problemas de desvanecimiento o de explosión del gradiente [142, 143] y por lo que se tienen que emplear técnicas adicionales para evitar estos problemas. Por esta razón, a veces se pueden utilizar otros enfoques o técnicas para tratar de mejorar el desempeño del método de AA, antes de recurrir a utilizar un método de mayor complejidad. Es por esto que se utilizó el método de clasificación de salidas multiclase y se estudiaron varios aspectos del mismo:

- Con este enfoque, dependiendo del número de clases en el que se quieren clasificar los patrones que se van a procesar, es la cantidad de error asociada al parámetro físico predicho, donde por ejemplo en el caso del estudio de curvas de Bragg se alcanzó hasta un 98.6 % de patrones clasificados correctamente estimando la energía inicial del haz de partículas con un error de $\pm 1\,\%$. En general, el desempeño de la red mejora cuando el número de clases consideradas es menor, pero a cambio de un mayor error asociado al parámetro que se predice.

- Al analizar las OB en una malla cuadrada, se estudió como cambia el desempeño de una RNA al cambiar el intervalo donde se varía el parámetro físico de interés al generar las simulaciones. En este caso, la intensidad del campo eléctrico externo E, aumentando el desempeño de la red cuando el intervalo considerado para E es más pequeño.

- Se estudió en el caso de las OB en grafeno, como el porcentaje de patrones clasificados correctamente por una RNA entrenada con un cierto error asociado, aumenta si se considera un error del doble. Esto indica, que aún cuando no se clasifica correctamente alguno de los patrones, la clase predicha se encuentra muy próxima a la clase a la que si pertenecen los patrones, es decir, se encuentra en una clase vecina.

- Además de requerir un menor tiempo de entrenamiento, en ocasiones es mejor utilizar el enfoque de salidas multiclase, debido a que se puede incluso mejorar el desempeño del método de AA utilizado, tal es el caso de cuando se usa un regresor o una red neuronal convolucional.

Aunado a estas propiedades del enfoque de clasificación con salidas multiclase, se desarrollaron dos técnicas adicionales, uno para aumentar la precisión en los parámetros estimados y otro para visualizar la región donde los patrones analizados son clasificados incorrectamente por los métodos de AA. Usando un proceso de refinamiento como el descrito en el Capítulo 4, se puede alcanzar una mayor precisión en el parámetro físico estimado, disminuyendo el error asociado al mismo, pero a cambio de realizar un mayor número de simulaciones. Esto se traduce en un mayor tiempo de cómputo al realizar tanto las simulaciones, como los entrenamientos adicionales de las redes. Adicionalmente, en el Capítulo 5, se desarrolló una herramienta a través de mapas de colores para localizar regiones donde los métodos de AA fallan en clasificar los patrones en la clase correspondiente. Se utilizó también un análisis parecido para visualizar los datos en una dimensión, donde comparamos los diferentes métodos y nos dimos cuenta que el comportamiento de tales regiones, al parecer depende únicamente del enfoque que se utiliza.

Tomando todo lo anterior en cuenta, creo que este enfoque de salidas multiclase tiene potencial para afrontar diferentes problemas como se mostró en los sistemas físicos analizados. Además, en la mayoría de los sistemas estudiados, se trató de mantener el menor número de parámetros internos de los métodos de AA, por lo que aún existe margen de mejora si a parte se introduce, por ejemplo, un parámetro de regularización y de momento en las redes neuronales.

Me percaté también de la importancia de realizar un estudio más detallado a la hora de preprocesar los datos de entrada y la manera en que se escalan los mismos, antes de emplear tiempo en optimizar los distintos parámetros internos del método utilizado. Todo esto, junto con la manera en que se proponen los objetivos, puede causar que el desempeño del método mejore, antes de pensar en considerar un método con mayor complejidad que requiera más tiempo de entrenamiento.

Finalmente, como trabajo a futuro, espero poder usar el enfoque de clasificación por salidas multiclase, incluyendo parámetros adicionales en los métodos de AA como es el parámetro de regularización y el término de momento. Es de interés usar este enfoque aunado al método de refinamien-

to y la visualización por mapa de colores, para saber el desempeño de los métodos al estudiar los sistemas expuestos en esta tesis, pero en situaciones de mayor complejidad.
Además, se espera incluir señales más reales, donde tales señales puedan incluir ruido, no tengan una dirección preferencial y el tiempo en el que se analizan no sea de manera tan conveniente como las que se presentaron en esta tesis. En situaciones del mundo real, las señales incluirán ruido y tal vez no se pueda tener tanto control sobre las mismas, ya que en nuestro caso nosotros simulamos las señales y convenientemente en nuestros ejes de referencia, las señales comienzan en el origen. Por este motivo, si trabajamos con señales reales, necesitamos considerar un preprocesado de las mismas, para poder transformar el problema a una situación como las que se describen en la tesis.

Espero haber podido convencer al lector con esta tesis, de que los métodos de aprendizaje automático son muy flexibles para analizar cualquier situación de interés, razón por la cual en la actualidad tienen un gran uso y el uso de tales métodos no se restringe sólo a sistemas físicos como los presentes en la tesis, sino que convivimos prácticamente con ellos día a día.

Bibliografía

[1] Ling J., Jones R. and Templeton J. *Journal of Computational Physics.* **318**. 2016

[2] Nathan J. *Journal of Fluid Mechanics.* **814**. 2017

[3] Sender D., Kyle W. and Joni D. *Monthly Notices of the Royal Astronomical Society.* **450**. 2015

[4] Zhenwei Li, James R. and Alessandro De Vita. *Phys Rev. Lett.* **114**. 2015

[5] Baldi P., Sadowski P. and Whiteson D. *Nature Communications.* **5**. 2014

[6] Weisberg S. *Wiley.* 2005

[7] Yan X., Gang Su X. *World Scientific.* 2009

[8] Aly Mohamed. *Survey on multiclass classification methods.* 2005

[9] Rojas R. Neural Networks. *Springer-Verlag.* 1996

[10] López C. *Tesis de maestría.* UMSNH. 2014

[11] C. Bishop. *Springer-Verlag.* 2006

[12] I. Goodfellow, Y. Bengio and A. Courville. *MIT Press.* 2016

[13] C. C. Chang and C. J. Lin, *ACM Transactions on Intelligent Systems and Technology.* **2**, 3. pp. 1-27. 2011

[14] S. Abe, *Springer*. 2005

[15] B. Scholkopf and A. Smola. *The MIT Press*. 2002

[16] Esaki, L. y Tsu R. *J. Res. Dev.* **61**. 1970

[17] Zener C. *Proc R. Soc. A* 137:696. 1932

[18] Feldmann J., Leo K., Shah J., Miller D. A. B., Cunningham J. E., Meier T., von Plessen G., Schulze A., Thomas P., and Schmitt-Rink S. *Phys. Rev. B.* **4**6. 7252. 1992

[19] von Plessen G. and Thomas P. *Phys. Rev. B.* **4**5, 9185 1992

[20] Leo K., Bolivar P. H., Brüggemann F., Schwedler R., and Köhler K. *Solid State Commun.* **8**4. 943. 1992

[21] Leisching P., Haring Bolivar P., Beck. W., Dhaibi Y., Brüggemann F., Schwedler R., Kurz H., Leo K., and. Köhler K. *Phys. Rev. B.* **5**0 14389. 1994

[22] Dekorsy T., Leisching P., Köhler K., and Kurz H. *Phys. Rev. B.* **50**. 8106. 1994

[23] Dekorsy T., Ott R., Kurz H., and Köhler K. *Phys. Rev. B.* **51**. 17275. 1995

[24] Waschke C., Roskos H. G., Schwedler R., Leo K., Kurz H., and Köhler K. *Phys. Rev. Lett.* **70**. 3319. 1993

[25] Roskos H. G., Waschke C., Schwedler R., Leisching P., Dhaibi Y., Kurz H., and Köhler K. *Superlattices and Microstructures.* **15**. 281. 1994

[26] Kolovsky A. R. and Korsch H. J. *Phys. Rev. A*. **67**. 063601. 2003

[27] Witthaut D., Keck F., Korsch H. J. and Mossmann S. *New J. Phys.* **6**. 41. 2004

[28] Dahan M. B, Peik E., Reichel J, Castin Y. and Salomon, C. *Phys. Rev. Lett.* **76**, 45084511. 1996

[29] Genske, M. et al., *Phys. Rev. Lett.*, **110**. 190601. 2013

[30] Pertsch T, Dannberg P, Elflein W, Bräuer, A and Lederer F. *Phys. Rev. Lett.* **83** 4752. 1999

[31] Morandotti R., Peschel U., Aitchison J. S., Eisenberg H. S., and Silberberg Y. *Phys. Rev. Lett.* **83** 47564759. 1999

[32] Sapienza R et al. *Phys. Rev. Lett.* **91**, 263902. 2003

[33] A. Block, et al. *Nat. Commun.* 53843. 2014

[34] Cheng Hemeng et al. *App. Phys. Lett.* **105**, 072103. 2014

[35] Changan Li et al. *Appl. Phys. Lett.* **103**, 172106. 2013

[36] Wallace, P. R.: The Band Theory of Graphite, *Phys. Rev.* **71** 622634. 1947

[37] Novoselov, K. S., et al. *Proc. Natl Acad. Sci. USA*. **102**, 10451. 2005

[38] Zhang Y, et. al. *Nature*. **438**, 201. 2005

[39] Geim, A. K. y Novoselov, K. S. *Nature Materials*. **6**, 183191. 2007

[40] Vafek, O. y Vishwanath, A. *Ann. Rev. Cond. Mat. Phys.* **5**, 83-112. 2014

[41] Mermin, N.D., Wagner, H. *Phys. Rev. Lett.* **17**. 1966

[42] J. A. González, C. E. López, S. Hernández-Ortiz and A. Raya, *Plasmonics*. **13**, 9. 2016

[43] Proakis J. G. and Manolakis D. G. *Prentice Hall*. 2006

[44] M. Carrillo, González J. A., Hernández-Ortiz S., López C. E. and Raya A. *Computational Materials Science*. 2017

[45] M. Carrillo, J. A. González, S. Hernández-Ortiz, C. E. López and A. Raya, *Comp. Cond. Mat.* **13**, 104. 2017

[46] Novoselov K S, McCann E, Morozov S V, Falko V I, Katsnelson M I, Zeitler U, Jiang D, Schedin F and Geim A K. *Nat. Phys.* **2**, 177. 2006

[47] Katsnelson M I. *Materials Today* **10**, 20. 2007

[48] Novoselov K S, Jiang Z, Zhang Y, Morozov S V, Stormer H L, Zeitler U, Maan J C, Boe-binger G S, Kim P and Geim A K, *Science* **315**, 1379. 2007

[49] Geim A K and Novoselov K S. *Nat. Mater.* **6**, 183. 2007

[50] Pereira V M, Castro Neto A H and Peres N M R. *Phys. Rev. B.* **80**, 045401. 2009

[51] García-Naumis G, Barraza-Lopez S, Oliva-Leyva M and Terrones H. *Rep. Prog. Phys.* **80**, 096501. 2017

[52] Chen R, Ma T, Wang L-G and Lin H-Q. e-print: arXiv:1301.3221 [cond-mat.mes-hall]. 2013

[53] González J.A., López C.E., Raya A. *Comp. Cond. Mat.* **16**. 2018

[54] S. Hernández-Ortiz et al., work in progress.

[55] Oliva-Leyva M and García-Naumis G. *Phys. Rev. B.* **88**, 085430. 2013

[56] F. Rivera-Paleo, C. E. López, F.S. Guzmán, J. A. González, *Phys. Rev. D.* **95**, pp. 1-9. 2017

[57] T. Piran, *Rev. Mod. Phys.* **76**, 1143. 2004

[58] S. E. Woosley, J. S. Bloom, *ARA&A*. **44**, 507. 2006

[59] D. B. Fox, P. Mészáros. *New. J. Phys.* **8**, 199. 2006

[60] N. Gehrels, E. Ramirez-Ruiz, D. B. Fox, *ARA&A*. **47**, 567. 2009

[61] P. Kumar, B. Zhang, *Phys. Rep.* **561**, 1. 2015

[62] B. Paczyński, *ApJ.* **494**, L45. 1998

[63] A. S. Fruchter, A. J. Levan, L. Strogler et al. *Nature.* **441**, 463-468. 2006

[64] J. M. Castro Cerón, M. J. Micha lowski, J. Hjorth, D. Watson, J. P. U. Fynbo, J. Gorosabel. *ApJ.* **653**, L85-L88. 2006

[65] B. Zhang, S. Kobayashi, P. Mészáros. *ApJ.* **595**, 950. 2003

[66] A. Mizuta, T. Yamasaki, S. Nagataki, S. Mineshige. *ApJ.* **651**, 960. 2006

[67] B. J. Morsony, D. Lazzati, M. C. Begelman. *ApJ.* **665**, 569. 2007

[68] O. Bromberg, E. Nakar, T. Piran, R. Sari. *ApJ.* **740**, 100. 2011

[69] D. Lazzati, B. J. Morsony, R. Margutti, M. C. Begelman. *ApJ.* **765**, 103. 2013

[70] D. López-Cámara, B. J. Morsony, M. C. Begelman, D. Lazzati. *ApJ.* **767**, 19. 2013

[71] F. De Colle, Lu Wenbin, P. Kumar, E. Ramirez-Ruiz, G. Smoot. arxiv:1701.05198 [astro-ph.HE]

[72] F. J. Rivera-Paleo and F. S. Guzmán. *MNRAS.* **459**, 2777-2786. 2016

[73] C. Cuesta-Martínez, M. A. Aloy, P. Mimica. *MNRAS.* **446**, 1716. 2015

[74] S. J. Russell and P. Norvig. *Prentice-Hall.* 563-566. 1995.

[75] Y. LeCun, L. Bottou, G. Orr and K. Mulle. *Springer.* 1998.

[76] D. E. Rumelhart, G. E. Hinton and R. J. Williams. *Nature.* **323**, 533-536 .1986

[77] M. Carrillo, M. Gracia-Linares, J. A. González, F. S. Guzmán. *Gen. Rel. Grav.* **48**, 141. 2016

[78] A. Mignone, T. Plewa, G. Bodo. *ApJS.* **160**, 199. 2005

[79] B. Dubroca, J. L. Feugeas. *CRAS.* **329**, 915. 1999

[80] M. González, E. Audit, P. Huynh, *A&A*. **464**, 429. 2007

[81] C. D. Levermore. *J. Quant. Spectrosc. Radiative Transfer.* **31**, 149. 1984

[82] A. Pe'er, F. Ryde. *ApJ.* **732**, 49. 2011

[83] D. Giannios. *MNRAS.* **422**, 3092. 2012

[84] F. De Colle, E. Ramirez-Ruiz, J. Granot, D. López-Camara. *ApJ.* **751**, 57. 2012

[85] G. B. Rybicky, A. P. Lightman. *WILEY-VCH.* 2004

[86] S. Mendoza, J. C. Hidalgo, D. Olvera, J. I. Cabrera. *MNRAS.* **395**, 1403. 2009

[87] M. Carrillo, J. A. González, C. E. López and U. Que. *Phys. Rev. E.* **96**, pp. 1-10. 2017

[88] A. A. Mohamad. *Springer-London.* 2011

[89] S. Chen and G. D. Doolen. *Annual Review of Fluid Mechanics* **30**, 329. 1998

[90] W. N. Kernan et al. *Stroke.* **45**, 2160. 2014

[91] C. E. Davies and A. Desai. *Powder Technology* **183**, 436. 2008

[92] E. D. Sloan. *Fluid Phase Equilibria.* **228**, 67. 2005

[93] F. Y. Fraige and P. A. Langston. *Granular Matter.* **8**, 67. 2006

[94] S. P. Datta, P. K. Das, and S. Mukhopadhyay. *Applied Thermal Engineering.* **70**, 925. 2014

[95] H. Wang, J. Tian, H. Ouyang, Y. D. Wu, and Z. H. Du. *International Journal of Refrigeration-Revue Internationale Du Froid.* **46**, 173. 2014

[96] H. F. Duan. *Journal of Water Resources Planning and Management.* **142**, 8, 04015073. 2016

[97] R. M. Ashley, A. Fraser, R. Burrows, and J. Blanksby. *Urban Water.* **2**, 263. 2000

[98] J. Bailey, E. Keedwell, S. Djordjevic, Z. Kapelan, C. Burton, and E. Harris. *Procedia Engineering.* **119**, 1288. 2015

[99] J. P. Rodriguez, N. McIntyre, M. Diaz-Granados, and C. Maksimovic. *Water Research.* **46**, 4571. 2012

[100] J. Ma, M. J. S. Lowe, and F. Simonetti. *Measurement Science and Technology.* **18**, 2629. 2007

[101] N. L. T. Lile, M. H. M. Jaafar, M. R. Roslan, and M. S. M. Azmi. *International Journal on Advanced Science, Engineering and Information Technology.* **2**. No. 3, 252. 2012

[102] C. Massari, T. C. J. Yeh, M. Ferrante, B. Brunone, and S. Meniconi. *Journal of Hydroinformatics.* **16**, 248. 2014

[103] M. Carrillo, U. Que, and J. A. Gonzalez. *Physical Review E.* **94**, 8, 063304. 2016

[104] C. X. Pan, L. S. Luo, and C. T. Miller. *Computers & Fluids.* **35**, 898. 2006

[105] C. K. Aidun and J. R. Clausen. *Annual Review of Fluid Mechanics.* **42**, 439. 2010

[106] D. Z. Yu, R. W. Mei, L. S. Luo, and W. Shyy. *Progress in Aerospace Sciences.* **39**, 329. 2003

[107] M. Bouzidi, M. Firdaouss, and P. Lallemand. *Physics of Fluids.* **13**, 3452. 2001

[108] Z. G. Feng and E. E. Michaelides. *Journal of Computational Physics.* **195**, 602. 2004

[109] D. Z. Yu, R. W. Mei and W. Shyy. *Progress in Computational Fluid Dynamics.* **5** (1-2), 3-12. 2005

[110] M. Schafer, S. Turek, F. Durst, E. Krause, and R. Ran-nacher. *DFG Priority Research Programme Results 19931995.* edited by E. H. Hirschel. 1996

[111] O. Bello, N. Virani, and S. O. Oyadiji. *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.* **1**, 551. 2011

[112] N. Adeleke, M. T. Ityokumbul, and M. Adewumi. *Spe Journal.* **18**, 355. 2013

[113] I. Bright, G. Lin, and J. N. Kutz. *Physics of Fluids.* **25**, 15, 127102. 2013

[114] P. Lauret, F. Heymes, L. Aprin, A. Johannet and P. Slangen. *Chem. Eng. Trans.* **43**, 1621-1626. 2015

[115] O. Duran, K. Althoefer, and L. D. Senevatne. *Ieee Transactions on Automation Science and Engineering.* **4**, 118. 2007

[116] M. Al-Naser, M. Elshafei, and A. Al-Sarkhi. *Journal of Petroleum Science and Engineering.* **145**, 548. 2016

[117] E. S. Rosa, R. M. Salgado, T. Ohishi, and N. Mastelari. *International Journal of Multiphase Flow.* **36**, 738. 2010

[118] B. S. Yang, W. W. Hwang, M. H. Ko, and S. J. Lee. *Journal of Sound and Vibration.* **287**, 25. 2005

[119] S. R. Mounce, R. B. Mounce, and J. B. Boxall. *Journal of Hydroinformatics.* **13**, 672. 2011

[120] D. S. Kim, S. Shin, G. B. Choi, K. H. Jang, J. C. Suh, and J. M. Lee. *Canadian Journal of Civil Engineering.* **44**, 707. 2017

[121] T. Krueger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, and E. M. Viggen. *Springer.* 2016

[122] M. Sheikholeslami, M. Gorji-Bandpy, and D. D. Ganji. *Powder Technology.* **254**, 82. 2014

[123] E. S. Boek and M. Venturoli. *Computers & Mathematics with Applications.* **59**, 2305. 2010

[124] H. a. Huang. *Wiley Blackwell.* 2015

[125] J. F. Zhang, P. C. Johnson, and A. S. Popel. *Physical Biology.* **4**, 285. 2007

[126] S. Izquierdo and N. Fueyo. *Physical Review E.* **78**, 7, 046707. 2008

[127] Q. S. Zou and X. Y. He. *Physics of Fluids.* **9**, 1591. 1997

[128] Podgorsak. *Springer.* 2010

[129] Hong et. al. *L. Phys. Med. Biol.* **41**. 1996

[130] Petti. *P. Int. J. Rad. Oncol. Biol. Phys.* **35**. 1996

[131] Deasy. *J. O. Med. Phys.* **25**. 1998

[132] Schaffier, B., and Pedroni, E., Lomax. *A. Phys. Med. Biol.* **44**. 1998

[133] Russel et. al. *K. R. Phys. Med. Biol.* **45**. 2000

[134] Szymanowski, H. and Oelfke. *U. Phys. Med. Biol.* **47**. 2002

[135] Ciangaru et. al. *G. Med. Phys.* **32**. 2005

[136] Sandison et. al. *G. Med. Phy.* **24**. 1997

[137] Hollmark et. al. *M. Phys. Med. Biol.* **49**. 2004

[138] Tourovsky et. al. *A. Phys. Med. Biol.* **50**. 2005

[139] Jiang, H. and Paganetti. *H. Med. Phys.* **31**. 2004

[140] Ulmer, W. and Matsinos, E. *Eur. Phys. J. Special Topics.* **190**. 2010

[141] Bortfeld. *T. Med. Phys.* **24**. 1997

[142] S. Hochreiter. *Technische Universität München.* 1991

[143] Y. Bengio, P. Simard, and P. Frasconi. *IEEE Transactions on Neural Networks.* 5(2):157–166. 1994.