UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO

INSTITUTO DE FÍSICA Y MATEMÁTICAS

"CLASIFICACIÓN DE ESPECTROS ELECTROMAGNÉTICOS EN EL RANGO VISIBLE UTILIZANDO APRENDIZAJE DE MÁQUINA"

Tesis para obtener el grado de:
MAESTRO EN CIENCIAS EN EL ÁREA DE FÍSICAS

PRESENTA:
Gonzalo Rafael Vargas Castañeda

Asesor:

Doctor en Ciencias
José Antonio González Cervera

Co-Asesor:

Doctor en Ciencias con especialidad en óptica
Mauricio Ortiz Gutiérrez

Morelia, Michoacán - Agosto de 2018

Dedicado a mi hija, mis padres, mi esposa y mis hermanos, gracias por siempre estar ahí.

II

Resumen

Los espectrofotómetros son instrumentos que miden parámetros de una muestra, tales como absorbancia, transmitancia o reflectancia, como función de la longitud de onda y sus principales componentes son la fuente emisora de luz, el elemento difractivo y el detector de irradiancia. Dichos instrumentos son de amplio uso en laboratorios de química, física de materiales, medicina, entre otros. En esta tesis se presenta el diseño de un espectrofotómetro en el rango visible del espectro electromagnético que esta compuesto por un LED de luz blanca, una rejilla holográfica y una cámara de la marca Samsung como detector; los espectros que se generarán al colocar una muestra líquida en el espectrofotómetro diseñado serán analizados mediante tres algoritmos de inteligencia artificial que son redes neuronales artificiales, redes neuronales convolucionales y máquinas de soporte vectorial. Este tipo de algoritmos pertecen al llamado aprendizaje de máquina, que actualmente es utilizado para resolver problemas de clasificación y regresión, como lo son reconocimiento facial, reconocimiento del habla, motores de búsqueda y diagnóstico médico, entre otros. En esta tesis se implementaron estos algoritmos con la finalidad de encontrar cual de ellos es el mejor para resolver el problema de clasificación de muestras, en cuestión de eficiencia y tiempo de cómputo.

Palabras clave: Espectrofotómetro, inteligencia artificial, red neuronal convolucional, máquinas de soporte vectorial.

Abstract

Spectrophotometers are instruments that measure parameters of samples as a function of wavelength and are composed of a source of broad spectrum light, a diffractive element and a detector. Those instruments are widely used in chemistry, physics materials and medicine labs among others. The design of a spectrophotometer in the visible range of the electromagnetic spectrum is presented is this thesis and is composed of a white LED, a holographic grating and the Samsung camera as a detector; the spectrum generated by placing a liquid sample in the spectrophotometer is analyzed by three artificial intelligence algorithms that are artificial neural networks, convolutional neural networks and support vector machines. These type of algorithms take part of machine learning which is currently used to solve classification and regression problems, for example facial recognition, speech recognition, search engine and medical diagnostic, amog others. In this thesis, these algorithms were implemented to determinate which one is the best to classificate the samples, considering account the accuracy and run time.

Keywords: Spectrophotometer, artificial intelligence, convolutional neural network, sopport vector machine.

Índice

Capítulo 1

Introducción

La absorción de la radiación electromagnética, como la luz, es el proceso por el cual la radiación es absorbida por la materia. Cuando la absorción se produce dentro del rango de la luz visible, recibe el nombre de absorción óptica. Esta radiación, al ser absorbida, puede ser reemitida o bien transformarse en otro tipo de energía, como calor o energía eléctrica.

En general, todos los materiales absorben en algún rango de frecuencias. Aquellos que absorben en todo el rango de la luz visible son llamados materiales opacos, mientras que si dejan pasar dicho rango de frecuencias se les llama transparentes. Es precisamente este proceso de absorción y reemisión de la luz visible lo que da color a la materia. De la cualidad de los materiales de absorber o emitir la luz que incide sobre ellos nace el concepto de espectroscopía [1].

La espectroscopía es el estudio de la interacción entre la radiación electromagnética y la materia, con absorción o emisión de energía radiante [2]. Tiene aplicaciones en astronomía, física, química y biología, entre otras disciplinas. El análisis espectral se basa en detectar la absorción o emisión de radiación electromagnética a ciertas longitudes de onda, estos procesos se rela-

cionan con los niveles de energía implicados en una transición cuántica.

La espectroscopía ultravioleta-visible es una espectroscopía de emisión de fotones y una espectrofotometría. Utiliza radiación electromagnética (luz) de las regiones visible, ultravioleta cercana (UV) e infrarroja cercana (IR) del espectro electromagnético, es decir, una longitud de onda entre 380 nm y 800 nm. La radiación absorbida por las moléculas desde esta región del espectro provoca transiciones electrónicas que pueden ser cuantificadas. La espectroscopía UV-visible se utiliza para identificar algunos grupos funcionales de moléculas, y además, para determinar el contenido y concentración de una sustancia sí esta tiene su espectro en el visible. Se utiliza de manera general en la determinación cuantitativa de los componentes de soluciones de iones de metales de transición y compuestos orgánicos[1]. La manera de medir algún componente en una solución es por medio de un espectrofotómetro.

El espectrofotómetro es un instrumento que permite comparar la radiación absorbida o transmitida por una solución que contiene una cantidad desconocida de soluto, y una que contiene una cantidad conocida de la misma sustancia. Esta espectrofotometría utiliza radiaciones del campo UV de 80 a 400 nm, principalmente de 200 a 400 nm (UV cercano) y de luz visible de 400 a 800 nm, por lo que es de gran utilidad para caracterizar las soluciones en la región ultravioleta-visible del espectro. Se rige por una ley muy importante la cual se resume en la ecuación de Beer-Lambert [1].

Otro elemento importante en los espectrofotómetros es la rejilla de difracción que es la herramienta preferida para la separación de los colores de la luz incidente en espectroscopía. Las variedades de las rejillas de difracción son rejillas de difracción echelle, rejillas regladas, rejillas holográficas, entre otras. La

rejilla holográfica es una elección ideal para experimentos de espectroscopía, ya que exhiben una difracción significativamente más clara que la mayoría de otras rejillas al reducir la cantidad de luz extraviada. Una rejilla de difracción holográfica ayudará en la obtención del espectro de la sustancia en cuestión. Una manera eficaz de encontrar la relación entre los espectros y su concentración puede ser utilizando aprendizaje automático.

El aprendizaje automático es una rama de la inteligencia artificial que se concentra en desarrollar técnicas que permitan a las computadoras "aprender" [3], este proceso de aprendizaje se da mediante programas capaces de generalizar comportamientos a partir de una información suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento. Existen distintos algoritmos en el aprendizaje automático como lo son árboles de decisiones, algorítmos genéticos, redes neuronales artificiales, convolucionales, máquinas de soporte vectorial, entre otros.

La computación convencional se caracteriza por el desarrollo de una formulación matemática del problema, el desarrollo de un algoritmo para implementar una solución, la codificación del mismo para un problema específico y por último la ejecución de dicho código. Como se ha observado, este tipo de procesamiento es muy exitoso para resolver modelos matemáticos complejos y de simulación, para realizar tareas repetitivas, rápidas y bien definidas. Por otro lado, la computación basada en el aprendizaje de máquina se caracteriza por ser masivamente paralela, adaptativa, altamente interconectada y tolerante al ruido. En publicaciones recientes, las redes neuronales y las máquinas de soporte vectorial han tenido aplicaciones en el área de procesamiento de imágenes y visión computacional [4], específicamente en el análisis de reconocimiento de patrones. Es por ello que en esta tesis se implementarán redes neuronales (convolucionales y

artificiales) y máquinas de soporte vectorial con el objetivo de comparar y determinar cual de estos será el mejor algorítmo capaz de resolver nuestro problema de clasificación bajo criterios de eficiencia y tiempo de cómputo.

1.1 Objetivos

1.1.1 Objetivo general

Implementar las redes neuronales artificiales, convolucionales y máquinas de soporte vectorial para clasificar el espectro de una muestra líquida con colorantes vegetales azul, verde y rojo obtenido con un espectrofotómetro.

1.1.2 Objetivos específicos

1. Con base en los principios de espectroscopía y los componentes de un espectrofotómetro UV-Vis, diseñar un arreglo óptico para tomar los espectros de las muestras a analizar.
2. Comprender el desarrollo matemático para el aprendizaje automático usando redes neuronales artificiales, redes neuronales convolucionales y máquinas de soporte vectorial.
3. Implementar las redes neuronales artificiales necesarias para la clasificación de las muestras según su color y concentración.
4. Implementar las redes neuronales convolucionales necesarias para la clasificación de las muestras según su color y concentración.
5. Implementar las máquinas de soporte vectorial necesarias para la clasificación de las muestras según su color y concentración.

6. Diseñar y construir un espectrofotómetro bajo los parámetros determinados por el experimento

1.2 Descripción de capítulos

En el capítulo 2 se habla de la espectroscopía y sus consideraciones a tener en cuenta al momento de analizar los datos, también se describe el funcionamiento y componentes de un espectrofotómetro. En el capítulo 3 se dan los fundamentos requeridos para la construcción de una red neuronal artificial, red neuronal convolucional y máquina de soporte vectorial, así como el procedimiento a seguir para la implementación del código y se desarrolla una comparación entre ellos al predecir el conjunto MNIST[1]. En el capítulo 4 se explica a detalle la construcción y las componentes del espectrofotómetro diseñado. En el capítulo 5 se muestran los resultados de las clasificaciones mediante los algoritmos descritos previamente. Finalmente, en el capítulo 6 se dan las conclusiones principales de la tesis.

[1]El conjunto de datos mixtos del Instituto Nacional de estándares y tecnología (MNIST) es una coleccin de 70.000 pequeñas imágenes de dígitos escritos a mano. Los datos fueron creados para actuar como un referente para los algoritmos de reconocimiento de imagen.

Capítulo 2

Espectroscopía

La espectroscopía es el estudio del espectro electromagnético, la composición de la luz en longitudes de onda, debido a las interacciones atómicas y/o moleculares con la luz. Por muchos años, la espectroscopía ha sido de importancia en el estudio de la física, y es ahora de igual importancia en astronomía, biología, química y otras áreas de investigación analítica.

La espectroscopía describe la interacción entre la radiación electromagnética y la materia, tal interacción se lleva a cabo con la cuanta de la radiación electromagnética y estos siempre están caracterizados por una longitud de onda λ, una frecuencia ν o una energía E. La relación entre ellas viene dada por la ecuación de Planck

$$E = h\nu = hc/\lambda. \tag{2.1}$$

En la Fig. 2.1 se da un esquema de los distintos valores de frecuencia y longitud de onda y la región del espectro que les corresponde. La radiación electromagnética ocasiona distintos efectos en la materia, tal como se muestra en la tabla 2.1.

Cuando la radiación incide sobre una sustancia, sólo un tipo de átomos son capaces de absorber la radiación; estos grupos se

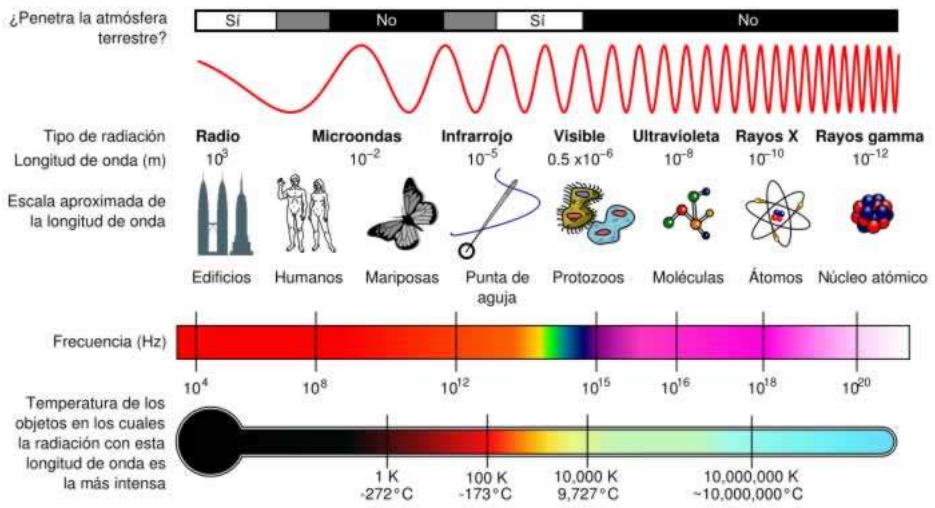


Fig. 2.1: Relación de los valores de frecuencia y longitud de onda de las distintas regiones del espectro de la luz. Imagen tomada de [1].

denominan *cromóforos* y serán distintos dentro de una misma molécula para cada técnica espectroscópica.

Los efectos de la radiación sobre la materia pueden usarse para obtener información sobre la estructura de la misma, así surgen distintas técnicas espectroscópicas tal como se muestra en la tabla 2.2.

Tabla 2.1: Efecto de la radiación electromagnética en la materia.

Radiación	**Efecto**
Rayos X y cósmicos	Ionizaciones de las moléculas
UV - Vis	Transiciones electrónicas entre los orbitales atómicos y moleculares
Infrarrojo	Deformación en los enlaces químicos
Microondas	Rotaciones de los enlaces químicos
Radiofrecuencia	Transiciones de espín electrónico o nuclear en los átomos de la molécula

Tabla 2.2: Técnicas espectroscópicas e información que se puede obtener de ellas.

Técnica espectroscópica	**Información obtenida**
Rayos X	Estructura total de la molécula incluida la estereoquímica
UV - Vis	Existencia de cromóforos y/o conjugación en la molécula
Infrarrojo	Grupos funcionales a partir de las absorciones observadas
Espectroscopía de masas	Fórmula molecular y subestructuras a partir de los iones observados
Resonancia magnética nuclear	Grupos funcionales, subestructuras, conectividades, estereoquímica, etc.

2.1 Espectometría ultravioleta visible (UV-Vis)

La espectroscopía UV-Vis utiliza la radiación del espectro electromagnético, cuya longitud de onda está entre los 100 y los 800nm y su efecto sobre la materia es producir transiciones electrónicas entre los orbitales atómicos y/o moleculares de la sustancia. En algunos casos, los efectos y la detección pueden llegar al IR cercano (800-900nm). En la espectroscopía UV-Vis una especie química (en general una molécula, aunque puede tratarse de una especie monoatómica, un ión o un complejo) absorbe UV-Vis, y la enegía adquirida por el sistema causa la transición de un electrón de un estado basal o fundamental (EF) a uno excitado (EE). La energía de transición está relacionada con la longitud de onda de radiación a través de la ecuación de Planck (Ec. 2.1).

Un gráfico de la respuesta del sistema en función de la longitud de onda o frecuencia se denomina espectro. En general, en los espectros UV-Vis, se observa una señal debida a cada transición

electrónica del EF al EE. Los átomos dan líneas muy delgadas, mientras que las móleculas poliatómicas dan señales en forma de bandas puesto que la absorción de luz involucra también energía suficiente para causar cambios en energía vibracional y rotacional de cada uno de sus estados electrónicos en el EE. Para una sustancia determinada, la longitud de onda a la cual se produce el máximo de absorbancia en el espectro se denotará como λ_{max}.

La señal espectral permite, por un lado, identificar algunos grupos funcionales presentes en las moléculas y, por el otro lado, estimar la concentración de una sustancia. La espectrometría es la técnica espectroscópica usada para evaluar la concentración de una especie y utiliza un instrumentro llamado espectrómetro. En el caso de la espectrometría que utiliza fotones (UV-Vis, IR), se suele hablar de espectrofotometría. Para la medición de la intensidad de absorción se usan espectrofotómetros en los cuales se puede medir la absorbancia o la transmitancia.

2.2 El espectofotómetro UV-Vis

Para las determinaciones analíticas es solamente necesario disponer de un espectrofotómetro UV-Vis y con este equipo medir el cambio de la absorbancia de un compuesto a diferentes concentraciones de un analito[1] por un método estandarizado para la especie en cuestión. Generalmente, se realiza una curva de calibración de la cual se puede obtener el coeficiente de absorción molar.

La espectrometría UV-Vis se emplea generalmente en la determinación cuantitativa de la concentración en solución de es-

[1]En química analítica, un analito es un componente (elemento, compuesto o ión) de interés analítico de una muestra. Es una especie química cuya presencia o contenido se desea conocer.

pecies químicas como iones metálicos de transición y algunos compuestos orgánicos. Muchas de las determinaciones incluyen un paso de reacción entre la especie y un compuesto que origine un derivado coloreado o absorbente en el UV-Vis.

Por lo general, se usa el agua como disolvente para compuestos inorgánicos y etanol para compuestos orgánicos porque este alcohol absorbe muy débilmente a la mayoría de las longitudes de onda.

El instrumento usado en la espectrofotometría ultravioleta-visible se denomina espectrofotómetro UV-Vis y permite comparar la radiación absorbida o transmitida por una solución que contiene una cantidad desconocida de soluto con una que contiene una cantidad de la misma sustancia. Se mide la transmitancia de la muestra que se expresa habitualmente como porcentaje ($\%T$) o bien la absorbancia (A).

2.2.1 Partes básicas de un espectrofotómetro

El espectrofotómetro se compone generalmente de una fuente de luz, por lo general una lámpara incandescente (de tungsteno) para las longitudes de onda en el rango visible, o una lámpara de arco de deuterio en el ultravioleta, sin embargo, las lámparas de mercurio-xenón también son empleadas, un soporte para la muestra, una rejilla de difracción o monocromador para separar las diferentes longitudes de onda de la luz y un detector. El detector suele ser un fotodiodo o un CCD (*Charge-coupled device*). Los fotodiodos se usan con monocromadores que filtran la luz, de modo que una sola longitud de onda alcanza el detector. Las rejillas de difracción se utilizan en conjunto con CCDs, que recogen radiación electromagnética de diferente longitudes de onda en pixeles.

Un espectrofotómetro puede ser de haz simple o de doble haz. En un instrumento de un sólo haz, toda la luz pasa a través de la celda de la muestra. La intensidad incidente I_a se mide análogamente en ausencia de la muestra. En un instrumento de doble haz, la luz se divide en dos haces antes de llegar a la muestra. Algunos instrumentos de doble haz tienen dos detectores y se puede medir simultáneamente tanto el haz de referencia como el de la muestra. En otros instrumentos, los dos haces pasan a través de un bloqueador que impide el paso de uno de los haces. El detector alterna entre la medida del haz de muestra y la del haz de referencia. Las muestras se colocan en una celda transparente, que suele ser rectangular con un ancho de 1 cm. Las mejores celdas están hechas con cuarzo de alta calidad, aunque son comunes las de vidrio o plástico. El cristal, el vidrio y la mayoría de los plásticos absorben en el UV, lo que limita su utilidad a las longitudes de onda en el visible.

Un espectrofotómetro convencional enfoca la luz policromática de la fuente en un monocromador. Éste tiene como componentes principales una ranura de entrada, un elemento que dispersa la luz en su composición por longitudes de onda y una ranura de salida que permite seleccionar la longitud de onda deseada. Esa luz monocromática atraviesa la muestra y llega al detector (Fig. 2.2). Las mediciones fotométricas se hacen en base a la relación entre la intensidad de la luz que alcanza al detector cuando está interpuesta la muestra y cuando no lo está.

En realidad, el monocromador no selecciona una única longitud de onda, sino un rango, cuya amplitud depende de la calidad del mismo. Esta resolución depende fundamentalmente del diseño del monocromador, de su distancia focal y de las dimensiones y densidad de líneas en la rejilla de difracción.

Para cambiar la longitud de onda de medición, o para hacer un barrido espectral, se mueve el elemento dispersor o algún espejo

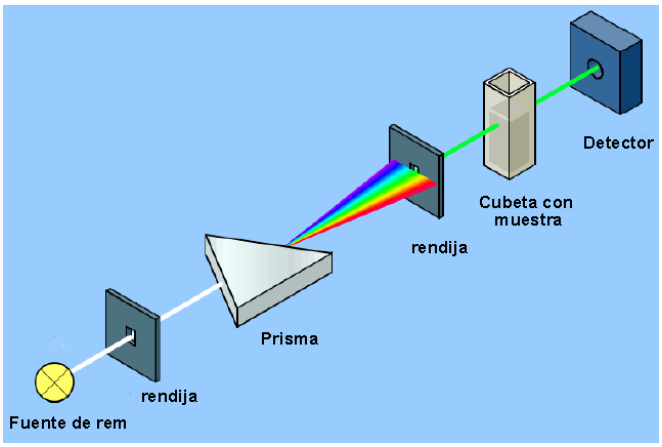


Fig. 2.2: Esquema de un espectrofotómetro UV-Vis.

por medio de un motor por pasos. En la siguiente sección, se describirá el funcionamiento de una rejilla de difracción que será utilizado en esta tesis como elemento difractivo.

2.3 Difracción y rejillas holográficas

Un cuerpo opaco colocado a medio camino entre una pantalla y una fuente puntual proyecta una sombra complicada hecha de regiones claras y oscuras, muy diferentes de las que podría esperarse de los principios de la óptica geométrica. El trabajo de Francesco Grimaldi en el siglo XVII fue el primer estudio detallado que se publicó sobre esta desviación de la luz de su propagación rectilínea, a la que denominó *diffractio*. El efecto es una característica general de los fenómenos ondulatorios que ocurren donde quiera que una parte del frente de onda esté obstruida de alguna manera. Si al encontrar un obstáculo transparente u opaco se altera la amplitud o la fase de una región del frente de onda, esto producirá difracción. Los varios segmentos del frente de onda que se propagan más allá del obstáculo interfieren, produciendo aquella distribución de densidad de energía

particular denominada patrón de difracción. No hay distinción física significativa entre interferencia y difracción. Sin embargo, se ha vuelto algo común, aunque no siempre apropiado, hablar de interferencia cuando se analiza la superposición de solamente unas pocas ondas y de difracción cuando se trata de un gran número de ondas.

Es posible producir una teoría de difracción basada en el principio de Huygens de la emisión secundaria y por este medio explicar casi la mayoría de los fenómenos de difracción observados. Antes de proceder por este camino se deben considerar algunas suposiones implícitas en este principio. Empezando por considerar exactamente como se debe hacer el cálculo de la difracción. Una respuesta se puede obtener escribiendo las ecuaciones de Maxwell y resolviéndose, considerandose las condiciones de frontera que describen al obstáculo difractivo. Por ejemplo, una onda plana incidiendo sobre un conductor perfecto esférico impondría la condición de que sobre la superficie el campo eléctrico debería ser cero. Aunque este cálculo puede ser realizado, no ayuda a la comprensión general del fenómeno de difracción. Una descripción más adecuada se obtiene de aproximar la luz como una variable escalar y que los efectos obtenidos de la polarización de las ondas pueden ser despreciados.

En principio los cálculos de una onda escalar deben ser realizados para cada componente del vector de onda, pero esto es rara vez necesario. Se pueden ver el tipo de condiciones bajo las cuales la dirección de polarización debería ser importante estudiando el problema de difracción por una hendidura en un conductor perfecto en forma de hoja. Considerando cada punto en la hendidura como un radiador potencial, se sigue que:

- Puntos en el metal no radiarán ya que el campo en un conductor perfecto es cero.

- Puntos dentro de la rendija radiarán igualmente bien en todas las polarizaciones ya que el campo puede estar en cualquier dirección en el espacio libre.
- Puntos cerca de los bordes de la hendidura radiarán mejor cuando el campo eléctrico sea paralelo. Esto ocurre porque el campo eléctrico paralelo cambia continuamente de cero, cuando está en el metal, a no cero, cuando se encuentra dentro de la hendidura, mientras el campo eléctrico perpendicular es no continuo a través de la superficie.

La diferencia de efectividad del tamaño de hendidura estará en el orden de la longitud de onda; normalmente las diferencias de este orden son raramente importantes cuando se está tratando con un obstáculo grande y así se puede utilizar la aproximación de onda escalar con un buen grado de confianza.

2.3.1 Difracción de Fraunhofer y Fresnel

Considérese una amplitud observada en un punto P que viene de la emisión de luz de una fuente puntual en Q y dispersada por un obstáculo plano R, Fig. 2.3. Supóngase que si un elemento de área dA en S dentro del plano es atravezado por una onda Ψ_1 éste mismo punto funcionará como una fuente emisora secundaria de longitud $f_s\Psi_1 dA$, donde f_s es llamada función de transmisión de R en el punto S. Un ejemplo simple de tratar consiste en tomar la función de transmisión como cero si el obstáculo es opaco o igual a la unidad en caso que el obstáculo sea transparente. La coherencia de la "re-emisión" es importante; la fase de la onda emitida debe ser igualmente relacionada a la onda incidente inicial, Ψ_1, de otra manera los efectos de la difracción cambiarán con el tiempo.

La onda emitida de la fuente puntual Q de longitud a_1 puede

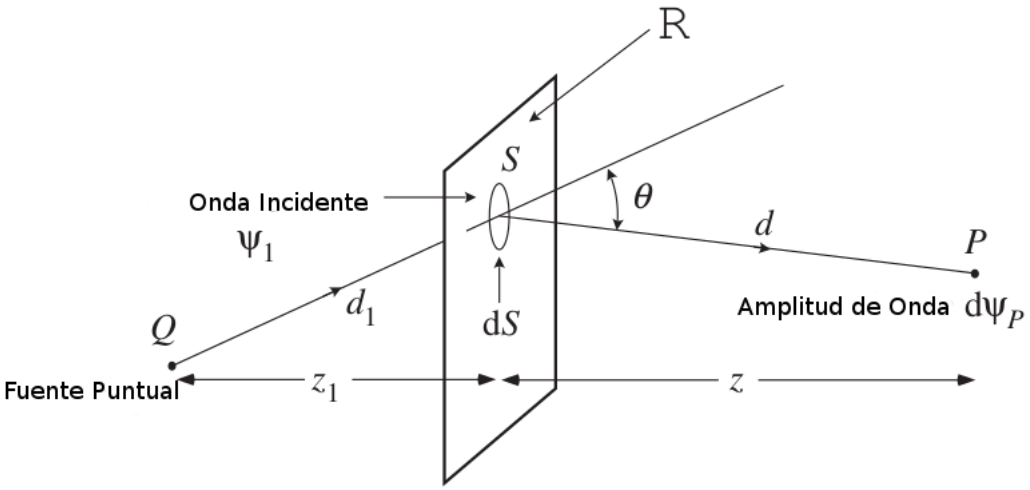


Fig. 2.3: Ilustración de la difracción de una fuente puntual por un obstáculo plano.

ser escrita como una onda esférica[5],

$$\Psi_1 = \frac{a_1}{d_1}\exp(ikd_1), \tag{2.2}$$

consecuentemente S actúa como un emisor secundario de longitud a,

$$a = bf_s\Psi_1 dS \tag{2.3}$$

donde b es la fuerza con la cual ocurre la reradiación del emisor secundario por unidad de superficie. Así que la contribución de Ψ que se recibe en P es

$$d\Psi_P = f_s\Psi_1 dS \cdot \frac{1}{d}\exp(ikd)$$
$$= f_s b a_1 \cdot \frac{1}{d\cdot d_1}\exp\{ik(d+d_1)\}dS. \tag{2.4}$$

La amplitud recibida en P es entonces la integral sobre $d\Psi_P$ sobre todo el plano que obstruye el camino de la luz

$$\Psi_P = \int\int_R a_1 b f_s \cdot \frac{1}{d\cdot d_1}\exp\{ik(d+d_1)\}dS \tag{2.5}$$

Se debe tener en cuenta que f_s,d y d_1 son funciones de la posición S. Los cálculos de difracción involucran integrales en la forma de la ecuación 2.5 bajo varias condiciones. Se considerarán condiciones las cuales simplifican los cálculos para ayudar a entender los principios de la difracción claramente. Primero supóngase un sistema iluminado por una onda plana. En la práctica esto se puede lograr si se supone la fuente puntual en el infinito y suficientemente brillante; esto hace que tanto d_1 como a_1 vayan al infinito siempre y cuando mantengan la razón constante: $\frac{a_1}{d_1} = A$.

Considérese ahora que el obstáculo plano R coincide con el plano de incidencia de la onda plana incidente. El eje del sistema es definido como el vector normal a R que atravieza su origen O. Si denotamos la posición de S por el vector $\vec{r}$ sobre el plano R, cambiando f_s por $f(\vec{r})$ entonces la ecuación 2.5 se reescribe como

$$\Psi = Ab \cdot \exp(ikz_1) \int\int_R \frac{f(\vec{r})}{d} exp(ikd) d^2\vec{r}. \tag{2.6}$$

El factor z_1 es la distancia normal entre Q y R. La intensidad observada en P es

$$I = |\Psi|^2 = \Psi\Psi^*. \tag{2.7}$$

Los efectos de la difracción pueden ser convenientemente clasificados en dos tipos: difracción de campo cercano (Fresnel) o difracción de campo lejano (Fraunhofer), esta clasificación depende de cómo cambia la fase kd a medida que se cruza el obstáculo. Esto depende de los valores relativos de la distancia entre S y el punto de observación, la extensión de R para los cuales $f(\vec{r})$ es distinto de cero (es decir, el tamaño de la región de transmitancia de la pantalla) y de la longitud de onda $\lambda = 2\pi/k$. Si kd es tal que varía linealmente con $|\vec{r}|$ la difracción es llamada de Fraunhofer; si la variación tiene términos no lineales mayores que $\pi/2$, la difracción es llamada de Fresnel. Traduciendo estos argumentos en cantidades cuantitativas, se define un círculo de

radio ρ el cual incluye exactamente todas las regiones de transmisión de R (Fig. 2.4). Posteriormente se observa la difracción en una pantalla P cuya normal es paralela al eje, colocada a una distancia z del plano difractivo. Entonces en un punto P en la pantalla de observación, siendo $\vec{p}$ el vector distancia sobre la pantalla, a P respecto al eje z, la fase kd de la onda desde $\vec{r}$ es

$$kd = k(z^2+|\vec{r}-\vec{p}|^2)^{1/2} \approx kz+\frac{1}{2}kz^{-1}(r^2-2\vec{r}\cdot\vec{p}+p^2)+\cdots , \quad (2.8)$$

se asume que $|\vec{r}|$ y $|\vec{p}|$ son pequeños comparados con z. Dicha

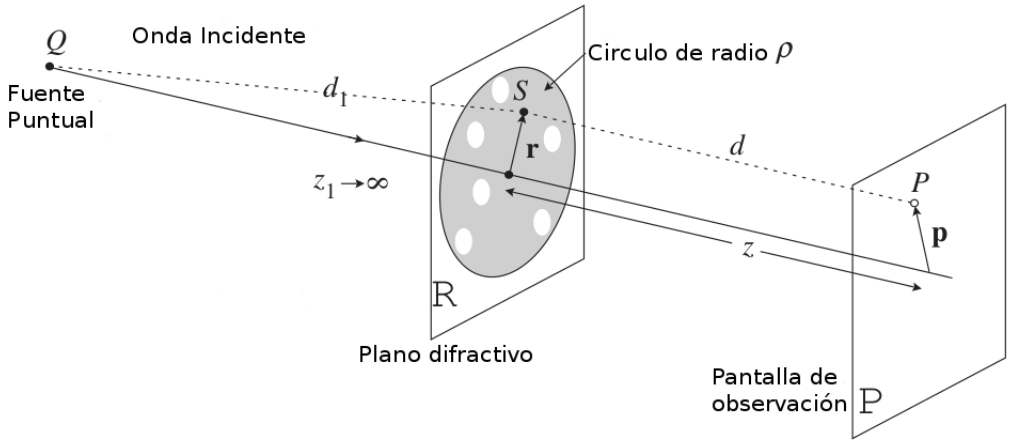


Fig. 2.4: Observación de la difracción sobre una pantalla de la luz emitida de una fuente puntual en el infinito difractada por un arreglo de elementos transparentes y opacos en un plano.

expresión contiene:

- El término constante $k(z + \frac{1}{2}p^2/z)$.
- El término $k\vec{r}\cdot\vec{p}/z$ que es lineal con $\vec{r}$.
- El término cuadrático $\frac{1}{2}kr^2/z$.

Ya que el valor máximo de $\vec{r}$ que contribuye a la ecuación es ρ el mayor valor del término cuadrático de fase es $\frac{1}{2}k\rho^2/z$. Esto

quiere decir que las condiciones de Fresnel o Fraunhofer son obtenidas dependiendo donde sea que $\frac{1}{2}k\rho^2/z$ sea mayor o menor que $\pi/2$. En términos de la longitud de onda λ esto da:

$$\text{Fresnel, o difracción de campo cercano: } \rho^2 \geq \lambda z; \quad (2.9)$$

$$\text{Fraunhofer, o difracción de campo lejano: } \rho^2 \ll \lambda z. \quad (2.10)$$

Cuando el plano es iluminado con una fuente puntual a una distancia finita z_1, la ecuación 2.8 puede ser modificada de la siguiente manera

$$\begin{aligned} k(d_1 + d) &= k[(z_1^2 + r^2)^{1/2} + (z^2 + |\vec{r} - \vec{p}|^2)^{1/2}] \\ &\approx k(z + z_1) + \frac{kr^2}{2}(z^{-1} + z_1^{-1}) + \frac{kp^2}{2z} - \frac{k\vec{r}\cdot\vec{p}}{z} + \cdots \end{aligned} \quad (2.11)$$

Se sigue una relacion equivalente para la difracción de Fraunhofer o Fresnel cambiando z por $1/(z^{-1} + z_1^{-1})$, así se obtiene

$$\text{Difracción de Fresnel: } \rho^2 \geq \frac{\lambda}{z^{-1} + z_1^{-1}}; \quad (2.12)$$

$$\text{Difracción de Fraunhofer: } \rho^2 \ll \frac{\lambda}{z^{-1} + z_1^{-1}}. \quad (2.13)$$

2.3.2 Rejillas de difracción

Una rejilla de difracción es la colección de elementos de reflexión (o de transmisión) separados por una distancia comparable con la longitud de onda de la luz en estudio. Pueden ser pensados como una colección de elementos de difracción, tales como un patrón de hendiduras (también llamadas aperturas o simplemente líneas) transparentes en una pantalla opaca, o una colección de ranuras que reflejan en un sustrato. Una rejilla de reflexión consiste en una rejilla superpuesta sobre una superficie reflectante, mientras que una rejilla de transmisión consiste en una rejilla superpuesta sobre una superficie transparente. En

cualquiera de los casos anteriores, las rejillas de difracción tienen como características fundamentales la modulación del índice de refracción, modulación del relieve o modulación de su función de transmitancia lo que ocasiona una modulación en la amplitud de la onda incidente. Tras la difracción, una onda electromagnética incidente en una rejilla tendrá su amplitud de campo eléctrico, o fase, o ambas, modificadas de una manera predecible, debido a la variación periódica en la región cerca de la superficie de la rejilla.

Ecuación de la rejilla

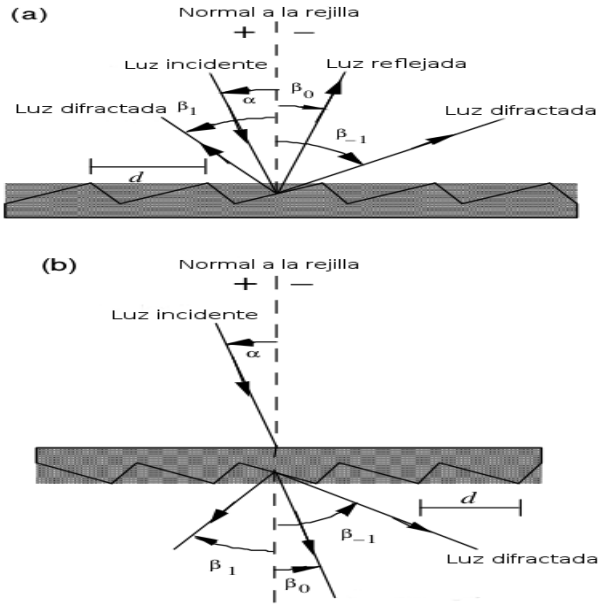


Fig. 2.5: (a) Rejilla de reflexión: la luz incidente y difractada se encuentran en el mismo lado de la rejilla. (b) Rejilla de transmisión: la luz difractada se encuentra en el lado opuesto de la rejilla respecto a la luz incidente. Imagen tomada de [6].

Cuando un haz de luz monocromático incide en la superficie de la rejilla, éste es difractado en direcciones discretas. La luz difractada por cada hendidura se combina a partir de un frente

de onda difractado. La utilidad de una rejilla depende del hecho de que existen un conjunto único de ángulos discretos a lo largo de los cuales, para una separación d dada entre las ranuras, la luz difractada de cada hendidura está en fase con la luz difractada desde cualquier otra hendidura, de modo que se combinan constructivamente.

La difracción por una rejilla puede ser vista por una geometría como en la Fig. 2.5, la cual muestra un rayo de luz con longitud de onda λ incidiendo en un ángulo α y difractado por una rejilla (por un espacio d entre hendiduras, también llamado paso) a lo largo de β_m ángulos. Estos ángulos son medidos respecto a la normal de la rejilla, ésta es indicada por la línea perpendicular a la superficie de la rejilla. La convención de signos depende de si la luz es difractada sobre el mismo lado o el lado opuesto de la superficie donde incide la luz. En la Fig. 2.5a se muestra una rejilla de reflexión, donde $\alpha, \beta_1 > 0$ y $\beta_0, \beta_{-1} < 0$. En la Fig. 2.5b se muestra una rejilla de transmisión.

Por convención, los ángulos incidentes y difractados son medidos sobre la normal de la rejilla al haz. En ambos diagramas, la convención para los ángulos es dada por la región en la cual se medirán, siendo positivo si se sitúa a la izquierda de la normal o negativo en caso contrario. Para ambas rejillas, de difracción o de transmisión, los signos de los ángulos difieren si estos son medidos sobre lados opuestos de la normal de la rejilla.

La relación entre los ángulos es dada por la ley de Bragg o también llamada ecuación de la rejilla para la difración por rejillas

$$m\lambda = d(\sin\alpha + \sin\beta), \tag{2.14}$$

la cual gobierna los ángulos de difracción para una rejilla de separación entre hendiduras d. Aquí m es el orden de difracción, el cual es un entero. Para una onda de longitud de onda λ,

todos los valores de m obedecen la relación $\|m\lambda/d\| < 2$ que corresponde a los órdenes de difracción físicamente posibles.

En muchas ocasiones se hace mención a $G = 1/d$ como la frecuencia de hendiduras o densidad de hendiduras, más comúnmente *líneas por milímetro*. Por lo tanto, se puede reescribir la ecuación 2.14 como

$$Gm\lambda = \sin\alpha + \sin\beta. \tag{2.15}$$

Un caso especial pero común es en el cual la luz es difractada en la dirección en la cual proviene (i.e., $\alpha = \beta$.), este caso es llamado configuración Littrow, por lo cual la Ec. 2.14 se escribe como

$$m\lambda = 2d\sin\alpha. \tag{2.16}$$

Órdenes de difracción

Generalmente varios enteros m satisfacen la ecuación de la rejilla y se nombra a cada entero como un órden de difracción.

Para una ranura particular de espaciamiento d, de longitud de onda λ y de ángulo incidente α, la ecuación de la rejilla (Ec. 2.14) es generalmente satisfecha por más de un ángulo de difracción β. De hecho, sujeto a las restricciones mencionadas a continuación, habrá varios ángulos discretos en los cuales se cumple la condición de interferencia constructiva. El significado físico de esto es que el refuerzo constructivo de las longitudes de onda difractadas por las ranuras sucesivas sólo exige que cada rayo sea retardado (o avanzado) en fase con cada otra; por lo tanto, esta diferencia de fase debe corresponder a una distancia real (diferencia de camino) que es igual a un múltiplo entero de la longitud de onda. Por ejemplo, esto sucede cuando la diferencia de camino es una longitud de onda, en cuyo caso se habla del

primer orden de difracción positivo ($m = 1$) o el primer orden de difracción negativo ($m = -1$), dependiendo de si los rayos son avanzados o retardados a medida que pasamos de una ranura a otra ranura. Del mismo modo, el segundo orden ($m = 2$) y segundo orden negativo ($m = -2$) son aquellos para los que la diferencia de caminos entre los rayos difractados de ranuras adyacentes es igual a dos longitudes de onda.

La Ec. 2.14 revela que únicamente los órdenes espectrales para los cuales $|m/d| < 2$ pueden existir; de lo contrario, $|\sin\alpha + \sin\beta| > 2$, lo cual no tiene sentido matemáticamente. Esta restricción evita que la luz de longitud de onda λ sea difractada en más de un número finito de órdenes. La reflexión especular ($m = 0$) es siempre posible; es decir, siempre existe el orden cero (que simplemente requiere $\beta = -\alpha$). En la mayoría de los casos, la ecuación de la rejilla permite que la luz de longitud de onda λ sea difractada en ambos órdenes negativos y positivos. Explícitamente, los espectros de todos los órdenes para los que m que existe

$$-2d < m\lambda < 2d. \tag{2.17}$$

Si la luz incidente en la rejilla es luz blanca[2], cada longitud de onda se abrirá a un distinto ángulo según la Ec. 2.14, en consecuencia para cualquier orden m se verán los colores correspondientes al espectro de la luz visible. Con base en esto, se puede obtener el espectro electromagnético completo de una sustancia en el espectro visible utilizando una rejilla de difracción, en el capítulo siguiente se describirán los fundamentos para implementar algoritmos con aprendizaje de máquina con la finalidad de poder asociar los espectros electromagnéticos obtenidos con su concentración respectiva.

[2]La luz blanca es una superposición de luces de diferentes colores las cuales presentan una longitud de onda y una frecuencia específicas.

Capítulo 3

Aprendizaje de máquina

El aprendizaje de máquina (también conocido como *Machine Learning* (ML) por su nombre en inglés) trata de crear algoritmos capaces de generalizar comportamientos o patrones a partir de una información suministrada en forma de ejemplos, en otras palabras, pretende "enseñar" a una computadora lo que es natural en humanos y animales: aprender de la experiencia. Los algoritmos de ML utilizan métodos computacionales para 'aprender' directamente de bases de datos sin requerir una determinada ecuación como modelo, a este proceso de aprendizaje se le conoce como entrenamiento [3]. La adaptabilidad de los algoritmos mejora conforme la cantidad de muestras para el entrenamiento aumenta.

Existen distintos modelos que utilizan ML, en este trabajo se discutirán tres de ellos; redes neuronales artificiales, redes neuronales convolucionales y máquinas de soporte vectorial. Dichos modelos pueden ser utilizados para resolver problemas de regresión o clasificación [3], en el desarrollo de este trabajo se hablará de los métodos de ML para resolver problemas de clasificación; un problema de clasificación consiste en asociar una entrada de datos a una categoría (de entre dos o más categorías), la cual contiene elementos que comparten características similares,

estas categorías son llamadas clases.

3.1 Redes neuronales artificiales

La inspiración para las redes neuronales artificiales (RNA) se origina del estudio de los mecanismos de procesamiento de la información en los sistemas nerviosos biológicos, en particular del cerebro humano. De hecho, gran parte de la investigación actual sobre algoritmos de redes neuronales se centra en obtener una comprensión más profunda de procesamiento de la información en los sistemas biológicos.

Una red neuronal de propagación para adelante [7] puede considerarse como una función matemática no lineal que transforma un conjunto de variables de entrada en un conjunto de variables de salida. La forma exacta de la transformación se rige por un conjunto de parámetros llamados pesos cuyo valor puede determinarse sobre la base de un conjunto de ejemplos. El proceso de determinación de los valores para los parámetros a menudo se llama aprendizaje o enseñanza, y puede ser una tarea computacionalmente intensa. Sin embargo, una vez que los pesos sean fijados, los datos pueden ser procesados por la red rápidamente. Será conveniente en varios puntos de este trabajo establecer una analogía entre las redes neuronales artificiales y la técnica estándar de ajuste de curvas usando funciones que es utilizada para entrenar la RNA.

3.1.1 Redes neuronales biológicas

El cerebro humano contiene alrededor de 10^{11} células eléctricamente activas llamadas neuronas. Éstas existen en una gran variedad de formas, aunque la mayoría tienen características comunes como se indican en la Fig. 3.1. El árbol de ramificación de

las dendritas proporciona un conjunto de entradas a la neurona, mientras que el axón actúa como una salida. La comunicación entre las neuronas se lleva a cabo en las uniones llamadas sinapsis. Cada neurona típicamente hace las conexiones a muchas miles de otras neuronas, por lo que el número total de las sinapsis en el cerebro excede 10^{14}. A pesar de que cada neurona es un sistema de procesamiento de información relativamente lento (que opera en una escala de tiempo efectivo de alrededor de 1 ms) el paralelismo masivo de procesamiento de la información en muchas sinapsis conduce simultáneamente a una potencia de procesamiento eficaz, que es muy superior a la de las supercomputadoras actuales. También conduce a un alto grado de tolerancia a fallos, por ejemplo, muchas neuronas mueren cada día con poco o nulo efecto adverso en el rendimiento de nuestra vida diaria. Muchas neuronas actúan de una manera todo-

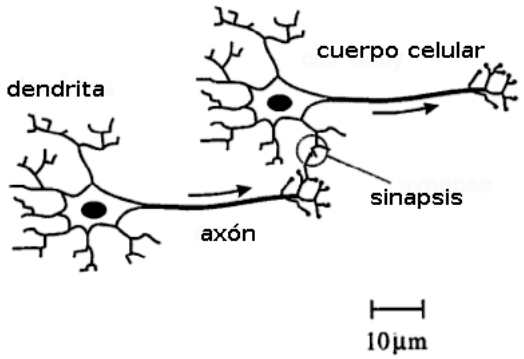


Fig. 3.1: Ilustración esquemática de dos neuronas biológicas. Imagen tomada de [8].

o-nada, esto es, una señal se envía por medio de un impulso eléctrico (llamado un potencial de acción) que se propaga desde el cuerpo celular a lo largo del axón, cuando esta señal llega a una sinapsis desencadena la liberación de neuro-transmisores químicos que atraviesan la unión sináptica a la siguiente neurona, dependiendo del tipo de sinapsis, esto puede aumentar

(sinapsis excitatoria) o disminuir (sinapsis inhibidora) la probabilidad de encender las siguientes neuronas. Cada sinapsis tiene una resistencia asociada (o peso) que determina la magnitud del efecto de un impulso en la neurona postsináptica. Cada neurona calcula de este modo una suma ponderada de las entradas de otras neuronas, y, si esta estimulación total supera cierto límite, las neuronas transmiten una señal de activación.

3.1.2 Funcionamiento de una RNA

Una red neuronal artificial (RNA) consiste en capas de neuronas altamente conectadas entre sí para asignar las entradas (X) con salidas predeterminadas, las capas intermedias de neuronas son llamadas capas ocultas. La red es entrenada por modificaciones iterativas sobre los pesos (W) que entrelazan las capas con el fin de mapear las entradas a una respuesta correcta (Y), Fig. 3.2. Un modelo matemático simple de una sola capa de neuronas

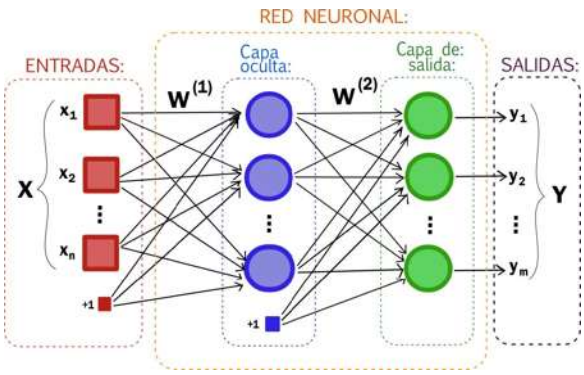


Fig. 3.2: Esquema general de la composición de una red neuronal artificial de una capa oculta.

ocultas se introdujo en un artículo por McCulloch y Pitts en 1943 [8], y adopta la forma indicada en la Fig. 3.3. En el artículo se consideró una función no lineal para transformar un conjunto de variables de entrada x_i, $(i = 1, ..., d)$ en una variable de salida z. En el modelo de McCulloch-Pitts, la señal x_i en la

entrada i se multiplica primero por un parámetro w_i conocido como un peso (que es análogo a la fuerza sináptica en una red biológica) y después se añade a todas las otras señales de entrada ponderada para dar un total de entrada a la unidad de la forma

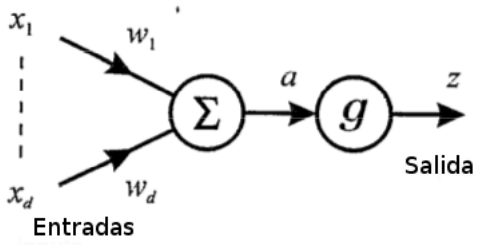


Fig. 3.3: Modelo de McCulloch-Pitts para una neurona simple con d entradas y una salida. Imagen tomada de [8].

$$a = \Sigma_{i=1}^{d} w_i x_i + w_0, \tag{3.1}$$

donde el parámetro w_0 se llama sesgo o bias. Formalmente, el sesgo puede ser considerado como un caso especial de un peso a partir de una entrada adicional cuyo valor (x_0) se establece de forma permanente igual a 1. Los sesgos desempeñan un papel importante en asegurar que la red pueda representar mapeos no lineales. Por lo tanto, podemos escribir la Ec. 3.1 en la forma

$$a = \Sigma_{i=0}^{d} w_i x_i. \tag{3.2}$$

Obsérvese que los pesos (y el sesgo) pueden ser de cualquier signo, que corresponde a sipnapsis excitatorias o inhibitorias. La salida de la unidad, z se da mediante la operación de una función de activación no lineal g:

$$z = g(a). \tag{3.3}$$

Algunas formas posibles para la función g se muestran en la Fig. 3.4. Aunque se ha introducido este modelo matemático

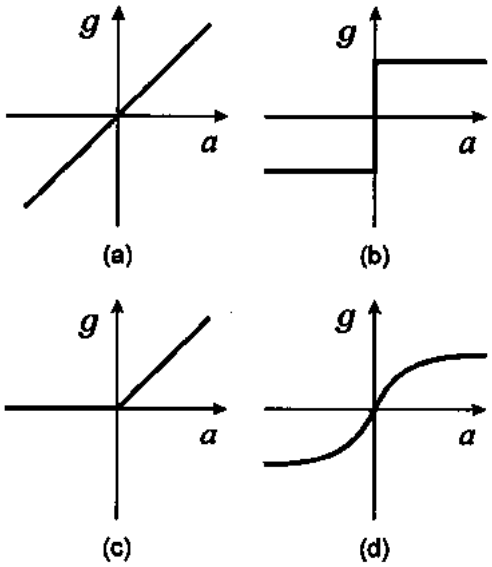


Fig. 3.4: Funciones típicas de activación: (a) Línea recta $g(a) = a$. (b) Escalón $g(a) = 1 \ \forall \ a > 0$, $g(a) = -1 \ \forall \ a < 0$. (c) ReLU $g(a) = a \ \forall \ a \geq 0$. (d) Función sigmoide $g(a) = \frac{1}{1+\exp^{-a}} - \frac{1}{2}$. Imagen tomada de [8].

de la neurona como una representación del comportamiento de neuronas biológicas, precisamente las mismas ideas también surgen cuando se consideran enfoques óptimos para la solución de problemas en el reconocimiento de patrones estadísticos. En este contexto, las expresiones tales como las ecuaciones 3.2 y 3.3 son conocidos como discriminantes lineales.

Función de error y formación de las redes

El entrenamiento para una red neuronal consiste en ajustar los valores de los pesos para obtener los resultados deseados de la misma. Esto se puede ver como una analogía en la aproximación por una curva polinomial a un grupo de puntos de n datos, etiquetados con el índice $q = 1, ..., n$. Utilizando a x^q para referirse a los valores del q-ésimo elemento del grupo de puntos y a t^q por el valor deseado u objetivo de la red.

El error entre el valor de salida deseado t^q y el correspondiente valor predicho por la función polinomial dada por $y(x^q; W)$, donde W representa las distintas combinaciones de los pesos, w_i, que intervienen en el valor de y. Una forma de calcular el error es implementando el error cuadrático utilizado para el ajuste por mínimos cuadrados en estadística, este es descrito por

$$E = \frac{1}{2}\sum_{q=1}^{n}[y(x^q : W) - t^q]^2. \tag{3.4}$$

Donde E es una función de W, por lo cual la curva puede ser ajustada a los datos eligiendo un valor de W que minimiza E. En otras palabras, se puede óptimizar la adaptabilidad del polinomio a los datos encontrando la solución de un conjunto de ecuaciones algebraicas lineales para minimizar E. La Fig. 3.5

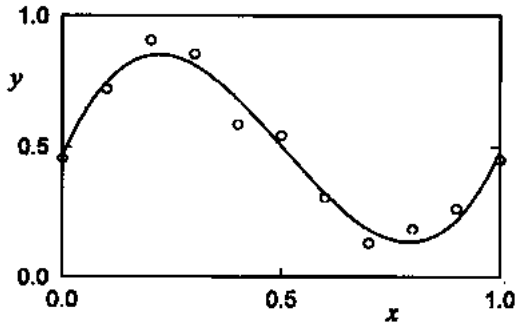


Fig. 3.5: Ejemplo del funcionamiento del ajuste cuadrático. Imagen tomada de [8].

muestra un ejemplo de un conjunto de puntos de datos junto con un polinomio cúbico que ha sido ajustado a los datos mediante la minimización del error de la suma de cuadrados. Vemos que la curva mínima de errores captura con éxito la tendencia subyacente en los datos.

La formación de una red neuronal procede de una manera análoga. Una función de error se define con respecto a un conjunto de

puntos de datos, y los parámetros (pesos) se eligen para minimizar el error. Las funciones de las redes neuronales dependen de forma no lineal en sus pesos, por lo que la reducción al mínimo de la función de error correspondiente es sustancialmente más difícil que en el caso de polinomios, y generalmente requiere el uso de algoritmos de optimización no lineales iterativos.

En el caso de una red neuronal se tienen varias muestras cada una de ellas con distintas entradas, $x^q = (x_1^q, ..., x_d^q)$, y una correspondiente salida deseada de la red, t^q. El error total se puede definir como los cuadrados de los errores sumado sobre la cantidad de muestras, k, que se tenga así como sus respectivas salidas, q, deseadas

$$E = \frac{1}{2} \sum_{q=1}^{n} \sum_{k=1}^{c} [y_k(x^q; W) - t_k^q]^2. \tag{3.5}$$

Perceptrón multicapas

Existen varios modelos de redes neuronales, uno de los modelos más sencillos y más utilizados es el Perceptrón multicapas. Esta clase de redes se han utilizado como base para la mayoría de las aplicaciones prácticas de redes neuronales hasta la fecha. En la sección anterior se describe el concepto de una sola neurona de procesamiento descrito por las Ecs. 3.2 y 3.3. Si se considera un conjunto de m neuronas, todas con entradas comunes, entonces se llega a una red neuronal que tiene una sola capa de parámetros adaptativos (pesos) como se ilustra en la Fig. 3.6. Las variables de salida se indican mediante z_j y están dadas por

$$z_j = g\left(\sum_{i=0}^{d} w_{ji} x_i\right), \tag{3.6}$$

donde w_{ji} es el peso de la entrada i a la unidad j, y g es una

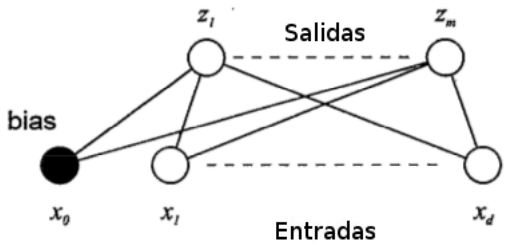


Fig. 3.6: Arreglo de la red simple donde se tienen d entradas y m salidas. Imagen tomada de [8].

función de activación como se mencionó anteriormente.

La Fig. 3.7 muestra una red con dos capas de neuronas sucesivas y por lo tanto de dos capas de pesos. La capa intermedia se conoce como capa de neuronas ocultas ya que sus valores de activación no son directamente accesibles desde fuera de la red. La activación de estas neuronas está de nuevo dada por la Ec. 3.6 como en el caso de la red de una sola capa. Las salidas de la red se obtienen actuando sobre la z de la segunda transformación, que corresponde a una segunda capa de neuronas, esto es

$$y_k = \widehat{g}\left(\sum_{j=0}^{m} \widehat{w_{kj}} x_j\right), \tag{3.7}$$

donde $\widehat{w_{kj}}$ denota un peso que conecta la capa oculta j a la capa de salida k. Nótese que se ha introducido una neurona oculta extra con $x_0 = 1$ para proporcionar el sesgo a las unidades de salida. Combinando las Ecs. 3.6 y 3.7 se obtiene la expresión completa para la transformación generada por la red

$$y_k = \widehat{g}\left(\sum_{j=0}^{m} \widehat{w_{kj}} g\left(\sum_{i=0}^{d} w_{ji} x_i\right)\right). \tag{3.8}$$

Obsérvese que la función de activación $\widehat{g}$ aplicada a las unidades de salida no tiene por que ser la misma que la función de ac-

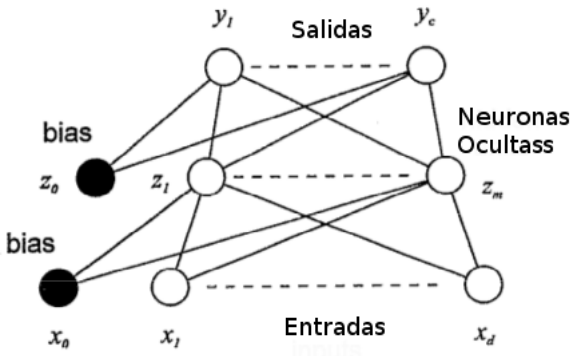


Fig. 3.7: Una red neuronal perceptrón multicapas que tiene dos capas de pesos. Imagen tomada de [8].

tivación g utilizada para las unidades ocultas. Este proceso se puede repetir m veces y el resultado sería una red neuronal de m capas ocultas.

Método de entrenamiento de propagación hacia atrás

Una red neuronal que propaga hacia adelante una activación para producir una salida y hacia atrás propaga el error para determinar los cambios que se requiere hacer a los pesos (como se muestra en la Fig. 3.8) es conocida como una red *feedforward*. El

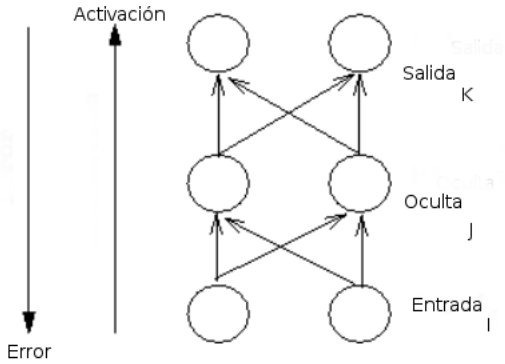


Fig. 3.8: Procedimiento en una red neuronal. Imagen tomada de [8].

algoritmo *backpropagation* (o propagación hacia atrás) se utiliza para actualizar los pesos de una red neuronal multicapa con una arquitectura fija. Se utiliza el método de descenso de gradiente para tratar de minimizar el error cuadrático entre los valores de salida de la red y los valores objetivos propuestos.

La Fig. 3.9 representa las componentes de una red que afectan a un cambio de peso en particular. Obsérvese que todos los componentes necesarios están relacionados a nivel local con el peso involucrado. Ésta es una característica de la propagación hacia atrás que parece biológicamente plausible. Sin embargo, las conexiones del cerebro parecen ser unidireccionales y bidireccionales.

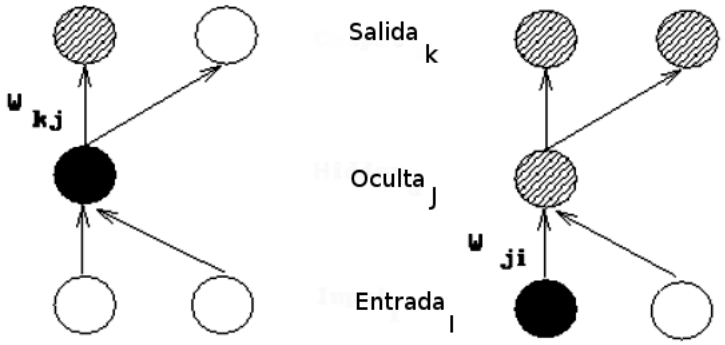


Fig. 3.9: Las neuronas cuyas salidas son propagadas por los pesos a corregir son las neuronas llenas, mientras que las neuronas con líneas representan las neuronas afectadas por las modificaciones en los pesos cuando se realiza la propagación hacia atrás. Imagen tomada de [8].

Notación

El orden de las posiciones de las capas ocultas de neuronas será decreciente, es decir, la capa más cercana a la capa de entrada será la n-ésima capa, mientras que la más cercana a la capa de salida será la primer capa de neuronas ocultas.

Se construirá un algoritmo el cual actualizará los valores de los pesos en función de la variación que tiene el error para un peso específico, para ello se derivará el error con respecto el peso en cuestión, procedimiento que se explica en la siguiente sección. Para el propósito de esta derivación, se utilizará la siguiente notación:

- El subíndice o denota los elementos de la capa de salida.
- El subíndice l denota los elementos de la l-capa oculta.
- w denota la representación de los pesos
- a denota un valor de activación.
- t denota un valor objetivo.
- net_x denota la entrada Neta, es decir, la suma del producto entre la salida de la capa x con sus respectivos pesos que la conectan con la capa anterior.

Descenso de gradiente sobre el error

Se motiva al algoritmo de aprendizaje de propagación hacia atrás como descenso de gradiente [8] sobre el error de la suma al cuadrado (elevamos al cuadrado el error, porque estamos interesados en su magnitud, no en su signo). El error total en una red está dada por la siguiente ecuación.

$$E = \frac{1}{2}\Sigma_o(t_o - a_o)^2. \tag{3.9}$$

Como se describe anteriormente, el objetivo del algoritmo es ajustar los pesos de la red para reducir el error total

$$\Delta W \propto -\frac{\partial E}{\partial W}. \tag{3.10}$$

Empezando con los pesos que conectan la primera capa de neuronas ocultas con las de salida.

$$\Delta w_{h_1 o} \propto -\frac{\partial E}{\partial w_{h_1 o}}. \tag{3.11}$$

Como el error no es directamente una función de los pesos, se expande de la siguiente manera.

$$\Delta w_{h_1 o} = -\epsilon \frac{\partial E}{\partial a_o} \frac{\partial a_o}{\partial net_{h_1}} \frac{\partial net_{h_1}}{\partial w_{h_1 o}}. \tag{3.12}$$

Donde ϵ es la constante de proporcionalidad. Derivando el error con respecto la función de activación se obtiene

$$\frac{\partial E}{\partial a_o} = \frac{\partial}{\partial a_o}\left(\frac{1}{2}(t_o - a_o)^2\right) = -(t_o - a_o). \tag{3.13}$$

Se sigue con la derivación de la entrada neta en o con respecto el peso en cuestión (nótese que únicamente un término de la suma sobre la entrada neta tiene derivada, con respecto a un peso fijo, distinta de cero).

$$\frac{\partial net_{h_1}}{\partial w_{h_1 o}} = \frac{\partial (w_{h_1 o} a_{h_1})}{\partial w_{h_1 o}} = a_{h_1}. \tag{3.14}$$

Sustituyendo las Ecs. 3.13 y 3.14 en la Ec. 3.12 se obtiene

$$\Delta w_{h_1 o} = \epsilon (t_o - a_o) a_{h_1} \frac{\partial a_o}{\partial net_1}, \tag{3.15}$$

donde se deja expresada la derivada de a_o respecto a la suma net_1 para tener libertad al momento de elegir las funciones de activación de las capas.

Ahora se determina el cambio de peso apropiado entre la primera y segunda capa oculta. Esto es más complicado, ya que al modificar un peso entre estas capas se modifica inmediatamente la salida de la neurona en la primera capa oculta a la cual estaba

ligado el peso. La regla de cambio de peso entre la segunda y primer capa oculta está dada por el desarrollo de la regla de la cadena sobre el error

$$\Delta w_{h_2 h_1} \propto -\frac{\partial E}{\partial w_{h_1 h_2}}. \tag{3.16}$$

Dada la dependencia entra las variaciones de los pesos y la salida de las neuronas de la primer capa oculta, la regla de la cadena se expande como

$$\Delta w_{h_2 h_1} \propto -\left[\Sigma_o \frac{\partial E}{\partial a_o}\frac{\partial a_o}{\partial net_{h_1}}\frac{\partial net_{h_1}}{\partial a_{h_1}}\right]\frac{\partial a_{h_1}}{\partial net_{h_2}}\frac{\partial net_{h_2}}{\partial w_{h_2 h_1}}. \tag{3.17}$$

Sabiendo que net_{h_1} es la suma sobre el producto de la salida de la primer capa oculta, a_{h_1}, y los pesos que la conectan con la salida de la red, $w_{h_1 o}$, entonces la derivada de la suma sobre la salida de la primer capa oculta resulta precisamente en los pesos $w_{h_1 o}$, análogamente la derivada de net_{h_2} sobre los pesos $w_{h_2 h_1}$ resulta en la salida de la segunda capa oculta a_{h_2}. Sustituyendo estos resultados en la Ec. 3.17 se obtiene:

$$\Delta w_{h_2 h_1} = \epsilon \left[\Sigma_o (t_o - a_o) w_{h_1 o}\frac{\partial a_o}{\partial net_{h_1}}\right] a_{h_2}\frac{\partial a_{h_1}}{\partial net_{h_2}}, \tag{3.18}$$

donde se utilizaron las derivadas que se realizaron previamente.

Desarrollando correctamente la regla de la cadena se puede obtener una relación que describe las variaciones adecuadas para los pesos entre las capas intermedias de la red neuronal. Sin pérdida de generalidad, sea una red neuronal de n capas ocultas, la relación para la correción en el peso entre la capa m y m-1 es dada por

$$\Delta w_{h_m h_{m-1}} = \epsilon \left[\sum_{h_{m-2}} \cdots \left[\sum_{h_1}\left[\sum_o (t_o - a_o) w_{h_1 o}\frac{\partial a_o}{\partial net_{h_1}}\right] w_{h_2 h_1}\frac{\partial a_{h_1}}{\partial net_{h_2}}\right] \cdots w_{h_{m-1} h_{m-2}}\frac{\partial a_{h_{m-2}}}{\partial net_{h_{m-1}}}\right] a_{h_m}\frac{\partial a_{h_{m-1}}}{\partial net_{h_m}}. \tag{3.19}$$

3.1.3 Construcción del algoritmo

En la primera parte del desarrollo de este trabajo se implementaron algunas redes neuronales de dos y tres capas ocultas en la plataforma de Pyhton utilizando para el entrenamiento el método de *Backpropagation* descrito previamente. Adicionalmente, para poder validar la eficiencia del algoritmo es necesario dividir las muestras obtenidas en dos categorías; entrenamiento y predicción. Como su nombre lo describe, la categoría de entrenamiento servirá para el aprendizaje o entrenamiento de la red, mientras que la categoria para predecir dará una medida de cómo se comporta la red al momento de clasificar muestras con las cuales no ha tenido contacto alguno.

El entrenamiento de la red consiste en un aprendizaje de un conjunto predefinido de pares de entradas-salidas dados como enseñanza, empleando un ciclo propagación-adaptación de dos fases:

1 Propagación hacia adelante:

   – Se definen los valores de los pesos de manera aleatoria entre -1 y 1

   – Se aplica un patrón de entrada como estímulo para la primera capa de las neuronas de la red, se propaga a través de todas las capas hasta generar una salida, se compara el resultado obtenido en las neuronas de salida con la salida que se desea obtener y se calcula un valor del error para cada neurona de salida.

2 Propagación hacia atrás:

   – Los errores se transmiten hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de las capas intermedias, recibiendo el porcentaje de error aproxi-

mado a la participación de la neurona intermedia en la salida original utilizando el método de entrenamiento. La actualización de los pesos se realiza mediante la relación:

$$w^n = w + \Delta w \tag{3.20}$$

– Este proceso se repite, capa por capa, hasta que todas las neuronas de la red hayan recibido un error que describa su aportación relativa al error total. Basándose en el valor del error recibido, se reajustan los pesos de conexión de cada neurona, de manera que en la siguiente vez que se presente el mismo patrón, la salida esté más cercana a la deseada (disminuya el error).

La enseñanza se repite para todos los elementos en la categoria de entrenamiento. Una vez terminado este proceso se propaga únicamente hacia adelante con los elementos de predicción y se calcula el porcentaje de aciertos. Ésto es un ciclo completo en el algoritmo de entrenamiento de la red. Por construcción de la red, el error en la propagación siempre decrece, sin embargo esta condición no se cumple necesariamente en la predicción. Dicha condición, en la predicción, se le conoce como sobreentrenamiento [7], si esto ocurriese, la configuración de pesos ya no sería adecuada al sistema por lo cual la red debe entrenarse menos.

Existen distintos parámetros que pueden modificar la configuración de una red neuronal, como lo son el número de neuronas por capa oculta, la cantidad de capas ocultas y el valor de ϵ en la actualización de los pesos y las funciones de activación. Una manera de encontrar un arreglo eficiente (esto es, el arreglo que tenga la cantidad mínima de operaciones la cual dé una buena eficiencia) es utilizar una técnica llamada validación cruzada o *Cross-validation* (CV), el cual es útil cuando el conjunto de datos

no es muy grande. La técnica consiste en fijar los parámetros en la configuración de la red y dividir los elementos de la categoría de entrenamiento en cinco particiones para su posterior entrenamiento sobre cuatro de estas particiones y predicción sobre la partición restante, dando por resultado un cierto error el cual se promediará con los errores obtenidos cuando se alterna la partición tomada de las 5 disponibles. Lo mismo se repite modificando uno o vários parametros de la red, los promedios (o medias) de estos procesos darán la pauta para determinar cuales son los parámetros óptimos para tener la mayor eficiencia con la menor cantidad de operaciones posible.

En la segunda parte de este trabajo, desarrollada en el siguiente capítulo, se implementó una red neuronal de tres capas ocultas en la plataforma de Google llamada *Tensorflow*. En dicha plataforma los algoritmos de propagación y entrenamiento estan sumamente optimizados por lo cual no es necesario implementar algoritmo alguno de entrenamiento, solo se necesita conocimientos de programación en Python y un buen entendimiento de la sintaxis requerida por *TensorFlow*. Por este motivo sumado al hecho de que se manejan como cajas negras la forma en la cual se optimizan las funciones de costo o error, no se profundizará en la forma en la cual trabaja *TensorFlow*. Basta con saber que la estructura de las redes neuronales no cambia. Para mostrar el funcionamiento de esta plataforma se utilizarán los datos del conjunto MNIST que consiste en una colección de imágenes en escala de grises de 28x28 pixeles tomadas a números que fueron hechos a mano entre el 0 y 9, son un total de 60,000 imágenes para entrenar la red y 10,000 para obtener una predicción del algoritmo sobre imágenes con las cuales no tuvo contacto alguno.

MNIST

Basandose en la descripción para la clasificación por medio de una RNA de la página oficial de *TensorFlow*, www.tensorflow.org, se implementó un algoritmo análogo en python[1]. El cual describe una red neuronal de tres capas ocultas de 500 neuronas ocultas en cada una de ellas, el método de entrenamiento utilizado se llama Optimizador de Adam el cual es una versión optimizada/mejorada del método *Backpropagation*, aunque en el contexto general descrito anteriormente de cómo funciona una RNA con el método *Backpropagation* se mantiene igual. Los errores o pérdidas en cada ciclo de la corrida sobre 10 iteraciones son mostradas en la Fig. 3.10 donde en la última iteración se observa que el porcentaje de aciertos que se obtienen sobre la categoria de predicción (o eficiencia del algoritmo) es al rededor del 95%.

3.2 Redes neuronales convolucionales

En el contexto de aprendizaje de máquina las redes neuronales convolucionales (CNN por sus siglas en inglés) pertenecen a una variación del perceptrón multicapas, sin embargo, se consideran de mayor eficiencia al momento de clasificar o segmentar imágenes, entre otras aplicaciones. Las CNNs fueron inspiradas en procesos biológicos [9], en los cuales los patrones de conectividad entre las neuronas se asemejan a la organización de la corteza visual animal.

Las CNNs combinan tres ideas para asegurar tener distintos grados de libertad: campos receptivos locales (también conoci-

[1]El algoritmo para la implementación de una red neuronal artificial utilizando *TensorFlow* para la clasificación del conjunto MNIST puede ser encontrada en el hipervínculo `http://www.ifm.umich.mx/~gvargas`, dentro de la carpeta llamada *MNIST*

H

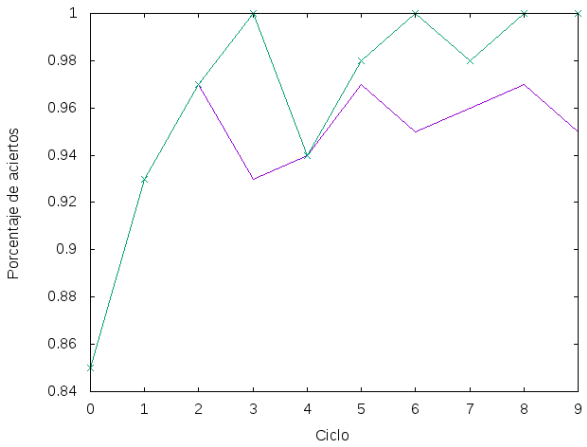


Fig. 3.10: Porcentaje de aciertos que tiene la red neuronal artificial sobre el conjunto MNIST, en la cual se aprecia que sobre el conjunto de predicción es al rededor del 95% (color morado) y para el conjunto de entrenamiento del 100% (color verde).

dos como filtros), submuestreo temporal o espacial (*pooling*) y una conexión entre pesos[10]. La primera de ellas contiene la convolución entre la imagen y distintos filtros que pretenden reconocer patrones en la imagen. La segunda es una reducción en el tamaño de las imágenes o salidas de la convolución. La tercera es similar a las conexiones de una RNA, como su nombre lo indica, todas las salidas del proceso previo, como puede ser *pooling* o convolución, se conectan enteramente a una capa de neuronas ocultas previo a la capa de salida para la clasificación de la imagen en cuestión.

3.2.1 Capa convolucional

La capa convolucional es la componente básica de una CNN y es la que lleva el mayor proceso computacional. Sus componentes son un conjunto de filtros con parámetros entrenables. Estos filtros son pequeños espacialemente (tambien referido como ancho y altura del filtro) y su profundidad depende de la imagen, por

ejemplo si la imagen a analizar fuera a color (RGB) la profundidad del filtro sería tres por los tres canales de color. Siguiendo este paso, para terminar la convolución es necesario trasladar o mover este filtro por toda la imagen, como se muestra en la Fig. 3.11, donde se procesa a través del producto punto entre los elementos del filtro y los elementos de la imagen, dando por resultado un mapa bidimensional que se conoce como mapa de activación o de características, que proporciona la respuesta de este filtro a cada posición de la imagen. Si se tiene un conjunto completo de filtros cada uno de ellos producirá un mapa de activación por separado produciendo una salida o volumen de salida si es que se tiene una imagen a color. En la analogía con el funcionamiento del cerebro sería como si cada elemento de la salida fuera el resultado de una neurona que solo ve una región y comparte características con la neurona de la izquierda y derecha (ya que este resultado se obtendrá al aplicar a la imagen el mismo filtro). Se analizará ahora la conexión entre las neuronas.

Conectividad local

Cuando se trata de entradas cuya dimensión es grande no es práctico conectar todas las neuronas entre sí. En cambio, se conectará cada neurona solo a una región local de la entrada. La extensión espacial de esta conectividad es un hiperparámetro llamado campo receptivo de la neurona (equivalentemente, este es el tamaño del filtro). La extensión de la conectividad a lo largo del eje de profundidad siempre es igual a la profundidad del volumen de entrada.

Como ejemplo, supóngase que se tiene una imagen RGB de 32x32 pixeles, Fig. 3.11, entonces se tendría un volumen de entrada 32x32x3, donde el 3 es debido a los tres canales de color.

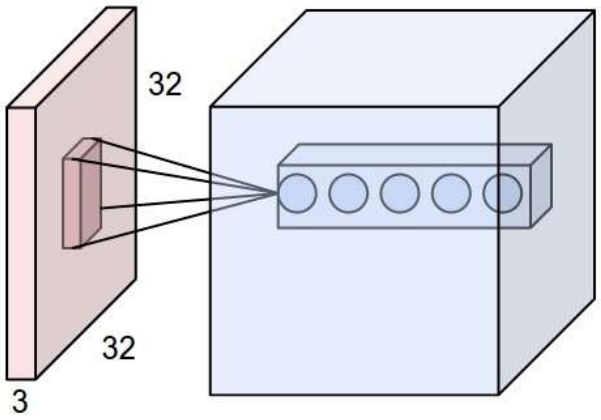


Fig. 3.11: Convolución de una imagen RGB de 32x32 pixeles por un filtro de 5x5.

Si el campo receptor, el tamaño del filtro, es de 5x5, entonces cada filtro de la capa de convolución tendrá $5 * 5 * 3 = 75$ pesos (y un parámetro de bias). Obsérvese que la extensión de la conectividad a lo largo del eje de profundidad debe ser tres, ya que es la profundidad de la entrada.

Arreglo espacial

Se ha explicado cómo es la conectividad entre las neuronas, pero no se ha dicho nada al respecto de cuantas neuronas hay en un volumen de salida o cómo están organizadas. Tres hiperparámetros controlan el tamaño del mapa de características: profundidad (depth), zancada (stride) y relleno de ceros (zero-padding):

- La profundidad del mapa de características es la cantidad de filtros que se pretende utilizar cada uno de ellos aprendiendo a buscar algo distinto en la entrada, como se muestra en la Fig. 3.12 en la cual se pretende tomar diez filtros de 5x5x3, ya que la imagen original es de tres canales, por lo tanto la profundidad del volumen de salida es de diez.

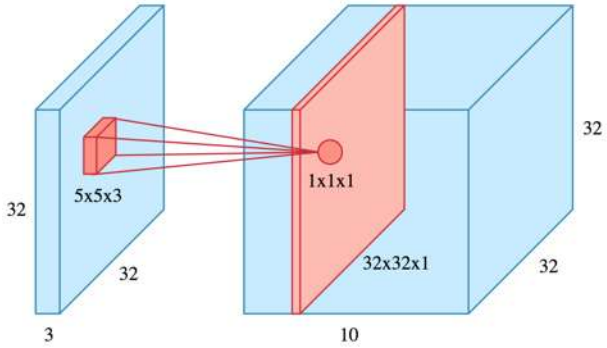
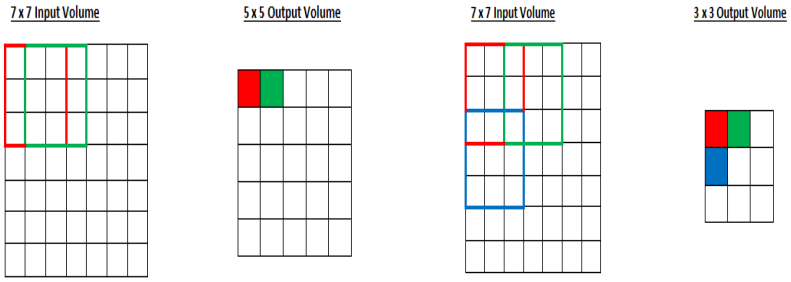


Fig. 3.12: Convolución de una imagen de 32x32x3 sobre diez filtros 5x5x3 dando por resultado un mapa de características de profundidad diez.



(a) Zancada de un pixel. (b) Zancada de dos pixeles

Fig. 3.13: Tamaño de zancada de (a) un pixel y (b) dos pixeles.

Se referirá a un conjunto de neuronas que están viendo la misma característica de la imagen como una matriz de profundidad.

- Se debe especificar la zancada con la cual se deslizará el filtro a través de la imagen. Por ejemplo, si la zancada es 1 significa que el filtro avanza un pixel a la vez como se muestra en Fig. 3.13.a, mientras que si es dos avanza de dos en dos pixeles Fig. 3.13.b, estó reduciría el tamaño del mapa de características espacialmente.

- A veces es conveniente rellenar con ceros alrededor de los bordes de las imágenes. El tamaño de este relleno de ceros

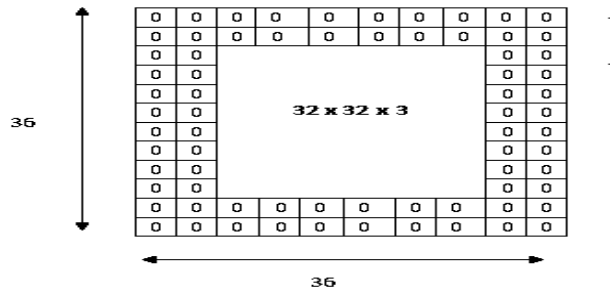


Fig. 3.14: Relleno de una imagen de 32x32x3 con dos columnas de ceros por lado, obteniendo una imagen de 36x36x3.

permitirá tener un control sobre el tamaño espacial de la salida, usualmente se utiliza para mantener igual el tamaño de la salida al tamaño de la entrada, por ejemplo, en la Fig. 3.14 se tiene una imagen original de 32x32x3 si se aumenta en tamaño de dos ceros por lado se obtiene una imagen de 36x36x3, la cual se convoluciona sobre filtros de 5x5x3 con una zancada de uno se obtendrá una salida de 32x32x3, igual que el tamaño original de la imagen.

3.2.2 Capa de *pooling*

En la estructura de una CNN es común utilizar una capa de *pooling* entre dos capas de convolución. La función de esta capa es reducir progresivamente el tamaño de la salida al reducir la cantidad de parámetros y operaciones en la red. La operación procedente de esta capa es independiente de la profundidad de la entrada ya que solo actua espacialmente. Usualmente el *pooling* consiste en tomar el valor máximo en una región del mapa de características. El filtro más común utilizado en el *pooling* es de 2x2 con *stride* de 2, esto significa que se reducirá el tamaño del arreglo a la mitad, como se muestra en la Fig. 3.15. Ahí se muestra un ejemplo de un mapa de características de 224x224 y profundidad 64 que se reduce a un mapa de características de

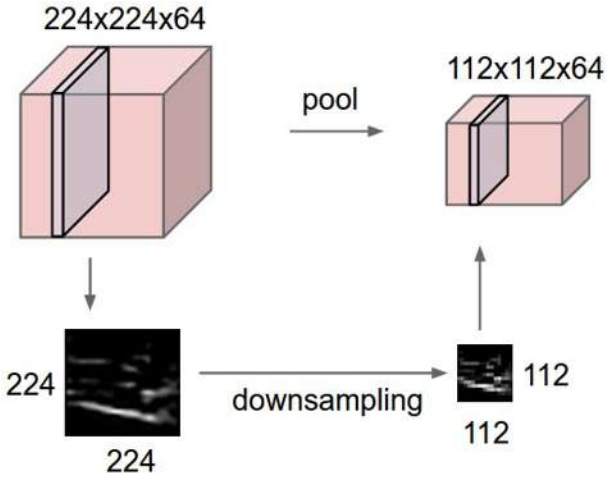


Fig. 3.15: *Pooling* de un mapa de características de 224x224x64 por medio de un *downsampling* a un mapa de características ahora de dimensión 112x112x64.

112x112x64, nótese que la profundidad no fue alterada.

3.2.3 Capa de conexión

Esta capa (también conocida como *fully-conected* por su nombre en inglés) es igual a las capas de la RNA, donde todas las neuronas tienen conexiones a todas las activaciones en la capa anterior, tal como se ve en las redes neuronales normales. Por lo tanto, sus activaciones y optimizaciones pueden calcularse como se describió en la sección 3.1. Usualmente esta capa es utilizada para llevar la información obtenida del último proceso, ya sea *pooling* o convolución, a la capa de salida mediante un proceso intermedio que consiste en una conexión total entre la salida del último proceso a una capa de neuronas ocultas.

3.2.4 Construcción del algoritmo

Al igual que la RNA, el método de entrenamiento consiste en una retroalimentación con algún metodo de entrenamiento, esto puede ser un *Backpropagation* o un *Adam Optimizer*. La optimización en el error pasa sobre las matrices de salida en el *pooling* y sobre todos los pesos de la capa *fully-conected*. Sobre el *pooling* consiste en un proceso simple, ya que solo se corregirá el peso correspondiente al valor máximo obtenido en el *pooling*. Por lo tanto, durante el paso de la propagación hacia adelante es común almacenar los índices de valor máximo para que la rutina sobre cualquier método de entrenamiento sea más eficiente. Se implementó un algoritmo únicamente en la plataforma de *TensorFlow* basado nuevamente en las observaciones dadas por su propia página web, mencionada anteriormente. Nuevamente tenemos una gran cantidad de parámetros arbitrarios en la construcción de la CNN, como lo es la cantidad de convoluciones y *poolings* por hacerse en la entrada, el tamaño de los filtros y así mismo la cantidad de filtros a convolucionar por capa de convolución. Al tener tantos parámetros libres y no un número infinito de muestras se optó por utilizar el método *Cross-Validation*, descrito previamente, para obtener una pauta del arreglo de parámetros más eficiente en esta configuración. Primero se presenta un ejemplo para observar los diferentes resultados obtenidos de la implementación de una CNN y una RNA.

Comparación entre una RNA y una CNN

Para poder tener un punto de comparación entre una RNA y una CNN, al igual que en la sección anterior se analizarán, con

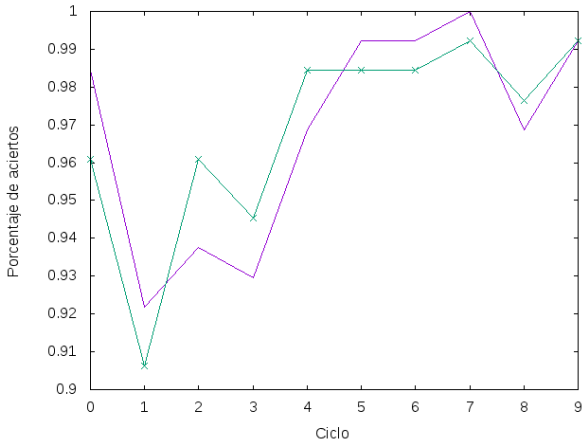


Fig. 3.16: Porcentaje de aciertos que tiene la red neuronal convolucional sobre el conjunto MNIST, en la cual se aprecia que el porcentaje de aciertos sobre la predicción es de aproximadamente 99% para el conjunto de predicción (color morado) y entrenamiento (color verde).

el código implementado[2], los datos proporcionados por MNIST, descrito anteriormente. El código consiste en una CNN con dos capas de convolución, dos *poolings* y una capa *fully-conected*. La primer convolución son treinta y dos filtros de 5x5x1, ya que es una imagen en escala de grises, la segunda convolución son sesenta y cuatro filtros del mismo tamaño, mientras que para la capa del *pooling* se redujo a la mitad el mapa de caracterísiticas y finalmente la capa de neuronas ocultas en la capa de *fully-conected* es de mil veinticuatro neuronas. Los resultados de esta CNN son mostrados en la Fig. 3.16.

Al igual que la RNA se entrenó un total de diez veces, obteniendo una eficiencia aproximada del 99% sobre el conjunto de predicción que es superior al 95% obtenido con la RNA, por lo tanto se puede inferir que para la clasificación de las imágenes del MNIST la red convolucional tiene una mejor eficiencia aunque

[2]El algoritmo para la implementación de una red neuronal convolucional utilizando *TensorFlow* para la clasificación del conjunto MNIST puede ser encontrada en el hipervínculo `http://www.ifm.umich.mx/~gvargas`, dentro de la carpeta llamada *MNIST*.

con un tiempo mayor de computo.

3.3 Máquinas de soporte vectorial

Las máquinas de soporte vectorial (SVM por sus siglas en inglés) son un conjunto de algoritmos de aprendizaje supervisado desarrollados por Vladimir Vapmik con su grupo de trabajo en AT&T. Estos métodos son utilizados, al igual que las redes neuronales, para solucionar problemas de clasificación y regresión, mediante métodos de enseñanza. Para el caso de reconocimiento de patrones las SVM's han sido utilizadas para reconocer dígitos escritos a mano [11], reconocimiento de objetos, identificación del habla, reconocimiento de caras en imágenes [12], entre otras aplicaciones.

La idea básica para la clasificación utilizando una SVM es tomar los datos de entrada y leerlos como un vector p-dimensional, la SVM busca un hiperplano que separe de forma óptima los puntos de una clase de la otra, cuando se dice de forma óptima se busca obtener el hiperplano con la máxima separación con los p-vectores que estén más cercanos al plano.

3.3.1 Reconocimiento de patrones por aprendizaje de ejemplos

Para el caso de clasificación sobre dos tipos de clase, se tratará de encontrar una función, $f : R^N \to (\pm 1)$, usando l datos para entrenar la SVM, donde N es la dimensión de las características de los conjuntos de entrada $\overrightarrow{x}_i$ y su correspondiente clasificación y_i,

$$(\overrightarrow{x}_1, y_1), (\overrightarrow{x}_2, y_2)...(\overrightarrow{x}_l, y_l) \in R^N \times (\pm 1), \tag{3.21}$$

esta función tratará de clasificar correctamente los datos de entrada, esto es mediante $f(\overrightarrow{x}) = y$, los datos pueden ser generados con una distribución de probabilidad $P(\overrightarrow{x}, y)$ que no siempre es conocida.

La teoría del aprendizaje estadístico [13] o VC (Vapnik-Chervonenkis), muestra que es crucial restringir los tipos de funciones que la máquina de aprendizaje puede implementar a las que tengan la "capacidad" adecuada para la cantidad de datos de entrenamiento disponibles.

3.3.2 Hiperplanos clasificadores

Para diseñar algorítmos de clasificación en una SVM se debe crear un tipo de funciones cuya capacidad pueda ser calculada. Los clasificadores en una SVM son basados en la clase de funciones llamadas hiperplanos,

$$D(\overrightarrow{x}) = \overrightarrow{w}^T \overrightarrow{x} + b \tag{3.22}$$

donde $b \in R$ es el bias de la función, $\overrightarrow{w} \in R^N$ son llamados pesos y el superíndice **T** indica transposición en el vector. La función encargada de determinar la clasificación de las entradas es llamada función de decisión,

$$f(\overrightarrow{x}) = sign(\overrightarrow{w} \cdot \overrightarrow{x} + b) \tag{3.23}$$

donde se define a la función *sign* como 1 para $x > 0$, 0 para $x = 0$ y -1 para $x < 0$, ya que en el procedimiento matemático para la clasificación de dos clases es conveniente definir las clases a un valor de 1 o -1 según sea su clasificación, donde el valor de 0 es correspondiente a los valores que no pueden ser clasificados [14].

Se define como hiperplano óptimo para una clasificación al cual tenga el máximo margen de separación entre dos clases. Considérese los elementos mostrados en la Fig.3.17, el cual contiene dos tipos de clases separables, verdes y rojos, clasificándose con dos hiperplanos. Ambos planos clasifican correctamente este conjunto, sin embargo hay uno que se considera óptimo

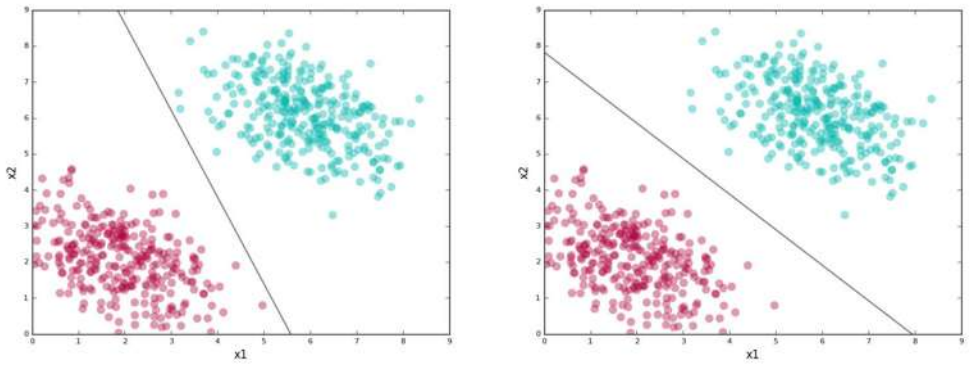


Fig. 3.17: Conjunto de datos con dos clasificaciones posibles, rojos o verdes, en el cual se realiza una clasificación con dos hiperplanos.

o mejor en comparación con el otro. Para decidir cual de ellos es el óptimo es necesario resolver un problema de optimización cuadrático cuya solución $\overrightarrow{w}$ tiene una expansión $\overrightarrow{w} = \sum_i y_i \alpha_i \overrightarrow{x}_i$ en términos de un subconjunto en los datos de entrenamiento llamados vectores de soporte, los cuales son mostrados en la Fig. 3.18, estos llevan toda la información relevante al problema de clasificación.

Existe un vector $\overrightarrow{w}$ y un bias b tal que $y_i(\overrightarrow{w} \cdot \overrightarrow{x} + b) > 0$. Reescalando $\overrightarrow{w}$ y b de tal forma que los puntos cercanos al hiperplano satisfagan

$$|\overrightarrow{w} \cdot \overrightarrow{x} + b| = 1, \tag{3.24}$$

de esta manera se obtienen los valores de $\overrightarrow{w}$ y b necesarios para formar el plano y los cuales deben satisfacer que $y_i(\overrightarrow{w} \cdot \overrightarrow{x} + b) \geq 1$ para la clasificación correcta de los datos. Una vez determinados estos parámetros la Fig.3.19 muestra el hiperplano encontrado como óptimo para esta clasificación.

Omitiendo los detalles de los cálculos necesarios, el punto crucial del algoritmo es que la solución de la optimización cuadrática

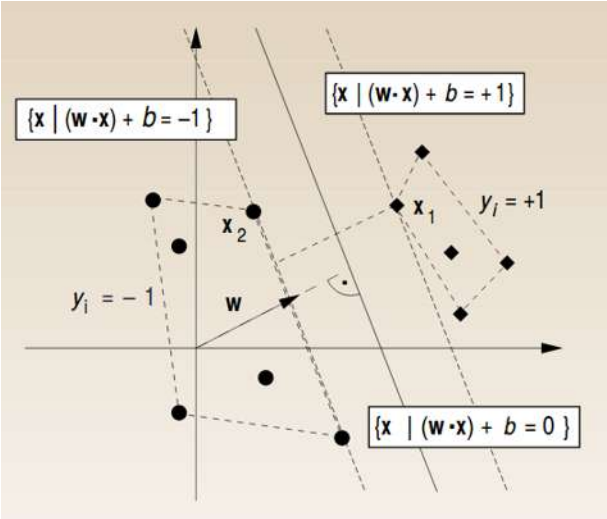


Fig. 3.18: Clasificación de un conjunto de datos separables. El hiperplano óptimo es paralelo al hiperplano formado por los elementos de la clase -1 más próximos al primer elemento de la clase +1, este hiperplano se situa en medio de la distancia que los separa ortogonalmente. Los elementos situados en las líneas punteadas corresponden a los vectores de soporte.

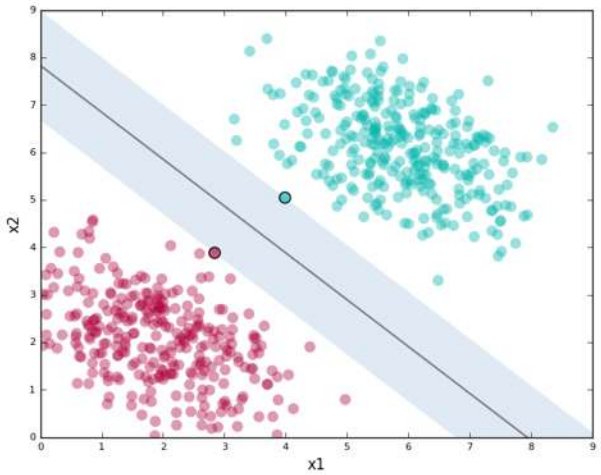


Fig. 3.19: Selección del hiperplano óptimo para la clasificación del conjunto de datos verdes y rojos, donde los colores remarcados corresponden a los vectores de soporte.

y de la función de decisión, $f(\overrightarrow{x}) = sign(\sum_i \alpha_i y_i(\overrightarrow{x} \cdot \overrightarrow{x_i}) + b)$, dependen únicamente del producto punto entre los elementos de entrada. Esto permitirá generalizar para el caso en el cual los datos no sean un conjunto separable de clases.

3.3.3 Espacio de características y kernel

La Fig.3.20 muestra un conjunto de datos no-separables, claramente si se desea clasificarlos con un hiperplano se podría obtener una eficiencia máxima del 75%. La idea básica de una SVM es mapear los datos originales a otro espacio de productos internos (llamado espacio de características) F mediante un mapeo no lineal,

$$\Phi : R^n \to F, \tag{3.25}$$

y así realizar el mismo procedimiento lineal descrito anteriormente en F. Como se mencionó, solo es requerido el resultado del producto interno

$$k(\overrightarrow{x}, \overrightarrow{z}) = \Phi(\overrightarrow{x}) \cdot \Phi(\overrightarrow{z}). \tag{3.26}$$

A este producto interno es llamado *kernel*. Si F es de mayor dimensión, el lado derecho de la Ec. 3.26 puede ser bastante complicado de calcular, sin embargo existen *kernels* bastante simples de calcular. Por ejemplo el *kernel* polinomial

$$k(\overrightarrow{x}, \overrightarrow{z}) = (\overrightarrow{x} \cdot \overrightarrow{z})^d, \tag{3.27}$$

este puede corresponden a un mapeo de Φ en un espacio de todos los productos de exactamente d dimensión en R^N. Por ejemplo, para $d = 2$ y $\overrightarrow{x}, \overrightarrow{z} \in R^2$ elementos del conjunto a mapear, mostrada en la Fig. 3.20, se tiene

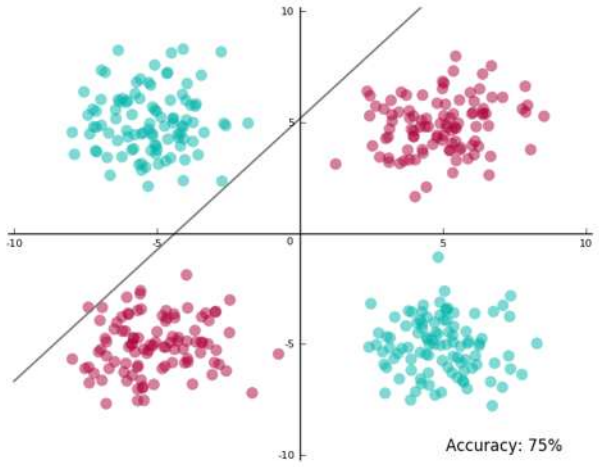
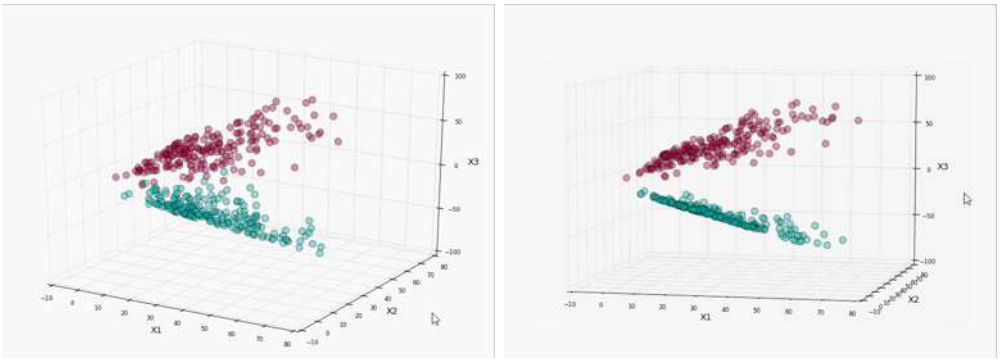


Fig. 3.20: Conjunto de datos no separables clasificados por un hiperplano.

$$
\begin{aligned}
(\overrightarrow{x} \cdot \overrightarrow{z})^2 &= \left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \cdot \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \right)^2 \\
&= (x_1 z_1 + x_2 z_2)^2 \\
&= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \\
&= (x_1^2, x_2^2, \sqrt{2} x_1 x_2) \cdot (z_1^2, z_2^2, \sqrt{2} z_1 z_2) \\
&= \Phi(\overrightarrow{x}) \cdot \Phi(\overrightarrow{z}) \qquad (3.28)
\end{aligned}
$$

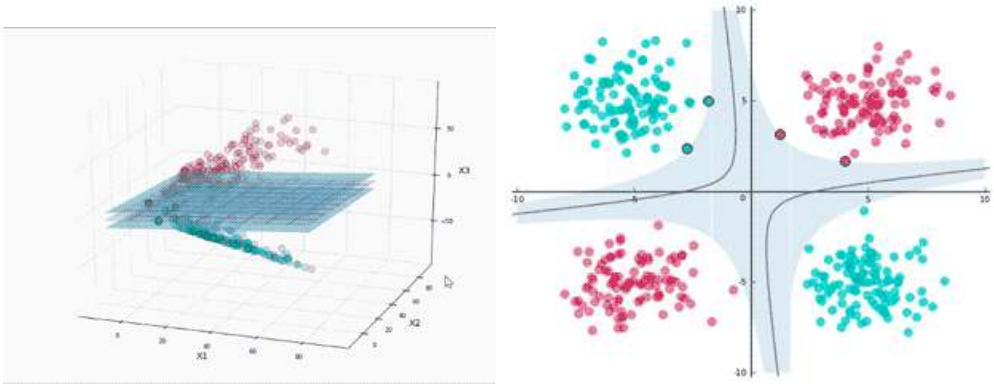
definiendo $\Phi(\overrightarrow{x}) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2)$. Aplicando este *kernel* a los datos de la Fig. 3.20 se obtienen los datos mostrados en la Fig.3.21, en las imágenes se puede observar que los datos parecen ser separables. Aplicando la SVM sobre este espacio se obtiene el hiperplano mostrado en la Fig. 3.22.a, el cual clasifica correctamente todos los elementos en este espacio. La Fig. 3.22.b muestra el hiperplano en el espacio de características mapeado en el espacio original, en el cual se aprecia que, mediante el *kernel* polinomial, la SVM puede clasificar perfectamente los datos de una relación no lineal entre las clases.

Existen otros dos tipos de kernel que son usados para la clasificación de datos como lo es el *kernel* función de base radial (RBF



(a) (b)

Fig. 3.21: Mapeo de los datos en la Fig. 3.20 al espacio de características mediante un *kernel* polinomial.



(a) (b)

Fig. 3.22: Funcionamiento de la SVM sobre el espacio de características. En la imagen (a) se obtiene un hiperplano al aplicar la SVM sobre los datos en el espacio de características de la Fig 3.21 y (b) muestra su proyección sobre el espacio original.

por sus siglas en inglés)

$$k(\overrightarrow{x}, \overrightarrow{z}) = exp(-||\overrightarrow{x} - \overrightarrow{z}||^2/(2\sigma^2)) \tag{3.29}$$

o el *kernel* sigmoidal

$$k(\overrightarrow{x}, \overrightarrow{z}) = tanh(\kappa(\overrightarrow{x} \cdot \overrightarrow{z}) + \Theta). \tag{3.30}$$

Una vez decidido cual *kernel* será utilizado para realizar el entrenamiento y obtener el hiperplano óptimo, la función de decisión en el espacio de características tendrá la forma

$$f(\overrightarrow{x}) = sign\left(\sum_{i}^{l} y_i \alpha_i \cdot k(\overrightarrow{x}, \overrightarrow{z}) + b\right). \tag{3.31}$$

Los valores α_i son calculados como solución del problema de optimización cuadrática. Este problema requiere un análisis matemático exhaustivo por lo que se mencionará la forma general de resolverlo. Para una SVM que pretende solucionar problemas reales, el problema de la optimización cuadrática es análogo a la minimización del Lagrangiano Q

$$Q(\overrightarrow{w}, b, \overrightarrow{\xi}, \alpha, \beta) = \frac{1}{2}||\overrightarrow{w}||^2 + C\sum_{i}^{N} \xi_i$$
$$- \sum_{i}^{N} \alpha_i(y_i(\overrightarrow{w}^T x_i + b) - 1 + \xi_i) - \sum_{i}^{N} \beta_i \xi_i, \tag{3.32}$$

donde α, β son los multiplicadores de Lagrange, ξ es un real positivo asociado a la clasificación de los datos, si su valor es mayor a uno la clasificación del elemento analizado es incorrecto y C es el parámetro de margen que determina la compensación entre la maximización del margen y la minimización del error de la clasificación. Esta minimización está restringida a la constricción

$$y_i(\overrightarrow{w}^T x_i + b) \geq 1 - \xi_i, \tag{3.33}$$

y las condiciones de Karush-Kuhn-Tucker (KKT) deben ser satisfechas [15]:

$$\frac{\partial Q}{\partial \overrightarrow{w}} = 0, \tag{3.34}$$

$$\frac{\partial Q}{\partial b} = 0, \tag{3.35}$$

$$\frac{\partial Q}{\partial \overrightarrow{\xi}} = 0, \tag{3.36}$$

$$\alpha_i(y_i(\overrightarrow{w}^T x_i + b) - 1 + \xi_i) = 0 \quad para \quad i \in [1, N], \tag{3.37}$$

$$\beta_i \xi_i = 0 \quad para \quad i \in [1, N], \tag{3.38}$$

$$\alpha_i \geq 0 \quad \beta_i \geq 0 \quad \xi_i \geq 0 \quad para \quad i \in [1, N]. \tag{3.39}$$

Siguiendo este desarrollo, que se encuentra totalmente desarrollado y explicado de forma matemática en [15], suponiéndose que se trabajará con un *kernel* del tipo RBF el hiperplano tendrá la forma

$$D(\overrightarrow{x}) = \sum_{i}^{N} \alpha_i y_i exp(-\gamma||\overrightarrow{x}_i - \overrightarrow{x}||^2) + b, \tag{3.40}$$

donde γ es un parámetro libre para controlar el tamaño del radio. De esta manera bajo las condiciones descritas anteriormente se pueden encontrar los parámetros adecuados para una clasificación óptima utilizando un algoritmo como lo es el *cross-validation*.

Problemas con multiclases

Un problema se considera multiclase si la clasificación debe ser sobre un número mayor a dos clases. Existen distintos métodos para esta selección, como lo son funciones de decisión indirecta, formulación uno contra todos, árbol de decisiones, entre otros [14]. El punto crucial aquí es que el desarrollo para el algoritmo

no es modificado. Comúnmente se utiliza el árbol de decisiones, este determina la i-ésima función de decisión $g_i(\overrightarrow{x})(i = 1, 2...n-1)$ donde n es el número de clases, donde $\overrightarrow{x}$ pertenece a la clase i si

$$g_i(\overrightarrow{x}) > 0 \tag{3.41}$$

y $\overrightarrow{x}$ pertenece a una de las clases en $[i + 1, .., n]$ si

$$g_i(\overrightarrow{x}) < 0. \tag{3.42}$$

Empezando por $i = 1$ hasta $n - 1$, se busca la primer función g la cual satisfaga $g_i(\overrightarrow{x}) > 0$, en caso contrario $\overrightarrow{x}$ pertenece a la clase n.

3.3.4 Implementación de una SVM

Se utilizó la biblioteca LIBSVM [16], que se puede obtener de `http://www.csie.ntu.edu.tw/~cjlin/libsvm`, la cual tiene implementada y optimizada una SVM que puede funcionar sobre los *kernel's* de tipo polinomial, sigmoidal o RBF, la cual también incluye una rutina en python llamada *grid* cuya función es desarrollar un *cross-validation* sobre una SVM con *kernel* del tipo RBF para obtener los mejores valores de los parámetros C y γ.

El procedimiento para desarrollar una correcta clasificación [16] es el siguiente:

- Transformar los datos a analizar a el formato requerido por LIBSVM
- Realizar un reescalamiento sobre los datos
- Considerar el *kernel* RBF, es decir, $K(\overrightarrow{x}, \overrightarrow{z}) = exp^{-\gamma||\overrightarrow{x}-\overrightarrow{y}||^2}$
- Usar la rutina grid para realizar el *cross-validation*
- Utilizar los mejores parámetros C y γ para entrenar la SVM
- Realizar la predicción

Como ejemplo se clasificará el conjunto de datos MNIST para explicar la implementación de una SVM con la biblioteca LIBSVM en python.

Comparación entre RNA, CNN y SVM

Una vez descargada la biblioteca se procede a obtener los ejecutables, esto se realiza utilizando el comando *make* en ubuntu, lo anterior generará tres ejecutables llamados *svm-train, svm-predict* y *svm-scale* cuyas funciones son: el primero se encarga de entrenar la SVM bajo el conjunto de entrenamiento, el segundo de predecir bajo el conjunto que sea heredado y el tercero reescala los datos de entrada. Las características de ellos pueden ser encontradas en [16]. La biblioteca contiene en la carpeta de *tools* la función *grid.py* encargada de realizar el *cross-validation* para encontrar los parámetros óptimos C y γ.

Una vez reescalados los datos del MNIST se procede a realizar el *cross-validation* cuya gráfica es mostrada en la Fig. 3.23 en la cual se grafican las curvas de nivel sobre la eficiencia de la SVM variando los parámetros en escala logarítmica de C y γ.

Los parámetros encontrados con los cuales la SVM tiene una mayor eficiencia son $C = 2.0$ y $\gamma = 0.03125$, con estos parámetros se procede a entrenar la SVM ejecutando *svm-train* con el comando siguiente; **./svm-train -s 0 -c 2.0 -g 0.03125 MNIST-train**. La variable -s 0 indica que se trabajará con el *kernel* RBF, mientras que las asignaciones de -c y -g corresponden a los valores de C y γ respectivamente, donde el modelo de la SVM será guardado en un arhivo llamado **MNIST-train.model**. De esta manera utilizando el modelo obtenido del entrenamiento para predicir la eficiencia ejecutando *svm-predict* mediante **./svm-predict MNISTtest MNISTtrain.model output** la SVM obtiene una eficiencia del 96.2% y la clasifi-

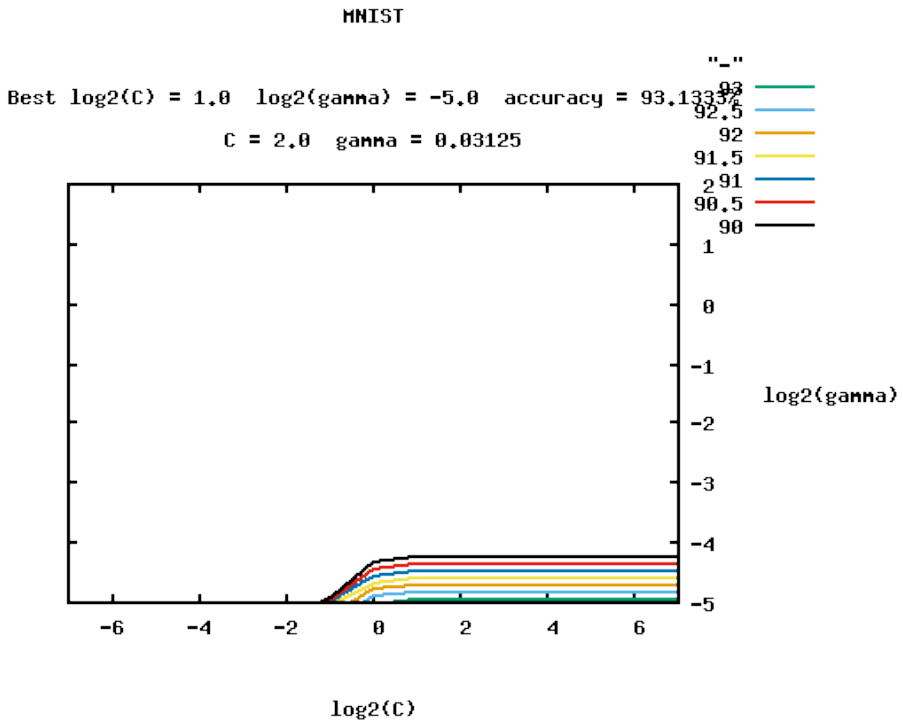


Fig. 3.23: *Cross-validation* sobre una SVM con el conjunto de entrenamiento del MNIST.

cación final de los datos de entrada son guardados en el archivo **output**[3]. En comparación con la RNA que obtiene un 95% y el 99% obtenido con la CNN, la SVM obtiene una eficiencia mayor que la RNA y menor que una CNN, sin embargo la principal ventaja fue que, en comparación con los algoritmos desarrollados para el *cross-validation* tanto para la RNA y CNN cuyos tiempos excedieron los dos días para obtener sus parámetros óptimos, el tiempo requerido por la función *grid.py* fué bastante rápido al ser de aproximadamente media hora.

Teniendo implementadas todas las herramientas computacionales que se utilizarán para clasificar las imágenes de los espectros, el siguiente capítulo describirá la forma en la cual se construyó un arreglo para la obtención de los espectros de las distintas concentraciones y su evaluación mediante un espectrofotómetro calibrado, previo a su análisis computacional para su clasificación.

[3]El algoritmo para convertir los datos al formato requerido por LIBSVM y los resultados correspondientes de la SVM pueden ser encontrada en el hipervínculo `http://www.ifm.umich.mx/~gvargas`, dentro de la carpeta llamada *MNIST*

Capítulo 4

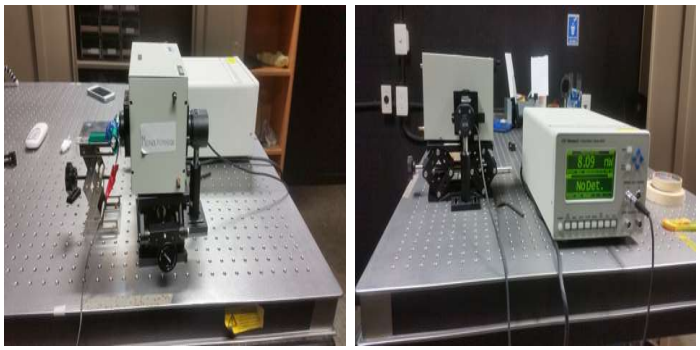
Clasificación de las muestras

4.1 Arreglo experimental

Con la finalidad de obtener los espectros de distintas muestras, que se describirán posteriormente, se montó un experimento basándose en la composición y funcionamiento de un espectrofotómetro convencional, el cual es conformado por un LED extraído de un foco Philips modelo 92900II237, una rejilla holográfica de 930 líneas/mm y una cámara samsung de 14 MP. Es de gran importancia saber la caracterización óptica del diodo, es decir la relación de la irradiancia respecto a cada longitud de onda que emite este LED. Esta relación fue obtenida con el arreglo descrito en la Fig. 4.1, el cual consiste en colocar el LED frente a un monocromador[1] seguido de un radiómetro. En este arreglo óptico se cambia la longitud de onda permitida con el monocromador y se mide la intensidad de salida con el radiómetro. Con este procedimiento se obtuvo una relación discreta mostrada en la Fig. 4.2 donde el eje y representa la medición de irradiancia en $nanoWatts$ y el eje x son las lon-

[1]Un monocromador es un dispositivo óptico que deja pasar solo una longitud de onda de todo el espectro que constituye una fuente de luz, así mismo permite variar esta longitud de onda emitida.

gitudes de onda en el rango de 400 nm a 700 nm.



(a) (b)

Fig. 4.1: Para caracterizar el LED en el espectro visible se utilizó un monocromador y un radiómetro tal como es mostrado en las imágenes (a) y (b). (a) Vista lateral del arreglo (b) vista frontal del arreglo.

Una consideración a tomar en cuenta es la sensibilidad de la cámara a una longitud de onda, es decir, podría suceder que no todo el espectro visible sea captado por la cámara, ya sea por la sensibilidad de la cámara o por la irradiancia a la cual emite el LED. Este problema fue solucionado al sustituir el radiómetro en la Fig. 4.1 por el celular. Variando el monocromador y usando la cámara se obtuvo que la sensibilidad de la cámara a este LED fue en el rango del visible tomado, es decir de 400nm a 700nm.

Sabiéndose las características del LED y de la cámara, se montó un arreglo como se muestra en la Fig. 4.3. Se situó un LED seguido de una rendija con apertura de $1mm$, una base para celdas, una rejilla de difracción y el celular colocados a $\theta = 58^o$ respecto al LED. Éste ángulo fue seleccionado por las características de la rejilla en la Ec. 2.14, considerando que se trabajó con el orden de difracción $m = 1$. La distancia entre la rendija y la celda que contiene a la muestra es de $9mm$ y la distancia entre el punto medio de la celda al centro de la cámara es de $34mm$.

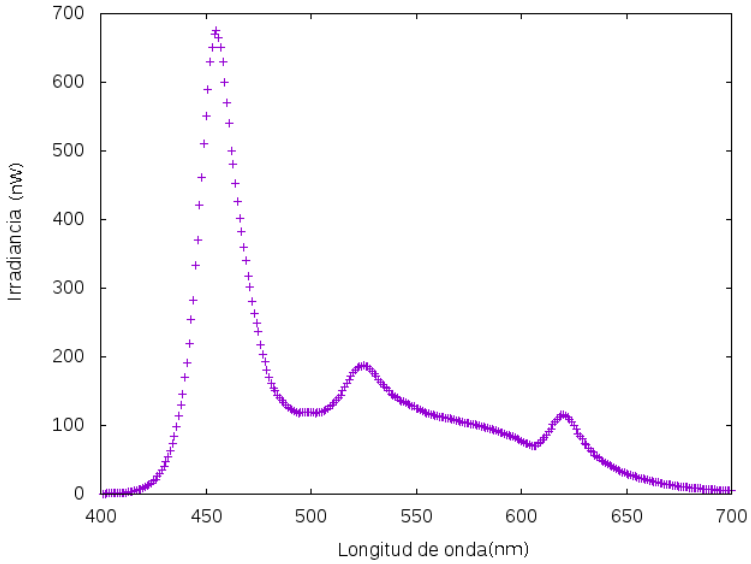


Fig. 4.2: Relación entre la irradiancia contra longitud de onda emitida por el LED I vs λ.

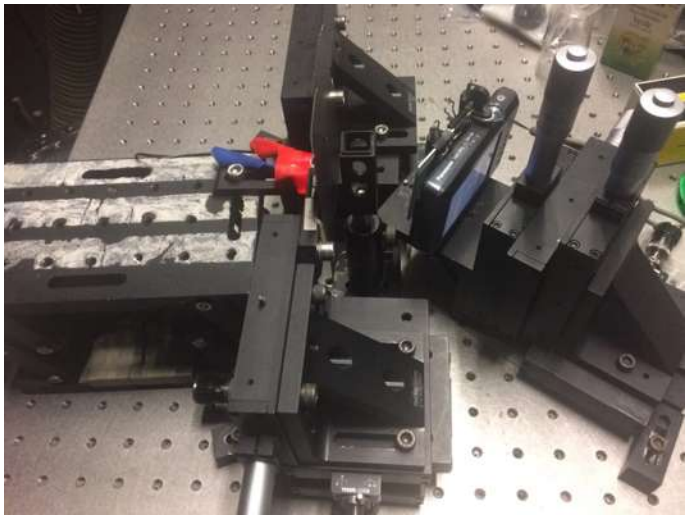
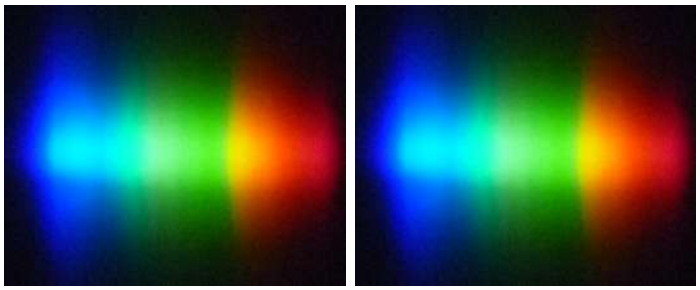


Fig. 4.3: Arreglo utilizado para la obtención del espectro con la cámara.

Las primeras imágenes que se tomaron fueron las correspondientes al espectro del LED las cuales se obtienen sin colocar objeto alguno entre el LED y la cámara, después de ellas se obtuvo el espectro de una muestra únicamente con agua, Fig. 4.4,

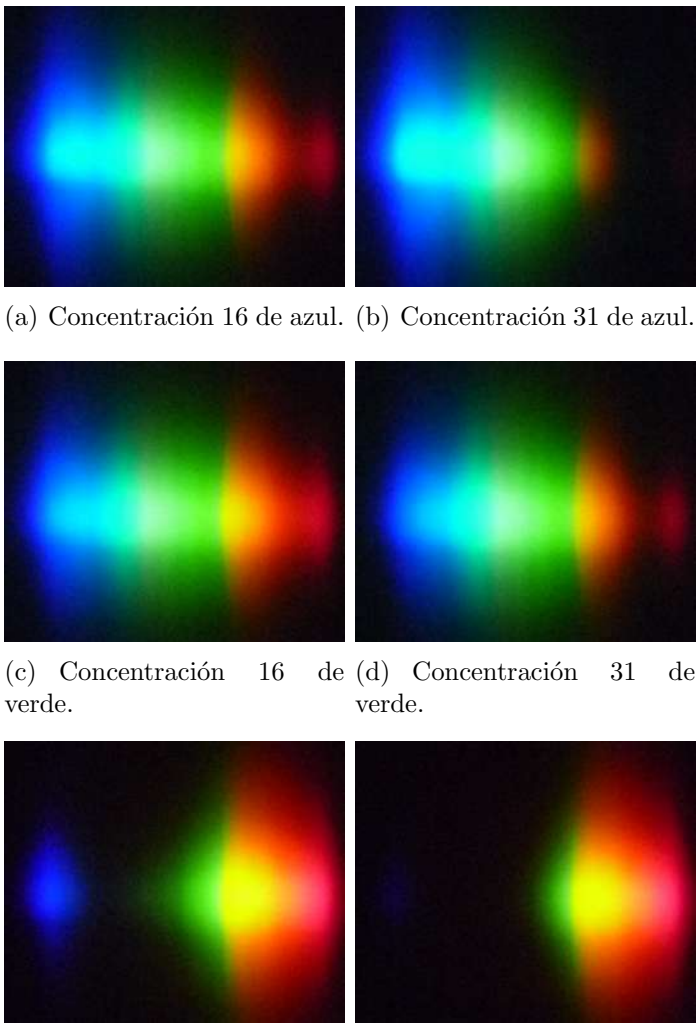
estos dos espectros serán utilizados como referencia en la comparación entre los espectros, en términos del funcionamiento de un espectrofotómetro este procedimiento también es conocido como auto-cero o calibración cero.



(a) (b)

Fig. 4.4: (a) Espectro tomado cuando no hay muestra, (b) espectro tomado a la muestra únicamente con agua.

Las sustancias a analizar fueron agua con colorantes vegetal verde al 1.78%, azul al 2% y rojo al 2% de McCormick. Para las muestras fue necesario diluir el colorante original en agua ya que es bastante concentrado, esto se realizó en relación de $1.25mL$ de colorante por $500mL$ de agua. Después de ello, la variación en la concentración fue de la siguiente manera: la mayor concentración es únicamente el colorante diluido con un volumen total de $3000\mu L$, la siguiente muestra consiste en retirar $20\mu L$ del colorante vegetal y reemplazarlo por la misma cantidad en agua, cada diez muestras la cantidad reemplazada de colorante fue de $50\mu L$ en lugar de $20\mu L$. Se tomaron 50 muestras por color dando un total de 150 muestras con distintas concentraciones. Para futuras referencias la mayor concentración de colorante será la muestra número 50 mientras que la más baja concentración será la número 1. Algunos espectros se muestran en la Fig. 4.5.



(a) Concentración 16 de azul. (b) Concentración 31 de azul.

(c) Concentración 16 de verde. (d) Concentración 31 de verde.

(e) Concentración 16 de rojo. (f) Concentración 31 de rojo.

Fig. 4.5: Espectros de las muestras a distintas concentraciones.

El siguiente paso consistió en analizar las distintas concentraciones con el espectrofotómetro Perkin-Elmer $\lambda - 35$ del laboratorio de óptica para ver el comportamiento de la transmitancia. Algunas relaciones de la transmitancia respecto a la longitud de onda obtenidas por el espectrofotómetro son mostradas en la Fig. 4.6.

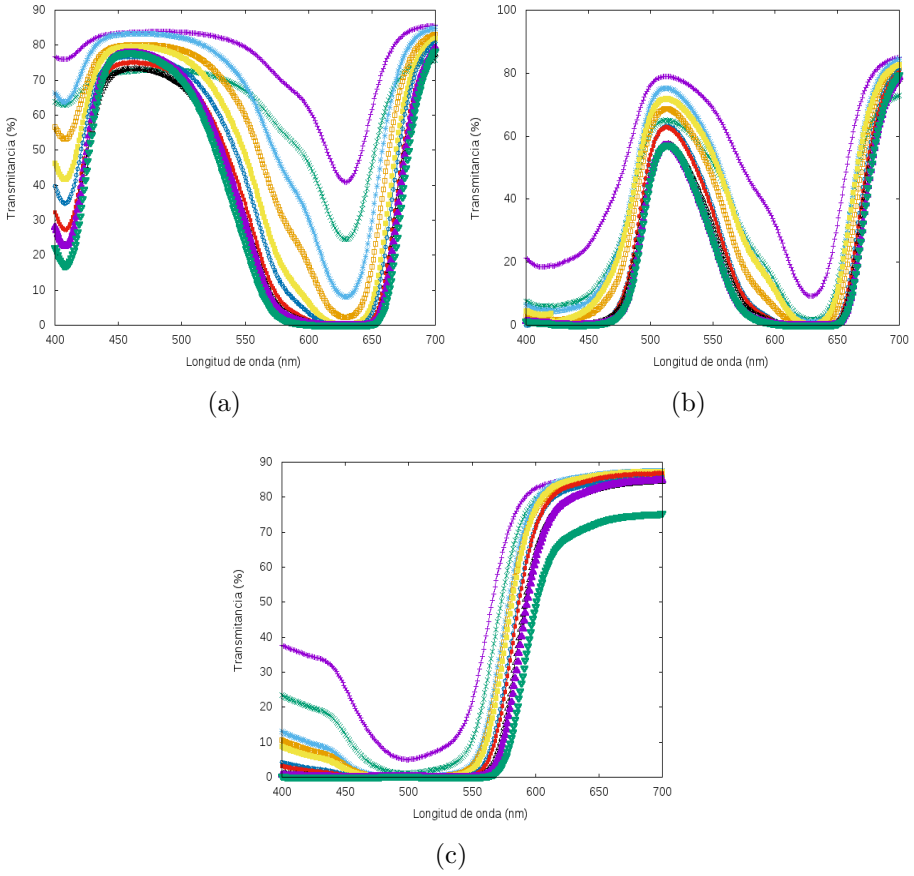


Fig. 4.6: Relación entre la transmitancia contra longitud de onda obtenidas con el espectrofotómetro Perkin-Elmer $\lambda - 35$. La relación para las muestras con colorante azul son mostrados en la imagen (a), (b) colorante verde y (c) colorante rojo. En los espectros se aprecia la variación en la transmitancia siendo el color morado de linea delgada la concentración número 5 para cada colorante vegetal y el color verde de linea gruesa la concentración número 50.

Una vez obtenida la información de las imágenes y analizadas por un instrumento calibrado, como lo es el espectrofotómetro Perkin-Elmer, se realizó la clasificación de las muestras en relación a su color y concentración con tres distintos algoritmos de clasificación que son las redes neuronales artificiales, redes neuronales convolucionales y máquinas de soporte vectorial mencionados en el capítulo 3. Esta clasificación se presenta en la siguiente

sección.

4.2 Algoritmos de clasificación

Se utilizó la plataforma de MatLab para procesar las imágenes previo a su clasificación, dicha plataforma asocia una imagen con tres matrices, RGB (Red, Green, Blue), también llamados canales. Cada posición de la matriz corresponde a los valores de cada pixel de la imagen, cuya escala es de 0 a 255. El auto-zero fué aplicado a todas las imágenes y esto consiste en restar de las imágenes la diferencia entre el espectro del LED y el espectro del agua mostrados en la Fig. 4.4.

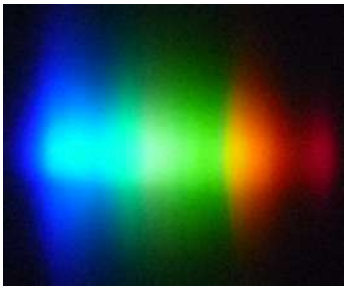
La división de muestras para los conjuntos de entrenamiento y predicción se realizará de la siguiente manera: para la clasificación por color se tomarán ciento treinta y cinco elementos para entrenar y quince para predecir y obtener la eficiencia del algoritmo clasificador en cuestión, mientras que para la clasificación por concentración será de cuarenta imágenes para entrenar y diez para predecir. La diferencia para la relación entre las imágenes para entrenar el algoritmo y para predecirlo entre la clasificación por color y concentración viene de la cantidad reducida de muestras con las que se cuenta, normalmente se utiliza una relación 90% para entrenar y 10% para predecir lo cual se puede realizar para la clasifiación por color, sin embargo esto no es pertinente para la clasifiación por concentración pues se tendrían únicamente 5 muestras para predecir. El algoritmo de *cross-validation* (CV) trabaja únicamente con el conjunto de entrenamiento y es necesario dividirlo en cinco subconjuntos con la misma cantidad de elementos, por lo tanto cada subconjunto para la clasificación por colores será de veintisiete imágenes de manera que 27x5 es igual a la cantidad de muestras de entre-

namiento (135) y para la concentración serán de ocho imágenes.

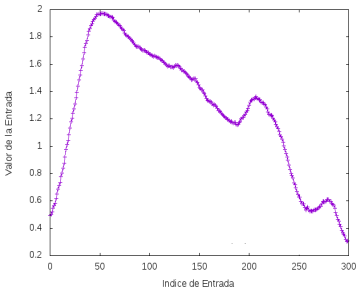
En la clasificación por concentración se realizarán dos clasificaciones; la primera de ellas consiste en cinco clases, la primer clase contiene la concentración 1 a la 10, la segunda de la 11 a la 20 y así consecutivamente para las restantes tres clases, esta clasificación será llamada primera clasificación. La segunda clasificación tendrá diez clases, donde la primera de ellas contendrá concentraciones entre la 1 y 5, para la segunda clase será de la concentración 6 a 10 y así concecutivamente. De esta manera se tendrá una aproximación de en que intervalos de concentración las variaciones en las imágenes son suficientes para clasificarlas correctamente. Los resultados para la predicción una vez desarrollado el *cross-validation* para las redes neuronales artificiales y convolucionales pueden ser encontradas en el hipervínculo `http://www.ifm.umich.mx/~gvargas` dentro de la carpeta llamada Prediccion.

4.2.1 Red neuronal artificial

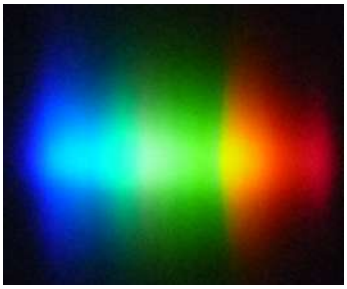
El tamaño de las imágenes mostradas en el capítulo 4 tienen más de mil pixeles por eje, si se analizara esta información podría requerirse un tiempo de cálculo bastante grande. Para el caso de la red neuronal artificial se decidió reducir la imagen a un tamaño de trecientos pixeles por eje utilizando la herramienta *imresize* de MatLab. La entrada de la red será sobre dos promedios en la imagen, el primer promedio será sobre los tres canales de color de la imagen mientras que el segundo promedio será sobre las columnas de la imagen para finalmente obtener una vector de trecientos elementos como la entrada de la red neuronal, Fig. 4.7.



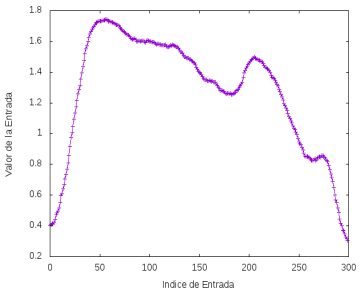
(a) Espectro de la concentración 16 de azul.



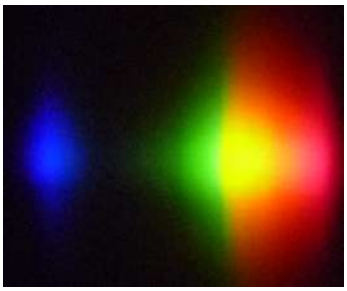
(b) Entrada de la red de la concentración 16 de azul.



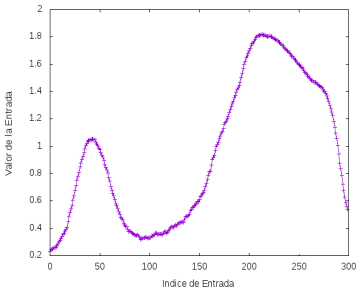
(c) Espectro de la concentración 16 de verde.



(d) Entrada de la red de la concentración 16 de verde.



(e) Espectro de la concentración 16 de rojo.



(f) Entrada de la red de la concentración 16 de rojo.

Fig. 4.7: Relación entre las imágenes de los espectros y las entradas a la red neuronal.

4.2.2 Código en python

Como se mencionó anteriormente, se utilizará la red neuronal como un clasificador, para ello se construyeron siete redes neu-

ronales, la primera de ella para clasificar las imágenes según sea su color, tres más para realizar la primer clasificación en la concentración y las restantes tres para realizar la segunda clasificación. La salida de la primera red será un arreglo de tres elementos, cada uno de ellos representará un color, por ejemplo si fuese azul la salida que se espera obtener sería un uno en el primer elemento y ceros en los restantes dos. Las siguientes tres redes, una por cada color, pretenden realizar la primera clasificación de la concentración en cinco clases para ello la salida de la red será de cinco neuronas, como ejemplo si se analizará la imagen correspondiente a la concentración número ocho de cualquier color debería ser uno la salida de la primer neurona al encontrarse entre las primeras diez concentraciones y cero los restantes cuatro. Análogamente, las últimas tres redes tratarán de realizar la segunda clasificación en diez clases, por lo que la salida de dicha red será de diez elementos, uno por cada clase.

Los datos que *TensorFlow* requiere son arreglos tipo numpy (Numpy es una paquete de Python que permite agilizar las operaciones matemáticas entre matrices) la cual es igualmente útil para la programación de las redes neuronales en Python por lo cual se realizó un algoritmo[2] el cual lee los datos de las imágenes obtenidas de su procesamiento previo en MatLab y entrega cuatro arreglos tipo numpy que corresponden a los conjuntos de entrenamiento y predicción, así como su valor deseado o valor objetivo de estos conjuntos.

Previo a la clasificación, se seleccionará cuales serán los parámetros con los cuales se programará la red mediante el método de *cross-validation*. Como se mencionó en el capítulo 3 existen distintos parámetros con los cuales se puede tratar de mejorar

[2] El código para convertir las imágenes en el formato requerido es llamado *crear_datos_entrada_salida.py* y puede ser encontrada en el hipervínculo `http://www.ifm.umich.mx/~gvargas`, dentro de las todas carpetas que tengan el nombre de red neuronal artificial, ya sea con Python o *TensorFlow*.

la eficiencia de la red, como son la cantidad de capas ocultas, número de neuronas por capa oculta, valor del tamaño de paso para la actualización de los pesos (ϵ), entre otros. Se tratará de encontrar un conjunto de parámetros adecuados variando únicamente las cantidad de neuronas en las capas ocultas. El código del *cross-validation* que se utilizó se puede encontrar en el hipervínculo `http://www.ifm.umich.mx/~gvargas` en la carpeta Red Neruronal Python dentro del directorio Cross-Validation, donde la tabla 4.1 muestra la relación entre los colores mostrados en las gráficas con la configuración de neuronas para cada capa utilizados para realizar el método.

Color	Primera capa oculta	Segunda capa oculta
Morado	250	250
Verde	300	250
Azul cielo	300	300
Naranja	350	300
Amarillo	350	350
Azul marino	400	350
Rojo	400	400
Nego	450	400
Morado línea delgada	450	450
Verde línea delgada	500	450
Azul cielo línea delgada	500	500

Tabla 4.1: Relación de colores en las gráficas de los resultados obtenidos con el método *cross-validation* para las clasificaciones utilizando la red neuronal en Python.

Las gráficas ilustradas en la Fig.4.8 muestran el resultado de aplicar el método de *cross-validation* sobre la clasificación de los colores, donde se aprecia que con doscientas cincuenta neuronas por capa, que es la configuración más pequeña que se programó, es suficiente para tener una buena clasificación.

Haciendo la evaluación de la red neuronal en cada ciclo de en-

trenamiento sobre el conjunto de predicción con los parámetros fijos de dos capas ocultas con doscientas cincuenta neuronas cada una, se obtuvo la gráfica mostrada en la Fig.4.9, que muestra la mejora en la predicción de la red con el paso de los ciclos. Claramente se ve que la red neuronal, con esta configuración, no tiene problema alguno para distinguir los colores.

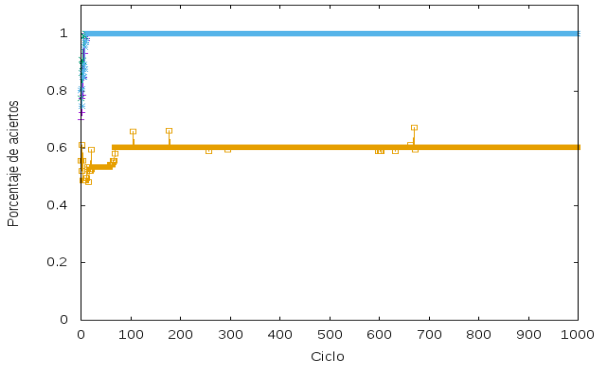


Fig. 4.8: *Cross-validation* sobre la clasificación de los colores con la red neuronal de dos capas ocultas programada en python.

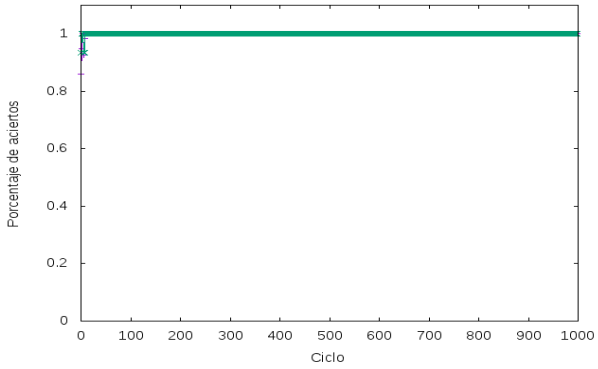


Fig. 4.9: Predicción sobre los conjuntos de predicción (color verde) y entrenamiento (color morado) en la clasificación de por colores con la red neuronal de dos capas ocultas con doscientas cincuenta neuronas en cada una.

La siguiente clasificación que se realizará será la primera clasificación sobre la concentración. Los resultados obtenidos del

método *cross-validation* son mostrados en las imagenes de la izquierda en la Fig. 4.10 mientras que las imágenes de la derecha muestran un ajuste de curvas a las imágenes de la izquierda mediante una función logarítmica, para poder seleccionar el mejor comportamiento y así mismo ayudar a seleccionar los parámetros de la red neuronal en cada clasificación sobre la concentración.

De esta manera se seleccionaron los mejores parámetros con el criterio de tener la menor cantidad de operaciones requeridas para obtener el mejor porcentaje de aciertos tal como se puede apreciar en el ajuste de las curvas en la Fig. 4.10. Los parámetros obtenidos en el análisis del color azul de la red neuronal con dos capas fueron de trescientas neuronas respectivamente en las capas ocultas, mientras que los parámetros de la red respectivos al color verde fueron trescientas neuronas en ambas capas e igualmente para el color rojo. Con estos parámetros establecidos, se procede a realizar el entrenamiento de la red previo a la propagación hacia adelante sobre el conjunto de predicción con el objetivo de encontrar la eficiencia de las redes, cuyos resultados son mostrados en la Fig. 4.11. Para la primer clasificación de concentración en el color azul se obtuvo un 100% de eficiencia, mientras que la eficiencia sobre el color verde y rojo fué del 90%. Repitiendo el proceso de *cross-validation* sobre la segunda clasificación de concentraciones en cada color se obtuvieron los resultados mostrados en la Fig. 4.12. La cual se utilizó nuevamente en la determinación de las mejores configuraciones en los parámetros de la red, en este caso fueron de doscientas cincuenta neuronas por capa para los colores azul y rojo, y un total de trescientos cincuenta neuronas por capa para el verde. Con esta configuración se implementó y entrenó la red neuronal para su posterior propagación hacia adelante en el conjunto de predicción, obteniendose así la eficiencia de la red, estos resultados son mostrados en la Fig. 4.13.

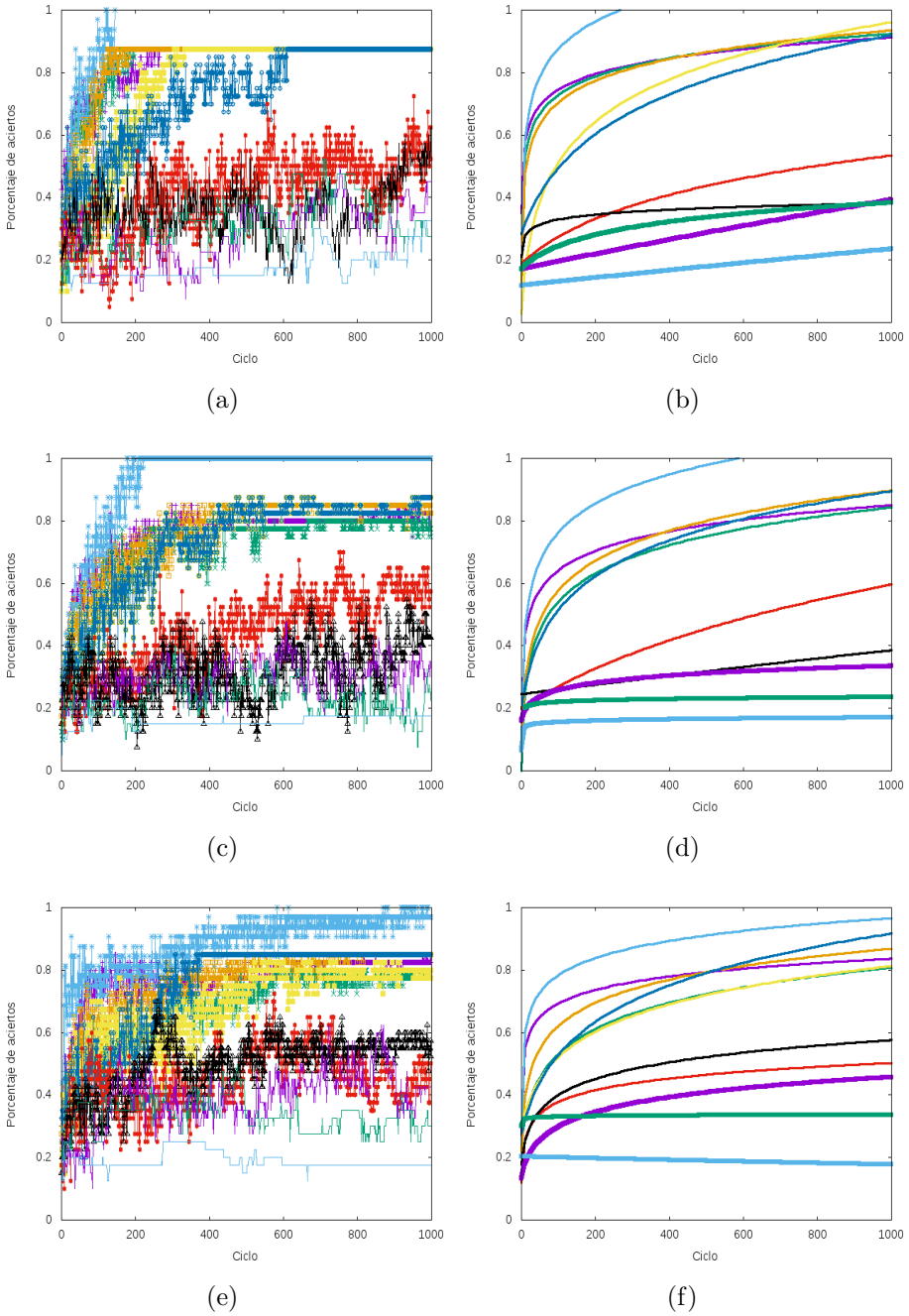
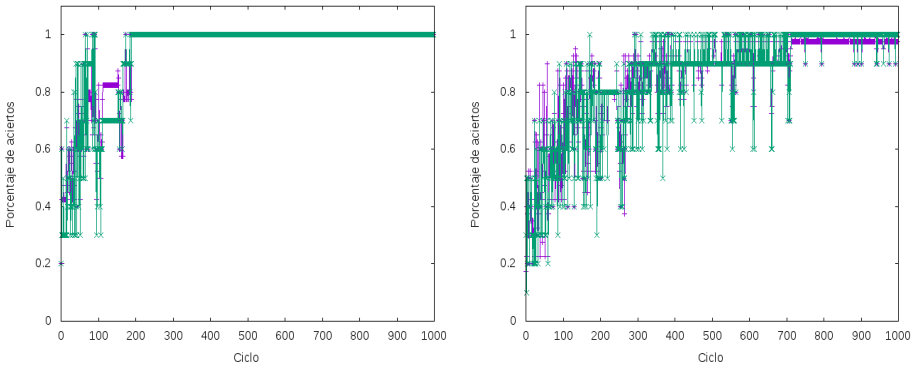
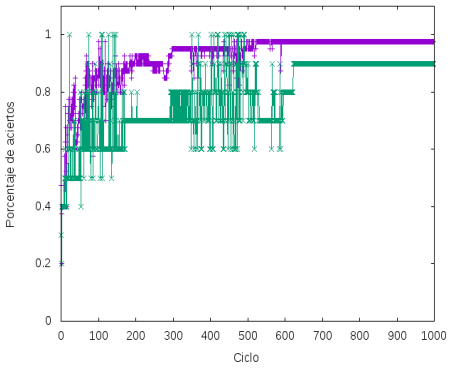


Fig. 4.10: *Cross-validation* sobre la primer clasificación de una red neuronal con dos capas para el color azul (a), verde (c) y rojo (e). El ajuste a los resultados del CV por una logaritmo son mostrados en las gráficas (b), (d) y (f) para el color azul, verde y rojo respectivamente.



(a)

(b)



(c)

Fig. 4.11: En estas gráficas se muestran los resultados de la red neuronal de dos capas ocultas sobre el conjunto de predicción (color verde) y entrenamiento (color morado) de la primera clasificación en las concentraciones de los colores azul (a), verde (b) y rojo (c).

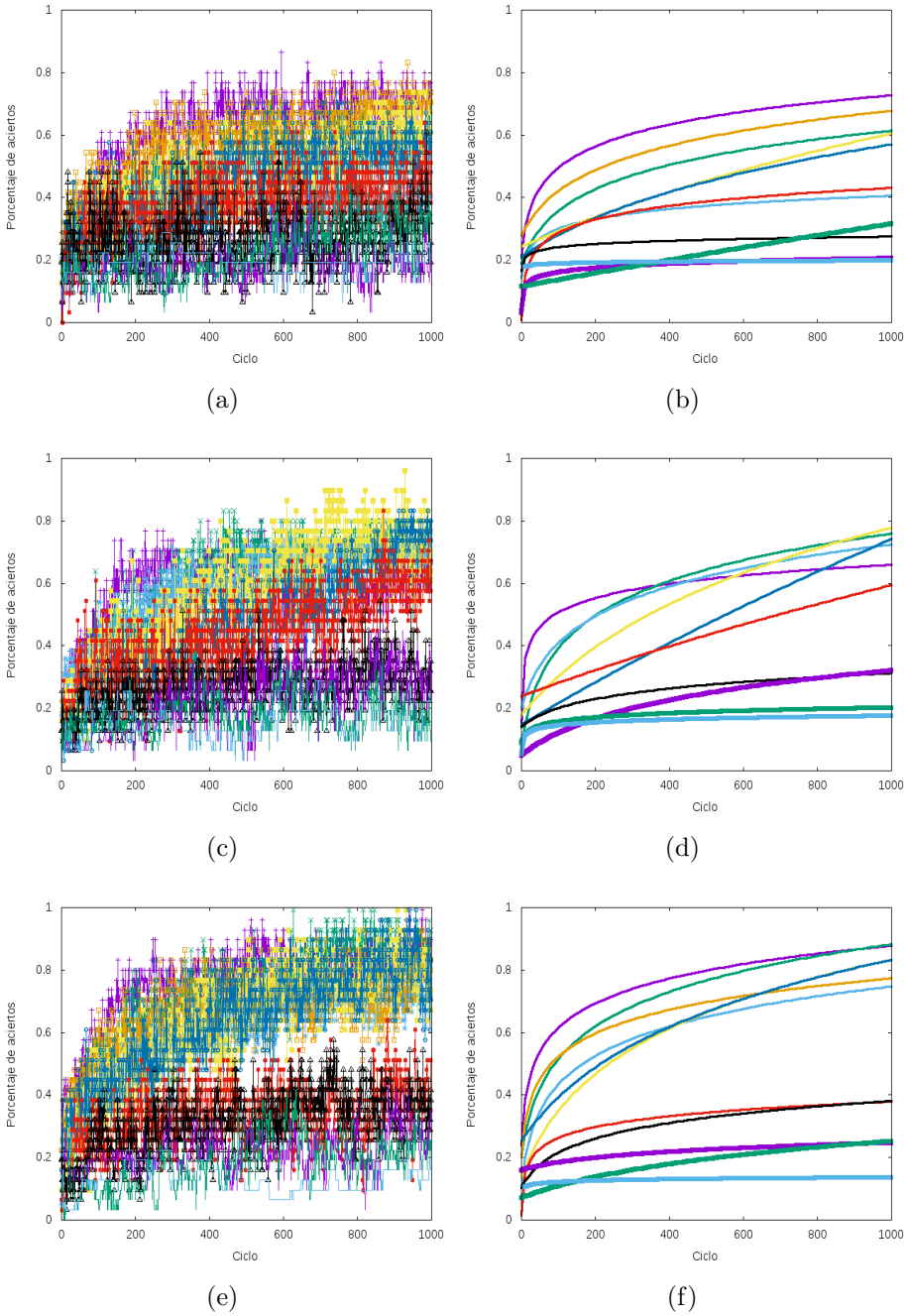
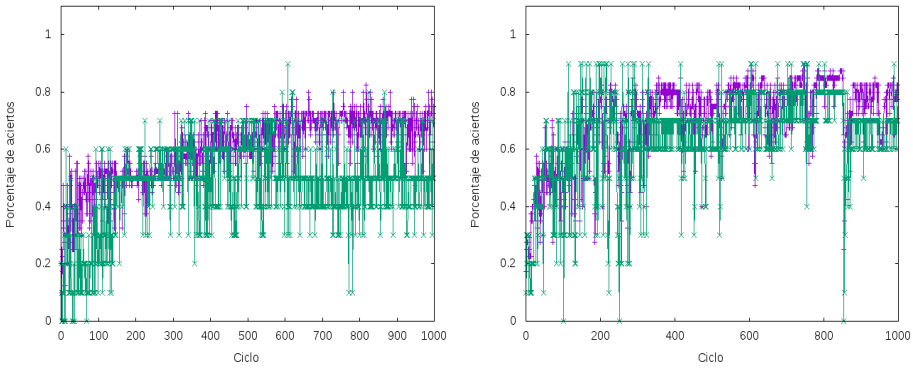
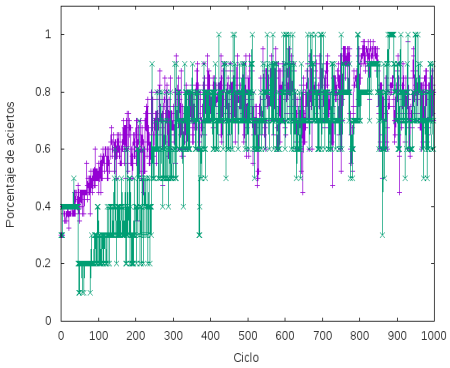


Fig. 4.12: (a) *Cross-validation* sobre la segunda clasificación de una red neuronal con dos capas para el color azul (a), verde (c) y rojo (e). El ajuste a los resultados del CV por una logaritmo son mostrados en las gráficas (b), (d) y (f) para el color azul, verde y rojo respectivamente.



(a)

(b)



(c)

Fig. 4.13: En estas gráficas se muestran los resultados de la red neuronal de dos capas ocultas sobre el conjunto de predicción (color verde) y entrenamiento (color morado) de la segunda clasificación en las concentraciones de los colores azul (a), verde (b) y rojo (c).

4.2.3 Código en TensorFlow

La clasificación de las imágenes se tratará igual que la red programada en Python con siete redes neuronales, donde previamente los datos obtenidos de las imágenes fueron llevados al formato que *TensorFlow* requiere.

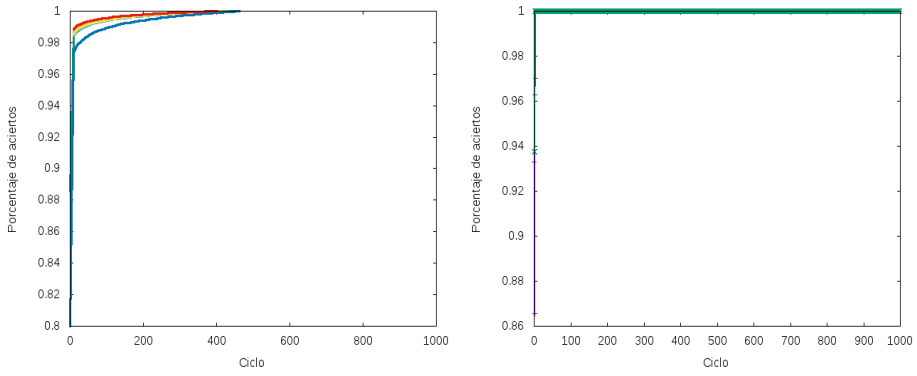
Partiendo de la red neuronal que se utilizó para catalogar los datos de la MNIST, en el capítulo 3, se implementó una red análoga para hacer el *cross-validation*, el cual se puede encontrar en el hipervínculo `http://www.ifm.umich.mx/~gvargas` en la carpeta Red Neruronal TensorFlow dentro del directorio Cross-Validation, para encontrar el arreglo de parámetros óptimo con el cual se obtendrá la mejor eficiencia, en este caso las capas ocultas se fijaron en tres y se modificaron las cantidades de neuronas por capa oculta para hacer el *cross-validation*. Realizando el *cross-validation*, primero sobre la clasificación por color, los resultados mostrados en la Fig. 4.14.a. La relación de colores y configuración de neuronas por capa oculta se encuentra en la tabla 4.2.

Color	Primer capa	Segunda capa	Tercer capa
Morado	100	100	100
Verde	250	100	100
Azul cielo	250	250	100
Naranja	250	250	250
Amarillo	250	250	500
Azul marino	250	500	500
Rojo	500	500	500

Tabla 4.2: Relación de colores en las gráficas de los resultados obtenidos con el método *cross-validation* para las clasificaciones utilizando la red neuronal en *TensorFlow*.

En vista de los resultados se optó por trabajar con la red que requiere menor cantidad de operaciones, es decir, la red que

contiene cien neuronas por capa oculta. Los resultados sobre el conjunto de predicción son mostrados en Fig. 4.14.b, al igual que el código programado con python, la red neuronal en *TensorFlow* no tiene problema alguno en distinguir que color está analizando.



(a) (b)

Fig. 4.14: *Cross-validation* y predicción sobre la clasificación en colores con la red neuronal de tres capas ocultas programada con *TensorFlow*.

La siguiente parte consiste en determinar la configuración de neuronas para la primera y segunda clasificación de las muestras por su concentración, esto se realizó utilizando nuevamente el método *cross-validation*, los resultados son mostrados en la Fig. 4.15 y Fig. 4.16 respectivamente. Las gráficas en el lado izquierdo presentan los resultados del método por iteración y su ajuste mediante un logaritmo a una curva son mostrados en las gráficas en el lado derecho.

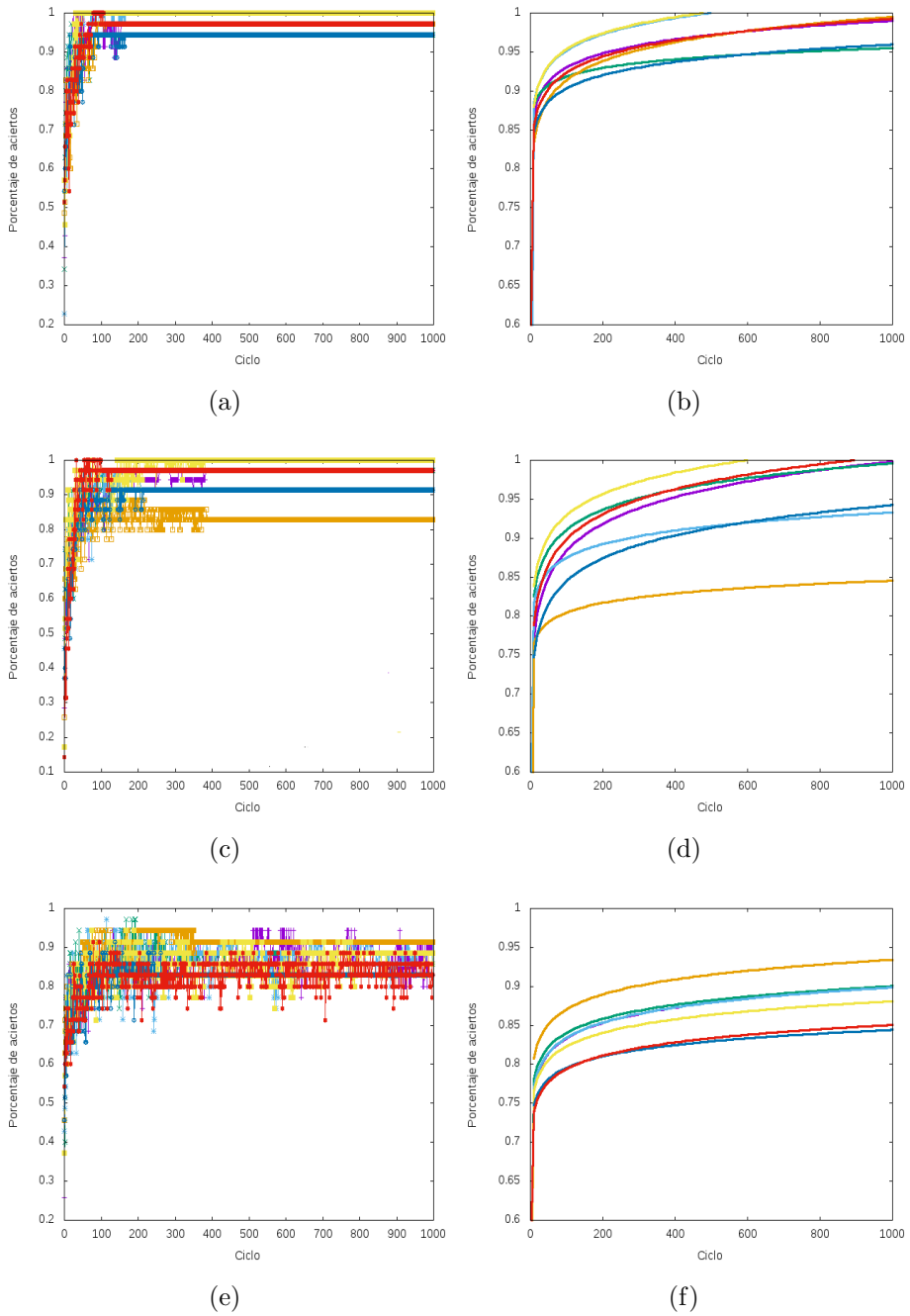


Fig. 4.15: *Cross-validation* sobre la primera clasificación de una red neuronal con dos capas para el color azul (a), verde (c) y rojo (e) utilizando *TensorFlow*. El ajuste a los resultados del CV por una logaritmo son mostrados en las gráficas (b), (d) y (f) para el color azul, verde y rojo respectivamente.

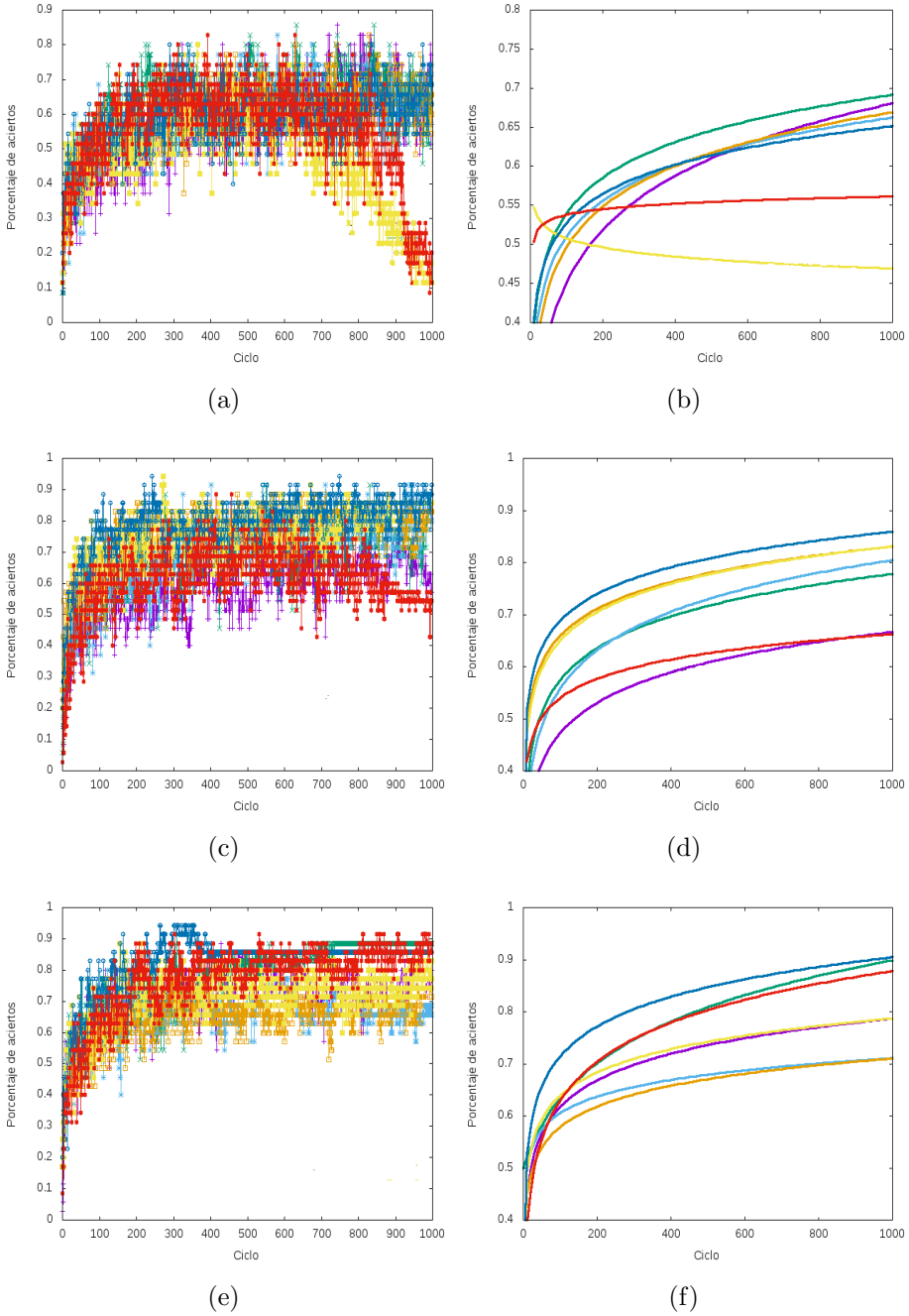
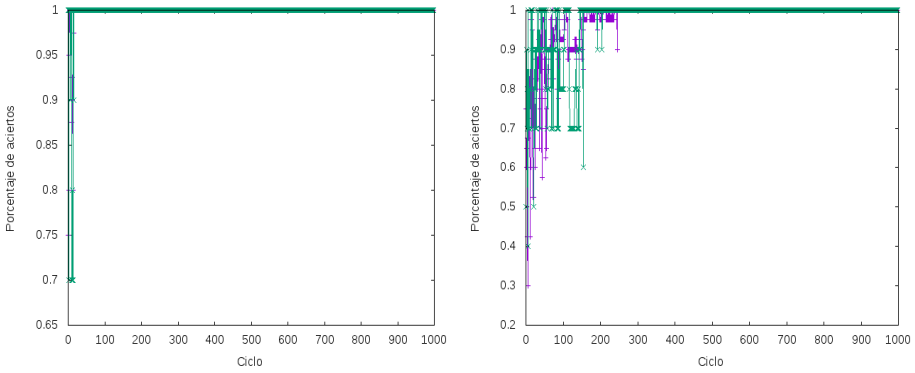


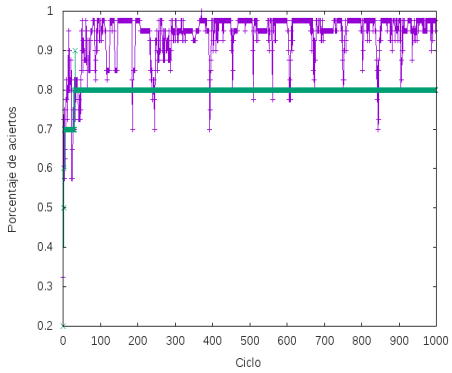
Fig. 4.16: *Cross-validation* sobre la segunda clasificación de una red neuronal con dos capas para el color azul (a), verde (c) y rojo (e) utilizando *TensorFlow*. El ajuste a los resultados del CV por una logaritmo son mostrados en las gráficas (b), (d) y (f) para el color azul, verde y rojo respectivamente.

Con ayuda del ajuste a los resultados de *cross-validation* se escogieron los siguientes parámetros para las distintas redes y su clasificación correspondiente: para la primer clasificación se escogieron para los colores azul y rojo docientos cincuenta neuronas por cada capa oculta y para el color verde docientos cincuenta en las primeras dos capas ocultas y quinientas en la última. Para la segunda clasificación fueron cien neuronas por capa oculta para el azul, doscientos cincuenta en la primer capa y quinientas en las restantes para el verde y doscientos cincuenta en la primer capa y cien en las restantes para el rojo. Los resultados del la propagación hacia adelante en el conjunto de predicción en cada ciclo del entrenamiento de la red con los parámetros escogidos son mostrados en la Fig. 4.17 y Fig. 4.18 para la primer y segunda concentración respectivamente.



(a)

(b)



(c)

Fig. 4.17: En estas gráficas se muestran los resultados de la red neuronal programada con *TensorFlow* sobre el conjunto de predicción (color verde) y entrenamiento (color morado) de la primera clasificación en las concentraciones de los colores azul (a), verde (b) y rojo (c).

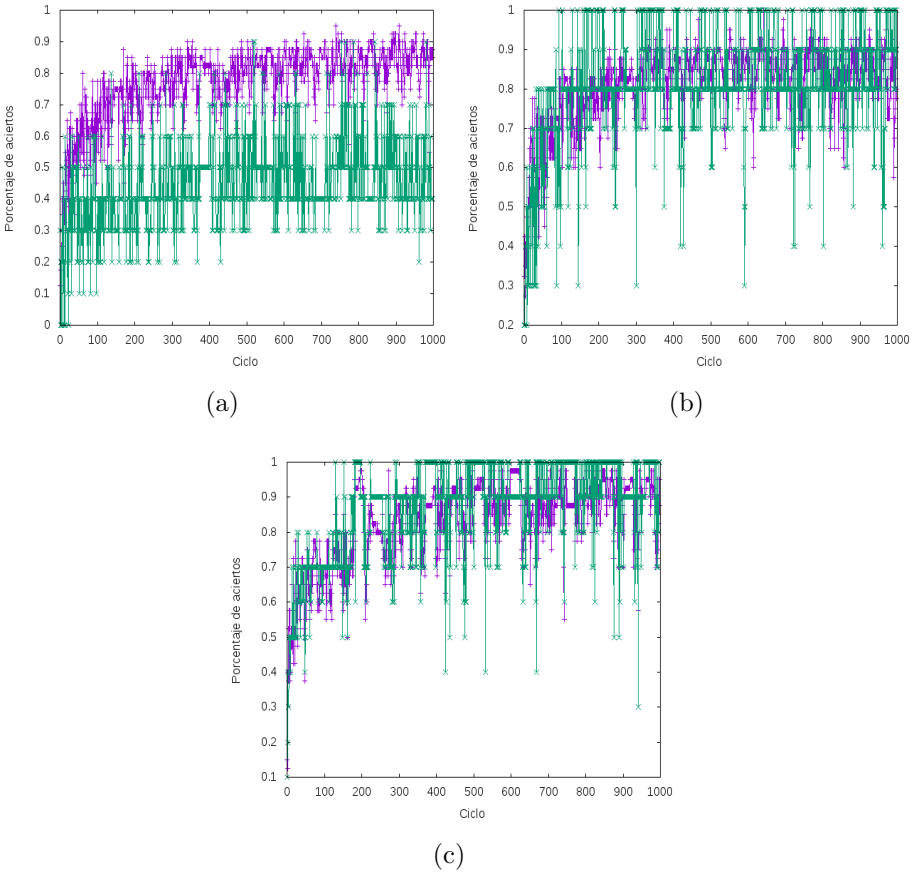
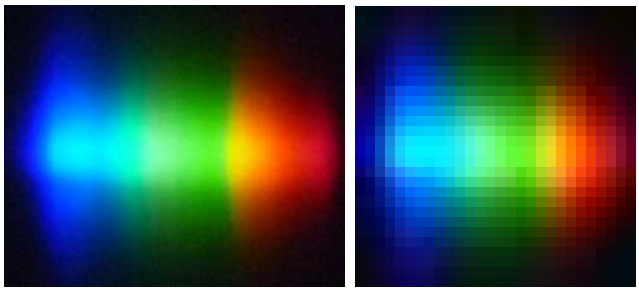


Fig. 4.18: En estas gráficas se muestran los resultados de la red neuronal programada con *TensorFlow* sobre el conjunto de predicción (color verde) y entrenamiento (color morado) de la segunda clasificación en las concentraciones de los colores azul (a), verde (b) y rojo (c).

4.2.4 Red neuronal convolucional

Por la razón mencionada en el capítulo 3, se redujo la escala de las imágenes de más de mil pixeles por eje a una imagen de 28x28, que tiene una cantidad considerable de información, la escala final es mostrada en la Fig. 4.19 en la cual se observa la disminución en la resolución en la imagen aunque de forma gene-ral conserva su topología. Nótese que al trabajar con los espectros en primera instancia se puede asumir que los colores como tal no son relevantes ya que cada espacio en la imagen corresponde a un valor esperado de color, en otras palabras se espera que del lado izquierdo de la imagen se obtenga la información que corresponde a la transmitancia del color azul mientras que del lado derecho se espera obtener la información correspondiente al rojo. Por esto, una segunda reducción que se tomó fue trabajar con la escala de grises de las imágenes reduciendo en tres la información por analizar.



(a) (b)

Fig. 4.19: Reducción de imágenes donde la imagen (a) es del tamaño otiginal y (b) es la reducción.

Una vez determinada como serán las entrada de la CNN se desarrolla el mismo proceso de determinación de parámetros con *cross-validation*[3] y una vez establecidos se sigue con la predicción

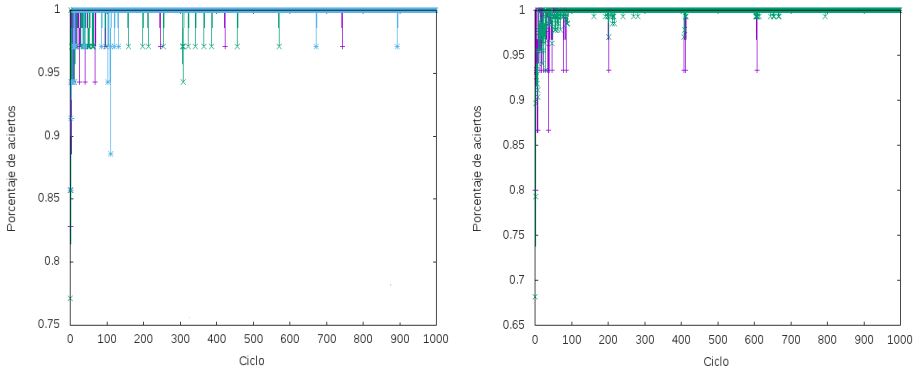
[3]Los algortimos necesarios para realizar el *cross-validation* se encuentran en el hipervínculo `http://www.ifm.umich.mx/~gvargas` dentro de se la carpeta CNN en el directorio Cross-Validation

tal como se hizo con las RNAs en Python y *TensorFlow*. Los parámetros que se modificaran en el *cross-validation* son el tamaño de los filtros y la cantidad de filtros por convolución. Se dejarán fijos los siguientes parámetros: dos capas de convolución, dos capas de *pooling* y las neuronas en la capa de conexión total serán mil sesenta y cuatro en la capa oculta. El análisis de la clasificación por colores es mostrado en la Fig. 4.20.a y su relación entre los colores de las gráficas y la configuración de la red convolucional es mostrada en la tabla 4.3. Al tener todas las configuraciones un excelente comportamiento para clasificar por color se optó por tomar el filtro de mayor tamaño (7x7) para realizar el entrenamiento final y predicción ya que es el que menor operaciones requiere (24 y 58 filtros por capa de convolución), este resultado es mostrado en la Fig. 4.20.b.

Finalmente se llevo a cabo el mismo procedimiento sobre la primera y segunda clasificación, los resultados del *cross-validation* son mostrados en la Fig. 4.21 y Fig. 4.22 respectivamente para la primer y segunda concentración.

Color	Tamaño del filtro	Cantidad de filtros	Cantidad de filtros
Morado	3x3	24	58
Verde	5x5	24	58
Azul cielo	7x7	24	58
Naranja	3x3	32	64
Amarillo	5x5	32	64
Azul marino	7x7	32	64
Rojo	3x3	40	70
Negro	5x5	40	70
Mousse	7x7	40	70

Tabla 4.3: Relación entre colores y configuración de la red neuronal convolucional, en las gráficas de los resultados obtenidos con el método *cross-validation* utilizando *TensorFlow*.



(a) (b)

Fig. 4.20: (a) *Cross-validation* sobre la segunda clasificación por color utilizando una CNN, (b) muestra la predicción de la CNN con 24 y 58 filtros por capa de convolución con un tamaño de filtro de 7x7 al ser la configuración de menor operaciones requeridas.

Con base en los ajustes a las curvas de los datos se determinó la siguiente selección de parámetros para las distintas clasificaciones: para la primer clasificación para el color azul serán filtros de dos capas de cuarenta y setenta con un tamaño de 5x5, para el rojo serán de veinticuatro y cincuenta y ocho filtros de 7x7 y la cantidad de filtros para el color verde serán de cuarenta y setenta respectivamente con un tamaño de 3x3. Para la segunda clasificación el color azul tendrá cuarenta y setenta filtros respectivamente en sus capas de filtros con un tamaño de 7x7, para el color verde tendrá la misma cantidad de filtros por capa con un tamaño de 3x3 y el color rojo tendrá veinticuatro y cincuenta y ocho filtros respectivamente en sus capas de un tamaño de 7x7. Con estos parámetros establecidos se implementó una CNN y los resultados se muestran en la Fig. 4.23 y Fig. 4.24 para la primer y segunda concentración respectivamente.

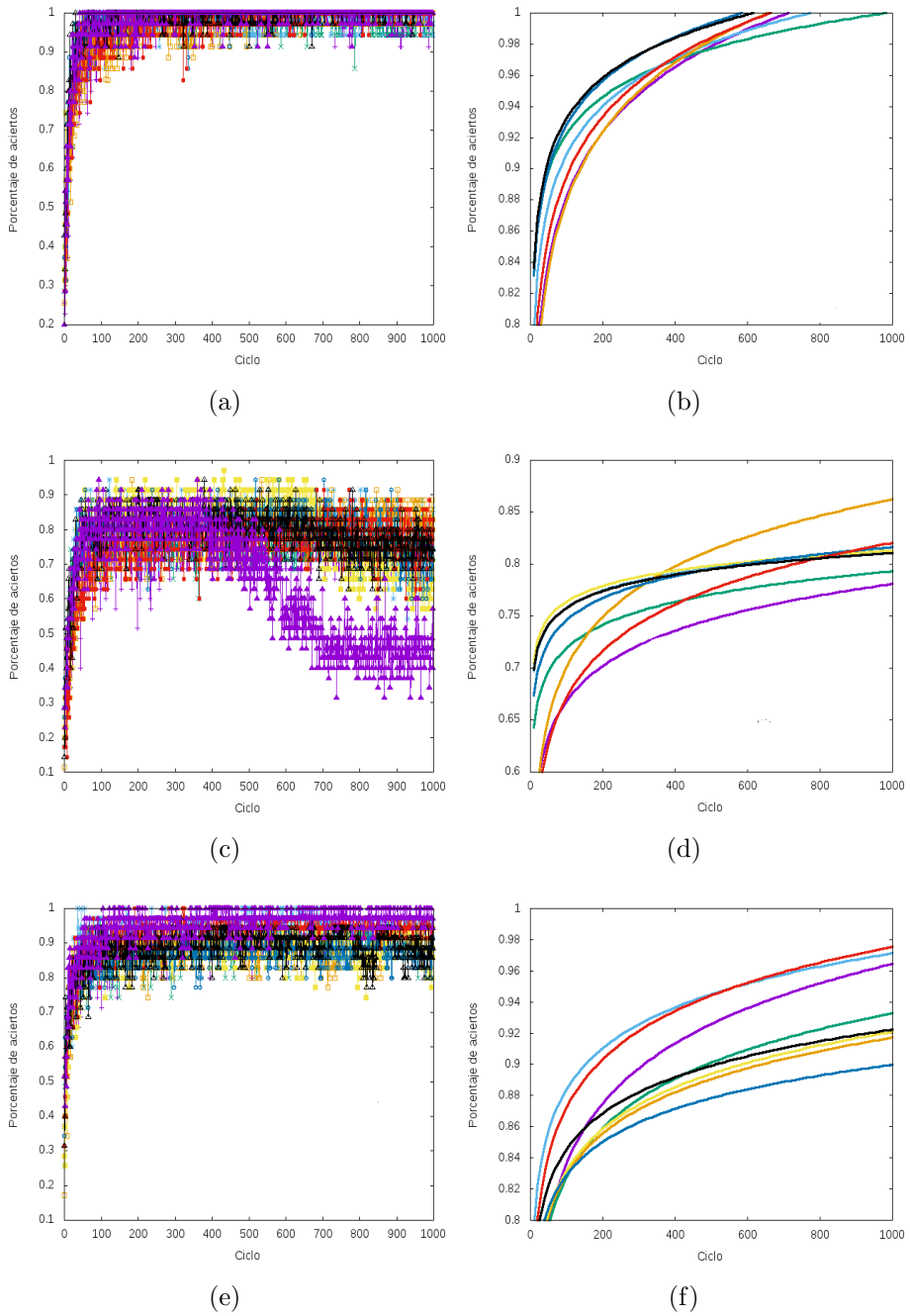
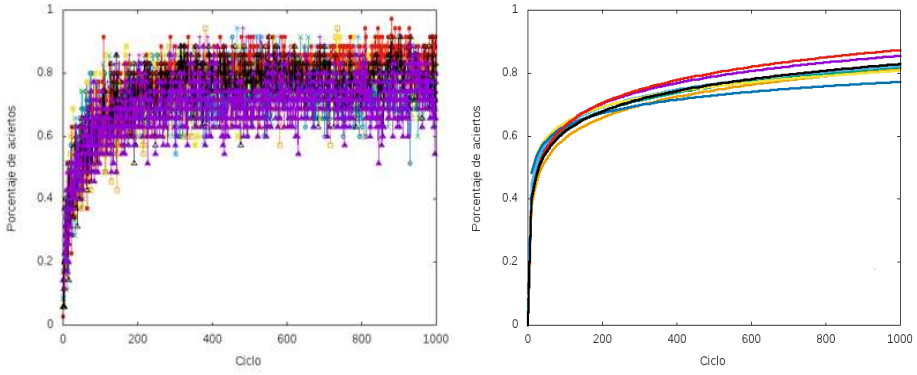
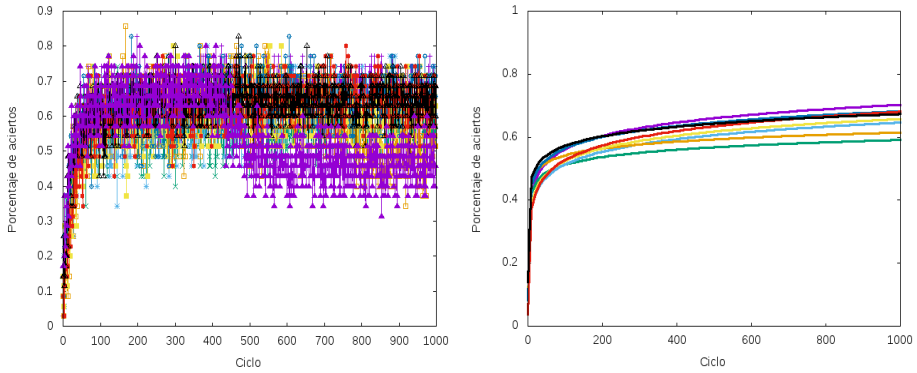


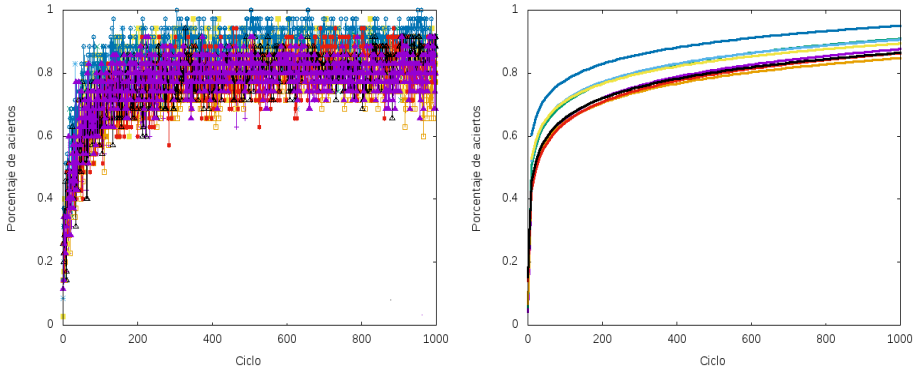
Fig. 4.21: *Cross-validation* sobre la primera clasificación de una red neuronal convolucional con *TensorFlow* para el color azul (a), verde(c) y rojo (e). Los ajustes a los resultados del CV por un logaritmo son mostrados en las gráficas (b), (d) y (f) para los colores azul, verde y rojo respectivamente.



(a) (b)



(c) (d)



(e) (f)

Fig. 4.22: *Cross-validation* sobre la segunda clasificación de una red neuronal convolucional con *TensorFlow* para el color azul (a), verde(c) y rojo (e). Los ajustes a los resultados del CV por un logaritmo son mostrados en las gráficas (b), (d) y (f) para los colores azul, verde y rojo respectivamente.

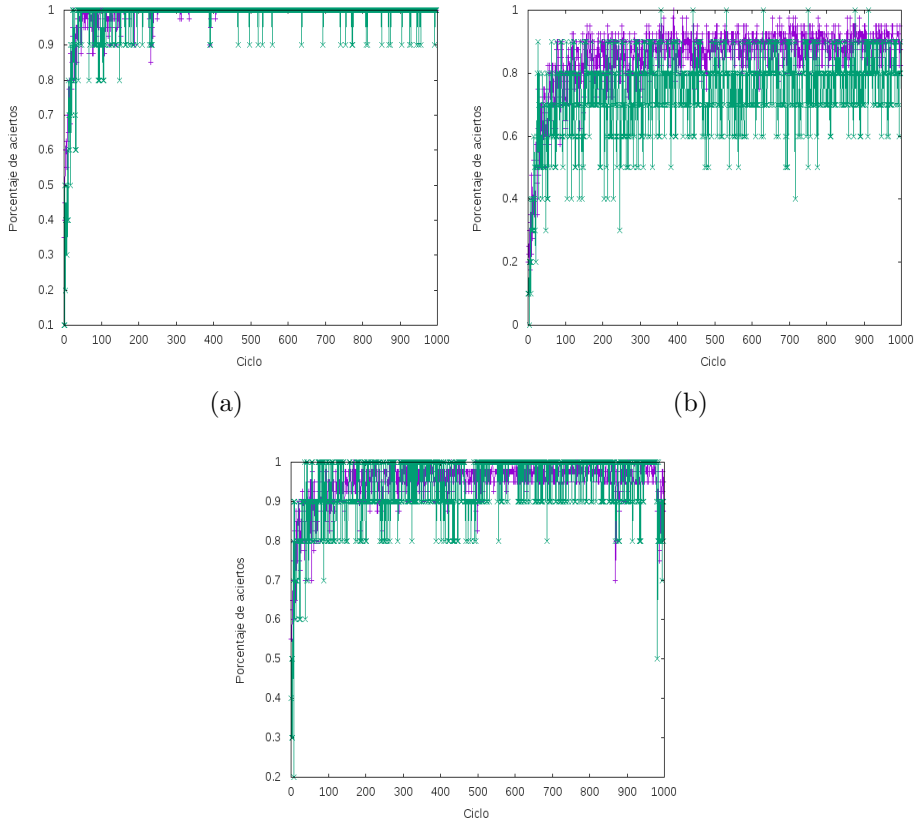


Fig. 4.23: En estas gráficas se muestran los resultados de la red neuronal convolucional programada con *TensorFlow* sobre el conjunto de predicción (color verde) y entrenamiento (color morado) de la primera clasificación en las concentraciones de los colores azul (a), verde (b) y rojo (c).

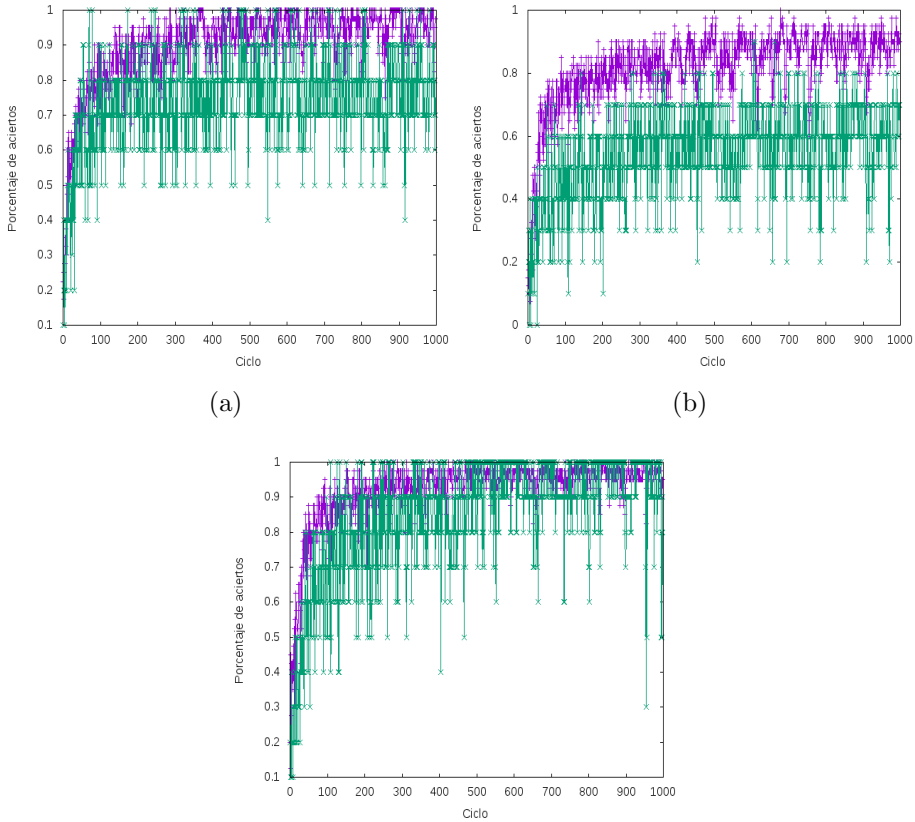


Fig. 4.24: En estas gráficas se muestran los resultados de la red neuronal convolucional programada con *TensorFlow* sobre el conjunto de predicción (color verde) y entrenamiento (color morado) de la segunda clasificación en las concentraciones de colores azul (a), verde (b) y rojo (c).

4.2.5 Máquinas de soporte vectorial

Después de reescribir los datos obtenidas de las imágenes reducidas, tal como se realizó para una CNN, se escriben con el formato requerido para la biblioteca LIBSVM [16]. Posteriormente se reescalan tomando valores entre -1 y 1, los archivos en el nuevo formato así como los ejecutables pueden ser encontrados en el hipervínculo `http://www.ifm.umich.mx/~gvargas` dentro de la carpeta llamada SVM. Utilizando nuevamente la

función *grid* para realizar el *cross-validation* primero para la clasificación por color para después pasar a la clasificación sobre la primer y segunda clasificación. Las curvas de nivel obtenidas para la clasificación por color son mostradas en la Fig. 4.25, con los parámetros $C = 0.25$ y $\gamma = 0.25$, requeridos para la optimización del *kernel* RBF, se obtiene una eficiencia del 100% sobre el conjunto de entrenamiento y predicción, al igual que la RNA y la CNN no hay problema alguno para clasificar las muestras por su color.

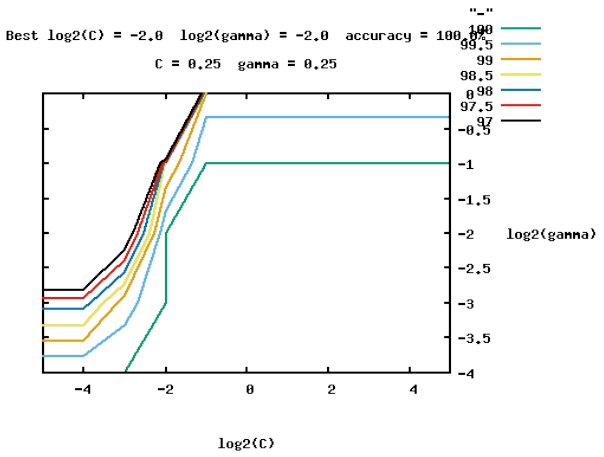


Fig. 4.25: Curvas de nivel del *cross-validation* sobre una SVM para la clasificación de las imágenes por su color.

Para la clasificación de los espectros por sus concentraciones, para la primera concentración el CV es mostrado en la Fig. 4.26, mientras que el CV sobre la segunda clasificación es mostrado en la Fig. 4.27. De esta manera obtenemos los parámetros con los cuales se llevará acabo el entrenamiento y su posterior predicción, los resultados son mostrados en la Tabla 4.4 para la primer concentración y en la Tabla 4.5 para la segunda concentración.

Primera concentración				
Color	C	γ	% entrenamiento	% predicción
Azul	0.0078125	0.015625	97.5	100
Verde	0.0078125	0.015625	87.5	90
Rojo	1.0	0.0625	97.5	100

Tabla 4.4: Parámetros óptimos obtenidos de la función *grid* para la primer clasificación utilizando la SVM con *kernel* RBF.

Segunda concentración				
Color	C	γ	% entrenamiento	% predicción
Azul	0.0078125	0.0625	100	80
Verde	8.0	0.0078125	92.5	70
Rojo	0.0078125	0.0625	97.5	80

Tabla 4.5: Parámetros óptimos obtenidos de la función *grid* para la segunda clasificación utilizando la SVM con *kernel* RBF.

4.3 Discusión de resultados

Utilizando los resultados de las redes neuronales, ya sean artificiales 4.2.1 o convolucionales 4.2.4, se realizó un ajuste a las salidas obtenidas en la predicción. Para el caso de la primer clasificación los ajustes sobre los datos obtenidos en la predicción para los colores azul, verde y rojo son mostrados en la Fig. 4.28 y para la segunda clasificación en la Fig. 4.29. El significado de cada color es mostrado en la tabla 4.6. Estas gráficas permíten decidir cual algoritmo tiene una mejor clasificación, se considerará como el mejor algoritmo clasificador al cuál tenga mayor eficiencia entre las distintas clasificaciones. Bajo este criterio, la red neuronal convolucional fue elegida como la mejor, al ser clasificar correctamente en cuatro (Fig. 4.28.a y c, Fig. 4.29.a y c) de las seis clasificaciones realizadas.

La forma de comparar los resultados obtenidos de las redes neuronales y la SVM es mediante la comparación entre la tendencia de los ajustes a las salidas de las redes y la eficiencia de la SVM,

Color	Método de aprendizaje de máquina
Morado	RNA en python con 2 capas ocultas
Verde	RNA en python con 3 capas ocultas
Azul cielo	RNA con *TensorFlow*
Naranja	CNN con *TensorFlow*

Tabla 4.6: Relación entre colores y el método de aprendizaje de máquina utilizado.

la cual es mostrada para la primer clasificación en la Tabla 4.7 y para la segunda clasificación en la Tabla 4.8.

Primer clasificación		
Color	% predicción máxima con redes	% predicción con la SVM
Azul	100 (CNN)	100
Verde	100 (RNA)	90
Rojo	100(CNN)	100

Tabla 4.7: Comparación de resultados para la primer clasificación entre la máxima eficiencia obtenida utilizando una red neuronal y la SVM.

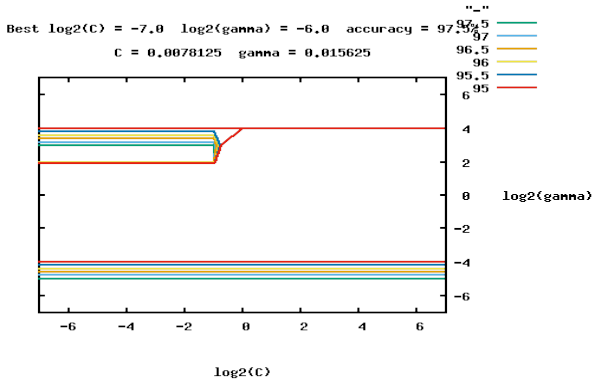
Segunda clasificación		
Color	% predicción máxima con redes	% predicción con la SVM
Azul	80 (CNN)	80
Verde	70 (RNA)	70
Rojo	100(CNN)	80

Tabla 4.8: Comparación de resultados para la segunda clasificación entre la máxima eficiencia obtenida utilizando una red neuronal y la SVM.

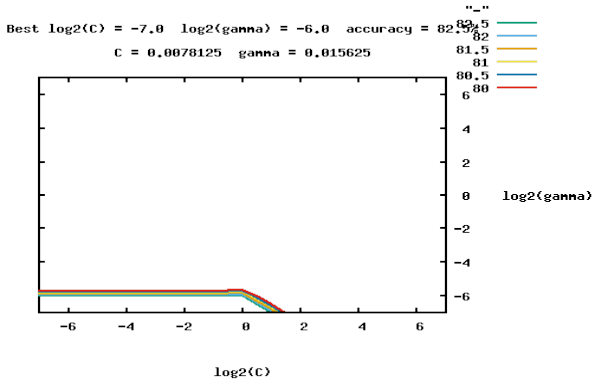
Para decidir cual método es mejor, entre una red neuronal y una SVM, es más complicado ya que obtenemos una clasificación parecida, sin embargo, si consideramos el tiempo de cómputo para el *cross-validation* o para el entrenamiento, es donde la SVM toma ventaja al requerirse un tiempo mucho menor para ambos procesos. Otra ventaja importante de utilizar la SVM

es que al trabajar con los ejecutables de la biblioteca LIBSVM los algoritmos pueden ser aplicados no solo por python, sino también por java o MatLab. Bajo estos criterios se eligió a la SVM como la mejor opción para tratar este problema.

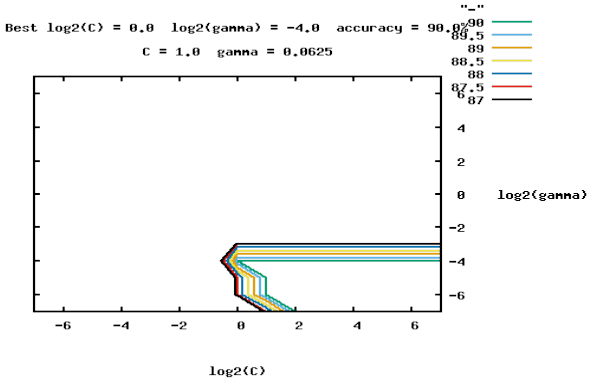
Finalmente, una vez desarrollada la parte computacional para la clasificación se desarrolló un modelo en 3D para el diseño mostrado en la Fig. 4.30, con el cual se pretende tomar mas muestras de forma eficiente para mejorar los resultados sobre los algoritmos. Este diseño es requerido para la impresión en 3D, el cual cuenta con un soporte para la rejilla y un soporte para el led en forma de cubo, así mismo fue diseñado bajo las mismas medidas con las cuales se realizo el experimento en el capítulo 4.



(a)

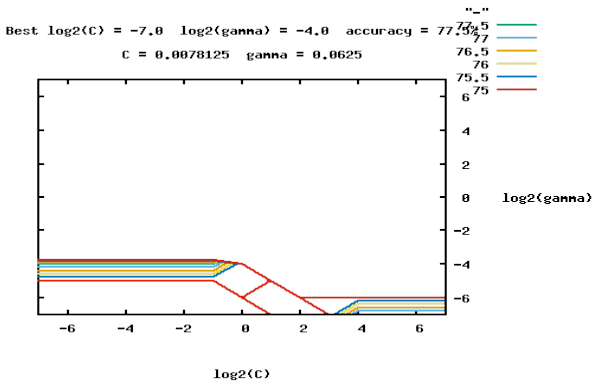


(b)

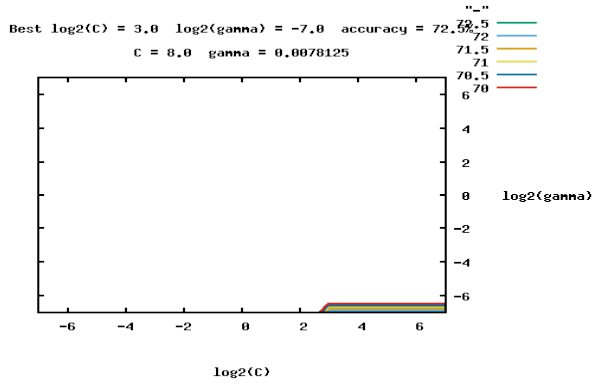


(c)

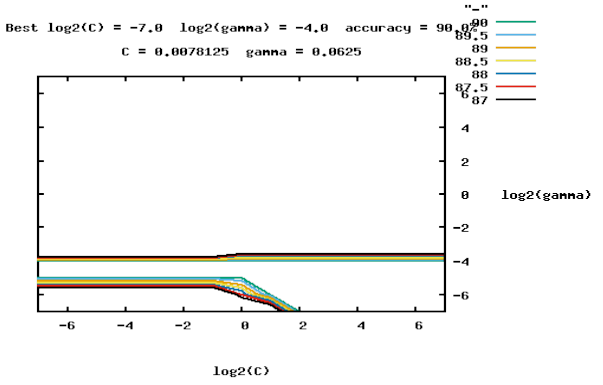
Fig. 4.26: Curvas de nivel del *cross-validation* obtenidas para una SVM para la clasificación de las imágenes en su primera clasificación para los colores azul (a), verde (b) y rojo (c). Cada color en la gráfica representa un distinto porcentaje de aciertos para una cierta configuración de C y γ.



(a)

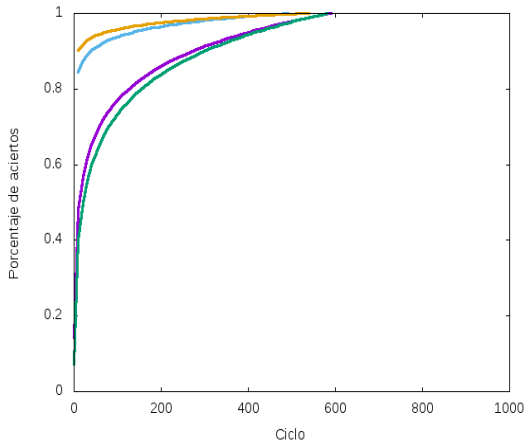


(b)

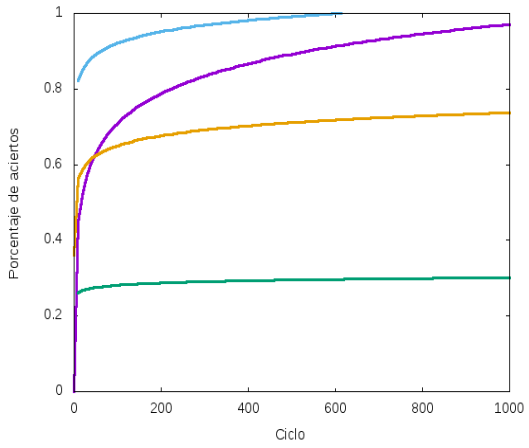


(c)

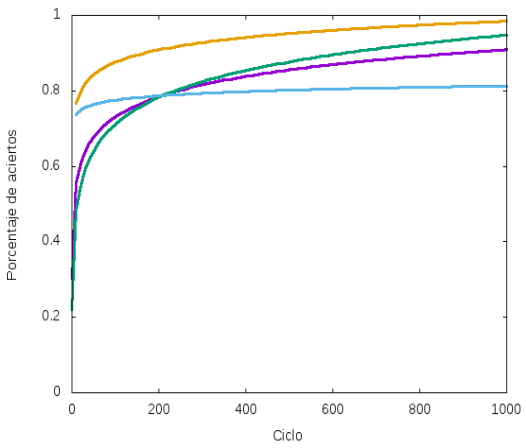
Fig. 4.27: Curvas de nivel del *cross-validation* obtenidas para una SVM para la clasificación de las imágenes en su segunda clasificación para los colores azul (a), verde (b) y rojo (c). Cada color en la gráfica representa un distinto porcentaje de aciertos para una cierta configuración de C y γ.



(a) Ajuste sobre la salida de los algoritmos en la primer clasificación para el color azul.

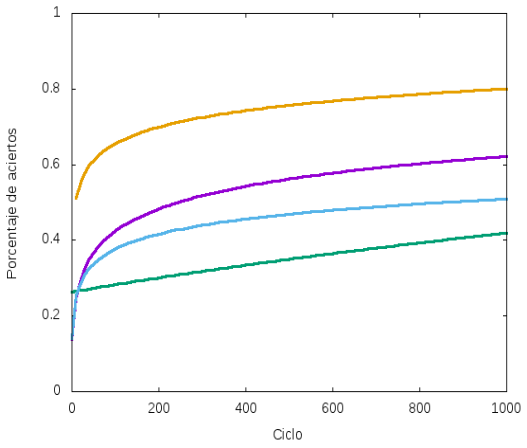


(b) Ajuste sobre la salida de los algoritmos en la primer clasificación para el color verde.

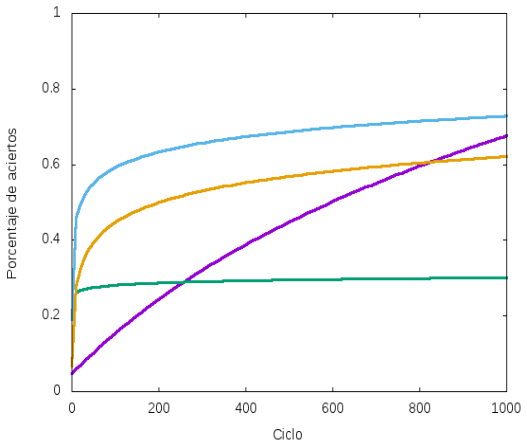


(c) Ajuste sobre la salida de los algoritmos en la primer clasificación para el color rojo.

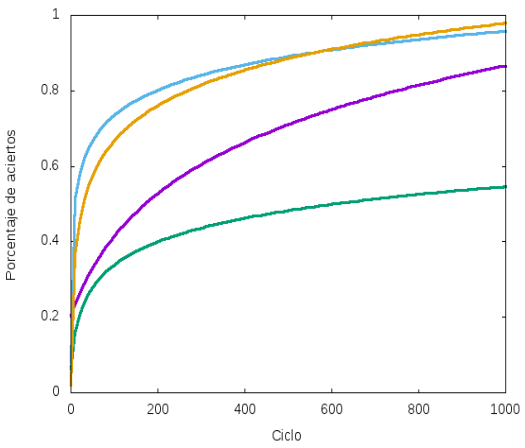
Fig. 4.28: Comparación entre las curvas obtenidas del ajuste sobre la salida de los algoritmos para una RNA, en python con y sin utilizar *TensoFlow*, y una CNN en la primer clasificación.



(a) Ajuste sobre la salida de los algoritmos en la seunda clasificación para el color azul.

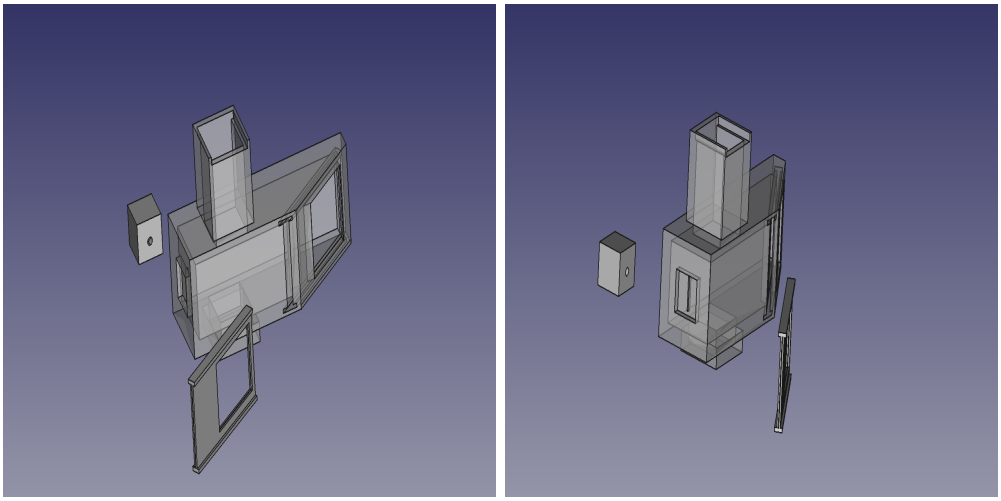


(b) Ajuste sobre la salida de los algoritmos en la segunda clasificación para el color verde.



(c) Ajuste sobre la salida de los algoritmos en la segunda clasificación para el color rojo.

Fig. 4.29: Comparación entre las curvas obtenidas del ajuste sobre la salida de los algoritmos para una RNA, en python con y sin utilizar *TensoFlow*, y una CNN en la segunda clasificación.



(a) Vista lateral del diseño en 3D. (b) Vista frontal del diseño en 3D.

Fig. 4.30: Diseño en 3D para la construcción de un espectrofotómetro en el rango visible utilizando un LED como fuente de luz, una rejilla de difracción y un dispositivo electrónico que puede ser un celular o una cámara para para la obtención del espectro.

Capítulo 5

Conclusiones

En la literatura no se ha realizado al día de hoy un análisis para la determinación de la concentración de una sustancia utilizando inteligencia artificial. Por ello se construyó un dispositivo, mostrado en la Fig. 4.30, que permitirá la obtención de más espectros electromagnéticos de una manera más rápida y eficiente. Una mayor cantidad de muestras ayudará a mejorar los algoritmos para la clasificación al aumentar el número de elementos con los cuales se realizará el proceso de entrenamiento y predicción.

Por otro lado, se pueden obtener mejores resultados en la clasificación si se trabaja con concentraciones espaciadas, como se puede observar en los datos obtenidos por el espectrofotómetro Perkin-Elmer en la Fig. 4.6, las variaciones en el espectro no son tan notorias. Bajo estas variaciones en la concentración la rejilla de difracción, utilizada a lo largo del desarrollo experimental, tiene resultados favorables al poder distinguir, utilizando la cámara en conjunto con los algoritmos de clasificación, los espectros correspondientes por color y para la primera clasificación sobre la concentración. Sin embargo, los resultados para la segunda clasificación no fueron los mejores. Esto puede deberse a que la sensibilidad de la cámara no es suficiente para distin-

guir en el intervalo de variaciones correspondientes a la segunda clasificación e implica que se debe modificar algo en la variación de la concentración o en el arreglo experimental para lograr que la cámara sea capaz de mostrar variaciones significativas para obtener una buena clasificación.

Una modificación que puede llevar a obtener mejores resultados puede ser la construcción de una rejilla con una frecuencia mayor para aumentar el tamaño del espectro sin tener una pérdida significativa de nitidez en las imágenes. Dado que la limitante principal sobre la rejilla es la función pupila de la cámara y considerando los resultados descritos en el capítulo 4, el espectro obtenido podría ser hasta tres veces más grande permaneciendo dentro de la función pupila. Con esta modificación, la cámara, al recibir un espectro más amplio, proporcionará mayor información de los espectros a los algoritmos.

Es importante mencionar que la eficiencia mínima requerida por las industrias para comprar un equipo a nivel global es de aproximadamente 98%, eficiencia que casi se obtuvo para todos los colores en la primera clasificación por concenración. Al poder obtener espectros de forma rápida y eficiente, utilizando el dispositivo diseñado, también se podrían realizar una mayor variación en los ordenes de concentración para llevar al límite los algortimos y determinar en que variación de concentración obtenemos los mejores resultados para cada algoritmo. Por otro lado una combinación de algoritmos para la clasificación también podría ser la solución para llegar a la eficiencia mínima requerida para comercializar nuestro diseño de espectrofotómetro. Con base en los resultados mostrados en el capítulo 4 no cabe duda que los algoritmos desarrollados serán suficientes para resolver este problema de clasificación.

Referencias

[1] Litter, M. I.; Armienta, M. A.; Farías, S. S. *Metodologías analíticas para la determinación y especiación de arsénico en aguas y suelos.* IBEROARSEN, CYTED, Buenos Aires, Spanish, 2009.

[2] Perkampus, Heinz-Helmut. *UV-VIS Spectroscopy and its Applications.* Berlin: Springer-Verlag, 1992.

[3] Bishop, Christopher. *Pattern Recognition and Machine Learning.* 2007.

[4] Schalkoff, Robert J. *Artificial neural networks.* McGraw-Hill Higher Education, 1997.

[5] Lipson. *Optics,* 4th. International edition, Addison-Wesley, San Francisco, 2002, vol. 3.

[6] Palmer, Christopher A.; Loewen, Erwin G. *Diffraction grating handbook.* Springfield, Ohio, USA: Newport Corporation, 2005.

[7] Hilera González, José Ramón, et al. *Redes neuronales artificiales: fundamentos, modelos y aplicaciones.* 2000.

[8] Michie, Donald; Spiegelhalter, David J.; Taylor, Charles C. *Machine learning, neural and statistical classification.* 1994.

[9] Matsugu, M., Mori, K., Mitari, Y., & Kaneda, Y. *Subject independent facial expression recognition with robust face detection using a convolutional neural network*. Neural Networks, 2003, vol. 16, no 5-6, p. 555-559.

[10] LeCun, Yann, and Yoshua Bengio. *LECUN, Yann, et al. Convolutional networks for images, speech, and time series.* The handbook of brain theory and neural networks, 1995, vol. 3361, no 10, p. 1995.

[11] Cortes, C., & Vapnik, V. *Support-vector networks.* Machine learning, (1995). 20(3), 273-297.

[12] Smola, Alex J.; Schölkopf, Bernhard. *A tutorial on support vector regression.* Statistics and computing, 2004, vol. 14, no 3, p. 199-222.

[13] V. Vapnik. *The Nature of Statistical Learning Theory* Springer-Verlag,New York,1995.

[14] Abe Shingeo. *TSupport vector machines for pattern classification.* London, Springer, 2005.

[15] Buerges, Christopher JC. *A tutorial on support vector machines for pattern recognition..* Data mining and knowledge discovery, 1998, vol. 2, no 2, p. 121-167.

[16] Chang, Chih-Chung and Lin, Chih-Jen *LIBSVM : A library for support vector machines.* ACM Transactions on Intelligent Systems and Technology, 2011, vol. 2, no 3, p. 27:1–27:27.

[17] Holladay, L. L. *Optical Society of America. Review of Scientific Instruments*, 1926, vol. 12, no 4.

[18] Marquina, José Ernesto, et al. *La metodología de Newton. Ciencias,* 2003, no 070.

[19] Kotler, Philip, et al. *Marketing.* Pearson Higher Education AU, 2015.

[20] Yáñez-Arancibia, Alejandro; Twilley, Robert R.; Lara-Domínguez, Ana Laura. *Los ecosistemas de manglar frente al cambio climático global.* Madera y Bosques, 1998, vol. 4, no 2, p. 3-19.

[21] Floréen, Patrik, et al. *Multicast time maximization in energy constrained wireless networks. En Proceedings of the 2003 joint workshop on Foundations of mobile computing.* ACM, 2003. p. 50-58.

[22] Sharma, Kailash K. *Optics: principles and applications.* Academic Press, 2006.

[23] Reitz, John R.; Milford, Frederick J.; Christy, Robert W. *Foundations of electromagnetic theory.* Addison-Wesley Publishing Company, 2008.

[24] Hecht, Eugene. *Optics,* 4th. International edition, Addison-Wesley, San Francisco, 2002, vol. 3.