Universidad Autonóma de Yucatán

Facultad de Matemáticas

Métodos de ajuste paramétrico e hiperparamétrico de Redes Neuronales con algoritmos biológicamente inspirados

TESIS

Presentada por:

Fernando Javier Aguilar Canto

En opción al título de:

Licenciado en Matemáticas

Asesor:

Dr. Carlos Francisco Brito-Loeza

Mérida, Yucatán, México
2020

A Lizzy

Agradecimientos

Al Dr. Carlos Francisco Brito-Loeza, por ser mi profesor de licenciatura de las asignaturas *Análisis Numérica* y *Optimización Numérica para problemas de Machine Learning y Visión Computacional* así como mi asesor de tesis, acompañándome a lo largo de mis años como estudiante de Matemáticas. A la profesora Dra. Anabel Martín-González, por sus clases en la asignatura *Redes Neuronales Convolucionales*, las cuales sin duda me impulsaron en este proyecto. Al Dr. Jorge Ríos-Martínez, quien me impartió la asignatura libre *Desarrollo de Aplicaciones* y por sus extensa revisión al escrito original. Al Dr. Javier Arturo Díaz Vargas, quien me impartió siete asignaturas, incluyendo dos de Lógica Matemática. A todos mis profesores y maestros de todos los niveles que me han inculcado un poco de su conocimiento que se expresa como piedras que me permitieron construir esta tesis. A lo largo de mis años de educación básica aprendí las raíces de las matemáticas y las fui introduciendo a mi naturalidad. Cada una de estas bases están inscritas en esta tesis, latentes. Algunas de estas piedras también proceden de enseñanzas de personas que no necesariamente fueron del área de matemáticas. A ellos también les agradezco. A Wilberth Paredes, docente de secundaria. I.F Ángel Francisco Koyoc Noh, profesor de Cálculo de bachillerato, fue decisivo para que tomara el rumbo a las matemáticas y me inculcó la intuición del Cálculo. A los doctores y profesores de la licenciatura. Profesores Dr. Didier Adán Solís Gamboa, Dr. Luis Celso Chan Palomo, Ernesto Ordoñez Solís, Juan Garcilazo, Dr. Juan Pablo Navarrete Carrillo, Pedro Sánchez Salazar, Dra. Lucía Belén Gamboa Salazar, Dr.Carlos Jacob Rubio Barrios, Enriqueta Castellanos, Dr. José Alejandro Lara Rodríguez, Dr. Ángel Estrella, Dr. Eric José Ávila-Vales, Dr. Waldemar Barrera, Dr. Arturo Espinosa Romero, James Sarao Cauich, Dr. Juan Carlos Pineda Cortés (neurocientífico con quien entablé diálogo) y Dr. Luis Basto. También agradezco

a mi director de servicio social, Dr. Enrique Rodríguez Balam. A mis compañeros de licenciatura, pilares en momentos cruciales. Asimismo menciono a Salvador Castro por motivos que indicaré en el Prefacio. A mi núcleo familiar cercano: mi hermana, madre Leticia y mi padre Bernabé. Y finalmente quisiera agradecer a mi novia, Carmen Lizeth Hernández Jiménez "*Lizzy*", quien me impulsó desde el momento que la conocí.

Si somos una suma ponderada de los aportes de nuestro contexto, he aquí algunos de los sumandos. Y yo soy sólo la función de activación.

Resumen

Las Redes Neuronales Artificiales están conformadas de uno o varios operadores conocidos como neuronas artificiales que emulan el comportamiento de las neuronas biológicas de acuerdo a diversos modelos de su comportamiento. En particular, el desarrollo de diferentes arquitecturas de redes neuronales convolucionales ha logrado resolver efectivamente distintos problemas computacionalmente difíciles relacionados con la clasificación de grandes bases de datos de imágenes. A pesar de lo anterior, los modelos de neurona artificial disponibles cuentan con importantes diferencias conceptuales con el conocimiento acumulado sobre las neuronas biológicas, por lo que la adición de aspectos biológicos a las neuronas artificiales puede mejorar las soluciones a los problemas previamente resueltos o ampliar el espectro de problemas a resolver.

La presente tesis se centra en ofrecer algoritmos biológicamente inspirados para el ajuste de redes neuronales tanto a nivel paramétrico (aprendizaje) como hiperparamétrico (evolución). Para el nivel hiperparamétrico se utilizaron las ecuaciones de Lotka-Volterra para diseñar un algoritmo de tipo genético que seleccione la configuración de neuronas por capa óptima, simulando la evolución de una presa en presencia de un depredador. De este modo fue posible ajustar las capas densas de una red convolucional para lograr una exactitud de 93.99% sobre el conjunto de prueba de caracteres EMNIST. Se ofrece la demostración de la convergencia del método sugerido, el cual además presenta una complejidad de tipo polinomial.

Para el caso de la optimización paramétrica, tradicionalmente las redes neuronales utilizan métodos basados en el gradiente en lugar de los inspirados en la regla de Hebb, modelo sobre el aprendizaje a nivel neuronal que ha recibido confirmación experimental. Una aplicación directa de la regla de Hebb permite disponer de algoritmos de apren-

dizaje más eficientes, pero menos efectivos. En esta tesis se realiza una hibridación de ambos enfoques para poder efectuar entrenamiento de redes neuronales en tiempo real sin pérdida notoria de la exactitud. Utilizando una red convolucional preentrenada con métodos basados en el gradiente, se extrajeron las características de las imágenes para finalmente aplicar una capa de clasificación con aprendizaje hebbiano. Este procedimiento demostró ser tanto eficiente como efectivo en las pruebas realizadas en tres diferentes bases de datos de imágenes, mostrando ser una opción válida para el aprendizaje en tiempo real de redes neuronales artificiales.

Prefacio

> *If machines can one day excel us in that one important quality [thinking] in which we have believed ourselves to be superior, shall we not then have surrendered that unique superiority to our creations?*
>
> Roger Penrose [201]

Probablemente la pregunta central sobre la Inteligencia Artificial (IA) sea la posibilidad (o no) de concebir una máquina capaz de emular (en su versión radical, de poseer) cualidades que hasta el momento sólo han sido encontradas en nuestra especie [224]. Aunque la pregunta no es nueva, la emersión de la computación en la segunda mitad del siglo XX, la masificación del ordenador personal durante finales del mismo siglo y el auge de la IA (en particular, de las *Redes Neuronales Artificiales*), han popularizado enormemente esta cuestión.

Esta pregunta ha sido respondida negativamente por destacados académicos, como Roger Penrose (recientemente galardonado como Premio Nobel de Física) en sus obras *The Emperor's New Mind* [201] y aún más radical en *Shadows of the Mind* [200]. El argumento central en la obra de Penrose consiste en la no computabilidad existente desde un nivel físico hasta el de la consciencia, basándose en el Teorema de Incompletitud de Gödel. En efecto, el problema sobre si es posible desarrollar una IA-Fuerte (*Strong AI*) tiene implicaciones notables en la filosofía de la mente y de la Epistemología en general.

No obstante, el argumento de Penrose ha sido señalado desde las matemáticas como inválido [141], ya el Teorema de Gödel está formulado para la Teoría de Números, y su aplicabilidad para la consciencia humana es controversial [174]. Posterior a las discusiones teóricas sobre si la IA-Fuerte es imposible, se han formulado impactantes proyectos para

verificar lo contrario. De hecho, de forma análoga a la Conjetura de Goldbach, solamente se puede probar la imposibilidad de la IA-Fuerte desde la teoría pero para afirmar su posibilidad basta únicamente un ejemplo. Dos proyectos en esa dirección son Blue Brain [171], el cual pretende simular neurona tras neurona al cerebro humano utilizando una supercomputadora, y quizá menos ambicioso pero con mejores resultados ha sido SPAUN 1.0 [59, 235] y 2.0 [41], que destaca del uso de redes pulsantes. A pesar de ello, si bien la segunda versión ha incluido reconocimiento de imágenes en RGB, aún no logra superar los modelos de redes convolucionales en cuanto a la tarea cognitiva de reconocer objetos.

No corresponde responder la pregunta abordada desde el primer párrafo en una tesis de licenciatura escrita en un área relativamente remota para el desarrollo tecnológico, en una lengua marginal para el mismo. ¿Por qué, entonces, he iniciado este prefacio con esta discusión? A pesar de tratarse de un tema lejano a los objetivos de la misma, fueron estas incógnitas que me motivaron a tomar esta dirección sobre la tesis y aún más, contribuyó en mi decisión sobre estudiar la licenciatura en Matemáticas. Por ende, he decidido utilizar este espacio para expresar mis motivos y mi perspectiva. No pretendo con ello dar una falsa impresión sobre los alcances de esta tesis, ni siquiera considerar que mi opinión sobre el tópico es *académica*. Simplemente quisiera realizar un recuento sobre las experiencias transcurridas, mis motivaciones, anhelos y reflexiones, así como el papel (ínfimo) de esta tesis en tales problemas.

0.1 Trayectoria personal

Mi primera experiencia con la investigación no fue en las Ciencias Exactas o en alguna Ingeniería. Durante los años oscuros de secundaria y bachillerato aspiraba a convertirme en un profesional de las Ciencias Sociales, a las que consideraba como infravaloradas en el mercado científico. Que yo, Javier Aguilar Canto, en mis años de adolescencia, considerara estudiar Matemáticas habría resultado plenamente impensable. No obstante, siempre permanecí atraído hacia la elegancia y formalidad de las matemáticas, y es de ahí donde planteé, dentro de mi propio mundo, los "Axiomas de la Praeteritulogía", disciplina que se encargaría del pasado humano desde un punto de vista holístico:

1. *Axioma del Tiempo:* El tiempo en la Praeteritulogía ocurre de forma única y lineal.

2. *Axioma Positivista:* Ningún principio físico puede ser violado.

3. *Axioma del Contacto:* Dos *corpus* de elementos se formaron de manera dependiente si y sólo si existió contacto entre los contextos.

4. *Axioma de la Información:* La razón de cambio de la abundancia de información es proporcional al tiempo $\frac{dI}{dt} = \alpha t$.

Particularmente el Axioma 1 me causaba conflicto, pero era necesario para negar la presencia de anacronismos. El Axioma 4 es una formulación actual del tratamiento de la exponencial. No discutiré los detalles de esta modelación, pues no es de interés para esta tesis. Lo que debo mencionar al respecto, es mi curiosidad por formalizar disciplinas caracterizadas por sus métodos cualitativos, lo cual finalmente me llevó a una versión más extrema de mi postura analítica al tratar de formalizar el *todo*. Eso me llevó al *Basicismo*, un intento por comprender cuáles son los enunciados con *certidumbre radical*, aquellos que no se pueden negar sin llegar a una eventual contradicción, las cuales serían las bases del conocimiento. Además de los enunciados *a priori* (que incluye a la totalidad de las ciencias formales), parecen existir dos proposiciones de carácter empírico que no se pueden negar sin derivar en una contradicción: *existe algo* y *existo yo*. La demostración del segundo enunciado también puede lograrse mediante la sintética y genial prueba ofrecida por René Descartes: *Cogito Ergo Sum*, donde el *ergo* debe ser entendido como una implicación $\Rightarrow$.

¿De qué carácter son estas demostraciones sobre la realidad empírica? El hecho que discutamos sobre la realidad misma nos ofrece pruebas sobre la existencia de ciertos entes como algo, el yo, las preguntas, el lenguaje, etcétera, pero no permite probar la existencia de la *res extensa* cartesiana, incluyendo al Otro. Eso lleva a la formulación de teorías alternativas acerca de la realidad, por ejemplo, asumir la existencia de la *res extensa* (que es la hipótesis natural), o bien, negarla dando origen a posturas extrañas como el solipsismo.

Es así como, entre discusiones filosóficas y estudios en ciencias sociales concretas como la Crítica Textual o la Historia, fue como a finales del 2015 presenté mi primera ponencia

llamada *Las designaciones mayas de Mérida y su relación con el glifo emblema de Dzibilchaltún desde un enfoque filológico* en el centro INAH regional, donde expuse algunos resultados que en su momento creí interesantes, pero la recepción de los mismos no fue del todo cálida. Si bien haber presentado una ponencia en un contexto importante debió haber consolidado mi camino hacia las ciencias sociales, lo cierto es que de algún modo me inclinó hacia la búsqueda de nuevas rutas. No fue hasta meses más tarde cuando presentaría avances en la Crítica Textual de los libros del *Chilam Balam* cuando consideraría que haría aportaciones reales, pero hasta la fecha no he logrado publicar los resultados en una revista apropiada[1].

A inicios del 2016, fui aconsejado por Salvador Castro y Ángel Campos de abandonar a las ciencias sociales, dirigirme hacia las ciencias exactas y aprender a programar. La iniciativa, que en un principio me pareció inapropiada, finalmente terminó por ser considerada. Para entonces, habían transcurrido cuatro años de la publicación de la AlexNet de Krizhevsky [133] y dos de la aparición de redes más profundas como la VGG19 de Simonyan [228] y la GoogleNet [239] de Szegedy, las cuales marcaron de alguna manera el auge de las redes neuronales convolucionales al resolver el problema de la visión[2]. Sin embargo, la razón principal por la cual decidí efectuar tal radical cambio fue teórica y sólo se abordará hasta la última sección de este párrafo.

Una vez concretado el cambio vocacional, habría de iniciar mis planteamientos que en un inicio serían meramente especulatorios, pero me permitieron resolver problemas de una forma que alcanzara mi entendimiento. Eso me llevaría, junto con mi equipo conformado por los mencionados compañeros (a los que se restaría Castro y después Campos), a participar en los eventos Feria Nacional de Ciencias e Ingenierías, Expociencias Yucatán y posteriormente a eventos de mayor envergadura.

De esta forma, mi primer intento por realizar un programa capaz de reconocer patrones visuales (que participó en la Feria Nacional de Ciencias e Ingenierías) consistía simplemente en efectuar una resta de matrices en valor absoluto. Por ejemplo, supong-

[1]Queda pendiente esta publicación.

[2]Por *problema de la visión* designamos al problema de clasificar grandes cantidades de imágenes. Tal logro, alcanzado gradualmente por los avances de las redes convolucionales, se refiere básicamente a poder disponer de un algoritmo efectivo que logre realizar esta tarea, algo que no se disponía hasta el 2012.

amos que deseamos reconocer un dígito 1, el cual matricialmente puede aparecer de la siguiente forma:

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{1}$$

El anterior es el 1 perfecto, el 1 platónico o quizá el 1 promedio representado en una matriz de 5×5. Los unos reales, naturalmente, se asemejan ligeramente a ese uno expresado que para términos prácticos debe encontrarse el centroide de una componente conexa de la imagen, recortarse en forma de cuadrado y escalarse evitando disponer una matriz completamente de unos. Un uno real, es, en efecto, una matriz como la que sigue:

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{2}$$

Si efectuamos el valor absoluto de la diferencia y promediamos obtenemos $\frac{3}{5\times5} = 0.12$. Si comparamos con otros patrones quizá podamos identificarlo con la clase correcta. Este método no es el más efectivo ni tampoco es novedoso, pero al menos podía entenderlo. Una mejora de este método consistía en agregar a su base de conocimiento un nuevo patrón por cada error que cometiera y reconocerlo como la clase debida.

Naturalmente eso exigía utilizar matrices de dimensiones no tan grandes para evitar una explosión combinatoria, por lo que pronto encontré problemas para su implementación práctica. No obstante, a pesar de ser poco aplicable a clases con alta varianza, fue posible utilizarlo para reconocimiento de señales fijas en contextos controlados. Originalmente se pensó en utilizar este algoritmo para el desarrollo de herramientas para invidentes, el cual en conjunto con un sistema de sensores ultrasónicos en una Raspberry Pi permitían disponer al usuario de un equipo para la navegación.

No obstante, actualmente pienso que habría sido de difícil aplicabilidad, ya que las señales de deberían estar en frente del usuario y no lateralmente. A pesar de sus limitaciones, el sistema funcionaba apropiadamente al presentarse y emitía un sonido de un buzzer que tal vez lo hizo llamativo ante el jurado. Puede resultar poco creíble, pero ese sistema simple me llevó a participar en la MILSET International ExpoSciences 2017 [3].

A estos prematuros éxitos debidos a una combinación de suerte y entusiasmo siguió un hiato dado por nuevos reveses en posteriores eventos. En este punto ya era consciente de las dificultades que presentaba el reconocimiento de imágenes por métodos ingenuos. Sin embargo, no volvería a tocar el tema activamente hasta el 2019, fecha de inicio de la investigación centrada en la regla de Hebb y las redes convolucionales.

0.2 Sobre esta tesis

En el 2019, tras haber abandonado mis anteriores trabajos en Lógica Matemática, inicié mis prácticas profesionales con el PhD Carlos Francisco Brito-Loeza, quien a la postre se volvería mi asesor de tesis. Inicialmente me centré en la llamada regla de Hebb (que se aborda en un capítulo de esta tesis), concretándose en un artículo finalmente publicado en *Eficacia de diferentes reglas hebbianas en el Aprendizaje Supervisado* [6], donde se observa una aplicación sencilla de la regla de Hebb. Este trabajo fue presentado en Puebla en el marco del XXXII Congreso Nacional y XVIII Congreso Internacional de Informática y Computación de la ANIEI. Posteriores ponencias derivadas del trabajo de Prácticas Profesionales fueron *Resultados teóricos sobre Clasificación Binaria mediante el empleo de Reglas de Hebb* en el 52 Congreso Nacional de la Sociedad Matemática Mexicana celebrado en Monterrey, *La Regla del Descenso como regla de Hebb débil para entrenamiento de redes neuronales* y *Entrenamiento de redes neuronales mediante reglas de Hebb para Online Machine Learning*, dadas en Mérida.

Finalizado el 2019, el trabajo concretamente de tesis se centró en dos aspectos principales: disponer de un mecanismo de ajuste para hiperparámetros y posibilitar el aprendizaje en tiempo real de imágenes, para lo cual el uso de redes convolucionales era re-

[3]Este evento no estuvo libre de polémica, véase el caso de William Gadoury.

querido. Estos dos objetivos se concretarían en las ponencias *Evolutionary Neural Networks with Lotka-Volterra models*[4] en el marco del Congreso Nacional de Inteligencia Artificial y *Convolutional Neural Networks with Hebbian-Based rules in Online Transfer Learning* [5] presentado en el Mexican International Conference on Artificial Intelligence, ambos eventos orquestados por la Sociedad Mexicana de Inteligencia Artificial.

0.2.1 Hacia una Semántica natural

El segundo objetivo mencionado encapsula la línea original que deseaba trabajar durante la licenciatura, puesto que desde el 2016 había considerado centrarme en el problema de la visión (*large image recognition*), el cual, sin embargo, ya había empezado a resolverse en gran medida por el avance de las redes profundas desde el 2012. No obstante, como se revisará, el esquema de aprendizaje de las redes convolucionales es diferente al aprendizaje en contextos naturales. Mientras que el entrenamiento usual de redes neuronales consiste en definir una función de costo específica y minimizarla sobre el conjunto de entrenamiento (y con ello reducir el error de clasificación), el aprendizaje natural se realiza en tiempo real, mediante la asociación de estímulos visuales con auditivos (véase figuras 1 y 2.)

El primer caso de entrenamiento se distingue por disponer de un método específico consistente en la recolección de n datos (necesariamente finitos), para finalmente efectuar una optimización a varias épocas hasta lograr un error aceptable y a la postre lograr el reconocimiento en tiempo real. Estos entrenamientos son usualmente costosos computacionalmente y en muchos casos solamente pueden ser realizados mediante supercomputadoras. Un entrenamiento más real debería efectuarse mediante la dicha asociación entre la información procedente de dos canales, presentando el estímulo visual y auditivo simultáneamente durante numerosas ocasiones hasta lograr el aprendizaje, por ejemplo.

En su obra de Neurosemántica, Plebe y Vivian [203] exponen un modelo jerárquico de procesamiento visual y auditivo hasta llegar a la asociación de conceptos en la Corteza Prefrontal. Como se verá, estas jerarquías han sido modeladas por medio de las capas convolucionales. De este modo, los objetos concretos o sustantivos forman parte de la

[4]En vías de publicación

[5]Publicado en [5]

Figure 1: Esquema de entrenamiento estándar de redes neuronales consistente de n datos.

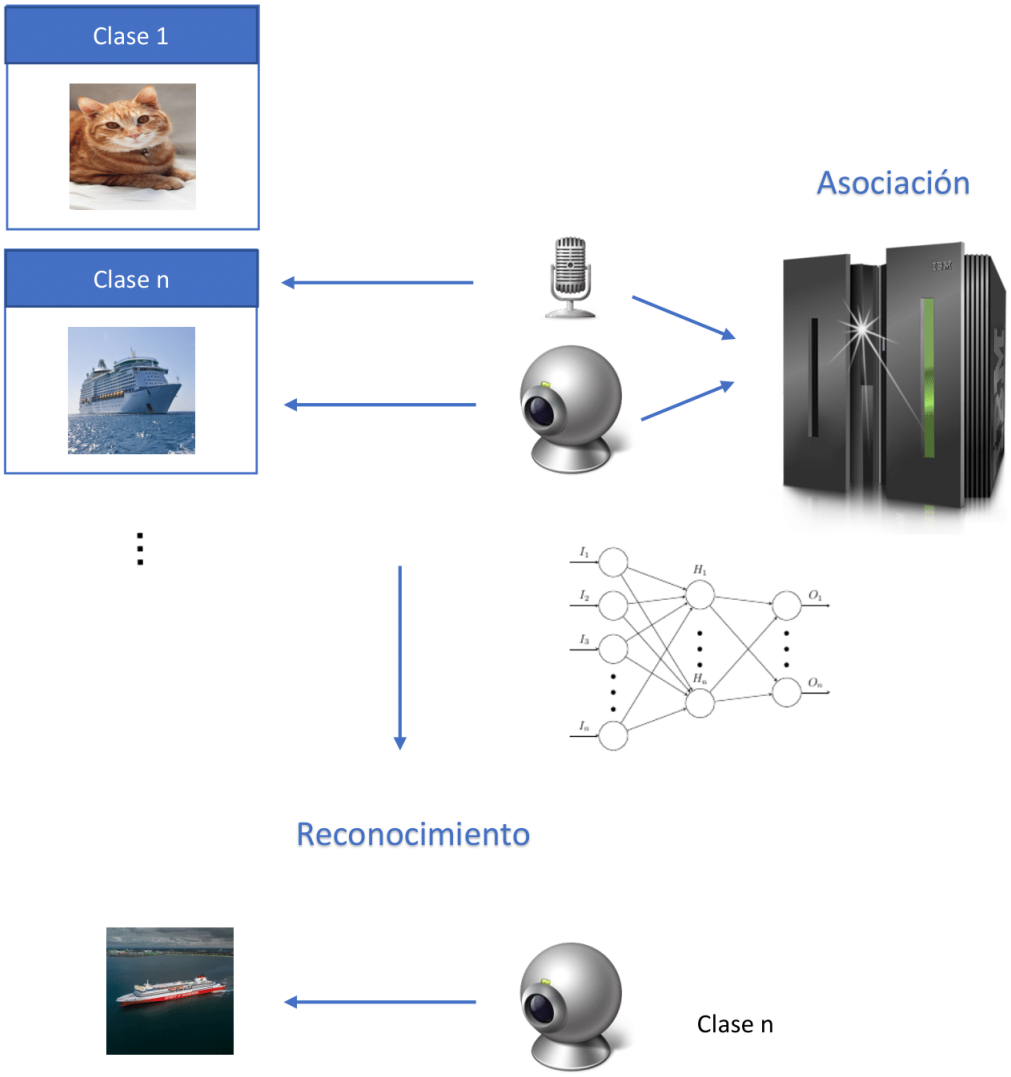


Figure 2: Esquema de entrenamiento de redes neuronales en tiempo real.

primera semántica de los infantes, consistente de 51 a 200 palabras. Por lo tanto, lograr una asociación artificial entre imágenes y etiquetas (o bien, sonidos) consistiría un importante avance para la formulación de una semántica artificial, un avance hacia el desarrollo de la comprensión del significado en redes neuronales, que consiste en un objetivo primario de la tesis. Sin embargo, se trata de apenas uno de los pasos necesarios para la modelación de un sistema aún más complejo. Estos objetivos serán atacados en las futuras investigaciones que se realicen.

0.2.2 Notas sobre el año de desarrollo de la tesis

Esta tesis fue escrita a lo largo del 2020, coincidiendo con la pandemia del COVID-2 por el SARS-CoV-2 perteneciente a la familia de los β-coronavirus. Por tal motivo este año es referido a lo largo de la redacción de la tesis. Sin embargo, es posible que por cuestiones de la publicación de la tesis, el año que se refiera sea el 2021, a pesar de que este texto haya sido redactado en noviembre de 2020, figurando como la última etapa de redacción.

0.3 Sobre la posibilidad teórica de la Inteligencia Artificial fuerte y sus consecuencias

El problema de la posibilidad de la IA-fuerte atraviesa numerosas disciplinas, pero ante todo es una cuestión filosófica. Al inicio del Prefacio se presentaron los argumentos en contra formulados por la imponente figura de Penrose. Lejos de intentar contraargumentar a Penrose, tarea que se han encargado otros matemáticos, intentaré ofrecer un argumento a favor y a examinar sus consecuencias en orden teórico.

¿Hasta qué punto la consciencia podría ser considerada computable? Quizá no se pueda responder esta pregunta directamente, puesto que no es fácil aplicar la computabilidad a algo tan difuso como la consciencia. Esta pregunta, que también se ha extrapolado a otros contextos como la física, requiere de una reformulación. Del mismo modo se aplican los predicados decidible o incompleto, que sólo tienen sentido en teorías matemáticas.

Para lograr extender los predicados enlistados a aspectos no matemáticos como la

física, la consciencia o biología, es preciso disponer de un modelo de la misma, que derive en una teoría que disponga del mismo sentido que las teorías matemáticas: un lenguaje así como de un *corpus* de axiomas en tal lenguaje. Hasta el momento no se disponen de axiomas relativos a la consciencia, mientras que las formulaciones que se disponen forman parte de la Neurociencia Teórica, aunque no de forma totalmente axiomática. Lo que tal vez podría funcionar como un ejemplo útil es el caso de la Física, cuyas axiomatizaciones han sido formuladas en distintas obras y constituyen una solución al Sexto Problema de Hilbert.

Este camino de tratar de describir a la consciencia o las capacidades humanas mediante un sistema axiomático no es seguido por Penrose. En su lugar, siguiendo el estilo de Penrose, podemos tratar de definir a una consciencia en particular, llámese $\mathcal{M}$, propia de un matemático capaz de demostrar teoremas. Entonces, si existe una máquina capaz de emular el funcionamiento de $\mathcal{M}$, dispondría de un *algoritmo de demostración* para todas las teorías, lo cual contradice el Primer Teorema de Incompletitud de Gödel, el cual indica lo siguiente (versión dada en [234]):

Teorema I (Primer Teorema de Incompletitud de Gödel)**.** *Toda extensión axiomatizada y consistente de la Teoría de Números es indecidible y por lo tanto, incompleta.*

Una Teoría $\mathcal{T}$ es decidible si existe un procedimiento efectivo para decidir si un enunciado de la teoría es consecuencia de sus axiomas o no. Por lo tanto, de disponer a la consciencia $\mathcal{C}$, en principio sin las necesidades fisiológicas del matemático, podría demostrar los enunciados de la Teoría de Números a manera de algoritmo, por lo que la Teoría de Números sería decidible en dicho caso.

Lo anterior parece mostrar la imposibilidad no sólo de la demostración automática sino de cualquier algoritmo que emule la actividad humana. No obstante, debemos entender claramente qué es lo que se supone que es algoritmo en estos contextos, cuya definición viene dada por Turing, Church y el mismo Gödel utilizando las máquinas de Turing, el Cálculo λ y las funciones recursivas respectivamente. Lo que mencionaremos preliminarmente sobre los algoritmos es que cuentan con propiedades específicas, como contar con un principio y fin[6].

[6]Para ver una definición explícita de las funciones recursivas, consúltese [234].

0.3.1 Una expansión del concepto de algoritmo

Las cualidades de los algoritmos hacen que esta definición original sea un tanto rígida permitiendo opciones más reducidas. Es claro que las personas no estamos restringidas a los algoritmos, ya que no todo lo realizamos mecánicamente y el ejemplo anterior muestra que efectivamente tenemos más opciones que la mecanicidad. Lo que es realmente sorprendente es el hecho que las máquinas tampoco están del todo limitadas a la noción de algoritmo, si bien la tesis de Church-Turing nos indica la equivalencia entre computabilidad y la noción de algoritmo.

Un ejemplo simple de lo anteriormente dicho sería un bucle infinito: abrir un programa en Python con la línea `while True:` y no darle una sentencia de terminación, o bien, que no se cumpla nunca. Los semialgoritmos (análogos al concepto de semirrecursividad) tienen esa característica: si escribimos un programa que trate de verificar la Conjetura de Goldbach posiblemente nunca termine, ya que únicamente finaliza en caso de que esta conjetura sea falsa, pero no puede acabar en caso contrario. Es notorio que matemáticos de todos los tiempos hayan intentando resolver este problema, sin hallar éxito.

Por ende, hablar de que las computadoras están limitadas al concepto de algoritmo resulta poco prudente, aunque en general se estimula a que los programadores escriban códigos que tengan principio y fin. A pesar de ello, considerar a los semialgoritmos como parte integral de la computación muestra ventajas que no se disponían previamente. Una de las mismas es la posibilidad de implementar un semialgoritmo de demostración. Para ello introduciremos las siguientes nociones:

Definición 1. *Una teoría $\mathcal{T}$ es semidecidible si existe un procedimiento semirrecursivo para verificar si $\mathcal{T} \vdash \sigma$ para todo enunciado σ.*

Definición 2. *Una teoría $\mathcal{T}$ es casi seguramente semidecidible si para cada enunciado σ de la teoría la probabilidad de verificar $\mathcal{T} \vdash \sigma$ converge a 1 cuando $n \to \infty$ cuando $\mathcal{T} \vdash \sigma$ se cumpla.*

La notación $\mathcal{T} \vdash \sigma$ significa que la teoría $\mathcal{T}$ demuestra σ, esto es, que existe una demostración de σ a partir de la teoría $\mathcal{T}$ (véase [64]). La última noción nos indica que podemos disponer de un semialgoritmo que se ejecute sobre todos los naturales y

si disponemos de los recursos computacionales suficientes podremos verificar cualquier teorema. Este término se asemeja al Teorema del Mono Infinito, puesto que se tiene un semialgoritmo que casi seguramente genera un texto arbitrario, como puede ser una obra de la literatura universal como lo es *Pedro Páramo*. De la misma manera podemos generar una demostración:

Proposición I. *Toda teoría $\mathcal{T}$ con lenguaje finito es casi seguramente semidecidible.*

Demostración. Sea σ un enunciado de $\mathcal{T}$. Consideremos todos los axiomas Σ_0 de la teoría $\mathcal{T}$ (unión los axiomas lógicos). Supongamos que $\mathcal{T} \vdash \sigma$. Entonces existe una secuencia finita de símbolos $\tau_1\tau_2\cdots\tau_m$ que expresa la demostración de σ. Por el Teorema del Mono Infinito, es posible generar la secuencia $\tau_1\tau_2\cdots\tau_m$ casi seguramente. Como podemos decidir si $\tau_1\tau_2\cdots\tau_m$ demuestra σ por medio de una función recursiva, entonces cuando $\mathcal{T} \vdash \sigma$ se cumple terminamos. □

Nótese aquí que aunque la anterior proposición vale para teorías con lenguajes finitos, dado que utilizamos un número finito de símbolos de nuestro lenguaje para expresar símbolos de la teoría (como x_i), podemos generalizar esta consecuencia del Teorema del Mono Infinito a cualquier teoría matemática.

No solamente las malas prácticas de programación nos proveen ejemplos sobre no-algoritmos que se pueden ejecutar desde una computadora. Los procesos estocásticos son en general no recursivos, pero se han encontrado formas de disponer una aproximación al azar y con ello simular aleatoriedad. Si los módulos `random` corresponden a una aleatoriedad real o no lo son, rebasa los objetivos de este ensayo, aunque de forma inicial asumiremos el desconocimiento de las semillas utilizadas para emular este procedimiento.

En la Proposición 1, hemos hecho uso tanto de la semirrecursión como de la aleatoriedad. Tal como el Teorema del Mono Infinito, es posible programar un semialgoritmo demostrativo siguiendo esa línea. No obstante, el procedimiento descrito es muy poco óptimo, por lo que para fines prácticos no representa un mayor avance en la Teoría de la Demostración, al menos no hasta contar con computadoras altamente eficientes. Sin embargo, el hecho de que lo anterior pueda programarse nos muestra que los alcances de la noción clásica de computabilidad es sumamente limitada en comparación de lo que en

realidad pueden efectuar las computadoras.

Del mismo modo que la implementación sugerida es impráctica, también existen algoritmos (de clase NP) cuya ejecución es costosa computacionalmente, por lo que los límites de cómputo también afectan a las nociones clásicas. Lo que es importante destacar es el hecho de que al menos se dispone de un procedimiento *programable* para lograr la demostración de los teoremas que se supongan verdaderos, por lo que la existencia de un procedimiento más óptimo no sería extraña.

En efecto, la demostración realizada por matemáticos humanos no se realiza a través de algoritmos fijos en general (algo que se ha probado imposible), sino que siguen heurísticas y diversas técnicas, pero también parece estar mediado por la creatividad y la aleatoriedad. Del mismo modo que Alexander Flemming descubrió la penicilina, el proceso por el cual un científico adquiere una idea valiosa puede tratarse de un proceso estocástico.

Asimismo, no se ha conocido un matemático que pueda demostrar absolutamente cualquier conjetura que se presente. Eso significa que el proceso de demostración tiene un principio, pero no necesariamente un fin: durante dos mil años los geómetras intentaron probar el V Postulado de Euclides sin llegar a algún resultado suficiente. Esto ocurre si dejamos correr un semialgoritmo *ad infinitum* para probar un resultado que es intrínsecamente falso.

Todo lo anterior nos da fuertes atisbos de que el *modus operandi* de un matemático es más próximo a la noción de semialgoritmo de tipo probabilístico, que a un algoritmo determinista. Dado que es posible programar un semialgoritmo estocástico, no existe contradicción en el argumento que presentamos utilizando el Primer Teorema de Incompletitud.

0.3.2 Programabilidad de la IA-Fuerte

La generalización de la programabilidad hacia el cómputo no determinista y estocástico no es suficiente para hablar de que ciertos modelos teóricos son expresables en una computadora. Las funciones recursivas toman la forma $f : \mathbb{N}^k \to \mathbb{N}^l$, por lo que en sentido estricto muchas de las funciones utilizadas en los modelos no son computables. Un ejemplo de ello es la función sigmoide $\sigma : \mathbb{R} \to \mathbb{R}$, puesto que los números reales no pueden ser expresados

en su totalidad por las computadoras.

El problema de la función sigmoide es relevante, ya que existen modelos de redes neuronales que utilizan a dicha función como activación de las neuronas (véase el capítulo de Redes Neuronales). En general, las funciones de activación están definidas sobre los reales, por lo que ningún modelo de red neuronal es, en *stricto sensu*, computable, ni siquiera para los modelos más básicos de redes neuronales artificiales.

Lo mencionado previamente es contraintuitivo y aún más extraño que en el caso de los semialgoritmos estocásticos, puesto que si bien no es común escribir programas sin término, abundan los códigos de redes neuronales que utilizan funciones de activación. No dudaríamos que tales códigos sean algoritmos, evidenciando aún más la limitada definición original de los mismos, dada incluso antes de la computación misma.

Por lo tanto, incluir las aproximaciones discretas a funciones de variable real parece un paso necesario para añadir al concepto de *programabilidad*. No sólo tales funciones pueden aproximarse, sino que también las ecuaciones diferenciales con el método de Euler y las integrales con la regla de Simpson, por ejemplo. Esto nos da un gran arsenal de herramientas matemáticas que se pueden expresar de manera aproximada utilizando computadoras.

Habiendo extendido el concepto de computabilidad al de programabilidad, podemos hacer uso de este concepto para tratar de reformular las preguntas iniciales. De este modo, la programabilidad aborda aquello que es posible de ejecutar en una computadora, incluso a nivel aproximado. ¿Qué tanto se sacrifica al realizar aproximaciones de los modelos? Si tales aproximaciones son lo suficientemente exactas podríamos utilizarlas sin temor a obtener resultados extraños. La naturaleza también parece ser inexacta en algún grado minúsculo, y aunque dicha inexactitud puede deberse al error de medición, pequeñas perturbaciones pueden estar presentes, pero incluso el hecho de que la naturaleza tenga un carácter continuo es discutible.

Un ejemplo simple de programabilidad puede manifestarse en leyes de la Mecánica Clásica como es el caso de las Leyes de Newton, las cuales pueden ser implementadas para simular el movimiento en entornos artificiales. Esto no es nuevo, puesto que los videojuegos suelen implementar (o variar) tales leyes para recrear escenarios realistas. La

posibilidad de extender este concepto a la Mecánica Cuántica y Relativista queda fuera del alcance de este Prefacio, pero en caso afirmativo teóricamente sería posible simular computacionalmente cualquier otro aspecto de la realidad (si todo tiene un sustrato físico).

¿Es posible aplicar el concepto de programabilidad al sistema nervioso humano, y por extensión, a la *consciencia*? No es fácil responder esta pregunta careciendo de un modelo completo del mismo. Sin embargo, los modelos que se disponen de neuronas en general lo son, ya que desde el modelo de Hodgkin-Huxley hasta las simplificaciones extremas como el modelo de tasa de disparo están dados como ecuaciones diferenciales, y lo mismo sucede para las reglas de aprendizaje. Ya que el sistema nervioso está conformado por estas unidades programables, en general también el mismo sistema es programable en este sentido, lo cual a pesar de su dificultad, es el objetivo de proyectos como Blue Brain.

¿Eso concluye que la consciencia es programable, en el sentido no determinista estocástico? La respuesta a esta pregunta es parcial. Carecemos de un modelo matemático preciso sobre el comportamiento de agentes como los neurotransmisores [98]. Por lo tanto, la respuesta a esta gran pregunta permanecerá en blanco hasta disponer de modelos más completos sobre el sistema nervioso. ¿Qué tan simples son los modelos que actualmente se disponen? Sin consideración de múltiples aspectos como los neurotransmisores, células gliales y otros que quizá ignoremos, tal vez sea poco probable emular algo tan complejo como la consciencia humana. Pero el precio de tal simplicidad está por verse.

0.3.3 Conclusiones

No hemos podido responder si es posible o no la IA-Fuerte ante la ausencia de modelos completos de la naturaleza humana. Los modelos disponibles son programables, pero al ser simplificaciones podrían no abstraer ciertas características que pudiesen estar relacionadas con las habilidades cognitivas superiores.

No obstante, se han logrado reproducir algunas de las llamadas habilidades cognitivas utilizando solamente el modelo de red neuronal artificial (muy simple), como es el caso del reconocimiento de objetos. Esto significa que modelos ligeramente inspirados pueden capturar algunas de estas cualidades incluso utilizando computadoras actuales. Eso nos da atisbos sobre su eventual posibilidad.

Índice general

Índice de cuadros

Índice de figuras

Capítulo 1

Introducción

I propose to consider the question, 'Can machines think?'

Alan Turing [247]

¿Qué tan *artificial* es lo que actualmente conocemos como *Inteligencia Artificial* (IA)? A 70 años de la publicación del artículo clásico de Alan Turing, *Computing Machinery and Intelligence* [247], no se ha podido dar una solución totalmente efectiva a la pregunta sobre si las máquinas podrían poseer enteramente cualidades humanas consideradas superiores como pensar, analizar, entre otros aspectos[1]. Esta rama de las ciencias computacionales, si se puede considerar de este modo, es tan antigua como las computación misma: la primera propuesta de redes neuronales artificiales aparece en 1943 [173], dando origen a esta área.

El término de Inteligencia Artificial parece estar ligado a la elaboración de programas que emulen procesos inteligentes. Una consulta de lo que significa "inteligencia" en español (el cual no es el idioma principal en este tópico, pero que comparte la misma raíz latina *intelligentia*) arroja las siguientes acepciones, de acuerdo con el Diccionario de la Real Academia de Lengua Española [67]:

1. Capacidad de entender o comprender.

2. Capacidad de resolver problemas.

3. Conocimiento, comprensión, acto de entender.

[1]Los *chatbots* más modernos aún no son totalmente conversacionales de acuerdo con [250]

4. Sentido en que se puede tomar una sentencia, un dicho o una expresión.

5. Habilidad, destreza y experiencia.

Las últimas acepciones fueron omitidas por apartarse del sentido que nos interesa. Es interesante notar que este diccionario de 1970 incluya una definición de Inteligencia Artificial, la cual está dada por: "Desarrollo y utilización de ordenadores con los que se intenta reproducir los procesos de la inteligencia humana"[2].

Por lo tanto, una primera aproximación del concepto es simplemente sobre tratar de reproducir mediante un programa un proceso propio de la inteligencia humana, entre los que se involucra la comprensión, la resolución de problemas, entre muchos otros aspectos. Esto involucra la capacidad de programar máquinas con cualidades atribuidas como únicas o especiales en los seres humanos. No obstante, esta concepción es un tanto ambigüa, puesto que realizar grandes cálculos es una "habilidad superior", realizada únicamente por (la mayor parte de los) seres humanos, y sin embargo no es considerado como parte de la Inteligencia Artificial, con excepción de que se pueda instruir a una computadora a realizar cálculos mediante ejemplos.

De la misma manera, existen actividades que no podrían considerarse como propiamente inteligentes o excepcionales pero que han representado problemas de compleja resolución para la modelación de la Inteligencia Artificial. Una de estas actividades es la capacidad de ver o bien, interpretar la información visual, presente en la mayor parte de los vertebrados, sin que se les considere necesariamente inteligentes por ello. Problemas quizá más complicados involucran la capacidad de entender el lenguaje natural y generarlo (*Natural Language Processing*). En estos términos también está la capacidad de generar sistemas que aprendan de los datos (*Machine Learning*).

Rusell y Norvig [216], en su obra general de Inteligencia Artificial, distinguen dos grandes enfoques en el desarrollo de esta materia, los cuales son

- Enfoque racional: Desarrollar programas o sistemas capaces de pensar o actuar ra-

[2]No es adecuado utilizar al DRAE para definir disciplinas académicas, búsquese la definición de Topología, por ejemplo. Sin embargo, el ejercicio es adecuado para emprender una exploración hacia el tema.

cionalmente.

- Enfoque antropocentrista: Desarrollar sistemas semejantes al ser humano, ya sea en el pensamiento o en la acción.

Esta frontera de ambos enfoques es muy difusa, incluso posiblemente más de lo que concibieron Russell y Norvig. Las redes neuronales convolucionales (*Convolutional Neural Networks* o CNNs, [146]), *verbigratia*, es un modelo inspirado en el funcionamiento de la Corteza Visual, pero no es propiamente un modelo de la misma corteza (*v. infra*), esto es, su formulación tomó algunas ideas producidas en la investigación de las redes neuronales biológicas de las cortezas visuales de animales como los gatos, pero en sí mismas no constituyen un modelo de la vista. Sin embargo, desde el 2013 y con el advenimiento de la AlexNet [133], el problema de clasificación de imágenes se considera en gran medida resuelto y la posterior investigación realizada en este campo aboga por el uso de este tipo de redes. De esta forma, un problema de resolución compleja, logró ver luz con el uso de modelos que estaban basados en otros modelos biológicos, sin ser un modelo estrictamente biológico.

En esta tesis, de la cual hasta ahora sólo hemos presentado ideas generales y un tanto vagas, será una defensa del segundo enfoque, o más bien, de un enfoque ligeramente más biológicamente plausible. En la práctica, la investigación sigue su propia evolución por lograr algoritmos más eficaces en la resolución de problemas concretos, que han fragmentado a la Inteligencia Artificial en una pléyade de disciplinas asociadas. En gran medida estos problemas pueden formularse en términos matemáticos para traducirse en modelos. Siguiendo este enfoque antropocentrista, a la parte de Inteligencia Artificial que nos restringiremos es a la correspondiente a las Redes Neuronales Artificiales (*Artificial Neural Networks* o ANNs). Se le ha adjuntado el adjetivo de *artificial* a este tipo de redes neuronales debido a que se tratan de simulaciones, consideradas como simplificaciones extremas dependiendo del contexto, de las operaciones que se realizan en el sistema nervioso. Por lo tanto, un modelo usando ANNs es un modelo biológicamente inspirado y cada solución a problemas que se realicen utilizando este enfoque puede darnos luz sobre cómo se resuelven realmente en nuestro cerebro, o bien basarse de la citoarquitectura del mismo. Pero tales

redes neuronales cuentan con artificialidad. No es lo mismo resolver un problema de clasificación de caracteres utilizando Máquinas de Soporte Vectorial a usar Redes Neuronales, las cuales cuentan con mayor sustento biológico, en especial las convolucionales. Del mismo modo, tampoco es similar emplear ANNs a usar Redes Neuronales Pulsantes (*Spiking Neural Networks* o SNNs) o redes con aprendizaje hebbiano. Sin embargo, la aplicación de métodos basados en la naturaleza no debe ser ingenua, la biología nos aporta ideas que se pueden abstraer para aplicarse en máquinas, pero la base de ambos es considerablemente distinta.

La pregunta central que vamos a abordar en esta tesis es la siguiente: *¿de qué manera las redes neuronales biológicas se optimizan?* Y las subsecuentes preguntas: *¿de qué forma se pueden incorporar los métodos biológicos de optimización en las redes neuronales artificiales? ¿Qué ventajas involucra utilizar modelos de aprendizaje naturales frente a los que emplean mayores puntos de artificialidad?* Hemos referido que las redes neuronales efectúan procesos de optimización, pero esta afirmación puede no ser cierta para redes neuronales biológicas. ¿Qué nos hace suponer que tales redes realizan una optimización? Además esto nos induce a pensar que el sistema nervioso resuelve problemas de clasificación supervisados, cuando puede ser más bien de corte no supervisada. Sin embargo, la posibilidad de resolver óptimamente problemas de clasificación computacionalmente complicados, nos permite hablar de que en la práctica los sistemas nerviosos son capaces de resolver tales problemas, aunque no sea lo único que realicen.

Como se verá, las redes neuronales se pueden optimizar en dos niveles: un nivel paramétrico y otro hiperparamétrico. En el nivel paramétrico se encuentran los *pesos* o fuerzas de conexión entre las neuronas, cuyo carácter de variable y clave para el aprendizaje fue defendido por Donald Hebb y es actualmente entendido como la plasticidad sináptica sirviendo como sustento para las redes neuronales artificiales. Un segundo nivel es el hiperparamétrico, que es el correspondiente a las constantes de la red externas a los pesos, como el número de neuronas.

La optimización paramétrica de redes neuronales generalmente se denomina como entrenamiento y es una forma de aprendizaje. Esto nos llevará a tratar de buscar los mecanismos de aprendizaje que ocurren a nivel neuronal, los cuales como se verá, tienen mayor

relación con la Regla de Hebb y el proceso de Potenciación a Largo Plazo. Por su parte, aspectos como el número de neuronas parecen estar más relacionados con la evolución debido a la diversidad de sistemas nerviosos con números variables de neuronas. Sin embargo esta afirmación es empírica y la regulación del número de neuronas o las conexiones entre las mismas puede no estar totalmente mediado por un mecanismo evolutivo. Por ejemplo, en el capítulo de la Regla de Hebb veremos que el Hipocampo cuenta con un proceso de Neurogénesis adulta, lo cual le permite añadir neuronas, pero esto no ha sido observado en la mayor parte de las regiones cerebrales.

De este modo, el objetivo central de la tesis será proponer mecanismos y algoritmos de optimización de redes neuronales utilizando métodos biológicamente plausibles, partiendo de las formulaciones existentes de ANNs funcionales, revisar sus *puntos de artificialidad* y reducirlos gradualmente, evitando, en la medida de lo posible, la pérdida de las ventajas obtenidas con el modelo base.

La *plausibilidad biológica*, por ende, se considerará relativa. No se desea realizar un modelo grande sobre la actividad eléctrica de las neuronas pues el enfoque de esta tesis parte del concepto de neurona artificial, con el cual se han construido redes neuronales que actualmente son el estado de arte en un amplio espectro de problemas. Tampoco se quisieran perder tales cualidades utilizando un modelo más plausible o imposibilitar su implementación en un sentido práctico sino tratar de obtener ventajas adicionales que se puedan facilitar a partir de la dicha reducción de la artificialidad.

1.1. Optimización de funciones

Muchos de los problemas específicos aludidos que se enfrentan son los problemas de clasificación. Una forma usual de abordar este problema es mediante la optimización numérica. Dado un conjunto de datos, representado como $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, donde $\mathbf{x} \in \mathbb{R}^m$ es un vector de características y y es la etiqueta del mismo, desde una primera aproximación modelada en el conjunto $\{0, 1\}$. Entonces definimos una función de clasificación $c : \mathbb{R}^m \to \{0, 1\}$. El problema puede plantearse como reducir un error de clasificación, por ejemplo, la suma de errores cuadrados o *Sum of Square Errors*, dada por

$$SSE(c) = \sum_{i=1}^{n} (c(\mathbf{x}_i) - y_i)^2. \tag{1.1}$$

De este modo, el problema base es encontrar la función que reduzca el error de clasificación, o expresado de manera más concreta

$$\underset{c \in \{c^* : \mathbb{R}^m \to \{0,1\}\}}{\arg\min} \sum_{i}^{n} (c(\mathbf{x}_i) - y_i)^2. \tag{1.2}$$

Sin embargo, minimizar en el espacio de funciones es muy complejo en términos matemáticos, pues el conjunto $\{c^* : \mathbb{R}^m \to \{0,1\}\}$ tiene una cardinalidad muy grande, la cual puede calcularse haciendo uso de Teoría de Conjuntos. La cardinalidad de este conjunto, por definición, está dada por

$$\Big| \prod_{\mathbf{x} \in \mathbb{R}^m} \{0,1\} \Big|. \tag{1.3}$$

Podemos aplicar algunos resultados de Teoría de Conjuntos, como la Desigualdad de Zermelo-König, para darnos una idea de su cardinalidad:

$$\begin{aligned} \Big| \prod_{\mathbf{x} \in \mathbb{R}^m} \{0,1\} \Big| &= \bigotimes_{\mathbf{x} \in \mathbb{R}^m} |\{0,1\}| \\ &= \bigotimes_{\mathbf{x} \in \mathbb{R}^m} 2 \\ &> \sum_{\mathbf{x} \in \mathbb{R}^m} 1 \\ &= \text{máx}(|\mathbb{R}^m|, 2) = |\mathbb{R}| = \mathfrak{c}. \end{aligned}$$

Esto quiere decir que tal conjunto es mayor que la cardinalidad de los números reales. Este problema se puede simplificar si en lugar de minimizar sobre un espacio de funciones, efectuamos la optimización sobre un espacio de parámetros, con valores reales, por ejemplo, un vector $\mathbf{w} \in \mathbb{R}^l$. Así, esta función es de la forma $c_{\mathbf{w}}$, es decir, paramétricamente dependiente.

Esta modificación es altamente conveniente, pues la optimización se realiza del siguiente modo

$$\text{argmin}_{\mathbf{w}\in\mathbb{R}^l} \sum_{i=1}^{n} (c_{\mathbf{w}}(\mathbf{x}_i) - y_i)^2. \tag{1.4}$$

Esto restringe el espacio de búsqueda y evita la ambigüedad que causaría tratar de definir algo como $c(\mathbf{x}_i) = y_i$. Sin embargo, también se puede sobreajustar, por lo que resulta conveniente la definición de hiperparámetros para ajustar un conjunto de validación, el cual sirve para estudiar las capacidades de generalización del modelo.

1.1.1. Métodos basados en el Gradiente

Los métodos basados en el gradiente son una familia de técnicas de optimización, que como se verá más adelante, son ampliamente utilizados en entrenamiento de redes neuronales artificiales. Están asociadas a una función objetivo $f : \mathbb{R}^n \to \mathbb{R}$, diferenciable en todo punto, que se pretende optimizar. En términos matemáticos, esto involucra hallar

$$\arg\min_{x\in\mathbb{R}^n} f(x). \tag{1.5}$$

Su eficacia y simplicidad los convierte en métodos adecuados para una gran variedad de problemas de optimización. Es posible preguntarnos, de manera casi preliminar, algunas consideraciones de naturaleza teórica o incluso filosófica: ¿realizan las neuronas un Descenso de Gradiente? ¿De qué forma lo hacen y cómo optimizan? ¿Cuál es la función objetivo que las neuronas reales tratan de optimizar? Estas preguntas son centrales para el presente trabajo ya que buscan establecer hasta qué punto los métodos conocidos actualmente emulan a las neuronas reales. Las descripciones que se realicen en esta sección están basadas en [4], salvo donde se indique, aunque la exposición teórica del Descenso de Gradiente Clásico está basada en [187].

Descenso del Gradiente Clásico

Sea $x \in \mathbb{R}^n$. Entonces el Descenso de Gradiente está dado por la siguiente regla recursiva:

$$x_0 = x \tag{1.6}$$

$$x_{i+1} = x_i + \alpha \nabla f(x_i) \quad \forall i = 1, 2, \ldots \tag{1.7}$$

a la constante $\alpha > 0$ se le conoce como *tasa de aprendizaje* o *learning rate*. Es importante señalar que la convergencia del Descenso del Gradiente en general no está garantizada, puesto que un valor de α suficientemente grande puede superar el mínimo buscado (por otro lado valores pequeños llevan a una lenta convergencia).

En general el Descenso del Gradiente forma parte de una familia de métodos de optimización conocidos como de Línea de Búsqueda dados por

$$x_{t+1} = x_t + \alpha_t p_t. \tag{1.8}$$

donde p_t es la dirección de descenso. Entre este tipo de métodos se incluyen la mayoría de los métodos esbozados en esta tesis. Una *dirección de descenso* está dada por $p_t^T \nabla f(x_t) < 0$. Para el caso del Descenso del Gradiente, esta dirección de descenso se satisface trivialmente porque $p_t = -\nabla f(x_t)$, por lo que $-\nabla f(x_t)^T \nabla f(x_t) = -||\nabla f(x_t)|| < 0$ cuando el gradiente no es 0. Podemos tratar de verificar en qué condiciones el Descenso de Gradiente converge. Una idea preliminar y deseable es que $f(x_{t+1}) < f(x_t)$, pero esto es insuficiente porque es posible encontrar sucesiones que decrezcan infinitivamente sin llegar a un ínfimo. Condiciones más contundentes son las Condiciones de Wolfe dadas por

$$f(x_{t+1}) \leq f(x_t) + c_1 \alpha_t \nabla f(x_t)^T p_t \tag{1.9}$$

$$c_2 \nabla f(x_t)^T p_t \leq \nabla f(x_{t+1})^T p_t, \tag{1.10}$$

donde $0 < c_1 < c_2 < 1$. Es posible probar que podemos encontrar intervalos apropiados para α_t que cumplan las condiciones de Wolfe si la función está acotada inferiormente y es suave. Para garantizarnos de que se cumplan tales condiciones, podemos utilizar el procedimiento de Retroceso o *Backtracking* que consiste en reducir α hasta satisfacer las condiciones de Wolfe. Una variante del Teorema de Zoutendijk muestra que este procedimiento nos permite llegar hasta un mínimo local o punto silla.

Teorema 1 (Zoutendijk)**.** *Consideremos un método de línea de búsqueda con p_t como una dirección de descenso que cumple con las condiciones de Wolfe. Sea $f : \mathbb{R}^m \to \mathbb{R}$, acotada inferiormente y continuamente diferenciable en un abierto U que contiene a $\{x : f(x) \leq f(x_0)\}$ donde x_0 es el punto inicial de la iteración. Si el gradiente ∇f es Lipschitz en U, entonces*

$$\lim_{t \to +\infty} ||\nabla f(x_t)|| = 0. \tag{1.11}$$

Esto significa que el Descenso del Gradiente con Retroceso logra minimizar el gradiente. Con la primera Condición de Wolfe (Condición de Armijo), como se toma una dirección de descenso, entonces $c_1 \alpha \nabla f(x_t)^T p_t < 0$, por lo que $f(x_{t+1}) < f(x_t)$, con lo cual nos garantiza que con el método decrementamos tanto el gradiente como la función objetivo, lo cual nos lleva o a un mínimo local o un punto silla.

Variantes

Además del Descenso de Gradiente Clásico se han propuesto variantes al método original. El **Descenso de Gradiente Estocástico** (*Stocastic Gradient Descend*) aplica el algoritmo a sólo una muestra aleatoria de los datos. Esta forma es la que generalmente se adapta para redes neuronales, debido a la cantidad de datos a procesar.

El **Descenso de Gradiente con Impulso** (*Momentum Gradient Descend*) busca reducir las oscilaciones que pueden ocurrir cuando las Condiciones de Wolfe no se satisfacen y si no se desea utilizar Retroceso. Este método está dado por

$$\nu_t = \gamma \nu_{t-1} - \alpha \nabla f(x_t), \tag{1.12}$$

$$x_{t+1} = x_t + \nu_t. \tag{1.13}$$

El vector ν se puede inicializar en 0 y guarda la dirección de los gradientes anteriores, mediado por $\gamma \in [0, 1)$. Este método también contribuye a evitar que el descenso se estanque en puntos silla o mínimos locales pequeños, debido a que apunta en la dirección de los gradientes históricos si el gradiente actual se hace 0.

Una mejora a este procedimiento fue propuesta por Nesterov [184], cuyo objetivo es aliviar la acumulación de gradientes, conocido como el **Gradiente Acelerado de Nesterov** (*Nesterov accelerated gradient*) cuya regla de optimización está dada por

$$\nu_t = \gamma\nu_{t-1} - \alpha\nabla f(x_t + \gamma\nu_{t-1}), \tag{1.14}$$

$$x_{t+1} = x_t + \nu_t. \tag{1.15}$$

Entre las propuestas dadas en la década pasada destacan Adagrad, RMSprop y Adam. Los gradientes adaptativos o **Adagrad** (*Adaptative Gradients*) fueron propuestos en 2011 [57] y la técnica consiste en actualizar una tasa de aprendizaje dinámica para cada entrada l del vector (x_t^l). La tasa de aprendizaje dinámica está dada por $\frac{\alpha}{\sqrt{\omega}+\varepsilon}$ donde ω_l es la suma cuadrada de las parciales

$$\omega_l = \sum_{l=1}^{m} \Big(\frac{\partial f(x_t)}{\partial x_t^l}\Big)^2 \tag{1.16}$$

y $\varepsilon \ll 1$ es una cantidad pequeña para evitar la división entre 0. Una versión con impulso de Adagrad es la Propagación de Raíces Cuadradas (*Root Means Square Propagation*) o **RMSprop** la cual no aparece por primera vez en un artículo de investigación o libro sino en un curso de Coursera [244] en el año 2012, impartido por Tieleman y Hinton.

Finalmente en el 2015, Kingma y Ba propusieron **Adam** [125], el cual junto con RMSprop ha tenido una fuerte influencia en la optimización de redes neuronales. Sea $g_t = \nabla f(x_{t-1})$, $\beta_1, \beta_2 \in [0, 1)$, m_t, v_t inicializados en 0. Entonces el algoritmo de Adam está dado por las siguientes actualizaciones

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)g_t, \tag{1.17}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)g_t^2, \tag{1.18}$$

$$\hat{m}_t = \frac{m_t}{1 - (\beta_1)^t}, \tag{1.19}$$

$$\hat{v}_t = \frac{v_t}{1 - (\beta_2)^t}, \tag{1.20}$$

$$x_t = x_{t-1} - \alpha\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}. \tag{1.21}$$

1.2. Artificialidad en Redes Neuronales Artificiales

Hemos designado como *artificialidad de un modelo* a los aspectos del mismo que no están respaldados por las ciencias empíricas, en este caso la Biología. De este modo, si bien la modelación matemática puede darse en forma de axiomas para crear una *Teoría*, tales axiomas o bases deben verificarse empíricamente para formular una teoría sólida y consistente con las observaciones, o rechazarse por completo. Otras artificialidades pueden darse en los algoritmos que se formulan para alcanzar un objetivo si no consideran los procesos exactos que ocurren biológicamente. Por ejemplo, es posible que las redes neuronales reales logren una optimización de la red mediante sus reglas de plasticidad. Alcanzarlo de otro modo (como por ejemplo usando Descenso del Gradiente) cumple con este principio, pero la forma de lograrlo podría ser diferente y aunque logre resolver el problema en cuestión, puede generar otros no considerados previamente. Asimismo, las artificialidades inicialmente propuestas pueden justificarse por la existencia de mecanismos similares en la naturaleza previamente desconocidos.

Como estudiaremos con detalle en el Capítulo 2 correspondiente a Redes Neuronales (se sugiere leer tal capítulo en caso de carecer de nociones de las mismas) el modelo de ANN está basado en el planteamiento de Neurona Artificial, inicialmente formulado por McCullogh y Pitts [173] pero que fue continuado hasta su esquema actual. Existen ciertas diferencias con respecto al modelo de ANNs y las redes neuronales biológicas (BNNs) que ponen en evidencia sus puntos de artificialidad. La existencia de artificialidades, sin embargo, es una motivación para introducir nuevas ideas a los modelos preexistentes.

Eluyode [63] enlista las principales diferencias entre los redes reales y las artificiales, algunas de las cuales se enlistarán a continuación:

- Las redes neuronales biológicas se comunican mediante pulsos (*spikes* o potenciales de acción) que utilizan como mecanismo para transmitir información, mientras las redes artificiales utilizan valores continuos tanto en sus parámetros como en sus salidas.

- Las redes neuronales biológicas tienen un tiempo de procesamiento relativamente lento en comparación con las computadoras actuales, aunque altamente paralelizadas

y capaces de resolver problemas aún no alcanzados por las computadoras actuales en general.

- El número de neuronas y de interconexiones es ampliamente superior en varios seres vivos que en las redes neuronales comunes (incluso para el caso de *Deep Learning*).
- La plasticidad en redes neuronales biológicas está en gran medida mediada por neuromoduladores, que no aparecen en redes neuronales clásicas.

A esta lista también señalaremos la forma en la que se suelen entrenar las redes neuronales para aprendizaje supervisado, que son los métodos basados en el gradiente. Dado que las reglas de plasticidad son un enfoque al parecer más bioplausible para el aprendizaje de redes neuronales, es por ello que se concluye que el Descenso del Gradiente es o un enfoque sin correspondiente biológico o su correlato biológico no ha sido encontrado o es una regla de plasticidad oculta. Esta posibilidad se abordará integramente en el capítulo de Redes de aprendizaje híbrido. Mención aparte merece la Retropropagación, a la cual no haremos énfasis en esta tesis pero la presencia de mecanismos similares en el cerebro son objeto de discusión en el artículo reciente de [153]. Otro aspecto fundamental es la arquitectura, aunque una revisión de los esquemas biológicos parece conveniente en este punto.

1.2.1. Propuestas para reducir la Artificialidad

Para reducir la artificialidad de las redes neuronales artificiales se han abierto muchas vías, muchas de las cuales no serán tratadas en esta tesis, en algunos casos justificaremos las razones de no incluir tales sugerencias. Podemos abordar a las redes neuronales desde los modelos más precisos de su actividad eléctrica en forma de potenciales de acción, que es la señal o pulso que emiten, descritas por el modelo de Hogking-Huxley [97], el cual ha tenido un fuerte impacto en el desarrollo de aspectos de Neurociencia Computacional. Un ejemplo de una implementación computacional de este modelo está expuesto en [259].

Las Redes Neuronales Pulsantes [39, 257] pueden considerarse como una simplificación del comportamiento de los potenciales de acción con respecto al modelo de Hogking-Huxley

pero que agrupan la naturaleza pulsante de la transmisión de la información por parte de las neuronas reales. Ejemplos de aplicaciones de las mismas se citarán en algunas partes del texto.

Otra propuesta relevante son las Redes Neuronales con Procesamiento Dendrítico, sugeridas por Spruston y Kath [233] para responder a la pregunta sobre qué hace al cerebro humano más poderoso que las redes artificiales, describiendo las operaciones dendríticas que se realizan y con ello ampliando al modelo clásico de Tasa de Disparo (también llamado *Integrate-and-fire*) de las neuronas individuales. Finalmente otras interesantes mejoras señalan a la integración de astrocitos artificiales en redes neuronales [204] y neuromoduladores en el aprendizaje de neuronas [98].

1.2.2. Enfoque principal

Como se ha mencionado, la vía principal que se tomará en esta tesis no va dirigida a las operaciones neuronales sino que simplemente al aprendizaje. El modelo de tasa de disparo, por lo tanto, no será reemplazado y será estándar para los trabajos que se realicen e incluso nos enfocaremos en las investigaciones que abordan cómo la tasa de disparo de neuronas individuales nos permite descubrir el comportamiento de las mismas. A pesar de tratarse de una simplificación quizá excesiva, resulta conveniente tal como señalan Dayan y Abbott en su obra [51]. Adicionalmente, para abordar los problemas previamente mencionados, es conveniente añadir una cantidad considerable de neuronas para lo cual simular procesos más exactos puede resultar innecesario e incluso consumir recursos de cómputo que no se tendrían en sistemas parcialmente linealizados.

Siguiendo esta línea lo que interesa por el momento es poder incorporar mecanismos más naturales de optimización en los dos niveles sin tener que sacrificar las ventajas ya obtenidas por las redes neuronales, las cuales es un hecho que funcionan. No se pretende, por lo tanto, realizar un modelo biomatemático de las neuronas o poblaciones de neuronas, sino describir sus aspectos funcionales e implementarlos desde un plano computacional. Tampoco se incluirán modelos de neurotransmisores sino que se considerará como pendiente en la modelación futura.

1.3. Objetivos

Los objetivos centrales de la tesis están referidos de manera implícita en las anteriores líneas. Serán reformulados aquí:

Objetivo general: Realizar propuestas para la formulación de mecanismos biológicamente más plausibles de optimización de redes neuronales artificiales que permitan resolver problemas de clasificación sin perder la generalidad lograda.

Objetivos específicos:

- Describir los modelos de aprendizaje neuronal más representativos e implementarlos para la resolución de problemas de clasificación visual, como MNIST y EMNIST.
- Definir las ventajas que se obtienen al operar bajo enfoques con mayor plausibilidad biológica.
- Proponer mecanismos evolutivos para la optimización de redes neuronales a nivel hiperparamétrico.
- Describir los puntos de artificialidad que se generan al utilizar los modelos propuestos.
- Desarrollar aplicaciones partiendo de la revisión de la literatura existente sobre la biología del aprendizaje.

1.4. Organización de la tesis

La organización que se realizó en la presente tesis no es del todo estándar. El orden de lectura sugerido es iniciar en el Capítulo 2 y después la Introducción para los lectores no familiarizados con las redes neuronales, aunque no es tan relevante lo que se requiere conocer para abordar la Introducción. Las bases matemáticas que se requieren no son más que las estándares para el área de la modelación de ANNs, que son Álgebra Lineal,

Cálculo Multivariable y Probabilidad, así como Ecuaciones Diferenciales que aparecen en la formulación de las reglas de Hebb y en los modelos poblacionales que se estudiarán en el capítulo 5. En ocasiones también se harán referencia a Teoría de la Medida e Integración e incluso Análisis Funcional, pero esto sólo será breve, al igual que la mención que se realizó de Teoría de Conjuntos. Adicionalmente he incluido breves aplicaciones referentes a Lógica Matemática.

El capítulo 2 de *Redes Neuronales* aborda las generalidades del tema, la formulación del modelo de neurona artificial desde el punto de vista de la tasa de disparo, que aporta un valor continuo y positivo a la salida de las redes neuronales. Tal capítulo es introductorio y sirve para definir los conceptos que serán fundamentales para el desarrollo de la tesis, por lo que las aportaciones que se realicen serán meramente teóricas. Se añade un sección correspondiente a redes recurrentes que no involucran a la regla de Hebb sino que operan bajo el métodos basados en el gradiente. Por otro lado, mención aparte merece la técnica de Retropropagación, que es una forma de calcular el gradiente en varias capas. Como la discusión estará restringida al aprendizaje en una capa, no haremos mayor énfasis a este tema por el momento.

El capítulo 3 es de *Redes Neuronales Convolucionales y Visión*, cuyas aplicaciones se han expandido más allá del reconocimiento de objetos. Se aborda el tema explicando sus operaciones relevantes y discutiendo sus ventajas frente a métodos tradicionales de Visión Computacional, los cuales pueden ser tomados por más artificiales que las ANNs. Luego revisaremos algunas de las arquitecturas más notables de redes neuronales hasta la fecha, siendo algunas útiles para el desarrollo de esta tesis. Finalmente haremos una revisión de los trabajos realizados sobre el procesamiento de la información visual y reconocimiento de objetos en animales, desde las investigaciones clásicas de Hubel y Wiesel hasta los hallazgos más recientes en el Lóbulo Medial Temporal, del cual realizaremos una revisión posterior. Al tratarse de una parte introductoria, tampoco se presentarán resultados en este capítulo, sino que únicamente bosquejaremos en sus conclusiones una discusión sobre por qué consideramos a las redes neuronales convolucionales como modelos apropiados a pesar de las desventajas señaladas por algunos neurocientíficos quizá sea lo mejor que se disponga para emular el funcionamiento de la Corteza Visual.

En el capítulo 4 abordamos el tema de las Redes Neuronales Evolutivas, centrada en la optimización de redes neuronales a nivel hiperparamétrico mediante algoritmos basados en la Evolución biológica. Sin embargo, no nos detendremos a realizar una revisión profunda sobre los algoritmos genéticos preexistentes y su impacto en redes neuronales sino que realizamos nuestra propia propuesta basada en el modelo de Lotka-Volterra.

El capítulo 5 introduce a la Regla de Hebb (término que se pluralizará), el cual es un tema central para el desarrollo de la tesis pues aborda uno de los modelos biológicos más utilizados en contextos de Inteligencia Artificial, que se ha materializado en dos temas relevantes del área de las Redes Neuronales Artificiales: las Redes de Hopfield y las de Kohonen. En este capítulo realizamos el primer intento de implementar a la regla de Hebb en una red de una sola capa en problemas de clasificación de imágenes reales. Asimismo, desarrollamos una primera aplicación de la regla de Hebb combinando a dos Redes de Hopfield con una de Kohonen.

En el capítulo 6 finalmente reabrimos la discusión entre las Reglas de Hebb y los algoritmos basados en el gradiente, discutiendo a nivel teórico hasta qué punto algunas reglas de Hebb se comportan como algoritmos basados en el Gradiente tratando de responder una duda relevante: ¿derivan las neuronas para optimizar? Posteriormente abordamos el tema a nivel práctico para implementar a las reglas de Hebb en procesos de Transfer Learning utilizando redes convolucionales preentrenadas con algoritmos basados en el gradiente. Este enfoque híbrido fue lo que mejor dio resultados y logra aprovechar las ventajas que aporta el utilizar una regla de Hebb sin perder tanta exactitud como se obtiene al aplicar completamente métodos basados en el gradiente. Este enfoque logra uno de los objetivos personales más importantes de esta tesis, el cual es desarrollar mecanismos de aprendizaje natural en tiempo real para redes neuronales artificiales. Como se sustentará, el proceso de *Transfer Learning* en tiempo real es una de las ventajas asociadas al empleo de la regla de Hebb frente a los métodos convencionales.

1.5. Aportaciones principales

Como se menciona en el Prólogo, una parte de los temas relacionados con el trabajo de tesis y de prácticas profesionales ha sido expuesto en diferentes congresos celebrados en lugares como Monterrey, Puebla así como la misma ciudad de Mérida, así como ha suscitado la publicación de artículos que exponen varios de los aportes realizados durante los años que se trabajó esta tesis. Al momento de la redacción de esta introducción, se cuentan con dos artículos publicados y uno en vías de publicación, los cuales son:

1. Eficacia de diferentes reglas hebbianas en el Aprendizaje Supervisado [6].

2. Redes Neuronales Evolutivas con modelos de Lotka-Volterra.

3. Convolutional Neural Networks with Hebbian-based rules in Online Transfer Learning [5].

El primer artículo si bien corresponde al tema de La Regla de Hebb, el contenido y las ideas no son expuestos en la presente tesis. Se desarrolla un algoritmo de clasificación por rejilla que utiliza a la regla de Hebb, el cual da resultados comparativos razonablemente buenos frente a algoritmos estándares. Si bien es posible continuar en tal dirección, he optado por el camino que se traza en la presente tesis, la cual está centrada en los artículos 2 y 3, mientras que los temas trabajados en las Prácticas Profesionales y su correspondiente artículo solamente fueron incluidos parcialmente.

El artículo de *Redes Neuronales Evolutivas con modelos de Lotka-Volterra* condensa el núcleo de las aportaciones logradas en el capítulo 4, en el cual se desarrolla un método de optimización hiperparamétrica que utiliza algunos elementos de los algoritmos genéticos y sigue una simulación poblacional del modelo de Depredador-Presa (Lotka-Volterra) para lograr aumentar la exactitud en la clasificación de imágenes como las provistas en el dataset EMNIST de caracteres.

Por su lado, el artículo *Convolutional Neural Networks with Hebbian-based rules in Online Transfer Learning* (la cual considero como mi principal elaboración) aplica a diferentes reglas de Hebb (Simple, Covarianza, Oja y BCM) utilizando capas convolucionales

preentrenadas para efectuar aprendizaje en tiempo real, el cual se desprende del capítulo 6 realizado en esta tesis.

Si bien muchas de las aportaciones relevantes han sido a la fecha transformadas en artículos y ponencias, los trabajos publicados aparecen en forma extendida en sus respectivos capítulos y con menores restricciones tanto de espacio como de expresión. De esta forma, se incluyó una mayor experimentación, mayor revisión de la bibliografía e incluso conclusiones que no habían sido alcanzadas en los artículos dictaminados. A su vez, en el capítulo 5 se incluyó una aplicación correspondiente a las redes HKH que no ha sido incluido en algún artículo.

Al tratarse de una tesis en Matemáticas (y no de Computación), se han añadido algunas proposiciones y discusiones a nivel matemático sobre las redes neuronales y la regla de Hebb. Muchas de estas proposiciones son inéditas y no aparecen (ni aparecerán) en algún artículo publicado por el autor, especialmente las que aparecen en la primera mitad del capítulo 6 considero que son de interés relevante. Sin embargo, esta tesis no es un trabajo matemático de la Regla de Hebb que llevaría a una discusión sobre el Análisis de Componentes Principales o una solución analítica de las mismas, dado que esa vía ha sido ya explorada y se aparta de los objetivos trazados previamente. Las proposiciones matemáticas que son originales del autor son las siguientes:

1. Proposición I (Prólogo)
2. Proposición 1 (capítulo 2)
3. Proposición 2 (capítulo 2)
4. Proposición 3 (capítulo 4)
5. Lema 1 (capítulo 4)
6. Proposición 4 (capítulo 4)
7. Teorema 4 (capítulo 6)
8. Proposición 7 (capítulo 6)

9. Teorema 5 (capítulo 6)

10. Proposición 8 (capítulo 6)

La Proposición 3 solamente aparece esbozada e incompleta en el trabajo por publicar de redes evolutivas. La Proposición 4 y el Lema 1 sí fueron incluidos en tal artículo. El Teorema 4, por su lado, es una observación relativamente común.

Como se verá, se realiza una revisión relativamente amplia de varios artículos relevantes en Neurociencias. Algunas de estas ideas no son aplicadas, sino que solamente se presentan y posiblemente formen parte de las siguientes contribuciones que se realicen en esta materia.

Finalmente debo aclarar que gran parte de los códigos realizados están alojados en `https://github.com/Pherjev`, siendo Python el lenguaje principal en el que fueron escritos, utilizando el módulo de Keras para diseñar algunas de las redes neuronales estándares. Habiendo enfatizado los puntos rectores en el desarrollo de esta tesis, así como el enfoque que se trabajó, puede proceder a pasar la hoja e iniciar la lectura propiamente dicha de esta tesis.

Capítulo 2

Redes Neuronales

Se considera[1] que el primer trabajo de Redes Neuronales Artificiales (y uno de los pioneros de Inteligencia Artificial) es el artículo *A logical calculus of the ideas immanent in nervous activity* de Warren S. McCulloch y Walter H. Pitts de 1943 [173]. Este trabajo pionero, estaba influenciado por los avances en Lógica Matemática y Neurociencias que se dieron durante el siglo XX.

Con el subsiguiente desarrollo de la computación, las redes neuronales pudieron ser finalmente implementadas. Una de estas primeras implementaciones está ligada a los trabajos de Frank Rosenblatt sobre el Perceptrón [214], al cual revisaremos en el siguiente capítulo. Rosenblatt introduce la *Regla de Aprendizaje del Perceptrón*, uno de los primeros algoritmos de entrenamiento de una red neuronal artificial. Las limitaciones de los perceptrones motivaron al origen de las redes multicapa y al desarrollo del algoritmo LMS (Mínimos Cuadrados medios o *Least Mean Square*), introduciendo la función de costo de Suma de Errores Cuadrados o *Mean Square Error* (MSE) dada por

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{2.1}$$

donde $\hat{y}$ es una aproximación de y, la cual puede optimizarse mediante Descenso de Gradiente, que ya se había adaptado para el caso de las redes neuronales en 1961 (véase [254]). El posterior desarrollo de la Retropropagación (*Backpropagation*) permitió culminar

[1] Utilizaremos [229] para basarnos en esta reconstrucción de la historia de las Redes Neuronales Artificiales

con un algoritmo sólido para el aprendizaje de redes neuronales conectadas hacia adelante multicapa (*Feedforward Neural Networks* o FNN, o bien, *Multilayer Neural Networks*).

De manera paralela, se realizan numerosos avances en materia de las Redes Neuronales Recurrentes (*Recurrent Neural Networks* o RNN), cuyos primeros ejemplos son las *Redes de Hopfield*, publicado en 1982 por John J. Hopfield [101]. En la década de los noventas, surgieron las redes de Elman [61] y en 1997, Hochreiter y Schmidhuber proponen las células de *Long Short-Term Memory* (LSTM), que resolverán problemas prácticos del entrenamiento de las redes recurrentes, siendo actualmente ampliamente utilizadas para diversas tareas como *Image Captioning* (véase el Capítulo 6).

Al año siguiente[2] del desarrollo de las células LSTM, Yan LeCun y sus colaboradores publicaron una de las primeras redes neuronales convolucionales [146] (*Convolutional Neural Networks* o CNN) (conocida como la LeNet-5) que orilló al desarrollo del Aprendizaje Profundo o *Deep Learning*, que resolvió en gran medida el problema de reconocimiento de caracteres manuscritos. El desarrollo de las CNNs, estuvo motivado por la preliminar propuesta del Neocognitrón de Fukoshima [69], que se enraiza en la teoría desarrollada por Hubel y Wiesel sobre la corteza visual en 1960 [108].

En el 2012, Krizhevsky, Sutskever y Hinton proponen la *AlexNet* [133], una red CNN que logró una gran exactitud (casos clasificados correctamente entre casos totales) en el problema de clasificación de 1000 categorías de imágenes de la Imagenet, alcanzando una exactitud de top-5[3] de 17 %, algo impresionante para el momento y que motivó a un mayor desarrollo de redes neuronales convolucionales para la clasificación de imágenes. El desarrollo de nuevas arquitecturas convolucionales (como la VGG-19 [228]) en los años venideros contribuiría a reducir aún más el error de clasificación.

¿Es la motivación biológica de las Redes Neuronales lo que explica su éxito como clasificadores en el Aprendizaje Supervisado? ¿Hasta qué punto son las Redes Neuronales Artificiales una aproximación del modelo biológico? Al punto, es conveniente preguntarse qué tan artificiales son las Redes Neuronales Artificiales. Como se ha visto, algunas de

[2] 1998 es el año de nacimiento del autor

[3] La medida top-n cuenta como caso clasificado correctamente si la clase correcta se encuentra en los primeros n lugares. Esta medida es útil para clases amplias como las de Imagenet, siendo un total de 1000.

las actividades cognitivas más relevantes como el reconocimiento y clasificación de objetos como patrones visuales se ha logrado en gran medida gracias al entendimiento del funcionamiento de la Corteza Visual en 1960. Pero desde los trabajos de Hubel y Wiesel hasta el surgimiento de la AlexNet, tuvo que transcurrir más de medio siglo.

Volviendo a las preguntas iniciales que motivan esta sección, posiblemente la mayor similitud entre las Redes Artificiales (ANN) y las Biológicas (BNN) sean las operaciones internas y la posibilidad de ordenarse en una arquitectura conveniente. Como se verá más adelante, las CNNs manifiestan cierta similitud con la estructura de la Corteza Visual. Esto nos da atisbos para considerar que el encéfalo no es una red recurrente completamente conectada con millones de parámetros sino que se organiza de forma estratégica para procesar información específica.

Asimismo, en el presente capítulo introduciremos los conceptos clásicos de Redes Neuronales que han derivado al desarrollo de áreas como *Deep Learning*, enfatizando a aquellos que tengan mayor inspiración biológica. Particularmente útil será realizar una revisión de las arquitecturas que hayan tenido un buen desempeño para realizar actividades cognitivas superiores. Esto nos lleva a reformularnos las preguntas iniciales: ¿hasta qué punto son biológicas las Redes Artificiales?, es decir, ¿qué aspectos podrían considerarse como modelos plausibles del funcionamiento de las neuronas reales? Para ello, deberemos tratar de sustentar la relación, posiblemente tibia, entre los modelos de Neurociencia Computacional y de Redes Neuronales Artificiales.

Adicionalmente debemos recalcar el hecho (probado en las diferentes mediciones de exactitud) que las Redes Artificiales *funcionan*[4]. Existen muchos aspectos funcionales de las redes biológicas que aún pueden desconocerse o aún más, no ser aplicables para el diseño de las computadoras modernas, en especial para las que procesan la información en serie. Es posible que no se disponga de la tecnología o los avances neurocientíficos necesarios para remediar tales lagunas de conocimiento, pero tales desventajes podrían ser solventadas por algunas técnicas sugeridas matemáticamente por los modelos de red artificial existentes.

[4]Revísese el concepto de verdad de Charles Pierce

2.1. Modelo de una neurona

Iniciaremos el estudio de las redes neuronales desde lo más simple desde el punto de vista funcional así como lo más antiguo desde el punto de vista cronológico. La teoría desarrollada por Santiago Ramón y Cajal a inicios del siglo XX, indica que las neuronas son la unidad básica de procesamiento de la información por el sistema nervioso de los animales, al probar que son entidades discretas y no continuas como lo sostenía Golgi. Esta teoría es conocida como la *Doctrina Neuronal* es la base de las Neurociencias actuales y puede considerarse como el primer pilar para la formulación de una teoría de Redes Neuronales Artificiales. Otra proposición de la Doctrina Neuronal, no siempre asumida en la Inteligencia Artificial[5], es el hecho que la trasmisión eléctrica de las neuronas ocurre en una sola dirección: de las dendritas al axón [72].

Sin pretender profundizar la dinámica precisa sobre procesos como la sinapsis (medio por el cual se comunican las neuronas, enviando información en forma de neurotransmisores a la neurona siguiente o *postsináptica*, la cual dispara un *potencial de acción* si el potencial eléctrico de la membrana alcanza un umbral), consideraremos una simplificación conveniente para fines computacionales[6]. Dichas simplificaciones son propuestas por [51], desde el punto de vista de la Neurociencia Teórica, que servirá como punto de partida. Definiremos

$$\delta(t) = \begin{cases} 1 & \text{Existe un impulso en t} \\ 0 & \text{No existe un impulso en t} \end{cases} \tag{2.2}$$

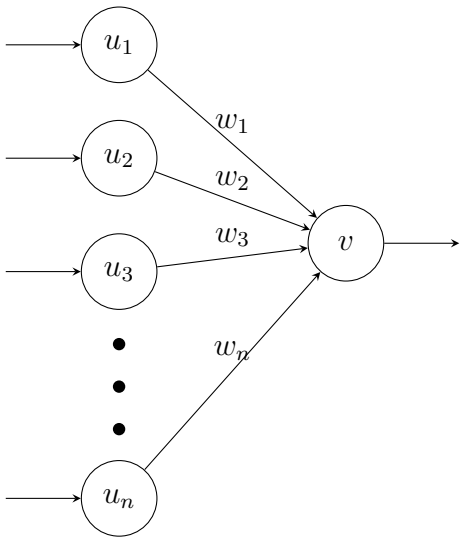
Tal impulso producido por una neurona es conocido como potencial de acción o *spike*. La tasa de disparo (*firing rate*) se define como el número de impulsos producidos por segundo y se mide en Hertz. En términos prácticos, dicha tasa de disparo puede medirse de forma discreta con los impulsos registrados y aproximarse con una función suave. Consideremos a una neurona con tasa de disparo v, que recibe información directamente de otras m neuronas, cuyas tasas de disparo son u_i, donde $i \in \{1, \ldots, m\}$. Entonces, un modelo para la tasa de disparo de v está dado por la siguiente ecuación diferencial

[5]Véase en el capítulo 5, la subsección correspondiente a las Redes de Hopfield.

[6]Para una revisión de los conceptos básicos de la biología de las neuronas, véase [72].

$$\tau_r \frac{dv}{dt} = -v + F(\mathbf{w} \cdot \mathbf{u}), \tag{2.3}$$

donde $\mathbf{w} = (w_1, \ldots, w_m)$ es el vector de pesos o fuerzas de conexión entre la neurona con tasa $\mathbf{u}$ y la neurona con tasa $\mathbf{v}$. F es una función de activación, típicamente sigmoide ($F(t) = \frac{1}{1+e^{-t}}$) para la modelación en Neurociencia Teórica. Un esquema de estas relaciones está provisto en el diagrama inferior



Empíricamente, $\tau_r \approx 0$, de donde observamos que

$$v = F(\mathbf{w} \cdot \mathbf{u}) \tag{2.4}$$

Esta última expresión es empleada para representar a una sola neurona artificial (a veces referida como perceptrón) y aparece en manuales como [216]. Hasta ahora, el diseño del perceptrón es solamente una simplificación del modelo de tasa de disparo neuronal. Sin embargo, la introducción de la neurona *bias* con valor constante $u_1 = 1$ le introduce artificialidad al modelo, pero permite construir un clasificador lineal completo.

2.1.1. Modelo de Amari

Discutiremos de manera breve un modelo global de actividad neuronal en el cerebro, que se conoce como el Modelo de Amari, propuesto originalmente en [8]. La construcción que aquí se expone es tomada de [205]. Sea $u(x,t)$ la actividad de una sola neurona, donde

x refiere a la posición (en el cerebro) y t al tiempo. Una variante de la ecuación 2.3 está dada por

$$\tau_r \frac{\partial}{\partial t} u(x,t) = -u(x,t) + \sum_{x \in I} w(x,y) F(u(y,t)), \tag{2.5}$$

donde $w(x,y)$ es el valor del peso que conecta a las neuronas con posiciones o índices x y y, mientras que I es el conjunto de índices de las neuronas presinápticas de x. La función de activación F es comúnmente sigmoide en este modelo, de forma que

$$F(s) = \frac{1}{1 + e^{-\beta(s-h)}}, \tag{2.6}$$

donde el parámetro h es el umbral y β controla la pendiente de la sigmoide en el centro. Podemos hacer $w(x,y) = 0$ si no existe conexión y generalizar a todas las posibles conexiones del cerebro, creando un conjunto B de todos los índices. Al tratarse de un modelo bastante grande, se puede tomar una versión continua de la ecuación anterior, que define el modelo de Amari:

$$\tau_r \frac{\partial}{\partial t} u(x,t) = -u(x,t) + \int_{x \in B} w(x,y) F(u(y,t)). \tag{2.7}$$

2.1.2. Formulación de una Neurona Artificial

En esta subsección y en las siguientes, utilizaremos más el término de *neurona artificial* a la unidad básica de procesamiento y no *perceptrón* para reservarlo a la construcción de Rosenblatt con la regla de aprendizaje propuesta [214]. En términos formales, una neuronal artificial es una función $NN_{\mathbf{w}}$ con parámetros $\mathbf{w}$ que recibe m entradas (o $m+1$ si consideramos el bias) y está dada por

$$NN : A \subset \mathbb{R}^m \to \mathbb{R} \tag{2.8}$$

$$\mathbf{x} \mapsto F(\mathbf{w} \cdot \mathbf{x}), \tag{2.9}$$

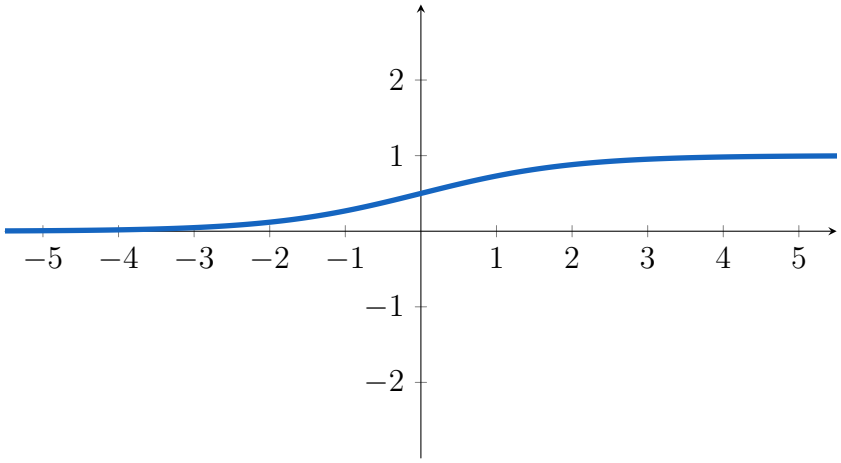
donde F es una función de activación, típicamente no decreciente. El subconjunto A puede ser todo $\mathbb{R}^m$ o estar acotado, por ejemplo $A = [0,1]^m$. Tales cotas pueden ser

entendidas como el disparo mínimo (usualmente 0) así como la tasa máxima de disparo para la cota superior. Aún más, podríamos considerar un modelo discreto con $A = \{0, 1\}$ representando que la neurona esté activa o no. Algunas funciones de activación son las siguientes [4]:

Función logística o sigmoide: Usualmente representada con $\sigma(t)$ está dada por

$$\sigma(x) = \frac{1}{1 + e^x}. \tag{2.10}$$

Esta función es considerada como "biológicamente plausible" [4, 51], por lo que será considerada como especialmente relevante. Esto está relacionado con la salida acotada de las neuronas, de forma que el dominio de la neurona artificial sea el mismo en todas las unidades. Sin embargo, no es ideal para su aplicación en métodos como Descenso de Gradiente, puesto que si $|x| \to \infty$ entonces $\sigma'(x) \to \infty$, lo cual conduce al problema del desvanecimiento de los gradientes, produciendo un aprendizaje lento.



Función Softmax: Esta función modela el proceso de inhibición lateral [98], siendo frecuentemente utilizado en la capa de clasificación [2]. La función Softmax, está dada por

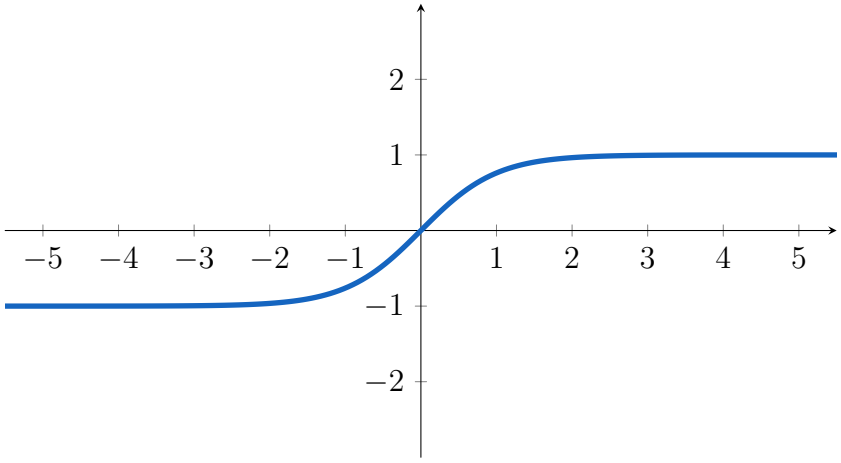
$$s_C = \frac{e^{n_C}}{\sum_{C' \in L} e^{n_{C'}}}, \tag{2.11}$$

donde C es el índice de una clase específica, n_C la neurona de salida de esta clase y L es el conjunto de índices de clase.

Función tangente hiperbólica: La función tangente hiperbólica tanh es similar a una logística escalada con un conjunto imagen de $(-1, 1)$, el cual es ideal en algunas aplicaciones (véase *Long Short-Term Memory*). Está dada por

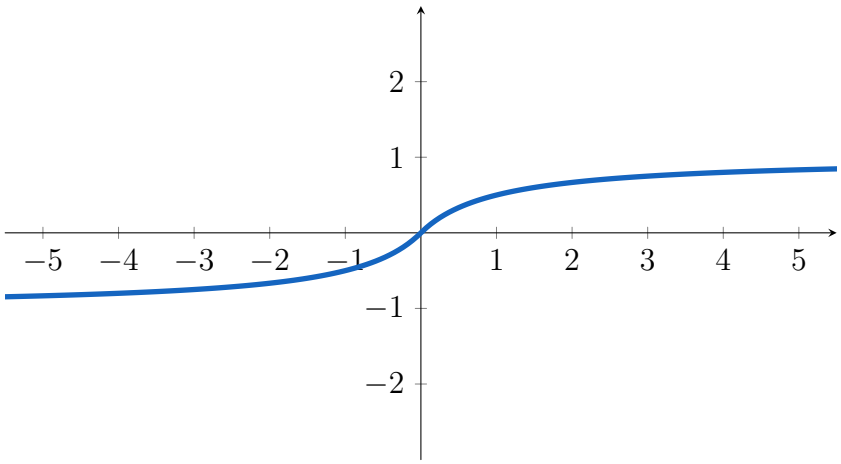
$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{2.12}$$

Esta función presenta el mismo problema de desvanecimiento de gradiente que la función sigmoide.



Función Softsign: Esta función es muy similar a la tangente hiperbólica, aunque cuenta con la ventaja de ser más eficiente en términos computacionales. Está dada por

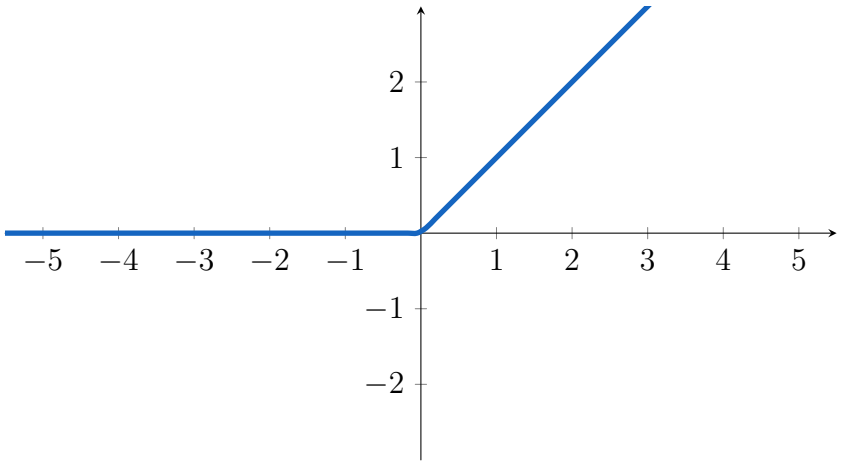
$$F(x) = \frac{x}{1 + |x|}. \tag{2.13}$$



Unidad lineal rectificada: La unidad lineal rectificada o *Rectified Linear Function Unit* (ReLU) fue implementadas exitosamente en la AlexNet [133] y como función de activación está dada por la siguiente expresión

$$\text{ReLU}(x) = \text{máx}(0, x). \tag{2.14}$$

Esta función es ideal para aplicar métodos basados en el gradiente, ya que su derivada no tiende a 0, en los puntos donde es derivable. Esta derivada es correspondiente a la función umbral (con umbral igual a 0), la cual se utiliza en el descenso de gradiente, a pesar de no ser derivable en 0. Al no tender a 0 en los extremos, como en el caso de la sigmoide, los valores muy grandes o muy negativos no arrojan un valor cercano a 0 en la derivada, lo cual produce pasos pequeños en el descenso del gradiente y una eventual convergencia muy lenta.



Esta función fue propuesta por Hahnloser *et al* en el 2000 [81]. El modelo de Tasa de Disparo manejado por estos autores es ligeramente diferente al presentado de manera previa:

$$\tau(y)\frac{dy}{dt} = -y + \text{ReLU}(b + \mathbf{w} \cdot x). \tag{2.15}$$

La τ presentada en este modelo depende de la salida de la neurona postsináptica, cuyo valor y representa la corriente de salida, mientras que b representa la corriente de entrada, dándole una interpretación al bias. La función de activación que propone es la que actualmente se conoce como ReLU. Esto quiere decir que la función ReLU apareció (sin este

nombre) en un contexto de Neurociencias, pero en el 2010, Nair y Hinton [183] las introdujeron en contextos de aprendizaje automático (Máquinas de Boltzmann Restringidas), dándole el nombre actual, aunque centrándose en las ReLUs con ruido (*Noisy Rectified Linear Unit* o NReLU), cuya función de activación está dada por

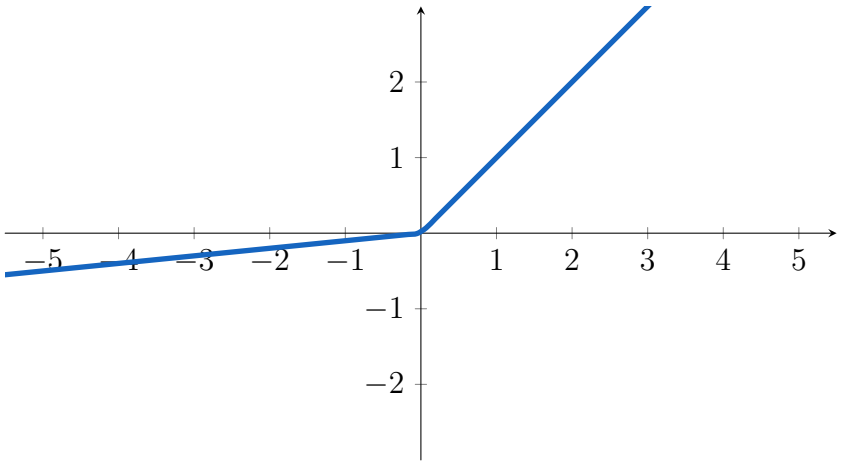
$$\text{NReLU}(x) = \text{máx}(0, x + N(0, \sigma(x))), \tag{2.16}$$

donde $N(\mu, v)$ es ruido gaussiano con varianza v y media μ, y σ representa a la función logística. Sin embargo, la ReLU sería implementada en la AlexNet. En general, las comparaciones realizadas con respecto a la función sigmoide son positivas para la ReLU en redes profundas [50]. En los últimos años, el uso de la función de activación ReLU ha sido ampliamente aceptado y por lo general se usan en las capas intermedias a las que se le añade una capa de clasificación con activación Softmax, aunque existen propuestas para utilizarla incluso en la capa de clasificación seguida de la función argumento máximo [2].

Función Leaky ReLU. La funcion *Leaky ReLU* fue propuesta originalmente por [164] y está dada por

$$\text{LReLU}(x) = \text{máx}(\alpha x, x), \tag{2.17}$$

donde $0 < \alpha < 1$ (en el artículo original $\alpha = 0{,}01$).



Exponential Linear Unit: La Unidad Exponencial Lineal es una versión suavizada de la ReLU fue propuesta por [43] y está dada por

$$ELU(x) = \begin{cases} \alpha(e^x - 1) & x < 0 \\ x & x \geq 0 \end{cases}. \tag{2.18}$$

Softplus: Esta función es otra versión suavizada de la ReLU pero derivable en todos los puntos y con rango de $(0, \infty)$. La expresión matemática de la *Softplus* es

$$F(x) = \ln(1 + e^x) \tag{2.19}$$

Como se verá, una variante de esta función también tiene sustento biológico (ver *Noisy Softplus*).

Además de las funciones derivables, consideraremos la función umbral

$$F_\theta(x) = \begin{cases} 1 & x > \theta \\ 0 & x \leq \theta \end{cases}. \tag{2.20}$$

Esta función forza a las neuronas a tener un conjunto imagen discreto, lo cual puede ser ideal para entender a las neuronas en dos estados únicos.

Además de la función logística, otra función biológicamente inspirada es la *Noisy Softplus* propuesta por [158], la cual ha sido considerada para *Spiking Neural Networks* y está dada por la siguiente expresión:

$$f_{ns}(x, \sigma) = k\sigma \log\left(1 + e^{\frac{x}{k\sigma}}\right), \tag{2.21}$$

donde el parámetro σ define el nivel de ruido y k controla la forma de la curva. x representa en este contexto la corriente promedio de entrada. Es interesante notar que la derivada de esta función de activación es la sigmoide escalada

$$\frac{\partial f_{ns}(x, \sigma)}{\partial x} = \frac{1}{1 + e^{\frac{-x}{k\sigma}}}. \tag{2.22}$$

2.1.3. Aprendizaje semántico por compuertas lógicas

Un ejemplo de cómo los perceptrones simples pueden ser entendidos (o aplicados) aparece en [216]. En sus orígenes, la Lógica se posicionó como una de las primeras maneras

para formalizar el pensamiento humano y eso es palpable en títulos como la obra magna de George Boole, *The laws of thought* [26]. Los primeros trabajos en Inteligencia Artificial y Redes Neuronales tuvieron una fuerte carga lógica, tal como el trabajo de McCullogh-Pitts [173][7]. Si bien la Lógica puede parecer en cierto punto artificial (ya que muchos de los lenguajes que estudia son lenguajes formales) existen aproximaciones más próximas al pensamiento natural como la Lógica Difusa [7].

Una dirección interesante es tratar de determinar la relación entre la lógica y las redes neuronales para tratar de construir formas de razonamiento artificial mediante las redes neuronales. Primeramente consideremos las compuertas lógicas, NOT, AND y OR, o bien, los símbolos lógicos de negación, conjunción y disyución. Las tablas de verdad de la negación, conjunción y disyunción son las siguientes:

x_1	$\neg x_1$
1	0
0	1

Cuadro 2.1: Tabla de verdad de la negación

x_1	$x_1 \wedge x_2$	x_2
1	1	1
1	0	0
0	0	1
0	0	0

Cuadro 2.2: Tabla de verdad de la conjunción

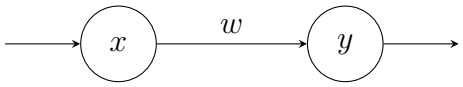
Para la negación, únicamente se necesita una entrada pero también una neurona bias. El conjunto de datos está dado por $(1,0),(0,1)$. En caso de no utilizar la neurona bias, se tienen problemas para generalizar la negación ya que sea w el peso de la neurona con activación x, entonces la neurona de salida $y = F(wx)$. Luego $1 = F(0)$ y $0 = F(w)$ de

[7]En parte esta situación debió deberse la relación existente entre Computación y Lógica Matemática, la primera surgiendo debido a los avances en el campo de la segunda durante la década de 1930.

x_1	$x_1 \vee x_2$	x_2
1	1	1
1	1	0
0	1	1
0	0	0

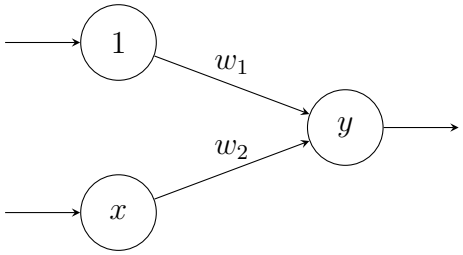
Cuadro 2.3: Tabla de verdad de la disyunción inclusiva

donde $w < 0$. Si la función que tenemos es la umbral F_θ, con $w < \theta < 0$ podemos construir la activación de forma que $F(w) = 0$ y $F(0) = 1$. Un diagrama está expuesto abajo

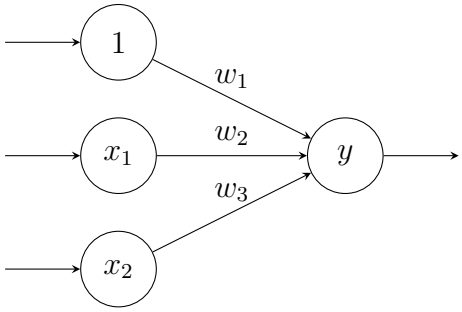


Así $\theta < 0$. Sin embargo, si en lugar de recibir esos datos, se tiene $\{(0,0),(1,1)\}$ como conjunto de datos entonces la función umbral F_θ con $\theta < 0$ no funciona, ya que $F(0) = 1$ siempre. En general, el problema de carecer de una neurona bias es que si $\theta = 0$ entonces $F(x) = 1$ si $x > 0$, lo cual impide generalizar $(1,0)$ (que siempre será error); si $\theta > 0$ entonces no podemos encontrar una red simple que aprenda la negación y si $\theta < 0$ entonces se tiene el otro problema mencionado.

Una posible salida, probablemente sin sustento natural pero sí matemático, es utilizar neuronas bias, es decir, una neurona siempre encendida. De esta forma modelamos la siguiente red para resolver el problema



Si fijamos $\theta = 0{,}5$ entonces $w_1 = 1$ y $w_2 = -1$ funcionan pues $F(-1(0)+1) = F(1) = 1$ y $F(-1(1)+1) = F(0) = 0$, lo cual es deseado. Para las conjunciones y disyunciones se puede utilizar la siguiente red.



Utilizando $\mathbf{w} = (-1, 1, 1)$ se modela la conjunción y con $(0, 1, 1)$ modelamos la disyunción, la cual no utiliza la neurona bias. También se puede omitir la neurona bias en el caso de la conjunción utilizando $\mathbf{w} = (0, 0{,}3, 0{,}3)$.

En términos generales, esto tiene implicaciones interesantes acerca de la relación de las redes neuronales con la Lógica Proposicional. Muchos aspectos pueden definirse a partir de una serie de características (*features*) y aplicando conjunciones o disyunciones, de forma que definimos una clase por medio de propiedades que necesariamente tiene, propiedades que puede tener y propiedades que no tiene. Definimos un vector de características $\mathbf{x}^{'} \in [0, 1]^m$, donde $x_i^{'} = 0$ y $x_i^{'} = 1$ representan no tener la característica y tenerla, respectivamente. Para no emplear bias, redefinimos $\mathbf{x} \in \mathbb{R}^{2m}$ como

$$x_i = \begin{cases} x^{'}_{\frac{i+1}{2}} & impar(i) \\ 1 - x^{'}_{i/2} & par(i) \end{cases}. \tag{2.23}$$

Esto significa que $\mathbf{x}$ es una versión extendida del vector original con las neuronas inversas. Sea I_1 los índices de las características extendidas que la clase tiene, I_2 los índices de las características extendidas que la clase puede tener e I_3 la características que no puede tener. Entonces una expresión lógica definitoria de la clase a partir de sus características está dada por

$$\bigwedge_{i \in I_1} x_i \wedge \bigvee_{i \in I_2} x_i \wedge \bigwedge_{i \in I_3} \neg x_i. \tag{2.24}$$

Proposición 1. *Sea F_θ la función de activación con umbral $\theta > 0$. Supongamos que una clase está definida por la expresión 2.24 y supongamos también que la expresión es satisfacible. Entonces existe una red neuronal NN de una capa y con la función umbral F_θ tal que $NN(\boldsymbol{x}) = 1$ si $\boldsymbol{x}$ satisface la definición de la clase y $NN(\boldsymbol{x}) = 0$ en caso contrario.*

Demostración. Sea $n_1 = |I_1|$ y $0 < \varepsilon < \frac{\theta}{n_1}$. Definamos

$$w_i = \begin{cases} \frac{\theta+\varepsilon}{n_1} & i \in I_1 \\ 0 & i \in I_2 \\ -2\theta & i \in I_3 \end{cases}. \tag{2.25}$$

Entonces supongamos que $\mathbf{x}$ satisface la definición lógica. Entonces $x_i = 1$ para toda $i \in I_1$ y $x_i = 0$ para toda $i \in I_3$, de donde

$$NN(\mathbf{x}) = F_\theta\left(\sum_{i\in I_1} w_i\right) \tag{2.26}$$

$$= F_\theta\left(\sum_{i\in I_1} \frac{\theta+\varepsilon}{n_1}\right) \tag{2.27}$$

$$= F_\theta(\theta+\varepsilon) = 1. \tag{2.28}$$

Ahora supongamos que $\mathbf{x}$ no satisface la definición. Entonces se tienen los siguientes casos:

Caso 1: $\bigwedge_{i\in I_1} x_i$ es falso.

Entonces $\bigvee_{i\in I_1} \neg x_i$ es verdadero. De ahí $x_i = 0$ para alguna $i \in I_1$. Por lo tanto

$$\sum_{i\in I_3} w_i x_i \leq (n_1 - 1)\frac{\theta+\varepsilon}{n_1} \tag{2.29}$$

$$= \frac{n_1-1}{n_1}\theta + \frac{n_1-1}{n_1}\varepsilon \tag{2.30}$$

$$< \frac{n_1-1}{n_1}\theta + \frac{n_1-1}{n_1}\frac{\theta}{n_1} \tag{2.31}$$

$$< \frac{n_1-1}{n_1}\theta + \frac{\theta}{n_1} \tag{2.32}$$

$$= \theta. \tag{2.33}$$

Como $\sum_{i\in I_3} w_i x_i \leq 0$ y $\sum_{i\in I_2} w_i x_i = 0$ entonces $\mathbf{w}\cdot\mathbf{x} < \theta$, por lo que $NN(\mathbf{x}) = 0$.

Caso 2: $\bigvee_{i\in I_2} x_i$ es falso.

Entonces $\bigwedge_{i\in I_2} \neg x_i$ se sostiene, por lo que $x_i = 0$ para toda $i \in I_2$. Si existe $j \in I_2$ tal que $x_j = \neg x_i$ con $i \in I_2$ entonces $x_i \wedge \neg x_i$ se sostiene, lo cual es contradictorio. Si existe

$j \in I_3$ tal que $x_j = \neg x_i$ para alguna $i \in I_2$ entonces $x_j = 1$ de donde $\mathbf{w} \cdot \mathbf{x} < \theta$ ya que $\sum_{i \in I_1 \cup I_2} w_i x_i \leq \theta + \varepsilon$ y $\sum_{i \in I_3} w_i x_i \leq -2\theta$, de donde

$$\mathbf{w} \cdot \mathbf{x} = \sum_{i \in I_1 \vee I_2 \cup I_3} w_i x_i \tag{2.34}$$

$$\leq \varepsilon - \theta < 0 < \theta. \tag{2.35}$$

Por lo tanto, $NN(\mathbf{x}) = 0$.

Si existe $j \in I_1$ tal que $x_j = \neg x_k$ para alguna $k \ \in I_2$ entonces $x_j = 1$. Sin embargo, se tiene una subexpresión de la forma

$$x_j \wedge \bigvee_{i \in I_2} x_i. \tag{2.36}$$

Por las Leyes de De Morgan,

$$\bigwedge_{i \in I_2} x_j \cap x_i, \tag{2.37}$$

por lo que $\neg x_k \wedge x_k$ se sostiene, lo cual contradice que la expresión sea satisfacible.

Caso 3. $\bigwedge_{i \in I_3} \neg x_i$ es falso.

Entonces $\bigvee_{i \in I_3} x_i$. Por lo que existe $j \in I_3$ tal que $x_i = 1$. Por lo tanto $w_j x_j = -2\theta$ y luego $\sum_{i \in I_3} w_i \leq -2\theta$. Como $\sum_{i \in I_1 \cup I_2} w_i x_i \leq \theta + \varepsilon$ entonces

$$\sum_{i \in I_1 \cup I_2 \cup I_3} w_i x_i \leq \varepsilon - \theta < 0 < \theta. \tag{2.38}$$

Por lo tanto $\mathbf{w} \cdot \mathbf{x} < \theta$ y luego $NN(\mathbf{x}) = 0$.

□

Para términos prácticos, esta proposición puede utilizarse para definir objetos con características muy precisas que permiten la clasificación de clases. Un ejemplo de ello, utilizando la expresión 2.24, está dado por la tabla 2.1.3, de una estructura mínima que llamaremos como el *Microuniverso* con las características

1. Teclas

2. Sin teclas
3. Cilíndrico
4. No cilindríco
5. Pantalla
6. Sin pantalla
7. Tinta
8. Sin tinta
9. Sirve para comunicación
10. No comunica
11. Tiene páginas
12. Sin páginas
13. Escribe
14. No escribe

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Celular	1	1	0	1	1	0	0	1	1	0	0	1	1	0
Calculadora	1	0	0	1	1	0	0	1	0	1	0	1	1	0
Pluma	0	1	1	0	0	1	1	0	1	0	0	1	1	0
Lápiz	0	1	1	0	0	1	0	1	1	0	0	1	1	0
Libro	0	1	0	1	0	1	1	0	1	0	1	0	0	1

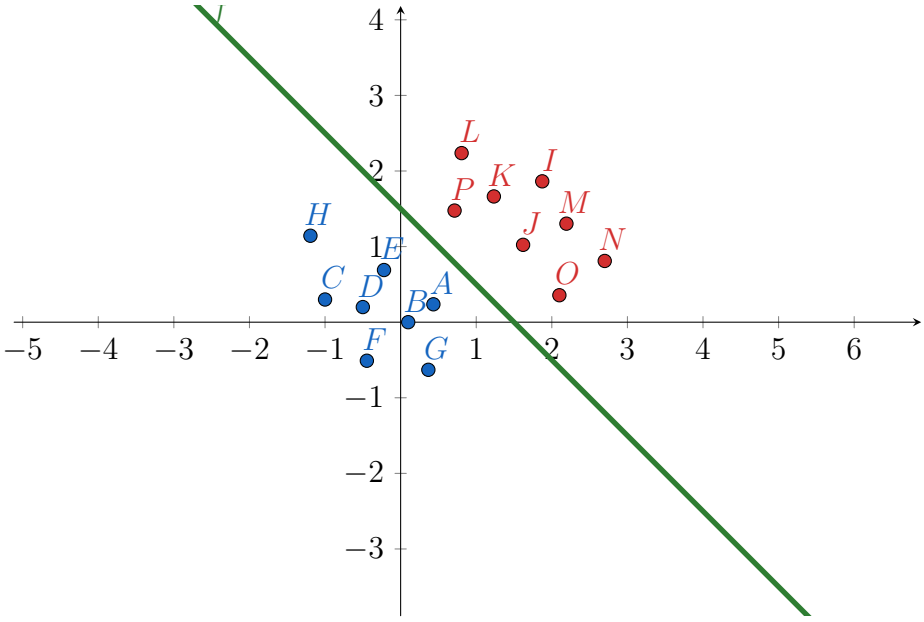
Cuadro 2.4: Descriptores semánticos

Para el caso de la categoría de los teléfonos celulares, $x_1, x_2 \in I_2$, $x_4, x_5, x_8, x_{12} \in I_1$ y los restantes están en I_3. Así, aplicando la Proposición anterior, podemos generar una red neuronal que recibe los datos de entrada específicos.

Estos primeros ejemplos nos brindan posibilidad de aplicar redes neuronales sin aún implementar un algoritmo de optimización.

2.2. Redes Multicapa: Feedforward Neural Networks

Las redes neuronales simples presentadas en los capítulos anteriores presentan un gran dinamismo y pueden ser explotadas ampliamente para resolver múltiples problemas de clasificación. En la práctica, la clasificación se da cuando $F(\mathbf{w} \cdot \mathbf{x}) \geq \phi$, etiquetando a la entrada como perteneciente a la clase 1, y si $F(\mathbf{w} \cdot \mathbf{x}) < \phi$ entonces le asignamos la clase 0 o no pertenencia a la clase en cuestión. Suponiendo que la función de activación es creciente y ϕ pertenece a su imagen, entonces sea $\theta = F^{-1}(\phi)$ con la función restringida a su imagen. Por lo tanto si $\mathbf{w} \cdot \mathbf{x} < \theta$ entonces $\mathbf{x}$ pertenece a la clase 0 y en caso contrario, pertenece a la clase 1. $\mathbf{w} \cdot \mathbf{x}$ es una ecuación de un hiperplano en $\mathbb{R}^m$, por lo que el método es un separador lineal. Gráficamente, un ejemplo de un conjunto linealmente separable en $\mathbb{R}^2$ se presenta a continuación.



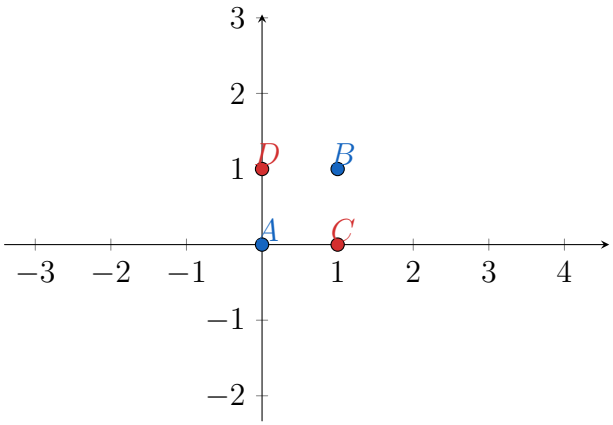
Las puertas lógicas presentadas anteriormente son linealmente separables, permitiendo que puedan ser expresables mediante una red neuronal. Dada la relación estrecha que supusieron las redes neuronales con la lógica, existía cierto interés en expresar las funciones

lógicas como redes neuronales. No obstante, la disyunción exclusiva (puerta XOR) representó un problema fuerte para las incipientes redes neuronales [3]. La tabla de verdad de la disyunción exclusiva, está dada por

x_1	$x_1 \veebar x_2$	x_2
1	0	1
1	1	0
0	1	1
0	0	0

Cuadro 2.5: Tabla de verdad de la disyunción exclusiva

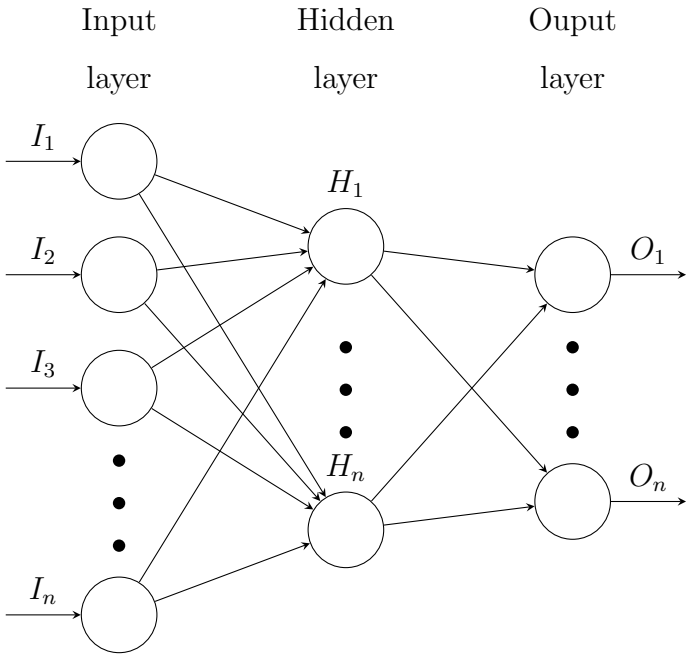
Geométricamente, podemos expresarla del siguiente modo. Sea $A = (0, 0), B = (1, 1), C = (1, 0), D = (0, 1)$. Sea $\{(A, 0), (B, 0), (C, 1), (D, 1)\}$ el conjunto de datos para XOR. Entonces una representación gráfica es la siguiente:



No existe recta alguna $w_0 + w_1x_1 + w_2x_2$ que separe este conjunto, puesto que en caso contrario, $w_0 < \theta$, $w_0+w_1+w_2 < \theta$ de donde $2w_0+w_1+w_2 < \theta$ pero $w_0+w_1 \geq \theta$ y $w_0+w_2 \geq \theta$ de donde $2w_0 + w_1 + w_2 \geq \theta$, lo cual es absurdo. Esto confirma analíticamente nuestra suposición geométrica y muestra la imposibilidad de lograr representar una compuerta lógica en una red neuronal simple.

Una solución a este problema está dada por el diseño de redes neuronales de varias capas, es decir, acoplando varias neuronas simples y disponiéndolas en capas. Para ello, se dispone de una capa de entrada (*input layer*), una capa de salida (*output layer*) y

un número arbitrario de capas intermedias u ocultas (*hidden layers*). En el diagrama inferior representamos un esquema de una *red neuronal de alimentación hacia adelante completamente conectada*, esto es, una *Feedforward Neural Network* o FNN (llamada así si la red no admite conexiones entre sí) de tipo *Fully conected*, esto es, si todas las capas están conectadas.



Esta tipo de arquitectura de red neuronal presenta muchas ventajas y mayor capacidad de abstracción que las neuronas simples. Matemáticamente, una red neuronal FNN es una composición de varias neuronas artificiales. Cada capa está organizada por un vector de neuronas con estados $\mathbf{v}_j$. Entre capa y capa, existe una matriz de pesos $\mathbf{W}_j$ de forma que

$$\mathbf{v}_j = F(\mathbf{W}_j \mathbf{v}_{j-1}) \tag{2.39}$$

donde F es un vector de funciones de activación. Por ende, una red FNN es una composición de varias capas de neuronas artificiales.

La disposición en capas resuelve el problema planteado originalmente sobre la compuerta XOR, y aún más, de cualquier expresión lógica. Esto puede enunciarse en la siguiente proposición, utilizando el hecho de que $\{\neg, \vee\}$ es completo (también se puede utilizar $\{\neg, \wedge\}$ para tal fin).

Proposición 2. *Toda expresión lógica puede ser calculada con una red neuronal FNN.*

Demostración. En el Teorema 15D de [64] se enuncia que $\{\neg, \vee\}$ es completo. Esto quiere decir que toda expresión en Lógica proposicional puede expresarse bajo composiciones de $\{\neg, \vee\}$, y dado que ambos conectivos pueden expresarse como una neurona articial, entonces cualquier enunciado puede expresarse como una red FNN. □

Radial Basis Function Neural Networks

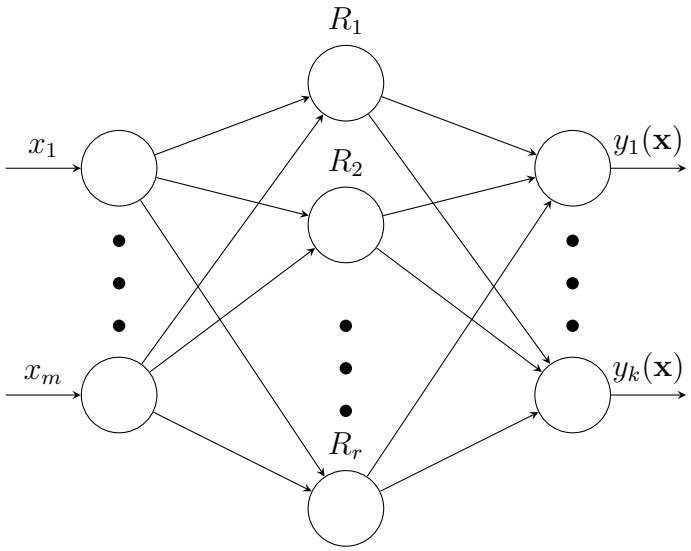
Discutiremos a continuación un tipo especial de red neuronal artificial conocido como *Radial Basis Fuction Neural Network* o RBFNN [78], originalmente propuestos en [29]. Se trata de una red de una capa intermedia de N unidades especiales con las siguientes operaciones

$$R_i(\mathbf{x}) = B_i\Big(\frac{||\mathbf{x} - \mathbf{c}_i||}{\sigma_i}\Big) \tag{2.40}$$

donde $\mathbf{x} \in \mathbb{R}^m$ es el vector de entrada, $\mathbf{c}_i \in \mathbb{R}^m$ es el vector central del nodo. Tales centros pueden escogerse aleatoriamente, aunque existen técnicas para seleccionarlos de la mejor manera [34]. Usualmente, la base utilizada es una función gaussiana, de modo que $R_i(\mathbf{x}) = e^{-\frac{||\mathbf{x}-\mathbf{c}_i||^2}{\sigma^2}}$. La salida es propiamente una red neuronal con pesos w_{ij} está dada por

$$y_j = \sum_{i=1}^{N} w_{ji} R_i(\mathbf{x}) \tag{2.41}$$

La red neuronal no incluye bias para no incrementar la complejidad. Un diagrama de una RBFNN está provisto abajo



2.2.1. Teoremas de Aproximación Universal

La proposición 2 aborda un hecho de las redes neuronales que puede generalizarse aún más. En esta sección abordaremos los Teoremas de Aproximación Universal que involucran algunas de las propiedades matemáticas más importantes de las Redes Neuronales FNN. Básicamente dicho teoremas muestran que es posible aproximar funciones mediante redes neuronales FNN completamente conectadas utilizando funciones de activación no polinomiales. Enunciaremos aquí algunos de los resultados más relaventes.

Teorema de Cybenko

El Teorema de Cybenko [49] es el primero de los importantes teoremas de aproximación. Básicamente, afirma que una red neuronal de una capa intermedia puede aproximar cualquier función $f : \mathbb{R}^m \to I_m$, donde $I_m = [O, 1]^m$. Este teorema es válido para funciones sigmoidales, que es una generalización a la activación sigmoide. En términos históricos, debido a su inspiración biológica, la función sigmoide era importante hacia 1989. Como hemos visto anteriormente, la popularización de la ReLU no se dio hasta su empleo en la AlexNet, por lo que no era de interés principal hacia esos años. El Teorema de Cybenko no es válido para funciones ReLU, sino únicamente para sigmoidales, definidas aquí.

Definición 1. *Una función σ se llama sigmoidal si* $\lim_{t\to+\infty} \sigma(t) = 1$ *y si* $\lim_{t\to-\infty} = 0$

El Teorema de Cybenko indica que una red neuronal con una capa intermedia de activación sigmoide y una capa de salida con pesos $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)$ (activación lineal) puede aproximar a cualquier función continua.

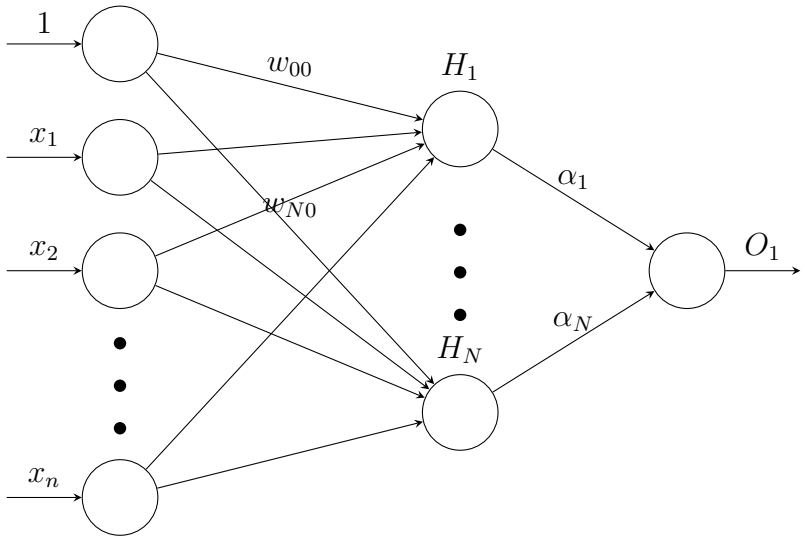
Teorema 2 (Cybenko, 1989)**.** *Sea σ cualquier función sigmoidal continua. Entonces la suma finita de la forma*

$$NN(\boldsymbol{x}) = \sum_{j=1}^{N} \alpha_j \sigma(\boldsymbol{w}_j \cdot \boldsymbol{x} + w_{j0}) \tag{2.42}$$

es denso en $C(I_m)$, esto es, para toda $f \in C(I_m)$ (el espacio de funciones continuas $\{f | f : \mathbb{R}^m \to I_m\}$) y $\varepsilon > 0$, existe una suma $NN(\boldsymbol{x})$ tal que $\forall \boldsymbol{x} \in I_m$

$$|NN(\boldsymbol{x}) - f(\boldsymbol{x})| < \varepsilon \tag{2.43}$$

La arquitectura de la red neuronal descrita por el Teorema anterior está dada por el siguiente diagrama



Teorema de Leshno-Lin-Pinkus-Schocken

Una generalización aún más fuerte del Teorema de Cybenko es el Teorema de Leshno-Lin-Pinkus-Schocken [149], que afirma básicamente que es posible construir una red neuronal completamente conectada con una capa intermedia que aproxime cualquier función

continua en $\mathbb{R}^m$ si la función de activación es no polinomial. Para ello, presentaremos algunas definiciones previas, la primera de las cuales las tomaremos de Teoría de la Medida e Integración, adaptada de [123], mientras que el resto son tomadas del artículo original de Leshno *et al.*

Definición 2. *Sea $(\Omega, \mathcal{S}, \mu)$ un espacio de medida y sea $u : \Omega \to \mathbb{R}^m$, una función medible. u está* ***esencialmente acotado*** *si existe $M > 0$ tal que*

$$\mu(\{x \in \Omega : ||u(x)|| > M\}) = 0. \tag{2.44}$$

El ***Supremo Esencial*** *es el ínfimo de tales M, dado por*

$$ess\sup_{x\in\Omega} |u(x)| = \text{ínf}\{M \in \mathbb{R} : \mu(\{x \in \Omega : ||u(x)|| > M\}) = 0\}. \tag{2.45}$$

Esta misma definición, aparece redactada en el artículo de una forma similar, pero definiendo una norma particular:

Definición 3. *Una función u definida en casi todo punto con respecto a la medida de Lebesgue ν sobre un conjunto medible Ω en $\mathbb{R}^m$ está esencialmente acotado sobre Ω si $|u(x)|$ está acotado en casi todo punto en Ω. Denotamos $u \in L^\infty(\Omega)$ con la norma*

$$||u||_{L^\infty(\Omega)} = \text{ínf}\{\lambda \mid \nu(\{x : |u(x)| \geq \lambda\}) = 0\} = ess\sup_{x\in\Omega} |u(x)|. \tag{2.46}$$

Definición 4. *Una función u definida para casi todo punto con respecto a la medida de Lebesgue sobre un abierto $\Omega \subset \mathbb{R}^m$ es esencialmente acotado en Ω si para cada compacto $K \subset \Omega$, $u \in L^\infty(K)$. Una función localmente esencialmente acotada sobre Ω se denota como $u \in L^\infty_{loc}(\Omega)$*

Definición 5. *Decimos que el conjunto F de las funciones en el espacio de funciones $L^\infty_{loc}(\mathbb{R}^m)$ es denso en $C(\mathbb{R}^m)$ y para cada conjunto compacto $K \in \mathbb{R}^m$ existe una sucesión de funciones $f_j \in F$ tal que*

$$\lim_{j\to\infty} ||g - f_j||_{L^\infty(K)} = 0. \tag{2.47}$$

Nótese que en la definición anterior se enfatiza la compacidad. Ahora podemos enunciar el teorema principal

Teorema 3 (Leshno-Lin-Pinkus-Schocken, 1993)**.** *Sea M Sea $\sigma \in M$. Sea*

$$\Sigma_m = span\ \{\sigma(\boldsymbol{w} \cdot \boldsymbol{x} + \theta) : \boldsymbol{w} \in \mathbb{R}^m, \theta \in R\}. \tag{2.48}$$

Entonces Σ_m es denso en $C(\mathbb{R}^m)$ si y sólo si σ no es un polinomio algebraico.

2.2.2. Entrenamiento de redes neuronales

Las redes neuronales artificiales cuentan con pesos adaptables que, como se ha mostrado previamente, permiten mejorar los procesos de clasificación. Anteriormente, sólo hemos considerado un método para asignar valores convenientes a los pesos para el caso de definiciones semánticas satisfacibles. Hasta ahora no hemos dado bosquejo alguno de cómo las neuronas reales modifican sus pesos, el cual corresponderá una discusión relevante en el desarrollo de esta tesis. Desde una primera instancia, podemos tratar de abordar el problema de clasificación mediante técnicas lineales. Por ejemplo, en el problema de clasificación de la conjunción, utilizando tres neuronas (incluyendo la neurona bias) con activación umbral, podemos desarrollar un sistema de inecuaciones deseable, dado por

$$w_0 + w_1 + w_2 > \theta \tag{2.49}$$

$$w_0 + w_1 \leq \theta \tag{2.50}$$

$$w_0 + w_2 \leq \theta. \tag{2.51}$$

de donde, $w_0 < 0$ y $w_1 \leq \theta + w_0$, $w_2 \leq \theta + w_0$. Encontrando valores que hagan satisfacibles los sistemas de desigualdades podemos efectuar una clasificación correcta. En la práctica, sin embargo, hemos visto que los problemas pueden no ser linealmente separables.

Otra forma, posiblemente más conveniente, consiste en minimizar funciones de costo, tal como se especificó en el capítulo 1, que es particularmente útil en contextos de aprendizaje supervisado, el cual considera un conjunto de datos (*dataset*) $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ donde el

vector $\mathbf{x}_i \in \mathbb{R}^m$ es el vector de características y y es la etiqueta o clase. Entonces se desea que la red neuronal NN reduzca una función de costo asociada. En general se desea que la función $NN_{\mathbf{w}}$ calcule adecuadamente la clase correcta a partir del vector de características, de forma que $NN_{\mathbf{w}}(\mathbf{x}) = y$. Una familia de funciones de costo utilizables están dadas por cualquier distancia dada por

$$d((NN_{\mathbf{w}}(\mathbf{x}))_{i=1}^{n}, (y_i)_{i=1}^{n}). \tag{2.52}$$

Un ejemplo está dado por la Suma de Errores Cuadrados, que es el cuadrado del Error Cuadrático Medio o la distancia euclidiana, esto es:

$$SSE = \sum_{i=1}^{n} (NN_{\mathbf{w}}(\mathbf{x}_i) - y_i)^2. \tag{2.53}$$

No obstante, la función de costo más común en el ámbito de redes neuronales es la Entropía Cruzada [182].

Dado que la red neuronal tiene un vector de pesos asociados, idealmente se desearía encontrar

$$\arg\min_{\mathbf{w}} Costo((NN_{\mathbf{w}}(\mathbf{x}))_{i=1}^{n}, (y_i)_{i=1}^{n}). \tag{2.54}$$

Es posible generalizar estas ideas, expuestas para una sola capa, para el problema de múltiples capas. Para reducir las funciones de costo, se puede emplear cualquier algoritmo de optimización, incluyendo de primer orden (métodos basados en el gradiente) como de segundo orden (Método de Newton) [19]. Para la optimización de primer orden, se ha desarrollado la técnica de retropropagación (*backpropagation*) en el caso de las redes multicapa, la cual parece ser la más ampliamente utilizada en el entrenamiento de redes neuronales (véase [4, 3]).

El uso del enfoque del gradiente, frente a otros, queda respaldado por su empleo en numerosos artículos de relevancia, como por ejemplo en el desarrollo de las redes neuronales convolucionales. Desde su inicio, la propuesta de redes convolucionales ha estado relacionada con los métodos basados en el gradiente [146]. Esta tendencia continuó en el desarrollo de la AlexNet [133], en la cual se empleó Descenso de Gradiente Estocástico

(SGD) + Momentum con un *batch* de 128 ejemplos, momentum de 0,9 y decaímiento de pesos de 0,0005, utilizando las siguientes reglas recursivas:

$$v_{i+1} = 0{,}9v_i - 0{,}0005\alpha w_i - \alpha\langle\frac{\partial L}{\partial w}|_{w_i}\rangle_{D_i} \tag{2.55}$$

$$w_{i+1} = w_i + v_{i+1}, \tag{2.56}$$

donde $\langle\frac{\partial L}{\partial w}|_{w_i}\rangle_{D_i}$ es el promedio de las derivadas evaluadas en w_i de la función de costo L, α es la tasa de aprendizaje. La inicialización de los pesos fue aleatoria, utilizando una distribución gaussiana con media 0. El caso de arquitecturas siguientes como las VGGs [228], mantuvo un esquema similar para el algoritmo de aprendizaje. Arquitecturas posteriores siguieron utilizando métodos basados en el gradiente, tal como se muestra en la tabla 2.6.

Arquitectura	Optimizador
AlexNet [133]	SGD + Momentum
VGGs [228]	SGD + Momentum
InceptionV3 [240]	SGD + Momentum, RMSProp
ResNet [87]	SGD + Momentum
InceptionResNetV2 [238]	SGD + Momentum, RMSProp
MobileNet [103]	RMSProp
Xception [40]	SGD + Momentum, RMSProp
DenseNet [105]	SGD + Nesterov Momentum
MobileNetV2 [219]	RMSProp

Cuadro 2.6: Métodos de optimización utilizados en algunas arquitecturas convolucionales notables

Todos los optimizadores utilizados en estas arquitecturas convolucionales, de las cuales se abordará con mayor detalle en el siguiente capítulo, utilizan algún método basado en el gradiente. Existe una transición de SGD + Momentum hacia RMSProp, que en casos como InceptionResNetV2 muestran mejores resultados. Sin embargo, en el contexto de redes neuronales, algunos *benchmarks* favorecen a propuestas como Adam [125]. En el

caso de redes recurrentes, el desarrollo de la retropropagación en el tiempo y las LSTMs manifiestan la importancia del enfoque del gradiente en redes neuronales distintas.

Frente a la hegemonía de los métodos basados en el gradiente, existen escasas alternativas además de la optimización de segundo orden. Una opción son los algoritmos genéticos. Por ejemplo, para el entrenamiento de redes convolucionales en el contexto de reconocimiento de caracteres Devanagari, [246] utilizó tanto Algoritmos Genéticos como L-BFGS (Limited Memory Broyden-Fletcher-Goldfarb-Shanno). Abordaremos más a detalle estos algoritmos en la sección correspondiente a Redes Neuronales Evolutivas, en donde defenderemos su uso para la optimización de hiperparámetros y no de pesos. Posteriormente, introduciremos a la Regla de Hebb, otro enfoque para la optimización paramétrica de redes neuronales. La discusión entre cómo incorporar a este método frente al uso de los optimizadores basados en el gradiente constituye uno de los ejes principales de esta tesis.

2.3. Redes Recurrentes

Hasta ahora solamente hemos considerado modelos de redes por capas, los cuales describen muy bien algunas entradas sensoriales (específicamente la visual). Un modelo menos simplificado de las conexiones interneuronales es el model de Redes Recurrentes, o mejor conocido como *Recurrent Neural Networks* o RNN. El modelo de Red Neuronal Recurrente (RNN) ha sido diseñado para el procesamiento de datos sequenciales como textos y series de tiempo, siendo ideales para problemas de NLP [3].

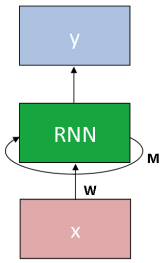


Figura 2.1: Esquema básico de una red recurrente. Tenemos un vector de entrada **x** que junto con la propia salida de la RNN forma parte de las entradas de la misma.

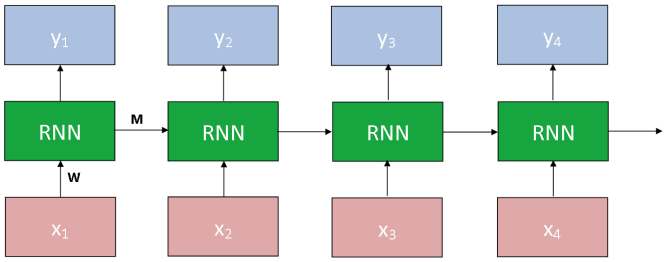


Figura 2.2: Red Recurrente desglosada. En este caso podemos observar que se ingresa una sucesión de datos de entrada $\mathbf{x}_1, \cdots$, y otra sucesión de salida. Un ejemplo de ello es la traducción automática, donde tenemos palabras que forman parte de la entrada y otras palabras conforman la salida.

Introduciremos el modelo recurrente desde el punto de vista de la Neurociencia Teórica, abordado por [51]. Consideremos a las activaciones de las neuronas de entrada con los valores $\mathbf{x} \in \mathbb{R}^{N_x}$, y salida $\mathbf{h} \in \mathbb{R}^{N_h}$ con la matriz de pesos asociados $\mathbf{W} \in \mathbb{R}^{N_x} \times \mathbb{R}^{N_h}$. El modelo recurrente considera que las neuronas de salida con tasas de disparo $\mathbf{h}$ están conectadas entre sí, cuyas conexiones están dadas por la matriz de pesos $\mathbf{M} \in \mathbb{R}^{N_h} \times \mathbb{R}^{N_h}$. De forma similar al caso de las redes FNN, los valores de activación de la salida están dados por:

$$\boldsymbol{\tau}_r \frac{d\mathbf{h}}{dt} = -\mathbf{h} + \mathbf{F}(\mathbf{W}\mathbf{x} + \mathbf{M}\mathbf{h}), \tag{2.57}$$

donde $F : \mathbb{R}^{N_h} \to \mathbb{R}^{N_h}$ es tal que $F_i : \mathbb{R} \to \mathbb{R}$ es una función de activación. Una discretización de este modelo biológico, similar al caso de las redes FNN, es considerar $\boldsymbol{\tau}_r = \mathbf{0}$, de donde

$$\mathbf{h}(t) = \mathbf{F}(\mathbf{W}\mathbf{x}(t-1) + \mathbf{M}\mathbf{h}(t-1)). \tag{2.58}$$

La ecuación anterior es la base del modelo recurrente, que ha sido construido a partir de un modelo extraído de la Neurociencia Teórica. Adicionalmente, las neuronas con estado $\mathbf{h}$ generalmente tienen una salida $\mathbf{y}$, pasando a ser una capa oculta, de forma que $\mathbf{h}$ se le conoce como el estado interno. La salida tiene unos pesos asociados $\mathbf{W}_y$ y una función de activación $\mathbf{F}_y$ asociada de forma que

$$\mathbf{y}(t) = \mathbf{F}_y(\mathbf{W}_y\mathbf{h}(t)). \tag{2.59}$$

Modificando los nombres a los pesos de x como $\mathbf{W}_x$ y a los de h como $\mathbf{W}_h$, podemos reescribir 2.58 como

$$\mathbf{h}(t) = \mathbf{F}_h(\mathbf{W}\mathbf{x}(t-1) + \mathbf{W}_h\mathbf{h}(t-1)). \tag{2.60}$$

A las redes de este tipo se les conoce como *Redes de Elman* [229] por el trabajo del linguista Jeffrey Elman [61]. Similarmente, podemos conectar la salida directamente al estado interno, modificando a la ecuación 2.60 como

$$\mathbf{h}(t) = \mathbf{F}_h(\mathbf{W}\mathbf{x}(t-1) + \mathbf{W}_h\mathbf{y}(t-1)). \tag{2.61}$$

A este tipo de arquitecturas se les conoce como *redes de Jordan* en honor a Michael I. Jordan [119] y en conjunto forman para de las Redes Recurrentes Simples. Una forma gráfica de las RNNs está expresada en la imagen 2.1, que puede ser desglosada en la 2.2.

Long Short Term Memory

Los entrenamientos de RNNs tienen problemas asociados al desvanecimiento y exposión de gradientes. En términos prácticos, las RNNs simples poseen una buena memoria a corto plazo pero una pobre memoria a largo plazo [3].Una solución a este problema aparece con la introducción de las Redes con Memoria a Corto y Largo Plazo o *Long Short-Term Memory* (LSTM) propuesta en [96]. Seguiremos la conceptualización basada en [229] y la notación basada en el artículo de [251] (véase figura 2.3).

Las Redes con LSTM cuentan con unidades (Unidades LSTM) que tienen operaciones intermedias. Funciona mediante compuertas (*gates*) que deciden qué información pasar y cuál olvidar. La *compuerta de olvido* (*forget gate*) controla cuánto de la entrada ponderada y cuánto del estado interno anterior debe ser olvidado. Para ello, utiliza una activación sigmoide (σ) cuya imagen es el intervalo $(0, 1)$ de forma que si devuelve un número cercano a 0 se aplique olvido y 1 en el caso extremo. Así, la función de olvido $\mathbf{f}(t)$ está dada por

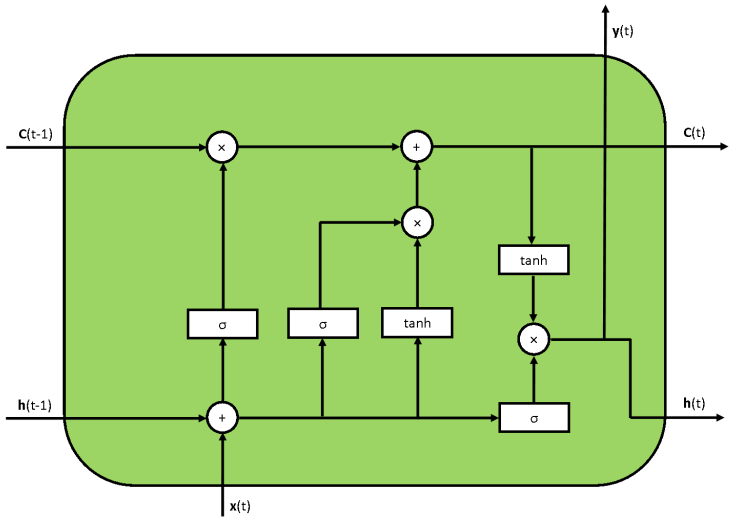


Figura 2.3: Esquema de una célula LSTM. Las entradas de la LSTM son $\mathbf{x}(t)$ y los estados internos $\mathbf{h}(t-1)$ y $\mathbf{C}(t-1)$, que se procesan y vuelven a ser considerados en el siguiente estado. El flujo de las operaciones está representado.

$$\mathbf{f}(t) = \sigma(\mathbf{W}_f(\mathbf{x}(t) + \mathbf{h}(t-1))) = \sigma(\mathbf{W}_{fx}\mathbf{x}(t) + \mathbf{W}_{fh}\mathbf{h}(t-1)). \tag{2.62}$$

La segunda igualdad es una forma alternativa de entender el flujo que aparece en [251]. Otras compuertas. activadas por la función sigmoide también tienen el mismo mecanismo. Así las compuertas $\mathbf{i}(t)$ y $\mathbf{o}(t)$ de entrada y salida tienen una estructura similar

$$\mathbf{i}(t) = \sigma(\mathbf{W}_i(\mathbf{x}(t) + \mathbf{h}(t-1))), \tag{2.63}$$

$$\mathbf{o}(t) = \sigma(\mathbf{W}_o(\mathbf{x}(t) + \mathbf{h}(t-1))). \tag{2.64}$$

Además de los pesos, las puertas se diferencian en la forma en la que se desarrollan con la salida $\mathbf{h}(t)$ y con el estado celular (*cell state*) $\mathbf{C}(t)$, el cual es un estado interno auxiliar que conserva información previa. Este canal contiene información a largo plazo que influye en la salida principal. Tal estado celular está dado por la siguiente expresión, donde $\odot$ representa el producto de vectores entrada por entrada:

$$\mathbf{C}(t) = \mathbf{f}(t) \odot \mathbf{C}(t-1) + \mathbf{i}(t) \odot \tanh(\mathbf{W}_C(\mathbf{x}(t) + \mathbf{h}(t-1))). \tag{2.65}$$

La salida de la función de olvido multiplica primero a la entrada $C(t-1)$ determinando cuánta información previa pasar o ser "olvidadas". Posteriormente aplicamos una activación tangente hiperbólica a la suma ponderada de $\mathbf{h}(t-1)$ y $\mathbf{x}(t)$. La tangente hiperbólica tiene $(-1, 1)$ como imagen, así que admite la posibilidad de restas. Con la compuerta de entrada, regulamos esta respuesta que va a ser añadida a $\mathbf{C}(t)$. Finalmente la última salida está dada por el producto de la compuerta de salida por el estado interno, pero ajustándolo en el intervalo $(-1, 1)$ de forma que

$$\mathbf{h}(t) = \mathbf{o}(t) \odot \tanh(\mathbf{C}(t)). \tag{2.66}$$

Entre las aplicaciones de las LSTMs destaca la generación de textos. Por ejemplo, [232] utiliza una LSTM bidireccional y el modelo de atención de Bahdanau [12] para tal fin. Eso manifiesta que las LSTMs pueden abstraer estructuras sintácticas. Esta idea fue utilizada para la generación de descriptores de imágenes (*Image Captioning*) junto con las redes convolucionales, tal como se expondrá posteriormente.

Capítulo 3

Redes Neuronales Convolucionales y Visión

> *The study of vision must [...] include not only the study of study of how to extract from images the various aspects of the world that are useful to us, but also an inquiry into the nature of the internal representations by which we capture this information*
>
> David Marr [172]

En el capítulo anterior se ha abordado la importancia de las redes neuronales FNN completamente conectadas y cómo, teóricamente, pueden emular cualquier función continua. En la práctica, no siempre es posible ajustar redes neuronales FNN a los datos no vistos y un sobreentrenamiento puede derivar a un sobreajuste. Un particular tipo de FNN no completamente conectado consiste en las Redes Neuronales Convolucionales o *Convolutional Neural Networks* (CNNs), las cuales presentan propiedades deseadas que las han hecho particularmente exitosas.

El planteamiento de las CNNs está altamente motivado en el *problema de la visión*, que consiste en tratar de responderse a la pregunta *¿cómo vemos?* . ¿Qué mecanismos internos nos posibilitan entender la escena visual y reconocer objetos en ella? Se trata de un problema complejo, ya que que podemos reconocer una amplia cantidad de objetos independientemente de su posición, tamaño, orientación y luminosidad. También podemos

reconocer objetos de una imagen incompleta.

Esta versatilidad para reconocer objetos llevó a los investigadores del siglo XX a tratar de encontrar respuestas a través de múltiples disciplinas incluyendo psicología cognitiva, neurobiología, neuropsicología e incluso desde la computación [159], donde algunos de los primeros trabajos se le deben a David Marr [172], quien tuvo una influencia importante en el desarrollo de la *Visión Computacional*.

Desde Marr (en 1982), la teoría computacional había reconocido la importancia de la detección de *primitivas*, es decir, elementos de las imágenes que en conjunto la determinan, como esquinas, discontinuidades, manchas pequeñas (*blobs*), cruces con cero, entre otros. El concepto de primitivas (*primitives*) ha sido sustituido por el de características de la imagen (*image features*), que son aspectos de la imagen como puntos, esquinas, color, entre otros que nos auxilian para su identificación. La *Visión computacional* se ha caracterizado en residir en esta extracción de características, generalmente definidas de forma manual [222]. Como veremos, el reconocimiento de objetos sigue cuatro fases diferenciadas: un preprocesamiento de las imágenes para eliminar objetos indeseables como ruido, una segmentación de las regiones de interés, la extracción de características y una fase final de clasificación. Las redes neuronales multicapa han sido originalmente propuestas para tareas como la resolución de problemas de clasificación. Sin embargo, también pueden ser aplicadas directamente como clasificadores omitiendo las fases previas, introduciendo como entrada a los valores de intensidad de los pixeles de la imagen.

Desde la Introducción, hemos indicado que la orientación de la presente tesis apunta hacia los modelos con cierto fundamento biológico. Como se verá para casos como el aprendizaje hebbiano, añadir modelos con mayor inspiración biológica, no necesariamente resulta en una mejora de los algoritmos previamente planteados. Esto puede deberse a que los modelos se encuentren desconsiderando aspectos que resulten necesarios en el problema en cuestión, o simplemente porque el modelo no es adecuado para su implementación computacional.

Las redes neuronales convolucionales constituyen un modelo biológicamente inspirado que ha tenido un éxito observable en los contextos en los que se ha aplicado, incluyendo en los problemas fuertes de reconocimiento de objetos. Desde la introducción, se ha

mencionado que a pesar de lo anterior, las CNNs no son un modelo exacto de la corteza visual, pero sus resultados dan cierta validez a los modelos de los cuales se inspira. Esta discusión sobre hasta qué punto las redes convolucionales son biológicamente plausibles será abordada con mayor detalle en este capítulo. Lo que es importante recalcar es que el estudio de cómo las neuronas pueden estar organizadas en la corteza visual motivó al desarrollo de redes neuronales más efectivas en el reconocimiento de objetos.

Las redes neuronales convolucionales se han aplicado no sólo para las tareas iniciales de clasificación de imágenes, sino que sus aplicaciones se han orientado a varios aspectos, teniendo relación con las imágenes o no. Se han encontrado aplicaciones de las redes convolucionales en contextos como reconocimiento de voz (*Speech recognition*) [1], análisis de sentimientos [261, 48] (la primera utilizando LSTMs). Sin embargo, destacan posiblemente más las aplicaciones que se han encontrado a contextos de imágenes, como la detección de rostros [37, 152, 268], segmentación semántica (usualmente utilizando la capas deconvolucionales) [161, 188, 253] e *Image captioning*, que se abordará posteriormente. Siendo un modelo que combina tanto cierta inspiración biológica como resultados adecuados, las redes neuronales convolucionales serán punto de partida para el desarrollo de los temas contenidos en la presente tesis.

3.1. Visión Computacional Clásica

El enfoque clásico de Visión Computacional puede enlazarse con la teoría hasta ahora descrita de Redes Neuronales. Las Redes Neuronales Multicapa FNN pueden recibir un vector de características como entrada y con una salida de las clases a clasificar. Un ejemplo de tales características son los *momentos geómetricos* de la imagen, en especial los *Momentos de Hu*, propuestos originalmente en [104]. Los momentos de orden p, q una imagen $I(x, y)$[1] de tamaño $M \times N$ están dados por

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^p y^q I(x, y). \tag{3.1}$$

[1]Una imagen digital puede considerarse como una función $I(x, y)$ que asigna a la posición (x, y) un valor o vector de intensidad.

Los momentos centrales están dados por

$$\mu_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x,y)(x-\overline{x})^p (y-\overline{y})^q, \tag{3.2}$$

donde $\overline{x} = \frac{m_{10}}{m_{00}}$ y $\overline{y} = \frac{m_{01}}{m_{00}}$. Los momentos centrales normalizados son

$$\eta_{pq} = \frac{\mu_{pq}}{m_{00}^{\frac{p+q}{2}+1}}. \tag{3.3}$$

Los siete momentos de Hu están dados por

$$\begin{aligned}
h_1 &= \eta_{20} + \eta_{02}, \\
h_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2, \\
h_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
h_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2, \\
h_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})\big((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})\big), \\
&\quad + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\big(3(\eta_{03} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\big), \\
h_6 &= (\eta_{20} - \eta_{02})\big((\eta_{30} + \eta_{12})^2 (\eta_{21} + \eta_{03})^2\big), \\
&\quad + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}), \\
h_7 &= (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\big(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\big), \\
&\quad - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})\big(2(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\big).
\end{aligned}$$

Podemos construir una red neuronal que tenga siete neuronas de entrada y cuya salida sean las capas de reconocimiento. Una propuesta es presentada por [32], donde organiza una red neuronal con las siete neuronas de entrada previamente descritas, una capa intermedia con seis neuronas y una capa de salida con una neurona que detecta la presencia o ausencia de rostros. Entre los trabajos enfocados al reconocimiento de objetos utilizando tanto redes neuronales como Momentos de Hu figura [226]. Otra aplicación es utilizar momentos de Zernike y Legendre para efectuar clasificación de rostros utilizando una red neuronal RBFNN [78].

De manera general, los métodos tradicionales disponen de cuatro fases diferenciadas, las cuales son [65]:

- *Preprocesamiento:* Operaciones sobre la imagen que sirven para mejorar su calidad eliminando ruido o artefacto, o convirtiendo la imagen a escala de grises, binario, o en alguna representación conveniente.
- *Segmentación:* Recorte de la imagen en regiones de interés.
- *Extracción de características:* Consiste en calcular descriptores de la imagen que funcionan como *items* o pistas para efectuar su reconocimiento.
- *Clasificación:* Consiste en el algoritmo destinado a clasificar la imagen partiendo del vector de características. Algunos clasificadores comunes son los k-Vecinos más Cercanos (kNN), distancia de Mahalanobis y Máquinas de Soporte Vectorial (*Support Vector Machines* o SVM).

3.2. Operaciones de las Redes Neuronales Convolucionales

En las páginas anteriores hemos visto que los problemas de reconocimiento de objetos en los métodos de Visión Computacional suelen abordarse mediante una detección de características a la cual sigue una clasificación, que puede llevarse a cabo mediante el empleo de una red FNN. Las CNNs, en lugar de emplear métodos de preprocesamiento y de extracción de características proponen crear capas especializadas para efectuar dichas tareas. De esta forma, se sugiere utilizar neuronas especializadas para efectuar dichas tareas, en lugar de utilizar herramientas "artificiales".

Introduciremos las operaciones efectuadas por las redes neuronales convolucionales más comunes y que aparecen compartidas por los modelos biológicamente más plausibles (como HMAX,*v. infra*) y que tienen cierto sustento biológico. Tales operaciones importantes de las redes neuronales son la **convolución** y **pooling**. Salvo cuando se indique, la descripción realizada se basará en [4].

3.2.1. Capas convolucionales

Sean f, g dos funciones definidas en $\mathbb{Z}$. La *convolución discreta* $f * g$ está dada por

$$(f * g)(x) = \sum_{n \in \mathbb{Z}} f(n)g(x - n). \tag{3.4}$$

En términos generales, si $f, g : \mathbb{Z}^m \to \mathbb{R}$ entonces la convolución está dada por [175]

$$(f * g)(x_1, \ldots, x_m) = \sum_{p_1 \in \mathbb{Z}} \cdots \sum_{p_m \in \mathbb{Z}} f(p_1, \ldots, p_m) \cdot g(x_1 - p_1, \ldots, x_m - p_m). \tag{3.5}$$

Para el caso de una imagen $I(x, y) \in \mathbb{R}^{A \times B}$, podemos considerar un filtro (también llamado *kernel*) $W \in \mathbb{R}^{P \times Q}$ (la cual puede ser entendida como la función g con ceros fuera del filtro). Entonces la convolución de la imagen con el filtro está dada por

$$(I * W)^l(x, y) = f\Big(\sum_{i=0}^{P-1} \sum_{j=0}^{Q-1} w_{i,j} I(x + i, y + j)\Big). \tag{3.6}$$

Observemos que si convertimos tanto la imagen como el filtro en vectores de forma conveniente, obtenemos un producto punto. Por lo tanto, una convolución puede ser calculada por una neurona con pesos dados por W sobre una subárea de la imagen (*campo receptivo*). La salida de la convolución es una matriz filtrada de dimensiones $A - M + 1 \times B - N + 1$, a la cual se aplica una función de activación. En una capa convolucional podemos aplicar un número arbitrario de filtros que comparten los mismos pesos ajustables de la red.

Asimismo, podemos definir un paso o *stride* s que es el número de pixeles a omitir para la aplicación de la convolución, de forma que el resultado de convolucionar sea $\forall m \in \{1, \ldots, A - 1\} \cap \{m | m = 0 \mod s\}$, $\forall n \in \{1, \ldots, B - 1\} \cap \{n | n = 0 \mod s\}$

$$(I * W)(x, y) = f\Big(\sum_{i=0}^{P-1} \sum_{j=0}^{Q-1} w_{ij} I(x + i, y + j)\Big). \tag{3.7}$$

El resultado de tal convolución es una matriz de dimensiones $\frac{A-P}{s} + 1 \times \frac{B-Q}{s} + 1$.

3.2.2. Capas de Pooling

Otra operación usada en las CNNs es el submuestreo o *pooling*, que básicamente reduce la dimensión de la capa anterior. Fue originalmente propuesto en la LeNet-1 [145] con el nombre de *submuestreo*. Consiste en dividir la imagen en regiones $R_j \in \mathbb{R}^{d\times d}$ y calcular una salida específica, mediante alguna activación, esto es

$$y = a(\{x_{ij}|i, j \in \{1, \ldots, d\}\}). \tag{3.8}$$

Un tipo común es el *max pooling* (véase la figura 3.1) que consiste en seleccionar el máximo de los valores de entrada (a = máx en la expresión anterior). Otra opción es el *average pooling*, que consiste en promediar a todas las entradas. De manera empírica, [221] ha verificado que el max pooling arroja mejores resultados que el average pooling.

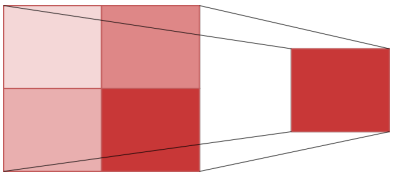


Figura 3.1: Ejemplo gráfico de la operación MaxPooling representando a los números como escalas de rojo

Otra propuesta es presentada por [266], conocida como *stochastic pooling* la cual consiste en seleccionar la salida escogiendo aleatoriamente con un probabilidad p_i dada por

$$p_i = \frac{a_i}{\sum_{k\in R_j} a_k}. \tag{3.9}$$

Otra opción, propuesta por los mismos autores, es obtener la salida mediante la suma ponderada de las probabilidades, de la forma siguiente

$$y_j = \sum_{i\in R_j} p_i a_i. \tag{3.10}$$

3.3. Benchmarks notables

Una forma empírica de comparar la eficacia (o eficiencia) de algoritmos distintos que resuelven un mismo problema es utilizar un *benchmark*, el cual consiste en simplemente medir sus resultados por medio de una métrica conveniente [216]. Para el caso de redes neuronales, dos bases de datos han tenido relevancia para contrastar qué tan efectivas son las soluciones que se han presentado. Tal es el caso de las bases MNIST e Imagenet, que por el número de imágenes que disponen así como su difusión, han contribuido a ofrecer un marco de referencia para la soluciones propuestas al problema de clasificar imágenes.

3.3.1. MNIST

La base de datos Modified National Institute of Standards and Technology (MNIST) [147] es un conjunto de 60000 imágenes de dígitos manuscritos para entrenamiento y 10000 para prueba de 28×28. El problema de clasificación consiste en etiquetar adecuadamente las imágenes de prueba de los dígitos del 0 al 9 y ha sido abordado mediante clasificadores lineales, kNN, clasificadores no lineales, SVMs y redes neuronales [52].

Las comparaciones realizadas de las CNNs con los métodos tradicionales de Visión Computacional muestran una tendencia claramente favorable hacia las mismas. Una importante comparación de algoritmos fue elaborada por el mismo LeCun [146], donde muestra el rendimiento de diferentes algoritmos. Los métodos tradicionales de Visión Computacional en general arrojaron resultados inferiores al desempeño logrado por las redes presentadas por LeCun y sus colaboradores. La primera red convolucional (LeNet-1) tuvo una tasa de error de 1.7 %, lo cual fue superior a la mayoría de los métodos tradicionales salvo los basados en SVM. La LeNet-5, aplicando un incremento de datos (*data augmentation*) logró resultados similares a los mejores métodos basados en SVM (0.8 %) y la Boosted LeNet-4 fue capaz de superarlos (0.7 %). Redes Neuronales Convolucionales más profundas lograron finalmente alcanzar una tasa de error de 0.35 %, utilizando tarjetas gráficas para su procesamiento [42].

Nuevas comparaciones confirman la eficacia de las redes convolucionales. Así, por ejemplo, [65] aplicó una serie de métodos de Visión computacional con los datasets MNIST1000

(MNIST pero con 10000 imágenes de entrenamiento) y MNIST5400 (MNIST con 54000 imágenes) obteniendo los resultados presentados en la tabla 3.1[2]. El mejor resultado se obtuvo utilizando *zoning* y k-NN con .92 con MNIST1000 y .95 con MNIST5400. Ninguno de estos métodos arrojó mejores resultados que la LeNet5 modificada propuesta (véase tabla 3.2).

Cuadro 3.1: Comparaciones realizadas por [65] con respecto a la exactitud sobre el conjunto de prueba.

Clasificador	Extractor de características	MNIST1000	MNIST5400
k-NN	Momentos de Hu	.38	.4
	Descriptores de Fourier	.53	.56
	Proyecciones de Histograma	.63	.66
	Proyecciones celulares	.91	.94
	Local Line Fitting	.88	.92
	Zoning	.92	.95
SVM	Momentos de Hu	.39	.4
	Descriptores de Fourier	.47	.5
	Proyecciones de Histograma	.61	.63
	Proyecciones celulares	.86	.89
	Local Line Fitting	.84	.87
	Zoning	.85	.87
Mahalanobis	Proyecciones celulares	.55	.6
	Zoning	.89	.93

Cuadro 3.2: CNN propuesta por [65]

Épocas	MNIST1000	MNIST5400
15	.93	.98
25	.94	.98

[2]Excluimos algunos resultados no representativos

3.3.2. Imagenet

El reto de *Imagenet* (*ImageNet Large Scale Visual Recognition Challenge* o ILSVRC) es un *benchmark* considerado estándar para el reconocimiento de objetos a larga escala [215]. Se llevó a cabo desde el 2010 y consta de un dataset de imágenes y una competencia anual, dividida en tres tareas: clasificación de imágenes (1000 categorías con más de un millón de imágenes), localización de un objeto único y detección de objetos. La primera tarea se evalúa con métricas como Exactitud top-5 (*top-5 Accuracy*), la cual mide si la clase pertenece al conjunto de las cinco clases más altas. La resolución efectiva de este problema de clasificación, intrínsecamente difícil, permitió a las redes convolucionales posicionarse como modelos válidos para los problemas de imágenes.

3.4. Algunas arquitecturas históricas

El diseño de las redes neuronales o *arquitectura* depende del tipo de problema a resolver, ya que éste determina el número de entradas y salidas de la red. Algunas de las mencionadas arquitecturas merecen una mención especial, ya que solucionan un problema de manera efectiva y sugieren nuevas perspectivas para el diseño de las redes. Dos arquitecturas particulares abrieron camino en la resolución de los problemas de clasificación de caracteres y de numerosas imágenes naturales, las cuales son la LeNet-5 [146] y la AlexNet [133]. Una vez resuelto en gran medida el problema de clasificación de MNIST por medio de las arquitecturas LeNet, el nuevo reto consistió en poder clasificar efectivamente a una gran cantidad de clases de imágenes naturales, que se materializó el problema de Imagenet, al principio abordado con métodos tradicionales hasta la irrupción de la AlexNet, que marcó un punto de inflexión para el desarrollo del *Deep Learning* y de la Inteligencia Artificial en general. A partir de la AlexNet se propusieron nuevas arquitecturas convolucionales que mejoraron los resultados hasta el momento logrados por dicha red.

Como se observa en el desarrollo histórico de las CNNs, las primeras arquitecturas se fueron moldeando hasta tener un formato típico, utilizando capas convolucionales y de submuestreo intercaladas (Convolucional-*Pooling*-Convolucional-*Pooling*), las cuales realizan el proceso de extracción de características, para recibir un clasificador consistente de una

red neuronal FNN. Este formato aparece en redes como la LeNet-5, AlexNet y las VGGs. Con el tiempo, este esquema fue siendo modificado con la aparición de nuevas propuestas como las arquitecturas Inception y ResNet. Presentaremos una revisión de algunas de las arquitecturas más relevantes en la resolución de los problemas de clasificación de MNIST y Imagenet.

3.4.1. Origen de las Redes Convolucionales y Arquitecturas LeNet

El artículo de Yann LeCun, Léon Bottou, Yoshua Bengio y Patrick Haffner [146] probó la eficacia de las redes neuronales convolucionales, utilizando tanto las capas convolucionales como las de *submuestreo* (*pooling*). Sin embargo, el origen de las redes neuronales convolucionales se sitúa a finales de la década de los ochenta, relacionándose con la figura de Yann LeCun.

En 1989, LeCun publicó algunos de los primeros bocetos de redes neuronales convoluciones, para tratar de clasificar dígitos de 16×16. Uno de los más antiguos es la *Net-3* propuesta en [148], que es una red *localmente conectada* (véase figura 3.2), la cual cuenta con dos capas locales y una capa completamente conectada con diez neuronas de salida. La primera capa localmente conectada tiene 8×8 neuronas con filtros de 3×3. La segunda capa tiene 4×4 neuronas con kernel de 5×5. Esta red presentó un desempeño ligeramente mejor que una FNN de dimensiones similares (88.5 % contra 87 %). Otras dos arquitecturas lograron resultados notablemente mejores y la Net-5 alcanzó 98.4 % de exactitud.

Figura 3.2: Arquitectura de la Net-3

Un nuevo modelo de red neuronal para clasificación de dígitos fue presentado en [144]. Esta red (figura 3.3) que cuenta con dos capas convolucionales y una completamente conectada. La arquitectura LeNet-1 aparece en [145] constando de cinco capas (3.4): H1, capa convolucional de 4 filtros de 5×5, obteniendo una salida de 24×24; la capa H2 es de submuestreo de promedio (*Average Pooling*) de 2×2; la capa H3 es convolucional con

12 filtros de 5×5; H4 es otro *average pooling* de 2×2; H5 es la capa de clasificación de 10 neuronas. El dataset de caracteres constaba de 7291 dígitos manuscritos y 2549 dígitos impresos para el entrenamiento, así como 2007 dígitos manuscritos y 700 impresos, para el conjunto de prueba. El error de prueba obtenido fue de 3,4 %. Esta arquitectura tenía una entrada de 28×28, en lugar de 16×16. Al aplicarse en la base de datos MNIST, esta red alcanzó un desempeño de 1,7 %.

Figura 3.3: Arquitectura convolucional previa a la LeNet-1

Finalmente, la arquitectura LeNet-5 [146], posiblemente la de mayor relevancia, fue aplicada a la base de datos MNIST. Esta red consta con una configuración de una capa convolucional de 6 filtros de 5×5, *average pooling* de 2×2, 16 filtros de 5×5, otro *average pooling* de 2×2, una capa convolucional con 120 filtros de 5×5 y finalmente una capa densa con 84 unidades (neuronas) y la capa de clasificación con 10 unidades. La figura 3.5 es una representación de la misma. Esta arquitectura logró una exactitud de 0,95 % sobre el conjunto de prueba, superando al mayor logro alcanzado con redes FNN (2,45 %). El uso de aumentación de datos (por medio de distorsiones), permitió a la LeNet-5 alcanzar 0,8 % de error. La arquitectura LeNet-4, utilizando la técnica de *boosting* logró 0,7 % de error, que supera a los métodos conocidos de Visión Computacional. El uso de arquitecturas más profundas como en [42], junto con los comités de redes neuronales, permitió alcanzar un error de $0{,}27\,\% \pm 0{,}02$.

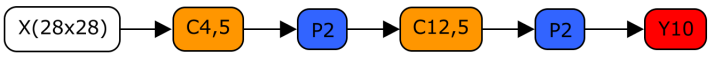


Figura 3.4: Arquitectura LeNet-1

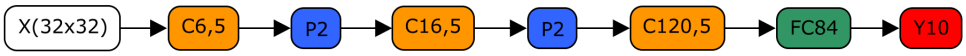


Figura 3.5: Arquitectura LeNet-5

3.4.2. AlexNet

La arquitectura AlexNet [133] es célebre por lograr un amplio margen en el concurso de Imagenet obteniendo 17 % de error top-5. A partir de este momento, las redes neuronales dominarán el ámbito de reconocimiento de imágenes, palpable en la ausencia de métodos tradicionales que habían dominado la escena hasta antes de la AlexNet [215].

AlexNet es una arquitectura de cinco capas convolucionales con tres operaciones de *MaxPooling* intercaladas, así como dos capas densas de 2048 neuronas, seguidas de la capa de clasificación de 1000 neuronas correspondientes a las 1000 clases. Es notable también el empleo de la activación ReLU, la cual es requerida dada la profundidad de la red.

3.4.3. Arquitecturas VGG

Las Arquitecturas VGG fueron propuestas en [228] por Karen Simonyan y Andrew Zisserman en el 2014, obteniendo el segundo lugar del ImageNet Large-Scale Visual Recognition Challenge (ILSVRC), detrás de GoogLeNet, con un top-5 error de 7.3 %, lo cual representa una mejora significativa sobre la ZFNet previa (11.1 %) [3].

Las arquitecturas VGG reciben una imagen con tamaño fijo de 224×224 en escala RGB (propios de ImageNet). Una innovación notable de estas arquitecturas es el uso de filtros pequeños de tamaño 3×3 para convoluciones con stride de 1 y 2×2 para Max Pooling con stride de 2, que contrasta con la arquictectura clásica de AlexNet. Finalmente tiene dos capas completamente conectadas de 4096 neuronas y de salida de 1000 (correspondientes a las 1000 clases) con activación softmax, mientras que las capas ocultas emplean activación ReLU. En total se proponen 5 arquitecturas, las cuales las D y E se han denominado VGG-16 y VGG-19 por su número de capas. La VGG-16 está configurada con un grupo de capas convolucionales de 64 filtros, grupo de dos capas con 128 filtros, grupo de tres con 256, grupo de tres con 512 y finaliza con 3 con otras 512 para terminar con las capas completamente conectadas. Entre los grupos de capas se encuentran intercaladas las operaciones de MaxPooling y en total tiene 138 millones de parámetros. La arquitectura VGG-19 (que presentó el menor error de clasificación) cuenta con una capa más en los últimos tres grupos de capas convolucionales [228] y está presentada en la imagen 3.6.

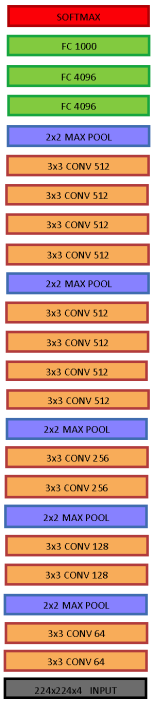


Figura 3.6: Arquitectura VGG19

3.4.4. Arquitecturas Inception de Google

Denominaremos *arquitecturas Inception* a una serie de redes profundas que siguieron a la GoogLeNet, las cuales comparten numerosos rasgos comunes, como el empleo de módulos *Inception*[3] y sus sucesores, así como la peculiaridad de haber sido desarrolladas por investigadores de la empresa Google, destacando autores como Szegedy y Chollet. La arquitectura GoogLeNet o InceptionV1 [239] busca incrementar la profundidad de la red, sin que ello afecte en demasía los recursos computacionales. Obtuvo un error top-5 de 6,67 % en el concurso de Imagenet 2014 (ILSVRC 2014), para entonces lo menor registrado.

Las arquitecturas Inception cuentan con *módulos Inception* (Figura 3.7), los cuales aplican separadamente convoluciones de 1×1, 3×3 y 5×5 de manera separada, así como un *Max Pooling*. El uso de *kernels* de tamaño 1 $\times 1$, el uso del Submuestro de

[3]El nombre *Inception* proviene del nombre *Network in Network* (el cual aparece por primera vez en [154]) así como del meme "We need to go deeper", extraído de un fotograma de la película Inception (revísese el artículo original *Going Deeper with Convolutions* [239] si la referencia parece sorprendente)

Promedio Global (*Global Average Pooling*), así como la disposición de submódulos de redes convolucionales aparece en [154] bajo el nombre de *Network In Network* (NIN). Estas redes cuenta con módulos de capas convolucionales seguidas de capas densas. Las convoluciones de 1×1 ayudan a reducir las dimensiones de la red. Al final es aplicada una operación de concatenación, consistente en adjuntar los valores de dos o más vectores en uno mayor: si $\mathbf{v} = (v_1, \ldots, v_a)$ y $\mathbf{u} = (u_1, \ldots, u_b)$ son vectores, entonces la concatencación se define como $[\mathbf{v}, \mathbf{u}] = (v_1, \ldots, v_a, u_1, \ldots, u_b)$. Estos módulos son incorporados uno a uno para formar la GoogLeNet, la cual cuenta con un total de 27 capas con 9 módulos Inception.

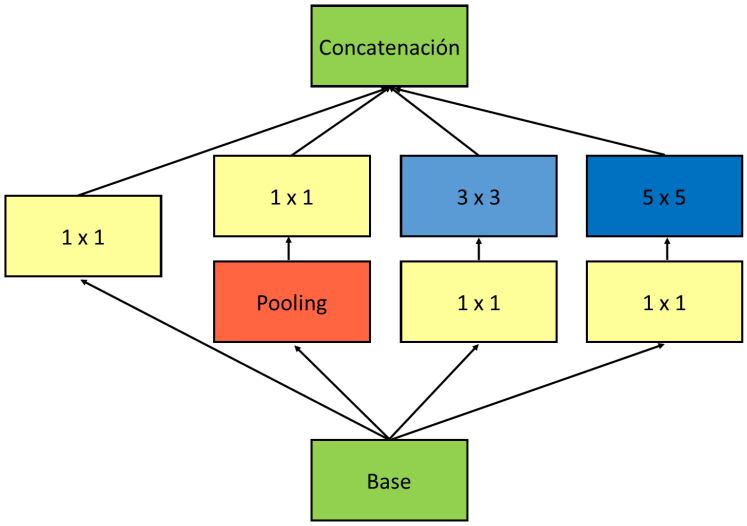


Figura 3.7: Módulo de *Inception*

La GoogLeNet (InceptionV1) es más eficiente que la rival VGG19 en número de parámetros requeridos. Sin embargo, la VGG19 utiliza *kernels* de 3×3 en lugar de 5×5 el cual puede representarse por medio de dos capas de 3×3. Asimismo se pueden reemplazar filtros de $n \times n$ por filtros de $n \times 1$, seguido de $1 \times n$, reduciendo el número de parámetros (factorización asimétrica), aunque en la práctica esta técnica no siempre da resultados adecuados. Estas medidas, junto con otras como la reducción eficiente del tamaño de rejilla, llevaron al desarrollo de las arquitecturas Inception-V2 e Inception-V3. La última mencionada, alcanzó un error top-5 de 3,58 %.

Entre las posteriores arquitecturas desarrolladas por Google destacan la Inception-V4 (4.6 % de error top-5), Inception-ResNet [238], la Xception [40](5.5 % de error top-5) y las MobileNets [103]. Las redes Inception-ResNet y Xception presentan tanto rasgos característicos de las Inception originales, así como ideas tomadas de las redes residuales, que se abordarán a continuación.

3.4.5. Arquitecturas residuales y densas (ResNet-DenseNet)

Una segunda forma de mejorar el desempeño de las redes neuronales convolucionales se alcanzó en el 2015 con el trabajo de Kaiming He y sus colaboradores de Microsoft [87] con el desarrollo de las redes residuales profundas o ResNets. Los desarrollos preliminares de las VGGs y las Inceptions, así como otras redes, hacían hincapié en la necesidad de incrementar la profundidad de las redes neuronales. Sin embargo, una observación que se había logrado radica en que un incremento ingenio de la arquitectura resultaba en problemas de desvanecimiento o explosión de los gradientes.

La solución presentada a los problemas enfrentados consistió en la conexión de capas mediante un mapeo identidad (Figura 3.8). Las operaciones son las siguientes: sea x la entrada al módulo y $CNN_{a,3}$ la aplicación de una convolución de 3×3 con a filtros. Entonces se obtiene una salida

$$\mathcal{F}(x) = \text{ReLU} \circ CNN_{b,3}(\text{ReLU} \circ CNN_{a,3}(x)), \tag{3.11}$$

donde $\circ$ es la composición de funciones. La salida del módulo estará dada por $\mathcal{F}(x)+x$, en lugar de x como se aplicaría en una arquitectura VGG. Las arquitecturas finales ResNet consisten de la adición de numerosos módulos ResNet, finalizados por un *Average Pooling*. En el concurso ILSVRC 2015, el error alcanzado con esta red fue de 3,57 %, siendo el algoritmo ganador. Las mejores arquitecturas ResNets sobre el conjunto de validación fueron la ResNet50, ResNet101 y ResNet152, siendo las primeras redes con más de 100 capas. Mejoras al planteamiento de estas redes residuales aparecen en forma de las arquitecturas ResNet50V2, ResNet101V2 y ResNet152V2 [88].

Es posible considerar a las Redes Densas (DenseNets) [105] como sucesoras de las Redes

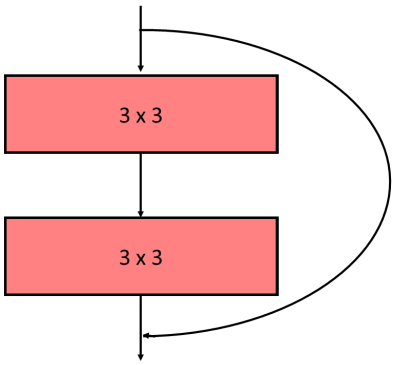


Figura 3.8: Módulo de ResNet (basado en [87]). El mapeo identidad se aplica en el arco mostrado. En la intersección de la salida de la identidad y las operaciones de convolución posteriores se realiza la concatenación.

Residuales, las cuales conectan varias capas previas a los módulos de capas convolucionales, en lugar de establecer una única conexión. Sea ℓ el número de la capa en un bloque denso y $\mathcal{F}_\ell = CNN_3 \circ \text{ReLU} \circ \text{BN}$, donde CNN_3 es una operación de convolución de 3×3 y BN es la normalización del batch (*Batch Normalization*, véase [114]). La función $\mathcal{F}_\ell$ recibe como entrada una concatenación de los mapas de características de las capas previas $x_0, \ldots, x_{\ell-1}$. Entonces se define la siguiente regla recursiva

$$x_\ell = \mathcal{F}_\ell([x_1, \ldots, x_{\ell-1}]), \tag{3.12}$$

donde $[\cdot]$ es la operación concatenación. Una red densa se forma agregando varios bloques densos y operaciones convolución y pooling intercaladas (véase figura 3.9).

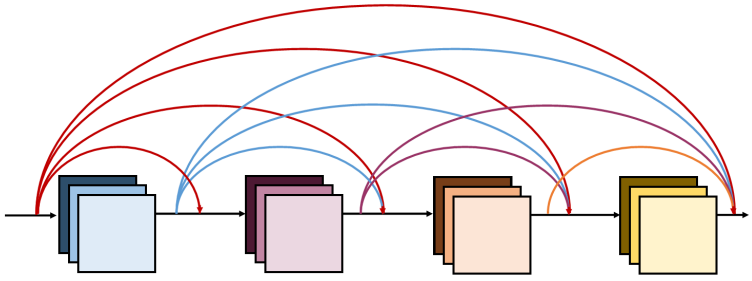


Figura 3.9: Módulo de DenseNet (basado en [105])

3.4.6. Arquitecturas híbridas

Algunas arquitecturas posteriores a la Inception-V1 y a las ResNets incorporaron algunas de las ideas abordadas en estos diseños de redes convolucionales, destacando la Inception-ResNet [238] y la Xception [40]. La última red mencionada está construida como una forma "extrema" de los módulos *Inception*, cuya representación base aparece en la figura 3.10, involucrando el uso de convoluciones separables. La arquitectura Xception cuenta con un total de 36 capas convolucionales agrupados en 14 módulos extremos con conexiones residuales, siguiendo el ejemplo de las ResNets [87].

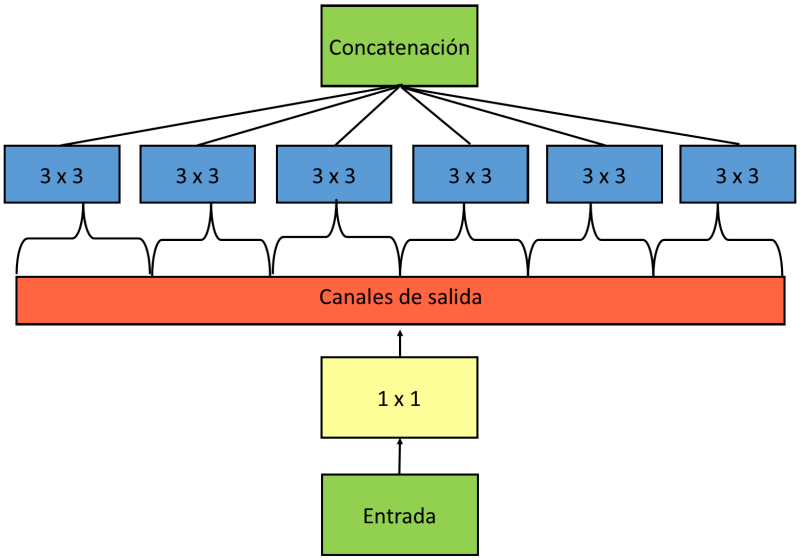


Figura 3.10: Módulo de *Xception*. La arquitectura de la red convolucional consiste en 14 módulos de este tipo.

3.5. Aplicaciones: *Image Captioning*

En esta sección abordaremos una de las principales aplicaciones de las Redes Convolucionales en conjunto con las redes LSTM. El problema de generación de textos descriptores (*captions*) de una imagen, conocido en inglés como *Image caption generation* o simplemente como *Image captioning* está fuertemente relacionado con el desarrollo conjunto tanto de

visión artificial y reconocimiento de patrones, como de la generación automática de texto y Procesamiento de Lenguaje Natural (NLP). Para que la imagen pueda ser descrita, primero es necesario que los objetos existentes puedan ser reconocidos, por lo que este problema depende ampliamente de la resolución del problema de reconocimiento de imágenes. Por lo tanto, a pesar de la dificultad del problema, soluciones bastante plausibles pudieron aparecer con el surgimiento de la AlexNet [133] en el 2012.

Tanti [241] identifica tres propuestas principales para abordar el problema de *Image Captioning*:

1. Sistemas que se basan en técnicas de Visión Artificial para la extracción de características (*features*) utilizando las técnicas de Generación de Lenguaje Natural (*Natural Language Generation*, NLG) propuestas en [211].

2. Sistemas que identifican el *caption* calculando la proximidad o relevancia de las cadenas de caracteres en los datos de entrenamiento para una imagen. Algunos de estos enfoques pueden basarse en redes neuronales [230].

3. Sistemas que utilizan una arquitectura CNN preentrenada y generan los *captions* utilizando una red recurrente, típicamente LSTM (véase imagen 3.5).

Los primeros ejemplos que evocan al problema de *Image captioning* surgieron como métodos que buscaban generar varias etiquetas acerca de una imagen. Ya desde el 2004, [195, 194] postuló el problema de generar p etiquetas a una imagen determinada, al que nombró como *auto-captioning*. El método utilizado buscaba extraer características de la imagen utilizando herramientas de Visión Artificial tras la aplicación de un algoritmo de segmentación, y posteriormente aplicaba reconocimiento a las zonas segmentadas y un etiquetado utilizando método basados en grafos. Entre los trabajos que utilizaron técnicas puramente de Visión Computacional para generar sentencias descriptoras de la imagen se incluyen [137, 180, 60]. Entre otros estudios que utilizan técnicas no neuronales pero perteneciendo a la segunda categoría figuran [74, 193].

En los artículos previamente mencionados, el problema de generación de descripciones de la imagen se muestra como un problema doble que involucra tanto la extracción de

características de la imagen como la generación de texto. Algunas arquitecturas convolucionales, como las VGGs, son particularmente útiles para la extracción de características, así como una arquitectura RNN para la generación del texto.

Uno de los primeros ejemplos de la categoría 3, surgió con la aparición de la AlexNet. Utilizando la AlexNet y una arquitectura multimodal de varias capas de RNN, [170] presentó uno de los primeros trabajos que involucraban el uso conjunto de una red CNN y otra RNN para lograr resolver el problema. El uso de redes RNN bidireccionales y multimodales fue implementado por [121] para desarrollar un programa capaz de segmentar regiones de interés en la imagen y describirla.

Algunas mejoras propuestas al modelo original fue la incorporación de arquitecturas más fuertes como la VGG19 en [126] (como en el presente trabajo) o GoogleNet en [251] pero también la introducción de unidades LSTM en ambos trabajos. Tales ideas pudieron generalizarse para su implementación en la descripción de videos propuesta por [55]. La incorporación de mecanismos de atención [263, 258] y la posibilidad de responder preguntas sobre la imagen [11] han figurado como avances recientes en el ámbito de *Image captioning*.

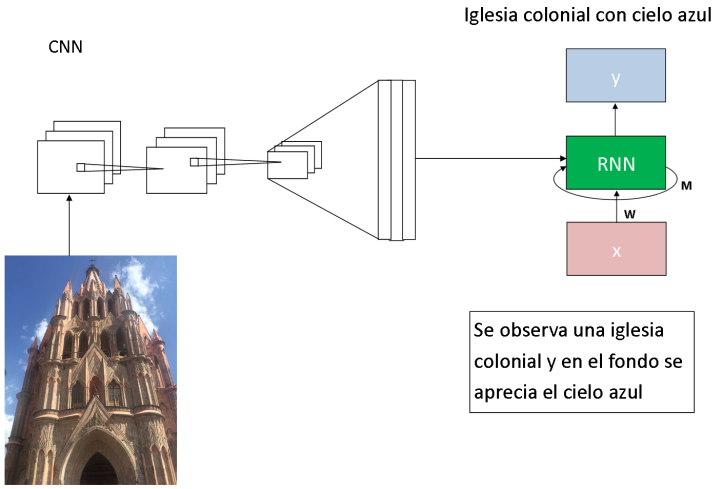


Figura 3.11: Modelo general gráfico de *Image Captioning* que representa a los sistemas con una CNN preentrenada con una red recurrente.

3.6. La Visión y las Redes Convolucionales

A lo largo de este capítulo hemos descrito el innegable éxito que han desarrollado las redes neuronales convolucionales y hemos afirmado la relación existente entre las redes convolucionales y los modelos de organización de la corteza visual, en particular sobre el modelo jerárquico de Hubel-Wiesel. En esta sección daremos argumentos por los cuales creemos que las redes neuronales convolucionales profundas, en particular la DenseNet, son los modelos más apropiados para emular la corteza visual (concretamente la Corriente Ventral), a pesar de las limitaciones que señala la literatura.

Los estudios prioneros que se realizaron a mediados del siglo XX permitieron el posterior desarrollo de las ideas que se usarán en Visión Computacional por medio de David Marr [172] y a la postre en las redes convolucionales por medio de Yann Lecun [144]. La cuestión que es relevante decir es que los teóricos computacionales ya habían advertido en la necesidad de observar qué procesos ocurren en los animales para permitir el reconocimiento de objetos y tratar de obtener un modelo computacional de la visión.

3.6.1. Estudios de Visión en Invertebrados

Un punto de partida para abordar el problema es seguir la dirección evolutiva. Nilsson [186], identifica cuatro etapas en la evolución de la visión en los animales, el cual es un rasgo distintivo de esta clase de seres vivos:

- Clase I, *Fotorrecepción no direccional*: consiste en la detección de la intensidad de luz del ambiente.
- Clase II, *Fotorrecepción direccional*: es capaz de detectar si la luz sigue algunas direcciones.
- Clase III, *Visión de baja resolución*: Se compone de cadenas de fotorreceptores que permiten la formación débil de imágenes. Es útil para controlar la velocidad y dirección del movimiento, evitando obstáculos.
- Clase IV, *Visión de alta resolución*: Visión capaz de efectuar tareas como reconocimiento de depredadores, presas y compañeros de la misma especie.

En el mismo artículo se muestra que filogenéticamente el desarrollo del ojo de alta resolución pudo tener tres orígenes independientes: en cefalópodos, artrópodos y vertebrados, no así las fotorrecepción, la cual tuvo un origen común.

Es posible utilizar fotorresistencias y cámaras de baja resolución para tratar de emular el comportamiento de animales con desarrollo visual de las clases I-III, aunque esta idea no se seguirá en esta tesis. El superfilo de los vertebrados es el Deuterostomia e incluye a los equinodermos, cordados y tunicados, los cuales cuentan con visión de baja resolución.

Un caso interesante de visión de baja resolución en Deuterostomia es el de los equinodermos, que poseen ojos en los extremos de los brazos. De acuerdo con el estudio de [71], la estrellas de mar *Linckia laevigata*, posee una visión de baja resolución que le permite reconocer únicamente objetos largos y estacionarios como los arrecifes de coral, en donde prefiere ubicarse. Los experimentos en conducta animal indican que la estrella de mar es efectivamente capaz de reconocer este tipo de objetos y avanzar hacia ellos cuando no está privado de vista. No obstante, la baja resolución de su visión le impide efectuar el reconocimiento de objetos más pequeños como depredadores potenciales.

Los artrópodos, por su lado, siguieron un proceso de evolución diferente, pero lograron desarrollar sus propios mecanismos de reconocimiento de objetos, el cual es más complejo que para el caso de los equinodermos. Algunos de los primeros estudios se efectuaron en arañas en la década de los cincuentas. Drees ([56], citado en [142]) estudió los efectos que tenían diversos estímulos visuales en la conducta de la araña *Epiblemum scenicum* encontrando que presentaba una conducta más agresiva al encontrar patrones de puntos, triángulos, círculos y cruces, pero mostraba una conducta de cortejo al encontrar patrones de puntos con líneas oblicuas (en forma de araña) (Figura 3.12).

El anterior es un remarcable ejemplo de detección de objetos realizado por un invertebrado, el cual adecuadamente puede clasificar a los objetos en tres clases: presa potencial, pareja potencial o clase nula. Esto lo logra mediante la detección de patrones específicos con los que permite efectuar dicha clasificación.

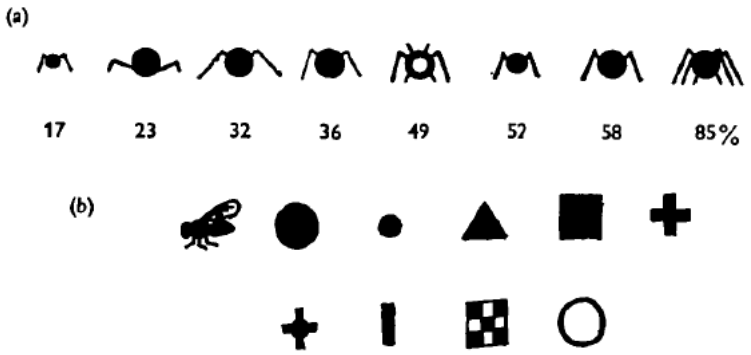


Figura 3.12: Los patrones (a) evocan una respuesta de cortejo en la araña, mientras que los patrones (b) de ataque. Tomado de [142], quien a su vez lo tomó de [56].

3.6.2. Estudios de Visión en Vertebrados

A pesar de que se puede tratar de seguir un esquema evolutivo para tratar de comprender el desarrollo cognitivo de la visión partiendo de animales más "simples", algunos de los primeros estudios sobre el tema se dieron directamente en vertebrados.

Algunos de los primeros estudios sobre la visión de vertebrados se hicieron en sapos, entre los cuales destaca el trabajo de Barlow [17] en 1952. McCullogh y Pitts, desarrolladores del primer modelo de neurona artificial, estuvieron involucrados en una investigación junto con Lettvin y Maturana [150] sobre la visión de los sapos, observando la presencia de detectores de insectos.

No obstante, algunos de los trabajos más destacados sobre la visión en vertebrados se hicieron en mamíferos como los gatos *Felis silvestris catus* y posteriormente en primates, destacando las investigaciones de Stephen Kuffler en las células ganglionares de la retina, pero sobretodo los trabajos de David Hubel y Torsten Wiesel en el Núcleo Geniculado Lateral (*Lateral Geniculate Nucleus* o LGN) y la Corteza Visual. Estos dos últimos autores fueron galardonados con el Premio Nobel de Medicina en 1981, junto con Roger Sperry [109].

Células ganglionares y LGN

El procesamiento de la información visual comieza desde la retina, por medio de células como las ganglionares. De la retina, el nervio óptico envía la información hacia el núcleo geniculado lateral (LGN), de donde parte a la Corteza Visual Primaria V1 [72]. Los trabajos de Kuffler [135, 136] permitieron descubrir cómo es la respuesta de las células ganglionares frente a patrones de luz específicos. En general, los campos receptivos de estas células son pequeños y se dividen en dos tipos:

- Células ON: Responden de forma fuerte (excitación) a los estímulos de luz en el centro y de forma débil (inhibitoria) a los estímulos de luz en los bordes.
- Células OFF: Responde de manera inversa, es decir, excitación a los estímulos de luz en la periferia del campo receptivo e inhibitoria en el centro.

Este tipo de respuesta también fue encontrada en el LGN por Hubel y Wiesel [111]. En las figuras 3.13, 3.14 y 3.15 podemos observar el tipo de respuesta que las células ON realizan ante la presencia de estímulos, teniendo una respuesta fuerte (con mayor tasa de disparo) al incidirse sobre el área del centro, pero menor al incidir en la periferia, propiciando inhibición, es decir, una disminución en la tasa de disparo. Se ha interpretado que estas operaciones realizadas permiten la detección de bordes [72], generando dos canales separados correspondientes a las células ON y OFF [178].

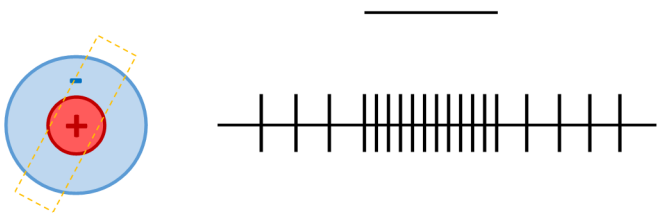


Figura 3.13: Respuesta típica de excitación de una célula del LGN o ganglionar ON al recibir un estímulo de luz que pasa por el centro. Basado en [72]. Una célula OFF produce una respuesta inversa: en este caso se produciría inhibición (menor actividad).

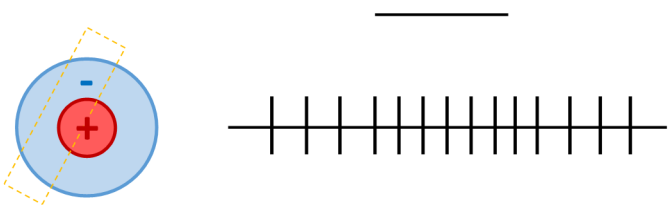


Figura 3.14: Respuesta típica de una célula del LGN o ganglionar ON al recibir un estímulo de luz que pasa por el centro y por la periferia. Basado en [72]

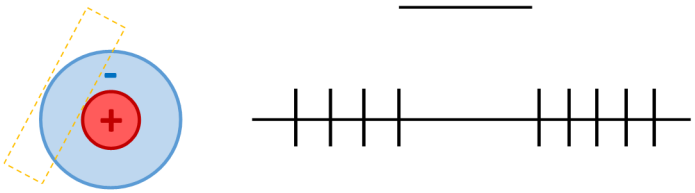


Figura 3.15: Respuesta típica de inhibición de una célula del LGN o ganglionar ON al recibir un estímulo de luz que pasa por la periferia. Basado en [72]. Una célula OFF produce una respuesta inversa: en este caso se produciría excitación (mayor actividad).

Corteza Visual

Corteza Visual Primaria (V1)

Anteriormente hemos expuesto cómo el procesamiento de las imágenes se efectúa antes de llegar al cerebro por medio de las células ganglionares de la misma retina y por las células del LGN. Esto indujo a introducir el término *campo receptivo* o *receptive field*, que es utilizado en los artículos de Hubel y Wiesel como [108]. Los campos receptivos de las células ON y OFF tiene una forma circular como se muestra en las figuras 3.16 y 3.17.

Siguiendo esta idea de tomar células individuales y observar su respuesta a determinados estímulos, Hubel y Wiesel abordaron a la corteza visual, concretamente al área conocida como V1 o Corteza Estriada (nombre con el que aparecen en los artículos originales). Al estudiar los patrones de respuesta de las neuronas corticales, observaron la presencia de células que respondían con mayor fuerza a barras con inclinación específicas, tal como se muestra en su artículo de 1959 [110]. Diez años después, Robert Wurtz com-

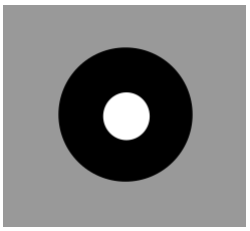


Figura 3.16: Campo receptivo de la célula ON ganglionar de la retina y del LGN. Basado en [178].

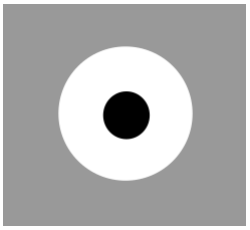


Figura 3.17: Campo receptivo de la célula OFF ganglionar de la retina y del LGN. Basado en [178].

probaría estos resultados en monos *Macaca mulatta* [256], observándose en la figura 3.18. De este modo, los campos receptivos de las células de la V1 tienen las formas mostradas en las figuras 3.19 y 3.20 [178]. Otros patrones de respuesta aparecen especificados en un artículo posterior [112].

A las neuronas anteriores se les clasificó como *células simples*, en contraposición de las *células complejas*, que desde el punto de vista de los autores, contaban con "propiedades mucho más intrincadas y elaboradas". Hubel y Wiesel distinguieron cuatro tipos de células complejas en su investigación original. Un tipo interesante es una célula compleja que responde fuertemente a barras de luz orientadas con cierto ángulo pero invariantes a posición, a diferencia de las células simples que responden a una orientación específica pero ubicadas en un lugar específico. Estos resultados fueron expuestos en el artículo [112].

Los resultados obtenidos por Hubel y Wiesel en sus estudios del LGN y la V1 tuvieron un impacto relevante en el entendimiento de la Corteza Visual, y su importancia sigue teniendo influencia en la actualidad [72]. Estudios más recientes han investigado la existencia de campos receptivos tridimensionales debido a los efectos de la visión binocular [220].

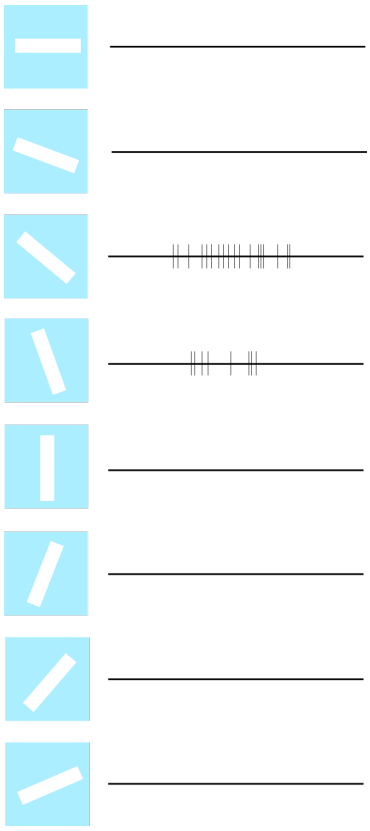


Figura 3.18: Respuesta de una célula simple de la región V1 de monos, adaptada del artículo de [256]. Observamos que solamente una inclinación muestra una respuesta suficientemente fuerte, mientras que una orientación ligeramente distinta muestra una respuesta más débil. Patrones de respuesta similares aparecen en el artículo de Hubel y Wiesel [110] aplicado en gatos.

Figura 3.19: Campo receptivo de la célula de V1. Basado en [178].

En general estas investigaciones puede ser una fuente de inspiración para los científicos computacionales, motivando al desarrollo de algoritmos con mayor sustento biológico. Por ejemplo, podría tratarse de introducirse la binocularidad y utilizar la profundidad para

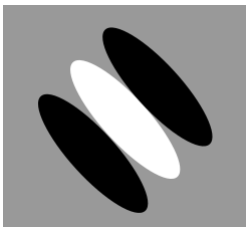


Figura 3.20: Campo receptivo de la célula de V1. Basado en [178]

favorecer la discriminación de objetos. Estas ideas no se pondrán en práctica aún, por lo que se seguirán empleando modelos de una sola cámara.

Área V2

Las siguientes áreas cerebrales relacionadas al procesamiento de imágenes, las cuales forman la Corteza Extraestriada, con mayor complejidad en las características que extraen [72]. El área siguiente a la V1 es naturalmente llamada como V2. Los estudios realizados en esta área cerebral muestran una selectividad hacia estímulos más complejos que los presentados en la V1. Ito y Komatsu en 2004 [116] encontraron que una buena parte de las neuronas de la capa superficial de la V2 de los macacos (91 de las 114 estudiadas) responden a cruces de líneas formando ángulos (figura 3.21), dando a pie a la hipótesis de que la información angular inicia en esta área. Además de ángulos, otros estudios han encontrado otros patrones procesados en la V2, incluyendo rejillas sinusoidales, arcos, semicírculos, estrellas de cinco y tres puntas, círculos, barras, espirales, rejillas hiperbólicas y concéntricas, además de los ya mencionados ángulos [90]. Estudios aún más recientes (2016-2017) subrayan el papel de la V2 para el reconocimiento de texturas [271, 190]. Estos trabajos se han realizado en especies de macacos como *Macaca mulatta* y *Macaca fuscata*.

Figura 3.21: Estímulo preferido por algunas células de la V2. Basado en [116].

Áreas V3 y V5 o Medio Temporal (MT)

Aunque el área V5 aparece posterior a la V4 en el orden numérico, no se debe entender a este esquema como lineal. En realidad, el área V4 como veremos, parece tener más rasgos en común con la V2, mientras que V3 y V5 cuentan con rasgos similares. El área conocida como V3, particularmente V3A, junto con V5, están asociadas a la percepción de movimiento de acuerdo con estudios realizados en resonancia magnética en humanos [27], lo cual parece coincidir con los estudios de lesiones realizados en macacos. Las mediciones realizadas de la tasa de disparo de las neuronas de la MT nos muestran a mayor detalle el papel que juegan en el procesamiento del movimiento. Por ejemplo, en [113] se observó la presencia de neuronas selectivas a la dirección y a la velocidad. Es interesante (aunque no tan sorpresivo, dada la realidad neuronal) notar que la clasificación de velocidad se realiza de forma discreta y no continua. Por ejemplo, una neurona revisada, tiene una preferencia hacia la velocidad angular de 52 $^\circ/s$ en sentido horario, y una respuesta cercana a 0 disparos por segundo (Hertz) para la dirección opuesta. Esto nos proporciona ideas de cómo se interpreta la velocidad en las neuronas.

V4

El área V4 introduce un nivel de complejidad aún mayor en el reconocimiento de formas. Observando las lesiones en el área V4 de los macacos, se muestra que pierden la capacidad de distinguir figuras tridimensionales en diferentes perspectivas, tal como se publicó en [176]. No obstante, este mismo artículo señala que la invarianza a posición solamente se logra en áreas más profundas, correspondientes a la Corteza Temporal Inferior. En el 2002, otra investigación [196] encontró que grupos de neuronas del área V4 pueden codificar patrones aún más complejos partiendo de sus elementos.

La relación existente entre las áreas V2 y V4 también ha sido estudiada en algunos estudios sobre macacos, mostrando evidencia de la complejidad creciente de los estímulos procesados, así como campos receptivos promedios más grandes en V4 con respecto a V2 [190], a pesar de que ambos juegan un papel importante en el reconocimiento de texturas. Sin embargo, como observa la continuación del estudio de [90], publicado como [91], las diferencias de los patrones que procesan ambas áreas no son tan radicales, sin encontrarse una fuerte distinción entre la complejidad de los estímulos preferidos de las neuronas de la V2 con respecto a la V4 (o incluso con la V1, aunque las respuestas fueron en

general más simples), ya que ambas zonas (V2 y V4) respondieron a patrones tanto simples como complejos, sin encontrarse evidencia clara de jerarquía, aunque sí de preferencia a determinados estímulos como los basados en rejillas para la V2. Por otro lado, estudios más recientes (2016 y 2020) indican que el área V4 puede tener un papel importante en la segmentación de imágenes de forma-fondo, representando una posible diferencia con respecto al área V2 [83, 260].

Corteza Temporal Inferior y Procesamiento Superior

Una revisión de los estudios realizados en la visión animal nos ha llevado a recorrer tanto un camino evolutivo pero también cronológico, empezando por los invertebrados hasta ir abordando primero estudios realizados en sapos y después gatos, para después abordar primates como los macacos. Los artículos que hemos citado tienen una larga trayectoria recorriendo la década de los cincuenta en los estudios de la retina, LGN hasta la corteza visual V1, y finalmente las áreas V2 y V4, que juegan un papel más complejo en el procesamiento de la información visual. Finalmente llegamos a las últimas fases del reconocimiento de objetos realizados por los primates, los cuales revelan el mapa completo por el cual se realiza este proceso.

Una importante salida de la V4 se dirige a la Corteza Temporal Inferior, la cual se distingue por efectuar tareas de reconocimiento de patrones más complejos, los cuales tienen una representación semántica. Esta corteza cuenta con dos áreas principales distinguidas como Área Temporal (*Temporal Area* o TE) y Área Temporo-Occipital (TEO) [202].

Desde 1984 (o incluso antes) se ha observado que la Corteza Inferior Temporal responde de manera selectiva a patrones visuales complejos como manos o rostros, tal como se muestra en el trabajo clásico de [53]. Los resultados manifiestan la existencia de una neurona con una alta tasa de disparo al presentarse una mano en diferentes orientaciones. Esta respuesta se veía considerablemente reducida al abstraer las propiedades de la mano o reducirle características. También se encontró una neurona que responde de manera fuerte a la presencia de rostros de primates, pero únicamente de forma frontal, disminuyendo la respuesta para el caso de rostros en cierto ángulo.

Por otro lado, la Corteza Temporal Anterior (*Anterior Temporal Cortex*) en humanos

también ha sido asociada al conocimiento semántico, abarcando hechos, personas, palabras e incluso hechos [25]. Utilizando resonancia magnética funcional, Peelen y Caramazza [198] encontraron que las propiedades abstractas de ciertos objetos cotidianos, como la forma y dónde son utilizados, se codifica en esta corteza cerebral.

En el 2009, Quian Quiroga y sus colaboradores [207] encontraron neuronas del Lóbulo Medial Temporal (*Medial Temporal Lobe* o MTL) que respondían fuertemente a la presencia de estímulos tanto auditivos, como visuales y escritos en humanos. Por ejemplo, una célula del hipocampo (parte del MTL) puede responder fuertemente a estímulos como imágenes de Oprah Winfrey, el texto "Oprah Winfrey" y también la voz "Oprah Winfrey". Estas neuronas resultan de particular interés para el trabajo, puesto que estas neuronas parecen codificar ideas abstractas concretas. Debido a que las primeras evidencias de la regla de Hebb (Potenciación a Largo Plazo) fueron descubiertos en esta área, volveremos a realizar una revisión más profunda sobre sus implicaciones en los capítulos posteriores.

3.6.3. Modelos de la Corteza Visual

Hemos completado este viaje (rápido) hacia lo profundo del procesamiento de la información visual en diferentes áreas del sistema nervioso de diferentes especies de animales. Casi todos los estudios que hemos citado utilizan un marco metodológico similar, que es medir la tasa de disparo de neuronas individuales localizadas en distintas áreas para tratar de interpretar qué función tienen y cómo responden frente a determinados estímulos. Sin embargo, a pesar de las ventajas de este enfoque funcional, resulta necesario tratar de entender cómo las neuronas adquieren tal selectividad y estructuración. En lo que respecta a esta parte, revisaremos algunos de los modelos téoricos relevantes tanto en ámbitos propios de las Neurociencias como propuestas más bien computacionales, hasta discutir el grado de relación de los datos del comportamiento observado de las neuronas, los modelos biológicamente plausibles y las redes neuronales convolucionales que designan a este capítulo.

Hipótesis de las Dos Corrientes

Nuestra revisión observó que las áreas V2 y V4 estaban relacionadas con el reconocimiento de formas complejas, pero las áreas V3A y MT (V5) tenían una mayor relación con la detección de movimiento, pero también áreas como la Medial Superior Temporal (MST), ubicada en la Corteza Temporal Inferior [113]. Los estudios de la organización topológica del cerebro muestran que la Corteza Visual se ramifica en dos vertientes principales, dirigiéndose hacia la Corteza Temporal Inferior como se ha visto, pero también a la Corteza Parietal, tal como se muestra en la figura 3.22.

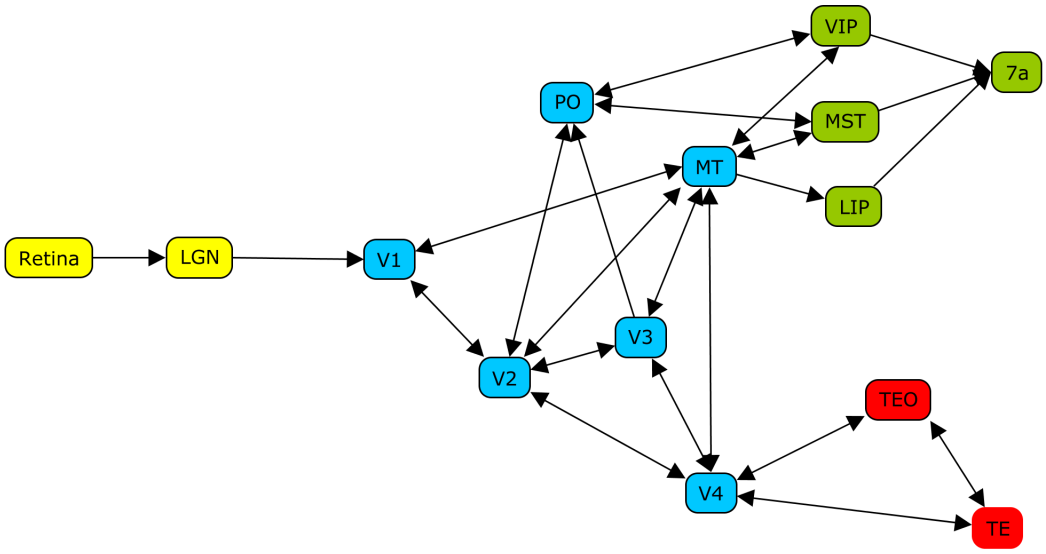


Figura 3.22: Organización en grafo de los módulos de procesamiento de información visual del macaco (basado en [72]). Los módulos azules representan a la Corteza Occipital, los verdes a la Corteza Parietal y los rojos a la Corteza Temporal Inferior. Para revisar un mapa más detallado, véase el estudio de [264]

Estas observaciones sobre la organización de las cortezas occipital (o visual), parietal y temporal inferior llevó a Goodale y Milner [73] a postular la Hipótesis de las Dos Corrientes, que dicta que la información visual es procesada por medio de dos vías diferentes, en orden jerárquico de acuerdo a las formulaciones que posteriormente abordaremos. Las corrientes son

- *Corriente Dorsal*: Está relacionada con la detección de movimiento y su velocidad.

Parte de las áreas V3A y V5 de la Corteza Visual y desemboca en la Corteza Temporal Inferior, donde tiene influencia en las acciones guiadas por la vista.

- *Corriente Ventral*: Se caracteriza por efectuar el reconocimiento de objetos, partiendo de rasgos simples procesados en la V1, rasgos complejos reconocidos en la V2 y V4, para finalmente reconocer categorías de objetos en la Corteza Temporal Inferior.

Nuevos estudios en organización topológica cerebral, sugieren una organización más compleja del encéfalo humano frente a otros primates, añadiendo una corriente adicional denominada como *lateral*, anteriormente entendida como parte de la dorsal, y relacionada con procesos tanto de la visión, como de la acción y el lenguaje [77].

Modelos jerárquicos de Hubel-Wiesel y Riesenhuber-Poggio

En cuanto a la Corriente Ventral (*Ventral Stream*), hemos observado que muchos estudios apuntan a un nivel de organización medianamente jerárquico, iniciando desde la Retina hasta la Corteza Temporal Inferior. Los mismos Hubel y Wiesel [112] postularon que la información procesada del LGN a las células simples y de las simples a las complejas indicaba una jerarquización. Por ejemplo, al unir campos receptivos de células de LGN, puedes obtener una respuesta similar a las de las células simples del área V1 (Figura 3.23).

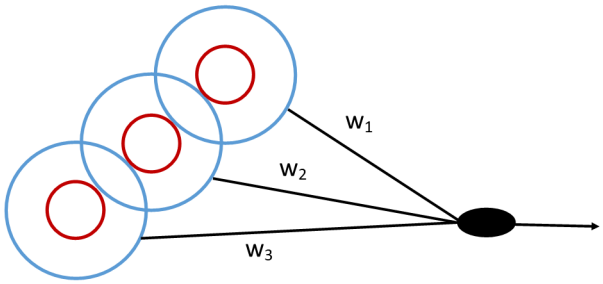


Figura 3.23: Esquema del modelo jerárquico de Hubel-Wiesel de las células de LGN a las células simples corticales. Basado en [112].

Patrones más complejos son estudiados por la V2 y V4, incluyendo ángulos que pueden ser entendidos con la unión de campos receptivos de la V1. Este modelo ha sido extendido

para las áreas V2 y V4 en el artículo de [190], donde se plantea que las imágenes de entrada son procesadas primeramente por filtros lineales, y después por sus combinaciones y luego patrones más complejos.

El modelo jerárquico ha sido sistematizado por Riesenhuber y Poggio en los artículos [212, 213] y también discutida por Serre en [225], siendo HMAX una versión computacional del mismo, aunque igualmente simplificada. Como modelo del procesamiento de la información visual de la Corriente Dorsal, señala que después de incrementar la complejidad de los patrones detectados de la retina a la V4, la complejidad de los patrones aumenta hasta la Temporal Inferior posterior (PIT), donde se pueden reconocer objetos no invariantes a rotaciones o escalas, y posiblemente de ahí a la Temporal Inferior Anterior, donde los objetos son invariantes a escalas. El modelo jerárquico original de Riesenhuber-Poggio consideraba una última capa en la Corteza Prefrontal, pero nueve años más tarde se publicó el estudio de Quian Quiroga y sus colaboradores [207] sobre la MTL, que nos permite completar el modelo jerárquico en asociación con la información procedente de la corteza auditiva. De esta forma, en un lapso de aproximadamente sesenta años se ha ido completando el estudio de la tarea del reconocimiento de objetos en el cerebro humano, quizá uno de los hitos más destacados en las Neurociencias. Por el número de áreas y neuronas involucradas, reconocer objetos parece ser una de las tareas más demandantes para el cerebro humano, y tal vez, una de las más significativas para su modelación computacional.

Modelos computacionales

Desde los primeros descubrimientos del funcionamiento de la Corteza Visual, así como los modelos explicativos como el jerárquico, se han propuesto implementaciones o modelos computacionales del procesamiento realizado por las neuronas de los diferentes estadios de procesamiento, destacando autores como Marr [172], quien tuvo influencia notable en el desarrollo de la Visión Computacional. A partir de esos primeros avances, se describió que una operación importante realizada por las neuronas era la convolución, y Marr mismo propuso algunos filtros para la extracción de características simples.

Para modelar la extracción de bordes de las células retinales y del LGN, se pueden utilizar filtros como el Sombrero Mexicano o Diferencia de Dos Gaussianas [117]. Para el

caso de las respuestas de células simples, se ha propuesto utilizar a los filtros de Gabor [118, 178, 225], dados por

$$w_{ij} = e^{-\frac{(x^2+\gamma^2 y^2)}{2\sigma^2}} \cos\Big(\frac{2\pi}{\lambda}x\Big), \tag{3.13}$$

donde $x = i\cos(\theta) + j\sin(\theta)$ y $y = -i\sin(\theta) + j\cos(\theta)$, y los hiperparámetros son la orientación θ, el radio de aspecto γ, el ancho efectivo σ, la fase ϕ y la longitud de onda λ. Al tratarse de imágenes, los pesos w_{ij} se aplican como pesos de filtro de convolución bidimensional.

Una primera aproximación computacional es tratar de implementar estos filtros para efectuar la extracción de características. Un problema de este enfoque directo es que no considera la plasticidad de las neuronas y es necesario completar los modelos de los filtros correctos en áreas posteriores a la V1. La versión computacional del modelo HMAX propuesto por Riesenhuber y Poggio [212] es uno de los modelos que sigue este ímpetu original de utilizar los filtros conocidos para implementar computacionalmente algoritmos de reconocimiento de objetos, que es una de las motivaciones fuertes del estudio de la Corteza Occipital.

Una versión extendida del modelo HMAX fue publicada en [243], el cual tiene elementos propios de las redes convolucionales, incluyendo capas convolucionales y de submuestreo. La primera capa convolucional cuenta con filtros de Gabor, la segunda es una aplicación de un *Max Pooling*, la tercera aplica filtros HL, los cuales son una combinación de filtros de Gabor y finalizan con un nuevo *Max Pooling* y una capa de clasificación. Existen otras posibilidades como el B-HMAX [269]. Una revisión de otras propuestas basadas en HMAX aparecen en [156]. Otro modelo computacional, que considera los llamados mapas autoorganizados y la plasticidad es denominado como LISSOM o *Laterally Interconneted Synergetically Self-Organizing Map* [178]. Mención importante merece el Neocognitrón de Fukushima [69], el cual tuvo influencia posterior en el desarrollo de las arquitecturas LeNet. Otros modelos con plasticidad resultan ser los que aplican la Regla de Hebb en redes convolucionales auténticas o redes pulsantes, que serán abordados en su correspondiente capítulo.

Redes Neuronales Convolucionales

El modelo Extendido HMAX puede considerarse como una red convolucional con filtros conocidos. Es un hecho bien conocido que los filtros de Gabor pueden sustituir a la primera capa de las redes convolucionales [35, 140] (revísese también [4]). Si bien el modelo jerárquico original contempla la detección de aspectos muy complejos, profundos en términos de capas, las implementaciones realizadas de los modelos HMAX rara vez superan cuatro capas, debido quizá a la carencia de modelos exactos sobre los filtros posteriores. Esto propicia que se traten de implementaciones incompletas con respecto al modelo jerárquico original.

Se han observado algunas limitaciones al modelo jerárquico, algunas encontradas en los ya citados estudios de Desimone sobre la Temporal Inferior [53] y la relación entre las áreas V1-V2-V4 revisada en [91]. En general, los patrones simples pueden aparecer en capas más profundas como la Temporal Inferior, aunque esto puede deberse a tener cierta necesidad de preservar patrones simples en capas posteriores. Otro problema del modelo general es que la organización en grafo (topológica) de las áreas visuales no es tan simple. Las investigaciones detalladas de organización topológica del cerebro como [264] evidencian que áreas primarias como la V1, tienen conexiones con áreas poco más profundas como el área V4.

Desde el punto de vista de neurocientíficos como [225], las redes convolucionales no son un modelo estricto de la Corteza Visual, pero al tratarse de una jerarquización del procesamiento de la información, su éxito aporta evidencia favorable al Modelo Jerárquico, computacionalmente incompleto. Sin embargo debemos enfatizar que la diferencia existente entre las CNNs y HMAX radica en que las primeras no usan los filtros que se han verificado como presentes en las etapas tempranas de la Corriente Dorsal. En contraparte, las redes convolucionales tratan de obtener los pesos que minimicen funciones de costo definidas previamente. En este sentido, las CNNs tienen pesos que pueden disminuir su error de clasificación, frente a los pesos fijos de otros modelos computacionales, que no explican cómo las neuronas se adaptan para tomar tales salidas.

Adicionalmente, si visualizamos a los patrones que generan una respuesta máxima en las neuronas de diferentes capas convolucionales de redes profundas, podemos notar

que desarrollan selectividad a patrones cada vez más complejos, empezando por patrones geométricos (algunos semejantes a los que aparecen en la V1), hasta incluir formas distorsionadas de objetos complejos [133, 169], observación que aparece incluso en los primeros artículos de Yann Lecun [144]. Algunos de los patrones intermedios no tienen una interpretación tan simple, lo cual puede complicar la tarea de tratar de encontrar los patrones intermedios en la corteza visual o inferior temporal. Si bien no son exactamente los mismos filtros que se usan en la naturaleza, sí logran encapsular los aspectos generales del modelo jerárquico.

Por otro lado, lo anterior confirma que las arquitecturas convolucionales lineales (al estilo de la VGG) mantienen una relación cercana con el modelo jerárquico. Aún más, algunas arquitecturas convolucionales disponen de una organización en grafo más compleja, que puede tratarse de una similitud con respecto al funcionamiento de la Corteza Visual, que parece conectar áreas jerárquicamente inferiores con superiores. Eso nos da pie a considerar arquitecturas como las ResNets o DenseNets como modelos aún más precisos de la Corteza Visual.

3.7. Conclusiones: una defensa del enfoque

A través de este capítulo se ha puesto la formulación de las redes neuronales convolucionales, introduciendo algunas arquitecturas representativas desarrolladas en la última década. Sin embargo, el denominador común del capítulo no son del todo las redes convolucionales sino el reconocimiento de objetos, tema al que nos hemos enfocado tanto para abordar a las CNNs como a los estudios sobre la visión.

Hemos destinado una parte importante de esta sección a la Visión biológica, ejercicio que se consideró necesario para tratar de discutir el enfoque que tomará esta tesis. Por un lado hemos visto que las redes convolucionales han tenido un éxito notable no sólo en tareas como reconocimiento de objetos, que consistió en su primer objetivo, aún sin tratarse de un modelo biológico completo.

En la tabla 3.3 hemos incluido una comparación de las redes convolucionales con el modelo HMAX y la visión en diversos ejes. Como podemos notar, las CNNs tienen aspectos

destacables frente a modelos que han sido subrayados como más plausibles. Quizá tales modelos son más precisos en cuanto a su formulación, pero no incluyen suficientes neuronas como para efectuar un reconocimiento de grandes categorías de objetos. Por otro lado, las redes biológicas siguen teniendo mayores capacidades, puesto que aunque en las tarea de reconocer 1000 imágenes pueden existir redes profundas que superan la capacidad humana, el número de objetos que el cerebro humano puede discriminar es mucho mayor. Los 42000 lemas (raíces lingüísticas) reconocidos por una persona promedio de 20 años dan una idea de tal capacidad [31]. Si bien no todos esos lemas corresponden a objetos concretos de la naturaleza, nos permiten darnos una idea de las capacidades de reconocimiento y abstracción que cuenta el cerebro humano.

¿Qué permite a las redes neuronales convolucionales funcionar adecuadamente frente a otros modelos propuestos en la literatura, ya sea biológicamente plausibles o no? Las redes convolucionales siguen un esquema que se asemeja un poco al modelo jerárquico que intenta explicar el funcionamiento de la visión en primates. Hemos discutido de igual forma que las CNNs cuentan con características que les permiten posicionarse en el marco de la modelación de aspectos biológicos. Conteniendo una enorme cantidad de capas y neuronas, estas redes superan por mucho a los modelos alternativos.

En general, los estudios considerados sobre la visión han sido abordados desde el punto de vista de la tasa de disparo o *firing rate*. Este enfoque será preferido por su simplicidad, frente a modelos como el de las redes neuronales pulsantes (SNN). En gran medida, podemos alegar que el éxito de las redes convolucionales nos da atisbos para valorar la adición de ideas procedentes de la biología a las redes neuronales. En términos generales, esto coincide con el enfoque general de esta tesis: partir de modelos preexistentes que hayan tenido éxito en sus niveles de exactitud, para añadir ideas con motivación biológica significativa que permita reducir la artificialidad de los modelos.

Cuadro 3.3: Comparación entre las redes neuronales convolucionales profundas, el modelo HMAX y la Visión humana

Aspecto	CNNs	HMax	Visión
Tipo de red	ANN	ANN	BNN (Biológicas)
Aprendizaje	Métodos basados en el gradiente	Fijo	Plasticidad
Capas	Mayor a 8	4	6 (pero con varias iteraciones)
Capacidad de reconocimiento	1000 categorías	256 categorías (Menos de 45 % de exactitud)	42000 lemas [31].

Capítulo 4

Redes Neuronales Evolutivas

Redes Evolutivas con modelos de Lotka-Volterra

Nature optimizes

Jorge Nocedal y Stephen Wright [187]

En los capítulos 2 y 3 hemos introducido a las redes neuronales multicapa y cómo estas capas densas (multicapa completamente contectadas) puede formar parte de una red neuronal convolucional. Las redes neuronales cuentan con parámetros a optimizar que son conocidos como los *pesos* de la red o *weights*. Tales pesos se entrenan normalmente mediante métodos basados en el gradiente. Otro aspecto optimizable de la red son los hiperparámetros, los cuales son variables de la red distintas a los pesos que no entran en consideración en el entrenamiento. Tanto los hiperparámetros como los parámetros definen al desempeño de la red, aunque la optimización tradicional se ha enfocado en los pesos. La minimización de una función de costo asociada ha sido un enfoque principal, que engloba a los métodos basados en el gradiente, Newton y cuasi-Newton. Por lo tanto esta optimización está centrada en minimizar la función de pérdida sobre el conjunto de entrenamiento, función a la que se le añaden regularizadores para evitar un posible sobreajuste.

En teoría, una red con un mayor número de neuronas (por ejemplo, infinita) es más óptima para la minimización del costo sobre el conjunto de entrenamiento, puesto que en caso de añadir neuronas innecesarias simplemente sus pesos pueden reducirse a 0. De este modo, optimizar estos hiperparámetros puede reducirse a añadir más neuronas. No obstante, tal como ocurre con el caso de las redes convolucionales (las cuales pueden entenderse como redes densas a las que se le eliminaron neuronas y con pesos compartidos (*weight sharing*)), añadir neuronas arbitrariamente no necesariamente mejora a la clasificación. Esto sucede porque solamente puede reducir la pérdida sobre el conjunto de entrenamiento, pero esto no es necesariamente cierto para el conjunto de prueba, donde el error puede incluso incrementar. Por lo tanto, la arquitectura no sólo juega un rol notable en el desarrollo de redes neuronales, sino que su optimización no es un asunto trivial. De acuerdo con [265], no existe una fórmula explícita para lograrlo y tradicionalmente (hasta el 2011) se ha efectuado de forma empírica [21, 20].

¿De qué forma podemos mejorar las capacidades de las redes dado un problema específico? Se discutirán en esta sección formas de optimizar los hiperparámetros de la red, particularmente la arquitectura de las mismas. Entre los hiperparámetros de una red FNN densa se encuentra el número de capas y el número de neuronas por capa, las cuales definen su arquitectura, los cuales recibirán especial atención. Para las redes convolucionales, existen otros hiperparámetros como el número y tamaño de los filtros. Como se verá, existen ciertos métodos que permiten la optimización de arquitecturas. Particularmente nos centraremos en los algoritmos de tipo evolutivo, por su motivación biológica. Cada red puede ser entendida como un individuo independiente perteneciente a un ecosistema artificial. De esta forma, así como la optimización paramétrica es equivalente al aprendizaje de la red, la optimización hiperparamétrica se asociará al término de *evolución de redes neuronales*. La configuración de las neuronas y sus conexiones en el cerebro humano debe entenderse como resultado de la evolución, no así el proceso de aprendizaje, el cual como se ha visto, está más asociado al ajuste de pesos de la red.

En esta sección introduciremos un algoritmo de evolución relacionado con modelos ecológicos, en particular el modelo de Lotka-Volterra. Consideraremos a las redes neuronales como organismos propios, con un código genético particular, los cuales siguen un

proceso de evolución. Tales redes pueden comportarse como presas (o incluso como depredadores) en un ecosistema artificial, por lo que su número estará restringido por las ecuaciones de Lotka-Volterra o una generalización de las mismas. Esto posibilitará evitar un crecimiento exponencial del número de presas, mejorando la complejidad del algoritmo evolutivo.

4.1. Estudios previos

Entre los métodos de optimización hiperparamétrica destacan [265]:

1. *Búsqueda manual*: Es el enfoque ingenuo, consistente en calibrar el número de neuronas de forma empírica.

2. *Búsqueda por rejilla (Grid Search):* Se trata de modificar los hiperparámetros realizando incrementos constantes en los mismos.

3. *Búsqueda aleatoria:* Propuesto en los artículos [21, 20], consiste en realizar incrementos aleatorios. Esta técnica demostró ser mejor que la búsqueda por rejilla.

4. *Optimización bayesiana*: ha sido aplicado por autores como [127].

5. *Métodos basados en el gradiente:* Una técnica utilizando SGD con momentum fue propuesta por Maclaurin, Duvenaud y Adams en el 2015 [166].

6. *Algoritmos genéticos*

Abordaremos el tema de algoritmos genéticos de manera especial, ya que cuentan con un mayor sustento biológico que los otros casos expuestos.

4.1.1. Algoritmos genéticos en Redes Neuronales

Los algoritmos genéticos (*genetic algorithms* o GAs) fueron originalmente propuestos por John Henry Holland en 1975 [99] y en su libro de 1992 [100], para simular un proceso de evolución. En estos algoritmos, los individuos representan configuraciones particulares

de los hiperparámetros, los cuales cuentan con genes artificiales que codifican a los hiperparámetros en forma de un vector. Una forma común de los algoritmos genéticos es el uso de las operaciones específicas en el siguiente orden [82]:

1. Inicializar con una población (conjunto) de individuos.
2. Calcular la aptitud (*fitness*) de cada individuo.
3. *Selección*: consiste en elegir a los individuos para cruzar, dependiendo de su aptitud.
4. *Cruzamiento:* Consiste en tomar dos individuos selectos y combinar sus genes artificiales para generar cruzamientos aleatorios de nuevos individuos [177]. Por ejemplo, si $I_1 = (GCCT)$ y $I_2 = (CCTA)$, entonces los resultados del cruzamiento pueden ser $I_3 = (CCCT)$ y $I_4 = (GCTA)$.
5. *Mutación:* Aplicación de cambios aleatorios a los genes artificiales de los individuos generados.
6. *Reemplazo:* Se reemplazan viejos individuos con los nuevos.
7. Repetir paso 2 hasta alcanzar condición de término.

Los algoritmos genéticos se han introducido en las redes neuronales tanto para optimización paramétrica como hiperparamétrica, en particular para el caso de redes neuronales convolucionales. Para el primer caso, destaca el trabajo de [246], en el cual se sugiere un ajuste de pesos utilizando primero un algoritmo genético y luego L-BFGS (Limited Memory Broyden-Fletcher-Goldfarb-Shanno). Sin embargo, los GAs han sido más aplicados en la optimización hiperparamétrica de CNNs, destacando propuestas como la NSGA-Net [162]. Otras aplicaciones más cercanas a los problemas de clasificación que nos competen por ahora aparece en la subsección siguiente.

4.1.2. El problema de clasificación de caracteres de EMNIST

EMNIST o *Extended MNIST*, es una colección de bases de datos propuesta por [44] en el 2017, que amplía o complementa a la base de datos MNIST [147]. Además de incluir

extensiones a la base de datos MNIST, cuenta con una base de datos referente a caracteres alfabéticos manuscritos contando con un total de 27 clases correspondientes a 27 letras mayúsculas, a la cual centraremos particular atención.

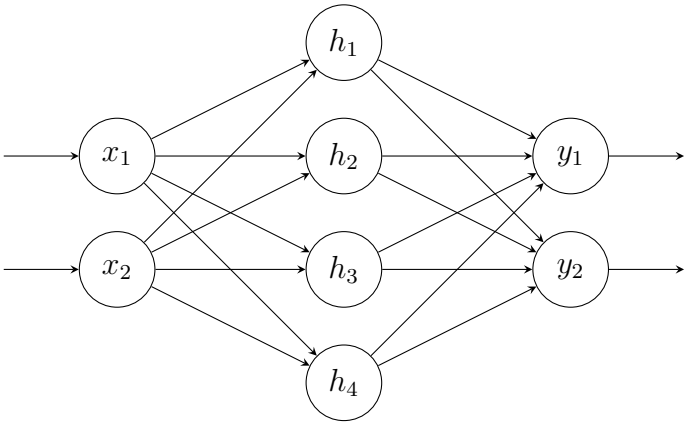
El método original propuesto por [44] consiste de una red de tres capas entrenadas con un algoritmo conocido como *Extreme Learning Machine* [106] y para los cálculos necesarios de la pseudo-inversa de matrices, debido al tamaño del *dataset*, incorporan una técnica conocida como *Online Pseudo-Inverse Update Method* (OPIUM) [249], logrando una exactitud de $85{,}15 \pm 0{,}12\,\%$ en la base de datos de letras, mientras que los clasificadores lineales arrojan apenas $55{,}78\,\%$.

De acuerdo con el *benchmarck* realizado en el 2019 por [16], los mejores métodos para resolver este problema de clasificación están basados en redes neuronales convolucionales, aunque existen propuestas diferentes. La mejor propuesta no convolucional para la clasificación de letras en NIST SD 19 (muy similar a EMNIST) es un algoritmo conocido como *Record-to-Record travel* y alcanza un $93{,}78\,\%$ de exactitud sobre el conjunto de prueba [210]. Para el caso de las redes FNNs densas, el trabajo de [128] alcanzó $87{,}79\,\%$ de exactitud, utilizando diversas características de las imágenes.

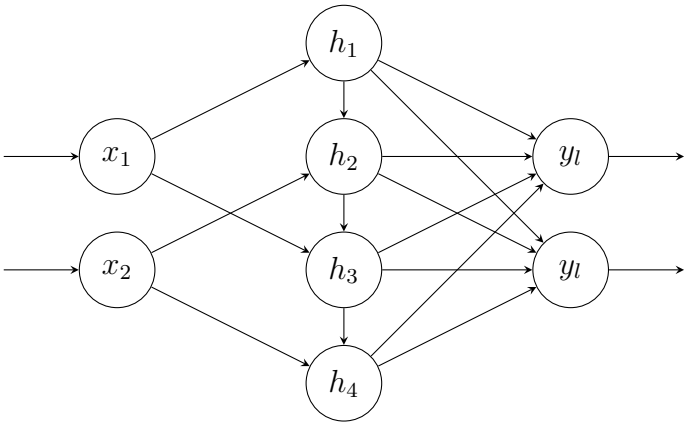
En cuanto a las redes convolucionales simples, la propuesta dada por [33] logra una exactitud de $93{,}63\,\%$ utilizando dos capas convolucionales y una densa. La aplicación de algoritmos genéticos ha estado patente en los comités de redes convolucionales en la propuesta de [15], alcanzando una exactitud de $95{,}36\,\%$, siendo una de las más altas registradas. En el *benchmarck* de [16], la mayor exactitud encontrada es la de [199], utilizando Campos aleatorios de Markov con redes convolucionales logrando una exactitud de $95{,}44\,\%$.

Entre las propuestas posteriores al 2019, destaca la SpinalNet, la cual alcanza la máxima exactitud que se ha registrado hasta el momento, de acuerdo con el *benchmark* realizado de manera aparte por `paperswithcode.com`, logrando una exactitud de $95{,}88\,\%$

La arquitectura de las Redes Espinales propuestas, o *SpinalNets* está inspirada en el funcionamiento de la médula espinal, a pesar de haber sido aplicada en problemas de clasificación de imágenes. Una red neuronal multicapa densa cuenta con la siguiente arquitectura



Tal arquitectura de FNN puede convertirse a una estructura espinal de la siguiente manera:



Estas redes pueden utilizarse en conjunto con capas convolucionales y de submuestreo (*pooling*), para los problemas de clasificación de imágenes. Los resultados de estos algoritmos diferentes se muestran en la tabla 4.1.

4.2. Metodología

Cada red neuronal puede ser concebida como un organismo independiente en un ecosistema artificial. Su número puede ser modelado por medio de modelos ecológicos con ecuaciones diferenciales, permitiendo una simulación del comportamiento poblacional de una *especie* de redes neuronales. Cada individuo cuenta con sus propios hiperparámetros, los cuales están definidos por medio de una representación única que se denominará como

Algoritmo	Exactitud
Clasificador lineal [16]	55,78 %
Tres capas + OPIUM	$85{,}15 \pm 0{,}12$ %
Características + FNN [128]	87,79 %
Record-to-Record Travel [210]	93,78 %
CNN (2 conv + 1 densa) [33]	93,63 %
Comités de CNNs [15]	95,36 %
Campos aleatorios de Markov + CNN [199]	95,44 %
SpinalNet (CNN) [120]	95,88 %

Cuadro 4.1: Resumen del estado de arte del problema de clasificación de letras EMNIST

código genético, el cual, como en su homólogo natural, varía de forma aleatoria. El *fitness* determina sus posibilidades de supervivencia y está dado por la exactitud en el conjunto de validación, o cualquier otra métrica conveniente (F1-Score, por ejemplo). La forma en que estos parámetros se codifican dependerá necesariamente del tipo de red neuronal asociado, que se conocerá como *especie*. Especies de redes neuronales figuran las Redes Neuronales Densas de Alimentación hacia adelante (FNNs), las redes convolucionales (CNNs), las redes LSTMs, entre muchas otras.

4.2.1. Evolución de FNNs

Una forma simple de codificación de FNNs está basada en el número de capas ocultas y el número de neuronas en las mismas. Para ello definiremos una sucesión $(s_1, s_2, \ldots, s_n, 0, \ldots)$ donde s_i representa el número de neuronas de la capa i. Es conveniente imponer la restricción $s_i = 0$ para $i > n$, donde $n \in \mathbb{N}$ representa el número de capas. Esta sucesión, que por conveniencia representaremos como $(s_1, s_2, \ldots, s_n)$, designará al *código genético* de la red FNN.

Mutaciones

Distinguiremos cuatro tipos importantes de mutación, las cuales se enuncian a continuación:

- α: Agrega k neuronas a una capa aleatoriamente escogida.

$$(r_1, \ldots, r_i, \ldots, r_n) \to (r_1, \ldots, r_i + k, \ldots, r_n). \tag{4.1}$$

- β: Agrega una capa con k neuronas.

$$(r_1, \ldots, r_n) \to (r_1, \ldots, r_n, k). \tag{4.2}$$

- γ: Reduce k neuronas en una capa específica o la vuelve k.

$$(r_1, \ldots, r_i, \ldots, r_n) \to (r_1, \ldots, \text{máx}(r_i - k, k), \ldots, r_n). \tag{4.3}$$

- δ: Elimina la última capa:

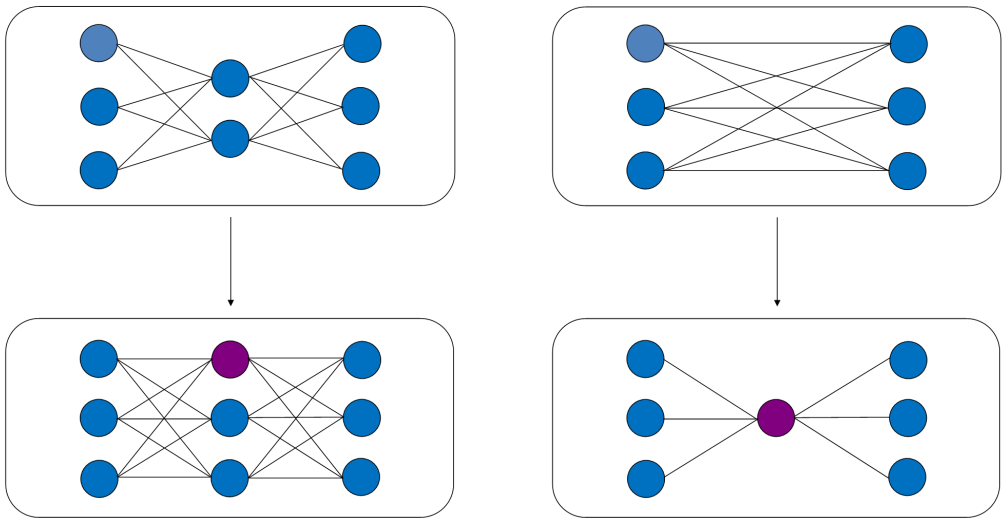
$$(r_1, \ldots, r_n, k) \to (r_1, \ldots, r_n). \tag{4.4}$$

Cuando $k = 1$, las mutaciones pueden recorrer todo el hiperespacio $\mathbb{R}^\infty$. La mutaciones α y β aparecen ilustradas en la figura 4.2. Las mutaciones γ y δ únicamente se emplearán cuando se inicie una evolución con una red neuronal dada cuyo código genético no sea (0). En caso de que el (0) forme parte de la primera generación, no existe necesidad matemática para aplicar deleciones (mutaciones que eliminan parte del código, en este caso γ y δ).

Cada tipo de mutación x tendrá una probabilidad de elección p_x que será controlado de acuerdo a las proporciones esperadas deseadas del tipo que se efectúen: una mayor p_β conducirá a redes más profundas. En general, parece conveniente un sistema $p_\delta \leq p_\gamma < p_\beta < p_\alpha$ para permitir un mayor crecimiento de las capas ya creadas y evitar excesivas capas con escasas neuronas.

4.2.2. Evolución de CNNs

La evolución de CNNs puede lograrse de dos formas. La primera consiste en diseñar una red convolucional fija y permitir su evolución como individuo, variando la última capa densa. Este procedimiento es adecuado en redes convolucionales estándares, las cuales



Cuadro 4.2: Mutación de tipo α (izquierda) y β (derecha) con $k = 1$

después de las capas convolucionales intercaladas con las de submuestreo (*pooling*), se siguen de capas densas, como es el caso de las LeNet [146], AlexNet [133] y VGGs [228].

Una segunda aproximación consiste en evolucionar a la red convolucional misma desde las capas convolucionales. Este enfoque, sin embargo, no será abordado por el momento.

4.2.3. Reproducción

La reproducción de los organismos puede llevarse por medio de diferentes estrategias, al que inicialmente consideraremos la fisión binaria. La principal desventaja de este mecanismo es que es realizada por seres vivos relativamente simples, y la mayor parte (o todos) de los organismos con sistema nervioso cuentan con mecanismos de reproducción sexual. En algunos casos, el propio sistema nervioso es responsable de efectuar la selección sexual, eligiendo parejas con colores más llamativos en el caso de las aves, por ejemplo, implica un reconocimiento de colores. Es posible que esto se deba a que los organismos no cuentan con alguna forma precisa de predecir la aptitud o *fitness*, sino que los organismos que llegan a la madurez pueden ser seleccionados, pero a diferencia de los programas, los organismos vivos no sólo están sujetos a optimizar un problema conciso como clasificar caracteres, sino simplemente sobrevivir, lo cual implica resolver múltiples problemas de

clasificación de manera simultánea.

Por motivaciones principalmente matemáticas utilizaremos a la fisión binaria como mecanismo inicial. La reproducción en este contexto de un organismo obtiene como resultado a dos organimos hijos: uno con el código genético exactamente igual al padre y otro con una mutación. Iniciando con un organismo (0) (que carece de capas intermedias), se obtiene a (0) y (k) en la siguiente generación. Ejemplo de lo mencionado para FNNs aparece en la Figura 4.1.

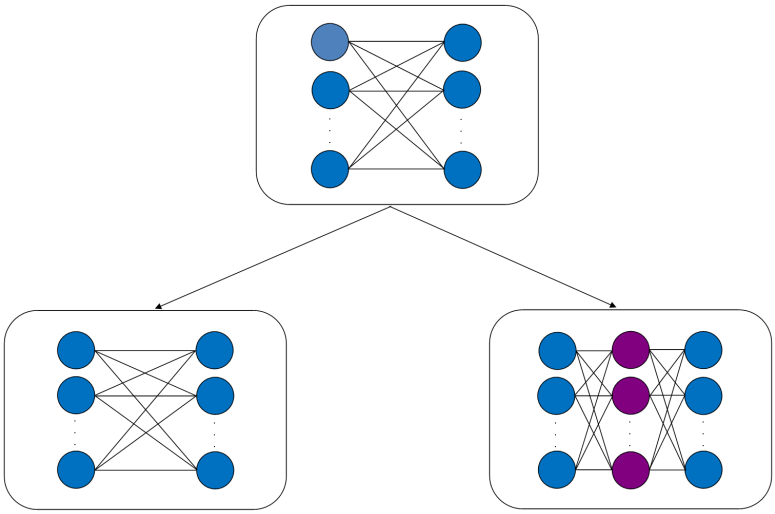


Figura 4.1: Duplicación del organismo (0) en (0) y (k).

4.2.4. Modelo de Lotka-Volterra

El algoritmo presentado previamente para $k = 1$ recorre de manera efectiva el hiperespacio de las soluciones por lo que teóricamente converge a un máximo local de exactitud, pero cuenta con la desventaja de tener una alta complejidad. Si consideramos el coste computacional como al número de entrenamientos (que es la operación más costosa) obtenemos que la complejidad del problema es de orden $O(t) = 2^t$. En la práctica podemos no efectuar todos los entrenamientos sino que exclusivamente a las nuevas configuraciones. Sin embargo cada organismo aporta una nueva red, por lo que $O(t) \approx 2^{t-1}$, aunque este número es una cota superior pues algunas mutaciones pueden repetir configuraciones

anteriores. No obstante, la complejidad sigue manifestando un desempeño no polinomial, lo cual propicia una convergencia lenta hacia un máximo global.

Una solución a este problema es introducir depredadores artificiales que regulan la cantidad de organismos en el ecosistema artificial. Estos depredadores pueden actuar de forma estocástica eliminando a las presas con una probabilidad asociada al *fitness* o bien de forma determinista, eliminando solamente a las presas con menor *fitness*. El enfoque estocástico tiene la ventaja de ser un modelo quizá más natural, pero la convergencia a máximos no estaría garantizada, por lo que el modelo determinista, aún con otros aspectos aleatorios, resulta conveniente para probar ciertas propiedades.

Una forma de emular este proceso de depredación es utilizando a las ecuaciones de Lotka-Volterra (LV), o bien conocidas como el modelo de Depredador-Presa. En forma general el modelo de LV está dado por

$$\frac{dx}{dt} = af(x) - bg(x,y)y, \tag{4.5}$$

$$\frac{dy}{dt} = cg(x,y)y - hy, \tag{4.6}$$

donde $x(t)$ denota la densidad de presas sobre el tiempo t, $y(t)$ es la densidad de depredadores sobre el tiempo, $f(x)$ es la razón de crecimiento poblacional de las presas en ausencia de depredadores, $g(x,y)$ representa a la respuesta funcional del depredador frente a su presa, a es la tasa de crecimiento de la población, b es la tasa de competencia, c representa la tasa de crecimiento del depredador y h a la tasa de muerte del depredador [70]. Una versión simple de este modelo es utilizar un crecimiento exponencial de las presas $f(x) = x$, comportamiento que se ve reflejado en la fisión binaria, de donde podemos deducir un parámetro. En ausencia total de depredadores y sin considerar una capacidad de carga (que llevaría a un modelo logístico), se tiene que $\frac{dx}{dt} = ax$. Como la población de presas aumenta de forma $x(t) = 2^t$, entonces esta ecuación es solución de la ecuación diferencial simple, de donde al derivar obtenemos $\frac{dx}{dt} = a2^t$, de donde $\ln 22^t = a2^t$, por lo que $a = \ln 2$. Si consideramos a la interacción de forma igualmente simple obtenemos $g(x,y) = x$, por lo que el sistema final está dado por el sistema de ecuaciones diferenciales

$$\frac{dx}{dt} = \ln 2\ x - bxy, \tag{4.7}$$

$$\frac{dy}{dt} = cxy - hy. \tag{4.8}$$

Usando Método de Euler, podemos alcanzar una simulación del sistema, expuesto en la Figura 4.2.

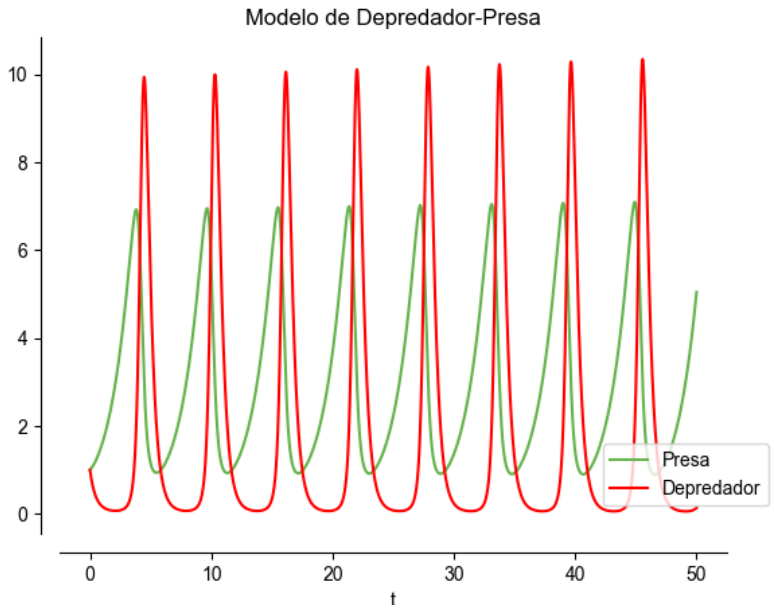


Figura 4.2: Simulación del Modelo de Lotka-Volterra utilizando Método de Euler

4.2.5. Discretización del modelo LV

De la simulación del modelo de Lotka-Volterra podemos observar algunos detalles notables. El primero es que las soluciones son periódicas, hecho que parece distorsionarse con el aumento de t, principalmente por problemas numéricos derivados de la aplicación del método de Euler. Sin embargo, como se demuestra [79] (como consecuencia del Teorema 1 del artículo, *Toda solución del sistema de LV es una órbita cerrada, exceptuando para los puntos de equilibrio y los ejes coordenados*), las soluciones del sistema son periódicas, exceptuando para los puntos de equilibrio y los ejes coordenados.

De esta manera podemos redefinir los parámetros b, c y h involucrados, a otros más simples de discretizar. Como vemos, existe un periodo de aumento, que puede aproximarse a una exponencial y luego decrece rápidamente a un punto mínimo. Sea T el periodo de LV, el cual ya sabemos que existe. Definamos

$$M = \arg \max_{t \in [0,T]} x(t), \tag{4.9}$$

$$m = \arg \min_{t \in [0,T]} x(t). \tag{4.10}$$

Es decir, M representa el punto del primer periodo donde x alcanza su máximo, mientras que m es el punto donde alcanza su mínimo. Finalmente definamos $\mu_m = x(m)$, que es el número mínimo de presas. Con estas cuatro nuevas constantes podemos discretizar LV. En la práctica, podemos calcular esas constantes a partir del modelo de LV y en general es posible que no exista un modelo para cualesquiera constantes. Sin embargo, esta representación es conveniente porque nos permite describir un algoritmo en términos simples para emular el comportamiento de LV, el cual está dado por:

1. Aumentar la población de presas exponencialmente utilizando $f(t) = 2^t$ (fisión binaria) hasta que $t = M$

2. Cuando $t = m$ reducir el número de presas hasta dejar μ_m presas con mayor *fitness*

3. Aplicar $m \leftarrow m + T$ y $M \leftarrow M + T$ y repetir el paso 1

donde la constante t_{max} define cuántas generaciones se van a considerar. t por lo tanto, en el mundo discreto, representa a las generaciones de redes neuronales presas. El crecimiento de la población de presas es regulado por los depredadores de forma cíclica, de forma que en los intervalos de crecimiento pueda permitir mayor diversidad genética y luego reducirse para dejar solamente a los mejores. Una versión formalizada de este método es el algoritmo 1.

La variación de las constantes M, t_{max} y μ_m no es una forma optimizar hiper-hiperparamétricamente, ya que a diferencia de los hiperparámetros, el aumento o disminución de algunos valores conduce a mejorar directamente a la exactitud sacrificando tiempos de iteración,

Algoritmo 1 Evolución de redes neuronales con Modelo de Depredador-Presa

Entradas: Constantes t_{max}, M, μ_m, Conjuntos de entrenamiento y validación

Salidas: Configuraciones de capas de redes neuronales con mejor rendimiento

```
1: Initialization t ← 0
2: while t < t_max do
3:    if t = M then
4:       t ← t + 1
5:       μ ← número de organismos
6:       Eliminar μ − μ_m organismos con menor rendimiento
7:       M ← M + T
8:    else
9:       Duplicar la población de presas
10:      Calcular el rendimiento de cada presa
11:   end if
12:   t ← t + 1
13: end while
```

sin que ello conduzca necesariemente a un problema de tipo *overfitting*. Si tomamos una mayor μ_m estamos permitiendo que más especies sobrevivan por cada eliminación, lo cual deja que más ejemplares se reproduzcan, pero aumenta el número de los mismos. De forma similar si aumentamos M iteramos más reproducciones, y si incrementamos t_{max} haremos más generaciones. Estos cambios dependerán del tiempo de ejecución que se utilice, así como los recursos que se dispongan.

4.2.6. Complejidad de la solución

Anteriormente hemos discutido que la complejidad del problema en ausencia de depredadores es de orden exponencial, a pesar de que para $k = 1$ converge a un máximo global de una función de evaluación e del conjunto de validación. La introducción de depredadores es una forma en la que se puede disminuir este problema. Otra posible salida es introducir un modelo logístico.

Veamos que la adición de depredadores convierte al algoritmo en uno de orden P, al acotarlo con un polinomio.

Proposición 3. *La complejidad del Algoritmo 1 es de clase P. Más aún, sigue la siguiente relación*

$$O(t) < 2^M + 2^T \mu_m \Big(\frac{t-M}{T} + 1 \Big). \tag{4.11}$$

Demostración. Supongamos que $t \leq M$. Entonces se efectúan 2^t entrenamientos, pero $2^t \leq 2^M \leq 2^M + 2^T \mu_m \Big(\frac{t-M}{T} + 1 \Big)$, pues las constantes son positivas. Si $t > M$ entonces existe una eliminación de organismos cada $\lfloor \big(\frac{t-M}{T} \big) \rfloor$ veces. En cada una de esas veces se produjeron $2^T \mu_m$ organismos, por lo que si $\big(\frac{t-M}{T} \big) \in \mathbb{N}$, entonces la complejidad está dada exactamente por

$$O(t) = 2^M + 2^T \mu_m \Big(\frac{t-M}{T} \Big). \tag{4.12}$$

Por lo tanto si t^* satisface que $\big(\frac{t^*-M}{T} \big) \in \mathbb{N}$ y además cumple que

$$\lfloor \big(\frac{t^*-M}{T} \big) \rfloor = \lceil \big(\frac{t-M}{T} \big) \rceil, \tag{4.13}$$

entoces $t < t^*$. Por lo tanto,

$$\lceil \big(\frac{t-M}{T} \big) \rceil = \lfloor \big(\frac{t-M}{T} + 1 \big) \rfloor. \tag{4.14}$$

Como $O(t) < O(t^*)$, entonces

$$O(t) < 2^M + 2^T \mu_m \Big(\frac{t-M}{T} + 1 \Big). \tag{4.15}$$

□

Este resultado es sumamente importante porque convierte un algoritmo de clase NP en uno de clase P, incluso es linealmente acotado. En las líneas siguientes mostraremos que esta ganancia lleva a un sacrificio: la convergencia hacia un máximo global no está garantizada, porque si se llega a un máximo local entonces cada nueva generación tendrá

menor *fitness* y podrían ser depredados todos los individuos distintos al de mayor exactitud, evitando un eventual aumento. Sin embargo, probaremos que es posible llegar al menos a un máximo local de la exactitud.

4.2.7. Convergencia para $k = 1$

Para $k = 1$ es posible observar una convergencia hacia un máximo local. Para otros valores de k esta convergencia no está asegurada, pero realiza un muestreo más amplio del hiperespacio, acercándose a valores posiblemente más grandes. Definamos el espacio ℓ_f de sucesiones finitas dado por

$$\ell_f = \{(s_n)_{n\in\mathbb{N}} \mid s_n \in \mathbb{N}, \exists m : s_m = 0\}. \tag{4.16}$$

Cada elemento de ℓ_f representa un código genético de una red neuronal. Cada elemento puede ser evaluado sobre el conjunto de validación utilizando una función de evaluación $e : \ell_f \to [0, 1]$, la cual se pretende maximizar. Formalmente, el algoritmo de evolución toma un conjunto inicial de códigos genéticos $\mathcal{C}_0$ y en cada iteración se produce una nueva generación de códigos genéticos (o bien, de redes neuronales), con mejor evaluación. De esta manera, el algoritmo es una función $\Xi : 2^{\mathcal{C}_i} \to 2^{\mathcal{C}_i}$ tal que

$$\mathcal{C}_{i+1} = \Xi(\mathcal{C}_i) \tag{4.17}$$

donde si $(s_i)_1, \ldots, (s_i)_{\mu_m} \in \mathcal{C}_i$ son los códigos con mayor evaluación, entonces $(s_i)_1, \ldots, (s_i)_{\mu_m} \in \mathcal{C}_{i+1}$, ya que en la operación de depredación eliminamos a todos exceptos los de mayor evaluación. Sin embargo, es posible que más de μ_m redes compartan la misma evaluación, para lo cual se eliminarán algunos códigos con mayor *fitness*. En la práctica, se puede añadir un ruido que permita mantener redes con distinta evaluación. Esto permite asegurarnos de que cada red tenga un diferente *fitness*, por lo que el máximo escogido (aleatoriamente) no se reemplazará sino por otro máximo. Esta consideración será importante. Definamos también

$$m_i = \text{máx}\{e((s_n))|(s_n) \in \mathcal{C}_i\}. \tag{4.18}$$

Esta es la sucesión de máximos en cada generación. Probemos que la función es no decreciente:

Lema 1. *(m_i) es no decreciente.*

Demostración. Sea $i \in \mathbb{N}$. Entonces como $\{e((s_n))|(s_n) \in \mathcal{C}_i\}$ es un conjunto finito, existe $(s_n) \in \mathcal{C}_i$ tal que $m_i = e((s_n))$. Si en la iteración $i+1$ se duplican los individuos, (s_n) se conserva. En caso de que reduzcan, dado que (s_n) tiene la mejor exactitud, solamente puede ser eliminado si existe $(r_n) \in \mathcal{C}_i$ tal que $e((r_n)) = m_i$. Por lo tanto, en todos los casos $(s_n) \in \mathcal{C}_{i+1}$, o bien, $(r_n) \in \mathcal{C}_{i+1}$ de donde $m_i \leq m_{i+1}$ □ □

Este lema es cierto para cualquier k. Si la población inicial satisface que $(0) \in \mathcal{C}_0$, para $k = 1$ se cumple la convergencia sin necesidad de utilizar mutaciones γ y δ.

Proposición 4. *Supongamos que la función e tiene un máximo global. Entonces (m_i) converge casi seguramente a un máximo local si $(0) \in \mathcal{C}_0$.*

Demostración. Por el Lema 1, la sucesión (m_i) es no decreciente. Ahora, veamos que si $m_i = m_j$ para $j > i$, entonces el algoritmo ha convergido a un máximo local. Por contradicción, supongamos que existe $(r_n) = \arg\max_{(s_n)\in\mathcal{C}_i} c((s_n))$, que no es máximo local, tal que $m_i = m_j$ para toda $j > i$. Como (r_n) no es máximo local, entonces existe (q_n) tal que $e((q_n)) > e((r_n))$ y $|(q_n) - (r_n)| = 1$, puesto que se tratan de sucesiones de naturales diferentes, por lo que difieren en solamente un punto. Notemos que $(r_n) \in \mathcal{C}_j$ para toda $j > i$. Por lo tanto, como $(q_n), (r_n) \in \ell_f$, ambas sucesiones son finitas. Si ambas se hacen 0 en el mismo índice, entonces se tiene que difieren en el índice t, de forma que

$$(r_n) = (r_n^1, \dots, r_n^t, \dots, r_n^d, 0, \dots), \tag{4.19}$$

$$(q_n) = (r_n^1, \dots, r_n^t + 1, \dots, r_n^d, 0, \dots). \tag{4.20}$$

Sea p_α la probabilidad de ocurrencia de la mutación tipo α. La probabilidad de que no ocurra el evento r_n muta a q_n es de $p_\alpha \frac{d-1}{d} + p_\beta$. Por lo tanto, la probabilidad de que no ocurra i veces está dada por $(p_\alpha \frac{d-1}{d})^i$, de donde

$$\lim_{i\to\infty}\left(p_\alpha\frac{d-1}{d}+p_\beta\right)^i=0, \tag{4.21}$$

ya que la constante $p_\alpha\frac{d-1}{d}+p_\beta<1$. Si ambas difieren en el índice en el que se hacen 0, entonces se tiene que

$$(r_n)=(r_n^1,\dots,r_n^d,0,0,\dots) \tag{4.22}$$

$$(q_n),=(r_n^1,\dots,r_n^d,1,0,\dots), \tag{4.23}$$

de donde la probabilidad evento de la mutación de (r_n) a q_n es p_β, por lo que la probabilidad de que no ocurra es p_α, por lo que la probabilidad de que no ocurra i veces está dada por $(p_\alpha)^i$, de donde $\lim_{i\to\infty}(p_\alpha)^i=0$.

Por lo tanto, (r_n) muta a (q_n) casi seguramente en alguna iteración $j>i$, de donde $(r_n)\in\mathcal{C}_j$ y por lo tanto $m_j\geq e((r_n))>m_i$, que contradice nuestra hipótesis original. Por lo tanto, si el algoritmo se estanca, ha llegado a un mínimo local.

En el paso anterior vimos que (m_i) crece casi seguramente si se estaciona en un máximo local. Por lo tanto es de la forma

$$m_1=\cdots m_{1+o_1}<m_2=\cdots \tag{4.24}$$

Por lo que tenemos una subsucesión estrictamente creciente. Como existe un máximo global y el dominio es discreto, no puede crecer indefinidamente. Por lo tanto (m_i) converge a un máximo local.

□

4.3. Resultados

Resulta necesario evaluar los algoritmos de forma experimental, con el objetivo de contrastar los resultados teóricos alcanzados. Para la aplicación del modelo de Lotka-Volterra se usarán los datasets MNIST y EMNIST. La división de las bases de datos se presenta en la Tabla 4.3.

Cuadro 4.3: División de las bases de datos consideradas

Dataset	Entrenamiento	Validación	Prueba
MNIST	50000	10000	10000
EMNIST	104000	20800	20800

Para las evaluaciones con el modelo LV, se diseñaron redes neuronales con entradas fijas de 28×28 y salida de 10 neuronas para el caso de MNIST y 27 para el caso de EMNIST. Se optimizó la función de costo de Entropía Cruzada Categórica Dispersa, utilizando el optimizador Adam.

Las constantes k, μ_m, M, T fueron variadas para observar su comportamiento en el tiempo y el costo. Como se había comentado, esto no es una forma de optimizarlos, ya que se pueden incrementar para observar mejores resultados, a costa de tener mayor tiempo de ejecución. Se consideraron las siguientes variantes para el caso de MNIST:

1. $A(k, \mu_m, M, T)$: Algoritmo con k fija.
2. $B(k_{\text{máx}}, \mu_m, M, T)$: Algoritmo con k aleatoria elegida en el intervalo $[1, k_{\text{máx}}]$.
3. $C(k_{\text{máx}}, \mu_m, M, T)$: Algoritmo con k aleatoria elegida en $[1, k_{\text{máx}}]$ y utilizando capas convolucionales fijas.

La introducción de aleatoriedad está inspirada en los artículos de Bergstra [21, 20], donde se sugiere el uso de una búsqueda aleatoria en lugar de una búsqueda constante por rejilla. La constante $k_{\text{máx}}$ es semejante en comportamiento a k, no se provee una regla para elegirlo, sino que al aumentarse se incrementa el espacio de búsqueda, mientras que al reducirse la búsqueda se realiza de manera más fina, lo cual se traduce en un mayor tiempo de ejecución.

En la variante C se involucra el uso de redes convolucionales, que experimentalmente presentan un mejor desempeño en el reconocimiento de imágenes, como se ha visto. La variante C, al partir de una red fija y no de (0), requiere de utilizar mutaciones γ y δ. Para el resto de las variantes, $\mathcal{C}_0 = \{(0)\}$.

La red inicial para C cuenta con la siguiente arquitectura:

1. Una capa convolucional con 169 filtros con kernels de tamaño 4×4 y activación ReLU. La entrada es de $28 \times 28 \times 1$.

2. Una capa convolucional con 36 filtros con kernels de 3×3 y activación ReLU.

3. Una capa de MaxPooling de 2×2.

4. Una capa densa de 30 unidades y activación ReLU.

5. Una capa de salida con 10 neuronas con activación Softmax.

De esta forma $\mathcal{C}_0 = (30)$, ya que solamente estamos consideramos a las capas densas. En lugar de la función de costo de entropía cruzada categórica se implementó la función de costo de entropía cruzada categórica dispersa. Las evoluciones se efectuaron hasta 15 generaciones, salvo para $k = 1$, en donde se usaron 60 porque su costo computacional es menor. Los entrenamientos se efectuaron hasta las 50 épocas. Los resultados se resumen en la Tabla 4.4

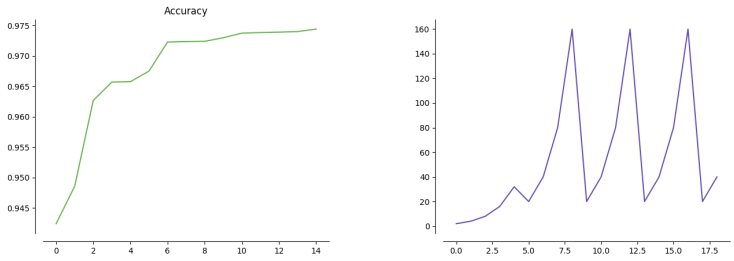
Cuadro 4.4: Resumen de los entrenamientos indicando la configuración del entrenamiento, la arquitectura óptima calculada, la validación obtenida y la exactitud sobre el conjunto de prueba, así como el tiempo de ejecución empleado en segundos.

Algoritmo	Arquitectura	Validación	Prueba	Tiempo
$A(1, 9, 4, 3)$	(24)	0.9506	0.9622	2233.99039006
$A(30, 9, 4, 3)$	$(150, 60)$	0.9732	0.9806	1661.98341298
$A(30, 20, 6, 3)$	$(150, 90)$	0.9738	0.9788	3051.702986
$B(70, 9, 4, 3)$	$(271, 131)$	0.9768	0.9801	3053.07773709
$B(70, 20, 6, 3)$	(362)	0.9767	0.9837	6803.21170092
$C(30, 3, 2, 1)$	$(30, 33)$	0.9837	0.9896	6742.15097094

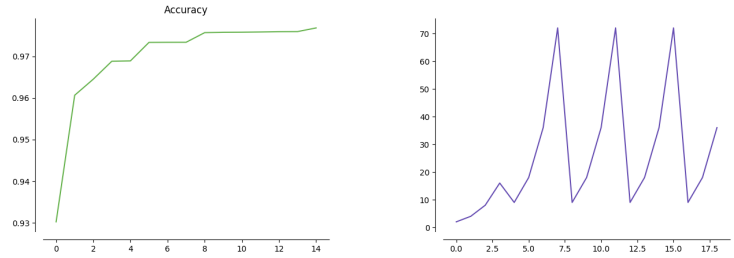
Como se esperaba, la arquitectura convolucional muestra mejores resultados comparativos, por lo que en términos prácticos resulta más conveniente utilizar la evolución en este sentido. También se observa que la introducción de aleatoriedad manifiesta una ligera mejoría con respecto a los experimentos realizados con k fija. Incrementar las otras cons-

tantes no siempre llevó a mejoras sustanciales notables, aunque sí un mayor consumo de recursos computacionales.

Figuras 4.5-4.6 muestran los resultados de la evolución de la exactitud, así como la dinámica poblacional simulada.



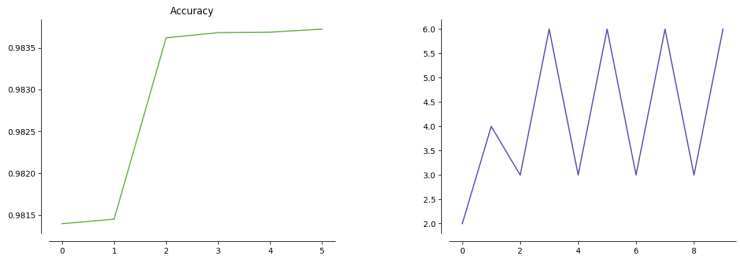
Cuadro 4.5: Mejora de la exactitud (eje y, izquierda) por generación (eje x, izquierda), así como número de individuos en $A(30, 20, 6, 3)$. La diferencia entre los números de individuos de ambas gráficas radica en que se introduce una generación de redución para la gráfica del número de individuos.



Cuadro 4.6: Mejora de la exactitud (eje y, izquierda) por generación (eje x, izquierda), así como número de individuos en $B(70, 9, 4, 3)$.

EMNIST

Los resultados dados con la base de datos MNIST nos dan atisbos de cómo podemos diseñar una apropiada evolución de hiperparámetros para abordar otros problemas de clasificación más recientes, como es el caso de EMNIST. Se ha observado que para fines prácticos, la evolución debe darse por medio de una red convolucional base y las mutaciones deben tener una k aleatoria en lugar de una fija.



Cuadro 4.7: Mejora de la exactitud (eje y, izquierda) por generación (eje x, izquierda), así como número de individuos en $C(30, 3, 2, 1)$.

Siguiendo los resultados de evolución de capas densas en CNNs, se empleó una esquema de evolución de $C(30, 3, 2, 1)$ y la misma arquitectura. La estructura de las capas densas original era (30), con una exactitud de 0,9209 sobre el conjunto de prueba. El proceso de evolución obtuvo como resultado a la red (104), alcanzando una exactitud sobre el conjunto de prueba de 0,9263.

Se propuso también una segunda arquitectura convolucional con un mayor número de parámetros, basada en las VGGs [228], la cual cuenta con la siguiente arquitectura:

1. 2 capas convolucionales con 64 filtros de 3×3
2. MaxPooling de 2×2
3. 2 capas convolucionales con 128 filtros de 3×3
4. Capa densa de 256 neuronas
5. Capa de clasificación de 26 neuronas

Un entrenamiento con diez épocas arrojó una exactitud de 0,9379. Utilizando un esquema de evolución $C(70, 2, 2, 1)$ se logró una exactitud de 0,9399 sobre el conjunto de prueba obteniendo una arquitectura densa de $(361, 85)$.

Una tercera red convolucional fue considerada, dada por la arquitectura siguiente:

1. 64 filtros de 3×3 con el mismo padding
2. Batch Normalization

3. 64 filtros de 3×3 con el mismo padding
4. Batch Normalization
5. Max Pooling de 2×2
6. Dropout de 0,25 %
7. 128 filtros de 3×3 con el mismo padding
8. Batch Normalization
9. 128 filtros de 3×3 con el mismo padding
10. Batch Normalization
11. Max Pooling de 2×2 con strides de 2×2
12. Dropout de 0,25 %
13. 256 filtros de 3×3 con el mismo padding
14. Batch Normalization
15. Dropout de 0,25 %
16. Capa completamente conectada con 512 neuronas.
17. Capa de clasificación con 10 neuronas

El entrenamiento se efectuó utilizando a la función de Entropía Cruzada Categórica dispersa y el optimizador Adam. Primero se entrenaron las capas convolucionales a 10 épocas y después se entrenó la capa de clasificación truncando la original utilizando la técnica de *Early Stopping* (véase [3]), logrando una exactitud clasificación de 95,23 % sobre el conjunto de prueba, 97,29 % en el conjunto de entrenamiento y 97,13 % en la validación. Para la evolución, se procedió a extraer las características primero para lograr un desempeño más eficiente. El esquema utilizado fue $C(120, 6, 4, 2)$ utilizando 10 generaciones. No obstante, no fue posible encontrar una configuración posterior (añadiendo capas) que mejore

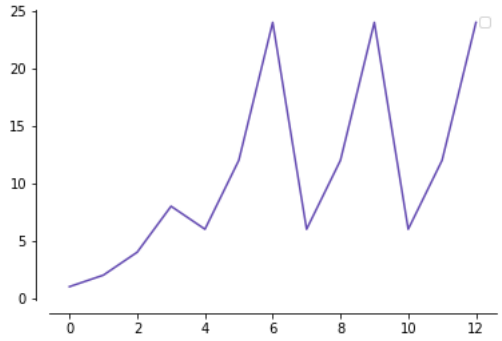


Figura 4.3: Dinámica poblacional de la evolución con una configuración $C(120, 6, 4, 2)$ usando la tercera arquitectura convolucional.

la exactitud sobre el conjunto de validación, lo cual nos da atisbos para creer que la adición de capas densas no es necesaria para esta red. La dinámica poblacional de la evolución aparece en la figura 4.3. Nótese que la red tiene el código genético (512) y fue variada para observar las combinaciones $(512, r_1, r_2, \dots)$. Sin embargo, no debe compararse con la red anterior, puesto que las capas no densas son diferentes.

4.4. Conclusiones y trabajo futuro

En el contenido de este capítulo hemos abordado la segunda rama de los métodos de optimización de redes neuronales correspondiendo al segundo nivel que es el hiperparamétrico, concretamente para el caso de los números de capas densas con sus respectivos números de neuronas. Se introduce un método de optimización que emula la evolución de una población presa mediada o controlada por un depredador, que para este contexto carece de significado. Aún más, debido a la abstracción que se realiza de las ecuaciones de Lotka-Volterra, que definen la dinámica poblacional de ambas especies, se puede considerar a la población objetivo como depredador o presa.

El algoritmo que se presenta muestra algunas simulitudes con los algoritmos genéticos base, aunque el método de reproducción es exclusivamente de tipo asexual. Se apreció que el soporte convolucional presenta mejores ventajas para efectuar la evolución, aunque este dato es evidente con base a la literatura documentada de la comparación entre redes

simples y convolucionales. Una observación empírica que resalta por ser menos obvia es que el uso de k aleatoria permite obtener mejores resultados, aunque esto también es anticipado por la literatura puesto que el uso de k fija es una estrategia de búsqueda similar a la búsqueda por rejilla, mientras que la k estocástica corresponde a la búsqueda aleatoria.

En general fue posible mejorar la exactitud de las redes neuronales sobre el conjunto de prueba, aunque fue posible encontrar casos donde no se encontraron mejoras significativas. Esto puede deberse a que la arquitectura considerada presenta ya el número adecuado de capas convolucionales y densas, sin necesitar de optimizarse, o dicho en otros términos, se inició en un máximo local.

Desde el punto de vista teórico, el algoritmo es interesante porque para valores particulares converge a un máximo local de forma polinomial, sacrificando la convergencia global con la adición de depredadores pero a cambio de sustituir la convergencia NP en P. ¿De qué manera podría generalizarse esta idea para otros tipos de algoritmos NP que puedan sacrificar un máximo global para volverse P?

Finalmente discutiré brevemente los puntos de artificialidad que presenta el algoritmo. Con respecto a los algoritmos genéticos, quizá la artificialidad más fuerte es el tipo de reproducción que maneja. Si bien existen especies de animales con sistema nervioso de reproducción asexual como el gusano *Enchytraeus japonensis* [262], parecen ser menos los animales con visión (los problemas que se abordaron son propiamente de reconocimiento de imágenes) de alta resolución con reproducción de tipo asexual (básicamente verebrados, moluscos y artrópodos, véase el capítulo de Redes Neuronales Convolucionales, animales de reproducción generalmente sexual) y menos por fisión binaria, que es característico de seres organizacionalmente más simples como las bacterias. De igual manera, la evolución misma parece estar ligada a la visión y las mismas redes neuronales pueden jugar un papel para la selección de parejas potenciales. Eso nos muestra que la implementación de redes neuronales capaces de estimar la aptitud de parejas puede ser una línea de investigación provechosa.

Un problema adicional que se presenta, al menos en apariencia, es que la evolución asume necesariamente la presencia de una interacción de tipo depredador-presa, cuando

en los siglos recientes la dinámica evolutiva humana no parece seguir ese esquema. No obstante, las tareas presentadas en este capítulo aún son sencillas en comparación de lo que puede realizar la visión humana, incluso en siglos anteriores. Más interesantemente, la evolución de la visión pudo darse en tiempos de interacción depredador-presa. Lynne Isbell lanzó la hipótesis de que la evolución de la visión especializada en el reconocimiento de objetos en primates estuvo en gran medida influenciada por el contacto con serpientes venenosas, la cual fue mayor en monos del Viejo Mundo, de los cuales se desprende el género *Homo* [115]. De esta forma, sugerir un algoritmo evolutivo basado en modelos de depredador-presa no es inadecuada.

Otros modelos poblacionales pueden ser considerados en futuros avances, como por ejemplo, el modelo de competencia ecológica por medio de las ecuaciones de Volterra-Bigon (ver [130]), el cual abstrae la competencia entre especies que comparten un mismo nicho ecológico, lo cual puede ser útil para el caso de algoritmos en competencia. Una segunda vía es considerar mutaciones en las capas convolucionales, las cuales tienen una influencia posiblemente más decisiva que para las capas densas.

Capítulo 5

La Regla de Hebb

Cells that fire together wire together

Aforismo de la Regla de Hebb [72].

La mayoría de los anteriores construcciones sobre Redes Neuronales (especialmente FNN) utilizan métodos de optimización basados en el gradiente (desde el Descenso de Gradiente Clásico hasta el optimizador Adam) que sirven para minimizar una función de costo previamente definida. Hemos estudiado en los capítulos anteriores sobre la importancia de disponer una arquitectura adecuada y sobre la relación que tienen con los ejemplos biológicos. En este capítulo discutiremos qué ocurre en las redes neuronales reales y la existencia de algún algoritmo de optimización. Si bien la naturaleza optimiza, ¿qué es lo que optimiza y cómo lo logra?

En el contexto de Machine Learning, optimización mantiene una especie de sinonimia con el aprendizaje, o más precisamente, el aprendizaje es una forma de optimización. De esta forma, las preguntas anteriores se reducen a indagar cómo ocurre el aprendizaje a nivel neuronal, interrogante que inquietó a una serie de psicólogos, neurocientíficos y por último, científicos computacionales y matemáticos.

Numerosos planteamientos clásicos de Redes Neuronales están basados en resolver problemas de Aprendizaje Supervisado que involucran una base de datos finita, la cual se entrena época tras época hasta lograr un rendimiento aceptable sobre un conjunto de validación, para finalmente medir el error con un conjunto de prueba. Para fines prácticos,

este procedimiento es adecuado para verificar si el algoritmo utilizado funciona, pero en las condiciones naturales el aprendizaje no se da por medio de grandes bases de datos sino de la captura de información en tiempo real. Así, en lugar de disponer grandes bases de datos, los organismos adquieren los datos mediante la recepción directa por medio de sus órganos sensoriales, muchas veces implicando en la saturación de información para lo cual necesitan desarrollar mecanismos de atención.

Siguiendo este esquema, en lugar de disponer una base de datos $\{(x_i, y_i)\}_{i=1}^n$ se tiene una sucesión de datos $\{(x_i, y_i)\}_{i\in\mathbb{N}}$ que son directamente recibidos por los sensores, analogía factible con los órganos de los animales que disponen de sistema nervioso. De esta forma, las aplicaciones directas de este capítulo estarán centradas en la robótica y sistemas autónomos que reciben información en tiempo real y que pueden mejorar su exactitud de acuerdo con los datos de sus respectivos entornos, en lugar de únicamente disponer de arquitecturas neurales entrenadas pero que carezcan de plasticidad con los nuevos ejemplos.

Asimismo, algunos métodos basados en el gradiente pueden resultar computacionalmente costosos por el cálculo de derivadas pero principalmente por la necesidad de recorrer numerosos ejemplos. Como se mencionó anteriormente, Descenso de Gradiente clásico está dado por la regla recursiva

$$\theta \leftarrow \theta - \alpha \nabla J_\theta(x, y), \tag{5.1}$$

donde $J(x)$ es la función de pérdida, las cuales tienen la forma

$$J_\theta(x, y) = \sum_{i=1}^{n} j_\theta(x_i, y_i), \tag{5.2}$$

donde $j_\theta(x_i, y_i)$ es generalmente una métrica de error entre x_i y y_i. De las ecuaciones 5.1 y 5.2 obtenemos que

$$\theta \leftarrow \theta - \alpha \sum_{i=1}^{n} \nabla j_\theta(x_i, y_i). \tag{5.3}$$

Este paso es utilizado en cualquiera de los métodos basados en el gradiente completo. Evidentemente, se tiene un problema para una sucesión de datos. Es posible intentar

aproximar la serie infinita para el caso de $\sum_{i=1}^{\infty} \nabla j(x_i, y_i)$ o simplemente considerar una muestra de los n_0 últimos datos.

Al respecto, existen estrategias que nos permiten reducir la rigidez del planteamiento original del clásico Descenso de Gradiente. Por un lado, podemos tratar de reestructurar el algoritmo a una forma propia para el entrenamiento en tiempo real, o bien replantear el algoritmo de optimización.

Otro problema de los métodos basados en el gradiente es, en cierto modo, su artificialidad. Este punto debe marcarse con cierto cuidado, puesto que no es claro si las neuronas reales siguen un método diferente que en la práctica se asemeje a alguna forma del Descenso de Gradiente. Sin embargo, conviene volver a preguntarse qué mecanismos subyacen en el aprendizaje de las neuronas y cómo se puede implementar en términos computacionales.

Durante este capítulo abordaremos la modelación de algunas formas de aprendizaje neuronal, particularmente el aprendizaje asociativo y la plasticidad. Un teórico de gran relevancia para el estudio de estos temas es el psicólogo canadiense Donald Hebb, cuya obra tuvo importantes consecuencias en el desarrollo de la Neurociencia Computacional. Finalmente revisaremos otros modelos que abordan el aprendizaje neuronal con el objetivo de tratar de resolver satisfactoriamente la pregunta cómo aprenden las neuronas biológicas y disponer con ello de un algoritmo de aprendizaje que nos permita lograr niveles satisfactorios de precisión en contextos de tiempo real. En ocasiones, hablaremos de Reglas de Hebb en plural para referirnos a los algoritmos que parten de la formulación original de Hebb y de los experimentos que verificaron la existencia de un proceso de aprendizaje similar en las neuronas reales. Un término quizá más preciso es el de Reglas de Plasticidad o modelos computacionales de la plasticidad neuronal, que es la capacidad de las neuronas de adaptarse a los datos presentados. Pero antes de hablar de modelos de la plasticidad debemos introducir algunos aspectos relacionados con la biología del aprendizaje.

5.1. Online Machine Learning

De acuerdo con Li, Liu y Dong [151], un algoritmo de aprendizaje es *online* si

1. Todos los datos de entrenamiento son presentados secuencialmente al modelo.

2. En cada iteración, solamente un dato es procesado.

3. Cada dato es descartado cuando la iteración está completa.

4. El modelo ignora cuántos datos serán presentados.

En otras palabras un algoritmo de aprendizaje en línea es aquel que solamente procesa un dato dado sin tomar en cuenta los datos anteriores o posteriores. El conjunto de entrenamiento puede ser finito, y en la práctica para evaluar un algoritmo *online* lo es, sin embargo resulta conveniente representarlo como una sucesión de datos (en forma vectorial o tensorial), puesto que n es desconocida.

Las redes neuronales convencionales abordadas en el capítulo 2 no manifiestan ese comportamiento puesto que procesan información por épocas y como se ha discutido, el cálculo del gradiente requiere considerar a todos los n datos de entrenamiento. Una alternativa conveniente sería reducir el batch a un solo dato y aplicar una sola época. El resultado de aplicar esta técnica sencilla se abordará posteriormente en el capítulo 5, cuando comparemos a los enfoques de entrenamiento de redes neuronales en cuestión (gradiente y hebbiano).

Adicional a lo anterior, se han formulado diferentes modelos que buscan permitir un desempeño *online* del Descenso del Gradiente, entre los cuales destacan [84, 272] a nivel teórico, el Perceptrón de Rosenblatt [214] para redes neuronales, del cual se hará una revisión más exhaustiva, y más recientes formulaciones como [217] extienden la formulación de la retropropagación para redes profundas.

Es posible dilucidar algunas ventajas (o la necesidad, en algunos casos) de aplicar *online learning*, a pesar de sacrificar exactitud. Como se ha comentado, está destinado al procesamiento de datos posiblemente masivos (*big data*) o que de forma natural se presentan secuencialmente como en el caso de los sensores. Tal como se ha esbozado, los seres vivos y por extensión los seres autónomos, procesan información secuencial proveniente de los sentidos o sensores, por lo que no resulta difícil entender por qué los algoritmos de aprendizaje *naturales* operan de este modo y la Regla de Hebb no es una excepción.

5.2. Bases empíricas de la Regla de Hebb

5.2.1. Formulación psicológica de la Regla de Hebb

Desde finales del siglo XIX se sospechaba que la estructa cerebral puede modificarse creando asociaciones. En 1888 Freud propuso la *ley de asociación por simultaneidad* que condensa los principios desarrollados posteriormente. Fue en 1949 cuando Donald Hebb sugirió que dadas dos neuronas A y B excitadas repetidamente se fortalecía la conexión entre ambas ([245, 30]). En su obra principal, *The Organization of the Behavior*, Hebb redactó su *Principio Neurofisiológico*:

> *When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.*
>
> Donald Hebb, 1949: 42 ([89])

De acuerdo con [236], Hebb sintetizó los trabajos en psicología de autores como Pavlov, de neuroanatomía de Ramón y Cajal y neuropsicología de Sherrington y Lorente de Nó. Sin embargo, la verificación experimental de la Regla de Hebb requirió esperar los avances en Electrofisiología hacia finales de la década de 1960.

5.2.2. Biología del aprendizaje asociativo y Lóbulo Medial Temporal

La Regla de Hebb fue formulada originalmente como una conjetura o principio hipotético de cómo se da el aprendizaje a nivel neuronal. El descubrimiento de la Potenciación a Largo Plazo (*Long-term Potentiation* o LTP) por medio del trabajo de Terje Lømo en 1966 [160] y el trabajo de Timothy Bliss y el mismo Lømo [24] son considerados como evidencias experimentales sólidas de la formulación original de Hebb [23, 185]. Aunque originalmente descubierto en el Hipocampo, se ha encontrado evidencia de LTP en Neocórtex, Amígdala, Ganglios Basales y en las células Purkinje del Cerebelo, áreas asociadas al aprendizaje

[28, 51, 227, 72], así como también en la Corteza Visual Primaria de primates *Macaca mulatta* [107]. Haremos una revisión preliminar de algunos aspectos relevantes de la biología de algunas estructuras relevantes para la formación y consolidación de la memoria como el Hipocampo y finalmente abordaremos los conceptos relevantes de esta sección, que son la Potenciación y Depresión a Largo Plazo.

El Hipocampo

Observaciones tempranas sobre lesiones en estructuras del Lóbulo Medial Temporal, incluyendo el Hipocampo, han contribuido a reforzar la idea de que el Hipocampo y otras estructuras adyacentes juegan un papel relevante en la formación y consolidación de la memoria a largo plazo, en especial de tipo declarativo [72, 185] y episódico [255]. A pesar de ser una estructura relativamente pequeña, el Hipocampo cuenta con funciones variadas relevantes tanto en la memoria como en la localización. Asimismo, el Hipocampo posee propiedades que las otras regiones generales carecen, ya que se trata de una de las dos regiones donde se produce neurogénesis adulta [93], específicamente en el Giro Dentado, tal como se verificó en 1998 en el artículo de [66]. Una pregunta sumamente interesante es sobre qué papel computacional adquiere la generación de nuevas neuronas exclusivamente en el Hipocampo, propiedad que carece por ejemplo, la Corteza Visual[1].

Anatómicamente, el Hipocampo está formado por el Giro Dentado y el Cornu Ammonis (CA) dividido en CA1, CA2, CA3 y CA4 [36]. Se han estudiado a detalle cada una de estas estructuras. Por ejemplo, estudios de lesiones en la región CA1 humana [18] muestran que esta región es importante para la memoria autobiográfica así como el viaje mental al pasado. La memoria autobiográfica episódica también es afectada si se lesiona la región CA3 [179], la cual también ha sido señalada como importante para la memoria espacial [85]. La investigación realizada por [54] muestra que tanto la CA1 como CA3 cuentan con

[1]En general los modelos de redes neuronales, incluyendo las convolucionales, no consideran la adición de neuronas durante el entrenamiento, sino que se fijan al principio. Esto está en correspondencia con lo que se conoce de la neurogénesis adulta si se trata del procesamiento de información visual o auditiva. En capítulos siguientes vemos que este número de neuronas sí puede variar después de realizarse un entrenamiento, pero esto está mayormente relacionado con la evolución de los sistemas nerviosos, los cuales varían enormemente en el número de neuronas en cada especie

funciones complementarias para la memoria episódica. La región CA2, menor estudiada, ha sido señalada como relevante para la memoria social [95, 248]. Fuera del Hipocampo, la Corteza Prefrontal también ocupa un rol relevante en la memoria utilizando un mecanismo de procesamiento donde la información avanza de manera bidireccional al recibir y emitir señales al Hipocampo, siendo de igual forma elementos complementarios en la formación de memoria [58].

Células de lugar y de tiempo

Los estudios en la actividad individual de células de roedores han llevado al descubrimiento de células altamente especializadas en el Hipocampo. En 1971, John O'Keefe (Premio Nobel de Fisiología 2014) y Jonathan Dostrovsky [191] encontraron neuronas de la CA1 dorsal selectivas a la posición del animal. A estas neuronas se les denominó como células de lugar (*place cells*).

En el 2005 Hafting y sus colaboradores descubrieron las células de rejilla (*grid cells*) en la Corteza Entorrinal Medial Dorsocaudal [80] las cuales se activan cuando el sujeto se encuentra en algún vértice de polígonos regulares no visibles que cubren la superficie. Debido a que la Corteza Entorrinal se conecta al Hipocampo por medio del Giro Dentado. Junto con las células de dirección de cabeza (*head-direction cells*) en el Postsubículo [242] así como células de borde (*border cells*), que son selectivas a los bordes de un entorno cerrado como una habitación [231], nos aportan ideas claves sobre cómo el sistema Corteza Entorrinal-Hipocampo codifica el espacio. Por ende, modelos de cómo se jerarquizan las células de rejilla con las células de lugar han sido propuestos [218].

Entre los descubrimientos recientes (la década pasada) en la materia destacan las células de lugar sociales en murciélagos y las células de tiempo en CA1, así como las células de velocidad de la Corteza Entorrinal Medial. Las células de lugar sociales (*social place cells*) fueron reportadas en el 2018 [192] en murciélagos *Rousettus aegyptiacus* y codifican la posición de otro murciélago. En cuanto a las células de tiempo (*time cells*) han sido descritas en artículos como [165], destacando la existencia de células selectivas a una temporalidad relativa específica, aunque interpretaciones alternativas de estos resultados existen [131]. Las células de velocidad (*Speed cells*) fueron encontradas en la Corteza

Entorrinal Medial (donde también aparecen las células de rejilla) mostraron una relación lineal con respecto a la velocidad de las ratas observadas [134]. Estos resultados muestran que tanto la Corteza Entorrinal como el Hipocampo codifican características relevantes del sujeto en cuestión para la formación de recuerdos desde la primera persona, incluyendo velocidad, tiempo y posición en el espacio. A continuación observaremos otras funciones integradoras que se presentan en estas estructuras junto como la Amígdala, que también forma parte del Sistema Límbico.

Células multimodales

En el capítulo de Redes Neuronales Convolucionales abordamos la existencia de células altamente especializadas en el Lóbulo Medial Temporal humano para el reconocimiento de conceptos específicos, invariante tanto a entradas visuales como imágenes o incluso texto, así como auditivas, presentado en la sucesión de artículos [132, 209, 207]. Detallaremos más los resultados obtenidos en dicho artículo. En en el Hipocampo Anterior Derecho fue encontrada una neurona que disparaba una alta tasa de disparo para imágenes de Oprah Winfrey, su nombre escrito y también para un audio mencionando su nombre. Una neurona de la Corteza Entorrinal hizo lo mismo para el concepto de Saddam Hussein. Unidades similares fueron encontradas en la Amígdala. Esta triple invarianza fue encontrada en el 35 %-40 % de las neuronas del Hipocampo y la Corteza Entorrinal, 14 % en la Amígdala, pero no en la Corteza Parahipocampal, cuyas neuronas fueron selectivas únicamente para entradas auditivas y escritas. Otro estudio realizado en el 2006 [237] en macacos *Macaca mulatta* encontró células de la Corteza Prefrontal Ventrolateral que respondían fuertemente tanto a entradas auditivas de un macaco vocalizando, como visuales (imágenes de macacos). Nuevamente, no todas las células presentaron selectividad multimodal.

Algunos de estos resultados han sido interpretados como favorables a la existencia de *Grandmother cells*, las cuales son neuronas hipotéticas que responden a un objeto físico en específico (por ejemplo, la abuela). Este concepto está relacionado con las células gnósticas (*Gnostic Cells*), las cuales son células que responden a todas las categorías de objetos [46]. Existen dos enfoques opuestos sobre cómo el cerebro codifica la información: el *paradigma localista*, que afirma que existen células únicas que responder a entes particu-

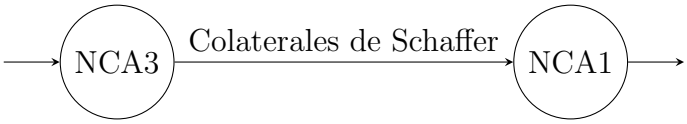
lares perceptibles; y el *paradigma asociativista*, que considera que la representación es más bien distribuida. Los mismos Quian Quiroga, Kreiman, Koch y Fried [208], que figuran como autores en las investigaciones mencionadas sobre las células multimodales del Lóbulo Medial Temporal, fuertes candidatas a ser *grandmother cells*, consideran que las células multimodales que estudiaron no se ajustan a la definición de *grandmother cell*, si bien comparten ciertas características. En su lugar, según los mencionados autores, debido al papel del sistema hipocampal-rinal en la formación de memoria, las células multimodales no parecen estar relacionadas con el reconocimiento y asociación de imágenes con audios (como es el caso de la Corteza Infratemporal, o posteriores, quizá la Corteza Prefrontal Ventrolateral) debido a la latencia que tienen en su respuesta, sino que es más plausible que estas células estén relacionadas con la formación de recuerdos sobre los objetos que evocan una mayor respuesta.

Potenciación a Largo Plazo

La revisión que realizaremos de los conceptos de *Long Term Potentiation* y *Long Term Depression* está basada en [206]. Con fines de simplicidad, no incluiremos detalles sobre la sinapsis química, que es el tipo de sinapsis que tiene mayor relación con estos fenómenos, puesto que el enfoque que manejaremos está centrado en la tasa de disparo. Tampoco revisaremos la influencia de aspectos moleculares como el receptor N-metil-D-aspartato (NMDA) en la Potenciación a Largo Plazo, sino que nos centraremos en los aspectos funcionales. El potencial postsináptico (*Post-Synaptic Potential* o PSP) está relacionado con la probabilidad de producir un potencial de acción (disparo o *spike*) en la célula postsináptica. Un potencial postsináptico es excitatorio si incrementa dicha probabilidad de ocurrencia de un *spike* en la célula postsináptica e inhibitorio si la decrementa. Al potencial postsináptico excitatorio usualmente se le denota como EPSP (*Excitatory Postsynaptic Potencial*), mientras que al inhibitorio como IPSP (*Inhibitory Postsynaptic Potential*). Una propiedad importante de los potenciales postsinápticos es que pueden sumarse tanto temporalmente como espacialmente, pudiendo alcanzar el umbral de disparo y producirse un potencial de acción.

Los experimentos realizados por Bliss y Lømo de 1973 [24] se dirigieron hacia las

neuronas del hipocampo, el cual cuenta con las áreas CA1 y CA3 compuestas por células piramidales, así como células granules en el Giro Dentado, que reciben conexiones de la Corteza Entorrinal. Las células granulares se conectan con las piramidales de CA3 por medio de las fibras musgosas, las cuales se conectan con las piramidales de CA1 mediante los Colaterales de Schaffer. Si se estimulan con baja frecuencia los Colaterales de Schaffer, no se produce un cambio en los potenciales postsinápticos relevantes. Sin embargo, si se produce una alta frecuencia, entonces es posible observar un aumento significativo de la amplitud del potencial postsináptico excitatorio (EPSP) de la membrana de la célula de CA1 (medido en milivoltios), que permanece por varias horas. Esto se muestra en el diagrama inferior, donde vemos una neurona de CA3 (NCA3), la cual realiza sinapsis con la neurona de CA1, cuyo axón pasa por los Colaterales de Schaffer. A este proceso se le denominó como Potenciación a Largo Plazo o *Long Term Potentiation* (LTP)



Una importante propiedad de la LTP aparece especificada en el artículo de Gustafsson *et al* en 1987 [75], denominada como *dependencia de estado*, que se refiere a que la ocurrencia o no de LTP depende del estado del potencial de membrana de la célula postsináptica. De este modo, LTP se produce si la neurona de la CA1 (postsináptica) se depolariza (lo cual genera un potencial de acción) al mismo tiempo que se estimulan los axones de la neurona de CA3. De este modo lo que produce LTP es la actividad simultánea tanto de la célula presináptica como de la célula postsináptica.

Otra propiedad es la de *especificidad de entrada*, la cual indica que si una neurona presenta actividad presináptica, pero otra no, entonces solamente se produce LTP en la activa. Sin embargo, una estimulación débil (bajo *firing rate*) que normalmente no conduciría a LTP en una neurona presináptica, produce a LTP si otra neurona presináptica vecina recibe una fuerte estimulación.

Depresión a Largo Plazo

Un fenómeno no originalmente propuesto por Hebb pero encontrado al estudiar la Potenciación a Largo Plazo es la Depresión a Largo Plazo, la cual funciona como una operación inversa a la LTP. Otra propiedad relevante sobre la LTP es la existencia de una cota superior en el incremento del EPSP. Sin la presencia de un decremento, existiría una saturación en la cota superior que impediría nuevos aprendizajes. La Depresión a Largo Plazo o *Long-Term Depression* (LTD) fue encontrada en el Hipocampo al igual que la LTP. Sin embargo, su efecto es el opuesto a LTP. La Depresión a Largo Plazo se produce al estimular a muy baja frecuencia los Colaterales de Schaffer (axones de las neuronas de CA3) produciendo un decremento en la amplitud del EPSP. De manera interesante esto puede borrar los efectos de una Potenciación a Largo Plazo, a manera de operación inversa.

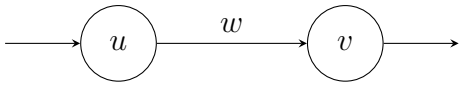
5.2.3. Modelación Matemática de la Regla de Hebb

Las propiedades de los efectos de LTP satisficieron en gran medida el Postulado Neurofisiológico de Donald Hebb. Tanto el proceso de LTP como el de LTD son entendidos como mecanismos claves en el aprendizaje y la memoria. Desde el punto de vista de la tasa de disparo, la cual es medida en Hertz, la LTP se produce si presentamos una tasa de disparo presináptica alta (por ejemplo, 100 Hz durante un segundo) y la LTD si la tasa de disparo es baja durante un tiempo prolongado, por ejemplo, 1 Hz durante 15 minutos. Los efectos de ambos pueden durar horas e incluso días [45].

Básicamente, la denominada Regla de Hebb puede plantearse matemáticamente de la siguiente manera: Sean N_v y N_u dos neuronas. El valor de su actividad (respectivamente v y u, véase la imagen) es la tasa de disparo medido en Hertz y tomado en los reales no negativos[2]. La fuerza o peso de la conexión w, está relacionada con la amplitud relativa de EPSP y tomará valores continuos definida en los reales tal como sugieren Dayan y Abbott, al igual que deben tener una cota superior ([51]). Los valores negativos representan a la inhibición. La regla de Hebb indica que si $v(t) > 0$ y $u(t) > 0$ son valores altos entonces

[2]Resulta más conveniente este modelo que utilizar simplemente los valores de encendido y apagado

w debe aumentar.



Es importante puntualizar que la inhibición y excitación están pobremente caracterizadas en las redes neuronales artificiales, de donde partimos para incluir a la regla de Hebb. Mientras que la inhibición la interpretamos como un peso negativo y la excitación como uno positivo, en realidad se aceptan postulados como la Ley de Dale, que afirma que una neurona no puede excitar a un conjunto de neuronas postsinápticas e inhibir a otro [51]. Los pesos libres permiten que un proceso diferente a la Ley de Dale se cumpla. Sin embargo temporalmente no consideraremos estos detalles, debido a que necesitaríamos resolver cómo se organizan las neuronas inhibitorias y las excitatorias.

Para ejemplificar este proceso, considere que u representa la actividad de una neurona que se enciende si identifica un patrón fijo, por ejemplo, la letra A. Supongamos que v se enciente cuando se identifica el sonido de A. Entonces el valor de w aumentará, de forma que, de realizarse de manera reiterada, el encendido de u por sí mismo podría encender a v, aún si la entrada auditiva no estuviese encendida. Este ejemplo se usará como motivación para el posterior desarrollo teórico.

En general, las Reglas Hebbianas satisfacen la ecuación

$$\frac{d\mathbf{w}}{dt} = H(\mathbf{u}, v, h, \mathbf{w}), \tag{5.4}$$

donde u es la actividad de las neuronas presinápticas, v es la actividad de la neurona postsináptica y h representa un "tercer factor" asociado con el efecto de neuromoduladores como la dopamina o la acetilcolina (como en el modelo de [98]), neurotrasmisores, factores gliales y señales axonales retrógradas [139].

La formulación de reglas basadas en la ecuación previa directamente nos indica que el aprendizaje en este enfoque siempre es de tipo *online*, ya que la variación de los pesos está en función del peso mismo

La formulación de la ecuación previa de las Reglas de Hebb sin considerar el tercer factor h nos garantiza que el aprendizaje de este enfoque es necesariamente *online* pues la

variación de los pesos en el tiempo t está en función únicamente de los datos de entrenamiento (y de los pesos mismos) en el mismo tiempo, por lo que los datos aparecen secuecialmente representados con las actividades presinápticas (datos $\mathbf{u}(t)$) y la postsináptica (la etiqueta $v(t)$), procesando únicamente los datos del tiempo t, que se descartan con el tiempo. Nótese que en dicha formulación, el tiempo tiende a infinito, por lo que desde un punto de vista teórico los datos son presentados *ad aeternum* y en la práctica hasta detener el programa. Como esta regla está considerada para sistemas autónomos como robots, la ejecución de la regla finaliza con el dispositivo.

La consideración descrita anteriormente muestra que las reglas de Hebb sin el tercer factor satisfacen la definición de *Online Machine Learning* presentada. El tercer factor también podría cumplir estas condiciones, pero por el momento, no consideraremos al tercer factor h, ya que su formulación es relavitamente posterior a las reglas originales. Se presentarán las Reglas de Hebb Simple, Oja, Covarianza y BCM, las cuales son modelos de la plasticidad neuronal. La Regla del Perceptrón de Rosenblatt, a pesar de no ser en el sentido estricto una regla de Hebb sino una regla basada en el gradiente, puede ser considerada como tal si denotamos a las reglas de Hebb únicamente como mapeos específicos. Esto motiva a la siguiente definición

Definición 6. *Sean $w(t), u(t), v(t) : \mathbb{R} \to \mathbb{R}$. Una regla de Hebb es una función $H : \mathbb{R}^3 \to \mathbb{R}$ tal que*

$$\frac{dw}{dt} = H(u(t), v(t), w(t)). \tag{5.5}$$

Si dado $u(t^*), v(t^*) > 0$ entonces $\frac{dw}{dt}(t^*) > 0$, la regla satisface la condición de Hebb fuerte.

Si dado $u(t^*), v(t^*) > 0$ entonces $\frac{dw}{dt}(t^*) \geq 0$, la regla satisface la condición de Hebb débil.

Estas definiciones pueden extrapolarse al caso discreto, el cual es más práctico puesto que las implementaciones computacionales se harán de este modo. Utilizando el método de Euler y considerando a las funciones $w(t), u(t), v(t)$ como sucesiones en $\mathbb{R}$, podemos elegir w^0 inicial y establecer la siguiente regla recursiva

$$w^{t+1} = w^t + \alpha H(u^t, v^t, w^t). \tag{5.6}$$

La Condición de Hebb Fuerte, por ejemplo, está dada en términos de $w^{t+1} > w^t$ si $u^t > 0$ y $v^t > 0$, mientras que la débil es análoga ($w^{t+1} \geq w^{t+1}$ en este caso).

5.3. Reglas de Hebb

5.3.1. Regla de Hebb Simple

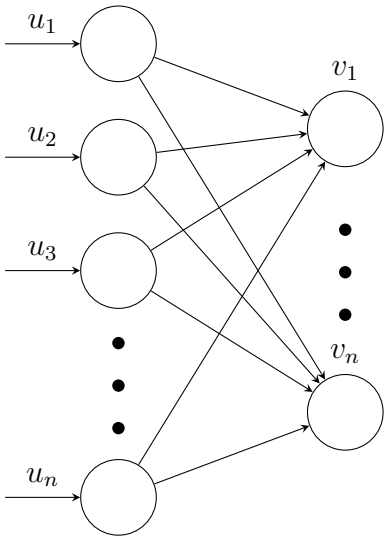
La forma más simple de implementar la Conjetura de Hebb es mediante la Regla de Hebb Simple. Consideremos m neuronas presinápticas con valor de tasa de disparo u_i cada una, conectada a una neurona con tasa de disparo v. Sea $\mathbf{u} = (u_1, \ldots, u_m)$ y $\mathbf{w} = (w_1, \ldots, w_m)$ el vector de pesos. Entonces, la Regla de Hebb Simple [51] está dada por la expresión

$$\tau_w \frac{d\mathbf{w}}{dt} = v\mathbf{u}, \tag{5.7}$$

donde τ_w es una constante de tiempo que controla la rapidez del cambio en los pesos. Mediante el Método de Euler podemos obtener una forma discreta para actualizar el cambio de los pesos mediante la fórmula recursiva

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \alpha v(n)\mathbf{u}(n). \tag{5.8}$$

De la misma forma podemos considerar el caso cuando la red esté completamente conectada, con $\mathbf{v} \in \mathbb{R}^{N_v \times 1}$ y $\mathbf{u} \in \mathbb{R}^{N_u \times 1}$, como aparece en el diagrama inferior, formando una matriz de pesos $\mathbf{W} \in \mathbb{R}^{N_u \times N_v}$



La forma matricial de la Regla de Hebb para este caso está dada por

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \alpha\ \mathbf{u}(n)\mathbf{v}(n)^T. \tag{5.9}$$

Intuitivamente, la Regla de Hebb Simple aumenta la fuerza de las entradas más repetidas, de forma que el peso de conexión entre dos neuronas es proporcional a la frecuencia de sus encendidos simultáneos.

Una posterior simplificación lineal de la Regla de Hebb Simple es considerar la función de activación lineal, de forma que $v = \mathbf{w} \cdot \mathbf{u}$. De este modo,

$$\begin{aligned} \tau_w \frac{d\mathbf{w}}{dt} &= \mathbf{w}(\cdot\mathbf{u})\mathbf{u} \\ &= \mathbf{Q}\mathbf{w}, \end{aligned}$$

donde $\mathbf{Q}$ es la matriz de correlación de los promedios de las entradas de forma que $\mathbf{Q}_{ab} = \langle u_a u_b \rangle$, representando el promedio de $u_a u_b$ en el tiempo (para convertirlo en constante). La matriz $\mathbf{Q}$ es precisamente la matriz de correlación.

5.3.2. Regla de Oja

Un problema que presenta la Regla de Hebb Simple en términos computacionales es el hecho de que los pesos no están acotados, lo cual contradice la evidencia experimental que

sugiere que $|w_i| \leq w_{max}$. Dos posibilidades para reducir este problema son la Normalización Subtractiva y la Multiplicativa, a la cual se le conoce mejor como Regla de Oja [189], la cual está dada por la siguiente expresión [51]

$$\tau_w \frac{d\mathbf{w}}{dt} = v\mathbf{u} - \beta v^2 \mathbf{w}, \tag{5.10}$$

donde $\beta > 0$ es la constante regularizadora. Una propiedad importante (probada en [51]) es la siguiente

Proposición 5. *La Regla de Oja es estable, es decir, $|\boldsymbol{w}|^2$ está superiormente acotado.*

Demostración. Notemos que

$$\begin{aligned}
\tau_w \frac{d|\mathbf{w}|^2}{dt} &= \tau_w \frac{d}{dt}\mathbf{w} \cdot \mathbf{w} && (5.11)\\
&= \tau_w (2\mathbf{w} \cdot \frac{d\mathbf{w}}{dt}) && (5.12)\\
&= 2\mathbf{w} \cdot \left(\tau_w \frac{d\mathbf{w}}{dt}\right) && (5.13)\\
&= 2\mathbf{w} \cdot (v\mathbf{u} - \beta v^2 \mathbf{w} \cdot \mathbf{w}) && (5.14)\\
&= 2(v(\mathbf{w} \cdot \mathbf{u})) && (5.15)\\
&= 2v^2(1 - \beta|\mathbf{w}|^2), && (5.16)
\end{aligned}$$

ya que $v = \mathbf{w} \cdot \mathbf{u}$. Los puntos estables están dados por $2v^2(1 - \beta|\mathbf{w}|^2) = 0$. Si $\mathbf{u} = \mathbf{0}$ entonces $\mathbf{w} = \mathbf{0}$. La otra posibilidad es que $|\mathbf{w}|^2 = \frac{1}{\alpha}$ y por lo tanto, en ambos casos, $|\mathbf{w}|^2$ es estable.

□

La forma discreta original de implementar a la Regla de Oja está dada por

$$\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha u(\mathbf{v}_i - \mathbf{w}_i). \tag{5.17}$$

Una importante propiedad observada por el mismo Erkki Oja[3] es que esta regla de aprendizaje tiende a extraer el componente principal de los datos de entrada dándole

[3]Varios de los autores mencionados en este capítulo, incluyendo a Kohenen (aún no mencionado), Oja como al mismo Lømo son finlandeses.

una interpretación de Análisis Multivariado a las reglas de Hebb. Para fines de esta tesis, la observación matemática más relevante es que la fuerza de conexión está dada por la probabilidad condicional de la actividad presináptica con respecto a la postsináptica [181], de forma que

$$\mathbf{w}_i = P(y|x_i). \tag{5.18}$$

5.3.3. Regla de la Covarianza

La Regla de Hebb Simple enunciada previamente solamente modela el proceso de LTP sin considerar la LTD. Una modificación simple de la Regla de Hebb fue propuesta en [223] como la *Regla de Covarianza*, la cual es formulada en [51] siguiendo la ecuación diferencial

$$\tau_w \frac{d\mathbf{w}}{dt} = (v - \theta_v)\mathbf{u}, \tag{5.19}$$

donde θ_v representa el umbral postsináptico. Si $v < \theta_v$, entonces la respuesta corresponderá a LTD, mientras que si $v > \theta_v$, se producirá LTP. Este umbral puede ser presináptico de tal forma que la Regla de Covarianza queda formulada como

$$\tau_w \frac{d\mathbf{w}}{dt} = v(\mathbf{u} - \boldsymbol{\theta}_u), \tag{5.20}$$

donde $\boldsymbol{\theta}_u$ es el vector umbral presináptico. Es posible incluir tanto un vector umbral presináptico como el umbral postsináptico, pero si $v = \mathbf{u}_i = 0$ para toda i entonces se producirá LTP, lo cual no es deseable.

El umbral θ debe satisfacer ser estrictamente mayor que 0 y menor que la actividad máxima para permitir la presencia de tanto LTP como LTD. Obsérvese que si $\theta = 0$ se reduce a la Regla de Hebb Simple.

5.3.4. Regla BCM

Los estudios realizados por Bienenstock, Cooper y Munro [22] derivaron a la formulación la llamada *Regla BCM*, la cual a diferencia de la Regla de la Covarianza emplea un

umbral dinámico, hecho que fue verificado experimentalmente [47, 143]. De esta forma, la Regla BCM se define a partir del siguiente sistema

$$\tau_w \frac{d\mathbf{w}}{dt} = v\mathbf{u}(v - \theta_v) \tag{5.21}$$

$$\tau_\theta \frac{d\theta_v}{dt} = v^2 - \theta_v, \tag{5.22}$$

donde el parámetro τ_θ regula la razón de cambio de θ_v, siendo más rápido que τ_v.

5.3.5. Perceptrón de Rosenblatt

La regla del Perceptrón de Rosenblatt [214] (un expresión moderna de esta regla aparece en [216]) consiste en una versión intermedia entre la Regla de Hebb y el Descenso de Gradiente, o más precisamente, es un Descenso de Gradiente simplificado a un batch de un elemento para permitir su comportamiento online. Utilizando una función de costo como la Suma de Errores Cuadrados, esta regla puede expresarse con la siguiente ecuación diferencial

$$\tau_{\mathbf{w}} \frac{d\mathbf{w}}{dt} = (y - \mathbf{w} \cdot \mathbf{x})\mathbf{x}. \tag{5.23}$$

De acuerdo con [167], esta regla es consistente con los resultados de los estudios realizados en la plasticidad de los cuerpos pedenculados de las moscas *Drosophila* presentado en [94], dando atisbos sobre cómo se minimiza una función de error conocida en el sistema nervioso.

5.3.6. Aprendizaje Anti-Hebbiano

La regla de Hebb Básica está enunciada para neuronas de tipo excitatorio, donde el incremento del valor del peso tiene sentido cuando ambas neuronas presentan una alta tasa de disparo. Sin embargo, si la neurona es inhibitoria, aumentar el valor del peso w significa disminuir su respuesta. Por lo tanto, para el caso de una neurona inhibitoria presináptica u y una postsináptica, la formulación de la Regla de Hebb básica debe considerar una reducción como la siguiente [38]

$$\frac{dw}{dt} = -\alpha uv. \tag{5.24}$$

5.3.7. Integración de entradas

Una operación adicional que es considerada en artículos como [98, 122] es aplicar una función logarítmica a los pesos, para evitar una saturación de los mismos dada por

$$S(w) = \begin{cases} w & w < 1 \\ \log(w) + 1 & w \geq 1 \end{cases}. \tag{5.25}$$

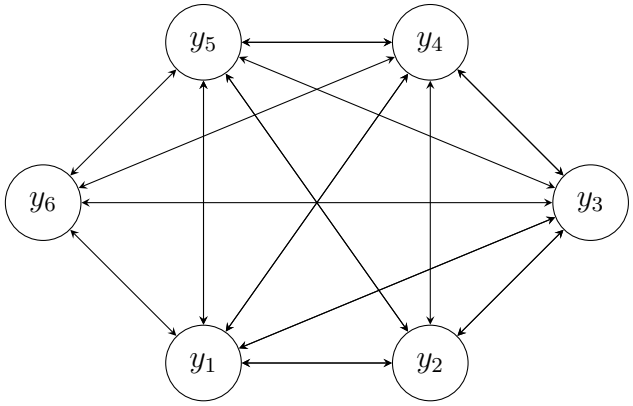
La salida estará dada, de este modo, por $y = \sum_{i=1}^{m} S(w_i)x_i$.

5.4. Aplicaciones de las Reglas de Hebb

Consideraremos dos principales aplicaciones de la Regla de Hebb en contextos de Inteligencia Artificial que han tomado una mayor influencia en este ámbito: las Redes de Hopfield y los Mapas Auto-Organizados de Kohonen. La primera está dada por un tipo de redes neuronales recurrentes mientras que el segundo es un método de aprendizaje no supervisado. Las explicaciones que ofrecemos están mayormente basadas en [10] salvo donde se indique lo contrario.

5.4.1. Redes de Hopfield

Una red de Hopfield [101] (una versión primitiva de la misma aparece en [155]) es un tipo de red recurrente de actualización hebbiana que modela el almacenamiento y recuperación de la información humana. La arquitectura de la red es una red recurrente cuyos nodos y_i se conectan con todos, eliminando autoconecciones (o bien, asignándoles 0), de la forma siguiente



El peso w_{ij} representa a la conexión entre las neuronas con pesos y_i y y_j, cuyos valores están restringidos en el conjunto $\{0, 1\}$. Para este tipo de red, los pesos son simétricos de forma que $w_{ij} = w_{ji}$. Entre las reglas de actualización, se incluyen a la Regla de Hebb Básica y a la Regla de Covarianza, tanto en su versión presináptica como postsináptica, dadas por (en respectivo orden):

$$w_{ij} = w_{ij} + y_i y_j, \tag{5.26}$$

$$w_{ij} = w_{ij} + y_i(2y_j - 1), \tag{5.27}$$

$$w_{ij} = w_{ij} + (2y_i - 1)y_j. \tag{5.28}$$

Asimismo, J.J. Hopfield añade una regla más dada por

$$w_{ij} = w_{ij} + (2y_i - 1)(2y_j - 1). \tag{5.29}$$

Este procedimiento de entrenamiento permite el almacenamiento de la información en los nodos cuyo número representa la dimensión de un vector a almacenar. Para la recuperación de la información, simplemente aplicamos las operaciones neuronales básicas (modelo simplificado de *firing rate*), esta vez especificado por la regla recursiva

$$\mathbf{y}(t) = \mathbf{F}_\theta(\mathbf{W}\mathbf{y}(t-1)). \tag{5.30}$$

donde $\mathbf{y}(0)$ representa un fragmento incompleto de la información y $\mathbf{F}_\theta$ representa a la función de activación de umbral θ multidimensional.

Esta actualización permite que la red recupere información almacenada previamente con un fragmento de la misma. Esto se logra minimizando una función de energía donde los atractores o mínimos locales corresponden a la información almacenada. La optimización se realiza no mediante las reglas de Hebb sino mediante la recursión dada por las operaciones neuronales al recuperar la información almacenada. Esto se condensa en la siguiente proposición:

Proposición 6. *Sea*

$$E = -\sum_{i}\sum_{j\neq i} \boldsymbol{w}_{ij}\boldsymbol{y}_i\boldsymbol{y}_j. \tag{5.31}$$

Sea $\boldsymbol{y}(0)$ *un fragmento inicial. Entonces* $\lim_{t\to\infty}\boldsymbol{y}(t)$ *converge para* $\theta = 0$ *para una actualización asincrónica de la Red de Hopfield.*

Demostración. Preliminarmente notemos que podemos aplicar la actualización dada en la ecuación 5.30 actualizando cada entrada asincrónicamente

$$\mathbf{y}_i(t) = F_\theta(\mathbf{W}_{i,*}\cdot\mathbf{y}_i(t-1)). \tag{5.32}$$

Entonces el cambio puntual de la función de energía ΔE representa cambiar solamente la modificación de una entrada $\mathbf{y}_i$, de modo que

$$\Delta E = -\Delta\mathbf{y}_i\sum_{j\neq i}\mathbf{w}_{ij}\mathbf{y}_j. \tag{5.33}$$

Si $\mathbf{y}_i(t) = 1$, entonces $\mathbf{w}\cdot\mathbf{y} = \sum_{i\neq j}\mathbf{w}_{ij}\mathbf{y}_j > 0$. Si $\mathbf{y}_i(t-1) = 0$ entonces $\Delta\mathbf{y}_i = 1$ y $\Delta E < 0$. Si $\mathbf{y}_i(t-1) = 0$ entonces $\Delta\mathbf{y}_i = 0$ y $\Delta E = 0$.

Si $\mathbf{y}_i(t) = 0$, entonces $\mathbf{w}\cdot\mathbf{y} = \sum_{i\neq j}\mathbf{w}_{ij}\mathbf{y}_j < 0$ ya que $\mathbf{w}_{ii} = 0$. Si $\mathbf{y}_i(t-1) = 1$ entonces $\Delta\mathbf{y}_i = -1$ y $\Delta E < 0$. Si $\mathbf{y}_i(t-1) = 1$ entonces $\Delta\mathbf{y}_i = 0$ y $\Delta E = 0$. Por lo tanto, en cualquier caso ΔE es negativa o cero, y por lo tanto decrece con el tiempo. □

La capacidad de almacenamiento de una red de Hopfield de n neuronas es muy baja, siendo de $\frac{n}{2\ln n}$, aunque Storkey propone una modificación para alcanzar un mayor almacenamiento ($\frac{n}{\sqrt{2\ln n}}$). Esto hace impráctica a la Red de Hopfield para el almacenamiento

computacional de la información. Sin embargo, como método de optimización de funciones de energía, se ha empleado redes de Hopfield para resolver problemas de optimización discreta como el clásico Problema del Viajero [102].

Algunos autores [10, 138] consideran que las Redes de Hopfield conforman un modelo de la región CA3 del Hipocampo. Esto se debe a que las neuronas de la CA3 muestran una conexión recurrente. Esto completa el mapa simple de las conexiones internas del Hipocampo, dada del Giro Dentado a CA3, CA3 consigo mismo y con CA1.

5.4.2. Mapas Auto-Organizados de Kohonen

El algoritmo de Kohonen [129] consiste en una red de dos capas, la primera consiste en la capa de entrada con valores $\mathbf{x}$, mientras que la segunda puede entenderse como las clases agrupadas, cuyos valores son $\mathbf{y}$. Como en los modelos trabajados, $\mathbf{y} = \mathbf{W}\mathbf{x}$ (véase [10]). La regla de actualización está basada en Hebb y dada por:

$$\mathbf{W}_{i_h}(t+1) = \frac{\mathbf{W}_{i_h}(t) + \alpha \mathbf{x}^\intercal}{||\mathbf{W}_{i_h}(t) + \alpha \mathbf{x}^\intercal||}, \tag{5.34}$$

donde i_h es el conjunto de los índices en vecindad con b dado por $y_b = \text{máx}_j\{y_j\}$. La idea central de este algoritmo es generar mapas de características organizando la información en representaciones próximas entre sí, por lo que es común representar a los mapas autoorganizados en dos dimensiones, lo cual modela la generación de mapas topográficos en el cerebro, los cuales se presentan especialmente en las cortezas sensoriales y motoras, pero posiblemente no en las cortezas de asociación [197].

5.5. Implementaciones propuestas

5.5.1. Aplicación directa de las Regla de Hebb

Hemos definido a las reglas de Hebb y destacado cómo se inspiran desde el punto de vista biológico. Para verificar experimentalmente su efectividad en problemas de clasificación reales, tomaremos como ejemplo a la base de datos MNIST. Para ello se emplea una

red sin capas intermedias con 28×28 entradas y una salida de 10 neuronas, clasificando cada dígito correspondiente.

El entrenamiento se realiza utilizando las siguientes reglas:

1. Regla de Hebb Simple (Hebb)
2. Regla de Hebb Simple con regulación logística (LHebb)
3. Regla de Covarianza (Cov)
4. Regla de Covarianza con regulación logística (LCov)
5. Regla de Oja con $\beta = 0{,}01$ (Oja)
6. Regla de Oja con regulación logística y $\beta = 0{,}01$ (LOja)
7. Regla BCM con $\tau = 0{,}5$ (BCM)
8. Regla BCM con regulación logística y $\tau = 0{,}5$ (LBCM)
9. Regla de Rosenblatt-Perceptrón (Perp)
10. Regla de Rosenblatt con regulación logística (LPerp)

Para contrastar tales reglas, utilizaremos Adam con una época, minimizando la función de costo de Entropía Cruzada Categórica Dispersa y función de activación sigmoide.

Resultados

La aplicación de la comparación anteriormente descrita representa un primer intento por implementar las Reglas de Hebb en un problema de clasificación fuerte y emprender una comparación con los métodos del estado de arte. Sin embargo, en general los resultados son ampliamente favorables a los métodos convencionalmente establecidos como se muestra en la Tabla 5.5.1.

Sin embargo, a pesar de los resultados en general adversos, una visualización de los pesos nos muestran las imágenes agrupadas en la tabla 5.2, lo cual nos da una ligera impresión de que la red de una capa presenta dificultades para agrupar datos tan variables, por lo que una capa no supervisada (como los Mapas de Kohenen) puede ser útil.

Regla	Exactitud (%)
Hebb	68.04
LHebb	71.52
Cov	68.04
LCov	71.51
Oja	68.04
LOja	71.51
BCM	68.00
LBCM	71.53
Perp	39.17
LPerp	79.93
Adam	91.32

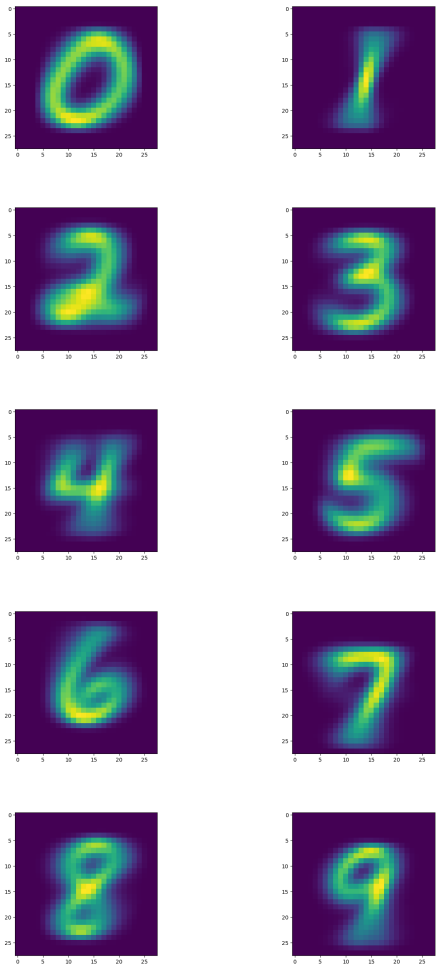
Cuadro 5.1: Exactitud obtenida utilizando diferentes reglas hebbianas y Adam

5.5.2. Redes HKH

Una propuesta de aplicación de las reglas de Hebb se dará con la combinación de las redes de Hopfield con los Mapas Autoorganizados de Kohonen. La idea de combinar ambos enfoques ha sido explorado por [92] para la clasificación de imágenes de cerebros anormales y en [68] se utilizan redes de Hopfield continuas para la optimización de Mapas Autoorganizados de Kohonen.

Formulamos aquí las redes de Hopfield-Kohonen-Hopfield (HKH) las cuales son un sistema de dos redes de Hopfield conectadas por medio de una red de Kohonen (figura 5.1). La primera es una red de Hopfield habitual que permite el aprendizaje de patrones sobre el tiempo. Esta red cuenta con una regla de aprendizaje de Covarianza presináptica.

Las conexiones entre la primera y segunda red de Hopfield permiten el desarrollo de una clasificación no supervisada para los patrones que se almacen de la primera red de Hopfield. De este modo, la idea central es realizar una clasificación no supervisada de los mínimos de la red de Hopfield de forma que se puedan caracterizar en la segunda red. De esta forma, si bien la memoria queda almacenada de manera distribuida, existe una neurona selectiva a tal patrón fijo, que se puede abstraer en la segunda red o capa. Para



Cuadro 5.2: Visualización de los pesos generados por la Regla de Hebb Simple en el entrenamiento de la base de datos MNIST

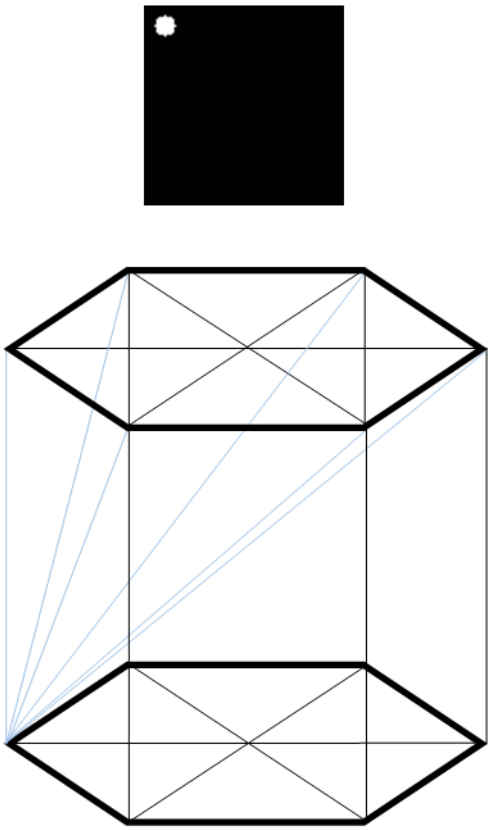


Figura 5.1: Arquitectura de la red HKH con la primera red de Hopfield conectada con la segunda por medio de conexiones de aprendizaje no supervisado similar al algoritmo de Kohonen. Las conexiones están simplificadas para permitir su visualización.

evitar que dos patrones converjan en un mismo punto, se utilizó la regla de la Covarianza postsináptica, ya que resta a los patrones diferentes. La inicialización de esta matriz de pesos, al contrario de las otras matrices inicializadas en 0, fue aleatoria para permitir el desarrollo del mapa. Como procedimiento de clasificación no supervisado, es muy similar al algoritmo de Kohonen, pero sin considerar conexiones cercanas y utilizando a la regla de la Covarianza.

Para el caso de la segunda red de Hopfield utilizaremos una regla de Hebb temporal dada por

$$\mathbf{W}_3(t+1) \leftarrow \mathbf{W}_3(t) + \alpha \mathbf{v}(t)\mathbf{v}^T(t-1), \tag{5.35}$$

donde $\mathbf{v}(t)$ representa el vector de actividad de la segunda red de Hopfield en el tiempo t y $\mathbf{W}_3$ es la matriz de pesos. Esta regla de aprendizaje representa una forma simple de una regla basada en el tiempo (véase [51]) que no introducimos por la aparente necesidad de usarse con modelos pulsantes. Estos trabajos se basan en investigaciones como [270] en el *Xenopus*, en las cuales se encontró la importancia del tiempo en la formación de LTP o LTD. Si se produce una alta actividad en la neurona presináptica dentro de 20 milisegundos (ms) antes de la actividad en la postsináptica entonces se produce LTP, mientras que por el contrario si se produce actividad en la presináptica dentro de 20 ms antes de alta actividad en la postsináptica se produce LTD. Por lo tanto, la regla dada en la ecuación anterior es una forma de la regla de Hebb Simple para incorporar algunas de estas ideas permitiendo la potenciación cuando se manifieste actividad posterior en el tiempo. Asimismo, el valor de la actividad de la segunda red está dada por

$$\mathbf{v}(t) = m(\mathbf{W}_2\mathbf{y}(t)), \tag{5.36}$$

donde $\mathbf{y}(t))$ es el vector de actividad de la primera red de Hopfield, $\mathbf{W}_2$ es la matriz de pesos y m es una función de activación no lineal definida por $m_i(y_i) = 1$ si $y_i = \text{máx}_j(y_j)$ y 0 en otro caso. De este modo, la primera red de Hopfield primero activa a una neurona v_a con mayor respuesta en la segunda red, la cual codifica a un mínimo de energía de la primera red de Hopfield, es decir, un patrón recordado. Posteriormente, un segundo patrón es presentado en la primera red y abstraído en la segunda red para activar a una neurona $v_b \neq v_a$. Como $v_b(t) = 1$ y $v_b(t+1) = 1$, entonces $(W_3)_{ba}$ se incrementa. Para lograr una efectiva recuperación de la información de la primera red de Hopfield a través de la segunda, definimos una matriz de pesos $\mathbf{W}_4$ con aprendizaje hebbiano simple.

Notemos que la matriz $\mathbf{W}_3$ es una matriz de transición y el aprendizaje cumple a la condición de Markov. La idea de tomar un aprendizaje con la Condición de Markov viene dada por la observación del proceso de recuperación de secuencias largas, aunque esto debería verificarse: para acceder a una letra del abecedario, recordamos a la inmediata siguiente y a partir de ella se pueden obtener las demás. De este modo, al activar la primera red de Hopfield se logra una convergencia a un mínimo local, que es un recuerdo almacenado, lo cual activa a la neurona asociada en la segunda red, la cual posteriormente

activa al nodo siguiente aplicando la matriz de transición y finalmente este nodo activa al siguiente patrón almacenado. Este procedimiento permite almacenar datos de forma secuencial y recuperarlos en el orden en el que aparecen. Sean $\mathbf{a}(1), \ldots, \mathbf{a}(n)$ los vectores de datos a almacenar. El algoritmo de entrenamiento, en resumen, está dado por

1. Inicializar $\mathbf{W}_1$, $\mathbf{W}_3$ y $\mathbf{W}_4$, $\mathbf{v}(-1)$ en ceros, $\mathbf{W}_2$ aleatoria (distribución uniforme dada en $[0, 1]$)
2. $t = 0$
3. $\mathbf{y}(t) = \mathbf{a}(t)$
4. $\mathbf{W}_1(t+1) = \mathbf{W}_1(t) + (2\mathbf{y}(t) - 1)(\mathbf{y}^T(t))$
5. $\mathbf{v}(t) = m(\mathbf{W}_2(t)\mathbf{y}(t))$
6. $\mathbf{W}_2(t+1) = \mathbf{W}_2(t) + (\mathbf{v}(t))(2\mathbf{y}^T(t) - 1)$
7. $\mathbf{W}_4(t+1) = \mathbf{W}_4(t) + \mathbf{y}\mathbf{v}^T$
8. $\mathbf{W}_3(t+1) = \mathbf{W}_3(t) + \alpha\mathbf{v}(t)\mathbf{v}^T(t-1)$
9. $\mathbf{v}(t-1) = \mathbf{v}(t)$
10. Repetir el paso 3 con $t \leftarrow t + 1$.

Para la recuperación, hacemos 0 a las diagonales de $\mathbf{W}_1$ y $\mathbf{W}_3$ siguiendo el procedimiento de Hopfield. Sea $\mathbf{a}$ un patrón incompleto. Entonces aplicamos el siguiente algoritmo para efectuar la recuperación

1. Inicializar $\mathbf{v}(-1)$ en 0 y $t = 0$.
2. $\mathbf{y}(t) = \mathbf{a}$
3. $\mathbf{y}(t+1) = F_0(\mathbf{W}_1(t)\mathbf{y}(t)$
4. Repetir 3 hasta convergencia o un número fijo.
5. $\mathbf{v}(t) = m(\mathbf{W}_2(t)\mathbf{y}(t))$

6. $\mathbf{v}(t+1) = \mathbf{W}_3(t)\mathbf{v}(t)$

7. $t \leftarrow t+1$

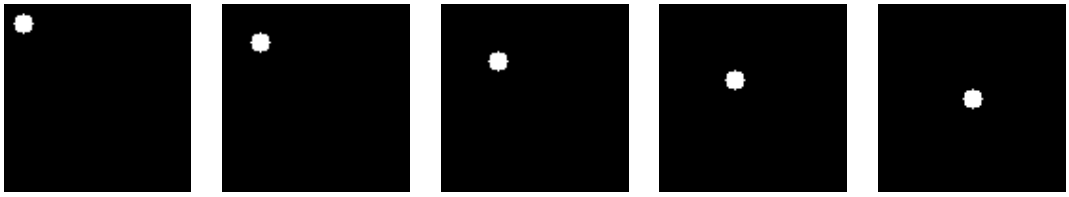
8. $\mathbf{y}(t+1) = \mathbf{W}_4(t)\mathbf{v}(t)$

9. Repetir el paso 3.

Como vemos, la matriz $\mathbf{W}_4$ tiene entrenamiento hebbiano para permitir que un nodo aprenda el patrón fijo y pueda ser recuperado al activarse.

Resultados

Se realizó una aplicación simple de las redes HKH para la memorización de una sucesión de imágenes binarias, representadas por el movimiento diagonal de una pelota. Las imágenes de tamaño 100×100 fueron efectivamente memorizadas por la primera red de Hopfield. Para la construcción de la segunda red se pueden utilizar los cálculos de la capacidad de la Red de Hopfield, aunque en este caso simple se definió como 15. En ninguno de los experimentos se encendió la misma neurona por medio de dos patrones distintos, lo cual sí ocurrió al cambiar la regla de aprendizaje. Las imágenes utilizadas fueron las siguientes en su respectivo orden (figuras dadas en 5.3).



Cuadro 5.3: Patrones utilizados para el entrenamiento. Después de aplicar el patrón incompleto, la red vuelve a reproducir el mismo patrón después del aprendizaje.

Posteriormente se ingresó el patrón incompleto especificada en la figura 5.2. En todos los experimentos fue posible recuperar el primer patrón original, por medio de la primera Red de Hopfield y la posterior secuencia por medio de la segunda.

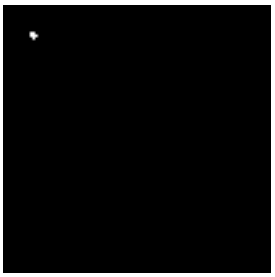


Figura 5.2: Patrón incompleto que se ingresó para la recuperación de información.

5.6. Conclusiones

A lo largo de este capítulo hemos introducido a las reglas basadas en Hebb, las cuales configuran un enfoque de entrenamiento de redes neuronales, diferente a los métodos basados en el Gradiente. Dichas reglas tienen cierto sustento biológico en los procesos de Potenciación a Largo Plazo (LTP) y Depresión a Largo Plazo (LTD), lo cual representa su ventaja principal frente a las reglas basadas en el Gradiente.

No obstante, la principal desventaja de las reglas de Hebb es que no han sido formuladas como optimizadores de funciones de costo relacionadas al error de clasificación *a priori*, en el sentido de que este hecho no es evidente en un principio, como es en el caso del Descenso de Gradiente, el cual ha sido formulado con este fin. En términos prácticos, la implementación directa de las reglas de Hebb en una red sin capas intermedias en un problema concreto como la clasificación de MNIST, resultó en una mayor inexactitud frente a los métodos basados en el Gradiente del estado de arte como Adam, incluso utilizando una sola época.

Aunque descepcionante, este resultado muestra que las reglas de Hebb implementadas de forma directa manifiestan una clara desventaja frente a Adam. Esto muestra que en términos prácticos y en una arquitectura simple, Adam sigue siendo un mejor algoritmo con respecto al costo y de forma suficientemente considerable como para sopesar las ventajas de los algoritmos hebbianos con respecto a su potencial aplicación en aprendizaje en tiempo real.

Por otro lado, también hemos sugerido una pequeña aplicación de la regla de Hebb para ampliar ligeramente el modelo propuesto por Hopfield utilizando algunas ideas sobre

el aprendizaje no supervisado dadas por Kohonen. Hemos incluido, de manera similar, una revisión más extensa sobre las estructuras relacionadas con la formación de memoria, destacando el Hipocampo y la Corteza Entorrinal. Siendo sugerido que la Red de Hopfield es un modelo de la región CA3 del Hipocampo, es relevente entender qué funciones tiene y cuál es la arquitectura de la misma para poder perfeccionar el modelo de Hopfield, así como el modelo HKH sugerido en esta tesis.

En el modelo HKH propuesto, se ha logrado entrenar una red para el aprendizaje de un patrón secuencial y la red fue capaz de reproducirlo de manera fiel. Queda pendiente ampliar esta investigación para ajustarlo a los modelos recientemente propuestos sobre la información temporal, espacial y conceptual que en apariencia es codificada por el Hipocampo, así como tratar de probar el algoritmo para imágenes más complejas.

Por lo tanto, la Regla de Hebb hasta el momento nos ha aportado resultados interesantes, pero al parecer la arquitectura de la red tiene una relevancia crucial para que estas reglas de plasticidad funcionen apropiadamente y una aplicación ingenua lleva a resultados adversos. Volveremos a abordar plenamente a la Regla de Hebb y su importancia hasta el Capítulo 6 donde discutiremos su uso y el papel que puede tomar en la Inteligencia Artificial moderna frente al uso de métodos de optimización como Adam en apariencia más robustos.

Capítulo 6

Redes de Aprendizaje Híbrido

Redes Convolucionales con Clasificación Hebbiana

> *The "fire together, wire together" Hebbian learning model is a central principle in neuroscience, but, surprisingly, it has found limited applicability in modern machine learning.*
>
> Aseem Wadhwa y Upamanyu Madhow [252]

En los capítulos 1 y 2 hemos abordado el tema de los métodos basados en el Gradiente y su relevancia en el entrenamiento de redes neuronales artificiales, al menos desde un enfoque contemporáneo. En el capítulo 5 se introdujeron a las reglas de Hebb, su importancia en el entrenamiento *online* así como su motivación y fundamento biológico. También verificamos experimentalmente que la Regla de Hebb, implementada ingenuamente, presenta graves desventajas frente a las Reglas basadas en el Gradiente, especialmente en cuanto a su exactitud. ¿Por qué una regla basada en la biología tiende a funcionar peor que una regla de carácter artificial?

Como se ha mencionado previamente, es bien sabido que la Regla de Hebb no ha sido completamente descrita. Incluso la Regla BCM y modelos más recientes, necesitan considerar la influencia de neuromoduladores, quienes tienen una importante influencia en el aprendizaje asociativo. ¿Es la Regla de Hebb una regla esencialmente incompleta? ¿El conocimiento acerca de los aspectos funcionales del aprendizaje asociativo yace latente o

lejos de nuestro alcance? Más aún, es posible incluso que la adición de moduladores y otros componentes quizá aún desconocidos terminen desembocando en una regla de aprendizaje posiblemente muy cercana a métodos como Adam. Es un hecho conocido que el aprendizaje Hebbiano está incompleto. Además de la actividad presináptica y postsináptica, [139] indica la existencia de un tercer factor en la plasticidad sináptica relacionado con los neuromoduladores y neurotransmisores, así como otros aspectos. Algunos de estos factores (específicamente la influencia de la acetilcolina y dopamina[1]) fueron modelados en [98], aunque se reconoce que los mecanismos exactos en los que operan siguen sin ser del todo comprendidos. También otros mecanismos neuronales pueden jugar un papel muy importante en el aprendizaje y con ello solventar las carencias manifestadas en versiones simples de la regla de Hebb. Esta discusión será objeto del presente capítulo.

Otra posible respuesta sobre este fenómeno recae simplemente en aceptar que las reglas basadas en el gradiente son mejores que las basadas en Hebb, al menos desde el punto de vista de la optimización. Esto es evidente desde lo expuesto anteriormente: tanto en sentido práctico como teórico esto es comprobable. Las reglas de Hebb, son por su lado más eficientes en el tiempo pero no en el costo, y la situación inversa se presenta con los descensos del Gradiente, pero una diferencia considerable en el costo como la encontrada previamente nos induciría a desconsiderar la regla de Hebb, al menos desde un sentido práctico. Puede que la mayor eficiencia de las reglas Hebbianas pueda traducirse en una mayor capacidad para su entrenamiento en bases de datos aún más grandes, con lo cual puedan alcanzar la exactitud de los métodos convencionales.

También es posible que la *regla de aprendizaje natural*, como llamaremos a la hipotética forma de aprendizaje presente en organismos complejos, tenga un peor desempeño que Adam, si se ejecuta para bases de datos finitas. Evidencia de lo anterior es que las modernas redes neuronales convolucionales pueden alcanzar (y hasta rebasar) el desempeño humano en el Imagenet. He y sus colaboradores [86] en el 2015 propusieron una CNN que lograba una exactitud de 4.94 % en el error top-5, lo cual supera a las mediciones realizadas con participantes humanos.

[1]Otros neuromoduladores como la noradrenalina y serotonina también parecen jugar un factor importante en la plasticidad sináptica

Asimismo, debemos considerar que posiblemente aspectos meta o hiperparamétricos tengan una notable influencia en el desempeño de la Regla de Hebb. Esta fue la conclusión alcanzada en nuestro artículo [6]. Modificar la arquitectura puede resolver algunos problemas y aumentar la exactitud de la red. Si bien la diferencia principal de los enfoques del gradiente y hebbiano radica en el método de optimización únicamente y bajo arquitecturas similares el costo tiende a reducirse mejor usando descenso de Gradiente, es posible que el desarrollo de arquitecturas más capaces permita mejorar considerablemente el desempeño del aprendizaje hebbiano, al punto de acercarlo a la exactitud obtenida usando Adam.

Finalmente debemos considerar la posibilidad de que los métodos basados en Hebb no sean apropiados para su implementación computacional, al menos no en modelos actualmente utilizados. Quizá modelos más realistas de redes neuronales sean más convenientes para su uso, en posiblemente ordenadores altamente paralelizados. En este sentido, es factible que los modelos más adecuados para las computadoras no coincidan con los mecanismos presentes en organismos vivos. Esta sugerencia está enlazada con la postura del cerebro-máquina en la Filosofía de la Mente (para una introducción a la discusió, revísese [124]), la cual podría no ser necesariamente cierta y tener consecuencias en aspectos aparentemente lejanos como éste.

Habiendo observado previamente que la implementación directa de las reglas de Hebb abordadas no supera al de los métodos basados en el Gradiente y basándonos en la discusión previa, podemos considerar las siguientes explicaciones potenciales:

- Aspectos funcionales del aprendizaje no han sido aclarados.
- Es posible que un algoritmo basado en el gradiente esté implícito en las neuronas.
- El aprendizaje dado por métodos como Adam es más efectivo que el aprendizaje natural.
- El enfoque hebbiano requiere de una mayor cantidad de datos para funcionar.
- La arquitectura tiene una influencia decisiva en el rendimiento de la regla de Hebb. Cambios en la misma podrían lograr una precisión cercana a los métodos tradicionales.

- La optimización Hebbiana está incomprendida o no es aplicable en computadoras modernas.

Una respuesta concreta puede involucrar uno o varios puntos anteriormente presentados. Como se ha especificado anteriormente, no abordaremos de golpe todas estas propuestas. No esperamos obtener una regla hebbiana más sólida en el costo que optimizadores modernos como Adam o RMSProp, aunque sí esperamos reducir el abismo aún existente entre ambas. Para probar cualquier propuesta es necesario recurrir a una base de datos finita, terreno en el que los métodos basados en el gradiente tienen ventaja.

En este capítulo abordaremos el problema de ambos enfoques y una propuesta para buscar cierta conciliación. Lo primero que haremos es tratar de definir cuál es la relación existente entre ambos métodos desde el punto de vista matemático.

Un segundo aspecto a considerar es la arquitectura. Hasta ahora, no hemos definido la regla de Hebb para varias capas, porque eso nos llevaría a plantear la retropropagación con la regla de Hebb (véase la discusión dada en [153]). Por el momento, no abordaremos este problema pero sí consideraremos variaciones en la arquitectura. Quizá la mayor variación sea el uso de redes convolucionales. El problema de la implementación de las reglas hebbianas dadas en el capítulo 5 es que no se ajusta a ningún modelo biológico. Al abordarse un problema exclusivamente visual, no se debe pretender resolver mediante una red de una capa. Un modelo de red convolucional, como se ha abordado previamente, es más plausible.

No pretendemos, por otro lado, tratar de resolver enigmas que las Neurociencias no han resuelto. Los métodos que se abordan a lo largo de esta tesis son exclusivamente matemáticos y computacionales, y por ende no se incluye experimentación alguna. Por ende, solamente podemos aspirar a introducir modelos nuevos, sugerir algunos, pero no podríamos comprobarlos experimentalmente. Sin embargo, los avances en el sentido teórico pueden motivar avances en términos más biológicos.

Una opción factible es tratar de dotar a las redes convolucionales de una regla de Hebb. No obstante, eso implicaría tratar de resolver el problema de la retropropagación, que mencionamos que no se abordará temporalmente. Una alternativa más económica es reducir este problema al de Transfer Learning y aprovechar de la eficiencia lograda

en las capas convolucionales por las grandes arquitecturas (desde AlexNet hasta las más recientes, pasando por VGG-19, GoogLeNet y ResNet-50) y dotar a la última capa de aprendizaje hebbiano. El uso conjunto de capas hebbiana y convolucional motiva al título, puesto que utiliza un *aprendizaje híbrido*, una parte mediante métodos aceptados por el estado de arte y la otra incorpora a las reglas Hebbianas. La posibilidad de alcanzar una exactitud aceptable es el tema medular de este capítulo.

6.1. Relación entre los enfoques Hebbiano y Gradiente

Primeramente trataremos de describir matemáticamente qué relaciones existen entre el aprendizaje hebbiano y el Descenso de Gradiente o sus formas más avanzadas. En una primera instancia debemos restringirnos a utilizar una forma del Descenso de Gradiente empleando el último dato obtenido y aplicar optimización sobre éste, en lugar de emplear una muestra (Descenso de Gradiente Estocástico) o todo los datos. De manera libre aparece la función de activación y la función de costo, aunque esta última debería ser plausible.

Regla del Perceptrón

Posiblemente, la regla con mayor relación con el Descenso del Gradiente sea la Regla del Perceptrón dada por Rosenblatt [214], discutida en el capítulo 3. Podemos probar que esta relación está bien fundada para una función de costo conocida en el siguiente teorema.

Teorema 4. *La Regla del Perceptrón es un Descenso de Gradiente de un batch de un dato y con función de costo de suma de errores cuadrados*

Demostración. El Descenso de Gradiente de la función de costo SSE de un batch está dado por la siguiente expresión

$$w_j \leftarrow w_j - \alpha \frac{\partial SSE}{\partial w_j}. \tag{6.1}$$

Utilizando un batch de un dato, la función SSE está dada por $\frac{1}{2}(y - NN_{\mathbf{w}}(\mathbf{x}))^2$, por lo tanto

$$\frac{\partial SSE}{\partial w_j} = (y - F(\mathbf{w}\cdot\mathbf{x})\frac{\partial}{\partial w_j}(y - F(\mathbf{w}\cdot\mathbf{x})) \tag{6.2}$$

$$= (F(\mathbf{w}\cdot\mathbf{x}) - y)F'(\mathbf{w}\cdot\mathbf{x})x_j. \tag{6.3}$$

Por lo tanto,

$$w_j \leftarrow w_j - \alpha(F(\mathbf{w}\cdot\mathbf{x}) - y)F'(\mathbf{w}\cdot\mathbf{x})x_j. \tag{6.4}$$

Utilizando la activación lineal $F(x) = x$, $F'(x) = 1$ y por lo tanto,

$$w_j \leftarrow w_j + \alpha(y - \mathbf{w}\cdot\mathbf{x})x_j. \tag{6.5}$$

Lo cual corresponde a la Regla del Perceptrón.

□

Este resultado, aunque en cierto sentido básico, da pie a nuevas consideraciones y permite deducir nuevas reglas de aprendizaje y funciones de costo de funciones conocidas. Por ejemplo, en el Teorema anterior, podemos sustituir a la función de activación lineal con la ReLU y obtenemos la regla de aprendizaje dada por

$$\Delta w_j = (\mathbf{w}\cdot\mathbf{x} - y)\mathbf{1}_0(\mathbf{w}\cdot\mathbf{x})x_j, \tag{6.6}$$

donde $\mathbf{1}_0$ es la función umbral en 0. Nótese que $(ReLU(\mathbf{w})\cdot\mathbf{x} - y)\mathbf{1}_0(\mathbf{w}\cdot\mathbf{x})x_j = (\mathbf{w}\cdot\mathbf{x} - y)\mathbf{1}_0(\mathbf{w}\cdot\mathbf{x})x_j$, ya que si el producto punto es menor que 0, entonces la salida general es 0 en cualquier caso. Esta regla es particularmente interesante ya que forza a todas las salidas a ser excitatorias, lo cual satisface la Ley de Dale [51]. A esta regla la nombraremos *Regla ReLU*.

Regla de Hebb

Utilizando los pasos del teorema podemos tratar de inferir cuál es la función objetivo para la Regla de Hebb. Para ello, sea E la función de costo. Idealmente en la regla de Hebb se debe satisface que

$$\frac{\partial E}{\partial w_j} = -yx_j. \tag{6.7}$$

Integrando ambos lados obtenemos que

$$E = \int -yx_j dw_j = -y(w_j x_j) + c. \tag{6.8}$$

En particular una curiosa elección de c nos lleva a

$$E = -y\mathbf{w} \cdot x. \tag{6.9}$$

En general para n datos esta función estaría dada por

$$E = -\sum_{i=1}^{n} y_i \mathbf{w} \cdot \mathbf{x}_i. \tag{6.10}$$

Lo cual representa a una función de *energía*. Hemos probado la siguiente proposición

Proposición 7. *La Regla de Hebb Simple es un Descenso de Gradiente para la función de costo* $E = -\sum_{i=1}^{n} y_i \boldsymbol{w} \cdot \boldsymbol{x}_i$ *con un batch de un dato y una función de activación lineal (ReLU si* $y \geq 0$ *y* $x_j \geq 0$*)*

En el modelo de tasa de disparo, cada salida de neurona representa una frecuencia de disparo, por lo que es siempre no negativa y la función lineal coincide con ReLU en este caso. Además en la Regla de Hebb los pesos son crecientes, por lo que siempre tendremos dicha salida positiva.

De la misma forma, podemos encontrar que la función objetivo para la regla de la Covarianza está dada por

$$E = -\sum_{i=1}^{n} (y_i - \theta)\mathbf{w} \cdot \mathbf{x}_i. \tag{6.11}$$

Interesante también resulta ser la Regla de Oja, puesto que una solución de la ecuación diferencial

$$\frac{\partial E}{\partial w_j} = -yx_j + \beta y^2 w_j, \tag{6.12}$$

está dada por

$$E = -y\mathbf{w} \cdot \mathbf{x} + \frac{\beta}{2} y^2 \sum_i^m |w_j|^2. \tag{6.13}$$

Lo cual es una función de energía con un regularizador que penaliza el tamaño de los pesos. El valor absoluto está garantizado porque los pesos nunca son negativos.

Estos resultados muestran en general que las reglas basadas en Hebb son en realidad reglas basadas en el gradiente con una función de costo específica y un batch de un dato, lo cual resuelve una pequeña pero significativa parte de la discusión, consistente en determinar la relación exacta entre ambos enfoques. Sin embargo, el descenso de Gradiente clásico sigue teniendo la ventaja de considerar más de un dato. Abordaremos este problema a continuación.

Optimización sobre n datos

Hasta ahora, los enunciados anteriores tienen una limitada aplicabilidad, puesto que mencionan que las reglas de Hebb son versiones limitadas del Descenso de Gradiente. No obstante, la Regla de Hebb Simple (y también la Regla de la Covarianza) son Descensos de Gradiente completos, tal como muestra el siguiente resultado

Teorema 5. *La Regla de Hebb Simple es un Descenso de Gradiente para la función de costo* $E = -\sum_i y_i F(\boldsymbol{w} \cdot \boldsymbol{x}_i)$ *con la función de activación ReLU o lineal.*

Demostración. El Descenso de Gradiente con la función de costo E está dado por la regla recursiva

$$w_j \leftarrow w_j - \alpha \frac{\partial E}{\partial w_j}. \tag{6.14}$$

Por lo tanto,

$$w_j \leftarrow w_j + \alpha \sum_{i=1}^{n} y_i x_{ij}. \tag{6.15}$$

La Regla de Hebb Simple para $\frac{\alpha}{n}$ está dada por

$$w_j \leftarrow w_j + \frac{\alpha}{n} y_i x_{ij}, \tag{6.16}$$

para cada $i = 1, \ldots, n$. Por lo tanto, sea w_j^0 el valor inicial de la sucesión. Tras una época aplicada del Descenso de Gradiente se tiene

$$w_j^1 = w_j^0 + \alpha \sum_{i=1}^{n} y_i x_{ij}. \tag{6.17}$$

Aplicando q veces la Regla de Hebb Simple, obtenemos

$$w_j^q = w_j^0 + \frac{q\alpha}{n} \sum_{i=1}^{n} y_i x_{ij}, \tag{6.18}$$

puesto que, si $q = 1$ tenemos la regla de Hebb aplicada al primer dato, y si $w_j^{q-1} = w_j^0 + \frac{(q-1)\alpha}{n} \sum_{i=1}^{n-1} y_i x_{ij}$, una segunda aplicación de la regla de Hebb conduce a

$$w_j^q = w_j^{q-1} + \frac{\alpha}{n} y_q x_{qj} \tag{6.19}$$

$$= w_j^0 + \frac{(q-1)\alpha}{n} \sum_{i=1}^{q-1} y_i x_{ij} + \frac{\alpha}{n} y_q x_{qj} \tag{6.20}$$

$$= w_j^0 + \alpha \sum_{i=1}^{q} y_i x_{ij}. \tag{6.21}$$

Por inducción sobre q observamos finalmente que la aplicación q veces de la Regla de Hebb conduce a $w_j^q = w_j^0 + \frac{q\alpha}{n} \sum_{i=1}^{q} y_i x_{ij}$. En particular, para $q = n$ tenemos la expresión deseada. Eso muestra que la aplicación n veces de la Regla de Hebb (sobre todos los datos) es equivalente a la aplicación de una época del Descenso de Gradiente. Las funciones de activación lineal y ReLU son intercambiables puesto que todas las variables y constantes son no negativas. □

El anterior es un resultado relevante para esta tesis pues resuelve uno de los problemas centrales planteados sobre la relación, hasta ahora incomprendida por el autor, entre la Regla de Hebb Simple y el Descenso de Gradiente, resolviendo la pregunta clave de *¿derivan las neuronas?* o más concretamente, *¿realizan una optimización por Descenso de*

Gradiente?. La respuesta es que el modelo básico de Hebb es un Descenso de Gradiente y como corolario, esto implica que optimizan a la función de energía.

Igualmente interesante aún es el hecho de que aplicar más épocas tanto en el Descenso de Gradiente como en la Regla de Hebb es innecesario: si aplicamos q veces el Descenso de Gradiente obtenemos

$$w_j^q = w_j^0 + q\alpha \sum_{i=1}^{n} y_i x_{ij}. \tag{6.22}$$

Lo cual simplemente simboliza un cambio en la tasa de aprendizaje. La Regla de Hebb, para mejorar su funcionamiento, requiere forzosamente de introducir nuevos datos. Este teorema, desafortunadamente, no se puede replicar para la Regla del Perceptrón, puesto que depende de los pesos en su aplicación.

¿Qué significa que la Regla de Hebb optimice una función de energía? Hasta este punto ya es claro que efectúa un proceso de optimización, pero no está dirigido al error de clasificación, como lo sería SSE. ¿Cuál es la relación de la función de energía con el error de clasificación? Con fines de simplicidad, supongamos que $y \in \{0, 1\}$, lo cual representa una clasificación binaria. Si $y = 0$, entonces cualquier salida es válida. Si $y = 1$ entonces minimizar la función de costo implica maximizar $\mathbf{w}$. Esto tiene sentido ya que la Regla de Hebb solamente aumenta si $y = 1$. Sin embargo, cualquier aumento en $\mathbf{w}$ minimiza la función de energía y al ser la regla de Hebb creciente, esta minimización está garantizada, demostrando de otra forma el teorema previo.

Tal teorema aporta información más relevante para el caso de la Covarianza, pues su función de energía no se comporta de manera tan trivial como en el caso de la regla de Hebb. Consideremos $\theta \in (0, 1)$. Entonces dadas estas condiciones, minimizar la función de energía $E = -\sum_{i=1}^{n}(y_i - \theta)\mathbf{w} \cdot \mathbf{x}_i$ minimiza, en ciertas condiciones, el error de clasificación, pues si $y_i = 0$, $y_i - \theta < 0$ y por lo tanto cualquier reducción sobre los pesos minimiza la energía pero también la pérdida $|y_i - \mathbf{w} \cdot \mathbf{x}_i|$ si los pesos son positivos. Si $y_i = 1$, la energía se reduce si se aumentan los pesos, lo cual reduce el error de clasificación si $\mathbf{w} \cdot \mathbf{x} < 1$.

El anterior análisis nos permite inferir una nueva regla, basada en la Covarianza, que permita tanto la reducción de la energía, como es buscado en las reglas de Hebb, como el error de clasificación. Por ejemplo, un candidato fuerte es utilizar la Regla de Covarianza,

pero añadiendo una activación sigmoide.

Las observaciones previas pueden condensarse en la siguiente proposición:

Proposición 8. *Si $y_i \in \{0,1\}$, $\theta \in (0,1)$, $x_i \geq 0$, la Regla de la Covarianza minimiza tanto $E = -\sum_{i=1}^{n}(y_i - \theta)\boldsymbol{w}\cdot\boldsymbol{x}_i$ como $(y_i - \sigma(\boldsymbol{w}\cdot\boldsymbol{x}_i))^2$ (únicamente para la i en cuestión), convergiendo a un mínimo local para E.*

Demostración. Siguiendo la demostración del teorema anterior, podemos verificar que la Regla de la Covarianza realiza un Descenso de Gradiente para E (usando activación lineal). Veamos que también reduce el error de clasificación SSE.

Si $y_i = 0$ en la iteración k, entonces se decrementan los pesos, obteniendo que $\sigma(\mathbf{w}^{k-1}\cdot\mathbf{x}_i) \geq \sigma(\mathbf{w}^k\cdot\mathbf{x}_i)$, de forma que, como la sigmoide es siempre positiva

$$
\begin{aligned}
(\sigma(\mathbf{w}^{k-1}\cdot\mathbf{x}_i))^2 &\geq (\sigma(\mathbf{w}^k\cdot\mathbf{x}_i))^2\\
(-\sigma(\mathbf{w}^{k-1}\cdot\mathbf{x}_i))^2 &\geq (-\sigma(\mathbf{w}^k\cdot\mathbf{x}_i))^2\\
(y_i - \sigma(\mathbf{w}^{k-1}\cdot\mathbf{x}_i))^2 &\geq (y_i - \sigma(\mathbf{w}^k\cdot\mathbf{x}_i))^2.
\end{aligned}
$$

Lo cual se traduce en un error en la clasificación. Si $y_i = 1$ en la iteración k entonces los pesos aumentan de forma que

$$
\begin{aligned}
\sigma(\mathbf{w}^{k-1}\cdot\mathbf{x}_i) &\leq \sigma(\mathbf{w}^k\cdot\mathbf{x}_i)\\
1 - \sigma(\mathbf{w}^{k-1}\cdot\mathbf{x}_i) &\geq 1 - \sigma(\mathbf{w}^k\cdot\mathbf{x}_i)\\
(1 - \sigma(\mathbf{w}^{k-1}\cdot\mathbf{x}_i))^2 &\geq (1 - \sigma(\mathbf{w}^k\cdot\mathbf{x}_i))^2\\
(y_i - \sigma(\mathbf{w}^{k-1}\cdot\mathbf{x}_i))^2 &\geq (y_i - \sigma(\mathbf{w}^k\cdot\mathbf{x}_i))^2.
\end{aligned}
$$

Por lo tanto se reduce el error de clasificación SSE en ambos casos.

□

Finalmente hemos podido probar que la Regla de la Covarianza disminuye tanto el error de clasificación como la energía de la red. Esta reducción, a pesar de seguir una dirección de descenso, no es la máxima, por lo que la Regla de la Covarianza puede tener

una convergencia mucho más lenta a un mínimo local que los métodos convencionales, especialmente Adam.

En las posteriores secciones abordaremos la posibilidad de modificar la arquitectura para ampliar la exactitud de clasificación, habiendo esclarecido la relación entre ambos enfoques desde un punto de vista formal.

6.2. Redes Convolucionales con Clasificación Hebbiana

Dos de las explicaciones potenciales del fallo de la aplicación directa de la regla de Hebb dada en la introducción del capítulo se centraron tanto en la arquitectura como en que el modelo sea inadecuado. Desde LeCun [146], el uso de Redes Neuronales Convolucionales se ha verificado como un modelo válido para el reconocimiento de caracteres manuscritos, y en general del reconocimiento de patrones visuales desde el surgimiento de la AlexNet [133]. Como hemos discutido, las redes neuronales convolucionales no son un modelo exacto de la Corteza Visual, pero paradójicamente su funcionamiento tiende a ser mejor que modelos como HMax, situación similar con respecto a la comparación de la regla de Hebb contra Adam.

Como también hemos afirmado, los modelos también pueden estar incompletos o incluso ser inadecuados para la arquitectura de computadoras modernas, especialmente para el procesamiento en serie. En cualquier caso, las CNNs han demostrado ser un modelo biológicamente inspirado que ha tenido un éxito notable. Esto quiere decir que la arquitectura selecta tiene una influencia notable en el desempeño de la red, dándonos atisbos de cómo superar este problema. Una solución, por lo tanto, sería introducir capas convolucionales al aprendizaje Hebbiano.

Tal como se ha presentado, las arquitecturas profundas de redes neuronales convolucionales suelen ser entrenadas mediante métodos basados en el gradiente, algo que aparece desde su concepción por medio de Yann Le Cun [146], cuyo título del artículo literalmente menciona *Gradient-based learning*. Podemos tratar de introducir el aprendizaje hebbiano a las CNNs de tres formas principales:

1. Disponer de una arquitectura convolucional y entrenarla de forma hebbiana.
2. Entrenar la CNN con métodos tanto hebbianos como del gradiente simultáneamente.
3. Pre-entrenar la CNN con métodos convencionales y utilizar la última capa para aprendizaje hebbiano.

La primera opción podría considerarse ideal y como un modelo más biológico. Sin embargo, podría no alcanzar la exactitud lograda utilizando Adam, ubicándose aún lejos. La opción parece razonable para evitar una eventual caída de la exactitud, pero podría generar una solución fuera del tiempo real. Es por ende que la opción 3 se ha considerado, tal como presentaremos a continuación.

La principal ventaja que proporciona el aprendizaje hebbiano con respecto al enfoque del gradiente radica en, como se ha discutido, la posibilidad de efectuarse de forma *online*. La Regla de Hebb, en particular la Simple, parece ser un forma eficiente de aprendizaje neuronal incluso desde términos computacionales, siendo posible efectuarse en grandes volúmenes de datos. Es por ello que el entrenamiento de CNNs utilizando ambos enfoques solamente reduce el potencial de aplicación de la Regla de Hebb (ya que al añadir métodos basados en el gradiente no puede efectuarse de forma online) y a la vez merma la exactitud posiblemente alcanzable utilizando puramente métodos basados en el gradiente.

En esta tesis proponemos una variante de la opción 3, consistente en utilizar a la red convolucional como extractor de características, preentrenado con una base de datos considerable, y utilizar a la última capa para la clasificación final. Por un lado, a pesar de que esto no resuelve las preguntas elaboradas sobre la regla de Hebb y el Descenso de Gradiente, supone una forma alternativa de aprovechar tanto el poder de las CNNs logrado como tratar de utilizar las ventajas ofrecidas por la Regla de Hebb.

Intuitivamente, las redes convolucionales pueden considerarse como identificadores de patrones visuales específicos, en orden de complejidad creciente, de forma que se obtenga un vector descriptivo de la imagen. Un ejemplo ligeramente artificial para entender esto es considerar a la clasificación de objetos como animales. Una neurona puede calcular la presencia de un patrón correspondiente a patas, otra a plumas, otra a alas, otras más asociadas a cabezas de diferentes animales. La salida de estas neuronas es fácilmente

discriminable y la asociación de los estas características con animales concretos puede lograrse aplicando la regla de Hebb Simple.

Otra ventaja que supone este enfoque está relacionada con la disposición de redes neuronales capaces de efectuar *Transfer Learning* (aplicar una red neuronal preentrenada para resolver un problema distinto, sin tener que reentrenar la red en su totalidad), hacia donde se dirige esta aproximación. Desde la AlexNet [133], han surgido numerosas arquitecturas convolucionales profundas que resuelven exitosamente el problema de clasificación de imágenes. Esto nos permite disponer de numerosas herramientas para comparar la eficacia de las reglas de Hebb en diferentes CNNs.

Trabajos relacionados

Se han mencionado dos formas principales adaptadas por la literatura reciente de introducir aprendizaje Hebbiano a arquitecturas profundas y convolucionales para la clasificación de imágenes. No todos los trabajos relacionados tienen el mismo objetivo de efectuar *Transfer Learning* en tiempo real y en muchos casos el entrenamiento se realiza en varias épocas. Asimismo muchos trabajos no realizan la clasificación de la última capa utilizando la Regla de Hebb sino que utilizan Máquinas de Soporte Vectorial (*Support Vector Machines*, SVM).

La primera aproximación es disponer de entrenamiento Hebbiano a una arquitectura convolucional o similar. Al respecto, en el 2016, Wadhwa y Madhow propusieron el Aprendizaje Hebbiano Adaptativo (*Adaptative Hebbian Learning*, AHL) así como el Aprendizaje Hebbiano Discriminativo (*Discriminative Hebbian Learning*, DHL) para el entrenamiento de las capas convolucionales. La clasificación es realizada mediante una capa final SVM. En la clasificación de MNIST, logró una exactitud de 99.35 % utilizando AHL.

En el 2017, Bahroun, Hunsicker y Soltoggio, por su parte, propusieron la Redes Hebbianas Profundas (*Deep Hebbian Networks* o DHN) las cuales utilizan una arquitectura similar a las redes convolucionales utilizando un sistema de capas de Codificación Dispersa (*Sparse Coding Layers*, SCL), seguida de *Deep Pooling Layer* (DPL) y *Spatial Pooling Layer* (SPL). Al igual que las redes convolucionales de LeCun y el Neocognitrón de Fukushima [69], este sistema está basado en el funcionamiento de las Células Simples y Células

Complejas. La salida de la red fue entrenada utilizando SVMs, obteniendo 41.4 % de exactitud máxima en el dataset MIT-67 y 79.1 % en el CIFAR-10. Este trabajo supone una continuación de [14], aunque no mejora la exactitud alcanzada en el CIFAR-10.

Amato *et al* en el 2019 propuso utilizar los algoritmos de Aprendizaje Hebbiano Competitivo (*Competitive Hebbian Learning*, CHL), en particular *Winner-Takes-All* y Aprendizaje Hebbiano Supervisado para entrenar redes convolucionales conjuntas, en conjunto con métodos basados en el gradiente. La experimentación realizada verificó que era más relevante entrenar a las capas iniciales y finales, pero no las intermedias. Se utilizó entrenamiento por medio de épocas, aunque se observó que no eran necesarias tantas para la Regla de Hebb.

Finalmente, en el 2019 (año pasado al desarrollo de esta tesis), Magotra y Kim [168] desarrollaron el algoritmo *Hebbian Transfer Learning* (HTL), el cual consiste en entrenar a una red convolucional mediante los métodos tradicionales utilizando un dataset relacionado (CIFAR-10 en este caso) y reentrenar utilizando las reglas hebbianas (la regla de Oja en este caso) al dataset objetivo (CIFAR-100). Este método requiere del empleo de numerosas épocas para su convergencia, por lo que no fue diseñado para el aprendizaje *online*.

Entre otras aproximaciones destacan las Redes Neuronales Pulsantes (SNNs), como la propuesta por [76] alcanzando 77 % de clasificación en MNIST. Otra red [157], que utiliza un sistema jerárquico logra 82 ± 2 % de clasificación en el problema MNIST. Otra reciente [167] aproximación del año 2020 no utiliza SNNs sino una arquitectura compleja con reglas de aprendizaje locales y un sistema recurrente, así como el modelo general de Hebb, la Regla de Oja así como la regla del Perceptrón para obtener un resultado de clasificación de 97,45 %.

6.3. Metodología

En la literatura se observa que la introducción del Aprendizaje Hebbiano a las Redes Neuronales Convolucionales fue gradual y es un fenómeno reciente. Esto puede deberse a que, a pesar de que las redes convolucionales existen desde la década de los noventas, solamente hasta el 2013 (con la AlexNet) volvieron a tomar relevancia capital, a la vez

fortaleciendo el entrenamiento por optimizadores basados en el gradiente como Adam y RMSProp, así como el empleo de funciones de activación como ReLU y sus derivados. De alguna manera su efectividad pudo haber eclipsado el desarrollo de métodos basados en Hebb.

Como vemos, esbozos de las formas (fases) 2 y 3 del desarrollo de la inclusión de Hebb en CNNs aparecen apenas en el 2019. No obstante, su desarrollo por épocas merma su capacidad para implementarse de forma *online*, el cual es uno de los objetivos centrales de esta tesis. Lo que se pretende, a grandes rasgos, es poder entrenar (enseñar) a redes neuronales grandes con imágenes de video en tiempo real para su clasificación efectiva. Esto permitiría disponer de una forma de entrenamiento más "natural" de las mismas.

Un resumen de la metodología propuesta se expone en la figura 6.3. Una red convolucional preentrenada es un operador sobre una imagen $I \in \mathbb{R}^{A \times B}$ tal que $CNN : \mathbb{R}^{A \times B} \rightarrow \mathbb{R}^{K}$ cuya salida $CNN(I)$ es un vector de características (*feature vector*). Dicho vector es procesado por una red, entrenada mediante alguna regla de Hebb, para el posterior reconocimiento de la clase.

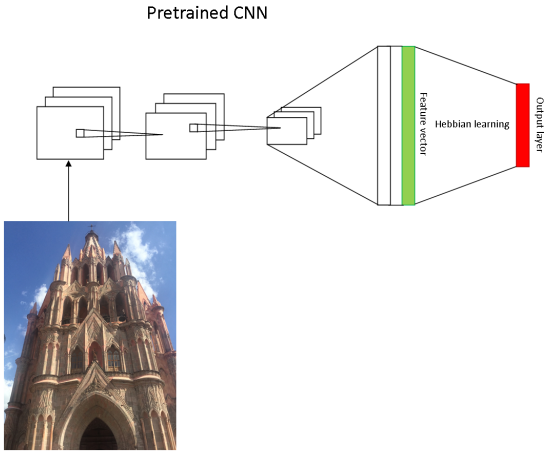


Figura 6.1: Red Convolucional con clasificación hebbiana propuesta para *Transfer Learning*

6.3.1. Bases de datos consideradas

Para verificar el funcionamiento de las reglas basadas en Hebb, se consideraron algunas bases de datos ya previamente consideradas como MNIST y EMNIST, así como el dataset de Cats-vs-Dogs de Tensorflow [62].

Adicionalmente se diseñó una base de datos consistente de frames de videos tomados de Pexeles considerando 10 categorías: avión, barco, caballo, computadora, desierto, pez, flor, perro, gato y fuego. Cada categoría utilizó 13 videos de 200 frames como conjunto de entrenamiento[2] y 2 videos de 200 frames como conjunto de prueba.

6.3.2. Capas convolucionales

Para la clasificación de MNIST, seleccionamos dos arquitecturas convolucionales inspiradas en la estructura VGG [228]. La primera, que nombraremos como $VGGS_1$, cuenta con la siguiente estructura:

1. 64 filtros convolucionales de 3×3
2. Batch Normalization
3. 64 filtros de 3×3
4. Batch Normalization
5. Max Pooling de 2×2
6. Dropout de 0,25 %
7. 128 filtros de 3×3
8. Batch Normalization
9. 128 filtros de 3×3
10. Batch Normalization

[2]Para las categorías flor y caballo se utilizaron dos videos de 150 y 50 frames en lugar de uno de 200 frames. Para la categoría computadora se utilizó un video de 190 y otro de 210.

11. Max Pooling de 2×2
12. Dropout de 0,25 %
13. Capa completamente conectada de 256 neuronas
14. Capa de clasificación con 10 neuronas

Todas las capas tienen funciones de activación ReLU a excepción de la última que cuenta con activación softmax. El modelo fue entrenado para minimizar la función de costo de Entropía Cruzada Categórica utilizando el optimizador Adam.

El segundo modelo, $VGGS_2$, tiene una mayor profundidad y está dado por

1. 64 filtros de 3×3 con el mismo padding
2. Batch Normalization
3. 64 filtros de 3×3 con el mismo padding
4. Batch Normalization
5. Max Pooling de 2×2
6. Dropout de 0,25 %
7. 128 filtros de 3×3 con el mismo padding
8. Batch Normalization
9. 128 filtros de 3×3 con el mismo padding
10. Batch Normalization
11. Max Pooling de 2×2 con strides de 2×2
12. Dropout de 0,25 %
13. 256 filtros de 3×3 con el mismo padding
14. Batch Normalization

15. Dropout de 0,25 %

16. Capa completamente conectada con 512 neuronas.

17. Capa de clasificación con 10 neuronas

Este modelo fue entrenado con 30 épocas utilizando RMSProp con $\alpha = 0{,}01$ inicial y $\rho = 0{,}9$. Esta red fue la que mayor exactitud observó en el capítulo de Redes Neuronales Evolutivas sobre el conjunto de prueba del EMNIST.

Para los datasets con imágenes RGB (Cats-vs-Dogs, Pexels), se utilizaron arquitecturas profundas preentranadas con el problema de clasificación de Imagenet. Tales arquitecturas son

1. VGG19 (2014) [228]

2. InceptionV3 (2016) [240]

3. ResNet50 (2016) [87]

4. InceptionResNetV2 (2016) [240]

5. Xception (2017) [40]

6. DenseNet201 (2017) [105]

7. MobileNetV2 (2018) [219]

6.3.3. Reglas de Hebb implementadas

En el capítulo sobre las Reglas de Hebb hemos introducido algunas de las reglas de aprendizaje más comunes, así como sus diferentes grados de motivación biológica y una implementación directa de las misma en una red simple. Como se ha visto, ninguna de estas reglas supera el desempeño alcanzado por las reglas basadas en el gradiente. En esta ocasión, volveremos a retomarlas pero ahora clasificando el vector de características que se extrae mediante las capas convolucionales. Este procedimiento permite hallar relaciones ocultas entre las características y las clases en tiempo real.

El hecho que se usen capas convolucionales en lugar de los pixeles mismos aporta mejores opciones. Primeramente nos ofrece un modelo más cercano al procesamiento de la información visual, puesto que la estructura de la corteza visual es más cercana a un modelo convolucional que a una red hebbiana sin capas intermedias. En términos de precisión, puede arrojar un conjunto linealmente separable que permita su fácil discriminación mediante los métodos hebbianos.

Para verificar estas ideas, utilizaremos los siguientes esquemas de reglas de Hebb:

- *Hebb*: Regla de Hebb Simple: $\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x}$.
- *L-Hebb*: Añade $S(w)$ a los pesos, utilizando la Regla de Hebb Simple
- *Covarianza*: Regla de Covarianza $\mathbf{w} \leftarrow \mathbf{w} + (y - \theta)\mathbf{x}$, con $\theta = 1$ y control logarítmico.
- *Oja*: Regla de Oja $\beta = 0{,}01$: $\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x} - \beta y^2\mathbf{w}$.
- *L-Oja*: Oja Rule con control logarítmico $S(w)$: $\mathbf{w} \leftarrow \mathbf{w} + v\mathbf{x} - \beta y^2 S(\mathbf{w})$.
- *BCM*: Regla BCM con $S(w)$: $\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x}(y - \theta_y)$ y $\theta_y \leftarrow \theta_y + \tau(y^2 - \theta_y)$. $\tau = 0{,}5$ con $\theta_y = 1$ inicial.
- *PLR*: Regla del Perceptrón: $\mathbf{w} \leftarrow \mathbf{w} + (y - \mathbf{w} \cdot \mathbf{x})\mathbf{x}$.
- *SPLR*: Regla del Perceptrón con σ: $\mathbf{w} \leftarrow \mathbf{w} + (y - \sigma(\mathbf{w} \ \cdot \mathbf{x}))\mathbf{x}$.
- *LSPLR*: Regla del Perceptrón con σ y $S(w)$: $\mathbf{w} \leftarrow \mathbf{w} + (y - \sigma(S(\mathbf{w}) \ \cdot \mathbf{x}))\mathbf{x}$

6.4. Resultados

Las reglas descritas anteriormente se aplicaron en su totalidad en el dataset de MNIST. Como la observación directa fue que añadir el control logarítmico era altamente necesario, en los siguientes casos fueron desconsideradas reglas sin este control.

6.4.1. MNIST

Para la primera base de datos, se efectuaron tres diferentes entrenamientos utilizando las arquitecturas $VGGS_1$ y $VGGS_2$. La segunda arquitectura se probó de dos maneras: una preentrenando las capas convolucionales mediante la base MNIST y la otra con EMNIST con caracteres para efectuar un *Transfer Learning*. Los resultados se recogen en la Tabla 6.1.

Cuadro 6.1: Exactitud sobre el conjunto de prueba utilizando los modelos $VGGS_1$, $VGGS_2$ sin y con Transfer Learning preentrenado con EMNIST. Los mejores resultados con las reglas basadas en Hebb se han destacado.

Model	$VGGS_1$	$VGGS_2$	Transfer Learning
Hebb	0.9844	0.9943	0.7086
L-Hebb	0.9914	0.9943	0.9456
Covarianza	0.984	0.9894	0.7086
Oja	0.9836	0.9925	0.7006
L-Oja	0.9914	0.9942	0.9472
BCM	0.9917	0.9942	0.9457
PLR	0.098	0.098	0.098
SPLR	0.094	0.094	0.1037
LSPLR	**0.9930**	**0.996**	**0.964**
Adam	0.9932	0.9966	0.9474

La regla del Perceptrón diferida (con control logarítmico y sigmoidal) mostró el desempeño más cercano al uso del algoritmo Adam en comparación, e inclusive superior en el problema de TL. En general el control logarítmico mostró mejoras significativas incluso para la Regla de Oja que podría pensarse como innecesario, pero sobretodo para la regla del Perceptrón, donde evitar su uso causa una notable caída de la exactitud. En cuanto a las reglas reguladas por el control logarítmico, la Rebla BCM, Oja e incluso la de Hebb Simple tuvieron un desempeño en general aceptable, no así la regla de la Covarianza, cuyo resultado fue ligeramente inferior.

6.4.2. Dogs-vs-Cats

Para el segundo dataset, consideramos a todas las redes convolucionales enunciadas, pero solamente a las reglas más significativas, que son las que incluyen control logarítmico, tal como se observó en los resultados anteriores. Dogs-vs-Cats se trata de una base de datos con imágenes RGB de tamaños variables, aunque suelen ajustarse al conveniente como 224×224, y utiliza dos categorías correspondientes a perros o gatos y es útil para probar algoritmos de Transfer Learning. Al tratarse de imágenes naturales, este paso es necesario para poder definir la utilidad de las ideas propuestas. Los resultados principales se condensan en la tabla 6.2.

Cuadro 6.2: Resultados del dataset Dogs-vs-Cats.

CNN	RMSprop	LSPLR	L-Hebb	Covarianza	L-Oja	BCM
VGG19	0.9858	0.9845	0.9699	0.9398	0.9742	0.9733
InceptionV3	0.9905	0.9901	0.9828	0.8422	0.9837	0.9832
ResNet50	0.9927	0.9914	0.9746	0.9424	0.9776	0.9802
Xception	0.9897	0.9897	0.9879	0.9622	0.9841	0.985
InceptionResNetV2	0.9931	0.9931	0.9767	0.9871	0.9746	0.9751
DenseNet201	0.9938	0.9936	0.9893	0.9884	0.9884	0.988
MobileNetV2	0.9898	0.9828	0.96	0.8387	0.9733	0.9678

En general todas las reglas bajo todas las arquitecturas presentaron un desempeño realmente aceptable, lo cual es un buen indicador para las reglas escogidas. La regla de la Covarianza, sin embargo, presenta una tendencia ligeramente inferior con respecto a las otras reglas consideradas, que se observa también al mostrar las gráficas de convergencia. Nuevamente, la regla diferida del Perceptrón tiene un desempeño bastante cercano a RMSprop, en muchos casos inclusive igual. Las reglas de Hebb Simple, Oja y BCM presentan una comparación más compleja, en algunos casos superior para una u otra regla. En promedio, sin embargo, la regla de Oja es mayor con 97,94 %, siguiendo BCM con 97,89 % y la regla de Hebb Simple con 97,74 %.

En cuanto a las arquitecturas, Xception pero principalmente DenseNet201 muestran los resultados en general más altos para las reglas de Hebb y más balanceados con respecto

a las reglas basadas en el gradiente, siendo la regla de Hebb Simple con 98,98 % lo mayor alcanzado por una regla basada en Hebb usando la DenseNet201.

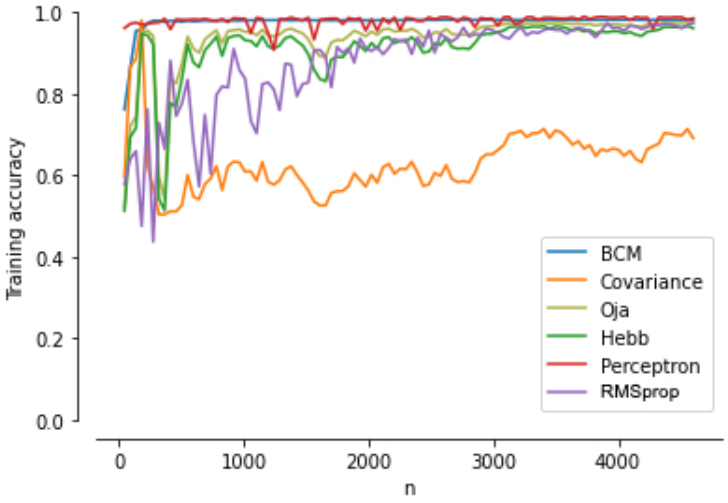


Figura 6.2: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red InceptionV3 para la extracción de características, utilizando los primeros 5000 ejemplos.

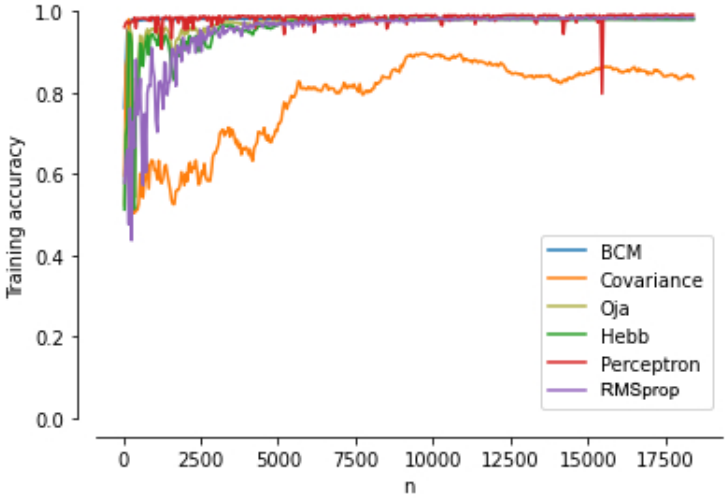


Figura 6.3: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red InceptionV3 para la extracción de características. Observamos cómo la exactitud incrementa con el número de ejemplos, salvo para la Regla de Covarianza, que falla en converger.

Las curvas de aprendizaje nos ayudan a describir cómo se da la convergencia en el conjunto de entrenamiento, si tal convergencia se presenta al menos empíricamente (no está

garantizada en las reglas de Hebb) y con qué rapidez lo hacen. Estas gráficas (figuras 6.3-6.10) muestran algunos patrones de cómo se dan tales convergencias. Para su graficación se entrenó con una parte incompleta del conjunto de entrenamiento y se probó con el conjunto total. Unos 400 pasos fueron utilizados para elaborar las gráficas.

La observación notoria es que las reglas BCM y del Perceptrón tienden a converger rápidamente sobre el conjunto de entrenamiento, mientras que Oja y Hebb Simple presentan un ritmo similar, incluso más elevado que RMSprop. La regla de la Covarianza, en general, parece no siempre converger y en los casos en los que no lo logra, termina oscilando o estancándose, lo cual descarta la posibilidad de añadir más datos para permitir su convergencia.

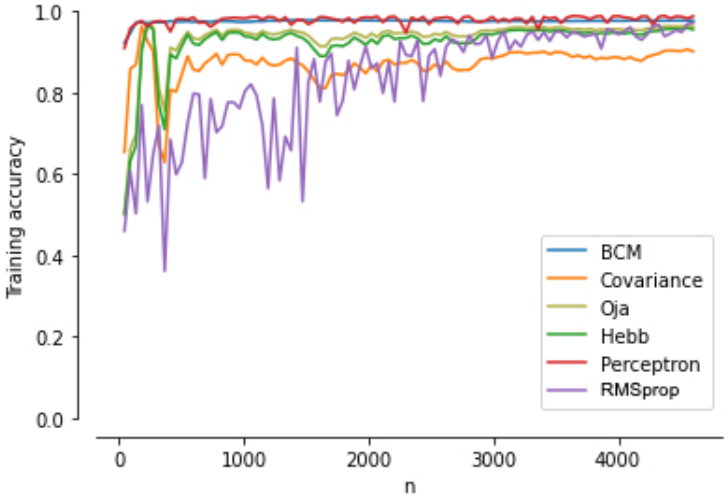


Figura 6.4: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red ResNet50 para la extracción de características, utilizando los primeros 5000 ejemplos.

6.4.3. Pexels

Las pruebas utilizando el dataset elaborado de Pexels, el cual ofrece un ejemplo de aplicación quizá más realista, arrojan resultados menos simples que los ofrecidos anteriormente, y que aparecen expuestos en el cuadro 6.3.

De manera extraordinaria, las reglas del Perceptrón y BCM, que hasta ahora habían reportado resultados muy adecuados, fallan en gran medida en converger, mientras que

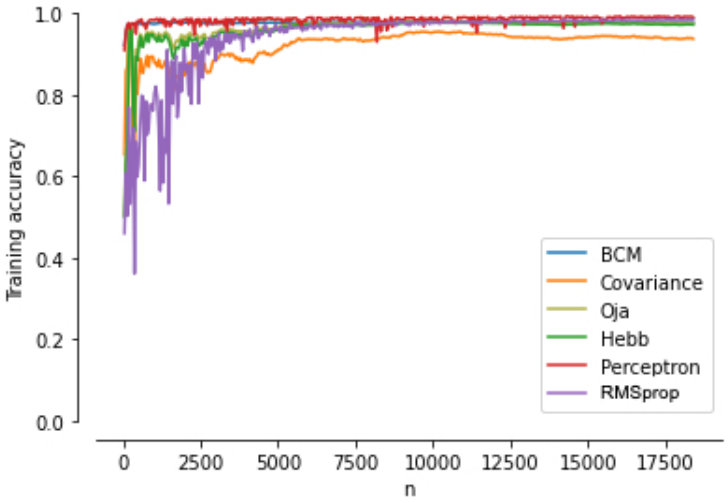


Figura 6.5: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red ResNet50 para la extracción de características. En este caso todas las reglas convergen de alguna manera.

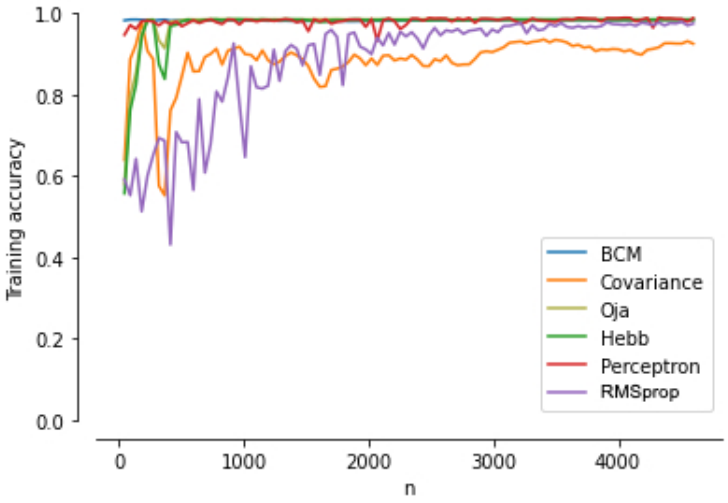


Figura 6.6: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red Xception para la extracción de características, utilizando los primeros 5000 ejemplos.

las reglas de Hebb, Covarianza y Oja se muestran ligeramente más estables. En este caso, la regla de la Covarianza ofreció resultados más altos compartativamente hablando.

Por otro lado, en este caso a diferencia del dataset Dogs-vs-Cats, la red convolucional asociada tuvo una mayor influencia en la convergencia o no en el error. La arquitectura Xception se muestra como más baleanceada, seguida de la DenseNet201, mientras que la

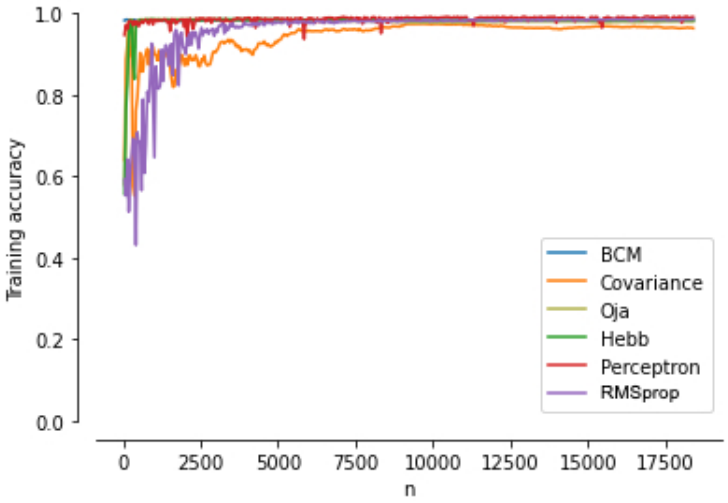


Figura 6.7: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red Xception para la extracción de características.

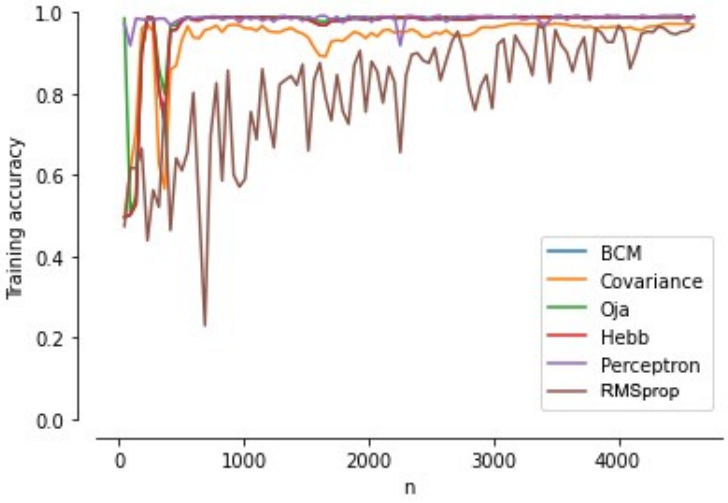


Figura 6.8: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red DenseNet201 para la extracción de características, utilizando los primeros 5000 ejemplos.

InceptionV3 produjo resultados significativamente más bajos.

6.5. Conclusiones

Este capítulo consiste en una continuación del capítulo de La Regla de Hebb, la cual se busca implementar para fines prácticos. Sin embargo, esta vez la idea central fue utilizar

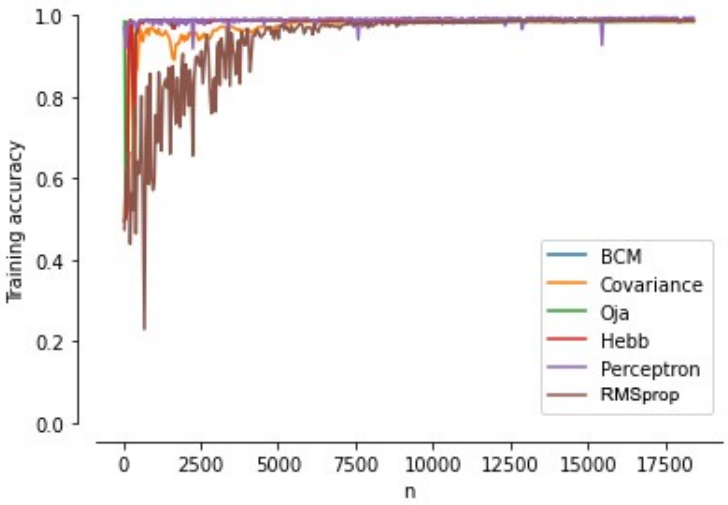


Figura 6.9: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red DenseNet201 para la extracción de características.

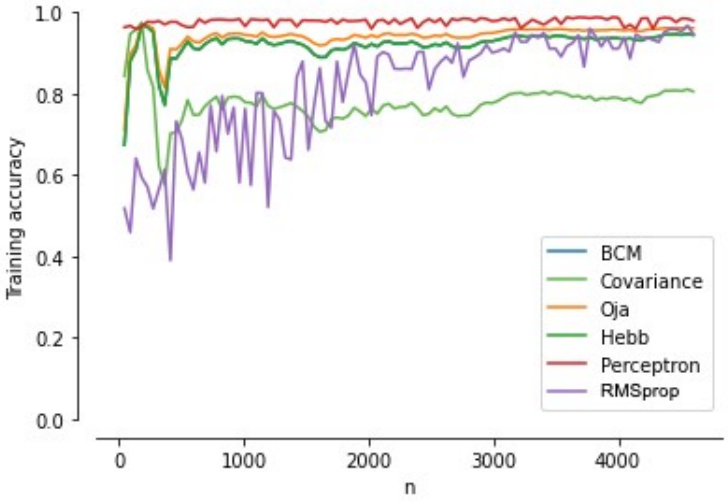


Figura 6.10: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red MobileNetV2 para la extracción de características, utilizando los primeros 5000 ejemplos.

a las redes convolucionales preentrenadas con algoritmos basados en el gradiente (como se ve en la sección de la Regla de Hebb) y entrenar solamente la capa de clasificación con las reglas de Hebb, de ahí que el nombre de la sección sea *Redes de aprendizaje híbrido* porque en lugar de desarrollar redes convolucionales de aprendizaje hebbiano como se ha tratado en la literatura, en esta ocasión se prefiere utilizar a las ventajas dadas por redes convolucionales preentrenadas, las cuales son lo mejor que se cuenta para el reconocimiento

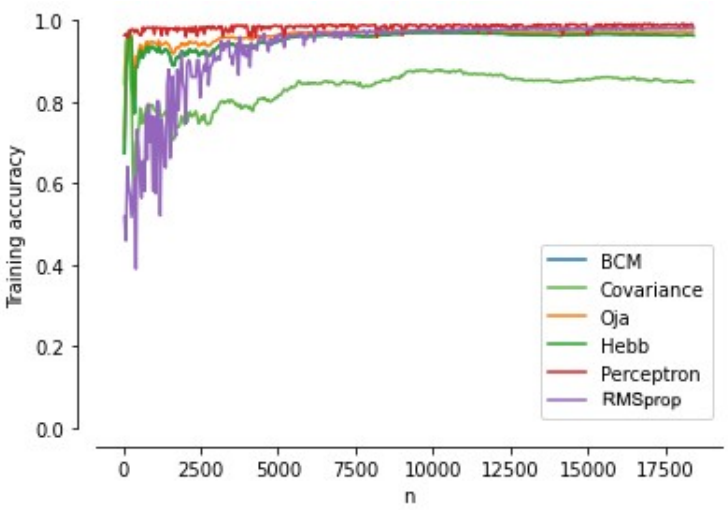


Figura 6.11: Curvas de aprendizaje sobre el conjunto de entrenamiento utilizando a la red MobileNetV2 para la extracción de características. Al igual que en la InceptionV3, la regla de Covarianza falla en converger.

Cuadro 6.3: Exactitud sobre el conjunto de prueba del dataset de Pexels

CNN	Adam	LSPLR	L-Hebb	Covarianza	L-Oja	BCM
VGG19	0.9057	0.335	0.8893	0.8673	0.8933	0.5345
InceptionV3	0.9973	0.1	0.71375	0.7355	0.7065	0.64975
ResNet50	0.9433	0.36175	0.84325	0.8705	0.82925	0.46975
InceptionResNetV2	0.9998	0.1502	0.8382	0.889	0.849	0.535
Xception	0.9995	0.16325	0.996	0.99625	0.9785	0.55225
DenseNet201	0.9833	0.1132	0.8878	0.9875	0.877	0.4333

de grandes categorías de objetos.

De esta forma, lejos de querer superar al Descenso del Gradiente y sus derivados, este capítulo trata de incorporarlo de forma apropiada. La principal desventaja que observamos de la regla de Hebb era su falta de exactitud, mientras que para el Descenso de Gradiente la desventaja primaria es su incapacidad de realizar aprendizaje en tiempo real. Sin embargo, una red neuronal convolucional preentrenada puede extraer características en tiempo real, por lo que para nuevas clases sólo se requiere reentrenar la última capa de clasificación. Adicionalmente, hasta el momento, no hemos definido una regla de Hebb multicapa que

sea correspondiente con las observaciones en la naturaleza. Una opción interesante es considerar a una capa de clasificación no supervisada (al estilo de los mapas autoorganizados de Kohenen) y una capa de clasificación final, como aplican de manera directa (sin capas convolucionales) [122, 98]. Por lo tanto, el enfoque presentado es conveniente para el fin práctico que se proyecta: disponer de un aprendizaje *online* de redes convolucionales.

Asimismo, este modelo es biológicamente más plausible que solamente incorporar la regla de Hebb de una sola capa, puesto que eso no considera mínimamente a la arquitectura necesaria para esos fines. No obstante, quizá la consecuencia más costosa de no entrenar a las capas convolucionales, especialmente a las últimas, recae en la imposibilidad de realizar *Transfer Learning* óptimo para numerosas clases, y hasta ahora sólo se ha probado la propuesta para 10 clases.

La adición de las capas convolucionales logró acercar enormemente al desempeño de las reglas basadas en el gradiente aplicadas a esta misma última capa, mejorando en todos los aspectos los resultados alcanzados en el capítulo 4 e incluso superando a los resultados de la literatura con reglas de Hebb en MNIST, pero esto debe tomar en cuenta que añade capas convolucionales preentrenadas, por lo que la comparación si bien no es justa es necesaria para justificar este cambio. Sin embargo, como en el capítulo 4, no hubo una relación clara entre el desempeño de las reglas y en este caso hasta de las redes convolucionales asociadas, y los resultados en cuanto a exactitud fueron en cierto modo inesperados. Debido a que la Regla de Oja es una mejora de la Regla de Hebb Simple y la Regla de la Covarianza es una extensión de esta regla para modelar LTD y a su vez la Regla BCM es un modelo biológicamente más plausible que la Regla de Covarianza, podríamos esperar una jerarquización en los resultados, la cual fue en gran medida inexistente. Por el contrario, la Regla de Hebb Simple fue una de las más estables (utilizando control logarítmico, el cual fue casi siempre una mejora relevante), junto con la Regla de Oja, aunque la última no siempre fue mejor que la regla simple.

La Regla de la Covarianza, por su parte, tuvo un desempeño menos evidente. Se trata de una regla que dio los resultados más bajos en la base de datos Dogs-vs-Cats pero más alto con respecto a las otras reglas hebbianas en el dataset de Pexels. Es posible que el parámetro θ requiera de ser ajustado de alguna manera. Sin embargo la regla BCM,

que ajusta el valor de θ, fue menos estable, desempeñándose bien en todas la pruebas a excepción de la última, donde falló al igual que la Regla del Perceptrón.

En cuanto a las arquitecturas convolucionales, las redes Xception y DenseNet201 mostraron los mejores resultados en ambas pruebas, haciéndolas apropiadas para incorporarlas en un sistema global. En el caso de la DenseNet, hemos discutido en el capítulo de Redes Neuronales Convolucionales que presenta ciertos atisbos para considerarla como un modelo de la Corteza Visual, aunque esta afirmación es arriesgada y el modelo es simplificado. Lo que se observó es que ambas redes son apropiadas, aunque la Xception resultó mejor para el problema de clasificación de Pexels que es el problema más conveniente. Un detalle notorio es que la Regla de la Covarianza funcionó bien en la DenseNet201 en ambos problemas, pero se desempeñó mejor con la Xception en el problema de Pexels. En cuanto a la regla BCM, es posible que su implementación en modelos de tasa de disparo sea poco adecuada o bien, necesite ciertos planteamientos previos sobre el comportamiento real de esta tasa, como la actividad espontánea. Un problema de la BCM es que si la actividades postsináptica o las actividades presinápticas son nulas, entonces no se produce ningún cambio, pero en el presente contexto de redes artificiales carentes de una actividad regular y espontánea, esto representa un caso fuerte de no asociación, por lo que la regla de la Covarianza se muestra más estable, aunque su parámetro parece no estar bien ajustado.

Por lo anterior y los resultados alcanzados en esta fase, la integración de los modelos convolucionales de la visión con las reglas de Hebb muestran la posibilidad de aprender categorías de imágenes naturales en tiempo real, el cual era uno de los objetivos fuertes de la presente tesis: dirigirse hacia un método de aprendizaje más natural para las redes neuronales en lugar de los clásicos entrenamientos por épocas. Esto nos abre un nuevo camino para el surgimiento de redes neuronales con comportamiento al menos aparentemente más natural y puede permitir a usuarios sin conocimientos previos entrenar sus propios clasificadores utilizando las redes preentrenadas y el entrenamiento *natural*.

Conclusiones

Why are real brains so much powerful than artificial neural networks?

Nelson Spruston y William L. Kath [233]

La pregunta con la que se inauguran estas conclusiones es quizá la pregunta que debió haberse situado en la Introducción. Es planteada por Spruston y Kath quienes sugieren que el problema está en la formulación de la neurona artificial y abren camino al área del Procesamiento Dendrítico, el cual no fue abordado en esta tesis. Son muchos los puntos de artificialidad que manifiestan las redes neuronales artificiales actuales. Pero también son muchas las ventajas que aportan y las aplicaciones que se han logrado obtener marcan una frontera entre lo conocido y la *Terra incognita* de las ciencias.

A través de los seis capítulos que integran esta tesis hemos finalmente logrado abordar el tema de la optimización de las redes neuronales en los dos niveles diferenciados, basando los algoritmos desarrollados en la evolución a nivel hiperparamétrico y en la regla de Hebb a nivel paramétrico. Sin embargo, no hemos podido dar respuesta a qué es lo que hace que las redes neuronales biológicas resuelvan problemas de clasificación tan complejos frente a los modelos existentes que no logran superar a los métodos de optimización tradicionales. En su lugar, hemos utilizado las ventajas de ambos enfoques para disponer con ello parte importante de la exactitud lograda por los métodos tradicionales como la rapidez ganada por las reglas de plasticidad formuladas.

Tres principales aportaciones son esbozadas en esta tesis: la primera (en el capítulo 4) se introduce una red HKH que permite el almacenamiento y recuperación de una secuencia de patrones; la segunda (capítulo 5) es un algoritmo evolutivo que recrea un ecosistema artificial utilizando las ecuaciones de Lotka-Volterra para optimizar la arquitectura de

las redes; y la tercera (capítulo 6) consiste en utilizar redes neuronales convolucionales preentrenadas y tomar la última capa para añadirle las reglas de plasticidad y con ello poder resolver problemas de clasificación en tiempo real. Con ello damos por concluido el esfuerzo que supuso tratar de disponer un modelo de la visión para autómatas.

Con la última oración del párrafo anterior hemos revelado el verdadero propósito de realizar investigaciones en este sector: tratar de emular el aprendizaje de organismos vivientes para su aplicación en autómatas (robots), los cuales pueden estar equipados con cámaras y sensores, recibiendo al igual que los organismos información en tiempo real que debe procesarse. Aprender en tiempo real es una de las ventajas más fuertes que ofrece el enfoque hebbiano frente a los métodos basados en el gradiente y cuenta con aplicaciones potenciales en el desarrollo de robots, sustituyendo al preentrenamiento que se realiza con grandes bases de datos, el cual además es lento.

Por otro lado, efectuar *Transfer Learning* por medio de la Regla de Hebb nos otorga la posibilidad de diseñar entrenamientos al mismo tiempo que se realiza la toma de datos, lo cual puede derivar en paqueterías más simples para el uso de redes neuronales, ya que los usuarios únicamente necesitarían tomar videos de las categorías requeridas para su clasificación y con ello disponer de un modelo útil para sus fines. Sin embargo, aún deben estudiarse los alcances de este enfoque, pues hasta ahora las pruebas realizadas se han hecho en *datasets* relativamente pequeños y con escasas categorías. No obstante, para contextos de *Transfer Learning* no suele ser tan recomendable incluir varias categorías, para lo cual se necesitaría reentrenar las capas convolucionales. En este tema de la regla de Hebb, eso se traduce en la necesidad de disponer de entrenamiento hebbiano para al menos las últimas capas convolucionales para permitir la adición de nuevas categorías.

Como se mencionó en la Introducción, el formato de esta tesis no ha sido estándar, al no plantearse un esquema clásico compuesto por una Introducción, Marco Teórico, Antecendentes, Metodología, Resultados y Conclusiones. Hemos diluido esta estructura en investigaciones separadas que conforman algunos capítulos, en especial los últimos. No obstante, su integración final es una labor pendiente, que se consagrará en las siguientes investigaciones que realice.

Con los resultados alcanzados, hemos podido tomar un punto de partida para la mode-

lación completa de numerosas operaciones neuronales. Como se habrá advertido, la revisión de la literatura ha sido más extensa de la realmente necesaria para realizar los aportes que se incluyeron. Esto no es accidental: se han puesto los cimientos para seguir con los pilares, y esta tesis debe seguirse escribiendo.

No obstante, así como el camino que se ha realizado ha sido largo, la senda que se proyecta es aún mayor. Sin embargo es de notar que las bases que soportan a los resultados logrados han requerido, de igual forma, un largo camino. De las investigaciones de Hubel y Wiesel en la Corteza Visual, la formulación del modelo jerárquico, hasta el surgimiento de las redes convolucionales y en particular de las redes profundas con la AlexNet en el 2013, se obtiene el lapso de más de medio siglo. Asimismo, muchas de las redes convolucionales que se han empleado fueron formuladas tres años antes de este trabajo. Nada de lo formulado sería posible sin esfuerzos como los anteriores, sino que tendríamos que iniciar de un punto de partida distinto.

La artificialidad sigue presente en los modelos elaborados, pero al parecer, ningún modelo escapa de tales puntos de artificialidad. Quizá la mayor artificialidad es que se sigan empleado métodos basados en el gradiente en el preentrenamiento de las capas convolucionales. Más bioinspirados son los enfoques que utilizan redes convolucionales con entrenamiento puramente hebbiano o aún más si utilizan redes neuronales pulsantes. Tampoco se utilizaron neuronas con procesamiento dendrítico.

Incorporar las mencionadas ideas resulta ampliamente interesante, pero no se ha procedido de esta forma por motivos meramente prácticos. La revisión que se realizó, por ejemplo, no arrojó redes pulsantes que superaran la exactitud lograda por las redes convolucionales y menos para el caso de largas categorías de imágenes. Muchos de estos modelos que se adjudican como biológicamente más plausibles como HMAX no incluyen más capas que las que ya se han descrito en la Corteza Visual, lo cual lleva al desarrollo de redes con un menor número de neuronas que las que utilizan las redes convolucionales. Por lo que se observa, tal parece que la carencia de neuronas es un punto de artificialidad más fuerte que utilizar métodos con una mayor base biológica de fondo.

Asimismo, como hemos dicho, muchos de los aportes se proyectan para considerarse como modelos para dispositivos autónomos, al simplificarse el aprendizaje en tiempo real

tanto en eficiencia como en recursos computacionales. En tales dispositivos, es posible añadir redes convolucionales preentrenadas, pero involucraría un cálculo mayor utilizar modelos más precisos que sustituyan a la tasa de disparo, la cual nos ha dado numerosas respuestas para la comprensión de los mecanismos subyacentes en el sistema nervioso central. Es por ello que considero que se puede seguir en esta dirección, siendo un camino que aún merece ser explorado.

6.6. Trabajo futuro

La integración de las aportaciones realizadas, así como el desarrollo de las aplicaciones prácticas descritas, son apenas una de las direcciones que pueden seguir al trabajo de tesis. Como hemos dicho, se han integrado más trabajos de los que se usaron con el fin de equiparlos en investigaciones futuras. De manera relevante, mencionaremos a la red HKH, la cual cuenta con dos redes de Hopfield que constituyen un modelo de la región CA3 del Hipocampo. Sería interesante tratar de considerar tanto los modelos existentes de la organización del Hipocampo como los resultados de las importantes investigaciones logradas en materia de neuronas multimodales, de lugar, de rejilla, de borde y demás. Esto podría derivar en un modelo *computacional* del Hipocampo, que logre con ello el almacenamiento y recuperación de memorias por parte de una red más grande.

6.6.1. Modelo del Anillo

Por fines de simplicidad llamaremos *anillo* a cada red de Hopfield. La red HKH está conformada por dos anillos. Sin embargo, tal red cuenta con conexiones con todas las neuronas entre sí, lo cual topológicamente es equivalente a un solo ciclo, pero las operaciones que realizan son diferentes, llevando a una diversidad de reglas plásticas[3]. Una extensión de la red neuronal planteada en el capítulo 6 que involucra a una red convolucional y a una capa con aprendizaje hebbiano, es convertir a dicha capa en un anillo, que relacione tanto entradas auditivas (para el momento por medio de reconocedores de voz) como visuales. Tal red tiene como nodos a neuronas selectivas a conceptos específicos, de una primera

[3]Lo cual es un punto de artificialidad, aparentemente

instancia sustantivos como iglesia, perro, gato, entre otras que aparecen en las categorías de Imagenet. Un esbozo de la arquitectura está dada por la figura 6.12.

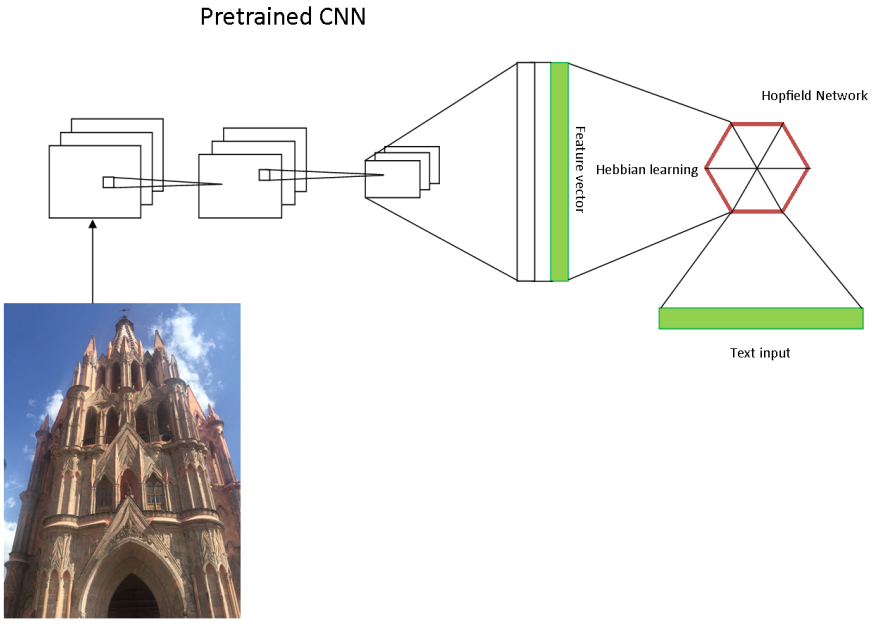


Figura 6.12: Modelo del Anillo con una red de Hopfield central. La entrada de texto puede ser considerada como auditiva si se incluye un reconocimiento de audio (*Speech-to-Text*).

De manera burda, el anillo central integra tanto las señales auditivas como visuales. De este modo, los nodos del anillo corresponden a neuronas selectivas a conceptos, siendo modelos de las células multimodales. Esto puede permitir la asociación de conceptos cercanos mediante el Anillo. Este sistema de red convolucional con una recurrente ha sido utilizado para el problema de *Image captioning*, como se observó. Un segundo módulo para la generación de sintaxis usando una LSTM puede ser añadido, dando dos redes recurrentes: una semántica y otra sintáctica, lo cual es consistente con la existencia de un área de Broca (que procesa sintaxis) y el área de Wernicke (que procesa semántica) [72].

No obstante, un problema de este planteamiento radica en que se toma una postura similar al enfoque de las *Grandmother cells*, que se ha discutido previamente y que no convence a todos los académicos. La red HKH considerada, por su parte, es una forma en la que los dos enfoques pueden entrar en armonía: en el primer anillo los recuerdos

se almacenan de forma distribuida pero de forma no supervisada logran asociarse a una sola neurona. Por ende, una segunda capa no supervisada puede añadirse a la hebbiana, la cual tanto puede contribuir a reducir el error de clasificación, como se logra en algunas implementaciones de la regla de Hebb como en [122].

Estas redes pueden ser capaces de reconocer objetos al asociarlos con la imagen acústica. A lo más, esto es un modelo para el reconocimiento de objetos estáticos que corresponden en el lenguaje como sustantivos o incluso adjetivos simples como colores. Un siguiente nivel requiere de la abstracción de verbos, la cual puede darse mediante un modelo de la vía dorsal de la Visión. Hasta el momento, no hemos considerados modelos de Aprendizaje por Refuerzo y la influencia de neuromoduladores como aparece en [98]. Su influencia será considerada en los trabajos futuros que se realicen en esta dirección.

A través de estas líneas redacto el fin de esta tesis de licenciatura, habiendo ya puesto sobre la misma el trabajo futuro que se proyecta. Con ello hemos definido una dirección que luce prometedora para el desarrollo de la Inteligencia Artificial en general. Sin embargo, dependemos de los resultados empíricos que se logren en el ámbito de Neurociencias para poder generar nuevos modelos. Tal vez ya se haya logrado colectar la información suficiente para dar el siguiente paso, pero tal vez persista incompleta para finalizarlo. Tal vez, incluso, su futuro resida en el mismo destino del Programa de Hilbert: el de la imposibilidad.

Bibliografía

[1] O. Abdel-Hamid, L. Deng, and D. Yu, *Exploring convolutional neural network structures and optimization techniques for speech recognition.*, in Interspeech, vol. 11, 2013, pp. 73–5.

[2] A. F. Agarap, *Deep learning using rectified linear units (relu)*, arXiv preprint arXiv:1803.08375, (2018).

[3] C. C. Aggarwal, *Neural networks and deep learning*, Springer, 10 (2018), pp. 978–3.

[4] H. H. Aghdam and E. J. Heravi, *Guide to convolutional neural networks*, New York, NY: Springer. doi, 10 (2017), pp. 978–3.

[5] F. J. Aguilar Canto, *Convolutional neural networks with hebbian-based rules in online transfer learning*, in Mexican International Conference on Artificial Intelligence, Springer, 2020, pp. 35–49.

[6] ——, *Eficacia de diferentes reglas hebbianas en el aprendizaje supervisado*, Tecnología Educativa Revista CONAIC, 7 (2020), pp. 92–97.

[7] C. R. Alavala, *Fuzzy Logic and Neural Networks: basic concepts & application*, New Age International, 2008.

[8] S.-i. Amari, *Dynamics of pattern formation in lateral-inhibition type neural fields*, Biological Cybernetics, 27 (1977), pp. 77–87.

[9] G. Amato, F. Carrara, F. Falchi, C. Gennaro, and G. Lagani, *Hebbian learning meets deep convolutional neural networks*, in International Conference on Image Analysis and Processing, Springer, 2019, pp. 324–334.

[10] T. J. Anastasio, *Tutorial on Neural Systems Modeling.*, Sinauer Associates, 2010.

[11] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, *Bottom-up and top-down attention for image captioning and visual question answering*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6077–6086.

[12] D. Bahdanau, K. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate*, arXiv preprint arXiv:1409.0473, (2014).

[13] Y. Bahroun, E. Hunsicker, and A. Soltoggio, *Building efficient deep hebbian networks for image classification tasks*, in International Conference on Artificial Neural Networks, Springer, 2017, pp. 364–372.

[14] Y. Bahroun and A. Soltoggio, *Online representation learning with single and multi-layer hebbian networks for image classification*, in International Conference on Artificial Neural Networks, Springer, 2017, pp. 354–363.

[15] A. Baldominos, Y. Saez, and P. Isasi, *Hybridizing evolutionary computation and deep neural networks: An approach to handwriting recognition using committees and transfer learning*, Complexity, 2019 (2019).

[16] ——, *A survey of handwritten character recognition with mnist and emnist*, Applied Sciences, 9 (2019), p. 3169.

[17] H. B. Barlow, *Summation and inhibition in the frog's retina*, The Journal of Physiology, 119 (1953), p. 69.

[18] T. Bartsch, J. Döhring, A. Rohr, O. Jansen, and G. Deuschl, *Ca1 neurons in the human hippocampus are critical for autobiographical memory, mental time travel, and autonoetic consciousness*, Proceedings of the National Academy of Sciences, 108 (2011), pp. 17562–17567.

[19] R. Battiti, *First and second-order methods for learning: between steepest descent and newton's method*, Neural Computation, 4 (1992), pp. 141–166.

[20] J. Bergstra and Y. Bengio, *Random search for hyper-parameter optimization*, The Journal of Machine Learning Research, 13 (2012), pp. 281–305.

[21] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, *Algorithms for hyper-parameter optimization*, in Advances in Neural Information Processing Systems, 2011, pp. 2546–2554.

[22] E. L. Bienenstock, L. N. Cooper, and P. W. Munro, *Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex*, Journal of Neuroscience, 2 (1982), pp. 32–48.

[23] T. V. Bliss and S. F. Cooke, *Long-term potentiation and long-term depression: a clinical perspective*, Clinics, 66 (2011), pp. 3–17.

[24] T. V. Bliss and T. Lømo, *Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path*, The Journal of physiology, 232 (1973), pp. 331–356.

[25] M. F. Bonner and A. R. Price, *Where is the anterior temporal lobe and what does it do?*, Journal of Neuroscience, 33 (2013), pp. 4213–4215.

[26] G. Boole, *Laws of thought (american reprint of 1854 edition)*, 1854.

[27] O. J. Braddick, J. M. O'Brien, J. Wattam-Bell, J. Atkinson, T. Hartley, and R. Turner, *Brain areas sensitive to coherent visual motion*, Perception, 30 (2001), pp. 61–72.

[28] C. Bromer, T. M. Bartol, J. B. Bowden, D. D. Hubbard, D. C. Hanka, P. V. Gonzalez, M. Kuwajima, J. M. Mendenhall, P. H. Parker, W. C. Abraham, et al., *Long-term potentiation expands information content of hippocampal dentate gyrus synapses*, Proceedings of the National Academy of Sciences, 115 (2018), pp. E2410–E2418.

[29] D. S. Broomhead and D. Lowe, *Radial basis functions, multi-variable functional interpolation and adaptive networks*, tech. rep., Royal Signals and Radar Establishment Malvern (United Kingdom), 1988.

[30] R. E. Brown and P. M. Milner, *The legacy of donald o. hebb: more than the hebb synapse*, Nature Reviews Neuroscience, 4 (2003), pp. 1013–1019.

[31] M. Brysbaert, M. Stevens, P. Mandera, and E. Keuleers, *How many words do we know? practical estimates of vocabulary size dependent on word definition, the degree of language input and the participant's age*, Frontiers in psychology, 7 (2016), p. 1116.

[32] R. CAPOR-HROSIK, M. T. M. VUKOVIC, and M. PIKULA, *Face detection by neural networks based on invariant moments*, Mathematical Models in Engineering and Computer Science, (2014).

[33] P. Cavalin and L. Oliveira, *Confusion matrix-based building of hierarchical classification*, in Iberoamerican Congress on Pattern Recognition, Springer, 2018, pp. 271–278.

[34] S. Chen, C. F. Cowan, and P. M. Grant, *Orthogonal least squares learning algorithm for radial basis function networks*, IEEE Transactions on neural networks, 2 (1991), pp. 302–309.

[35] Y. Chen, L. Zhu, P. Ghamisi, X. Jia, G. Li, and L. Tang, *Hyperspectral images classification with gabor filtering and convolutional neural network*, IEEE Geoscience and Remote Sensing Letters, 14 (2017), pp. 2355–2359.

[36] E. Cherubini and R. M. Miles, *The ca3 region of the hippocampus: how is it? what is it for? how does it do it?*, Frontiers in cellular neuroscience, 9 (2015), p. 19.

[37] S. W. Cho, N. R. Baek, M. C. Kim, J. H. Koo, J. H. Kim, and K. R. Park, *Face detection in nighttime images using visible-light camera sensors with two-step faster region-based convolutional neural network*, Sensors, 18 (2018), p. 2995.

[38] Y. Choe, *Anti-Hebbian Learning.*, Springer Publishing Company, Incorporated, 2015.

[39] Y. Choe and R. Miikkulainen, *Self-organization and segmentation in a laterally connected orientation map of spiking neurons*, Neurocomputing, 21 (1998), pp. 139–158.

[40] F. Chollet, *Xception: Deep learning with depthwise separable convolutions*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1251–1258.

[41] F.-X. Choo, *Spaun 2.0: Extending the world's largest functional brain model*, (2018).

[42] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, *Convolutional neural network committees for handwritten character classification*, in 2011 International Conference on Document Analysis and Recognition, IEEE, 2011, pp. 1135–1139.

[43] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, *Fast and accurate deep network learning by exponential linear units (elus)*, arXiv preprint arXiv:1511.07289, (2015).

[44] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, *Emnist: Extending mnist to handwritten letters*, in 2017 International Joint Conference on Neural Networks (IJCNN), IEEE, 2017, pp. 2921–2926.

[45] G. L. Collingridge, S. Peineau, J. G. Howland, and Y. T. Wang, *Long-term depression in the cns*, Nature reviews neuroscience, 11 (2010), pp. 459–473.

[46] M. Coltheart, *Grandmother cells and the distinction between local and distributed representation*, Language, Cognition and Neuroscience, 32 (2017), pp. 350–358.

[47] L. N. Cooper and M. F. Bear, *The bcm theory of synapse modification at 30: interaction of theory with experiment*, Nature Reviews Neuroscience, 13 (2012), pp. 798–810.

[48] U. B. Corrêa and R. M. Araújo, *Ae-charcnn: Char-level convolutional neural networks for aspect-based sentiment analysis*, in Mexican International Conference on Artificial Intelligence, Springer, 2019, pp. 124–135.

[49] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, Mathematics of control, signals and systems, 2 (1989), pp. 303–314.

[50] G. E. Dahl, T. N. Sainath, and G. E. Hinton, *Improving deep neural networks for lvcsr using rectified linear units and dropout*, in 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 8609–8613.

[51] P. Dayan and L. F. Abbott, *Theoretical neuroscience: computational and mathematical modeling of neural systems*, (2001).

[52] L. Deng, *The mnist database of handwritten digit images for machine learning research*, IEEE Signal Processing Magazine, 29 (2012), pp. 141–142.

[53] R. Desimone, T. D. Albright, C. G. Gross, and C. Bruce, *Stimulus-selective properties of inferior temporal neurons in the macaque*, Journal of Neuroscience, 4 (1984), pp. 2051–2062.

[54] H. R. Dimsdale-Zucker, M. Ritchey, A. D. Ekstrom, A. P. Yonelinas, and C. Ranganath, *Ca1 and ca3 differentially support spontaneous retrieval of episodic contexts within human hippocampal subfields*, Nature communications, 9 (2018), pp. 1–8.

[55] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, *Long-term recurrent convolutional networks for visual recognition and description*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2625–2634.

[56] O. Drees, *Untersuchungen über die angeborenen verhaltensweisen bei springspinnen (salticidae)*, Zeitschrift für Tierpsychologie, 9 (1952), pp. 169–207.

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

[58] H. Eichenbaum, *Prefrontal–hippocampal interactions in episodic memory*, Nature Reviews Neuroscience, 18 (2017), pp. 547–558.

[59] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen, *A large-scale model of the functioning brain*, science, 338 (2012), pp. 1202–1205.

[60] D. Elliott and F. Keller, *Image description using visual dependency representations*, in Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013, pp. 1292–1302.

[61] J. L. Elman, *Finding structure in time*, Cognitive Science, 14 (1990), pp. 179–211.

[62] J. Elson, J. J. Douceur, J. Howell, and J. Saul, *Asirra: A captcha that exploits interest-aligned manual image categorization*, in Proceedings of 14th ACM Conference on Computer and Communications Security (CCS), Association for Computing Machinery, Inc., October 2007.

[63] O. Eluyode and D. T. Akomolafe, *Comparative study of biological and artificial neural networks*, European Journal of Applied Engineering and Scientific Research, 2 (2013), pp. 36–46.

[64] H. B. Enderton, *Una introducción matemática a la lógica*, no. 164 E533U., Universidad Nacional Autónoma de México - Instituto de Investigaciones Filológicas, 2004.

[65] E. A. Enriquez, N. Gordillo, L. M. Bergasa, E. Romera, and C. G. Huélamo, *Convolutional neural network vs traditional methods for offline recognition of handwritten digits*, in Workshop of Physical Agents, Springer, 2018, pp. 87–99.

[66] P. S. Eriksson, E. Perfilieva, T. Björk-Eriksson, A.-M. Alborn, C. Nordborg, D. A. Peterson, and F. H. Gage, *Neurogenesis in the adult human hippocampus*, Nature Medicine, 4 (1998), pp. 1313–1317.

[67] R. A. Española and E. Madrid, *Diccionario de la Lengua Española*, vol. 19, Espasa-Calpe, 1970.

[68] M. Ettaouil, M. Lazaar, K. Elmoutaouakil, and K. Haddouch, *A new algorithm for optimization of the kohonen network architectures using the continuous hopfield networks*, wseas transactions on computers, 12 (2013).

[69] K. Fukushima, *Analysis of the process of visual pattern recognition by the neocognitron*, Neural Networks, 2 (1989), pp. 413–420.

[70] K. Garain, U. Kumar, and P. S. Mandal, *Global dynamics in a beddington–deangelis prey–predator model with density dependent death rate of predator*, Differential Equations and Dynamical Systems, (2019), pp. 1–19.

[71] A. Garm and D.-E. Nilsson, *Visual navigation in starfish: first evidence for the use of vision and eyes in starfish*, Proceedings of the Royal Society B: Biological Sciences, 281 (2014), p. 20133011.

[72] M. S. Gazzaniga, R. B. Ivry, and G. R. Mangun, *Cognitive Neuroscience. The biology of the mind*, W. W. Norton & Company, 2019.

[73] M. A. Goodale, A. D. Milner, et al., *Separate visual pathways for perception and action*, (1992).

[74] A. Gupta, Y. Verma, and C. Jawahar, *Choosing linguistics over vision to describe images*, in Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

[75] B. Gustafsson, H. Wigstrom, W. Abraham, and Y. Huang, *Long-term potentiation in the hippocampus using depolarizing current pulses as the conditioning stimulus to single volley synaptic potentials*, Journal of Neuroscience, 7 (1987), pp. 774–780.

[76] N. M. Gyöngyössy, M. Domonkos, J. Botzheim, and P. Korondi, *Supervised learning with small training set for gesture recognition by spiking neural networks*, in 2019 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2019, pp. 2201–2206.

[77] K. V. Haak and C. F. Beckmann, *Objective analysis of the topological organization of the human cortical visual connectome suggests three visual pathways*, Cortex, 98 (2018), pp. 73–83.

[78] J. Haddadnia, K. Faez, and P. Moallem, *Neural network based face recognition with moment invariants*, in Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205), vol. 1, IEEE, 2001, pp. 1018–1021.

[79] V. Hadžiabdić, M. Mehuljić, and J. Bektešević, *Lotka–volterra model with two predators and their prey*, Tem Journal, 6 (2017), pp. 132–136.

[80] T. Hafting, M. Fyhn, S. Molden, M.-B. Moser, and E. I. Moser, *Microstructure of a spatial map in the entorhinal cortex*, Nature, 436 (2005), pp. 801–806.

[81] R. H. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, *Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit*, Nature, 405 (2000), pp. 947–951.

[82] A. H. Hamamoto, L. F. Carvalho, L. D. H. Sampaio, T. Abrão, and M. L. Proença Jr, *Network anomaly detection system using genetic algorithm and fuzzy logic*, Expert Systems with Applications, 92 (2018), pp. 390–402.

[83] M. Hasuike, Y. Yamane, H. Tamura, and K. Sakai, *Representation of local figure-ground by a group of v4 cells*, in International Conference on Neural Information Processing, Springer, 2016, pp. 131–137.

[84] E. Hazan, A. Rakhlin, and P. L. Bartlett, *Adaptive online gradient descent*, in Advances in Neural Information Processing Systems, 2008, pp. 65–72.

[85] J. He, K. Yamada, and T. Nabeshima, *A role of fos expression in the ca3 region of the hippocampus in spatial memory formation in rats*, Neuropsychopharmacology, 26 (2002), pp. 259–268.

[86] K. He, X. Zhang, S. Ren, and J. Sun, *Delving deep into rectifiers: Surpassing human-level performance on imagenet classification*, in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.

[87] ——, *Deep residual learning for image recognition*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[88] ——, *Identity mappings in deep residual networks*, in European Conference on Computer Vision, Springer, 2016, pp. 630–645.

[89] D. O. Hebb, *The Organization of Behavior: a Neuropsychological Theory*, J. Wiley; Chapman & Hall, 1949.

[90] J. Hegdé and D. C. Van Essen, *Selectivity for complex shapes in primate visual area v2*, Journal of Neuroscience, 20 (2000), pp. RC61–RC61.

[91] ——, *A comparative study of shape representation in macaque visual areas v2 and v4*, Cerebral Cortex, 17 (2007), pp. 1100–1116.

[92] D. J. Hemanth, C. K. S. Vijila, A. I. Selvakumar, and J. Anitha, *Performance enhanced hybrid kohonen-hopfield neural network for abnormal brain image classification*, in International Conference on Signal Processing, Image Processing, and Pattern Recognition, Springer, 2011, pp. 356–365.

[93] J. D. O. Hernández, E. J. Aguilar, and F. G. García, *El hipocampo: neurogénesis y aprendizaje*, Rev Med UV, (2015), pp. 21–28.

[94] T. Hige, Y. Aso, M. N. Modi, G. M. Rubin, and G. C. Turner, *Heterosynaptic plasticity underlies aversive olfactory learning in drosophila*, Neuron, 88 (2015), pp. 985–998.

[95] F. L. Hitti and S. A. Siegelbaum, *The hippocampal ca2 region is essential for social memory*, Nature, 508 (2014), pp. 88–92.

[96] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural Computation, 9 (1997), pp. 1735–1780.

[97] A. L. Hodgkin and A. F. Huxley, *A quantitative description of membrane current and its application to conduction and excitation in nerve*, The Journal of physiology, 117 (1952), p. 500.

[98] R. Holca-Lamarre, J. Lücke, and K. Obermayer, *Models of acetylcholine and dopamine signals differentially improve neural representations*, Frontiers in Computational Neuroscience, 11 (2017), p. 54.

[99] J. H. Holland, *Adaptation in natural and artificial systems*, 1976.

[100] J. H. Holland et al., *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, MIT press, 1992.

[101] J. J. Hopfield, *Neural networks and physical systems with emergent collective computational abilities*, Proceedings of the National Academy of Sciences, 79 (1982), pp. 2554–2558.

[102] J. J. Hopfield and D. W. Tank, *'neural'computation of decisions in optimization problems*, Biological Cybernetics, 52 (1985), pp. 141–152.

[103] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, *Mobilenets: Efficient convolutional neural networks for mobile vision applications*, arXiv preprint arXiv:1704.04861, (2017).

[104] M.-K. Hu, *Visual pattern recognition by moment invariants*, IRE transactions on information theory, 8 (1962), pp. 179–187.

[105] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, *Densely connected convolutional networks*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.

[106] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, *Extreme learning machine: Theory and applications*, Neurocomputing, 70 (2006), pp. 489–501.

[107] S. Huang, C. Rozas, M. Trevino, J. Contreras, S. Yang, L. Song, T. Yoshioka, H.-K. Lee, and A. Kirkwood, *Associative hebbian synaptic plasticity in primate visual cortex*, Journal of Neuroscience, 34 (2014), pp. 7575–7579.

[108] D. Hubel and T. Wiesel, *Receptive fields of optic nerve fibres in the spider monkey*, The Journal of Physiology, 154 (1960), pp. 572–580.

[109] D. Hubel and T. Wiesel, *David hubel and torsten wiesel*, Neuron, 75 (2012), pp. 182–184.

[110] D. H. Hubel and T. N. Wiesel, *Receptive fields of single neurones in the cat's striate cortex*, The Journal of Physiology, 148 (1959), p. 574.

[111] ——, *Integrative action in the cat's lateral geniculate body*, The Journal of Physiology, 155 (1961), p. 385.

[112] ——, *Receptive fields, binocular interaction and functional architecture in the cat's visual cortex*, The Journal of Physiology, 160 (1962), p. 106.

[113] U. J. Ilg, *The role of areas mt and mst in coding of visual motion underlying the execution of smooth pursuit*, Vision research, 48 (2008), pp. 2062–2069.

[114] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, arXiv preprint arXiv:1502.03167, (2015).

[115] L. A. Isbell, *Snakes as agents of evolutionary change in primate brains*, Journal of Human Evolution, 51 (2006), pp. 1–35.

[116] M. Ito and H. Komatsu, *Representation of angles embedded within contour stimuli in area v2 of macaque monkeys*, Journal of Neuroscience, 24 (2004), pp. 3313–3324.

[117] A. M. Jeffries, N. J. Killian, and J. S. Pezaris, *Mapping the primate lateral geniculate nucleus: a review of experiments and methods*, Journal of Physiology-Paris, 108 (2014), pp. 3–10.

[118] J. P. Jones and L. A. Palmer, *An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex*, Journal of neurophysiology, 58 (1987), pp. 1233–1258.

[119] M. I. Jordan, *Attractor dynamics and parallelism in a connectionist sequential machine*, in Artificial Neural Networks: concept learning, 1990, pp. 112–127.

[120] H. Kabir, M. Abdar, S. M. J. Jalali, A. Khosravi, A. F. Atiya, S. Nahavandi, and D. Srinivasan, *Spinalnet: Deep neural network with gradual input*, arXiv preprint arXiv:2007.03347, (2020).

[121] A. Karpathy and L. Fei-Fei, *Deep visual-semantic alignments for generating image descriptions*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3128–3137.

[122] C. Keck, C. Savin, and J. Lücke, *Feedforward inhibition and synaptic scaling - two sides of the same coin?*, PLoS Comput Biol, 8 (2012), p. e1002432.

[123] S. Kesavan, *Measure and Integration*, Springer, 2019.

[124] J. Kim, *Philosophy of mind*, Routledge, 2018.

[125] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in 3rd International Conference for Learning Representations, San Diego, 2015.

[126] R. Kiros, R. Salakhutdinov, and R. S. Zemel, *Unifying visual-semantic embeddings with multimodal neural language models*, arXiv preprint arXiv:1411.2539, (2014).

[127] A. Klein, S. Falkner, S. Bartels, P. Hennig, and F. Hutter, *Fast bayesian optimization of machine learning hyperparameters on large datasets*, in Artificial Intelligence and Statistics, 2017, pp. 528–536.

[128] A. L. Koerich, *Unconstrained handwritten character recognition using different classification strategies*, in International Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR), 2003.

[129] T. Kohonen, *Self-organized formation of topologically correct feature maps*, Biological Cybernetics, 43 (1982), pp. 59–69.

[130] E. P. Kolpak and E. V. Gorynya, *Mathematical models of ecological niches search*, Applied Mathematical Sciences, 10 (2016), pp. 1907–1921.

[131] B. J. Kraus, R. J. Robinson II, J. A. White, H. Eichenbaum, and M. E. Hasselmo, *Hippocampal 'time cells': time versus path integration*, Neuron, 78 (2013), pp. 1090–1101.

[132] G. Kreiman, C. Koch, and I. Fried, *Category-specific visual responses of single neurons in the human medial temporal lobe*, Nature Neuroscience, 3 (2000), pp. 946–953.

[133] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, in Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[134] E. Kropff, J. E. Carmichael, M.-B. Moser, and E. I. Moser, *Speed cells in the medial entorhinal cortex*, Nature, 523 (2015), pp. 419–424.

[135] S. W. Kuffler, *Discharge patterns and functional organization of mammalian retina*, Journal of Neurophysiology, 16 (1953), pp. 37–68.

[136] ——, *The single-cell approach in the visual system and the study of receptive fields*, Investigative Ophthalmology & Visual Science, 12 (1973), pp. 794–813.

[137] G. Kulkarni, V. Premraj, V. Ordonez, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg, *Babytalk: Understanding and generating simple image descriptions*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 35 (2013), pp. 2891–2903.

[138] E. Kuriscak, P. Marsalek, J. Stroffek, and P. G. Toth, *Biological context of hebb learning in artificial neural networks, a review*, Neurocomputing, 152 (2015), pp. 27–35.

[139] Ł. Kuśmierz, T. Isomura, and T. Toyoizumi, *Learning with three factors: modulating hebbian plasticity with errors*, Current opinion in Neurobiology, 46 (2017), pp. 170–177.

[140] B. Kwolek, *Face detection using convolutional neural networks and gabor filters*, in International Conference on Artificial Neural Networks, Springer, 2005, pp. 551–556.

[141] G. LaForte, P. J. Hayes, and K. M. Ford, *Why gödel's theorem cannot refute computationalism*, Artificial Intelligence, 104 (1998), pp. 265–286.

[142] M. Land, *Movements of the retinae of jumping spiders (salticidae: Dendryphantinae) in response to visual stimuli*, Journal of Experimental Biology, 51 (1969), pp. 471–493.

[143] C. C. Law and L. N. Cooper, *Formation of receptive fields in realistic visual environments according to the bienenstock, cooper, and munro (bcm) theory*, Proceedings of the National Academy of Sciences, 91 (1994), pp. 7797–7801.

[144] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, *Backpropagation applied to handwritten zip code recognition*, Neural Computation, 1 (1989), pp. 541–551.

[145] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, *Handwritten digit recognition with a back-propagation network*, in Advances in Neural Information Processing Systems, 1990, pp. 396–404.

[146] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE, 86 (1998), pp. 2278–2324.

[147] Y. LeCun, C. Cortes, and C. Burges, *Mnist handwritten digit database.*, URL http://yann. lecun. com/exdb/mnist, 7 (2010), p. 23.

[148] Y. LeCun et al., *Generalization and network design strategies*, Connectionism in perspective, 19 (1989), pp. 143–155.

[149] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, *Multilayer feedforward networks with a nonpolynomial activation function can approximate any function*, Neural networks, 6 (1993), pp. 861–867.

[150] J. Y. Lettvin, H. R. Maturana, W. S. McCulloch, and W. H. Pitts, *What the frog's eye tells the frog's brain*, Proceedings of the IRE, 47 (1959), pp. 1940–1951.

[151] G. Li, M. Liu, and M. Dong, *A new online learning algorithm for structure-adjustable extreme learning machine*, Computers & Mathematics with Applications, 60 (2010), pp. 377–389.

[152] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, *A convolutional neural network cascade for face detection*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5325–5334.

[153] T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, and G. Hinton, *Backpropagation and the brain*, Nature Reviews Neuroscience, (2020), pp. 1–12.

[154] M. Lin, Q. Chen, and S. Yan, *Network in network*, arXiv preprint arXiv:1312.4400, (2013).

[155] W. A. Little, *The existence of persistent states in the brain*, Mathematical Biosciences, 19 (1974), pp. 101–120.

[156] C. Liu and F. Sun, *Hmax model: A survey*, in 2015 International Joint Conference on Neural Networks (IJCNN), IEEE, 2015, pp. 1–7.

[157] D. Liu and S. Yue, *Visual pattern recognition using unsupervised spike timing dependent plasticity learning*, in 2016 International Joint Conference on Neural Networks (IJCNN), IEEE, 2016, pp. 285–292.

[158] Q. Liu, Y. Chen, and S. Furber, *Noisy softplus: an activation function that enables snns to be trained as anns*, arXiv preprint arXiv:1706.03609, (2017).

[159] N. K. Logothetis and D. L. Sheinberg, *Visual object recognition*, Annual Review of Neuroscience, 19 (1996), pp. 577–621.

[160] T. Lømo, *Frequency potentiation of excitatory synaptic activity in dentate area of hippocampal formation*, in Acta Physiologica Scandinavica, BLACKWELL SCIENCE LTD PO BOX 88, OSNEY MEAD, OXFORD OX2 0NE, OXON, ENGLAND, 1966, p. 128.

[161] J. Long, E. Shelhamer, and T. Darrell, *Fully convolutional networks for semantic segmentation*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[162] Z. Lu, I. Whalen, V. Boddeti, Y. Dhebar, K. Deb, E. Goodman, and W. Banzhaf, *Nsga-net: Neural architecture search using multi-objective genetic algorithm*, in Proceedings of the Genetic and Evolutionary Computation Conference, 2019, pp. 419–427.

[163] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, *Learning word vectors for sentiment analysis*, in Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 2011, pp. 142–150.

[164] A. L. Maas, A. Y. Hannun, and A. Y. Ng, *Rectifier nonlinearities improve neural network acoustic models*, in Proc. ICML, vol. 30, 2013, p. 3.

[165] C. J. MacDonald, K. Q. Lepage, U. T. Eden, and H. Eichenbaum, *Hippocampal 'time cells'bridge the gap in memory for discontiguous events*, Neuron, 71 (2011), pp. 737–749.

[166] D. Maclaurin, D. Duvenaud, and R. Adams, *Gradient-based hyperparameter optimization through reversible learning*, in International Conference on Machine Learning, 2015, pp. 2113–2122.

[167] S. Madireddy, A. Yanguas-Gil, and P. Balaprakash, *Multilayer neuromodulated architectures for memory-constrained online continual learning*, arXiv preprint arXiv:2007.08159, (2020).

[168] A. Magotra and J. kim, *Transfer learning for image classification using hebbian plasticity principles*, in Proceedings of the 2019 3rd International Conference on Computer Science and Artificial Intelligence, 2019, pp. 233–238.

[169] A. Mahendran and A. Vedaldi, *Visualizing deep convolutional neural networks using natural pre-images*, International Journal of Computer Vision, 120 (2016), pp. 233–255.

[170] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille, *Deep captioning with multimodal recurrent neural networks (m-rnn)*, arXiv preprint arXiv:1412.6632, (2014).

[171] H. Markram, *The blue brain project*, Nature Reviews Neuroscience, 7 (2006), pp. 153–160.

[172] D. Marr, *Vision: A computational investigation into the human representation and processing of visual information*, Inc., New York, NY, 2 (1982).

[173] W. S. McCulloch and W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, The Bulletin of Mathematical Biophysics, 5 (1943), pp. 115–133.

[174] D. McCullough, *Can humans escape gödel?*, Psyche, 2 (1995).

[175] E. H. Meijering, K. J. Zuiderveld, and M. A. Viergever, *Image reconstruction by convolution with symmetrical piecewise nth-order polynomial kernels*, IEEE Transactions on Image Processing, 8 (1999), pp. 192–201.

[176] W. H. MERIGAN and H. A. Pham, *V4 lesions in macaques affect both single-and multiple-viewpoint shape discriminations*, Visual Neuroscience, 15 (1998), pp. 359–367.

[177] N. Metawa, M. K. Hassan, and M. Elhoseny, *Genetic algorithm based model for optimizing bank lending decisions*, Expert Systems with Applications, 80 (2017), pp. 75–82.

[178] R. Miikkulainen, J. A. Bednar, Y. Choe, and J. Sirosh, *Computational Maps in the Visual Cortex*, Springer Science & Business Media, 2006.

[179] T. D. Miller, T. T. Chong, A. M. A. Davies, M. R. Johnson, S. R. Irani, M. Husain, T. W. Ng, S. Jacob, P. Maddison, C. Kennard, et al., *Human hippocampal ca3 damage disrupts both recent and remote episodic memories*, Elife, 9 (2020), p. e41836.

[180] M. Mitchell, X. Han, J. Dodge, A. Mensch, A. Goyal, A. Berg, K. Yamaguchi, T. Berg, K. Stratos, and H. Daumé III, *Midge: Generating image descriptions from computer vision detections*, in Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, 2012, pp. 747–756.

[181] Y. Munakata and J. Pfaffly, *Hebbian learning and development*, Developmental Science, 7 (2004), pp. 141–148.

[182] P. Murugan, *Implementation of deep convolutional neural network in multi-class categorical image classification*, arXiv preprint arXiv:1801.01397, (2018).

[183] V. Nair and G. E. Hinton, *Rectified linear units improve restricted boltzmann machines*, in ICML, 2010.

[184] Y. Nesterov, *A method of solving a convex programming problem with convergence rate* $o(1/k^2$, in Soviet Mathematics Doklady, vol. 27, 1983.

[185] R. A. Nicoll, *A brief history of long-term potentiation*, Neuron, 93 (2017), pp. 281–290.

[186] D.-E. Nilsson, *Eye evolution and its functional basis*, Visual Neuroscience, 30 (2013), pp. 5–20.

[187] J. Nocedal and S. Wright, *Numerical optimization*, Springer Science & Business Media, 2006.

[188] H. Noh, S. Hong, and B. Han, *Learning deconvolution network for semantic segmentation*, in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1520–1528.

[189] E. Oja, *A simplified neuron model as a principal component analyzer*, Journal of Mathematical Biology, 15 (1982), pp. 267–273.

[190] G. Okazawa, S. Tajima, and H. Komatsu, *Gradual development of visual texture-selective properties between macaque areas v2 and v4*, Cerebral Cortex, 27 (2017), pp. 4867–4880.

[191] J. O'Keefe and J. Dostrovsky, *The hippocampus as a spatial map: Preliminary evidence from unit activity in the freely-moving rat.*, Brain research, (1971).

[192] D. B. Omer, S. R. Maimon, L. Las, and N. Ulanovsky, *Social place-cells in the bat hippocampus*, Science, 359 (2018), pp. 218–224.

[193] V. Ordonez, G. Kulkarni, and T. L. Berg, *Im2text: Describing images using 1 million captioned photographs*, in Advances in neural information processing systems, 2011, pp. 1143–1151.

[194] J.-Y. Pan, H.-J. Yang, P. Duygulu, and C. Faloutsos, *Automatic image captioning*, in 2004 IEEE International Conference on Multimedia and Expo (ICME)(IEEE Cat. No. 04TH8763), vol. 3, IEEE, 2004, pp. 1987–1990.

[195] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu, *Gcap: Graph-based automatic image captioning*, in 2004 Conference on Computer Vision and Pattern Recognition Workshop, IEEE, 2004, pp. 146–146.

[196] A. Pasupathy and C. E. Connor, *Population coding of shape in area v4*, Nature neuroscience, 5 (2002), pp. 1332–1338.

[197] G. H. Patel, D. M. Kaplan, and L. H. Snyder, *Topographic organization in the brain: searching for general principles*, Trends in Cognitive Sciences, 18 (2014), pp. 351–363.

[198] M. V. Peelen and A. Caramazza, *Conceptual object representations in human anterior temporal cortex*, Journal of Neuroscience, 32 (2012), pp. 15728–15736.

[199] Y. Peng and H. Yin, *Markov random field based convolutional neural networks for image classification*, in International Conference on Intelligent Data Engineering and Automated Learning, Springer, 2017, pp. 387–396.

[200] R. Penrose, *Shadows of the Mind*, vol. 4, Oxford University Press Oxford, 1994.

[201] R. Penrose and N. D. Mermin, *The emperor's new mind: Concerning computers, minds, and the laws of physics*, 1990.

[202] M. d. C. R. Pita, *Análisis de la actividad celular en la corteza inferotemporal, la amígdala, el caudado y el putamen, durante la realización de una tarea visuomotora.*, Univ Santiago de Compostela, 2008.

[203] A. Plebe and M. Vivian, *Neurosemantics: Neural Processes and the Construction of Linguistic Meaning*, vol. 10, Springer, 2016.

[204] A. B. Porto-Pazos, N. Veiguela, P. Mesejo, M. Navarrete, A. Alvarellos, O. Ibáñez, A. Pazos, and A. Araque, *Artificial astrocytes improve neural network performance*, PloS one, 6 (2011), p. e19109.

[205] R. Potthast, *Amari Model.*, Springer Publishing Company, Incorporated, 2015.

[206] D. Purves, G. J. Augustine, D. Fitzpatrick, W. C. Hall, A. S. Lamantia, J. O. McNamara, and S. M. Williams, *Neurociencia*, Sinauer Associates, 2004.

[207] R. Q. Quiroga, A. Kraskov, C. Koch, and I. Fried, *Explicit encoding of multimodal percepts by single neurons in the human brain*, Current Biology, 19 (2009), pp. 1308–1313.

[208] R. Q. Quiroga, G. Kreiman, C. Koch, and I. Fried, *Sparse but not 'grandmother-cell'coding in the medial temporal lobe*, Trends in Cognitive Sciences, 12 (2008), pp. 87–91.

[209] R. Q. Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried, *Invariant visual representation by single neurons in the human brain*, Nature, 435 (2005), pp. 1102–1107.

[210] P. V. Radtke, R. Sabourin, and T. Wong, *Using the rrt algorithm to optimize classification systems for handwritten digits and letters*, in Proceedings of the 2008 ACM Symposium on Applied Computing, 2008, pp. 1748–1752.

[211] E. Reiter and R. Dale, *Building applied natural language generation systems*, Natural Language Engineering, 3 (1997), pp. 57–87.

[212] M. Riesenhuber and T. Poggio, *Hierarchical models of object recognition in cortex*, Nature Neuroscience, 2 (1999), pp. 1019–1025.

[213] ——, *Models of object recognition*, Nature neuroscience, 3 (2000), pp. 1199–1204.

[214] F. Rosenblatt, *The perceptron: a probabilistic model for information storage and organization in the brain.*, Psychological Review, 65 (1958), p. 386.

[215] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., *Imagenet large scale visual recognition challenge*, International Journal of Computer Vision, 115 (2015), pp. 211–252.

[216] S. J. Russell and P. Norvig, *Inteligencia Artificial: un enfoque moderno*, no. 04; Q335, R8y 2004., 2004.

[217] D. Sahoo, Q. Pham, J. Lu, and S. C. Hoi, *Online deep learning: Learning deep neural networks on the fly*, IJCAI, (2017).

[218] H. Sanders, C. Rennó-Costa, M. Idiart, and J. Lisman, *Grid cells and place cells: an integrated view of their navigational and memory function*, Trends in neurosciences, 38 (2015), pp. 763–775.

[219] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, *Mobilenetv2: Inverted residuals and linear bottlenecks*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520.

[220] K. S. Sasaki, Y. Tabuchi, and I. Ohzawa, *Complex cells in the cat striate cortex have multiple disparity detectors in the three-dimensional binocular receptive fields*, Journal of Neuroscience, 30 (2010), pp. 13826–13837.

[221] D. Scherer, A. Müller, and S. Behnke, *Evaluation of pooling operations in convolutional architectures for object recognition*, in International Conference on Artificial Neural Networks, Springer, 2010, pp. 92–101.

[222] R. Scherer, *Computer vision methods for fast Image Classification and retrieval*, Springer, 2020.

[223] T. J. Sejnowski and G. Tesauro, *Building network learning algorithms from hebbian synapses*, Brain Organization and Memory: Cells, Systems and Circuits. Oxford University Press, New York, (1989), pp. 338–355.

[224] A. Serov, *Subjective reality and strong artificial intelligence*, arXiv preprint arXiv:1301.6359, (2013).

[225] T. Serre, *Hierarchical Models of the Visual System.*, Springer Publishing Company, Incorporated, 2015.

[226] A. Sharma, D. K. Kumar, S. Kumar, and N. McLachlan, *Recognition of human actions using moment based features and artificial neural networks*, in 10th International Multimedia Modelling Conference, 2004. Proceedings., IEEE, 2004, p. 368.

[227] H. G. Shim, D. C. Jang, J. Lee, G. Chung, S. Lee, Y. G. Kim, S. J. Kim, et al., *Long-term depression of intrinsic excitability accompanied by synaptic depression in cerebellar purkinje cells*, Journal of Neuroscience, 37 (2017), pp. 5659–5669.

[228] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, arXiv preprint arXiv:1409.1556, (2014).

[229] S. Skansi, *Introduction to Deep Learning: from Logical Calculus to Artificial Intelligence*, Springer, 2018.

[230] R. Socher, A. Karpathy, Q. V. Le, C. D. Manning, and A. Y. Ng, *Grounded compositional semantics for finding and describing images with sentences*, Transactions of the Association for Computational Linguistics, 2 (2014), pp. 207–218.

[231] T. Solstad, C. N. Boccara, E. Kropff, M.-B. Moser, and E. I. Moser, *Representation of geometric borders in the entorhinal cortex*, Science, 322 (2008), pp. 1865–1868.

[232] L. Song, Y. Zhang, Z. Wang, and D. Gildea, *A graph-to-sequence model for amr-to-text generation*, arXiv preprint arXiv:1805.02473, (2018).

[233] N. Spruston and W. L. Kath, *Dendritic arithmetic*, Nature neuroscience, 7 (2004), pp. 567–569.

[234] S. M. Srivastava, *A course on mathematical logic*, Springer Science & Business Media, 2013.

[235] T. Stewart, F.-X. Choo, and C. Eliasmith, *Spaun: A perception-cognition-action model using spiking neurons*, in Proceedings of the Annual Meeting of the Cognitive Science Society, vol. 34, 2012.

[236] C. Stinson and J. Sullivan, *Mechanistic explanation in neuroscience*, The Routledge Handbook of Mecahnisms and Mechanical Philosophy, (2018), pp. 375–387.

[237] T. Sugihara, M. D. Diltz, B. B. Averbeck, and L. M. Romanski, *Integration of auditory and visual communication information in the primate ventrolateral prefrontal cortex*, Journal of Neuroscience, 26 (2006), pp. 11138–11147.

[238] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, *Inception-v4, inception-resnet and the impact of residual connections on learning*, Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), (2016).

[239] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, *Going deeper with convolutions*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

[240] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, *Rethinking the inception architecture for computer vision*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.

[241] M. Tanti, A. Gatt, and K. P. Camilleri, *What is the role of recurrent neural networks (rnns) in an image caption generator?*, arXiv preprint arXiv:1708.02043, (2017).

[242] J. S. Taube, R. U. Muller, and J. B. Ranck, *Head-direction cells recorded from the postsubiculum in freely moving rats. i. description and quantitative analysis*, Journal of Neuroscience, 10 (1990), pp. 420–435.

[243] C. Theriault, N. Thome, and M. Cord, *Extended coding and pooling in the hmax model*, IEEE Transactions on Image Processing, 22 (2012), pp. 764–777.

[244] T. Tieleman and G. Hinton, *Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude*, COURSERA: Neural networks for Machine Learning, 4 (2012), pp. 26–31.

[245] T. Trappenberg, *Fundamentals of computational neuroscience, ser. fundamentals of computational neuroscience*, 2010.

[246] A. Trivedi, S. Srivastava, A. Mishra, A. Shukla, and R. Tiwari, *Hybrid evolutionary approach for devanagari handwritten numeral recognition using convolutional neural network*, Procedia Computer Science, 125 (2018), pp. 525–532.

[247] A. Turing, *Computing machinery and intelligence*, Mind, 59 (1950), p. 433.

[248] N. Tzakis and M. R. Holahan, *Social memory and the role of the hippocampal ca2 region*, Frontiers in Behavioral Neuroscience, 13 (2019).

[249] A. van Schaik and J. Tapson, *Online and adaptive pseudoinverse solutions for elm weights*, Neurocomputing, 149 (2015), pp. 233–238.

[250] A. L. F. Velazquez, *Chateando con mitsuku*, Revista Digital Universitaria, 21 (2020).

[251] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, *Show and tell: Lessons learned from the 2015 mscoco image captioning challenge*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 39 (2016), pp. 652–663.

[252] A. Wadhwa and U. Madhow, *Bottom-up deep learning using the hebbian principle*, 2016.

[253] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, *Understanding convolution for semantic segmentation*, in 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2018, pp. 1451–1460.

[254] B. Widrow and M. A. Lehr, *Perceptrons, adalines, and backpropagation*, Arbib, 4 (1995), pp. 719–724.

[255] J. T. Wixted, S. D. Goldinger, L. R. Squire, J. R. Kuhn, M. H. Papesh, K. A. Smith, D. M. Treiman, and P. N. Steinmetz, *Coding of episodic memory in the human hippocampus*, Proceedings of the National Academy of Sciences, 115 (2018), pp. 1093–1098.

[256] R. H. Wurtz, *Visual receptive fields of striate cortex neurons in awake monkeys.*, Journal of Neurophysiology, 32 (1969), pp. 727–742.

[257] J. Xin and M. J. Embrechts, *Supervised learning with spiking neural networks*, in IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222), vol. 3, IEEE, 2001, pp. 1772–1777.

[258] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, *Show, attend and tell: Neural image caption generation with visual attention*, in International Conference on Machine Learning, 2015, pp. 2048–2057.

[259] S. Yaghini Bonabi, H. Asgharian, S. Safari, and M. Nili Ahmadabadi, *Fpga implementation of a biological neural network based on the hodgkin-huxley neuron model*, Frontiers in neuroscience, 8 (2014), p. 379.

[260] Y. Yamane, A. Kodama, M. Shishikura, K. Kimura, H. Tamura, and K. Sakai, *Population coding of figure and ground in natural image patches by v4 neurons*, Plos one, 15 (2020), p. e0235128.

[261] A. Yenter and A. Verma, *Deep cnn-lstm with combined kernels from multiple branches for imdb review sentiment analysis*, in 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), IEEE, 2017, pp. 540–546.

[262] C. Yoshida-Noro, M. Myohara, F. Kobari, and S. Tochinai, *Nervous system dynamics during fragmentation and regeneration in enchytraeus japonensis (oligochaeta, annelida)*, Development Genes and Evolution, 210 (2000), pp. 311–319.

[263] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, *Image captioning with semantic attention*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4651–4659.

[264] M. P. Young, *Objective analysis of the topological organization of the primate cortical visual system*, Nature, 358 (1992), pp. 152–155.

[265] S. R. Young, D. C. Rose, T. P. Karnowski, S.-H. Lim, and R. M. Patton, *Optimizing deep learning hyper-parameters through an evolutionary algorithm*, in

Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments, 2015, pp. 1–5.

[266] M. D. Zeiler and R. Fergus, *Stochastic pooling for regularization of deep convolutional neural networks*, arXiv preprint arXiv:1301.3557, (2013).

[267] S. Zekovich and M. Tuba, *Hu moments based handwritten digits recognition algorithm*, Recente Advances in Knowledge Engineering and Systems Science, (2013).

[268] C. Zhang and Z. Zhang, *Improving multiview face detection with multi-task deep convolutional neural networks*, in IEEE Winter Conference on Applications of Computer Vision, IEEE, 2014, pp. 1036–1041.

[269] H.-Z. Zhang, Y.-F. Lu, T.-K. Kang, and M.-T. Lim, *B-hmax: A fast binary biologically inspired model for object recognition*, Neurocomputing, 218 (2016), pp. 242–250.

[270] L. I. Zhang, H. W. Tao, C. E. Holt, W. A. Harris, and M.-m. Poo, *A critical window for cooperation and competition among developing retinotectal synapses*, Nature, 395 (1998), pp. 37–44.

[271] C. M. Ziemba, J. Freeman, J. A. Movshon, and E. P. Simoncelli, *Selectivity and tolerance for visual texture in macaque v2*, Proceedings of the National Academy of Sciences, 113 (2016), pp. E3140–E3149.

[272] M. Zinkevich, *Online convex programming and generalized infinitesimal gradient ascent*, in Proceedings of the 20th International Conference on Machine Learning (icml-03), 2003, pp. 928–936.