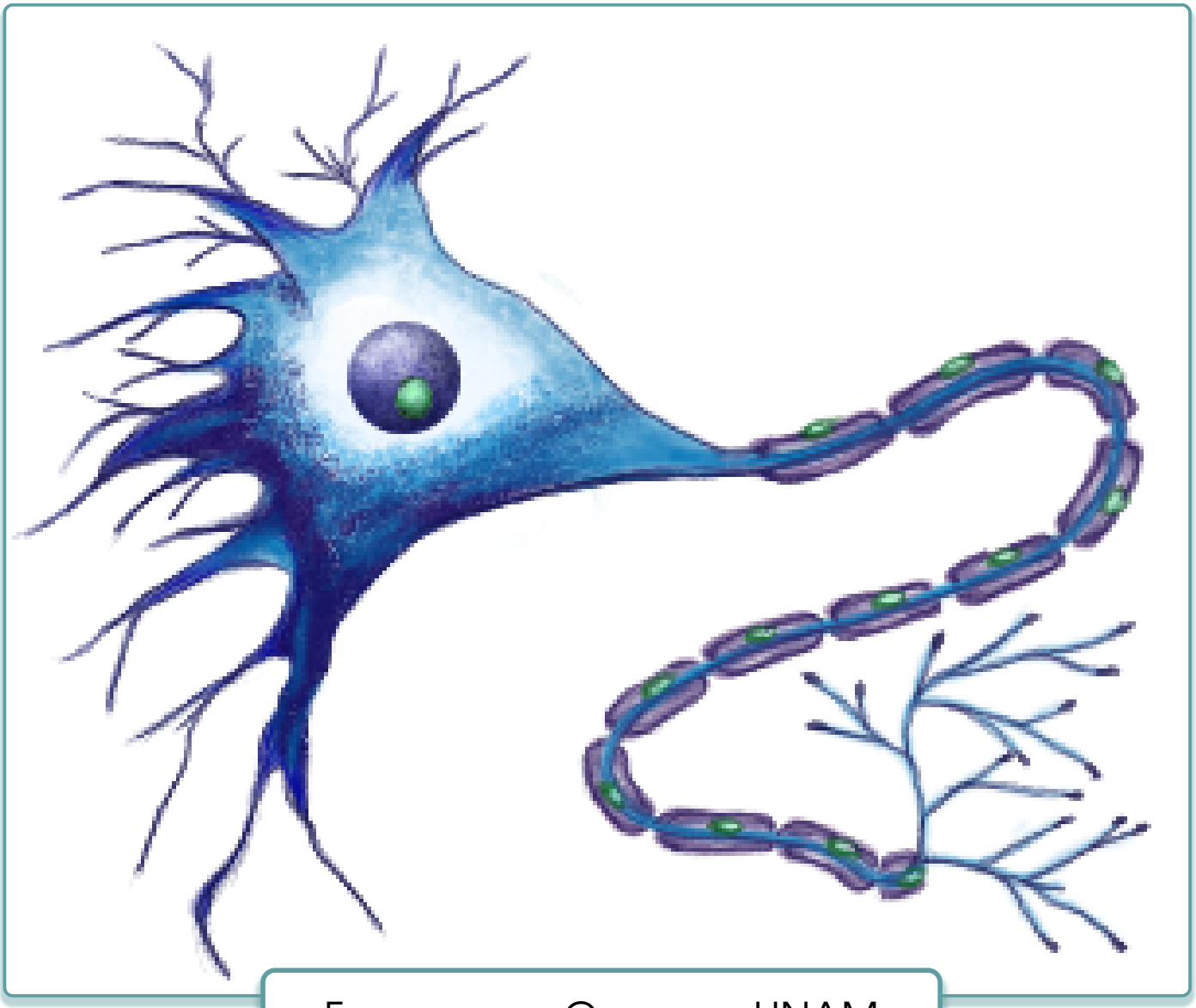
Redes Neuronales

Notas de clase

Karla Fernanda Jiménez Gutiérrez
Verónica Esther Arriola Ríos



Facultad de Ciencias, UNAM

Índice general

ÍNDICE GENERAL

Etc

A lo largo del texto se utilizará la siguiente notación para diversos elementos:

Conjuntos	C
Vectores	$\boldsymbol{x}$
Matrices	$\boldsymbol{M}$
Unidades	cm

Parte I
Antecedentes

1 | Neurona biológica

Neurociencias computacionales

El campo conocido como **neurociencias computacionales**Trappenberg 2010 es el que se dedica explícitamente al estudio/modelado de los sistemas biológicos con ayuda de varios campos de estudio. Se interesa notablemente en descripciones y modelos funcionales biológicamente realistas de neuronas y sistemas neuronales.

Los campos de estudio de los cuales nos ayudamos son:

- **Ciencias cognitivas** dedicadas a tratar directamente con los humanos, de estás tomamos los conceptos de aprendizaje.
- **Biofísica** por el estudio de las propiedades físicas en de los sistemas biológicos.
- **Neurociencias tradicionales** con modelos matemáticos.
- **Ciencias de la computación** modelar e implementar los modelos dados, por las aréas aquí listadas. Simulaciones computacionales.
- **Ingeniería eléctrica** diseño de hardware especifico y eficiente para tomar los pulsos y medidas exactas.

Las neurociencias computacionales, estudian modelos del sistema nervioso y clasifican estos modelos en tres tipos, recordemos el método cientifico:

1. **Modelos descriptivos**, Describen. Por ejemplo describe el comportamiento de los ratones a ciertas sustancias. La **comparación**, lo que estaba pasando antes de la sustancia, con la sustancia y después.
2. **Modelos mecanistas**, Mecánicamente. El **cómo** paso el evento o la acción. Siguiendo con el ejemplo. El ratón se cayo y se levanto, ¿Cómo? Doblo sus articulaciones, tomo impulso con sus musculos, roto su cuerpo, hasta reicorporarse totalmente.

3. **Modelos interpretativos**, Se interpreta. ¿**Por qué** hizo los movimientos mecanicos?. Siguiendo con el ejemplo. ¿Por qué el ratón se reincorporo? Porqué el ratón quiere regresar a su estado anterior antes de la pertubación por la sustancia. ¿Sera verdad? Es lo que se interpreta, no la verdad absoluta. Se tiene que buscar intencionalidad, razonamiento de más alto nivel.

Los **objetivos del modelado** en neurociencias de nuestro interes en el curso son; Las corrientes, las proteínas, las oscilaciones de las redes completa, la arquitectura topografica y de columnas, el aprendizaje y la memoria. Por lo menos para el curso de Redes Neuronales en la Universidad Nacional Autonoma de México con la doctora Verónica Esther Arriola Ríos.

Para más información en Neurociencias Kandel 2012, siempre está el interes propio.

Las redes neuronales artificiales (RNA), estan inspiradas en las redes neuronales biológicas, tales como las de los animales Francisco López-Muñoz 2006, en un primer momento. Hasta llegar al ser humano con el sistema nervioso y nuestras neuronas. Waldeyer-Hartz 1891.

Los modelos de redes neuronales que tomaremos, son simples. Distan mucho de los sistemas naturales y aun así dan solución. Lo que nos interesa es que resuelvan los problemas inmediatos y de corto plazo. Si el cerebro humano funciona, la imitación debe darnos algunos resultados, idea que ha funcionado para procesar información y dar solución con modelos diseñados.

Los problemas más notorios a resolver son:

- Problemas de visión por computadora.
- Procesamiento del lenguaje natural.

Sistema Nervioso

El sistema nervioso se estudia principalmente en dos partes, sistema nervioso periférico (SNP), y sistema nervioso central (SNC). SNC es donde nos enfocaremos, se compone principalmente por la medula espinal y el encéfalo. Es en el encéfalo donde se encuentra principalmente el cerebro, cerebelo, y el tallo encefálico.Snell 2001

¿Qué es un nervio?

Un nervio es una estructura conductora de impulsos nerviosos. Esta compuesto por una colección de axones los cuales son prolongaciones largas y delgadas de las células nerviosas agrupados en fibras y rodeados de vasos sanguíneos. Se origina desde la médula espinal o el encéfalo.

¿Qué son los nervios?

En conjunto son la escrutura organizada por tejido conectivo en el sistema nervioso, encargada de transmitir información entre diferentes partes del cuerpo del SNP y del SNC, permitiendo así el control y coordinación eficientes de las actividades corporales. Se extienden desde el cráneo y la médula espinal para abarcar todo el cuerpo humano. Kandel 2012

Estos pueden ser clasificados en:

- **Motores** salidas, ejecución/acción, se conectan y ejercen su acción sobre los músculos.
- **Sensitivos** reciben señales de entrada, como en ojos, oídos y piel.
- **Mixtos** son mayoría, tienen tanto fibras sensitivas como motoras.

Las ideas tomadas del SNC para el desarrollo de las RNA, son:

- La serie de entradas, del mundo exterior hacia los nervios receptores.
- Una o varias salidas/respuestas ante un estímulo.
- Conexiones entre los nervios motores, sensitivos, mixtos y el encéfalo o médula espinal.

Entonces si vemos el sistema nervioso, solo como un sistema que nos permite sentir el mundo con el que interactuamos. A través de los nervios que están en nuestra piel que transmiten la información hasta nuestra médula espinal o cerebelo. Tenemos entonces la primera idea, una serie de entradas, que va a provocar de alguna forma, una o varias respuestas de nuestro cuerpo. Esa forma es una serie de conexiones entre nuestros nervios y el encéfalo o médula espinal. La manera en la que estan hechas estas conexiones (estructura) es la que da pie a las muchas formas de modelar una RNA.

Cerebro

Desde la parte tangible del cerebro hasta la intangible del pensamiento, esta directamente relacionada con la forma que funciona nuestro pensamiento y reacciones motoras.

El cerebro, por su gran tamaño, da lugar a un mejor registro de su actividad debido a la cantidad de sangre que se está bombeando continuamente. Se tienen identificadas regiones que se activan ante cierto estímulo.John G. Nicholls 2011 Se detecta cuánta sangre se está bombeando en diferentes regiones del cerebro dependiendo de los estímulos que se le presentan a una persona; o si alguna persona tiene un padecimiento, se toman

escaneos para ver qué regiones del cerebro están funcionando y cuáles presentan lesiones. A partir de las lesiones y de la identificación de la actividad que ya no se puede realizar de forma normal, se infiere qué región era responsable de esa actividad que ahora está dañada.Doidge 2007

Para obtener más detalle de los estudios de la actividad del cerebro, ver el documental "The Brain with David Eagleman".

Zonas funcionales

En el cerebro, las diversas zonas indentificadas con funciones especificas, estan conectadas, por una ruta que se conoce como "la ruta desde la sensación hasta la cognición" notemos un diagrama de la parte funcional del cerebro. M Marsel Mesulam 1998a

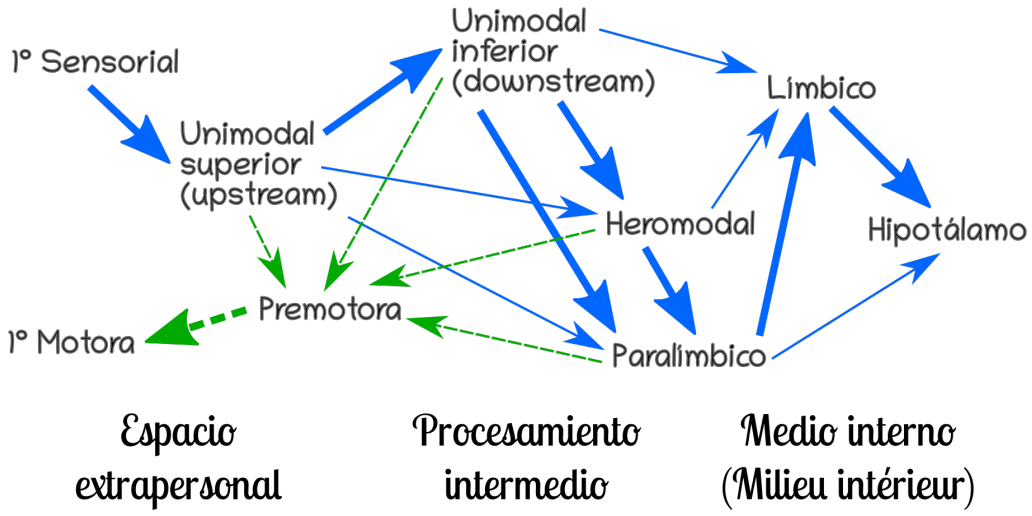


Figura 1.1 Diagrama de la arquitectura del cerebro a nivel funcional. Enfocado en sentido de la visión o la audición. (M. Marsel Mesulam 1998b)

Explicación: El punto de partida de la ruta es la etapa/capa sensorial, que la vamos a conciderar como **la entrada**. El cerebro está recibiendo señales/información en el momento que existe una sensación que viene del mundo exterior.

Su primera conexión es hacia la capa unimodal superior, que la vamos a conciderar como **la primera capa**. Donde se procesa la información más básica de la señal (del sentido), tales como el color, las formas o los tonos de sonido.

Luego están las capas intermedias (unimodal, heromodal, paralimbico, limbico), donde se dará un procesamiento intermedio; se juntarán varias características interpretadas por la primera capa. Conforme avance la señal a las siguientes capas, las neuronas notaran características más complejas.

Finalmente, llegamos al hipotálamo, que con todas las características dadas mandará

a través de las capas intermedias una o varias respuestas, cuya etapa final se dará en la capa premotora, donde las neuronas de esta capa se comunican directamente con la **salida** motora.

Neurona biológica

La neurona

La neurona es un tipo de célula perteneciente al sistema nervioso central, que se comunica tanto por señales eléctricas como por señales químicas. Cada neurona tiene Snell 2009:

- Un cuerpo celular llamado **soma** que contiene un núcleo y otros componentes celulares.
- Una zona de recepción con protuberancias elongadas llamadas **dendritas**.
- Una zona de emisión llamado **axón**, el cual está compuesto de:
  - ⋆ Cono axónico.
  - ⋆ Membrana plasmática axónica y citoplasma.
  - ⋆ Recubrimientos de mielina, interrumpido a intervalos regulares por nodos de Ranvier.
  - ⋆ Terminales del axón donde se encuentan los botones sinápticos.

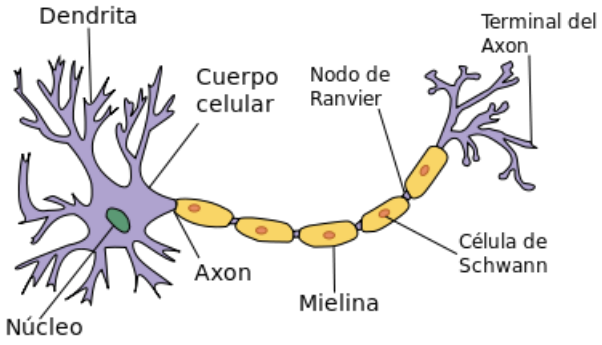


Figura 1.2 Neurona, Acracia, 14 January 2007, Wikimedia Commons, https://commons.wikimedia.org/wiki/File:Neurona.svg, Creative Commons Attribution-ShareAlike 2.5 Generic

Como todas las neuronas pueden trasmitir o recibir información, para denotar claramente en que nos estamos enfocando para una neurona particual es necesario hacer las siguientes ditinciones:

- Neurona presináptica, transmite una señal.
- Neurona postináptica, recibe una señal.

La transmisión de señales y almacenamiento de información en las neuronas se da de la siguiente forma:John G. Nicholls 2011

1. La neurona desde sus dendritas recibe señales de otras neuronas vecinas.
2. Cada señal se va acumulando en su cuerpo hasta el cono axónico, donde se van a estar sumando la contribución de todos los efectos de cambios de potencial.
3. En el momento que se rebase un cierto valor umbral, la diferencia de potencial se propaga hasta los botones terminales.
4. La neurona entra en un período refractario, donde empieza a cambiar el potencial entre el cono axónico y el axón de la neurona.
5. Se transmite un disparo eléctrico en seguida.
6. La neurona se va a quedar totalmente quieta, durante un breve momento para que la señal pueda viajar hacia el axón.
7. Se va a notar un cambio muy violento en el voltaje, que se va recorriendo a lo largo de todo el axón.

La neurona (típica) a lo largo de su axón, está cubierta de nodos (y de celulas de mielina). Estos nodos están para evitar que se distorcionen o se pierda la señal recibida en la dendritas, en ellos se recarga la señal para que llege hasta el soma de la neurona. Donde se mandará o no una respuesta, un pulso electrico.

Elementos de las neuronas

Los principales elementos que participan durante la transmisión de señales entre neuronas son:

- **Impulsos eléctricos:** potenciales de acción, cambios de voltaje que ocurrien a lo largo del axón. Pueden generar dos efectos en la membrana de la neurona:
  - ⋆ **El efecto excitatorio**, despolariza la membrana postsináptica. La neurona es más propensa a mandar un pulso electrico.
  - ⋆ **El efecto inhibitorio**, hiperpolariza la membrana postsináptica. La neurona no manda pulso electrico.

- **Neurotransmisor/es:** donde se encuentra la sustancia quimica, son los mensajeros químicos que se comunican entre neuronas adyacentes. La liberación de neurotransmisores de una neurona ayudará a despolarizar o hiperpolarizar (aumentar la magnitud de la carga) de la neurona adyacente, lo que hará que sea más o menos probable que ocurra un potencial de acción en la siguiente neurona.

- **Plasticidad:** La modificación a largo plazo de las conexiones entre neuronas.

Sinapsis

Donde dos neuronas en estrecha proximidad, se transmiten información se llama **sinapsis**. Snell 2009

La sinapsis (típica) se establece entre el axon de una neurona y la dendrita de otra neurona.

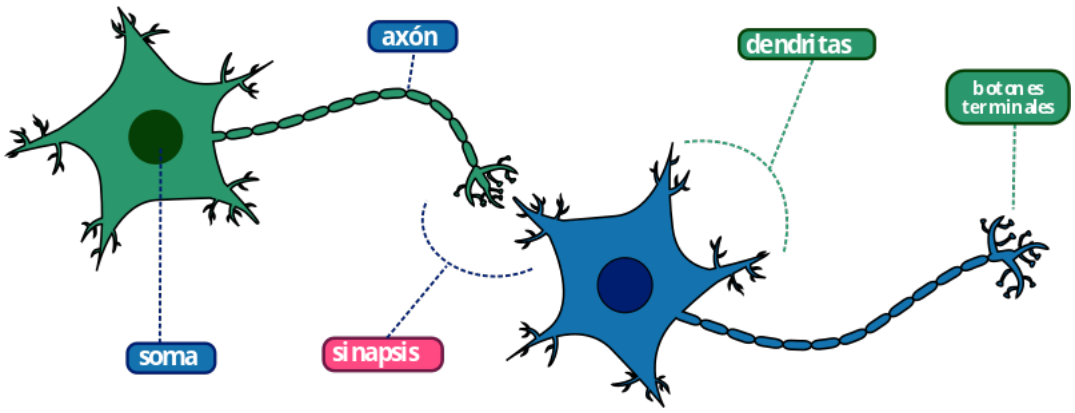


Figura 1.3 Part of neurons in Spanish, Dana Scarinci Zabaleta, 24 febrero 2019, Wikimedia Commons, https://commons.wikimedia.org/wiki/File:Part_of_neurons_in_Spanish.svg, OpenStax, CC0

Clasificación de sinapsis

Los dos tipos de sinapsis (relevantes para el curso) son:

Sinapsis eléctrica: las membranas de las células pre y postinápticas se unen en la brecha sináptica, que son pequeños canales que permiten el paso de iones (Figura 1.4).

Sinapsis química: la neurona libera moléculas neurotransmisoras a otra neurona adyacente en un pequeño espacio, la brecha sináptica (Figura 1.5).

1. Neurona biológica

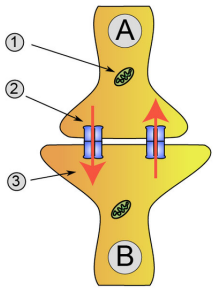


Figura 1.4 Synaptical transmission (electrical). Neurona A transmisora, Neurona B receptora, 1. Mitocondria, 2. Uniones gap formadas por conexinas, 3. Señal eléctrica, Nrets commonswiki, 23 September 2005, Wikimedia Commons, https://commons.wikimedia.org/wiki/File:Synapse_diag2.png, Inkscape 0.42, CC-BY-SA 3.0

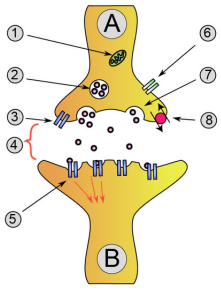


Figura 1.5 Synaptical transmission (chemical). Neurona A transmisora, Neurona B receptora, 1. Mitocondria, 2. Vesícula sináptica llena de neurotransmisor, 3. Autorreceptor, 4. Brecha sináptica, 5. Receptor de neurotransmisores, 6. Canal de calcio, 7. Neurotransmisor liberador de vesículas fusionadas, 8. Bomba de recaptación de neurotransmisores, Utilisateur:Dake, 23 September 2005, Wikimedia Commons, https://commons.wikimedia.org/wiki/File:Synapse_diag1.png, Inkscape 0.42, CC-BY-SA 3.0

Señal eléctrica

El paso de las señales eléctricas entre neuronas, se dan gracias a los canales de iones, los que destacan son los siguientes:

- **Canal por fuga:** Estos se abren y cierran aleatoriamente, todo el tiempo están activos en la neurona. Por ejemplo: sodio y potasio.
- **Canal regulado por ligado:** En este un neurotransmisor que es el que provocar o impede que se abran.
- **Canal por estímulo mecánico:** Permiten que pasen más iones o menos iones dependiendo, si se ejerció una presión, por ejemplo, con las neuronas cerca de la piel.

- **Canales regulados por el voltaje:** Un voltaje, es el que abre o cierra el canal. Permite o no el paso del pulso electrico.

Existen realmente una buena cantidad de iones presentes en el cerebro, pero los más protagónicos son **potasio**, el **sodio** y el **cloro**. Los que vamos a utilizar para un modelo matemático de las neuronas.

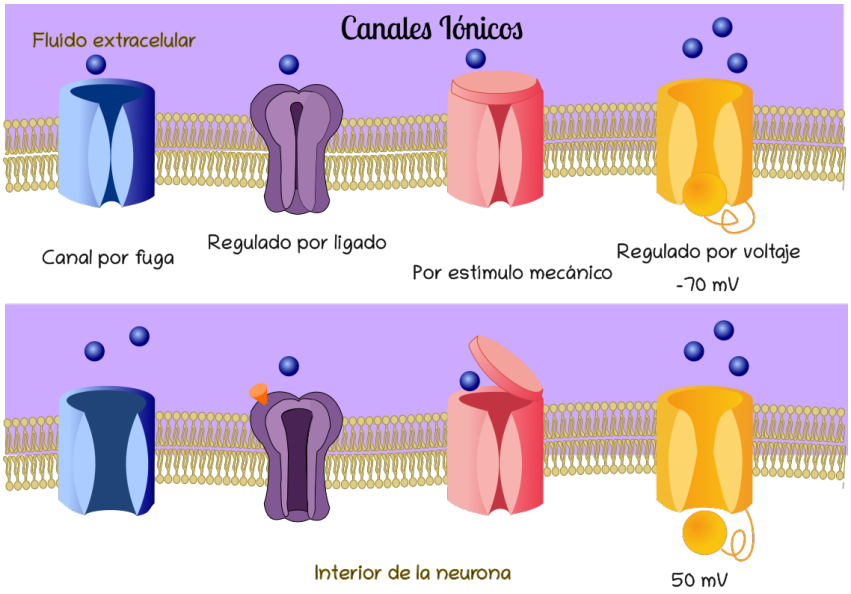


Figura 1.6 Representación de la clasificación de los canales iónicos.

Neurotransmisón

La neurotransmision está detallada en la figura Figura 1.7.

Por último veamos brevemente **la neuroplasticidad**, es lo que nos permite el aprendizaje a largo plazo en el cerebro, es un mecanismo de aprendizaje del cerebro en el cual cuando las neuronas se activan simultáneamente con frecuencia la conexión entre ellas se fortalece.

Este mecanismo constituye la principal inspiración para el diseño de las redes neuronales artificiales, concretamente en esto se inspiran los algoritmos de entrenamiento. Lo que se hace es calcular, qué conexiones debemos reforzar y cuáles debemos de debilitar para que nuestras redes neuronales calculen las funciones que a nosotros nos interesan.

1. Neurona biológica

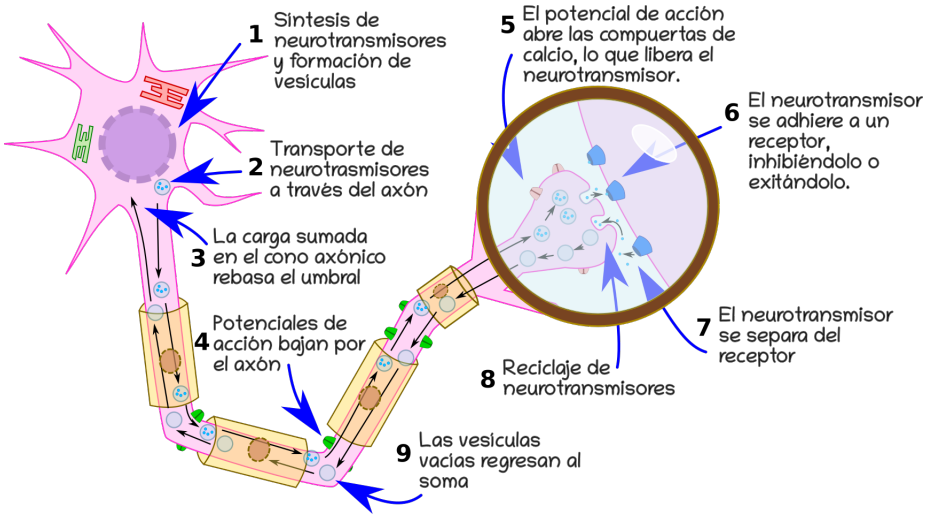


Figura 1.7 Esquema detallado de una neurotransmisión.

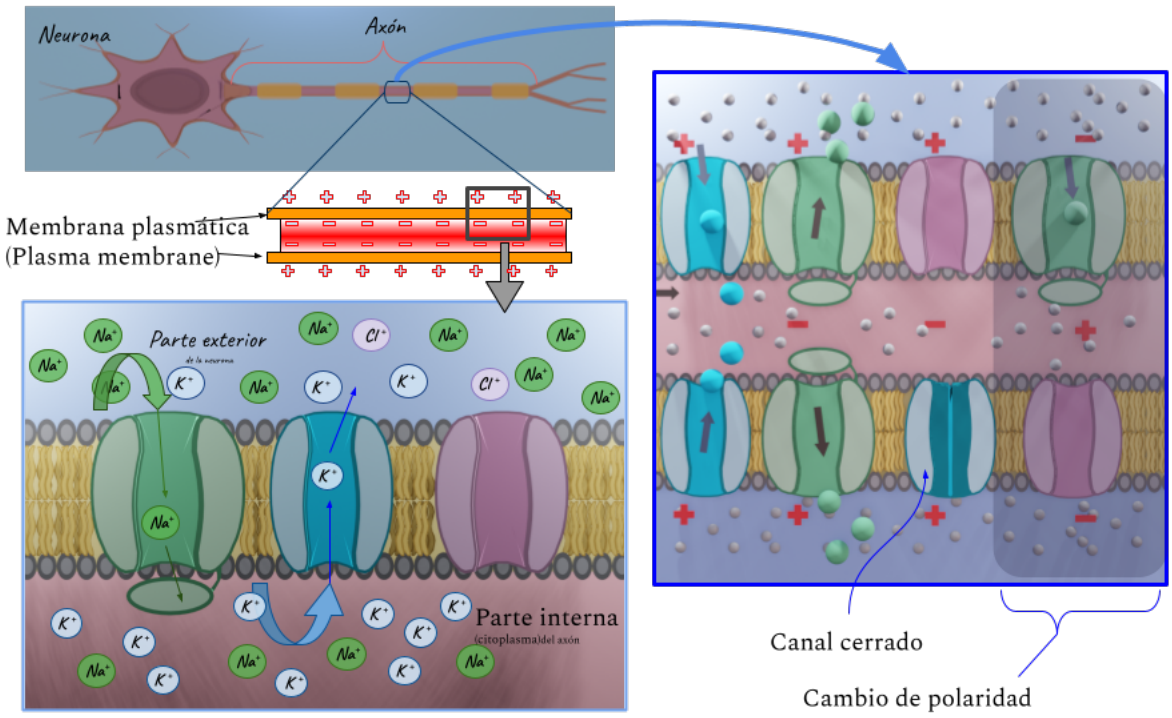


Figura 1.8 Representación de la membrana axónica en potencial de reposo en la parte infererior izquierda, y en la parte derecha con un estímulo que genera el cambio de polaridad en la misma, así como el cierre de canales y paso de iones.

2 | Modelo de Hodgkin-Huxley

Introducción

El modelo de HH.

3 | Aprendizaje de máquina

Introducción

En este capitulo se desarrolla el procesos que pasa una maquina para que *aprenda*, para esto notemos el concepto de aprender. En lo seres humanos se denota como el hecho de adquirir el conocimento de algo mediante el estudio o la experiencia a partir de ejemplos específicos en nuestro medio, entonces aquellos problemas que inicialmente no pueden resolverse, puedan ser resueltos después de obtener más información acerca del problema. Desde pequeños empezamos por aprender por palabras (conceptos) que asociamos con algo especifico, para después relacionar que varios objetos pertenecen a un tipo de conjunto y otros no. Como que un muñeco, una pelota y unos bloques de construcción de plastico, pertecen a un conjunto denotado como "juguetes", y que plato, taza, y tazón no pertecen a este conjunto sino al conjunto "vajilla". Entonces para organizar los conceptos que vamos aprediendo hacemos uso de una función booleana, donde la entrada es el concepto, la pregunta es, ¿Este objeto pertence a un cierto conjunto de objetos con características similares? y la salida es falso o verdadero. A este proceso, se le conoce como *aprendizaje de conceptos* y en este curso lo simplificamos como *función boleana de aproximación mediante ejemplos*.

Ahora lo que denotamos como el hecho que una maquina aprenda, lo vemos como cualquier programa que mejore su desempeño en alguna tarea mediante la experiencia. Con más formalidad se denota como:

Definición 3.1

Apredizaje maquina: Se dice que un programa de computadora aprende, si su desempeño en T, medido por P , mejora con la experiencia E. Tal que:

- T es un tipo de tarea.
- P es una medida de desempeño.

- E la experiencia con ejemplos (entrenamiento).

, [a].

[a] Machine Learning, Mitchell 1997, pag. 14.

Algunos ejemplos para definición anterior pueden ser los siguientes:

- T, como:
  - ⋆ Jugar un juego de mesa.
  - ⋆ Clasificar varios tipo de hojas.
  - ⋆ Reconocer una voz en particular.
- P, como:
  - ⋆ Porcentaje de juegos ganados en las partidas.
  - ⋆ Porcentaje de hojas correctamente clasificadas.
  - ⋆ Porcentaje de reconocimiento de timbre de la voz.
- E, como:
  - ⋆ Jugar juegos de práctica.
  - ⋆ Una secuencia de imagenes etiquetadas.
  - ⋆ Una secuencia de audios etiquetados.

A partir de ahora nos dedicaremos a definir correctamente tareas que nos interese que un programa aprenda, para entender la forma más abstracta del problema y así proponer los algoritmos que nos ayuden a resolverlo.

Consideremos también que los sistemas de redes neuronales artificiales son un tipo de algoritmo para la representación del proceso de aprendizaje. Un problema de aprendizaje bien definido requiere una tarea bien especificada, medidas de desempeño y datos para obtener experiencia.

El aprendizaje maquina se apoya de disciplinas, como la inteligencia artificial, probabilidad, estadística, complejidad computacional, psicología, neurobiología y filosofía.

Para proponer un algoritmos de aprendizaje automático necesitamos, elegir el tipo de experiencia de entrenamiento, definir la función objetivo a aprender y un algoritmo para aprender la función objeto a partir de ejemplos de entrenamiento.

Los algoritmos de aprendizaje maquina han sido utilizados apliamente por la industria bancaria, por gobiernos y por su puesto por el area de la salud. En la industria bancaria por

ejemplo, donde es necesario aprender las reglas generales para determinar la solvencia crediticia, a partir de las bases de datos. Por los gobiernos para el reconocimiento de rostros humanos a partir de imágenes. En el area de salud para a partir de bases de datos de pacientes descubrir automaticamente regularidades implicitas en los resultados de tratamientos.

Espacio de Hipotesis

El aprendizaje automático, es utilizar datos disponibles para, aprender una tarea mediante una función que mejor mapee entradas a ciertas salidas. A esto se le llama aproximación de función, en el que aproximamos una función de destino desconocida (que suponemos que existe) que puede asignar mejor las entradas a las salidas en todas las observaciones posibles del dominio del problema.

Una función de un modelo que se aproxima a la función objetivo y realiza asignaciones de entradas a salidas se denomina hipótesis.

Ahora estas funciones pueden tener formas muy generales en el aprendizaje de máquina pueden tener forma, por ejemplo, de estructuras de datos, como los árboles de decisión, donde cada nodo pregunta si o no, pertenece una clasificación. pueden ser también funciones matemáticas como el caso de las redes neuronales, entonces la forma que tomen estas hipótesis en general puede abarcar muchos métodos y estructuras.

Entonces el aprendizaje consiste en, explorar un espacio de posibles hipostesis para encontrar la hipotesis (una función) que mejor encaje, deacuerdo a lo se obtuvo en los ejemplos de entrenamiento, y predecir alguna característica de salida deseada. Usualmente se denotan como sigue:

- *h* (hipótesis): una sola hipótesis, por ejemplo una instancia o modelo candidato específico que asigna entradas a salidas, se puede evaluar y se usa para hacer predicciones.
- *H* (conjunto de hipótesis): Un espacio de posibles hipótesis para mapear entradas.

Una breve ejemplo para denotar un espacio de hipostesis sería el problema es saber los días que nos conviene ir al cine, donde nuestra tarea T es aprender a predecir el conjunto de dias que nos conviene ir al cine, basado en los atributos de los dias, donde cada hipotesis la representaremos apartir de un conjunto de atributos de las instancias (dias), estonces cada hipotesis es un vector con tres atributos, *tiene2x1, esEstrenoDePelicula, actoresConocidos*. Para cada atributo de la hipotesis tendría uno de los siguientes valores; $\mathrm{Si}, \mathrm{No}, ?$. Donde ? indica que cualquier valor es valido para ese atributo.

Cuando alguna instancia x cumpla con todos los atributos de una h, entonces $\mathrm{h}(\mathrm{x}) = 1$ y x es un ejemplo positivo. Entonces para representar la hipostesis, que nos conviene ir

solo los dias con 2x1, y que hay peliculas donde los actores son conocidos, la escribimos como $h(<Si, ?, Si>) = 1$, la hipotesis que cualquier día nos conviene ir al cine la denotamos como $h(<?, ?, ?>) = 1$, nuestra función objetivo la denotamos como una función booleana $c : X \rightarrow 0, 1|X, elconjuntodelos365diasdelaño$, entonces $c(x) = 1$ cuando en los datos nos dicen que con la instancia x conviene ir al cine, $c(x) = 0$ en caso que no. Por tanto para aprender la tarea T, necesitamos *una hipotesis h en H tal que h(x) = c(x) para todas las x en X* . La tarea de aprendizaje del concepto c requiere aprender el conjunto de instancias que lo satisface, describiendo este conjunto mediante una conjunción de restricciones sobre los atributos de la instancia.

Estas hipotesis (funciones) pueden llegar a ser sumamente complejas y tener que mapear datos de entrada con muchas formas ej. imagenes, trayectorias, etc. En el caso de las redes neuronales, el espacio de hipótesis está determinado por la arquitectura de la red. Vamos a definir el espacio de hipótesis, cuando decidamos qué neuronas vamos a poner en nuestro sistema, como las conectamos entre sí y cómo van a transferirse información de una a la otra y cuántas neuronas van a ser. Lo que veremos a lo largo del curso son diferentes arquitecturas y el impacto que tiene hacer diferentes modificaciones así como las matemáticas que existen detrás de estas.

Clasificación de los conjuntos de datos

La experiencia E para aprender la vamos a obtener mediante un conjunto datos, llamados datos de entrenamiento, estos se separan en tres bloques:

- **Entrenamiento:** Datos con los cuales se ajustan los parámetros de la hipótesis (del 50 % al 80 % de los datos). En este bloque se escoje que función del espacio fue mejor para el aprendizaje.
- **Validación:** Datos utilizados para ajustar los parámetros (hiperpametros) del algoritmo de entrenamiento, que puedan afectar qué hipótesis es seleccionada (del 25 % al 10 % de los datos y no deben ser usado durante el entrenamiento). Un ejemplo de un hiperparámetro para redes neuronales son el número de nodos ocultos en cada capa.
- **Prueba:** Datos utilizados para evaluar la posibilidad de que la hipótesis aprendida generalice [1] a datos no vistos anteriormente. Esta porción que se mantiene aparte. Con estos se evalua el modelo, se reporta la eficacia del modelo según los resultados en este conjunto (del 25 % al 10 % de los datos).

[1] Se desea que nuestro modelo de aprendizaje, una vez entrenado con datos que ya hemos visto, se pueda usar con datos nuevos. Para ello debemos asegurarnos que el modelo no ha simplemente memorizado las muestras de entrenamiento, sino que ha aprendido propiedades del conjunto.

El conjunto de datos de entrenamiento se usa para aprender una hipótesis y el conjunto de datos de prueba para evaluarla.

Tipos de aprendizaje

Aprendizaje Supervisado , el modelo usa datos etiquetados a una respuesta especifica(labaled data), durante el entrenamiento se intenta encontrar una función que aprenda a asignar los datos de entrada (input data) con los datos en el etiquetado. Para depues predecir una relación, dado un dado totalmente nuevo para el modelo. Los modelos pueden ser:

- Regresión: Un modelo de regresión busca predecir valores de salida continuos. Por ejemplo, en predicciones meteorológicas, de expectativa de vida, de crecimiento de población.
- Clasificación: En un problema de clasificación se desea predecir una salida discreta. Por ejemplo, identificación de dígitos, diagnósticos.

Aprendizaje no supervisado , es usado cuando no se tienen datos "etiquetados" para el entrenamiento. Solo sabemos los datos de entrada. Por tanto, únicamente podemos describir la estructura de los datos, para intentar encontrar algún tipo de organización que simplifique un análisis. Por ello, no se tienen valores correctos o incorrectos (es utilizado para aprender de una manera autoorganizada).

Aprendizaje por refuerzo , inspirado en la psicología conductista; donde el modelo aprende por sí solo el comportamiento a seguir basándonos en *recompensas y penalizaciones*. Este tipo aprendizaje se basa en mejorar la respuesta del modelo usando un proceso de retroalimentación (*feedback*). Su información de entrada es el feedback que obtiene del mundo exterior como respuesta a sus acciones. A aprende a base de ensayo-error.

Mientras que el aprendizaje supervisado y el no supervisado aprenden a partir de datos obtenidos en el pasado, el aprendizaje por refuerzo aprende desde cero, es decir, con un estado inicial y son su ambiente, va aprendiendo a futuro, mediante posibles penalizaciones o recompensas. El *aprendizaje por refuerzo* es usado en videojuegos porque cada vez que se realizan las acciones correctas se ganan puntos y entonces se entrena a la gente para que pueda conseguir la mayor cantidad de puntos. En este siempre hay: un agente, un ambiente definido por estados, acciones que el agente lleva a cabo (que le llevan de un estado a otro), y recompensas o penalizaciones que el agente obtiene.

En cada acción, el agente solo conoce el estado en el cual se encuentra y las acciones posibles que puede elegir a partir de ese estado. No sabe si llegando al siguiente estado, obtendrá mejores o peores recompensas, irá aprendiendo en cada estado qué acciones lo llevará a obtener una mayor recompensa a largo plazo, y que el valor de las acciones en ese estado puedan subir. *Se enfoca en que el agente aprenda una política óptima*

para alcanzar el objetivo. El agente siempre está en fases de *exploración* y *explotación*, en la fase de exploración el agente toma una acciones de manera aleatoria, y en la de explotación va a tomar acciones basándose en cuán valiosa es realizar una acción a partir de un estado dado.

En plataforma de ventas en línea es donde podemos encontrar este tipo de modelo que están entrenados con este tipo de aprendizaje, donde al iniciar la sesión no conoce nada del usuario, solamente tiene un ambiente dado por los productos de la plataforma y su estado inicial es cero, para hacer individual la experiencia del usuario y que compre más. El algoritmo realiza la acción de mostrar ciertos productos (algún estado) si el usuario da clic a estos productos, el agente recibirá un punto de recompensa, por lo cual pasará a otro estado donde ofrecerá productos del mismo estilo donde pueda maximizar una venta, así sé ira adaptando a cada usuario.

Parte II
Redes dirigidas acíclicas

4 | Perceptrón simple

Perceptrón

El perceptrón fue la primera red neuronal artificial (o ANS, Artificial Neural Systems) descrita algoritmicamente. En las decadas de los 60's y 70's, los popularizo el psicólogo Franck Rosenblatt, en su libro llamado Principios de neurodinámica, donde presentó varios modelos de perceptrones, en el Laboratorio Aeronáutico de Cornell en Estados Unidos, originalmente estaba diseñado para ser una máquina, en vez de un algoritmo. Estaba diseñado especificamete para el reconocimiento de imagenes donde, cada peso era un cable físico por pixel de entrada, este erá una matriz de 200 x 200, conectados aleatoriamente a las "neuronas", las actualizaciones de los pesos se realizarón mediante motores eléctricos.

El perceptrón es en sí, es la representación de un sola neurona, este se ocupa para la clasificación de patrones en un conjunto de datos multivariados, con ese se obtienen fronteras lineales en el plano, mediante un algoritmo de aprendizaje que veremos más adelante.

Recordando, una neurona es un celula elemental que a partir de un vector de entrada procedente del exterior o de otras neuronas (estimulo), proporciona una única respuesta (si activo el potencial de acción o no), ver figura 4.1. Los elementos que actuan en una neurona los podemos listar como:

- **Entradas:** $x_j(t)$. Las variables de entrada y salida pueden ser binarias (digitales) o continuas (analógicas) dependiendo del modelo de aplicación.
- **Pesos sinápticos:** w_{ij}. Representan la intensidad de interacción entre cada neurona presináptica j y la neurona postsináptica i.
- **Regla de propagación:** $h_i(t) = \sigma(w_{ij}, x_j(t))$. Proporciona el valor del potencial postsináptico, de la neurona i en función de sus pesos y entradas.
  - $h_i(t) = \sum_{i=0}^{n} w_{ij} x_i$, Es una suma ponderada de las entradas con los pesos sinápticos. Así, si la entrada es positiva, dependiendo de los pesos podemos

saber si fue una sinapsis excitadora (pesos positivos) o inhibidora (pesos negativos).

- **Función de activación o de transferencia:** $a_i(t)$ Proporciona el estado de activación actual, de la neurona i en función de su estado anterior, $a_i(t-1)$ y de su potencial postsináptico actual.
  - ⋆ $a_i(t) = f_i(a_i(t-1), h_i(t))$, es la que usalmente se usa.
  - ⋆ $a_i(t) = f_i(h_i(t))$, en algunos modelos solo se concidera que el estado actual no dependende del tiempo anterior.
- **Función de salida:** $F_i(a_i(t))$ Da la salida actual, $y_i(t)$, de la neurona i en función de su estado de activación actual. El estado de activación de la neurona se considera como la propia salida.
  - ⋆ $y_i(t) = F_i(a_i(t))$
  - ⋆ $y_i(t) = F_i(f_i(a_i(t-1), \sigma(w_{ij}, x_j(t))))$

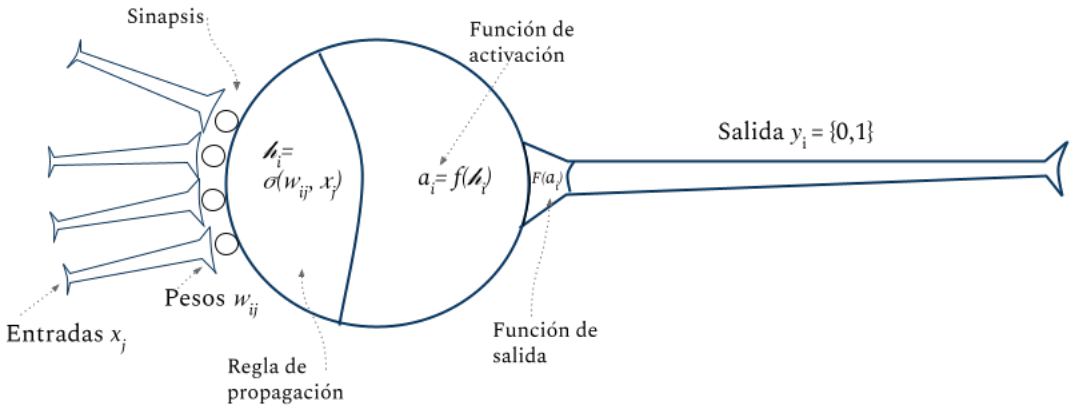
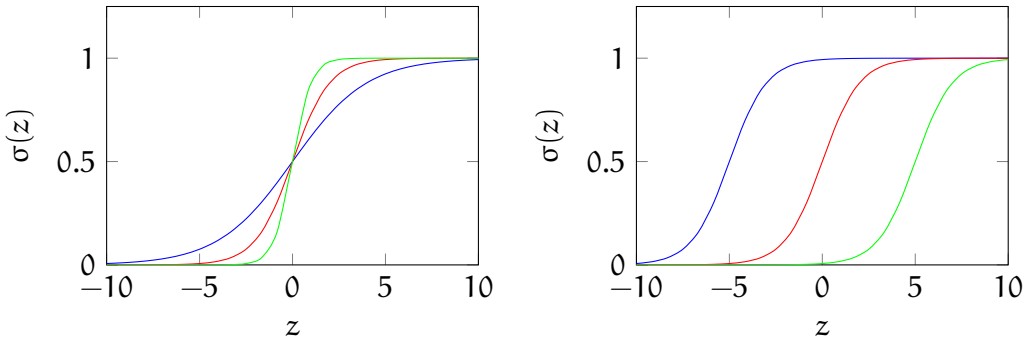


Figura 4.1 Neurona vista como un modelo artificial (perceptrón).

Un perceptrón toma un vector de entradas de numeros reales, calcula una combinación lineal de estas entradas, luego emite un 1 si el resultado es mayor que algún umbral y -1 de lo contrario. Es decir, dadas las entradas $x_1 ... x_n$, la salida $o(x_1 ...., x_n)$ calculada por el perceptrón es 1 si $w_0 + w_1 x_1 + w_2 x_2 + ... + w_n x_2 > 0$ y -1 de lo contrario, donde cada w es una constante $\mathbb{R}$, un peso, que determina la contribución de la entrada x a la salida del perceptrón. La constante w_0 es un *umbral* (bias) que la suma de las entradas con los pesos debe superar para que el perceptrón emita un 1. En otras palabras es un peso que va a actuar junto con una entrada de valor 1, que vamos a poder ajustar para que nuestrá función de activación, se mueva de derecha a izquierda en el plano para ayudarnos a ajustar nuestros resultados, provocando un gran impacto en el aprendizaje.



(a) Función sigmoide. **(b)** Función sigmoide con bias.

Figura 4.2 Comportamiento de la función de activación (sigmoide) de un perceptrón con una sola entrada, *(b)* el perceptrón sin el uso de bias, *(b)* con el uso del bias, donde apesar de estár representandos con la misma entrada , el uso del bias afecta en los resultados de salida. Así si quisieramos que este perceptrón nos diera $y = 0$ con una entrada $x = 2$ sin el uso, ni ajuste del bias sería imposible, pues en (a) apesar que la gráfica azul está la entrada está ajustada con el peso $w = 0.5$, la roja con el $w = 1$, y la verde con el $w = 2$ solo lo gramos alargarla un poco, haciendo que entradas que antes eran correctas ahora caigan 0 también. Entonces lo que necesitamos es más bien "mover"la gráfica, esto lo logramos con la gráfica *(b)* donde la entrada (única) está sumada con un bias (umbral) $x_0 = 1$, ajustado en azul con peso $w_0 = 5$, en rojo con $w_0 = 0$, en verde con $w_0 = -5$ y el peso $w_1 = 1$. Donde con $w_0 = -5$ logramos nuestro objetivo de tener una salida $y = 0$ con $x = 2$. El bias nos permite mover la función fuera del origen.

Esto se muestra en la siguientes graficas 4.2b. Más adelante hablaremos de su regla de entrenamiento (Training rule).

El hecho de que un perceptrón aprenda implica elegir valores para los pesos denotados también por θ. Ahora, el espacio H de las hipótesis candidatas consideradas en el aprendizaje del perceptrón, es el conjunto de todos los posibles vectores de pesos.

$$H = \boldsymbol{w} | \boldsymbol{w} \in \mathbb{R}^{n+1}$$

.

Si bien en el momento que se publicó los logros con el modelo del percetrón las espectativas eran bastante altas, a medida de los años los cientificos Marvin Minsky and Seymour Papert desestiman en gran medida los alcances que realmente se puede tener con el perceptrón, al mostrar [1] que no puede predecir operaciones lógicas que no sean linealmente separables, tal es el caso de la función XOR, que no es separable linealmente, siendo imposible que pueda aprender está función. Esto y el gran costo que representaba procesar todos los elementos que implicaba el entrenamiento, causa que por un buen

[1] En 1969 publican Marvin Minsky y Seymour Papert que perceptrones de una sola capa (simples) solo son capaces de aprender a distinguir patrones linealmente separables, en el libro "Perceptrons".

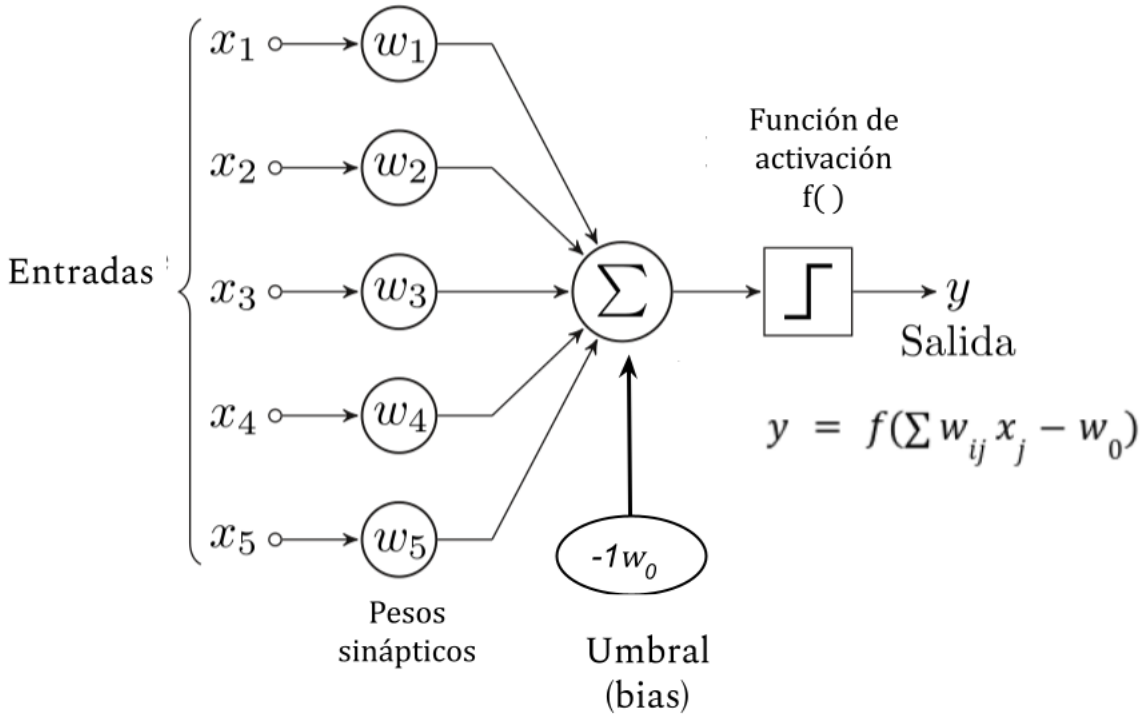


Figura 4.3 Modelo estandar de un perceptrón.

tiempo se desetime el uso del perceptrón. Es hasta después de unas decadas que vuelve a tener relevancia con la propuesta de un perceptrón multicapa usando retropagación (Feedforward), siendo estos capaces de resolver la función XOR

Compuertas lógicas con neuronas

Aquí se muestra como se puede utilizar un perceptrón para simular compuertas lógicas tales como el or, not, and.

Para simular la compuerta *not*, como está es una función boolean de $\mathrm{B} \to \mathrm{B}$, talque $\mathrm{not}(\mathrm{x}) = -\mathrm{x}$ entonces en un plano de dos dimensiones, la podemos representar con dos puntos, el $\mathrm{p}_1 = (0,0)$ y $\mathrm{p}_2 = (1,0)$donde p_1 representa cuando $\mathrm{not}(0) = 1$, p_1 representa cuando $\mathrm{not}(0) = 1$. Teniendo el espacio de la función definido lo que nos toca es, separar el plano para clasificar las entradas, este claramente se puede separar con un linea vertical, o con lineas con pendiente 1 o -1. Para esta función solo necesitamos de una entrada y un sesgo (bias), donde la entrada la combinaremos con un peso, este peso lo asignaremos a tanteo (por la sencillez de la operación). Así el peso $w_1 = -1$ y el peso asignado al bias sera $w_0 = 0.5$, ahora con esto datos podemos:

- Hacer la función de propagación donde $\mathrm{h}(\mathrm{x}) = (\mathrm{x} * -1) + (0.5) * 1 = 0.5 - \mathrm{x}$

- Haccer la función de activación escalón $a(x) = sgn(h) = sgn(0.5 - x)$
- Dar la salida donde $s(1) = sgn(0.5 - 1) = 0$ y $s(0) = sgn(0.5 - 0) = 1$, en este caso la salida es la identidad de la activación.

x	h	s
0	0.5	1
1	−1.5	0

Algo similar va a pasar con la compuerta *and* y *or* donde al necesitar de dos entradas para la compuerta, asignaremos dos entradas para el perceptrón igualmente y las representaremos en el plano con cuatro puntos, donde cada punto representa una instancia y se le asigna valor positivo o negativo en el plano, así pues para el and tenemos los puntos $p_1 = (0,0)$, $p_2 = (0,1)$, $p_3 = (1,0)$, negativos y $p_4 = (1,1)$ el único positivo. Así nos damos cuenta que necesitamos una recta con pendiente negativa y fuera del origen, que nos separe estás clases de puntos. Por tanto para el bias le asignamos un peso de $w_0 = -1.5$, $w_1 = 1$ y $w_2 = 1$, con estos datos podemos:

- Hacer la función de propagación donde $h((x_1, x_2)) = (x_1 * 1) + (x_2 * 1) + (-1.5) * 1 = x_1 + x_2 - 1.5$
- Haccer la función de activación escalón $a((x_1, x_2)) = sgn(h) = sgn(x_1 + x_2 - 1.5)$
- Dar la salida donde $s = a(x)$, es la identidad de la activación.

x_1	x_2	h	s
0	0	−1.5	0
0	1	−0.5	0
1	0	−0.5	0
1	1	0.5	1

Para la compuerta *or* es algo muy similar pues podemos igualmente representar la función con cuatro puntos en el espacio cada uno representando una instancia, solo que ahora tres de estos puntos serán positivos y solo uno negativo, los puentos positivos serían $p_2 = (0,1)$, $p_3 = (1,0)$ y $p_4 = (1,1)$, mientras que $p_1 = (0,0)$ negativo, el plano lo podemos dividir con una linea recta con pendiente negativa, así le asignamos $w_0 = -0.5$, $w_1 = 1$ y $w_2 = 1$, con esto hacemos los paso que ya sabemos:

- Hacer la función de propagación donde $h((x_1, x_2)) = (x_1 * 1) + (x_2 * 1) + (-0.5) * 1 = x_1 + x_2 + 0.5$
- Haccer la función de activación escalón $a((x_1, x_2)) = sgn(h) = sgn(x_1 + x_2 + -0.5)$

- Dar la salida donde $s = a(x)$, es la identidad de la activación.

x_1	x_2	h	s
0	0	−0.5	0
0	1	0.5	1
1	0	0.5	1
1	1	−1.5	1

Tomando el hecho que en la naturaleza las neuronas van pasando información en una estructura que forma niveles de abstración, esto lo modelamos como capas de neuronas conectadas entre sí, cada capa haciendo su trabajo de abstración. El perceptrón simple es un modelo neuronal unidireccional, una capa de entrada y otra de salida, que por si solo no puede separar todas la funciones lógicas pues tenermos el XOR, para resolver esto usaron perceptrones multicapa que se explicará más adelante en el curso.

Funciones de activación

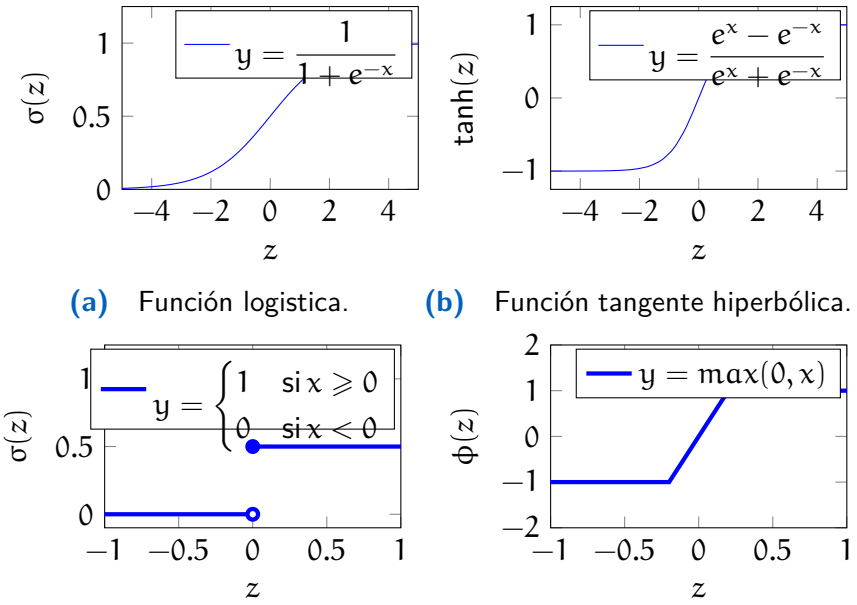
Recordando que la forma de las funciones de activación es $y = f(x)$, donde x representa el potencial postsináptico e y el estado de activación de la neurona, es decir si va a lanzar un disparo o no. Las funciones de activación más empleadas son:

Funciones de error

Entonces tomando como base el perceptrón, una vez obtenidas las salidas con una primera iteración nos daremos cuenta de que tan lejos o que tan cerca estuvimos de la respuesta correcta, con esto darnos la oportunidad de que pesos ajustar respecto a sus entradas, en la segunda iteración. Ahora para facilitarnos esto y tomando en cuenta que el entrenamiento consiste en varias iteraciones hasta llegar a aprender la tarea T, hacemos uso de una función de error que nos ayude a minimizar la diferencia de error en las salidas.

Primero veamos el entrenamiento para una sola neurona, para esto haremos uso de la regla de aprendizaje del perceptrón (learning rule perceptron), donde para cada entrada, en la capa de salida se le calcula la desviación a la función objetivo. El cual utilizamos para ajustar los pesos del perceptrón (ver fig 4.5).

Usualmente al principio del entrenamiento se asignan pesos aleatorios, a medida que avance el entrenamiento, se van modificando con cada iteración, así $w_i \leftarrow w_i + \Delta w_i$. Esto con base a *la regla de aprendizaje* donde:



(a) Función logistica.

(b) Función tangente hiperbólica.

(c) Función escalón.

(d) Función lineal a tramos. Tangente hiperbólica rectificada.

Figura 4.4 Las funciones de activación más usadas son la función sigmoide $\sigma(z)$ y la tangente hiperbólica $\tanh(z)$.

$$\Delta w_i = \alpha(y - y_{out})x_i \tag{4.1}$$

Con y es la salida deseada, y_{out} la salida generada, α la taza de aprendizaje (learning rate) y x_i la entrada i. Lo que hace la taza de aprendizaje es moderar el grado en que los pesos son cambiados con cada iteración, se le asigna un valor muy pequeño (0.1 o 0.2) y conforme se logran ajustar los pesos se minimiza aún más.

Para entrenar un perceptrón, utilizamos cualquier método de optimización de funciones para encontrar los parámetros w que minimizan el error con alguna de las siguientes funciones de error:

Diferencias al cuadrado, también conocida como regla aprendizaje delta, es la suma de cuadrados de errores que se tuvieron con cada ejemplar el el conjunto de entranamiento. Los podemos decribir como:

$$\frac{1}{2m}\sum_{m=0}^{M-1}(y_m - a_m)^2 \tag{4.2}$$

con y_m y am, la salida obtenida dado un ejemplar m y la salida correcta del ejemplar m respectivamente.

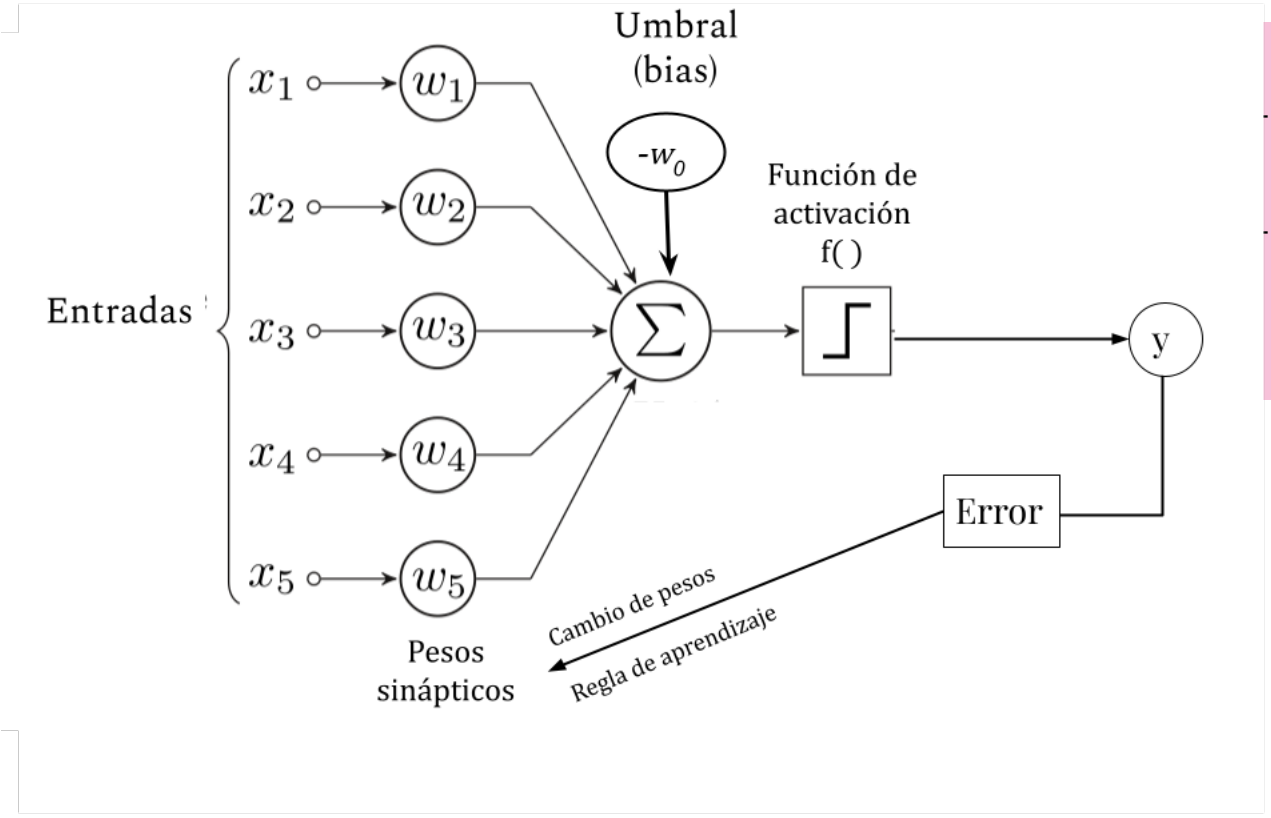


Figura 4.5 Modelo estandar de un perceptrón.

Entropía cruzada. Se usa para problemas de clasificación ya que su comportamiento es más suave (soft) y permite hacer clasificaciones más certeras. Se define como:

$$L(\Theta) = -\frac{1}{m}\sum_{M-1}^{m=0}(y^{(}m)\log(a_m) + (1-y_m)\log(1-a_m)) \tag{4.3}$$

Entonces juntando los conceptos que ya sabemos podemos describir el algoritmo de entrenamiento para el perceptrón de la siguiente forma:

1. Iteramos sobre todos los ejemplares.

2. Para cada ejemplar se calcula la función de propagación, es decir la suma ponderada.

3. Se calcula la función de activación con esta suma.

4. Se calcula la función de salida.
5. Actualización de pesos de acuerdo a la regla de aprendizaje.
6. Repetir de 1-6 hasta que los pesos nos satisfagan, un número de iteraciones establecidas.

Medidas de rendimiento

Las medidas de rendimiento de una red nos sirven para ver de manera croncreta como se comportó nuestra red, durante el entrenamiento. Que fue lo que pudo aprender (clasificar). Si tuvo un sobreajuste, es decir, memorizo y por tanto, será incapaz de predecir la clase a la que pertenece realmente el ejemplar. Para esto se usan las siguientes herramientas:

Matriz de confusión : (Confusion matrix) Es una matriz, donde las celdas representan las predicciones que hizo nuestro modelo de clasificación de clases, respecto a las salidas esperadas. Así siendo las columnas las salidas $\mathtt{y}$'s del modelo entrenado y las filas las salidas esperadas $\mathtt{y}_{\mathtt{true}}$. Nos facilita a ver cuando un clasificador está confundiendo clases, contabilizando a que clase etiqueto a los diferentes ejemplares. Ahora veamos, que puede estar representando cada celda en la matriz de confusión, estos pueden ser:

1. **VP** Verdaderos positivos *(TP, True Positive)*: La clasificación de los ejemplares predichos, condicen con las etiquetas esperadas de los ejemplares.
2. **VN** Verdaderos negativos *(TN, True Negative)*: El ejemplar que no es parte de clase, no son asignados a esa clase, son predichos correctamente.
3. **FP** Falso positivo *(FP, False Positivo)*: El ejemplar que **no** es parte de una clase $\mathtt{i}$ fue clasificado como tal.
4. **FN** False negativo *(FN, False Negative)*: El ejemplar que es parte de una clase $\mathtt{i}$ no fue clasificado como tal.

Ejemplo 4.1. *Notemos un ejemplo sencillo, supongamos que tenemos una tarea binaria, donde queremos indicar que una persona está embarazada (de acuerdo a unos estudios). Ahora tenemos que:*

- *VP, sería predecir que una mujer está embarazada y que en efecto esté embarazada.* ***(Correcto)***
- *VN, sería con un hombre que no está embarazado y pues en efecto no está embarazado.****(Correcto)***
- *FP, sería predecir que un hombre está embarazado y* ***no*** *este embarazado.* ***(Error, tipo 1)***

- *FN*, *sería predecir que una mujer* ***no*** *está embarazada y esta embarazada.* ***(Error, tipo 2)***

Entonces dados los resultados binarios que nos entregó el modelo, los médicos nos dan las respuestas correctas a 10 estudios, representadas en la siguiente tabla.

Ejemplar	1	2	3	4	5	6	7	8	9	10
Sujeto	M	F	M	F	M	M	F	F	F	M
Etiqueta	No	Si	No	Si	No	No	No	Si	No	No
Clase	0	1	0	1	0	0	0	1	0	0
Predicho	0	0	0	1	0	1	0	1	0	0
Valores	VN	FN	VN	VP	VN	FP	VN	VP	VN	VN

Con estos datos, podemos construir nuestra matriz de confusión MC *contabilizando las salidas obtenidas respecto a las deseadas. Quedando así de la siguiente manera:*

		Salidas y	
		Si	No
Etiquetas	Si	$\mathrm{VP} = 2$	$\mathrm{FN} = 1$
	No	$\mathrm{FP} = 1$	$\mathrm{VN} = 6$

Tabla 4.1 Matriz de confusión binaria.

Ahora para una modelo que sea multiclase, en el que tengamos que clasificar varias clases de ejemplares. Tendremos una matriz M de $n * n$ donde n es el número de clases, así los valores *VP, VN, FP, FN*, son calculados para cada clase e identificados en la matriz M de la siguiente forma, también puedes ver la figura 4.6:

- *VP*, para la clase i será la celda $M[i][j]$ con la $i = j$.
- *VN*, para la clase i será la suma de los valores de toda la matriz menos los valores de la fila i ni los valores de la columna i.
- *FN*, para la clase i será la suma de los valores en la fila i, excepto la celda $M[i][j]$, con $i = j$ que es el *VP*.
- *FP*, para la clase i será la suma de los valores en la columna i, excepto la celda $M[i][i]$ que es el *VP*.

Valores para la clase i		Salidas *y* del clasificador			
		clase 1	clase 2	clase i	clase n
Etiquetas ytrue	clase 1	VN	VN	FP	VN
	clase 2	VN	VN	FP	VN
	clase i	FN	FN	**VP**	FN
	clase n	VN	VN	FP	VN

Figura 4.6 Matriz de confunsión para clasificador multiclases.

Ejemplo 4.2. *Tenemos un clasificador encargado de identificar las cinco vocales del español, escritas a mano. Entonces tenemos un total de 5 clases representadas por* $\mathrm{a}, \mathrm{e}, \mathrm{i}, \mathrm{o}, \mathrm{u}$*, cada letra representada por un número del 0 al 4. Así la clase* $0 = \mathrm{a}$*, la clase* $1 = \mathrm{e}$ *y así respectivamente. Este fue entrenado con 15 ejemplares etiquetados y se obtuvieron los siguientes resultados:*

Ejemplar e	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Etiqueta	a	e	i	o	u	a	a	e	i	i	o	u	o	u	a
Clase $\mathrm{y}_{\mathrm{true}}$	0	1	2	3	4	0	0	1	2	2	3	4	3	4	0
Predicho y	0	1	3	3	2	1	1	1	2	2	4	4	3	2	0

Ahora para hacer la matriz de confusión MC *notamos que tenemos una matriz de* 5x5*, donde* $\mathrm{MC[i]} = \mathrm{Predicho}$ *y* $\mathrm{MC[][j]} = \mathrm{EtiquetasReales}$*. Entonces necesitamos contabilizar las predicciones y asignarlas a sus respectivas celdas, donde con la tabla anterior notamos que en el ejemplar* e3 *con* $\mathrm{y}_{\mathrm{true}}(\mathrm{e3}) = 2$*, se predijo que* $\mathrm{y}(\mathrm{e3}) = 3$*, entonces* $\mathrm{MC}[2][3] += 1$*, pues la etiqueta nos posiciona en la fila* 2 *y lo predicho en la columna* 3*. Ahora en el ejemplar* e4 *con* $\mathrm{y}_{\mathrm{true}}(\mathrm{e4}) = 3$ *se predijo que* $\mathrm{y}(\mathrm{e4}) = 3$*, entonces* $\mathrm{MC}[2][2] += 1$*. Así con cada ejemplar vamos a ir sumando su clasificación. Para construirla tendríamos un pseudocódigo siguiente:*

```
def getMC (Ejemplares, y):
    """
    :param y:  salidas obtenidas (int)
    """
    MC = [len(y)][len(Ejemplares] # y == Ejemplares
    full_zeros(MC)
    for e in range (0, len(Ejemplares)):
        predicho = y[e]
        y_true = Ejemplares[e].etiqueta
```

```
        MC[predicho][y_true] += 1
retrun MC
```

Dados los datos anteriores tenemos, la siguiente matriz de confusión:

		Salidas y				
		1	2	3	4	5
Etiquetas	1	2	2	0	0	0
	2	0	2	0	0	0
	3	0	0	2	1	0
	4	0	0	0	2	1
	5	0	0	2	0	1

Tabla 4.2 MC = Matriz de confusión multiclase.

Para calcular los valores *VP, VN, FP,FN*, por clase se propone el siguiente pseudo-código:

```
MC = getMC(y_salidas, y_true)

VP = VN = FP = FN = [0,0,0,0,0]

for i in range(len(MC)):
  for j in range(len(MC[0])):
      FN[i] = FN[i] + MC[i][j] if (i != j) else FN[i]
      FP[i] = FP[i] + MC[j][i] if (i != j) else FP[i]
      VP[i] = MC[i][j] if (i == j) else VP[i]
  VN[i] = sum(MC) - FN[i] - FP[i] -VP[i]
```

Si quisiéramos saber los valores *VP, VN, FP,FN*, para todo el desempeño total, simplemente sumamos lo obtenido en cada clase así:

```
VP_total = sum(VP)
VN_total = sum(VN)
FP_total = sum(FP)
FN_total = sum(FN)
```

Ahora los valores de cada celda nos representan lo siguiente:

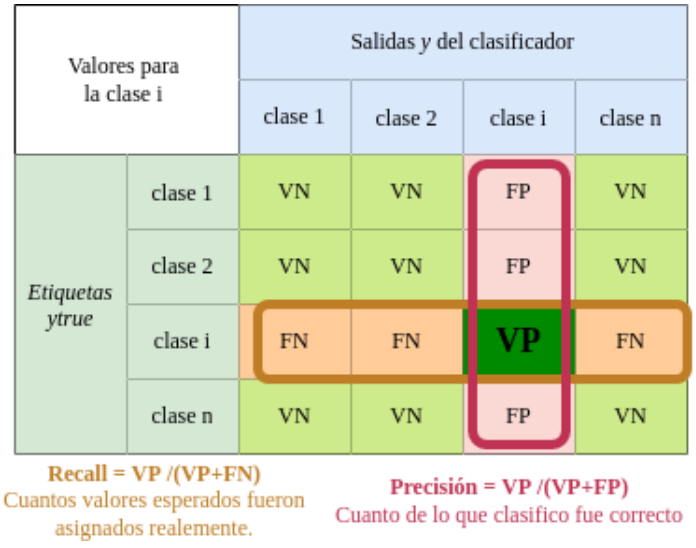
- VP_{total}, toda la diagonal de la matriz. La salida del modelo coincide con lo etiquetado con el ejemplar.
- VN_{total}, todas las veces que no era la clase i y dijo que no era de la clase i.
- FN_{total}, todas las veces que era $clase_i$ y dijo que era $clase_x$. (deseado $FN = 0$)
- FP_{total}, todas las veces que predijo que era $clase_i$ y era $clase_x$. (deseado $FP = 0$)

Así que, si los valores que no están en la diagonal de la matriz son cero o todas nuestras clasificaciones están en la diagonal, podemos decir que nuestro modelo aprendió a clasificar correctamente todas las clases.

Precisión y recuperación : La precisión (*precision*) nos dice la proporción de ejemplares que se logró clasificar correctamente. Mientras que recuperación (*recall*) nos dice cuantas asignaciones de lo que nos interesa pudo clasificar correctamente en otras palabras donde del total de las respuestas correctas que se pueden tener, cuantas respuestas positivas acertadas se tuvo.

$$P = \frac{VP}{VP + FP} = \frac{VerdaderosPositivos}{ValoresEsperados} \tag{4.4}$$

$$R = \frac{VP}{VP + FN} = \frac{VerdaderosPositivos}{ValoresPredichos} \tag{4.5}$$



- Cuando el modelo detecta los ejemplares, pero los incluye en otras clases también: *P* es bajo y *R* es alto.
- Cuando el modelo no detectó bien los ejemplares, pero tampoco los incluyo en otras clases: *P* es alto y *R* es bajo.

- Cuando el modelo detecta los ejemplares y no los incluye en otras clases:P y R es alto.
- Cuando el modelo no detecta los ejemplares:P y R es bajo.

En ocasiones le daremos más importancia al recall y en otras a la precisión. Retomando el ejemplo previo 4.1, no nos importa tanto los casos de error tipo 1, donde el modelo se equivoco con los ejemplares negativos, nos importa los errores de tipo 2, los Falsos Negativos, en este caso le tomaremos especial atención al Recall y que este sea alto. Pero si bien nuestra tarea es detectar cuando un correo es spam, no impacta tanto que aunque en ocasiones un correo sea spam, no lo clasifique como tal (error tipo 2 FN), nos es crucial que un correo que no sea spam nos lo clasifique como tal (error tipo 1 FP). Así deseando que los falsos positivos se acerquen a cero, dándonos como resultado una precisión alta aunque el recall sea bajo.

Exactitud y medida f : La exactitud (*accurrancy*) es una medida de cuántas predicciones correctas en total hizo el modelo para el conjunto de datos completo (no se recomienda usar si tienes clases desbalanceadas, es decir, muchos elementos de una clase y poco de otra pues nos puede fallar totalmente con las clases pequeñas y aun así su valor sería alto). La medida f (*(f score)*) se utiliza para combinar las medidas de precisión y recall en un solo valor. Es práctico porque hace más fácil el poder comparar el rendimiento combinado de la precisión y la recall. Se dan las ecuaciones a continuación:

$$Acurrancy = A = \frac{VP + VN}{VP + VN + FP + FN} = \frac{VP + VN}{TodosLosValoresClasificados} \tag{4.6}$$

$$F = \frac{2}{\frac{1}{P} + \frac{1}{R}} = 2\frac{P * R}{P + R} \tag{4.7}$$

La medida f, también la podemos escribir (con un poco de aritmética) como:

$$F = \frac{2VP}{2VP + FP + FN} \tag{4.8}$$

5 | Perceptrón multicapa

Intro

Vacio, si vacio.

XOR

Vacio no vacio.

Propagación hacia adelante manual

Vacio no vacio.

Propagación hacia adelante vectorizada (con matrices)

Vacio no vacio.

Interpretación matemática del mapeo no lineal

Vacio no vacio, pero si vacio :P.

Propagación hacia adelante para el perceptrón multicapa

Cuasi vacio.

6 | Entrenamiento por retropropagación

Esquema general entrenamiento

Vacio pero no vacio.

Función de error: Entropía cruzada

Vacio pero no vacio.

Derivada de la función logística

Vacio, pero no vacio :P

Entrenamiento en la última capa

Vacio no vacio pero si vacio :P

Parcial con respecto a los pesos en la última capa

Vacio >:|

Vectorización

La vectorización.

7 | Optimización del entrenamiento

Problemas en redes profundas

Vacio.

Gradiente desvaneciente (o que explota)

Vacio.

Estilos de entrenamiento

Vacio.

Normalización y normalización por lotes

Vacio.

Regularización

Vacio.

8 | Caso de análisis e interpretación

Red Hinton árbol familiar con numpy (entrenamiento)

Vacio.

Red Hinton árbol familiar con pytorch

Vacio.

9 | Entrenamiento con genéticos

Algoritmos genéticos

Vacio.

Neuroevolución

Vacio.

Antecedentes: Aprendizaje por refuerzo en videojuegos

Vacio.

Arquitectura para estimar la función de recompensa

Vacio.

Entrenamiento

Vacio.

10 | Mapeos autoorganizados

Introducción

Vacio.

Aprendizaje no supervisado

Vacio.

Mapeos autoo-organizados

Vacio.

Kohonen

Vacio.

11 | Redes Neuronales Convolucionales

Convolución

Redes Convolucionales

Softmax

MNIST

Parte III
Redes con ciclos

12 | Redes Neuronales Recurrentes

Derivadas ordenadas

Retropropagación en el tiempo

Sistemas dinámicos y despliegue del grafo

Arquitectura recurrente universal

Función de error

Forzamiento del profesor

13 | Atención

14 | LSTM

15 | GRU

16 | Casos de análisis: etiquetado de palabras y conjugación de verbos

Parte IV
Redes no dirigidas

17 | Redes de hopfield

Entrenamiento

18 | Máquinas de Boltzman

Entrenamiento

Partículas y partículas de fantasía

Máquinas de Boltzman Restringidas

19 | Redes adversarias

GANs

A | Ecuaciones diferenciales

Bibliografía

Waldeyer-Hartz, Wilhelm von (1891). «Forschungen im Gebiete der Anatomie des Zentralnervensystems. Deutsche medizinische Wochenschrift 1891; 17 (44)». En: (Sobre algunas investigaciones recientes en el campo de la anatomía del sistema nervioso central)volumen 17, número 44.

Doidge, Norman (2007). *The Brain That Changes Itself: Stories of Personal Triumph from the Frontiers of Brain Science*. Penguin Life; Reprint edition.

Francisco López-Muñoz, Jesús Boya y Cecilio Alamo (2006). «"Neuron theory, the cornerstone of neuroscience, on the centenary of the Nobel Prize award to Santiago Ramón y Cajal"». En: *Brain Research Bulletin. 70 (4–6): 391–405. doi:10.1016/j.brainresbull.2006.07.010*.

John G. Nicholls, A. Robert Martin (2011). *From Neuron to Brain*. Sinauer Associates Inc.

Kandel, Eric R. (2012). *Principles of Neural Science, Fifth Edition*. McGraw-Hill Education.

Mesulam, M Marsel (1998a). *From sensation to cognition*. Brain 121.

– (1998b). «From Sensation to Cognition». En: *Brain* 121, págs. 1013-1052.

Snell, Richard S. (2001). *Neuroanatomía Clínica*. En español. 3pp,50pp, varias más. Editorial medica panamericana.

– (2009). *Clinical Neuroanatomy*. 3pp. Lippincott Williams y Wilkins.

Trappenberg, Thomas P. (2010). *Fundamentals of Computational Neuroscience*. Oxford University Press Inc.