

0. INTRODUCCIÓN

Definiciones y enfoques

La inteligencia (general) es "la capacidad de entender, asimilar, elaborar información y utilizarla para resolver problemas" (Wikipedia).

John McCarthy - 1956: La IA es la ciencia y la ingeniería de hacer máquinas inteligentes, pero especialmente de hacer programas inteligentes. Está relacionado también con el uso de máquinas para mejorar la comprensión de la inteligencia humana.

Para Inteligencia Artificial las definiciones de los autores actuales varían en torno a dos dimensiones principales:

- Razonamiento-conducta
- Eficiencia humano-ideal

Para nuestra materia: "El estudio de cómo lograr que las computadoras realicen tareas que, por el momento, los humanos hacen mejor". - *Elaine Rich y Kevin Knight*.

Eficiencia humano-ideal	
razonamiento - conducta	Sistemas que piensan como humanos
	Sistemas que piensan racionalmente
razonamiento - conducta	Sistemas que actúan como humanos
	Sistemas que actúan racionalmente

La inteligencia artificial es importante porque:

- Automatiza el aprendizaje y descubrimiento repetitivos a través de datos
- Agrega inteligencia a productos existentes
- Se adapta a través de algoritmos de aprendizaje progresivos para permitir que los datos lleven a cabo la programación
- Analiza más datos y datos más profundos utilizando redes neurales que tienen muchas capas ocultas
- Logra una precisión increíble a través de redes neuronales profundas
- Saca el mayor provecho de los datos

¿Por qué es interesante la IA?

- **Sentido de la identidad:** El hombre se ha aplicado a sí mismo el nombre científico de homo sapiens ("capaz de conocer") como una valoración de la trascendencia de nuestras habilidades mentales tanto para la vida cotidiana como en el propio sentido de la identidad.
- **Comprensión de entidades inteligentes:** A diferencia de la filosofía y de la psicología, que también se ocupan de la inteligencia, los esfuerzos de la IA están encaminados tanto a la comprensión de entidades inteligentes como su construcción.
- **Es un problema complejo, pero... la solución existe:** El investigador del campo de la IA cuenta con pruebas contundentes de que tal búsqueda es factible, no solo teóricas.
- **El estudio de la inteligencia es una de las disciplinas más antiguas:** Por más de 2000 años los filósofos se han preguntado cómo se ve, aprende, recuerda y razona. La llegada de las computadoras a principio de la década de los cincuenta permitió pasar de la especulación, a su abordaje mediante una auténtica disciplina teórica y experimental.

- **¿Qué pasará en el futuro?** Las computadoras que posean inteligencia a nivel humano (o superior) tendrán repercusiones muy importantes en nuestra vida diaria así como en el devenir de la civilización. El caso extremo es la *singularidad tecnológica*. Un punto en el tiempo donde, para algunos pensadores, la inteligencia artificial superará a la humana.
- **Otras razones más mundanas:** incremento de la demanda para perfiles de data scientist.

Fundamentos

Si bien la IA es un campo joven, es heredera de diversas ideas, puntos de vista y técnicas de otras disciplinas. Estas disciplinas son:

- Filosofía
- Matemáticas
- Psicología
- Neurología
- Computación
- Lingüística

Breve historia

En 1936 Alan Turing crea la Máquina de Turing, la base de las computadoras.

En 1943, Warren McCulloch y Walter Pitts crean la primera neurona artificial.

Alan Turing, consciente de los alcances de su máquina, define en 1950 el Test de Turing. Una prueba para determinar si una máquina es inteligente (como un humano).

En 1956, John McCarthy convocó el Dartmouth workshop, una reunión de dos meses de duración a la que asistieron cerca de 10 científicos que trabajaban en temas relacionados con "máquinas pensantes". Estuvieron, entre otros, Marvin Minsky, Nathaniel Rochester y Claude Shannon. En esta reunión se acuñó el nombre "Inteligencia Artificial".

La primera década estuvo marcada por un optimismo exagerado. Se rápidamente resolvieron problemas como la demostración de teoremas y se logró que las máquinas superarán a los humanos en algunos juegos simples.

Etapas de depresión (AI winter). Cuando se intentaron atacar problemas del dominio del sentido común o aquellos que necesitan de conocimiento general, como los relacionados con el lenguaje, los esfuerzos no fueron suficientes. Esta situación llevó a un desánimo generalizado y a la pérdida de inversiones.

Después de la etapa de depresión, ya en los 80's, la IA artificial resurgió de la mano de los Sistemas Expertos (los veremos en el 2do cuatrimestre). Desde ese momento el área no deja de crecer. Los sistemas de visión artificial y los de reconocimiento de patrones han provisto innumerables soluciones en los últimos años. La cantidad de datos disponibles, los recursos computacionales y los avances como el de las redes neuronales profundas (deep learning) han formado un círculo virtuoso donde tanto las empresas como la comunidad científica invierten cada vez más recursos. Los resultados están a la vista.

Corrientes

Desde los inicios de la IA se plantearon dos enfoques diferentes, uno por parte de los investigadores del MIT y otro por los del Carnegie-Mellon. Estos caminos se fueron alejando hasta llegar a plantearse un distanciamiento que caracterizó las investigaciones en IA de los veinte años siguientes y que fue sin lugar a dudas muy perjudicial por el atraso que provocó en algunas líneas de investigación.

- **Corriente conexionista - subsimbólica - ascendente** (Carnegie-Mellon)

Proponía desarrollar modelos del comportamiento humano a partir de elementos que simularan en todo lo posible al cerebro con un enfoque claramente cognitivo. Así iniciaron una corriente denominada "de aproximaciones subsimbólicas" que sigue un diseño de modelo "ascendente", comenzando en el nivel más bajo y operando hacia niveles superiores.

Según esta corriente, para concebir máquinas inteligentes hay que seguir muchos de los pasos evolutivos que acompañaron el desarrollo de la inteligencia en los seres vivos. De esta forma hay que comenzar por replicar la capacidad de procesamiento de señales y subir por la cadena evolutiva en pasos sucesivos, construyendo progresivos niveles superiores de inteligencia que constituirán las bases sólidas para los siguientes pasos. Al amparo de esta propuesta se desarrollaron los modelos conexionistas, redes neuronales inspiradas en los modelos biológicos.

- **Corriente tradicional - simbólica - descendente** (Minsky y McCarthy del MIT)

Orientó su actividad a obtener comportamientos "inteligentes" sin procurar que la estructura de los componentes tuviese semejanza con sus equivalentes biológicos y llegó a su mayor auge en los primeros 20 años de la IA. Esta escuela iniciada en el MIT fue denominada "tradicional", "de procesamiento de símbolos" o "de diseño descendente" y se apoyó en representar el conocimiento mediante sentencias declarativas y en la deducción de sus consecuencias a partir de la aplicación de reglas de inferencia.

Sin embargo, poco a poco se fue comprobando que muchas de las soluciones propuestas eran apropiadas para modelos pequeños, pero completamente ineficientes o inaplicables cuando se pretendía resolver problemas de tamaño real. Los investigadores se encontraron con que sus sistemas sucumbían ante la creciente longitud y complejidad de su programación.

Presente

Universidades y empresas, muchas veces en conjunto, no dejan de romper barreras. Las tareas que son realizadas mejor por los humanos que por las máquinas se reducen vertiginosamente. Autos que se manejan solos, máquinas que responden preguntas de conocimiento general mejor que una persona, reconocimiento de objetos con precisión superior a la humana, etc.

1. PROCESAMIENTO DE IMÁGENES

El **Procesamiento de Imágenes** es un área del conocimiento que se ocupa de los algoritmos que permiten realizar transformaciones sobre una imagen digital. La Visión Computarizada, va más allá, haciendo uso de estos algoritmos como base, integrándose con sistemas apropiados de hardware.

El área de Visión Computarizada introduce sus propios algoritmos. Estos últimos constituyen una frontera de la ciencia y podríamos intentar clasificarlos en dos tipos diferentes:

- Algoritmos orientados a aplicaciones prácticas.
- Algoritmos orientados a actividades de investigación.

1.1 Conceptos básicos

Una **imagen** es una representación visual de un objeto iluminado por una fuente radiante. No es un objeto real en sí mismo sino una representación que se obtiene mediante un proceso visual.

Nuestros ojos ven los objetos en *imágenes analógicas*. La intensidad de una imagen analógica puede ser considerada una función continua $f(x,y)$ en un espacio continuo de dos dimensiones.

Para que esta imagen pueda ser procesada por un computador debe ser previamente digitalizada. La digitalización es resuelta mediante dos operaciones principales:

- **Muestreo de intensidades** correspondiente a una matriz de $R \times C$ puntos. El módulo del circuito que efectúa esta operación es denominado Sample and Hold.
- **Cuantificación de los niveles de intensidad** continuos en niveles de gris por el conversor A/D. Siendo k el número de bits por muestra de la imagen. Cada punto muestreado se denomina píxel (de las palabras en inglés Picture Element).

La imagen digital queda así representada por $R \times C$ pixels, cada uno codificado con k bits. *Una vez que la imagen ha sido digitalizada se pueden aplicar los algoritmos de Visión Computarizada mencionados.*

La naturaleza básica de una imagen representada por una función de dos variables independientes está caracterizada por dos componentes: **iluminación** (cantidad de luz incidente) y **reflectancia** (cantidad de luz reflejada).

Una imagen digital puede ser representada por una matriz $n \times m$:

Cada elemento de la matriz es un píxel y nos da idea de la intensidad de la imagen en ese punto. Es decir, para cada valor de posición (x,y) de la imagen obtenemos un valor de intensidad que se vuelca en la matriz.

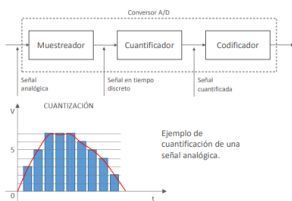
$$f = \begin{bmatrix} f(1,1) & \cdots & f(1,m) \\ \vdots & \ddots & \vdots \\ f(n,1) & \cdots & f(n,m) \end{bmatrix}$$

↑
Intensidad lumínica en la posición (x,y)

Imagen digital: es una matriz donde cada elemento se llama píxel.

Pixel: Acrónimo de Picture Element . Es la menor unidad homogénea en color e intensidad de una imagen digital.

- El valor mínimo de intensidad es cero (negro)
- El valor máximo de intensidad (blanco) depende de la codificación/escala utilizada.
- Se "direccionan" por las coordenadas que tienen en la imagen.



Ejemplo de cuantificación analógica con un conversor A/D. *La cuantificación reduce la calidad de la imagen y hace que se pierda información.*

Muestreo: proceso de un conversor A/D de tomar muestras de la señal a intervalos periódicos. Luego se cuantifica y se codifica.

Formato PGM (Portable Gray Map)

Es un formato de imagen digital diseñado para ser fácil de comprender y manipular por cualquier lenguaje. Se emplea usualmente con fines didácticos o experimentales. Una imagen PGM representa una imagen de escala de grises.

Cada línea del archivo se trabaja como si fueran los elementos de una matriz.

Como ejemplo de valores de parámetros de una imagen:

Niveles de gris: 256 (Donde por ejemplo $g = 0$ = negro y $g = 255$ = blanco)

Formato PGM

```
P2                                     #código del tipo de formato PGM
#diagonal.pgm                         #nombre comentado
10 10                                 #ancho x alto de la imagen en píxeles
255                                  #niveles de grises
0 255 255 255 255 255 255 255 255 255
255 0 255 255 255 255 255 255 255 255
255 255 0 255 255 255 255 255 255 255
255 255 255 0 255 255 255 255 255 255
255 255 255 255 0 255 255 255 255 255
255 255 255 255 255 0 255 255 255 255
255 255 255 255 255 255 0 255 255 255
255 255 255 255 255 255 255 0 255 255
255 255 255 255 255 255 255 255 0 255
255 255 255 255 255 255 255 255 255 0
```

Se convierte a escala de grises porque es conveniente tener un solo valor para cada píxel (intensidad). En las imágenes a color cada píxel está representado por una terna rgb que son como en capas superpuestas. Tener la imagen en escala de grises **no le quita información** y permite, desde el punto de vista algorítmico concentrarnos más en el procesamiento.

Relaciones básicas entre píxeles

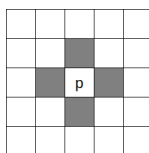
Se requiere poder transformar las imágenes mediante una gran cantidad de métodos existentes para este fin. Estos métodos efectúan diversas transformaciones a partir del cálculo de relaciones matemáticas entre un píxel y los que lo rodean.

• Vecindad

Es el conjunto de píxeles adyacentes a un píxel dado.

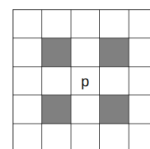
4-vecinos de p o $N_4(p)$: tienen en común un borde.

Un píxel p con coordenadas (x,y) tiene dos vecinos horizontales y 2 verticales cuyas coordenadas son $(x+1,y)$, $(x-1,y)$, $(x,y+1)$ y $(x,y-1)$.

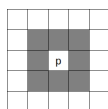


D-vecinos de p o $N_D(p)$: solo tocan al píxel central en la esquina.

Los píxeles adyacentes ubicados en diagonal con respecto a p tienen las coordenadas $(x+1,y+1)$, $(x-1,y-1)$, $(x+1,y-1)$ y $(x-1,y+1)$.



8-vecinos de p o $N_8(p)$: un borde en común, incluyendo los diagonales. conjunto formado por $N_4(p) \cup N_D(p)$



6-vecinos de p o $N_6(p)$: a la 4-v se le agregan los píxeles opuestos sobre una de las diagonales

La vecindad adoptada en un paso particular es determinante para considerar válido el resultado que se desea obtener. Por ejemplo, la vecindad puede significar que dos áreas están separadas o conectadas. Adoptar una vecindad es frecuentemente un criterio inicial del diseño de un sistema.

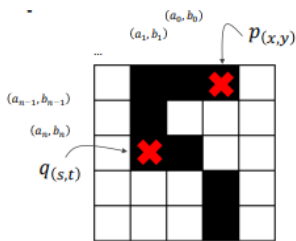
- **Conectividad**

Para establecer que dos píxeles p y q están conectados es necesario que:

- p y q sean vecinos para alguna vecindad definida.
- p y q cumplan con algún criterio de semejanza.

Criterio de semejanza: Se define un conjunto V de intensidades de píxeles. Los píxeles que tengan intensidades en V se consideran semejantes.

- **Camino**



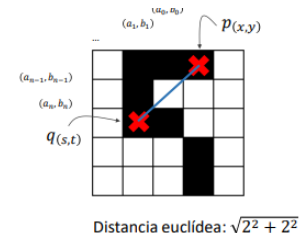
Un camino desde el píxel p, con coordenadas (x,y) al píxel q con coordenadas (s,t) es una sucesión de píxeles con coordenadas $(a_0,b_0), (a_1,b_1), \dots, (a_n,b_n)$ donde $(a_0,b_0)=(x,y)$ y $(a_n,b_n)=(s,t)$, estando (a_i,b_i) conectado a (a_{i+1},b_{i+1}) .

De acuerdo a esta definición, la longitud del camino es n. Si no hay píxeles que conecten a p y a q, no se puede establecer un camino entre ellos.

- **Distancia**

La distancia del píxel p al píxel q es la cantidad de píxeles que hay que recorrer para llegar a q partiendo desde p.

Ejemplos de distancia:



Distancia euclídea: $\sqrt{2^2 + 2^2}$

La distancia D4 entre el píxel p, con coordenadas (x,y) y el píxel q con coordenadas (s,t) es: $D4(p,q)=|x-s|+|y-t|$

4	3	2	3	4
3	2	1	2	3
2	1	0	1	2
3	2	1	2	3
4	3	2	3	4

La distancia D8 entre el píxel p, con coordenadas (x,y) y el píxel q con coordenadas (s,t) es: $D8(p,q)=\max(|x-s|,|y-t|)$.

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

La distancia euclídea entre el píxel p, con coordenadas (x,y) y el píxel q con coordenadas (s,t) es:

$$D(p,q) = \sqrt{(x-s)^2 + (y-t)^2}$$

- **Contorno**

Los contornos en una imagen se definen como variaciones locales en la intensidad de la imagen. Representa un cambio de la intensidad de los niveles de gris presentes en ella. El paso de nivel oscuro a uno brillante, o viceversa, determinan un contorno.

Determinar un buen contorno en una imagen depende de la fuente de radiación, la iluminación y la distancia a la que se encuentra el objeto de la fuente de radiación.

En este sentido, es necesario aplicar a la imagen operaciones de filtrado que realcen los cambios en los valores de gris y atenúen las áreas de la imagen donde existan valores de gris constantes, para posteriormente introducir el resultado de esta operación a un detector de borde por umbral.

Histograma

Es la distribución de frecuencias relativas de los niveles de gris de esa imagen.

La escala de niveles de gris puede ser representada en forma normalizada variando entre 0 y 1, o bien con valores enteros. El histograma de niveles de gris de una imagen digital corresponde a la distribución de probabilidad de intensidad de la imagen. La cantidad de información almacenada en el histograma resulta sólo una fracción de la imagen original.

El histograma se arma barriendo la imagen de izquierda a derecha, de arriba a abajo, y se realiza el conteo de la cantidad de veces que aparece cada nivel de gris.

El histograma de una imagen digital con niveles de gris en el rango $[1, L]$ es una función discreta con valores: $h(g_i) = \frac{n_i}{n}$ donde: $i = 1, 2, 3, \dots, L$; g_i = i-ésimo nivel de gris; n_i = es la cantidad de píxeles en la imagen con el nivel de gris i ; n = es la cantidad total de píxeles de la imagen.

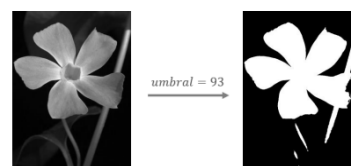
¿Para qué sirve el histograma?

- $h(g_i)$ es la probabilidad de ocurrencia del nivel de gris g_i .
- No da información sobre el contenido de la imagen, pero sí sobre su contraste y otras características.
- Binarización (detección del umbral óptimo).
- Es empleado para aplicar un procedimiento de mejoramiento de la imagen denominado ecualización o igualación de Histograma.
- El histograma sirve para mejorar la imagen y aumentar el contraste.
- A partir del histograma se pueden calcular el valor medio (m_g) y la desviación estándar de los niveles de gris.

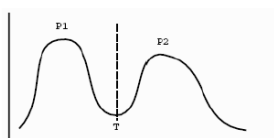
$$m_g = \sum_{g=0}^n g * h(g) \quad \leftarrow \begin{array}{l} \text{nivel de gris máximo} \rightarrow n \\ \text{nivel de gris} \end{array} \quad \leftarrow \begin{array}{l} \text{cantidad de píxeles} \\ \text{con ese nivel de gris} \\ \text{(normalizada)} \end{array}$$
$$\sigma = \sqrt{\sum_{g=0}^n (g - m_g)^2 * h(g)}$$

Binarización: Se denomina imagen binaria a una imagen que solo posee dos tonos, 0 y 1, o simplemente blanco y negro.

Las imágenes binarias se obtienen en principio fácilmente “gatillando” los valores de intensidad de la imagen con respecto a un valor umbral.



De 0 hasta ese valor umbral va a ser un 0 (negro) y del valor umbral para arriba va a ser 1 (blanco). Para calcular el valor umbral de manera óptima va a depender de las características de la imagen que yo quiero binarizar. El valor más adecuado de binarización es el llamado **umbral óptimo**.



Existe un caso particular llamado histograma bimodal. De este tipo de imagen y a partir de su histograma puede obtenerse fácilmente la imagen binaria, para esto se emplea como valor de umbral el valor mínimo dado por el histograma entre los picos.

Otras adaptaciones de esta idea básica calculan el umbral con la media, o permiten procesar las imágenes por vecindades de modo de optimizar localmente el umbral de gatillado para cada zona de la imagen.

Frecuencia espacial: Es una referencia al gradiente de intensidad en una dirección en ese punto. Para los casos donde los niveles de gris cambian de forma abrupta, esta se determina al recorrer horizontalmente una secuencia de píxeles.

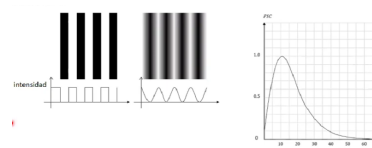
Contraste

Está relacionado con la sensibilidad del ojo a los distintos niveles de intensidad en áreas adyacentes de la imagen. Si se perciben las diferencias, la imagen tiene el contraste suficiente. El contraste se describe frecuentemente de dos maneras:

Como función de la frecuencia espacial → Función de sensibilidad al contraste (FSC)

La función de sensibilidad al contraste nos indica cuán sensitivos somos al estímulo visual de distintas frecuencias. Si la frecuencia del estímulo visual es demasiado grande no es posible reconocer el patrón del estímulo.

Imaginemos una imagen consistente en bandas verticales negras y blancas. Si las bandas son muy delgadas (i.e. miles por milímetro) no nos será posible distinguirlas individualmente. Todo lo que vemos es una imagen gris. Si las bandas se hicieran cada vez más anchas, habrá un espesor umbral, es posible distinguir las bandas.



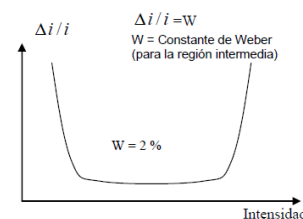
Como función de la intensidad → Constante de Weber

La respuesta del ojo al contraste no es lineal. La mínima diferencia de intensidad que puede detectar el ojo es pequeña para valores pequeños de f y crece proporcionalmente con f para valores altos de intensidad.

Definimos el contraste en función de la intensidad como la relación entre la intensidad en un punto de la imagen y la variación de intensidad con respecto a un punto adyacente. Para valores intermedios de intensidad esta relación se suele considerar constante y queda definida por un valor denominado constante de Weber.

Para observar un objeto sobre el fondo, necesitaremos contraste:

- Si las intensidades son muy altas (imágenes y objetos brillantes)
Ej: en una playa al rayo del sol, si iluminamos con una linterna no podemos distinguir la luz de la misma.
- Si las intensidades sean muy bajas (imágenes y objetos oscuros), también necesitamos mayor contraste para detectar los objetos. Ej: en la noche, en la oscuridad no podemos distinguir entre un árbol y una piedra por ejemplo.



Para este caso sirve la ecualización. Ley de Weber (1860). "El menor cambio discernible en la magnitud de un estímulo es proporcional a la magnitud del estímulo" $\Delta_x = k_w x$

Cuantización

La intensidad de una imagen se cuantifica en niveles de gris, suele ser lineal, pero para resaltar el contraste puede emplearse cuantización no lineal.

Para reducir el almacenamiento puede cuantificarse la imagen con menos de 8 bits perdiendo poca información, ya que: la pérdida de información es conocida a partir de los 5 bits, los bits menos significativos 0 y 1, generalmente contienen ruido.

1.2 Procesamiento

Ecualización del histograma

Este método justifica su aplicación en imágenes de bajo contraste con áreas grandes de niveles medios de gris. *Aumenta el contraste de la imagen, permite aplicar luego otras herramientas, como reconocimiento de patrones o bordes.*

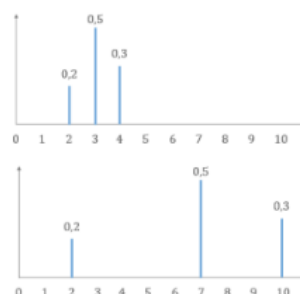
El método consiste en una transformación de niveles $g \rightarrow g'$. Reacomodando las líneas del histograma con nuevos intervalos Δg de manera de obtener niveles de densidad correspondientes entre todos los valores de g y g' . La suma de la distribución de $h(g)$ normalizada resulta: $\sum h(g) = 1$

Luego obtenemos una constante de proporcionalidad k : $k = \text{niveles de grises} / \sum h(g)$. Si está normalizado, directamente los niveles de grises es la constante de proporcionalidad.

Ahora pueden calcularse los valores de los intervalos armando la siguiente tabla:

- Columna 1: g (previo)
- Columna 2: $h(g)$
- Columna 3: $\Delta g = k * h(g)$
- Columna 4: g' (nuevo)

g (previo)	$h(g)$	$k * h(g)$	g' (nuevo)
0	0	0	0
1	0	0	0
2	0.2	2	2
3	0.5	5	7
4	0.3	3	10
5	0	0	10
6	0	0	10
7	0	0	10
8	0	0	10
9	0	0	10



Las frecuencias no cambian, sino que se distribuyen los niveles de gris, es decir, se modifican los intervalos.

Filtros de la mediana y del valor medio

Estos filtros calculan un nuevo valor de píxel en una imagen transformada en función de la vecindad local de la imagen original. *Mientras mayor sea la vecindad, más información se pierde.* Se aplican en el dominio del espacio (aplico máscaras) y de la frecuencia (con Transf. de Fourier)

¿Para qué se usan? •Suavizar una imagen. •Eliminar ruido. •Realzar una imagen. •Detectar bordes.

Filtro Mediana:

- Elimina el ruido. *Es muy bueno con ruidos tipo sal y pimienta.*
- Para aplicarlo se consideran todos los píxeles de la vecindad, se ordenan y se selecciona el del medio, este valor reemplaza al original en la imagen transformada.
- Desventajas:
 - No son lineales: Dadas dos imágenes A y B , $\text{med } A + B \neq \text{med } A + \text{med } B$.
 - Pierden detalles (puntos, líneas finas).
 - Redondean las esquinas de los objetos.

Filtro Media:

- Suaviza la imagen.
- Es un filtro convolucional porque se aplica como una sumatoria de la contribución o peso que ejerce cada píxel de la imagen respecto al píxel centra.
- Reemplaza un píxel por el promedio de los que se encuentran en la vecindad.
- Desventajas:
 - Son bastante sensibles a cambios locales.
 - Será necesario aplicar algún criterio extra de redondeo.

$$P_{xy} = \frac{1}{9} \sum_{i=-1}^1 \sum_{j=-1}^1 P_{x-i, y-j}$$

Convolución

La convolución discreta (o, en el contexto de la materia, convolución), al igual que la convolución de funciones, es un operador matemático. Por lo tanto no tiene sólo una aplicación, sino que, de la misma forma que el producto o la derivación, se puede utilizar con distintos propósitos prácticos.

- **Definición**

Sea W una ventana de tamaño $n \times m$ (según la vecindad definida) sobre una imagen y M una máscara

$$conv = \sum_{i=1}^n \sum_{j=1}^m W_{ij} M_{ij}$$

(matriz) del mismo tamaño. La convolución digital entre W y M se define

El resultado de la convolución, es la **suma ponderada** entre el valor de los píxeles vecinos a un píxel central. En procesamiento de imágenes, la convolución se realiza entre los valores de los píxeles de una imagen y una matriz que contiene los valores de un filtro. Dicha matriz, se llama Máscara.

- **Procedimiento**

Se recorre la imagen con una ventana W y para cada posición se calcula un nuevo valor para el pixel central. El cálculo se hace multiplicando cada elemento de la imagen bajo la ventana, por el elemento de la máscara M que está en la misma posición y luego sumando todos los productos.

Gráficamente, el cálculo para la primera posición de la ventana \rightarrow

$$M = \begin{bmatrix} a & b & c \\ h & p & d \\ g & f & e \end{bmatrix}$$

donde $z=1a+2b+3c+4d+5e+6f+7g+8h+9p$

1	2	3	15	0	0	14	12	10	15	4	13	6
8	9	4	2	5	14	10	14	4	16	8	15	7
7	6	5	4	6	5	5	13	10	10	1	10	11
15	5	14	7	3	7	12	1	7	5	11	7	15
13	8	10	0	10	10	11	5	15	13	4	15	1
9	1	6	16	4	16	12	4	15	8	8	10	4
5	16	4	10	11	12	3	3	3	16	15	9	12
5	1	13	14	15	5	2	2	12	15	11	9	14

A 10x10 grid with a red arrow pointing down to the top-left cell, which contains the letter 'z'.

Algunas consideraciones:

- El resultado se guarda en una **estructura auxiliar** (no deben usarse en el cálculo otros).
- Los **bordes** no se pueden calcular, cómo completarlos depende del objetivo de la conv.
 - Se mantienen los valores originales (puede generar distorsión)
 - Agregar 0 en los bordes (no influye en el resultado)
- El **resultado** de la convolución **no es una imagen**. Aunque tiene la misma forma, sus valores en la mayoría de los casos no son intensidades válidas.
- La transformación realizada por la convolución, y en consecuencia la aplicación práctica, **dependen exclusivamente de la máscara** elegida (tamaño y valores).
- Como el resultado de aplicar la operación de convolución es una suma ponderada, los valores de la máscara son los **pesos** que se le asignan a los píxeles de la imagen.
- Algunas de sus aplicaciones son:
 - Detección de bordes
 - Suavizado
 - Filtrado (algunos tipos). Dependiendo de los valores que tenga la máscara, es el tipo de filtro que se le aplica a la imagen.

Ejemplo: la máscara para lograr el filtro de media con convolución sería:

$$P_{xy} = \frac{1}{9} \sum_{i=-1}^1 \sum_{j=-1}^1 P_{x-i y-j}$$

20	30	25
40	23	15
67	40	38

20	30	25
40	33	15
67	40	38

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

Detección de bordes

Los **bordes** son partes de la imagen con alta frecuencia espacial, en otras palabras, con niveles de intensidad que cambian bruscamente. Estas características se pueden representar por medio del gradiente de intensidad de la imagen.

El gradiente de la intensidad de la imagen se puede estimar a través de la convolución con algunas máscaras diseñadas para ese propósito.

Ejemplo de máscaras más utilizadas:

Prewitt

1	0	-1
1	0	-1
1	0	-1

1	1	1
0	0	0
-1	-1	-1

Sobel

1	0	-1
2	0	-2
1	0	-1

1	2	1
0	0	0
-1	-2	-1

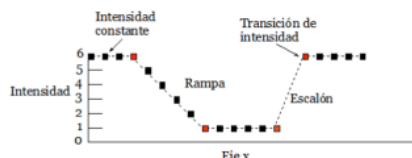
Las dos máscaras superiores se usan para detectar bordes verticales, mientras que las dos inferiores para bordes horizontales.

La máscara de Prewitt conviene ser utilizadas en imágenes con más ruido ya que no lo va a realzar tanto, mientras que la de Sobel debería ser utilizada en imágenes con poco ruido, sino, al tener como valor el 2, va a resaltar mucho dichos píxeles que introducen ruido.

El proceso es sencillo, se toma una de las máscaras y se aplica la convolución. El resultado será una estructura con la misma forma de la imagen, pero con valores positivos y negativos. Sobre estos valores se calcula el valor absoluto. Después, los valores cercanos a cero serán zonas con baja frecuencia espacial (sin bordes) y los valores grandes serán bordes. Para este caso, es conveniente completar con ceros las zonas donde no se puede realizar la convolución (límites de la imagen).

Detección de bordes - Características:

- Operador local de derivación: el borde existe cuando se produce un cambio brusco en los niveles de intensidad de un píxel con sus vecinos. Mientras más brusco sea el cambio, más fácil es detectar el borde. *Cambio → derivadas.*
- Inconveniente: El ruido es colateralmente realzado. El ruido es un píxel que difiere mucho en intensidad con los píxeles vecinos y el detector de bordes hace eso, resalta los cambios bruscos de intensidad, por ende, resalta los ruidos.



6	6	6	5	4	3	2	1	1	1	1	1	6	6	6	6
0	0	0	-1	-1	-1	-1	0	0	0	0	0	5	0	0	0
0	0	-1	0	0	0	0	1	0	0	0	0	5	-5	0	0

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} \quad \begin{aligned} G_x &= \frac{\partial f(x,y)}{\partial x} \\ G_y &= \frac{\partial f(x,y)}{\partial y} \end{aligned}$$

0	0	0
-1	1	0
0	0	0

0	-1	0
0	1	0
0	0	0

Gradiente digital:

Tiene que ver con la dirección de cambio. Permite saber la dirección x e y donde la imagen tiene un cambio brusco.

Las máscaras pueden aplicarse por separado para obtener los bordes en ambas direcciones independientemente (sean G_x y G_y).

Los bordes horizontales y verticales se calculan de la misma forma (con diferentes máscaras). Si queremos encontrar todos los bordes, sin importar la orientación, pueden combinarse para obtener el

valor del módulo $|\nabla f| = \sqrt{G_x^2 + G_y^2}$ y la orientación del borde está dada por: $\theta = \arctg\left(\frac{G_y}{G_x}\right)$

Posprocesamiento de bordes

En ciertos casos las líneas de borde obtenidas deben ser posprocesadas. Para solucionar problemas como ramificaciones, cortes y ambigüedades por bordes anchos.

Los bordes reales de los objetos, al ser detectados con un detector de bordes, resultan en un conjunto de bordes aislados y con ramificaciones. Por este motivo las ramificaciones y las interrupciones deben ser eliminadas.

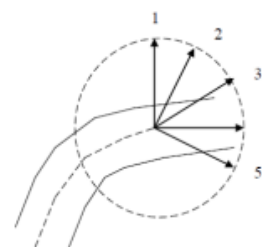
En el caso particular de una imagen de valores de grises, ocurre que generalmente los bordes de niveles de gris:

- Tienen valores máximos fluctuantes
- Poseen interrupciones

Técnicas de posprocesamiento

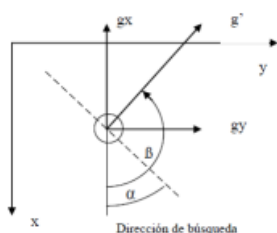
- Búsqueda en arco del pixel sucesor

Una regla simple para eliminar interrupciones entre los bordes, obtenidos a partir de imágenes de tonos de gris es elegir como pixel sucesor al vecino con mayor nivel de gris. Consiste en efectuar una búsqueda en una vecindad ampliada (mayor que la vecindad ocho).



Se define un espacio de búsqueda como un sector limitado a ambos lados de la dirección principal del borde en su extremo. El pixel sucesor queda determinado por el rayo de búsqueda con el mayor valor medio. Como resultado de esto último las fluctuaciones de niveles de gris en el borde son relativamente neutralizadas.

Es el píxel más homogéneo con el central en intensidad, textura o alguna otra característica. Si no encuentro alguno alrededor que sea homogéneo, se busca hacia píxeles más lejanos, pero para eso se debe definir un límite de búsqueda y análisis.



- Búsqueda con dirección controlada por gradiente

El píxel inicial debe de pixel de borde. La dirección principal de búsqueda puede calcularse de los gradientes de un detector de bordes local en el pixel investigado. Esta dirección puede describirse por el ángulo relativo al eje x. El píxel sucesor queda determinado por este vector de búsqueda.

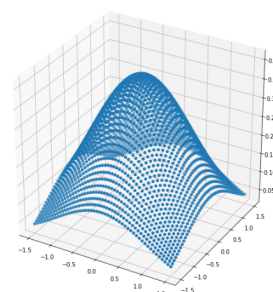
Suavizado

Otra aplicación de la convolución es la operación de "suavizado".

El objetivo del suavizado de una imagen es moderar los gradientes de las zonas de alto gradiente, es decir, hacer los bordes menos nítidos o más borrosos. *Importante en el reconocimiento de patrones.*

El método consiste simplemente en hacer la convolución entre la **imagen** y una **máscara gaussiana** (máscara con valores discretos de la campana de Gauss).

La máscara toma valores con distribución gaussiana. Para que la convolución no modifique el rango de las intensidades, se divide cada elemento de la máscara por la suma de todos ellos. (Suma = 1)



Segmentación

Es el proceso de extraer (identificar y aislar) regiones de interés en una imagen. Se obtiene una lista de regiones, que luego podrán ser medidas, analizadas y clasificadas en un nivel superior de procesamiento. Posibilita:

- Compresión de imágenes
- Aislar regiones en una imagen (diferenciar objetos en una imagen, o bien decidir una acción a ejecutar en función de estos objetos y sus características)

Dos tipos clásicos de segmentación:

- Determinación de áreas que corresponden a objetos de interés. (Separan fondo de forma)
- Determinación de perímetros de objetos de interés.

Desde este punto de vista los procesos de segmentación se pueden clasificar en dos categorías principales:

- **Orientados a regiones (áreas).**
 - Los píxeles adyacentes a una región son asignados a esa región en tanto sean homogéneos con respecto a los píxeles de la región (similares valores de intensidad).
 - *Segmentación binaria*: La imagen de grises es binarizada y luego dividida en subregiones a través de un algoritmo de segmentación binaria. Una etiqueta particular (1,2,3,...,n) es asignada a cada píxel de cada nueva región.
- **Orientados a perímetros (líneas).**
 - En este caso no se obtiene generalmente un borde cerrado que define a una región sino que debe efectuarse un postproceso para obtener dicho contorno, este postproceso comprende operaciones tales como:
 - Seguimiento de contorno
 - Conexión de bordes abiertos
 - Eliminación de bordes sobrantes

Region Aggregation

Es un algoritmo de segmentación. La técnica parte de una imagen binarizada, donde los píxeles con intensidad = 0 no pertenecen a regiones de interés y los píxeles con intensidad = 1 pertenecen (o podrían pertenecer) a regiones de interés.

Después se eligen aleatoriamente algunos píxeles que son llamados "semillas" (píxeles espaciados en la imagen, que son primeramente evaluados). Para cada semilla, evalúa la pertenencia a una región de interés (valor = 1). Si pertenece, se etiqueta con un número, si no, se descarta.

Para todas las semillas etiquetadas, se realiza un proceso iterativo donde se analizan sus 8-vecinos, por ejemplo en sentido antihorario, y los 8-vecinos de estos sucesivamente, siempre y cuando tengan intensidad = 1 en la imagen binarizada. Todos los píxeles encontrados a partir de una semilla se etiquetan con el mismo número. De esta forma, cada semilla crece y se transforma en una región de interés dentro de la imagen.

Cuando el proceso termina, se obtiene una "Imagen Etiquetada" que contiene una lista de regiones de interés con los píxeles que las integran. Sobre estas regiones se pueden aplicar algunas condiciones, por ejemplo de forma o tamaño, para determinar si realmente pertenecen a los objetos buscados en la imagen.

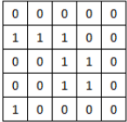

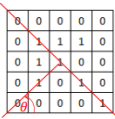
En base a una imagen, se elige un conjunto de píxeles semilla de forma aleatoria y se define un criterio para determinar si los píxeles vecinos son considerados parte de la región o no.

Se van etiquetando los píxeles que pertenecen a una misma región enumerando cada una de estas regiones. Se puede armar una pila con los pixeles que pertenecen a las regiones.

14 15 11 12 10 14 13	14 15 11 12 10 14 13	1 1 1 1 1 1 1	1 1 1 1 1 1 1
14 11 11 14 15 15 11	1 11 11 14 15 15 11	1 1 1 1 1 1 1	1 1 1 1 1 1 1
11 12 13 10 11 15 41	11 12 13 10 11 15 41	1 1 1 1 1 1 1	1 1 1 1 1 1 2
12 13 10 12 44 44 44	12 13 10 12 44 44 44	1 1 1 1 1 44 44 44	1 1 1 1 2 2 2 2
13 15 43 43 42 42 40	13 15 43 43 42 42 40	1 1 43 43 42 42 40	1 1 2 2 2 2 2 2
13 42 43 43 45 45 45	13 42 43 43 45 45 45	1 42 43 43 45 45 45	1 2 2 2 2 2 2 2
83 82 80 40 44 41 42	83 82 80 40 44 41 42	83 82 80 40 44 41 42	3 3 3 2 2 2 2 2

Invariantes geométricos

Son características de la imagen que no varían ante algunas transformaciones como el cambio de escala, la traslación o la rotación. Son muy útiles para clasificar o reconocer objetos en una imagen. Para calcularlos, partimos de una imagen binaria que contiene una región de interés, es decir, el resultado de una segmentación.

I.G.	Descripción	Ejemplo
Área	Es la superficie que ocupa la región. Se suman los valores de todos los píxeles de la imagen. Para una imagen de tamaño $n \times m$:	 $A = \sum_{i=1}^n \sum_{j=1}^m p_{ij}$ <p>El área es 7</p>
Baricentro	El baricentro o centro de gravedad se calcula como el promedio de las coordenadas de cada píxel que pertenece a la región.	$j_0 = \frac{1}{A} \sum_{i=1}^n \sum_{j=1}^m j p_{ij} \quad i_0 = \frac{1}{A} \sum_{i=1}^n \sum_{j=1}^m i p_{ij}$  $i_0 = (1/7) (2 * 3 + 3 * 2 + 4 * 2)$ $i_0 = 2,8571$ $j_0 = (1/7) (1 * 1 + 2 * 1 + 3 * 3 + 4 * 2)$ $j_0 = 2,8571$ <p>$(i_0; j_0) = (3; 3)$</p>
Perímetro	La forma más simple de calcularlo es encontrando los píxeles del contorno y sumándolos.	
Número de Euler	Es un indicador de la cantidad de espacios internos que tiene la región. Si el número de Euler de una imagen es inferior a otra, indica que la primera es más porosa. <i>Los patrones dependen de lo que se busca en la imagen.</i>	Se calcula como la diferencia entre las ocurrencias del patrón 1 (izquierda) y el patrón dos (derecha).
Spread	Es el cociente entre la suma de los momentos max y min, y el cuadrado del área. Indica cuán irregular es el contorno de una región.	$Spread = (I_{xx} + I_{yy}) / A^2$ $Spread = \frac{9 + 9}{8^2} = \frac{18}{64} = 0,28125$
Ejes ppales. de inercia	Los ejes principales de inercia de un objeto físico, son propiedades que dependen de la forma del objeto. Todos los píxeles de una región multiplicados por la distancia al eje, sumados estos valores, dan la inercia de la región respecto al eje. Para el cálculo, considerar que cualquier región pasa por el baricentro.	$I_{xx} = \sum (\bar{x} - x_i)^2 = 9$ $I_{xy} = \sum (\bar{x} - x_i)(\bar{y} - y_i) = 8$ $I_{yy} = \sum (\bar{y} - y_i)^2 = 9$
Orientación	A partir de los ejes principales de inercia se puede conocer la orientación de una región en el plano. Es uno de los métodos más robustos para determinar ángulos.	 $tg(2\theta) = \left(\frac{2I_{xy}}{I_{xx} - I_{yy}} \right)$ $tg(2\theta) = \left(\frac{2 * 8}{9 - 9} \right)$ $\theta = \frac{\pi}{4}$

Transformada de Hough

Es una técnica utilizada para detectar morfologías simples como líneas o círculos.

Para poder representar todas las posibles rectas que puedan aparecer en la imagen, podemos utilizar la ecuación de la recta en coordenadas polares: $x \cos \theta + y \sin \theta = r$.

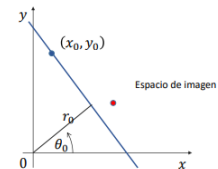
Luego, se transforma cada punto (x, y) de la imagen de origen, en los puntos (r_i, θ_i) , el espacio definido por (r, θ) se denomina **espacio de Hough** para el conjunto de rectas en dos dimensiones.

Para un punto arbitrario en la imagen con coordenadas (x_0, y_0) , las rectas que pasan por ese punto son los pares (r, θ) con $r = x \cos \theta + y \sin \theta$. Esto corresponde a una **curva sinusoidal** en el espacio (r, θ) que es única para ese punto.

Si las curvas correspondientes a dos puntos se interceptan, el punto de intersección en el espacio de Hough corresponde a una línea en el espacio de la imagen que pasa por estos dos puntos.

Generalizando, un conjunto de puntos que forman una recta, producirán sinusoides que se interceptan en los parámetros de esa línea.

Para la detección de rectas, se obtienen los parámetros de la forma normal Hessiana (r, ϕ) y el número de píxeles dispuestos sobre la recta. Los parámetros (r, ϕ) satisfacen la ecuación $r/r_A = \cos(\phi - \phi_A)$, donde (r_A, ϕ_A) describen la ubicación de A en el plano (x, y) de coordenadas polares.



Siendo A y B píxeles de un borde, se desea encontrar la línea de contorno que pasa por ambos.

Se considera el racimo de todas las rectas que pasan por A y B. Cada una de estas puede ser descrita por el ángulo ϕ y el módulo r de la normal. Para tener un número limitado L de líneas de racimo, se cuantifican los parámetros (r, ϕ) . Cada punto (r_i, ϕ_i) corresponde a una línea del racimo.

Pasos de la transformada de Hough:

1. Elegir los parámetros de cuantización $\Delta\phi$ y Δr (para limitar el racimo)
2. Para cada píxel P con coordenadas polares (r_p, ϕ_p) en la imagen B, se calculan las posiciones de L puntos de racimo cuantificados de la curva. $r/r_p = \cos(\phi - \phi_p)$ e incrementar la correspondiente cuenta de píxeles $z(r_p, \phi_p) = z(r_p, \phi_p) + 1$
3. Encontrar los píxeles (r, ϕ) con altas acumulaciones $z(r_p, \phi_p) \gg 1$. Generalmente estos puntos aparecen en clusters, los cuales indican que se han detectado varias líneas con similares parámetros (r, ϕ) . Un cluster de este tipo, podría ser causado por el objeto de la figura.

La transformada de Hough es robusta con respecto a las distorsiones. Las líneas rectas son detectadas aún si están interrumpidas o no todos los píxeles yacen exactamente sobre la línea.

Detección de otros tipos de curvas

El proceso de detección de otros tipos de curvas es similar. Cambiarán los parámetros de la ecuación y, por lo tanto también el significado, y probablemente la cantidad, de las dimensiones de la tabla de búsqueda. Por ejemplo, para detectar círculos con la función $r = \sqrt{(x - a)^2 + (y - b)^2}$ será necesario un espacio de parámetros de tres dimensiones, a , b y r . El resto del procedimiento es igual.

2. RECONOCIMIENTO DE PATRONES

2.1 Generalidades

- Reconocimiento → función básica de los organismos vivos.
 - HUMANO: Estimación del parecido relativo entre datos de entrada y pobl. conocidas.
 - AUTOMÁTICO: Clasificación de datos de entrada entre pobl. mediante la búsqueda de características o atributos invariantes entre los miembros de cada población.
- Patrón → descripción de un objeto definida como una relación entre sus características
- En la mayoría de los casos el objetivo es la clasificación.

Tarea de clasificación	Datos de entrada	Salida
Detección de spam	Correo electrónico	Spam/No spam
Reconocimiento de caracteres	Imagen	Código del carácter
Reconocimiento del audio	Ondas acústicas	Palabras
Reconocimiento de voz	Ondas acústicas	Id persona
Predicciones bursátiles	Datos de mercado	Alza/Baja de acciones
Reconocimiento facial	Imagen	Id persona
Diagnóstico médico	Síntomas e historia clínica	Diagnóstico

Términos relacionados:

- Clasificación - Discriminación - Categorización/Predicción
- Regresión - Aproximación/Predicción
- Aprendizaje automático (machine learning)

Tipos de objetos reconocidos:

- Objetos concretos (reconocimiento perceptual) → como por ejemplo una forma (patrón espacial) o una secuencia (patrón temporal).
- Objetos abstractos (reconocimiento conceptual) → como un viejo argumento o la solución de un problema.

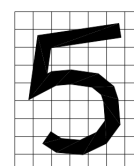
Herramienta: Scikit-learn

Es una librería de aprendizaje automático para Python. Es probablemente la más utilizada. Salvo deep learning, provee casi todo lo necesario para un proyecto de aprendizaje automático. Se suele complementar con Pandas y librerías para graficar.

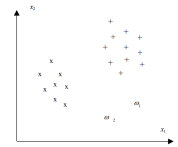
Problemas a resolver (etapas)

1. SENSADO

Obtención de datos: Es el acto de medir u obtener los valores de ciertos atributos del objeto a ser reconocido. Otros atributos frecuentemente usados en distintas áreas son: Intensidades lumínicas (imágenes), Presión sonora (audio), Texto (mails, tweets, artículos), Microarrays de expresión génica.



Representación de datos: Los datos censados se representan mediante un vector $x=(x_1,x_2,\dots,x_n)$, donde x_i es el valor del atributo i . Ej: en una imagen $\rightarrow x_i$ = intensidad de píxel i , audio $\rightarrow x_i$ = energía en el tiempo i .



- **Variables cuantitativas:** Se representan como números del tipo que sea necesario. En una imagen, las intensidades de los píxeles se pueden representar con enteros.
- **Variables categóricas:** Pueden tomar solo un valor dentro de un conjunto limitado de valores. Pueden ser ordinales (existe una escala) o nominales (no tienen un orden).

Para variables categóricas donde no existe una relación de orden, la codificación mediante enteros no suele ser adecuada. En estos casos se suele aplicar la codificación **One-Hot**, es un método en el cual se agrega una nueva variable binaria para cada valor de categoría posible. Permite etiquetar a qué clase pertenecen los datos. *Se asigna 0 a toda la dimensión, excepto 1 para la clase a la que pertenecen los datos.*

Otra transformación útil para la representación de los datos es el **escalado**. Es deseable que los valores de todas las variables se encuentren en el mismo rango. *Se suele aplicar una ecuación matemática para convertir valores entre 0 y 1. Luego para mostrar aplico nuevamente la ecuación.*

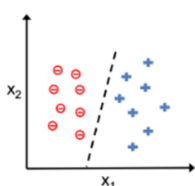
2. EXTRACCIÓN DE CARACTERÍSTICAS

Lo que se logra con esto es reducir la dimensión de los patrones pero con la posibilidad de perder un bajo porcentaje de la información contenida en ellos.

Dentro de esta etapa podemos encontrar las siguientes tareas:

- Creación de nuevas características
- Eliminación de características
- Reducción de la dimensión
- Otras transformaciones

3. CLASIFICACIÓN



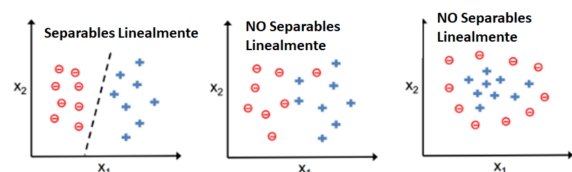
Determinar a qué clase pertenece cada vector de entrada. Para lograr esto, se requiere diseñar un mecanismo que, dado un conjunto de patrones de los cuales no se conoce a priori la pertenencia a la clase de cada uno de ellos, permita determinar a qué clase pertenece cada patrón.

Generación de límites de decisión entre regiones. \rightarrow Funciones de decisión.

Se deben generar tantas funciones de decisión como clases haya. Tomando un dato como entrada, se evalúa en las x funciones, y el valor más alto es la clase a la que pertenece. Cuando 2 funciones dan el mismo valor, el punto forma parte del límite entre las regiones.

Tipos de problemas reales

- Linealmente separables
- No linealmente separables



Si el sistema de reconocimiento puede autoajustar ciertos coeficientes internos que definen las funciones discriminantes estaremos en presencia de un sistema adaptivo o con capacidad de aprendizaje.

Clasificación de los Métodos

Definiendo a una clase como una **categoría** determinada por algunos atributos comunes y un patrón como la descripción de cualquier miembro de una categoría que representa a una clase de patrones, decimos que la teoría del Reconocimiento de Patrones se ocupa de buscar la solución al problema general de reconocer miembros de una clase dada en un conjunto que contienen elementos de muchas clases diferentes.

- Heurísticos: basados en la intuición y experiencia.
- Matemáticos:
 - Determinístico: de base estadística pero esta no está explicitada. Clasificadores entrenables.
 - Estadístico: de base estadística explícita. Ej: clasificador de Bayes.
- Sintácticos: basados en relaciones entre los objetos. Expresan una gramática. Útiles cuando los patrones no pueden ser apropiadamente descritos por mediciones (numéricamente).

Resumiendo las etapas

El resultado de las etapas de sensado y extracción de características es un conjunto de vectores de características. Cada vector tiene la forma: $\mathbf{x}=[x_1, x_2, \dots, x_n]$ donde x_i es el valor de la característica i .

- $u \rightarrow$ es un escalar.
- $\mathbf{u} \rightarrow$ (en negrita) es un vector formado por escalares u_i
- $U \rightarrow$ es una matriz formada por escalares u_{ij}

Después de obtener los vectores de características se realiza una de las siguientes tareas:

- **Clasificación.** Asignar cada vector de características a una clase. Por ejemplo, para detectar qué carácter aparece en una imagen.
- **Predicción o regresión.** Predecir un valor (continuo) para cada vector de características. Por ejemplo, para calcular el valor futuro de ciertas acciones bursátiles.

Los métodos de aprendizaje automático que vemos son clasificadores, pero con un mínimo cambio es posible adaptarlo para hacer predicciones.

Entrenamiento

Es el proceso de ajuste de los parámetros internos del método (de la función matemática) para determinar la salida.

Existen dos tipos:

- **Entrenamiento supervisado:** Los parámetros se ajustan de forma tal que la salida del clasificador se aproxime a una salida esperada conocida.
- **Entrenamiento no supervisado:** Los parámetros se ajustan para encontrar relaciones o agrupaciones entre los valores de las características.
 - **Agrupamiento:** trata de encontrar una estructura o patrón en una colección de datos no categorizados. Procesan los datos y encuentran grupos o clústeres naturales si existen en los datos.
 - **K-means:** agrupa objetos en k grupos basándose en sus características. Minimiza la suma de distancias entre cada objeto y el centroide de su grupo o cluster.

2.2 La función de decisión

El clasificador lineal pertenece al grupo de métodos de aprendizaje supervisado, es necesario contar con un conjunto de vectores previamente clasificados para entrenarlo. Este conjunto de vectores se llama **vectores de entrenamiento**. El clasificador lineal utiliza una función de decisión lineal $d(x)$. En el caso más simple, con solo dos clases de salida, cuando $d(x) > 0$, x pertenece una de las clases conocida, en caso contrario, a la otra.

Función de Decisión Lineal

Si las entradas de un clasificador lineal tienen la forma $x = [x_1, x_2, \dots, x_n]$, la función de decisión se

define como: $d(x) = w_1 x_1 + w_2 x_2 + \dots + w_n x_n + w_{n+1}$

O con vector de entradas aumentado, $x = [x_1, x_2, \dots, x_n, 1]$: $d(x) = w_1 x_1 + w_2 x_2 + \dots + w_n x_n + x_{n+1} w_{n+1}$

Como la misma cantidad es agregada a todos los patrones de todas las clases en juego, las propiedades geométricas entre las mismas se mantienen.

Cálculo con vectores

El vector de entradas x es un vector fila. Si representamos el conjunto de coeficientes con un vector columna $w = [w_1, w_2, \dots, w_{n+1}]^T$, entonces $d(x) = xw$. *Notar que la salida de $d(x)$ es un escalar.*

w_1
w_2
w_3
w_4

x_1^1	x_2^1	x_3^1	1	d^1
x_1^2	x_2^2	x_3^2	1	d^2
x_1^3	x_2^3	x_3^3	1	d^3
x_1^4	x_2^4	x_3^4	1	d^4
x_1^5	x_2^5	x_3^5	1	d^5

Es común calcular la f. lineal al mismo tiempo para todos los vectores de entrada. En ese caso, $d(X) = Xw$, donde X es la matriz cuyas filas son los vectores de entrada y el resultado de $d(x)$ es un vector columna.

x_1^1	x_2^1	x_3^1	1	y^1
x_1^2	x_2^2	x_3^2	1	y^2
x_1^3	x_2^3	x_3^3	1	y^3
x_1^4	x_2^4	x_3^4	1	y^4
x_1^5	x_2^5	x_3^5	1	y^5

Cálculo de los coeficientes

Para entrenar el modelo es necesario conocer la clase a la que pertenece cada uno de los vectores x . Supongamos que estas clases son c_1 y c_2 . Se desea que $d(x) > 0$ para c_1 y

$d(x) < 0$ para c_2 , entonces se crea una salida esperada y para cada vector, $y = \begin{cases} +1 & \text{si } x \in c_1 \\ -1 & \text{si } x \in c_2 \end{cases}$ donde y es un escalar, pero si se contempla la salida esperada para todos los vectores de entrada, se obtiene el vector columna y .

Al igual que en la regresión lineal, se buscan los coeficientes que minimizan el valor esperado del error. Para hallar el vector w óptimo se parte de la aproximación del error cuadrático medio:

$$S^2(w) = \frac{\sum_{i=1}^m \left(y_i - \sum_{j=1}^{n+1} w_j x_{ji} \right)^2}{m} = E(|y - Xw|^2)$$

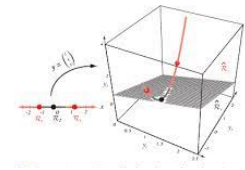
donde, m : cantidad de vectores de entrenamiento; n : cantidad de características.

Haciendo $\frac{\partial S^2(w)}{\partial w} = 0$, se llega a la siguiente expresión para el cálculo de los coeficientes (para

vectores de entrada definidos como filas): $w = (X^T X)^{-1} X^T y$ donde, w : vector de pesos; X : matriz de entradas; y : vector de salida. **w es la solución del sistema lineal** determinado por todos los patrones de ambas clases.

La desventaja más importante es que solo puede clasificar clases linealmente separables.

La ecuación $g(x) = 0$ define la superficie de decisión que separa puntos asignados a la categoría 1 de puntos asignados a la categoría 2. Cuando $g(x)$ es lineal, la superficie de decisión es un **hiperplano**. r es la distancia algebraica de el punto x al hiperplano H , (la mínima distancia de un punto fijo a cualquier punto de un plano).



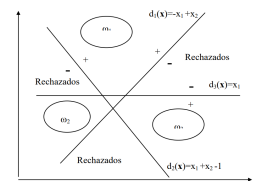
Partimos de un plano con 2 variables (x,y) . Para clasificarlos usamos una función $f(x)$ que nos da un valor para cada punto de x_1, x_2 . Estos valores se diagraman en una tercera dimensión que es la de la función de decisión.

Conforme los puntos van tomando valores positivos y negativos sobre esta dimensión, se los puede clasificar (ya que estarán por encima o debajo del hiperplano).

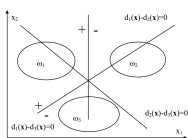
Clasificación multiclase

Uno contra todos (Piecwise)

- Cada clase tiene su propia función de decisión.
- Cada función se entrena para separar los vectores propios del resto.
- La salida del clasificador queda definida por la función de decisión de mayor valor para un vector determinado.



Por pares (Pairwise)



- Hay una función de decisión por cada par de clases $(N(N-1)/2)$.
- Cada función se entrena para separar los vectores del par correspondiente.
- La salida del clasificador queda definida por salida con frecuencia más alta entre todas las funciones decisión.

Funciones de Decisión Generalizadas

La función de decisión lineal para un vector de entradas $x=[x_1, x_2, \dots, x_n]$, definida como

$d(x) = w_1x_1 + w_2x_2 + \dots + w_nx_n + w_{n+1}$ solo puede resolver problemas de clasificación donde las **clases son linealmente separables**. Se extiende el modelo de clasificador lineal para poder resolver problemas más complejos y difíciles.

La nueva función de decisión será: $d(x) = w_1f_1(x) + w_2f_2(x) + \dots + w_mf_m(x) + w_{m+1}$

- Ya no se multiplica cada característica, sino por $f_i(x)$, que es una función que forma como parámetros el vector de características completo
- Tiene m términos, no n . No hay correspondencia con cada término como en la función lineal.
- Cada $f_m(x)$ no tiene parámetros modificables, solo se pueden modificar los coeficientes que multiplican las $f_m(x)$.

El cálculo de los coeficientes w_i . Para eso es necesario una transformación en el vector de entradas.

Se define un **nuevo vector** x^* cuyos componentes son las funciones: $x^* = [f_1(x), f_2(x), \dots, f_m(x)]$

O, si se utiliza en vector de entradas aumentado: $x^* = [f_1(x), f_2(x), \dots, f_m(x), 1]$

Una vez calculadas las funciones $f_i(x)$, el cálculo de los coeficientes es el mismo que en la función de decisión lineal. Es un espacio distinto, con nuevas características, este espacio es linealmente separable.

Clasificador polinomial

Es un caso particular de la función de decisión generalizada. El caso más simple se da para x^2 , donde la función de decisión toma la siguiente forma (para un vector de entradas de dimensión 2):

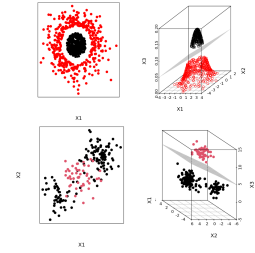
$$d(\mathbf{x}) = w_{11}x_1^2 + w_{12}x_1x_2 + w_{22}x_2^2 + w_1x_1 + w_2x_2 + w_3$$

La que podría transformarse a forma lineal como: $d(\mathbf{x}^*) = \mathbf{w}\mathbf{x}^*$ donde

$$\mathbf{x}^* = [x_1^2, x_1x_2, x_2^2, x_1, x_2, 1]$$

$$\mathbf{w} = [w_{11}, w_{12}, w_{22}, w_1, w_2, w_3]$$

- Al aumentar la cantidad de características del clasificador polinomial, se hace muy complejo y no se puede entrenar de forma directa.



Máquinas de vectores de soporte (VSM)

- También llamados clasificadores de **margen óptimo**.
- Se consideran sólo los vectores de entrenamiento próximos a la frontera entre las clases.
 - Los valores extremos en el cálculo de la $d(\mathbf{x})$ incorporan un sesgo a la función de decisión, por eso solo se utilizan los más próximos.
- Se maximiza la **mínima distancia** entre las clases y el hiperplano discriminante.

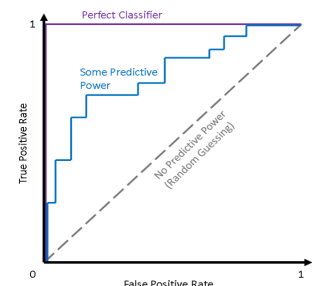
Evaluación de modelos: Matriz de confusión

Luego de resolver con un clasificador, es importante evaluarlo, qué tan bueno es. Se arma una matriz de confusión: valores predichos en columnas y reales en filas. Se le pueden calcular métricas:

		Valor real	
Valor predicción	P	Verdaderos positivos	Falsos positivos
	N	Falsos negativos	Verdaderos negativos
Total		P	N

- Factor de clasificación → $FC = \frac{a}{b}$ donde a= Suma de elementos diagonales de M (VP+VN) y b= Suma de todos los elementos de M (VP+VP+FP+FN), porcentaje de casos bien clasif.
- Sensibilidad - Razón de Verdaderos Positivos (VPR) → $VPR = \frac{VP}{P} = \frac{VP}{VP+FN}$ proporción de casos positivos que fueron correctamente identificadas
- Exactitud (Accuracy) → $ACC = \frac{VP+VN}{P+N}$ lo cerca que está el resultado de una medición del valor verdadero, cantidad de predicciones positivas que fueron correctas.
- Especificidad - Razón de Verdaderos Negativos (VNR) → $SPC = \frac{VN}{N} = \frac{VN}{FP+VN} = 1 - FPR$ casos negativos que el algoritmo ha clasificado correctamente
- Precisión (Positive predictive value) → $Precisión = \frac{VP}{VP+FP}$ dispersión del conjunto de valores obtenidos, porcentaje de casos positivos detectados.
- Razón de Falsos Positivos (FPR) → $FPR = \frac{FP}{N} = \frac{FP}{FP+VN}$

La **curva ROC** evalúa el rendimiento de los algoritmos de clasificación binaria. Con un mismo clasificador calculo las métricas de sensibilidad y FPR para **varios umbrales**, y debería utilizar el que caiga en el punto (0,1), es decir FPR=0 y sensibilidad=1.



Ayuda a encontrar un umbral de clasificación que se adapte a nuestro problema. *El área debajo de la curva determina lo bueno de un clasificador.* En la mayoría de los casos reales existe un umbral de clasificación (costo de equivocarse en la decisión).

3. REDES NEURONALES

Una **red neuronal** es un modelo simplificado que emula el modo en que el cerebro humano procesa la información: Funciona simultaneando un número elevado de unidades de procesamiento interconectadas que parecen versiones abstractas de neuronas.

3.1 Perceptron

Un perceptrón es una **neurona artificial**, y, por tanto, una unidad de red neuronal. Efectúa cálculos para detectar características o tendencias en los datos de entrada.

Se trata de un algoritmo para el aprendizaje **supervisado** de clasificadores binarios. Permite que las neuronas artificiales aprendan y traten los elementos de una serie de datos.

$$y = f \left(\sum_{i=1}^N w_i x_i - \theta \right)$$

La salida del perceptrón se define como: donde:

- x_i es el elemento i del vector de entradas $x=[x_1, x_2, \dots, x_N]$
- w_i es el elemento i del vector de pesos $w=[w_1, w_2, \dots, w_N]$
- θ es el umbral de activación
- f es la función umbral (también threshold o hard-lim) $f(z) = \begin{cases} 1 & \text{si } z \geq 0 \\ 0 & \text{si } z < 0 \end{cases}$

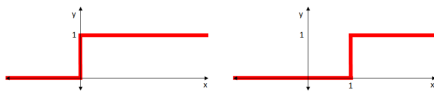
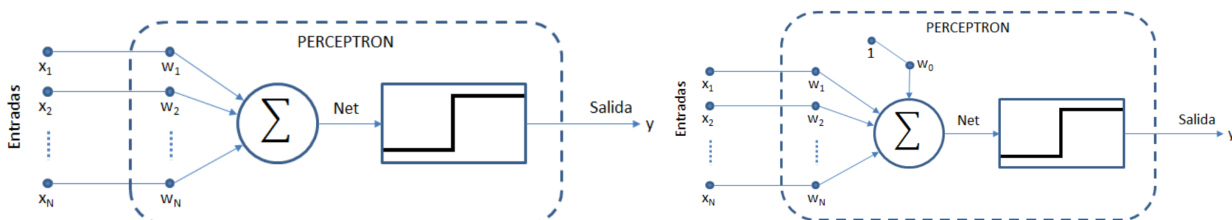


Diagrama del perceptron

En definitiva se trata de una red que al recibir el vector de entrada calcula la función de decisión lineal y le aplica una función umbral que fuerza a la red a obtener solo dos valores posibles a la salida. (Usualmente 1 y 0 o 1 y -1 según como se defina F).

La diferencia fundamental está en la forma de calcular el vector de pesos W , no se hace por cálculos matriciales sino por un **método iterativo de aproximación**.



Perceptrón con vector de entradas aumentado

$$y = f \left(\sum_{i=0}^N w_i x_i \right)$$

La salida del perceptrón se define como: donde: $w_0 = -\theta$ (Diagrama derecha)

En lugar de decir que expandimos el vector de entrada x , se adiciona a la neurona una entrada que siempre recibe el valor 1, que es lo mismo. Sin el término w_0 , siempre pasaría por el 0 en eje, ya que es la ordenada al origen. Es el componente unitario de x suele definirse en forma separada.

En realidad el perceptrón es una **función matemática**. Los datos de entrada (x) se multiplican por los coeficientes de peso (w). El resultado es un valor que puede ser positivo o negativo. La neurona se activa si el peso calculado de los datos de entrada supera un umbral determinado.

El resultado predicho se compara con el resultado conocido. En caso de diferencia, el error se retropropaga para permitir ajustar los pesos.

Entrenamiento

Para ajustar los parámetros internos (pesos) del perceptron se utiliza un método de aprendizaje supervisado, por lo tanto es necesario contar con datos en la forma:

x_0	x_1	x_2	x_3	...	x_N	y'
1	0.14	0.24	0.34	...	0.55	1
1	0.11	0.89	0.33	...	0.9	1
1	0.97	0.27	0.34	...	0.53	0
1	0.93	0.36	0.72	...	0.75	1
1	0.92	0.15	0.29	...	0.97	0
1	0.58	0.31	0.52	...	0.14	0
1	0.29	0.52	0.54	...	0.88	1
1	0.58	0.59	0.51	...	0.06	1

Los pesos se adaptan según la ley de aprendizaje: $\Delta w_i = \alpha(y' - y)x_i$
donde α es la tasa de aprendizaje e y' es la salida esperada.

Los pasos del método son:

1. Definir el valor de α
2. Inicializar los pesos con valores aleatorios
3. Mientras error > 0 para algún vector de entradas, hacer:
 - a. Ejecutar ciclo completo de entrenamiento (epoch). Mientras existan vectores de entrenamiento hacer:
 - i. Tomar un vector entrada/salida del set de datos
 - ii. Calcular la salida $y = f\left(\sum_{i=0}^N w_i x_i\right)$
 - iii. Calcular el $error = y' - y$
 - iv. Calcular $\Delta w_i = \alpha(y' - y)x_i$
 - v. Modificar los pesos haciendo $w_{i,t+1} = w_{i,t} + \Delta w_i$

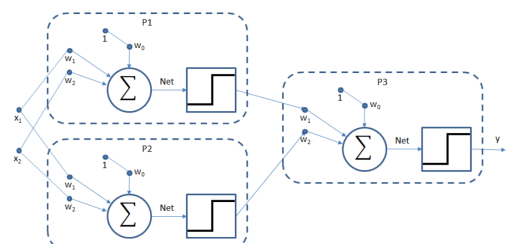
A menos que se defina una cantidad máxima de ciclos, el entrenamiento termina cuando **todos los vectores están bien clasificados**.

Desventaja del perceptron

Al igual que el clasificador lineal, el perceptron no puede clasificar correctamente si las clases no son **linealmente separables**. Ejemplo: compuerta XOR.

Este problema NO se puede solucionar, el perceptron NO puede resolver el problema, sin embargo, hay una alternativa **teórica** que involucra otro modelo y abre un camino para tratar este tipo de problemas. Conectando perceptrones entre sí, en lo que se llama perceptron multicapa, se podría resolver el problema **si supiéramos cómo adaptar los pesos**.

¿Por qué no se puede aplicar este modelo? No sabemos cómo entrenarlo... No sabemos cómo modificar los pesos de los perceptrones de la primera capa porque no sabemos cuánto contribuyó cada uno al error de la última salida final.



3.2 Adaline

Adaline es un tipo de neurona con dos diferencias importantes respecto al perceptron:

- Cambia el mecanismo de aprendizaje, se utiliza la regla delta, basada en el mínimo error cuadrático medio (Least Mean Squared o LMS error).
 - El aprendizaje busca encontrar el mejor vector de pesos posible en términos del criterio del error cuadrático medio.
- Tiene una función de activación lineal.
- Adaline está limitada a una única neurona de salida. Un vector x como su entrada y un número real y como salida

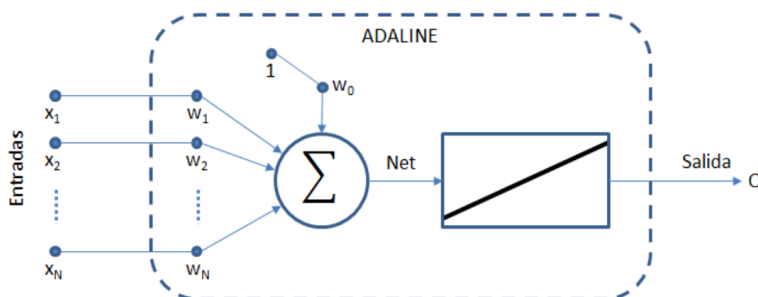
Nomenclatura:

- O : salida de la red (por output). En Adaline $O = \sum_{i=0}^N w_i x_i = Net$
- y : salida esperada
- k : identificador de un vector de entrenamiento

$$E = \frac{1}{2L} \sum_{k=1}^L (y_k - O_k)^2$$

El error cuadrático: donde L es la cantidad de vectores de entrenamiento.

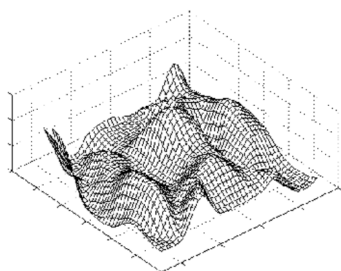
La adaptación de los pesos se hace a través de una búsqueda sobre la superficie del error. Para encontrar el mínimo de la función del error los pesos se modifican en cantidades proporcionales al gradiente decreciente de la función E .



Ley de aprendizaje de Adaline o regla delta

La ley de aprendizaje de Adaline, $\Delta w_{ik} = \alpha(y_k - O_k)x_i$ es igual a la del perceptron, pero no hay que confundirse, el comportamiento no es el mismo porque el error puede tomar cualquier valor en R

El mecanismo de aprendizaje conduce a actualizaciones de tipo continuo, siendo la actualización de los pesos proporcional al error cometido por la neurona.



El **método de entrenamiento** también es similar al del perceptron, con la diferencia de que el error nunca es cero, así que se necesita otra condición de corte. Típicamente se corta el entrenamiento cuando se llega a un umbral de error previamente definido o cuando el error se estabiliza.

Los **pesos se ajustan simultáneamente**, una tasa alta implica saltos oscilatorios, que impiden la convergencia hacia una solución; una tasa pequeña puede demorar demasiado el proceso.

3.3 Backpropagation

- Backpropagation es un método para entrenar redes neuronales multicapa.
- ¿Para qué sirve entrenar redes multicapa? Para clasificar datos que no sean linealmente separables.

La regla de entrenamiento backpropagation está basada en descenso de gradiente y regla de la cadena. Los pesos se inicializan aleatoriamente y cambian en la dirección que reduce el error

mediante la regla Delta: $\Delta w_i = -\alpha \delta x_i$ donde $\delta = -(y - O)$

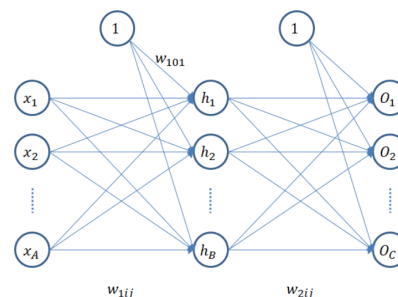
La regla delta ha sido extendida (regla delta generalizada o backpropagation) a redes con capas intermedias con **conexiones hacia adelante** (feedforward networks) y cuyas neuronas tienen **funciones de activación** continuas, no decrecientes y derivables.

Red multicapa con conexiones hacia adelante

El ejemplo más simple, con una sola capa oculta, donde i es la i -ésima entrada de la neurona j .

Notar que:

- No hay conexiones entre neuronas de la misma capa.
- No hay conexiones hacia neuronas de capas anteriores.
- Cada capa puede tener distintas cantidades de neuronas.
- La imagen corresponde a una red de tres capas (una oculta, la de salida y la de entrada).



Funciones de activación

Recordar que un Adaline, o cualquier neurona artificial antes de aplicar la función de activación, realiza una **transformación lineal**. Entonces, una red de n capas formadas por neuronas lineales se puede reemplazar por una red de una sola capa y, por lo tanto, no puede resolver problemas no linealmente separables. Por esta razón es necesario agregar no-linealidades entre las capas.

La función de activación no lineal más utilizada es la función sigmoidal: $f(x) = \frac{1}{1 + e^{-x}}$

La derivada de la función sigmoidal es: $\frac{\partial f(x)}{\partial x} = f(x)(1 - f(x))$

Entonces, por ejemplo, si las neuronas de salida de la red anterior tienen funciones de activación sigmoidal, $O_j = f(Net_j) = \frac{1}{1 + e^{-Net_j}}$ y la

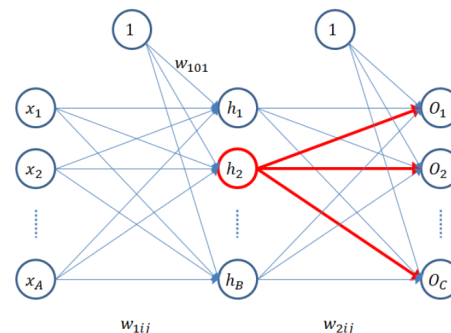
derivada de la salida con respecto a Net $\frac{\partial O_j}{\partial Net_j} = \frac{1}{1 + e^{-Net_j}} \left(1 - \frac{1}{1 + e^{-Net_j}}\right) = O_j(1 - O_j)$

Otras funciones de activación: (hay muchas más → [link](#))

Name	Plot	Equation	Derivative (with respect to x)	Range	Order of continuity	Monotonic	Monotonic derivative	Approximate identity near the origin
Identity		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$	C^∞	Yes	Yes	Yes
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$	$\{0, 1\}$	C^{-1}	Yes	No	No
Logistic (a.k.a. Sigmoid or Soft step)		$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$ [1]	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$	C^∞	Yes	No	No
Tanh		$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$	C^∞	Yes	No	Yes
Arc Tan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$	$(-\frac{\pi}{2}, \frac{\pi}{2})$	C^∞	Yes	No	Yes

El nombre del método significa **retropropagación** o **propagación hacia atrás** (del error).

- El error y la modificación de los pesos de la capa de salida se calcula de la forma conocida.
- Para las capas ocultas, el error de cada neurona se calcula como la suma ponderada (por los pesos) de los errores de la capa siguiente.



Método de entrenamiento

Cada paso (ciclo) del método tiene dos fases:

- En la primera se toma un vector de entrada y se lo propaga hacia adelante, a través de todas las capas, hasta calcular la salida.
- En la segunda fase se calcula el error de la capa de salida y se lo propaga hacia atrás para calcular el error de todas las neuronas de las capas ocultas. Una vez calculados todos los errores, se modifican los pesos.

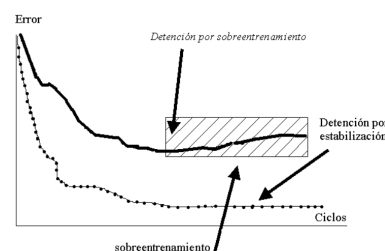
Pasos:

1. Definir el valor de α
2. Inicializar los pesos con valores aleatorios
3. Mientras error medio > error aceptado, hacer:
 - a. Ejecutar ciclo completo de entrenamiento (epoch). Mientras existan vectores de entrenamiento hacer:
 - i. Tomar un vector entrada/salida del set de datos
 - ii. Calcular la salida de cada capa hasta llegar a la capa de salida.
 - iii. Calcular el error de las neuronas de la capa de salida.
 - iv. Calcular el error de las neuronas de la última capa oculta.
 - v. Repetir el punto anterior para todas las capas ocultas (desde la salida hacia la entrada).
 - vi. Calcular los deltas de cada neurona en función de su propio error.
 - vii. Modificar los pesos.

Otras consideraciones sobre Redes Neuronales

- Dificultad para elegir los hiperparámetros
 - Arquitectura
 - Inicialización de los pesos
 - Tasa de aprendizaje: Adición de un momento
- División del juego de datos en:
 - Datos de entrenamiento
 - Datos de validación
 - Datos de test
- Sobreentrenamiento
 - Luego de un período de entrenamiento normal, el error tiende a aumentar nuevamente

$$\Delta w_{i(t+1)} = \alpha \delta x_i + \beta \Delta w_{i(t)}$$



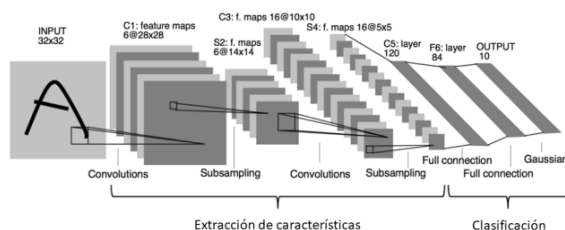
3.4 Deep Learning

El aprendizaje profundo es un conjunto de herramientas de aprendizaje automático que logra entrenar redes neuronales con varias capas de profundidad. También es:

- Un tipo muy poderoso de aprendizaje automático.
- La reencarnación moderna de las redes neuronales artificiales.
- Un conjunto de funciones matemáticas simples y entrenables.
- Es el estado del arte para la mayoría de los desafíos que enfrenta el rec. de patrones.
 - Visión artificial, Reconocimiento de audio, Procesamiento del lenguaje natural (NLP), Sistemas de diagnóstico médico, Predicción de terremotos, energía, mercado, etc.

End-to-End

- Proveer una **solución de extremo-a-extremo** es una de las grandes diferencias entre el aprendizaje profundo y el resto del aprendizaje automático. Se supone que es una ventaja.
- Los modelos de aprendizaje profundo incluyen la etapa de **extracción de características**.



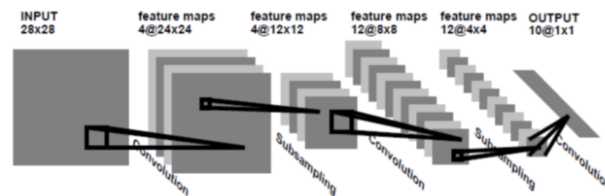
Tipos más comunes de capas en un modelo de aprendizaje profundo

<p>Densamente conectadas</p>	<p>La salida se calcula como la aplicación de la función de activación sobre la suma ponderada de las entradas.</p> <p>Las neuronas se conectan con todas las de la capa anterior.</p> <p>No hay conexiones entre neuronas de la misma capa.</p>
<p>Convolucionales</p>	<p>La salida se calcula como la aplicación de la función de activación sobre la suma ponderada de las entradas.</p> <p>Las neuronas tienen un campo perceptivo acotado.</p> <p>No hay conexiones entre neuronas de la misma capa.</p> <p>Los pesos se comparten entre neuronas de la misma capa y kernel*</p>
<p>Pooling</p>	<p>(MaxPooling, AveragePooling)</p> <p>Cada elemento calcula el máximo o la media de los valores en su campo receptivo.</p>
<p>Recurrentes</p>	<p>La salida es retroalimentada pasando por un retardo.</p> <p>Procesan secuencias de longitud indefinida.</p>
<p>Long Short Term Memory (LSTM)</p>	<p>Solucionan el problema de las dependencias Long-Term.</p> <p>Tienen la capacidad de añadir y eliminar información del estado de la célula.</p>

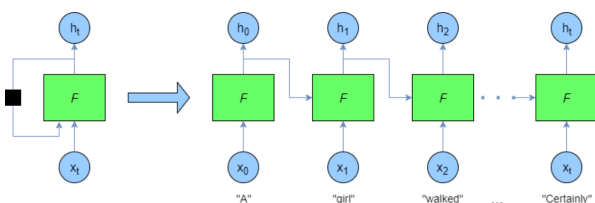
* **Kernels de convolución:** Cada capa de convolución está compuesta por varios kernels que se especializan en detectar distintos patrones

Las redes convolucionales combinan capas convolucionales y de pooling.

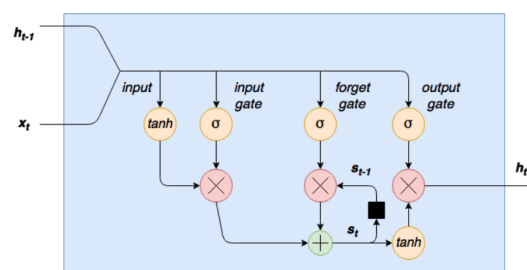
- Las capas convolucionales detectan patrones en distintos niveles.
- Las capas de pooling reducen la dimensión y hacen que la detección sea invariante a la escala y a la traslación.



Capas Recurrentes

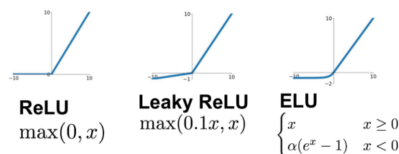


Capas LSTM

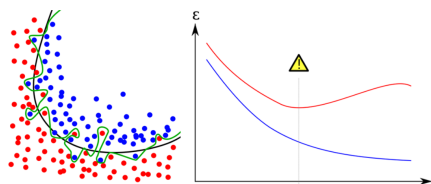


Complicaciones del Aprendizaje Profundo

- Desvanecimiento del gradiente
 - Una solución es cambiar la función de activación por una función que no sature.



- Utilizar residual networks.
- Modelos más complejos (más parámetros)
 - **Sobreentrenamiento:** Los modelos complejos son más propensos al este.



Formas de reducirlo:

- Early-stop
- Aumentar la cantidad de datos de entrenamiento
- Regularización
- Dropout previene la co-adaptación desmedida de las neuronas
- **Mayor necesidad de recursos**
 - Modelos muy complejos (millones de parámetros)
 - Por suerte son cálculos paralelizables
 - La mejor solución son las GPUs (por ahora)

Transformada de hough