5. INTRODUCCION A LAS REDES NEURONALES ARTIFICIALES

José J. Martínez P.

josejesusmp@gmail.com
Octubre 2022

5.1 Introducción

Los datos más importantes para los seres vivos son los provenientes del ambiente que filtran nuestros ojos, nuestros oídos y también de los otros sentidos. La principal fuente de supervivencia de las especies proviene de estos datos masivos que internamente analizan en su cerebro, buscando patrones para determinar si hay presas, calculando el esfuerzo para atraparlas, o si son predadores para eludirlos. De esa interacción permanente con el ambiente los animales realmente hacen Machine Learning, pues son muchos los datos que se van almacenando y otros que llegan permanentemente, a partir de los cuales logran el conocimiento, para enfrentar la incertidumbre del ambiente. En la figura 5.1, el pájaro observa la mariposa y su modelo mental reconoce el tipo de mariposa, para tomar una decisión.

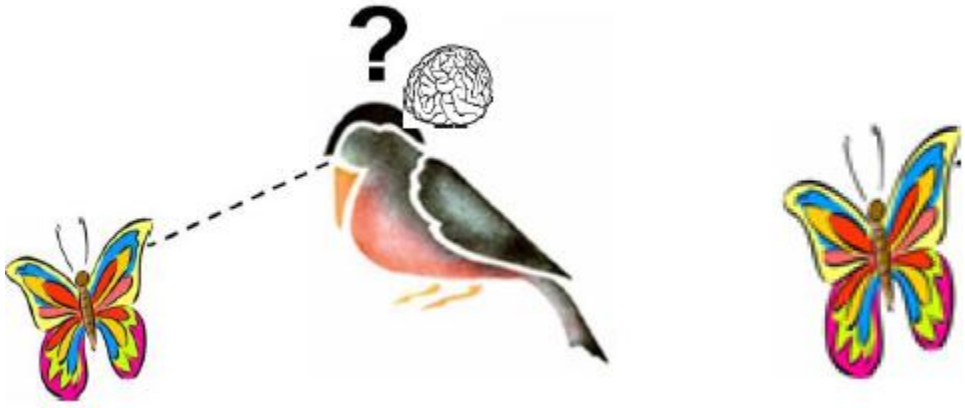


Figura 5.1. Modelo mental para predecir.

El modelo mental elimina detalles no pertinentes y enfatiza en los detalles de interés. En este caso en la mariposa, filtra de la cantidad de datos que ofrece el ambiente y con base en su visión y otros sentidos, encuentra patrones y actúa en consecuencia. Cuando el agente no detecta apropiadamente el patrón, debe realizar cambios en su estructura mental y adaptar el modelo para que le permita una mejor anticipación de las consecuencias, cuando se encuentre nuevamente con ese patrón o uno similar. Los agentes que fallan pierden recursos que generalmente favorecen a los exitosos.

Ese cambio en su modelo mental es lo que llamamos aprendizaje a partir de la experiencia. Si se examinan problemas, cuya descripción algorítmica es muy compleja, como identificar en fracciones de segundo una persona entre una multitud, sus soluciones tienen una característica común: la experiencia. Los animales superiores, como el hombre, son capaces de resolver estas situaciones acudiendo a la experiencia acumulada.

Así, de manera muy simplista, es razonable pensar que una forma de solucionar este tipo de problemas es construir sistemas que sean capaces de simular esta característica. La observación anterior fue considerada por los científicos de ciencias de la computación en lo que denomina Neurocomputación cuyo objeto de estudio son las Redes Neuronales Artificiales, RANs. Las RANs no son más que un modelo artificial y simplificado del cerebro de los animales. Una red neuronal artificial es un sistema para el procesamiento de datos, cuya unidad básica de procesamiento está inspirada en la neurona.

En el artículo "A Logical Calculus of the Ideas Inmanent in Nervious Activity" de MacCulloch y Pitts describen el primer modelo de redes neuronales, el perceptrón. Pocos años después Frank Rosemblat publicó el concepto de perceptrón. La neurona artificial o perceptrón es el bloque de construcción y unidad básica de proceso de una red neuronal artificial. Marvin Minsky, junto con Seymor Papert, escribieron el libro "Perceptrones", donde se establece cómo una sola neurona es incapaz de implementar una función como XOR, y llegaron a que las grandes redes también tienen limitaciones similares. Este concepto limitó el desarrollo de las redes neuronales artificiales por aproximadamente diez años. Luego se retomó el tema y tuvo un primer florecimiento a finales de los 80s y comienzos de los 90s. A finales de los 90s disminuyo nuevamente el interés. Actualmente hay un interés renovado y se considera como una técnica del estado del arte de ML, aprendizaje de máquina en muchas aplicaciones.

5.2 Modelos de Neuronas

5.2.1 Neurona Natural

La neurona natural, de manera muy simplificada para nuestro interés, consta de un núcleo, un cuerpo denominado soma, al cual están conectadas unas fibras nerviosas llamadas dendritas, fibras terminales de entrada, y un axón, que es una fibra delgada y larga que se divide en su extremo en una serie de fibras terminales de salida.

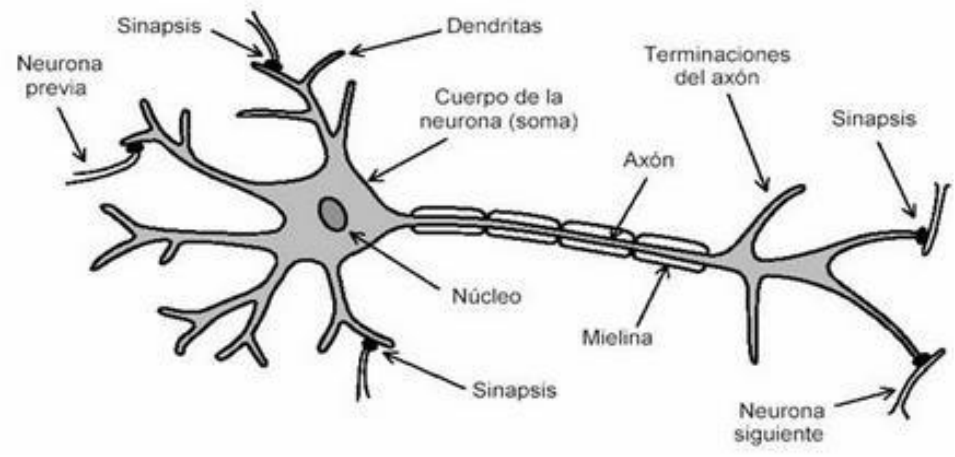


Figura 5.2. Esquema de una neurona natural.

La neurona recibe señales de entrada, provenientes de otras células nerviosas, por las dendritas de la neurona. Las neuronas reciben impulsos de los axones de otras neuronas a través de una conexión llamada sinapsis. La sinapsis (del griego “sin”, que significa “con” y “haptenia”, que significa “con firmeza”) es una conexión más fuerte o más débil dependiendo de la importancia de la señal que llega de la otra neurona. Las señales a través de las cuales se permite la captación y el proceso de control y coordinación sobre los órganos son de tipo eléctrico. El axón es el encargado de conducir el impulso nervioso desde el soma de una neurona hacia otras células nerviosas.

Las neuronas transmiten su señal eléctrica a consecuencia de un cambio transitorio en la permeabilidad de la membrana plasmática. Cuando el potencial de una membrana, la suma de todas las señales de entrada, se despolariza más allá de un umbral (de 65mV a 55mV), la célula dispara un potencial de acción que se envía a través del axón, como señal de salida de la neurona. El ser humano tiene en promedio 10^{12} neuronas y en promedio cada neurona tiene 1.000 conexiones sinápticas.

5.2.2 NEURONA ARTIFICIAL

O perceptrón, es un modelo matemático, planteado en términos vectoriales que recibe una serie de señales (valores) $\boldsymbol{X}$ de entrada, que representa el conjunto de dendritas, ponderadas por un peso $\boldsymbol{W}$, que representa el conjunto de sinapsis, y produce una señal de salida $\boldsymbol{Z}$, que representa el potencial de salida del axón. La variable **θ** representa el umbral o el mínimo potencial a partir del cual se dispara el axón. Figura 5.3.

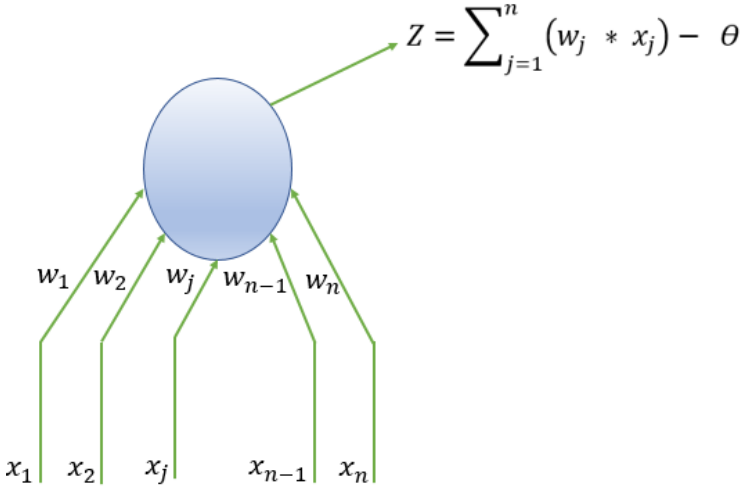


Figura 5.3 Neurona Artificial, modelo abstracto de una neurona natural

Habíamos hablado de que a través del axón se propaga el disparo de la neurona, después de que se supera el umbral. Se define la función la función de activación $g(Z)$, en este caso, como la función paso unitario, que es una combinación lineal de los valores de entrada **X** y los pesos **W**. Donde **Z** si es mayor que un umbral determinado, toma el valor 1, sino toma el valor 0. En el perceptrón no hay una función de activación, que es la que produce la complejidad en la red neuronal.

$$g(Z) = \begin{cases} 1 \text{ si } Z \geq \theta \\ 0 \text{ si } Z < \theta \end{cases}$$

Donde

$Z = -\theta + (w_1\, x_1\, , w_2\, x_2\, , \ldots , w_j\, x_j\, , \ldots , w_n\, x_n\,)\; = \sum_{j=1}^{n} w_j x_j - \theta$, para propósitos de notación, hacemos $\theta = -b$. Esta notación es muy importante ya que nos permite trabajar todo el desarrollo matemático en forma vectorial. Entonces tenemos que:

$$z = \sum_{j=1}^{n} w_j x_j + b$$

$$z = b * 1 + (w_1 x_1 + w_2 x_2 + \cdots + w_n x_n)\,,$$

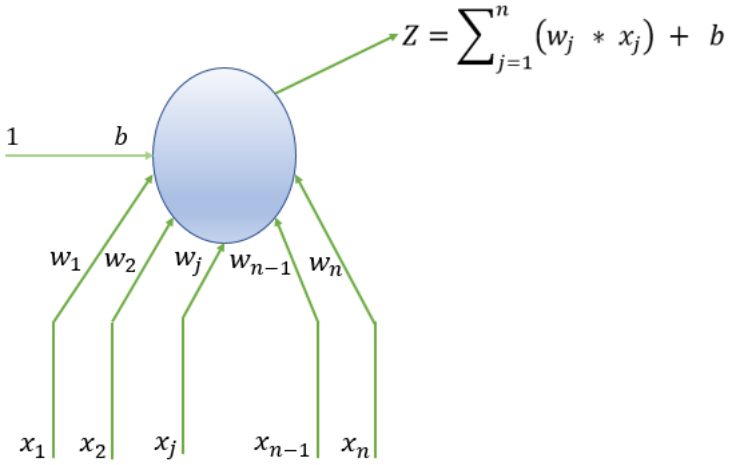


Figura 5.4 Neurona Artificial, modelo abstracto de una neurona natural, con sesgo (bias) b.

En términos de aprendizaje de máquina, vamos a denominar el vector X, como el vector de rasgos o características (features) y el vector W como el vector de pesos.

$$W = \begin{bmatrix} b \\ w_1 \\ \vdots \\ w_n \end{bmatrix} \quad X = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$$

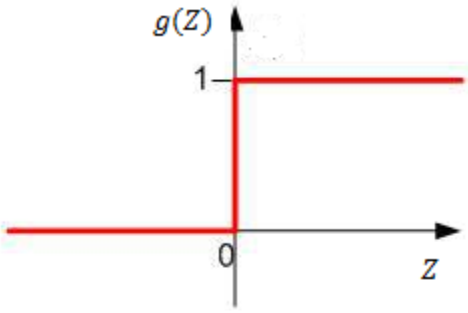


Figura 5.5 Función paso unitario

Sin embargo, esta función tiene varios defectos, en primer lugar, en la naturaleza nunca se presentan cambios en tiempo cero y además es una función que no es derivable, algo bien importante para el algoritmo de aprendizaje. De manera que se utilizan varias funciones de activación que son derivables, la más común es la función de activación sigmoidal, figura 5.6.

$$\varsigma(z) \equiv \frac{1}{1+e^{-z}}$$

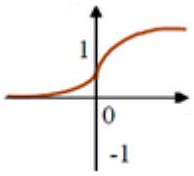


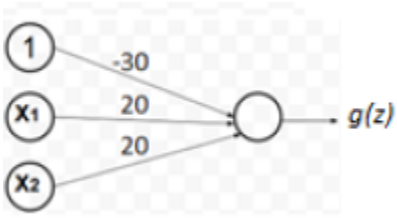
Figura 5.6 Función sigmoide

La función de activación sigmoidal tiene la ventaja adicional que: $\sigma' = \sigma(1 - \sigma)$. Es importante resaltar que la introducción de la función de activación le da la complejidad de la red neuronal, sin la función de activación, las redes neuronales son simplemente combinaciones lineales.

5.3 Ejemplos de clasificación.

Veamos la compuerta AND, figura 5.7.

Con los pesos 30, 20 y 20, se calcula la salida $g(Z)$,



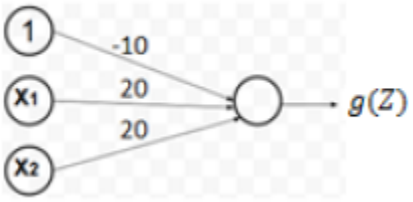
X1	X2	Z	$g(Z)$
0	0	-30	~0
0	1	-10	~0
1	0	-10	~0
1	1	10	~1

W
-30
20
20

Figura 5.7 Red para clasificar una compuerta AND, topología, tabla y vector de pesos.

Ahora la compuerta OR, figura 5.8.

Ahora con los pesos -10, 20 y 20, se calcula $g(Z)$,



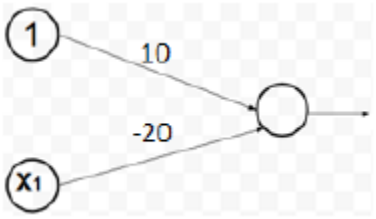
X1	X2	Z	$g(Z)$
0	0	-10	~0
0	1	10	~1
1	0	10	~1
1	1	30	~1

W
-10
20
20

Figura 5.8 Red para clasificar una compuerta OR, topología, tabla y vector de pesos.

Como se puede observar ambos perceptrones tienen el mismo número de entradas, solo cambia el peso en la primera entrada, aunque el primero clasifica una compuerta AND y el segundo una compuerta OR. En otras palabras, el solo cambio de peso en una entrada hace que cambie completamente la función de la red. Veamos ahora la compuerta NOT, figura 5.9.

$Z = (10 - 20x_1)$ y luego se calcula $g(Z)$



X1	Z	g(Z)
0	10	~ 1
1	-10	~ 0

W
10
-20

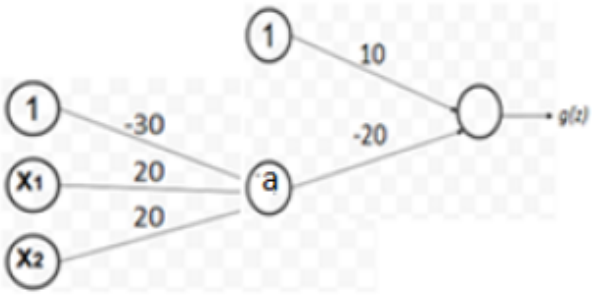
Figura 5.9 Red para clasificar una compuerta NOT, topología, tabla y vector de pesos

Aquí tenemos una clasificación binaria, $Y = [1, 0]$

Ahora, con base en las conformaciones anteriores vamos a armar la clasificación de una compuerta NAND.

$Z^1 = (-30 + 20x_1 + 20x_2)$, luego se calcula $A^1 = g(Z^1)$; luego $Z^2 = (10a_1 - 20a_2)$ y $A^2 = g(Z^2)$;

X Z^1 A^1 Z^2 A^2



X1	X2	Z1	a	Z2	g(Z)
0	0	-30	~ 0	10	~ 1
0	1	-10	~ 0	10	~ 1
1	0	-10	~ 0	10	~ 1
1	1	10	~ 1	-10	~ 0

W1	W2
-30	10
20	-20
20	

Figura 5.10 Red para clasificar una compuerta NAND, topología, tabla y matriz de pesos

Ahora tenemos dos capas. En la primera capa obtenemos un valor que hemos llamado Z^1 que se calcula con el vector de pesos W^1 y luego aplicando la función sigmoide obtenemos como A^1 resultado intermedio. En la capa 2, obtenemos un valor que hemos llamado Z^2 que se calcula con el vector de pesos W^2 y luego aplicando la función sigmoide obtenemos el resultado $g(Z^2)$, o salida de la red, que corresponde a la compuerta NAND. Como podemos observar ahora tenemos una matriz de pesos W.

Podemos hacer lo mismo para obtener la compuerta NOR, uniendo las conformaciones OR y NOT como se hizo en el caso anterior. Pero también podemos unir las redes de clasificación de las compuertas AND y NOR, que nos da la siguiente topología, figura 5.11

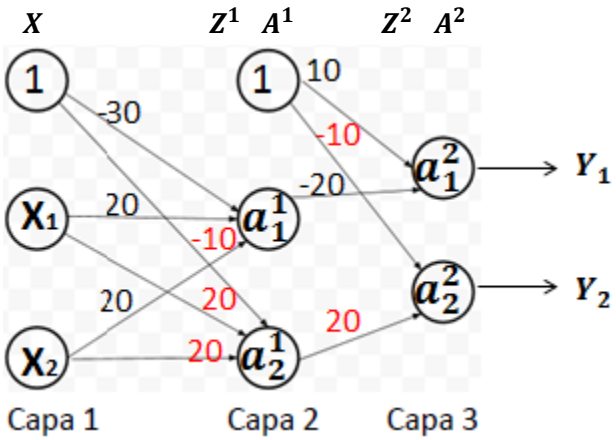


Figura 5.11. Red para clasificar las compuertas AND y OR.

$z_1^1 = (-30 + 20x_1 + 20x_2)$ y $z_2^1 = (-10 + 20x_1 + 20x_2)$; calculamos $a_1^1 = g(z_1^1)$ y $a_2^1 = g(z_2^1)$, que son la salida de la capa 1. Para la capa 2, calculamos $z_1^2 = (10 - 20a_1^1)$ y $z_2^2 = (10 + 20a_2^1)$ y finalmente obtenemos $y_1 = a_1^2 = g(z_1^2)$ y $y_2 = a_2^2 = g(z_2^2)$

Los superíndices nos indican la capa, así, el vector $Z^1 = [z_1^1\ z_2^1]$, el vector $A^1 = [a_1^1\ a_2^1]$, que son las salidas de la capa 1. La matriz de pesos de la capa 1 es $W^1 = [w_1^1\ w_2^1]$ y $W^2 = [w_1^2\ w_2^2]$:

$W^1 =$

W_1^1	W_2^1
-30	-10
20	20
20	20

$W^2 =$

W_1^2	W_2^2
10	10
-20	-20

En este caso tenemos una multi-clasificación, es decir $g(Z^2)$ es un vector. En términos de clasificación. Si k es el número de clases, $k = 2$, en este caso tenemos dos clases NAND y NOR.

$Y =$

Y1	Y2
1	0
0	1

Si por ejemplo estamos clasificando números arábigos, tenemos $k = 10$.

Clasificación

Se puede definir de manera general la clasificación como el agrupamiento de cosas que comparten ciertas características y cualidades. Por ejemplo, clasificar formas geométricas, números escritos a mano, o simplemente huevos. En IA, la clasificación es el problema de identificar a que conjunto de categorías pertenece una nueva observación, con base en un proceso computacional en el que se utiliza un modelo algorítmico y en el que se ha tenido un conjunto de datos de entrenamiento que contiene observaciones cuya membresía de categorías se conoce.

5.4 Aprendizaje Supervisado

Uno de los procesos más importantes en RNA es el proceso de aprendizaje. El aprendizaje puede hacerse de manera supervisada o no supervisada. En el aprendizaje o entrenamiento supervisado, se proveen tanto las entradas como las salidas, o rótulos, correspondientes a esas entradas.

Por ejemplo, un conjunto de datos es Iris, muy utilizado en ML, para reconocimiento de patrones. Se compone de tres clases de especies Iris, con 50 instancias cada una. Tiene cinco atributos: longitud del sépalo, ancho del sépalo, longitud del pétalo, ancho del pétalo, con todas las dimensiones en centímetros y el nombre, Iris Setosa, Iris Versicolor e Iris Virginica. Con base en los primeros cuatro atributos, se le enseña a reconocer el quinto atributo, este es el que se conoce como el rótulo.

La red entonces procesa las entradas y compara sus salidas resultantes contra las salidas deseadas o rótulos. En la mayoría de los casos no corresponden y se debe calcular su diferencia o error. Entonces ahora se busca ajustar los pesos que controlan la red para disminuir el error, hasta que de acuerdo con un determinado criterio se considera que el error es aceptable.

Generalización del aprendizaje

Se busca que con la red neuronal se pueda, por ejemplo, predecir apropiadamente el costo de cualquier inmueble, es decir la red debe generalizar el concepto para cualquier tipo de inmueble. Se pueden presentar dos problemas, que el modelo sea demasiado general o que sea demasiado especifico. Esto se debe tener en cuenta al entrenar la red.

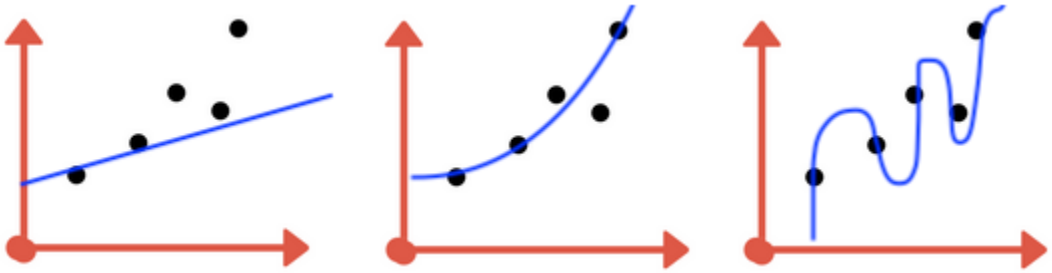


Figura 5.12. En la figura de la izquierda se tiene sub-entrenamiento, también conocido como underfitting; en el centro, el entrenamiento adecuado, y en la figura de la derecha se tiene sobre entrenamiento o overfitting.

Para solucionar este problema, el dataset se divide en tres conjuntos, el conjunto de entrenamiento, generalmente del 80% del dataset; el conjunto de prueba, con el que se busca obtener una estimación del comportamiento de la red en el mundo real; y el conjunto de validación, que nos permite una evaluación de la generalización del modelo.

Propagación

Las redes que hemos visto hasta ahora se conocen como feedforward, cuya característica es que el proceso va desde la capa de entrada hasta capa de salida. No hay retroalimentación hacia atrás. Es la red más simple y utilizada. Entonces, vamos a generalizar una red feed-forward, con su nomenclatura, con base en la figura 5.13.

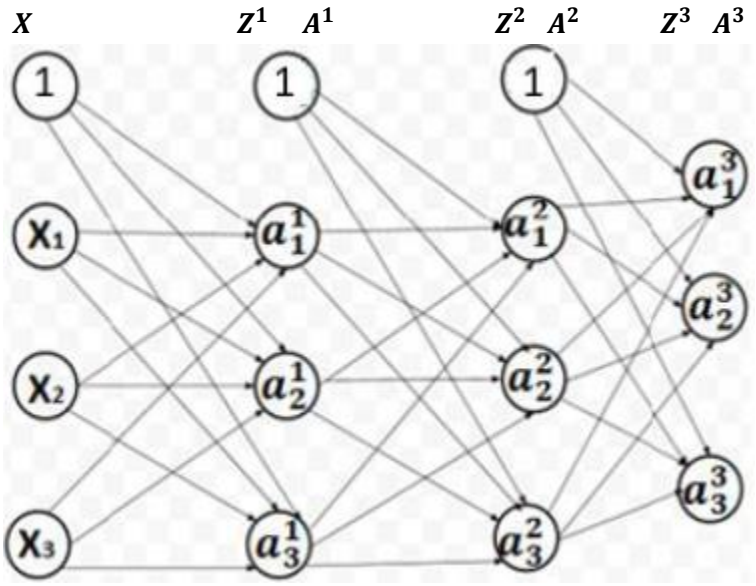


Figura 5.13. Red de tres capas para mostrar la propagación, desde el vector de entrada.

Vamos a describir cómo se propagan los datos en la red desde la entrada.

Capa 1:

$$z_1^1 = b_1^1 + (w_{11}^1 x_1 + w_{21}^1 x_2 + w_{31}^1 x_3), \quad a_1^1 = g\,(z_1^1)$$

$$z_2^1 = b_2^1 + (w_{12}^1 x_1 + w_{22}^1 x_2 + w_{32}^1 x_3), \quad a_2^1 = g\,(z_2^1)$$

$$z_3^1 = b_3^1 + (w_{13}^1 x_1 + w_{23}^1 x_2 + w_{33}^1 x_3), \quad a_3^1 = g\,(z_3^1)$$

con

$$W_T^1 = B^1 + W^1 = \begin{matrix} b_1^1 \\ b_2^1 \\ b_3^1 \end{matrix} + \begin{matrix} w_{11}^1 & w_{21}^1 & w_{31}^1 \\ w_{12}^1 & w_{22}^1 & w_{32}^1 \\ w_{13}^1 & w_{23}^1 & w_{33}^1 \end{matrix}$$

Matriz de pesos de la capa 1

Matricialmente, $Z^{(1)} = B^1 + \left(W^{(1)T}\right)X$ y $A^{(1)} = g(Z^{(1)})$; para generalizar podemos decir que $A^{(0)} = X$, y así la ecuación matricial para la capa 1 sería $Z^{(1)} = B^1 + (W^{(1)T})A^{(0)}$.

Capa 2:

$$z_1^2 = b_1^2 + (w_{11}^2 a_1^1 + w_{21}^2 a_2^1 + w_{31}^2 a_3^1), \quad a_1^2 = g\,(z_1^2)$$

$$z_2^2 = b_2^2 + (w_{12}^2 a_1^1 + w_{22}^2 a_2^1 + w_{32}^2 a_3^1), \quad a_2^2 = g\,(z_2^2)$$

$$z_3^2 = b_3^2 + (w_{13}^2 a_1^1 + w_{23}^2 a_2^1 + w_{33}^2 a_3^1), \quad a_3^2 = g\,(z_3^2)$$

con

$$W_T^2 = B^2 + W^2 = \begin{matrix} b_1^2 \\ b_2^2 \\ b_3^2 \end{matrix} + \begin{matrix} w_{11}^2 & w_{21}^2 & w_{31}^2 \\ w_{12}^2 & w_{22}^2 & w_{32}^2 \\ w_{13}^2 & w_{23}^2 & w_{33}^2 \end{matrix}$$

Matriz de pesos de la capa 2.

Matricialmente $Z^{(2)} = B^2 + (W^{(2)T})A^{(1)}$

Capa 3:

$$z_1^3 = b_1^3 + (w_{11}^3 a_1^2 + w_{21}^3 a_2^2 + w_{31}^3 a_3^2), \quad a_1^3 = g\,(z_1^3)$$

$$z_2^3 = b_2^3 + (w_{12}^3 a_1^2 + w_{22}^3 a_2^2 + w_{32}^3 a_3^2), \quad a_2^3 = g\,(z_2^3)$$

$$z_3^3 = b_3^3 + (w_{13}^3 a_1^2 + w_{23}^3 a_2^2 + w_{33}^3 a_3^2), \quad a_3^3 = g\,(z_3^3)$$

$$W_T^3 = B^3 + W^3 = \begin{matrix} b_1^3 \\ b_2^3 \\ b_3^3 \end{matrix} + \begin{matrix} w_{11}^3 & w_{21}^3 & w_{31}^3 \\ w_{12}^3 & w_{22}^3 & w_{32}^3 \\ w_{13}^3 & w_{23}^3 & w_{33}^3 \end{matrix}$$

Matriz de pesos de la capa 3

Matricialmente
$Z^{(3)} = B^3 + (W^{(3)T})A^{(2)}$ y $A^{(3)} = g(Z^{(3)})$. Generalizando, para cualquier capa oculta i, se tiene que:

$$Z^{(i)} = B^{(i)} + (W^{(i)T})A^{(i-1)} \text{ y } A^{(i)} = g(A^{(i)} = g(Z^{(i)}).)$$

Al analizar esta ecuación, encontramos que se pasa de una capa a la otra con la ecuación de la izquierda y el vector de salida $A^{(i)}$ se encuentra aplicando la función de activación a $Z^{(i)}$.

Volviendo a la figura 5.11, como la capa 3 es la última capa de la red, los $A^{(3)}$ son la respuesta de la red que notamos como $H_w(X)$. En otras palabras, el resultado de la propagación del vector X a través de la red, es el vector $H_w(X)$.

Aprendizaje

Con el fin de encontrar los pesos apropiados para que la red responda apropiadamente a los patrones que queremos, debemos trabajar en el ajuste de los pesos. Para entrenar la red en esos patrones hacemos ***aprendizaje supervisado***, proveyendo, para este proceso un conjunto de entrenamiento: $(X_{(1)}, Y_{(1)})$, $(X_{(2)}, Y_{(2)})$, … , $(X_{(m)}, Y_{(m)})$. Donde los vectores X son las características (features) o patrones de entrada y los vectores Y son los patrones que debe reconocer o las clasificaciones que debe producir la red.

Analicemos un poco el conjunto de entrenamiento. Tomemos por el ejemplo el conjunto de entrenamiento de MNIST que consiste en dígitos escritos a mano de 10.000 ejemplos. Los dígitos son imágenes de un tamaño normalizado de 28 X 28 pixeles, 784 pixeles, en escala de grises, con un rótulo que dice que número es, de manera que los valores de los rótulos son de 0 a 9. De manera que en el par $(X_{(m)}, Y_{(m)})$, el $X_{(m)}$ corresponde a un vector de 784 valores y $Y_{(m)}$corresponde a un rótulo entre 0 y 9. Se trata entonces de darle como entrada a una red los 784 pixeles y un rótulo, a la red, pero 10.000 veces, para que se encuentren los pesos que clasifiquen cualquier número escrito a mano que se normalice a 28 X 28 pixeles, ver figura 5.14.

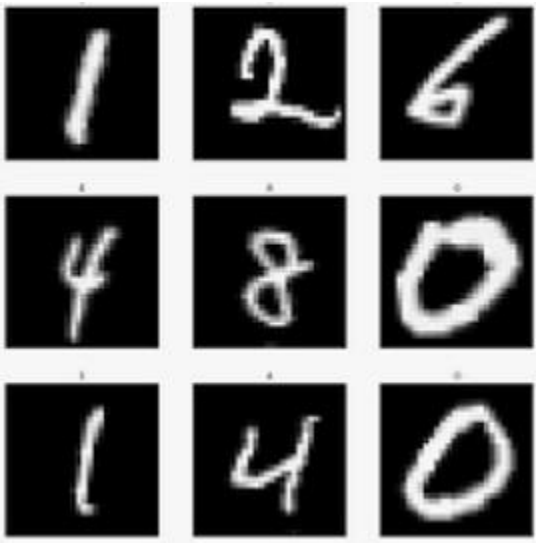


Figura 5.14. En la figura aparecen los números escritos a mano como imágenes de 28 X 28, 784 pixeles, valores de X, aparte vienen los valores 1, 2, 6, 4, 8, 0, 1, 4, 0; que conforman los valores de Y, o rótulos.

5.5 Algoritmo de aprendizaje

La topología de la red depende del número de características (features) del vector de entrada, del número de salidas en la capa de salida, que corresponde a las clasificaciones que hace la red o del resultado que se espera y del número de capas ocultas, que generalmente es de dos o tres capas. Además, todas las neuronas de la capa i se conectan con todas las neuronas de la capa $i + 1$.

Inicialmente se asignan valores aleatorios entre [0, 1] a los pesos de la red, aunque algunos autores recomiendan valores menores de 0.5. Una vez se hace la propagación a través de la red de los vectores X, con esos valores de pesos, se obtiene el vector de salida Y_c. La diferencia con el rotulo de entrenamiento Y, es $e = |Y - Y_c|$, es el error de clasificación de la red $e = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}$

El proceso de entrenamiento o aprendizaje consiste en hacer que ese error sea menor que un Ɛ definido. También se define el error promedio E como:

$$E = \frac{1}{2}(e)^2 = \frac{1}{2}(e)^T e$$

Donde $e = |Y_r - Y|$, el valor del rótulo menos el valor calculado por la propagación.

Minimización del error o entrenamiento

La idea es minimizar el error haciendo que se afecten todos los pesos en cada una de las capas de la red. Como solo se conoce el error en la última capa, se comienza minimizando el error en esa capa. Y luego se sigue hacia las capas internas, hasta llegar a la capa de entrada. Esto es lo que se conoce como algoritmo de retropropagación o Backpropagation.

Backpropagation.

Tercera capa. Dado que tenemos 3 capas, se minimiza el error con respeto a los pesos en la capa 3, en este caso la capa de salida, ver figura 5.14. Siguiendo la regla de la cadena se tiene:

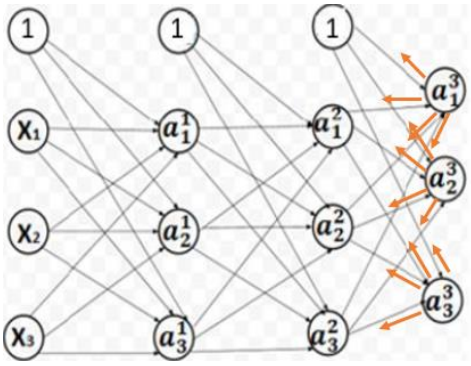


Figura 5.15, Con base en el error promedio E, se ajustan lo valores de la matriz de pesos de la última capa.

De manera que se busca encontrar el cambio en el error promedio E con base en el cambio en la matriz de pesos W_T^3 : $\frac{\partial E}{\partial W_T^3}$. Pero lo que conocemos es Z^3, de manera que hacemos:

$$\frac{\partial E}{\partial W_T^3} = \frac{\partial E}{\partial Z^3}\frac{\partial Z^3}{\partial W_T^3}$$

O También:

$\frac{\partial E}{\partial W_T^3} = \frac{\partial Y}{\partial Z^3}\frac{\partial e}{\partial Y}\frac{\partial E}{\partial e}\frac{\partial Z^3}{\partial W_T^3}$, recordando que $A^3 = Y = g(Z^3)$. $Y = \sigma(Z^3)$

Es importante recordar que $\sigma' = \sigma(1 - \sigma)$

Así, $\frac{\partial Y_i^3}{\partial Z_j^3} = \begin{cases} 1, si\ i = j \\ 0, si\ i \neq j \end{cases}$ o $\frac{\partial Y^3}{\partial Z^3} = \begin{bmatrix} \sigma'(Z_1^3) & 0 & 0 \\ 0 & \sigma'(Z_2^3) & 0 \\ 0 & 0 & \sigma'(Z_3^3) \end{bmatrix}$ y notando

Nos queda que: $\frac{\partial Y}{\partial Z^3} = diag\left(\sigma'(Z^3)\right)$. Recordando $e = |Y_r - Y|$

Se tiene, $\frac{\partial e}{\partial Y} = \frac{\partial e_i}{\partial Y_j} = \begin{cases} -1, si\ i = j \\ 0, si\ i \neq j \end{cases}$ o $\frac{\partial e_i}{\partial Y_j} = -\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1) \end{bmatrix} = -1$

1

Ahora, $\frac{\partial E}{\partial e} = \frac{\partial}{\partial e}\left(\frac{1}{2}(e)^T e\right) = e$. Recordando $Z^{(3)} = B^{(3)} + (W^{(3)T})A^{(2)}$

Ahora, $\frac{\partial Z^3}{\partial W_T^3} = \frac{\partial}{\partial W_T^3}(B^{(3)} + (W^{(3)T})A^{(2)})$ 0, $\frac{\partial Z^3}{\partial W_T^3} = 1 + A^{(2)T}$

Reuniendo todos los términos:

$$\frac{\partial E}{\partial W_T^3} = \frac{\partial Y^3}{\partial Z^3}\frac{\partial e}{\partial Y^3}\frac{\partial E}{\partial e}\frac{\partial Z^3}{\partial W_T^3}$$

$$\frac{\partial E}{\partial W_T^3} = diag\left(\sigma'(Z^3)\right) - 1\ e\ (1 + A^{(2)T})$$

$$\frac{\partial E}{\partial W_T^3} = -diag\left(\sigma'(Z^3)\right)\ e\ (1 + A^{(2)T})$$

Haciendo $\delta^3 = diag\left(\sigma'(Z^3)\right) e$, entonces

$$\frac{\partial E}{\partial W_T^3} = -\delta^3(1 + A^{(2)T}) \text{ o } \frac{\partial E}{\partial W_T^3} = -\delta^3 - \delta^3 A^{(2)T}$$

donde $\frac{\partial E}{\partial W_T^3} = \frac{\partial E}{\partial B^3} + \frac{\partial E}{\partial W^3}$

con $\frac{\partial E}{\partial B^3} = -\delta^3$ y $\frac{\partial E}{\partial W^3} = -\delta^3 A^{(2)T}$

De manera que con estos valores ajustamos lo pesos de la última capa.

Segunda capa. Ahora se minimiza el error promedio E con respeto a los pesos de la capa 2, ver figura 5.16. Siguiendo la regla de la cadena se tiene:

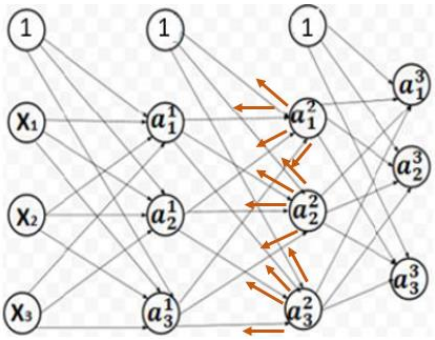


Figura 5.16. Con base en el error promedio E, se ajustan lo valores de la matriz de pesos de la segunda capa.

$$\frac{\partial E}{\partial W_T^2} = \frac{\partial E}{\partial Z^2}\frac{\partial Z^2}{\partial W_T^2}$$

$$\frac{\partial E}{\partial W_T^2} = \frac{\partial A^2}{\partial Z^2}\frac{\partial Z^3}{\partial A^2}\frac{\partial E}{\partial Z^3}\frac{\partial Z^2}{\partial W_T^2}$$

Ahora, $\frac{\partial A_i^2}{\partial Z_j^2} = \begin{cases} 1, si\ i = j \\ 0, si\ i \neq j \end{cases}$ o $\frac{\partial A^2}{\partial Z^2} = \begin{bmatrix} \sigma'(Z_1^2) & 0 & 0 \\ 0 & \sigma'(Z_2^2) & 0 \\ 0 & 0 & \sigma'(Z_3^2) \end{bmatrix}$

$$\frac{\partial A^2}{\partial Z^2} = diag(\sigma'(Z^2))$$

Ahora, $\frac{\partial Z^3}{\partial A^2} = \frac{\partial}{\partial W_T^3}(B^{(3)} + (W^{(3)T})A^{(2)})$ o, $\frac{\partial Z^3}{\partial A^2} = W^{(3)T}$

Ahora, para calcular $\frac{\partial E}{\partial Z^3}$ Teníamos que: $E = \frac{1}{2}(e)^2$ o $E = \frac{1}{2}(Y_r - Y)^2$ o $E = \frac{1}{2}(Y_r - g(Z^{(3)}))^2$ ahora, derivando con respecto a $Z^{(3)}$, se tiene:

$\frac{\partial E}{\partial Z^3} = \left(Y_r - g\left(Z^{(3)}\right)\right) - diag\left(\sigma'(Z^3)\right)$, pero $e = |Y_r - g\left(Z^{(3)}\right)|$, entonces reacomodando

$\frac{\partial E}{\partial Z^3} = -diag\left(\sigma'(Z^3)\right)e$, y se había definido $\delta^3 = diag\left(\sigma'(Z^3)\right)e$

$$\frac{\partial E}{\partial Z^3} = -\delta^3$$

Ahora, $\frac{\partial Z^2}{\partial W_T^2} = \frac{\partial}{\partial W_T^2}\left(B^{(2)} + W^{(2)T}A^1\right)$ o, $\frac{\partial Z^2}{\partial W_T^2} = 1 + A^{(1)T}$

Reuniendo los términos

$$\frac{\partial E}{\partial W_T^2} = -diag\left(\sigma'(Z^2)\right) W^{(3)T}\ \delta^3\ (1 + A^{(1)T})$$

Haciendo $\delta^2 = diag\left(\sigma'(Z^2)\right) W^{(3)T}\ \delta^3$, entonces

$$\frac{\partial E}{\partial W_T^2} = -\delta^2\left(1 + A^{(1)T}\right) \text{ o } \frac{\partial E}{\partial W_T^2} = -\delta^2 - \delta^2 A^{(1)T}$$

Donde $\frac{\partial E}{\partial B^2} = -\delta^2$ y $\frac{\partial E}{\partial W^2} = -\delta^2 A^{(2)T}$

$$\frac{\partial E}{\partial W_T^3} = \frac{\partial E}{\partial B^3} + \frac{\partial E}{\partial W^3}$$

Primera capa. Ahora se minimiza el error promedio E con respeto a los pesos de la capa 1, ver figura 5.17. Siguiendo la regla de la cadena se tiene:

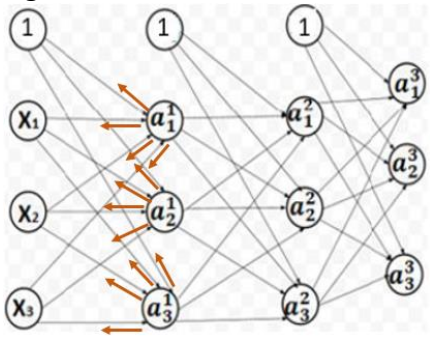


Figura 5.17. Con base en el error promedio E, se ajustan lo valores de la matriz de pesos de la primera capa.

$$\frac{\partial E}{\partial W_T^1} = \frac{\partial E}{\partial Z^1}\frac{\partial Z^1}{\partial W_T^1}$$

$$\frac{\partial E}{\partial W_T^1} = \frac{\partial A^1}{\partial Z^1}\frac{\partial Z^2}{\partial A^1}\frac{\partial E}{\partial Z^2}\frac{\partial Z^1}{\partial W_T^1}$$

Ahora, $\frac{\partial A_i^1}{\partial Z_j^1} = \begin{cases} 1, si\ i = j \\ 0, si\ i \neq j \end{cases}$ o $\frac{\partial A_i^1}{\partial Z_j^1} = \begin{bmatrix} \sigma'(Z_1^1) & 0 & 0 \\ 0 & \sigma'(Z_1^1) & 0 \\ 0 & 0 & \sigma'(Z_1^1) \end{bmatrix}$

$$\frac{\partial A^1}{\partial Z^1} = diag(\sigma'(Z^1))$$

Ahora, $\frac{\partial Z^2}{\partial A^1} = \frac{\partial}{\partial A^1}(B^{(2)} + W^{(2)T}A^1)$ o $\frac{\partial Z^2}{\partial A^1} = W^{(2)T}$

Ahora, para calcular $\frac{\partial E}{\partial Z^2}$ Teníamos que: $E = \frac{1}{2}(e)^2$ o $E = \frac{1}{2}(Y - Y_c)^2$ o $E = \frac{1}{2}\left(Y - g\left(Z^{(3)}\right)\right)^2$, reemplazando $Z^{(3)}$ por $B^3 + (W^{(3)T})A^{(2)}$ se tiene: $E = \frac{1}{2}\left(Y - g\left(B^3 + (W^{(3)T})A^{(2)}\right)\right)^2$, pero $A^{(2)} = g(Z^{(2)})$; reemplazando: $E = \frac{1}{2}\left(Y - g\left(B^3 + (W^{(3)T})g(Z^{(2)})\right)\right)^2$ y derivando

$$\frac{\partial E}{\partial Z^2} = \left(Y - g\left(Z^{(3)}\right)\right)\left(-diag\left(\sigma'\left(Z^3\right)\right)\right) W^{(3)T} diag\left(\sigma'\left(Z^2\right)\right)$$

$\frac{\partial E}{\partial Z^2} = -diag\left(\sigma'\left(Z^3\right)\right) e\, W^{(3)T}\, diag\left(\sigma'\left(Z^2\right)\right)$ pero $\delta^3 = diag(\sigma'(Z^3))\, e$

$\frac{\partial E}{\partial Z^2} = \delta^3\, W^{(3)T}\, diag\left(\sigma'\left(Z^2\right)\right)$ y $\delta^2 = diag(\sigma'(Z^2))\, W^{(3)T}\, \delta^3$

$$\frac{\partial E}{\partial Z^2} = \delta^2$$

Ahora, $\frac{\partial Z^2}{\partial A^1} = \frac{\partial}{\partial A^1}\left(B^{(2)} + W^{(2)T}A^1\right)$,

$\frac{\partial Z^1}{\partial W_T^1} = 1 + A^{(0)T}$

$$\frac{\partial E}{\partial W_T^1} = -diag(\sigma'(Z^1))\ W^{(2)T}\ \delta^2\ (1 + A^{(0)T})$$

Haciendo $\delta^1 = diag(\sigma'(Z^1))\ W^{(2)T}\delta^2$, entonces

$$\frac{\partial E}{\partial W_T^1} = -\delta^1\,(1 + A^{(0)T})$$

Donde $\frac{\partial E}{\partial B^1} = -\delta^1$ y $\frac{\partial E}{\partial W^1} = -\delta^1 A^{(1)T}$

En este caso la operación + nos indica se debe considerar una operación con dos componentes, uno que es la actualización de los pesos que representan el umbral de manera separada y el otro que corresponde a la actualización de la matriz de pesos.

Actualización de los pesos

Los pesos se actualizan gradualmente usando el descenso del gradiente. Con η como la tasa de aprendizaje.

Generalizando, para cualquier capa interna:

$$\delta^k = diag(\sigma'(Z^k))\ W^{(k+1)T}\delta^{k+1}$$

$$\Delta W^{(K+1)} = -\eta\, \delta^k - \eta\, \delta^k\, A^{(k-1)T}$$
$$W^{(K)} = W^{(K)} + \Delta W^{(K+1)}$$

En las fórmulas anteriores, la variable η se conoce como tasa de aprendizaje. Es un valor que se toma entre 0 y 1. Si por ejemplo se toma $\eta = 0.2$, quiere decir que solamente se tiene en cuenta el 20% del error para la actualización de los pesos, se considera como un hiperparámetro de configuración.

Resumiendo, en el aprendizaje por Backpropagation se calcula el error en la salida. Luego se pasa el error hacia atrás y se pondera a lo largo de cada capa, permitiendo calcular los cambios que se deben realizar en la matriz W de pesos y el vector B de sesgos, en cada capa. Cuando se llega a una unidad, se propaga, se multiplica el error propagado hacia atrás por la derivada de la unidad. Es un algoritmo recursivo.

5.6 Ejemplo numérico

Se presenta un ejemplo numérico en un Notebook de Jupyter, que se puede ejecutar directamente en Python.

5.7 Aprendizaje profundo (Deep Learning)

Como se puede apreciar en la figura 5.19, en Deep Learning se tiene una RNA, como las que se han estudiado, simplemente que el adjetivo “Deep” hace referencia a que el número de capas ocultas es relativamente alto, hablamos del orden de 7, 8 o 9 capas. Claramente funciona el mismo algoritmo de retropropagación.

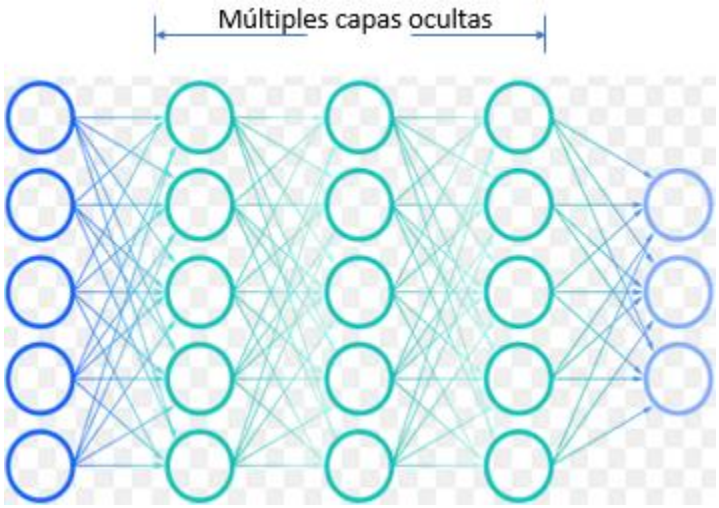


Figura 5.19. Red neuronal profunda. Sin varias capas ocultas.

5.8 Redes Convolucionales

Si se toma el caso del reconocimiento de imágenes con redes neuronales se puede tener mayor claridad sobre su utilización. Veamos por ejemplo el caso en que se quiere hacer el reconocimiento

de una cara. Una cara tiene una serie de características interesantes: los ojos no quedan muy distantes entre sí, bajo los ojos está centrada la nariz y bajo la nariz esta la boca. Estos elementos que conforman la cara se consideran detectores locales, figura 5.20. Los detectores locales buscan en la imagen puntos localmente interesantes.

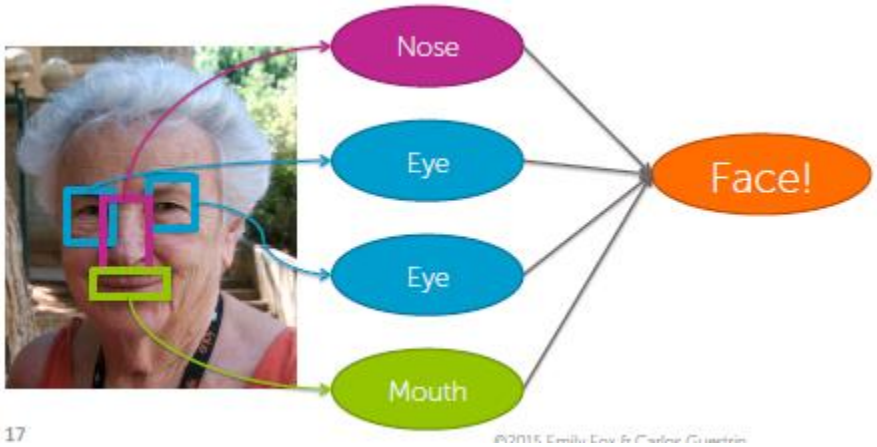


Figura 5.20. imagen con detectores locales-

Las colecciones de puntos localmente interesantes se combinan para construir clasificadores. Con Dee Learning se aprenden implícitamente estos rasgos (features).

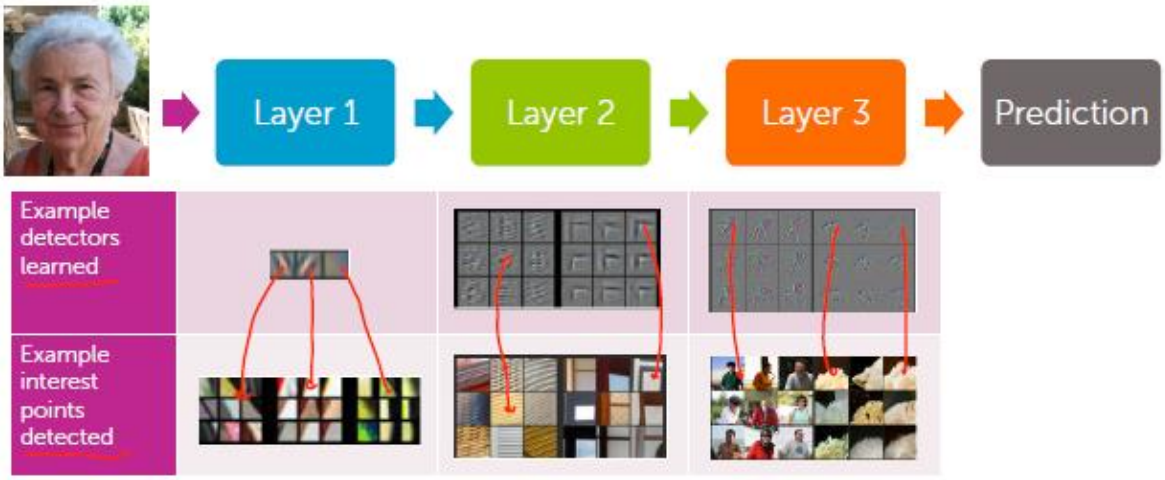


Figura 5.21. En cada capa se va aprendiendo una serie de rasgos, hasta que en la última capa se hace la predicción o reconocimiento.

Pero además se puede pensar que estos rasgos son comunes a una gran cantidad de imágenes. Luego las primeras capas lo que hacen es encontrar rasgos que son comunes a muchas imágenes. Esta es la idea de la trasferencia de aprendizaje en redes con muchas capas.

En el caso de la figura 5.22, la red neuronal se conforma de 8 capas para realizar la clasificación entre perros y gatos. Una vez que la red a aprendido esta clasificación, es decir, se tiene los pesos calculados en cada capa para realizar esta clasificación, se puede tomar la misma red neuronal dejando estos pesos ya calculados hasta la capa 7, incluir otra capa para que prenda a reconocer otra clase de imágenes, como sillas, elefantes, carros y cámaras de fotografía. Realmente es muy provechoso aprovechar el conocimiento de las capas previas.

Figura 5.21.

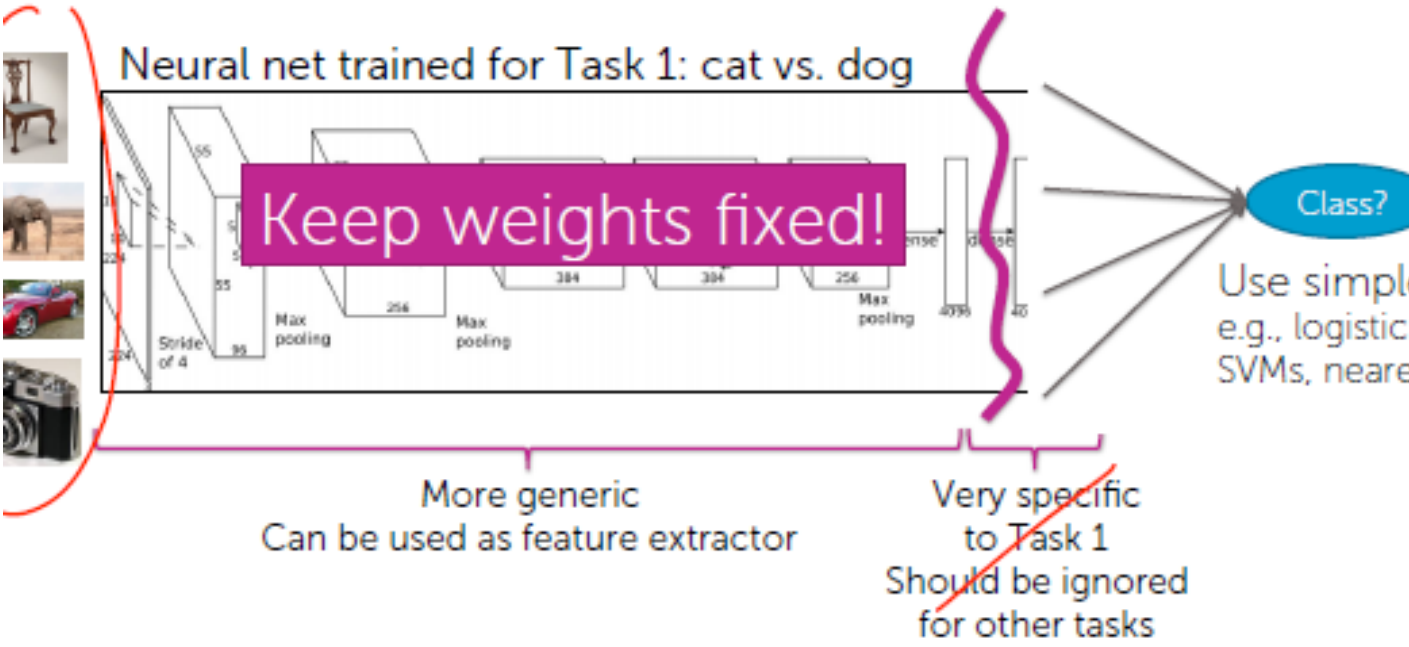


Figura 5.22. Red neuronal de ocho capas. El aprendizaje para clasificar perros y gatos se puede utilizar en otro tipo de reconocimiento o clasificación.

Veamos un ejemplo, para detectar bordes. La idea es que a la matriz que representa una imagen se le aplican ciertos filtros que permiten detectar los bordes. En este caso tenemos la matriz:

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

Veamos ahora en que consiste la operación convolución. Tomamos una matriz 3 X 3 y la usamos como un filtro

$$\begin{pmatrix} 3 & 0 & 1 & 2 & 7 & 4 \\ 1 & 5 & 8 & 9 & 3 & 1 \\ 2 & 7 & 2 & 5 & 1 & 3 \\ 0 & 1 & 3 & 1 & 7 & 8 \\ 4 & 2 & 1 & 6 & 2 & 8 \\ 2 & 4 & 5 & 2 & 3 & 9 \end{pmatrix} * \begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix}$$

Imagen 6 X 6 Filtro 3 X 3

Para aplicar el filtro, primero la matriz de la imagen 4 X 4, se divide en 16 matrices, 3 X 3

3	0	1
1	5	8
2	7	2

0	1	2
5	8	9
7	2	5

1	2	7
8	9	3
2	5	1

2	7	4
9	3	1
5	1	3

1	5	8
2	7	2
0	1	3

5	8	9
7	2	5
1	3	1

8	9	3
2	5	1
3	1	7

9	3	1
5	1	3
1	7	8

2	7	2
0	1	3
4	2	1

7	2	5
1	3	1
2	1	6

2	5	1
3	1	7
1	6	2

5	1	3
1	7	8
6	2	8

0	1	3
4	2	1
2	4	5

1	3	1
2	1	6
4	5	2

3	1	7
1	6	2
5	2	3

1	7	8
6	2	8
2	3	9

Ahora, cada elemento de la nueva matriz se calcula tomando cada una de las submatrices y multiplicándola, de una manera especial, por el filtro.

$$\begin{bmatrix} 3 & 0 & 1 \\ 1 & 5 & 8 \\ 2 & 7 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

Así se calcula el primer elemento de la nueva matriz:

$$3 \text{ X } 1 + 1 \text{ X } 1 + 2 \text{ X } 1 + 0 \text{ X } 0 + 5 \text{ X } 0 + 7 \text{ X } 0 + 1 \text{ X } -1 + 8 \text{ X } -1 + 2 \text{ X } -1 = -5$$

Se hace el mismo calculo para todas las otras submatrices, obteniéndose la siguiente matriz:

-5	-4	0	8
-10	-2	2	3
0	-2	-4	-7
-3	-2	-3	-16

De esta manera al hacer la convolución de una matriz 6 X 6 con un filtro 3 X 3, obtenemos una matriz 4 X 4. De la misma manera se calculan los siguientes elementos de la matriz 4 X 4:

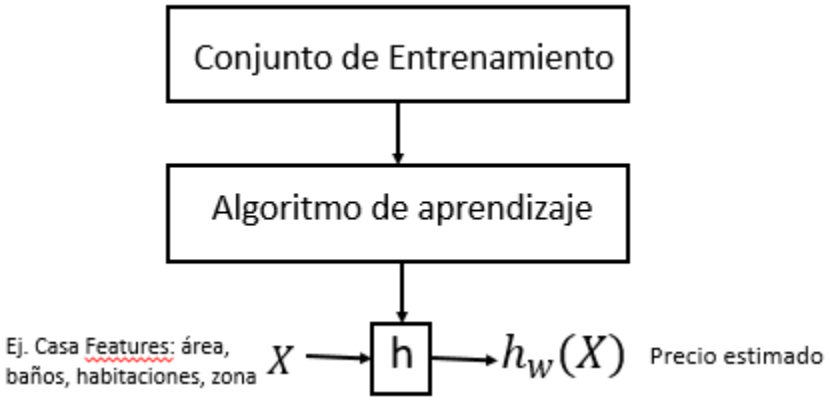
-5	-4	0	8
-10	-2	2	3
0	-2	-4	-7
-3	-2	-3	-16

Algunos de los problemas de visión computacional que se resuelven con CNN son: clasificación de imágenes, detección de objetos y transferencia de estilo neuronal. Un problema importante con los problemas de visión computacional es que los datos de entrada son realmente cuantiosos.

5.9 Hacia Machine Learning

Los datos en el aprendizaje supervisado, incluye la respuesta correcta Y_r o valores rotulados, para cada ejemplo X, el par (X, Y_r).

Notación
m = número de ejemplos de entrenamiento.
X = variable de entrada, es un vector con todas las características (features) del ejemplo.
n = número de características (features) de la variable de entrada.
Y = variable de salida, valores objetivo (target), o valores rotulados.



En las diferentes heurísticas de ML, se define una función de costo J que es la diferencia entre la hipótesis que produce un modelo $h_w(X)$ y el valor real Y en el conjunto de aprendizaje:

$$J(W) = \frac{1}{2m}(\sum_{i=1}^{m} (h_w(X) - Y)^2$$

Donde $\frac{1}{2}(h_w(X) - Y)^2$, es el error cuadrático que se ha trabajado, para el conjunto de aprendizaje. Cuando se tiene en el conjunto de aprendizaje m ejemplos, lo que nos dice la ecuación es que se toma el promedio de todos los errores al entrenar la red con todos los m ejemplos del conjunto de entrenamiento.

5.10. Ejercicios.

1. Se busca entrenar una red neuronal para predecir el valor del dólar de mañana. Busque valores del dólar y de otras dos variables que considere influyen en el valor del dólar.
2. Tome un conjunto de manzanas, un conjunto de peras, un conjunto de bananos y un conjunto de fresas. Desarrolle un sistema clasificador de estas cuatro frutas, de manera que después de entrenar la red se le den imágenes de frutas de internet y el sistema las clasifique. Obténgalos de: https://www.kaggle.com/moltean/fruits

5.9 Bilbiografía

1. K. He, X. Zhang, S. Ren y J. Sun, Deep Residual Learning for Image Recognition, Microsoft Reasearch.

3. A. Ng, Machine Learning, Coursera

2. https://www.analyticsvidhya.com/blog/2018/12/guide-convolutional-neural-network-cnn/