1. RED NEURONAL ADALINE (ADAptative LINear Element)

1.1 Arquitectura

La red ADALINE es similar al Perceptron pero en vez de tener una función de activación binaria o bipolar posee una función de activación lineal. Esto se puede observar en la figura 1.

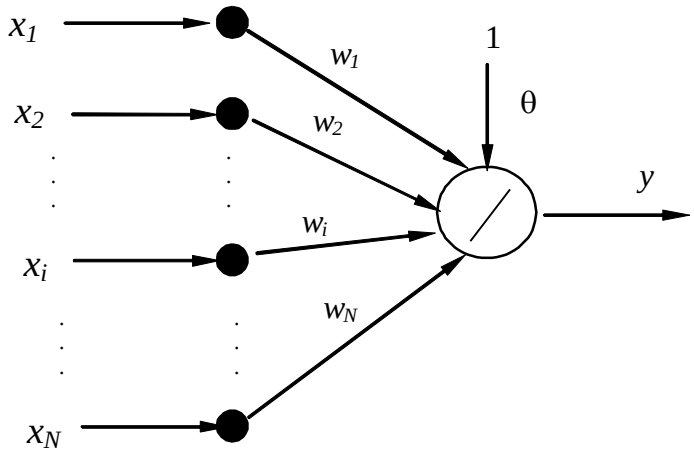

Fig. 1 Estructura de una red tipo ADALINE.

El modelo del ADALINE fue desarrollado por Bernard Widrow y su estudiante Marcian Hoff a comienzos de la década de los 60. Como la neurona de esta red tiene una función de activación lineal, el valor de salida será igual al valor de la entrada neta que la neurona esté recibiendo, que lo podemos expresar con la ecuación 1.

$$y = \sum_{i=1}^{N} w_i x_i + \theta \qquad (1)$$

Es importante recalcar que la red tipo ADALINE puede tener más de una neurona en su capa de procesamiento, dando origen a un sistema con N entradas y M salidas, como el de la figura 2.

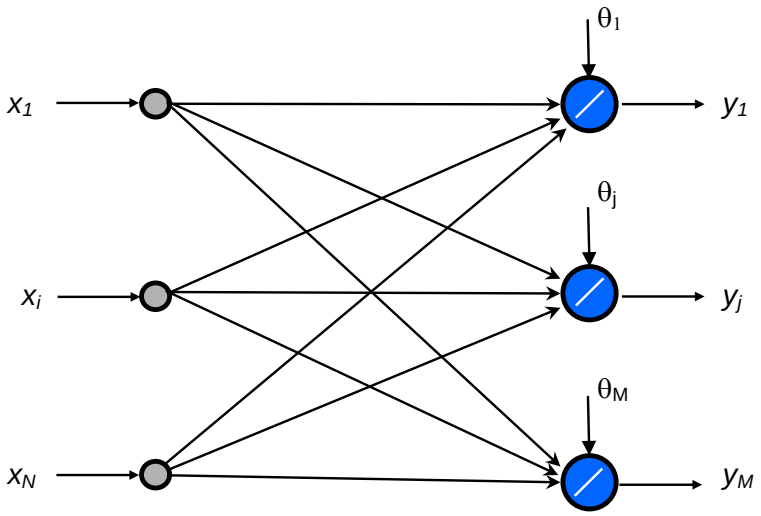

Fig. 2 Estructura de una red tipo ADALINE.

1.2 Algoritmo de Aprendizaje

Supongamos que nos encontramos en un sistema montañoso muy complejo en un lugar a gran altura y totalmente cubierto por neblina. Estamos perdidos, pero sabemos que en el valle hay una población donde podremos obtener ayuda, ¿qué estrategia utilizamos para llegar a dicha población? Muy seguramente estamos de acuerdo en que lo mejor es empezar a bajar y siempre mantener esta tendencia, por ejemplo, podríamos seguir el cause de un río, pues esta corriente de agua nos garantiza que siempre caminaremos paulatinamente hacia un lugar más bajo.

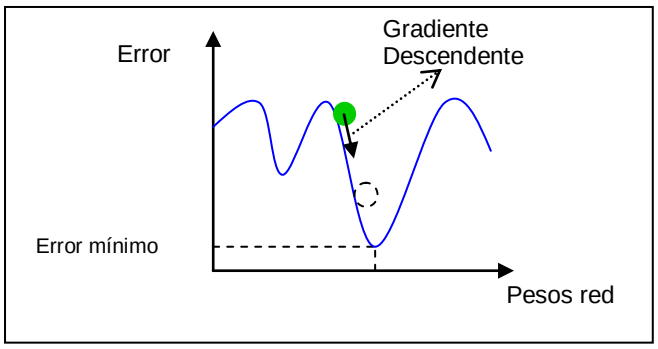

Fig. 3 Gradiente Descendente.

Este es el principio del algoritmo utilizado en el proceso de aprendizaje de la red ADALINE, en donde, partiendo desde un punto aleatorio en la superficie de error, buscaremos un punto donde el error sea mínimo, siguiendo una trayectoria descendente, como se ilustra en la figura 3. A esta técnica de búsqueda la denominaremos Algoritmo de Gradiente Descendente.

La ecuación de actualización de pesos para una red neuronal la podemos definir en términos del peso en el instante *t* y de la variación del peso sináptico, como se muestra en la ecuación 2.

$$w(t+1) = w(t) + \Delta w(t) \qquad (2)$$

donde,

w(t+1)	:	Valor actualizado del peso sináptico
w(t)	:	Valor actual del peso sináptico
□*w(t)*	:	Variación del peso sináptico

Si variamos los pesos de la red en un factor proporcionar al gradiente negativo del error de la red respecto a los pesos, logramos que los pesos así generados disminuyan el índice de desempeño (*J*) que para este caso lo definimos en términos del error cuadrático promedio en la ecuación 3 y para una red ADALINE de una neurona de salida, con P patrones de entrenamiento.

$$J = \frac{1}{2P}\sum_{p=1}^{P}(d_p - y_p)^2 \qquad (3)$$

p	:	*p-ésimo* patrón del conjunto de entrenamiento
d_p	:	Valor deseado de salida para el *p-ésimo* patrón
y_p	:	Valor de salida de la red ADALINE para el *p-ésimo* patrón

El objetivo de algoritmo es encontrar el conjunto de pesos que hagan que *J* sea mínimo; si hacemos la variación de los pesos proporcional al gradiente negativo, el cambio en el peso de la *i-ésima* conexión la representamos por la ecuación 4, donde α es el factor de aprendizaje de la red.

$$\Delta w_i = -\alpha \frac{\partial J}{\partial w_i} \qquad (4)$$

Si en la ecuación anterior aplicamos la regla de la cadena para obtener la derivada, queda en términos de la derivada del índice de desempeño respecto del error, la derivada del error respecto de la salida de la red ADALINE y la derivada de la salida de la red respecto del peso de la *i-ésima* conexión.

$$\Delta w_i = -\alpha \frac{\partial J}{\partial e} \cdot \frac{\partial e}{\partial y} \cdot \frac{\partial y}{\partial w_i} \qquad (5)$$

Tras evaluar las derivadas parciales de la ecuación 5, obtenemos los siguientes valores,

$$\frac{\partial J}{\partial e} = e = (d_p - y_p)$$
$$\frac{\partial e}{\partial y} = -1$$
$$\frac{\partial y}{\partial w_i} = x_i$$

Estos valores de las derivadas parciales los reemplazamos en la ecuación 5 para obtener el valor final de la variación del el peso de la *i-ésima* conexión.

$$\Delta w_i = -\alpha e_p(-1)x_i$$
$$\Delta w_i = \alpha e_p x_i \qquad (6)$$

Retomando la ecuación 2 y 6, la expresión para la actualización del peso de la *i-ésima* conexión está dada por la ecuación 7.

$$w_i(t+1) = w_i(t) + \Delta w_i$$
$$w_i(t+1) = w_i(t) + \alpha e_p x_i \qquad (7)$$

Con base en las ecuaciones obtenidas, propongamos un algoritmo de aprendizaje para la red ADALINE, cuyos pasos los describimos a continuación.

Paso 1

Asignamos los valores iniciales de forma aleatoria a los pesos w_{ji} y el umbral θ_j
Asignamos un valor al parámetro de aprendizaje α
Definimos un valor para el error mínimo aceptado en el entrenamiento de la red

Paso 2

Mientras la condición de parada sea falsa, ejecutamos los pasos 3 al 7.

Paso 3.

Para cada patrón de entrenamiento $\{\boldsymbol{x_p}, \boldsymbol{y_p}\}$ ejecutamos los pasos 4 al 5

Paso 4.

Calculamos la salida de la red ADALINE

$$y_{pj} = Neta_j = \sum_{i=1}^{N} w_i x_{pi} + \theta_j \qquad (8)$$

Paso 5.

Calculamos el error e_p en la salida de la red ADALINE

$$e_p = \frac{1}{2}\sum_{j=1}^{M}(d_{pj} - y_{pj})^2 \qquad (9)$$

Paso 6.

Actualizamos los pesos

$$w_i(t+1) = w_i(t) + \alpha e_p x_i \qquad (10)$$

$$\theta_j(t+1) = \theta_j(t) + \alpha e_p \qquad (11)$$

Paso 7.

Calculamos el error global de la red y si es menor que un valor mínimo detenemos el algoritmo. En caso contrario, retornamos al paso 2.