Redes Adaline y Madaline

ABRAHAM SÁNCHEZ LÓPEZ
GRUPO MOVIS
FCC-BUAP

Introducción

- Widrow y Hoff (1960) desarrollaron una regla de aprendizaje que está muy estrechamente relacionada con la regla de aprendizaje del perceptrón.
- Esta regla, llamada regla delta, ajusta los pesos para reducir la diferencia entre la entrada neta y la unidad de salida, y la salida deseada, lo que resulta en un error cuadrático medio (error LMS).
- Las redes Adaline (Adaptive Linear Neuron) y Madaline (Adaline de varias capas) utilizan esta regla de aprendizaje LMS.
- Estas redes se utilizan en diversas aplicaciones de redes neuronales.
- Los pesos de las interconexiones entre las redes Adaline y Madaline son ajustables.
- Las redes Adaline y Madaline se discuten en detalle en esta parte del curso.

Adaline, I

- En la Adaline que fue desarrollada por Widrow y Hoff, se utilizan activaciones bipolares para sus señales de entrada y salida de destino.
- Los pesos y el sesgo de la Adaline son ajustables.
- La regla de aprendizaje utilizada puede ser llamada también regla delta, regla Least Mean Square o regla Widrow-Hoff.
- La derivación de esta regla con una única unidad de salida, varias unidades de salida y su extensión ya se han discutido en temas anteriores.
- Dado que la función de activación es una función identidad, la activación de la unidad es su entrada neta.
- Cuando la red Adaline se utilizará para la clasificación de patrones, entonces, después del entrenamiento, se aplica una función de umbral a la entrada neta para obtener la activación.
- La activación es

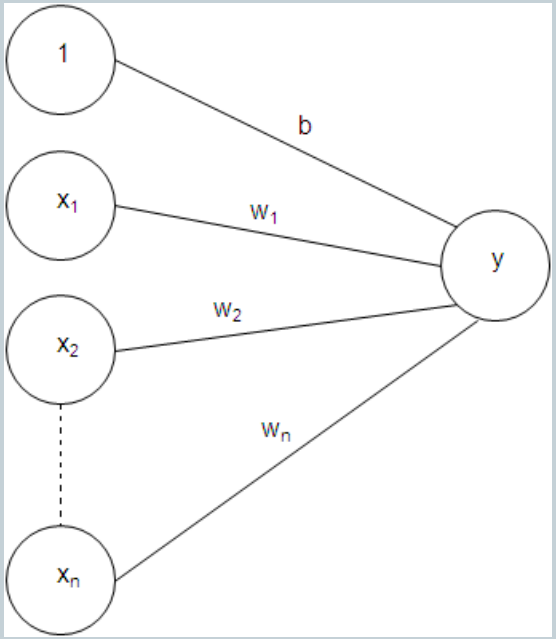
$$y = \bullet \bullet \bullet$$

Adaline, II

- La unidad Adaline puede resolver el problema de la separabilidad lineal si este se produce.

Arquitectura

- La arquitectura de una red Adaline se muestra a continuación.



Adaline, III

- La red Adaline tiene sólo una unidad de salida.
- Esta unidad de salida recibe información de varias unidades y también del sesgo; cuya activación es siempre +1.
- La red Adaline también se asemeja a una red de una sola capa. Esta recibe información de varias neuronas.
- Cabe señalar que también recibe la entrada desde la unidad, la cual es siempre '+1', llamado el sesgo.
- Los pesos del sesgo también se entrenan en la misma forma que los otros pesos.
- En la figura del acetato 4 se presenta una red con una capa de entrada $x_1,\ldots, x_i,\ldots, x_n$ y sesgo, una capa de salida con una sola neurona.
- El vínculo entre las neuronas de entrada y de salida poseen interconexiones ponderadas.
- Estos pesos se cambian a medida que avanza en el entrenamiento.

Algoritmo, I

- Básicamente, los pesos iniciales de la red Adaline tienen que ser inicializados con valores aleatorios pequeños y no a cero como se discutió en las redes de Hebb o en los perceptrones, porque esto puede influir en el factor de error a considerar.
- Después de establecer los pesos iniciales, se establecen las activaciones para la unidad de entrada.
- La entrada neta se calcula sobre la base de los patrones de entrenamiento de entrada y los pesos.
- Mediante la aplicación de la regla de aprendizaje delta discutida en diapositivas anteriores, se llevará a cabo la actualización de pesos.
- El proceso de entrenamiento se continúa hasta que el error, que es la diferencia entre el objetivo y la entrada neta es un mínimo.
- Los pasos para el algoritmo de entrenamiento de una red Adaline son como sigue:
  - **Paso 1**: Inicializar los pesos (no a cero, sino que se utilizan pequeños valores aleatorios). Establecer el coeficiente de aprendizaje α.

Algoritmo, II

- **Paso 2**: Mientras que la condición de paro es falsa, hacer los pasos 3-7.
- **Paso 3**: Para cada par de entrenamiento bipolar *e*:*o*, llevar a cabo los pasos 4-6.
- **Paso 4**: Calcular la activación de las unidades de entrada $x_i = e_i$, para $i = 1$ hasta n.
- **Paso 5**: Calcular la entrada neta de la unidad de salida $y_{-in} = b + \Sigma x_i w_i$
- **Paso 6**: Actualización del sesgo y los pesos, $i = 1$ a n.

  w_i (nuevo) = w_i (viejo) + $\alpha(o - y_{-in})\, x_i$

  b (nuevo) = b (viejo) + $\alpha(o - y_{-in})$
- **Paso 7**: Prueba de la condición de paro.

- La condición de paro puede ser cuando el cambio de peso alcanza pequeños niveles o el número de iteraciones, etc.

Aplicación del algoritmo

- El procedimiento de aplicación, que se utiliza para probar la red entrenada es como sigue.
- Se basa principalmente en la activación bipolar.

❖ **Paso 1**: Inicializar los pesos obtenidos a partir del algoritmo de entrenamiento.

❖ **Paso 2**: Para cada vector bipolar de entrada *x*, realizar los pasos 3-5.

❖ **Paso 3**: Establecer las activaciones de la unidad de entrada.

❖ **Paso 4**: Calcular la entrada neta para la unidad de salida.

$$y_{\text{-in}} = b + \Sigma x_i w_i$$

❖ **Paso 5**: Por último, aplicar las activaciones para obtener la salida *y*.

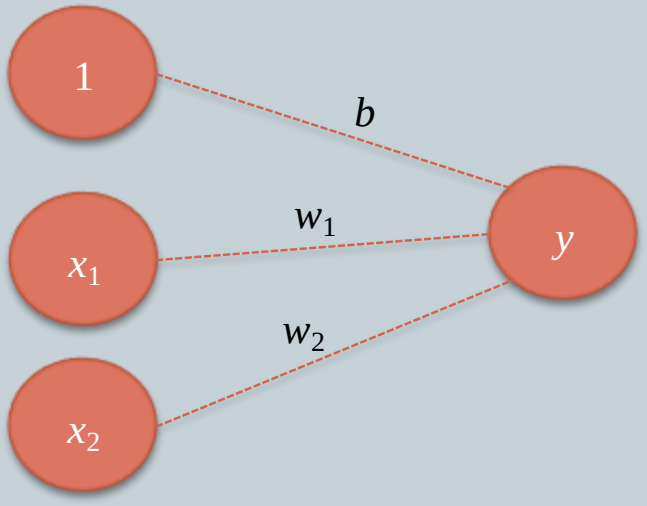
$$y = f(y_{in}) = \begin{cases} 1, & \text{si } y_{-in} \geq 0 \\ -1, & \text{si } y_{-in} < 0 \end{cases}$$

Ejemplo, I

- Desarrollar una red Adaline para la función ANDNOT con entradas bipolares y objetivos bipolares.
- La tabla de verdad para la función ANDNOT con entradas y salidas bipolares es:

x_1	x_2	o
1	1	-1
1	-1	1
-1	1	-1
-1	-1	-1

- La arquitectura es como sigue:



- Inicialmente los pesos y el sesgo se inicializan con valores aleatorios, es decir $w_1 = w_2 = b = 0.2$ y $\alpha = 0.2$.

Ejemplo, II

	Entradas			Objetivo	Net	Cambio en los pesos			Pesos			Error E
	x_1	x_2	b	o	Y_{in}	(0-yin)	Δw_1	Δw_2	Δw_3	w_1	w_2	w_3	$(o-y_{in})^2$
Época 1	1	1	1	-1	0.6	-1.6	-0.32	-0.32	-0.32	-0.12	-0.12	-0.12	2.56
	1	-1	1	1	-0.12	1.12	0.22	-0.22	0.22	0.10	-0.34	0.10	1.25
	-1	1	1	-1	-0.34	-0.66	0.13	-0.13	-0.13	0.24	-0.48	-0.03	0.43
	-1	-1	1	-1	0.21	-1.2	0.24	0.24	-0.24	0.48	-0.23	-0.27	1.47 E=5.71

Llegar hasta la época 6, w_1=0.5, w_2=-0.5, b = -0.5

Usando los pesos, se calcula el error LMS: $E = \Sigma(o-y_{in})^2 = 0.25+0.25+0.25+0.25 = 1$

$$y_m = b + \sum x_i w_i$$

$$E = (0 - y_{in})^2$$

Entradas			Entrada neta y_{in}	o	$E = (o-y_{in})^2$
x_1	x_2	b	w_1=0.5, w_2=-0.5, b = -0.5		
1	1	1	-0.5	-1	0.25
1	-1	1	+0.5	1	0.25
-1	1	1	-1.5	-1	0.25
-1	-1	1	-0.5	-1	0.25

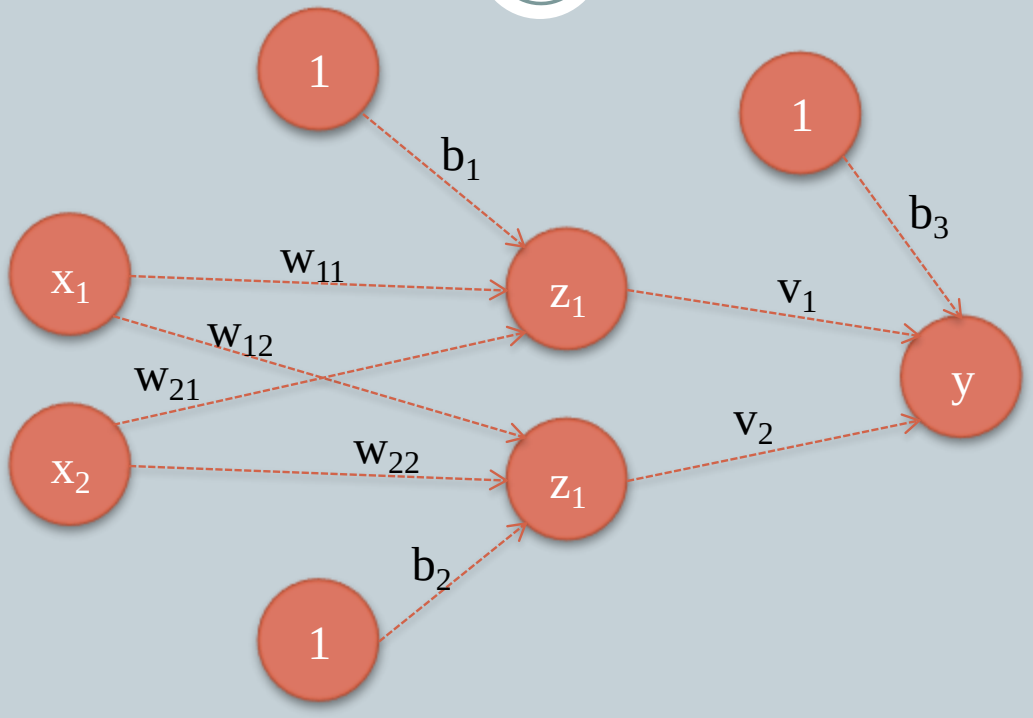
Madaline, I

- Madaline es la combinación de adalines. También se le llama Adaline de varias capas.
- Si las adalines se combinan tal que la salida de algunas de ellas se convierte en la entrada de las demás, entonces la red se convierte en varias capas. Esto forma la Madeline.
- Madaline tiene dos algoritmos de entrenamiento, MRI y MRII.

Arquitectura

- La arquitectura de una simple Madaline se muestra en la figura del acetato 12.
- La arquitectura se explica con 2 *i*/*p* neuronas, 2 neuronas ocultas y 1 neurona de salida .
- Esta arquitectura consta de dos adalines ocultas y una adaline de salida.
- La entrada x_1 y x_2 determinan la señal de salida de las adalines ocultas z_1 y z_2.
- La salida de la adaline se denota por '*y*'. z_1 y z_2 proporcionan capacidades computacionales netas que no están presentes en la red neural de una sola capa.

Madaline, II



- Adaline era una red especial que tenía sólo una unidad de salida, pero la combinación de las adalines resulta en una red Madaline como se muestra en la figura .

Madaline, III

- Esta arquitectura se asemeja a la red de alimentación hacia adelante de múltiples capas.
- Puede haber cualquier número de capas ocultas entre la entrada y la capa de salida, pero esto aumenta el cálculo de la red.
- En esta arquitectura, tanto las unidades ocultas tienen conexiones de polarización independientes como las conexiones de entrada
- La salida de las neuronas ocultas están vinculados a la neurona de salida, donde se calcula el resultado final de la red.

Algoritmo MRI, I

- El algoritmo fue propuesto por Widrow y Hoff en 1960.
- Aquí los pesos de las unidades ocultas Adaline se adaptan únicamente, mientras que los pesos de la unidad de salida son fijos.
- En primer lugar, se realiza la inicialización de los pesos entre la entrada y las unidades ocultas (que son pequeños valores aleatorios positivos).
- Se presenta la entrada, y en base a las interconexiones ponderadas la entrada neta se calcula para las unidades ocultas.
- A continuación, mediante la aplicación de las activaciones se obtiene la salida, para las unidades ocultas.
- Con esto como entrada para la capa de salida, y las interconexiones ponderadas constantes que actúan entre la capa oculta y la de salida, se encuentra la entrada de red para la neurona de salida.
- Por medio de la entrada neta, la aplicación de las activaciones permite calcular el resultado final de la red. A continuación, esto se compara con el objetivo, y se ejecuta la adaptación adecuada de pesos.

Algoritmo MRI, II

Parámetros

- Los pesos en y, v_1 y v_2 son fijos con el valor 0.5, así como el sesgo b_3 como 0.5.
- La función de activación de las unidades z_1, z_2 y y viene dada por,

$$f(p) = \begin{cases} 1, & \text{si } p \geq 0 \\ -1, & \text{si } p < 0 \end{cases}$$

- El algoritmo de entrenamiento es como sigue. El algoritmo se deduce para la arquitectura mostrada en la figura 12.
- **Paso 1**: Inicializar los pesos y sesgos, establecer el coeficiente de aprendizaje $\propto$.

  $v_1 = v_2 = 0.5$ y $b_3 = 0.5$. Los otros pesos deben ser valores pequeños aleatorios
- **Paso 2**: Cuando la condición de paro es falsa, realizar los pasos 3-9 .
- **Paso 3**: Para cada par de entrenamiento bipolar *e*:*o*, hacer los pasos 4-8.
- **Paso 4**: Fijar las activaciones de las unidades de entrada:

Algoritmo MRI, III

$$x_i = e_i \text{ para } i = 1 \text{ hasta } n$$

- **Paso 5**: Calcular la entrada neta de las unidades adaline ocultas.

$$z_{\text{-in1}} = b_1 + x_1 w_{11} + x_2 w_{21}$$

$$z_{\text{-in2}} = b_2 + x_1 w_{12} + x2 w_{22}$$

- **Paso 6**: Encontrar la salida de la unidad adaline oculta mediante la activación mencionado anteriormente.

$$z_1 = f(z_{\text{-in1}})$$

$$z_2 = f(z_{\text{-in2}})$$

- **Paso 7**: Calcular la entrada neta de salida.

$$y_{\text{-in}} = b_3 + z_1 v_1 + z_2 v_2$$

Aplicar la activación para obtener la salida de la red.

$$y = f(y_{\text{-in}})$$

- **Paso 8**: Encontrar el error y actualizar los pesos.

Algoritmo MRI, IV

Si $o = y$, no se actualizan los pesos

Si $o \neq y$, entonces

Si $o = 1$, entonces adaptar el peso de la unidad z_j cuya entrada neta es cercana a 0

$$w_{ij}(\text{nuevo}) = w_{ij}(\text{viejo}) + \propto(1 - Z_{\text{inj}})\, x_i$$

$$b_j(\text{nuevo}) = b_j(\text{viejo}) + \propto(1 - Z_{\text{inj}})$$

Si o = -1, entonces actualizar los pesos de todas las unidades z_k los cuales tienen una entrada neta positiva

$$w_{ik}(\text{nuevo}) = w_{ik}(\text{viejo}) + \propto(-1 - Z_{\text{ink}})\, x_i$$

$$b_k(\text{nuevo}) = b_k(\text{viejo}) + \propto(-1 - Z_{\text{ink}})$$

- **Paso 9**: Prueba para la condición de paro.

- La condición de paro puede ser el cambio de peso o el número de épocas, etc.

Algoritmo MRII, I

- Este algoritmo, propuesto por Widrow, Winter y Banter en 1987, proporciona un método para actualizar todos los pesos en la red.
- Por lo tanto, permite el entrenamiento de los pesos en todas las capas de la red. Aquí se pueden utilizar varias unidades de salida.
- El error total de cualquier patrón de entrada se da como la suma de los cuadrados de los errores en cada unidad de salida.
- Este algoritmo es diferente de algoritmo de MRI en la parte de actualización de los pesos.
- El algoritmo de entrenamiento es el siguiente:
- **Paso 1**: Inicializar los pesos (todos los pesos con un valor aleatorio). Establecer el coeficiente de aprendizaje α.
- **Paso 2**: Cuando la condición de paro es falsa, realizar los pasos 3-11.
- **Paso 3**: Para cada par de entrenamiento bipolar, realizar los pasos 4-10.
- **Paso 4**: Establecer la activación de las unidades de entrada.

Algoritmo MRII, II

$$x_i = e_i$$

- **Paso 5**: Calcular la entrada neta de cada unidad oculta.

$$z_{\text{-in1}} = b_1 + x_1 w_{11} + x_2 w_{21}$$

$$z_{\text{-in2}} = b_2 + x_1 w_{12} + x2 w_{22}$$

- **Paso 6**: Aplicar las activaciones para calcular la salida de cada unidad oculta.

$$z_1 = f(z_{\text{-in1}})$$

$$z_2 = f(z_{\text{-in2}})$$

- **Paso 7**: Calcular la entrada neta de y.

$$y_{\text{-in}} = b_3 + z_1 v_1 + z_2 v_2$$

Calcula la salida y aplicando las activaciones

$$y = f(y_{\text{-in}})$$

- **Paso 8**: Determinar el error y realizar la actualización de los pesos.

Si $o \neq y$.

Algoritmo MRII, III

Realizar los pasos 9-10 para cada unidad oculta cuya entrada neta es la más cercana a 0. Iniciar con la unidad cuya entrada neta es cercana a 0, a continuación, para la segunda más cercana, etc.

- **Paso 9**: Cambiar la salida de las unidades.

  Si es +1, cambiar a -1

  Si es -1, cambiar a +1

- **Paso 10**: Recalcular la salida de la red.

  Si el error es reducido:

  Ajustar el peso en esta unidad.

  (Tomar la salida como la nueva salida calculada y utilizarla para la regla delta).

- **Paso 11**: Prueba de condición de paro.

- La condición de paro puede ser el número de iteraciones de la actualización de pesos.

Ejemplo

- Formar una red Madaline para la función XOR con la entrada y destino bipolar, utilizando el algoritmo MRI.
- La tabla de verdad para la función XOR es:

x_1	x_2	Y
1	1	-1
1	-1	1
-1	1	1
-1	-1	-1

- La arquitectura de la red Madaline es: