Las redes neuronales como modelo de computación distribuida y autoprogramable

1 . Características de la computación neuronal

- A / Arquitectura modular
- B / Organización en multicapas
- C / Gran número de procesadores elementales con gran conectividad entre ellos
- D / Cada procesador realiza una función local generalmente analógica y no lineal (usualmente suma ponderada seguida de sigmoide o abrupta)
- E / Eliminación parcial de necesidad de programación
- F / Empleo de algoritmos de aprendizaje
- G / Tolerancia a fallos

2 . Dificultades de empleo

- A / Hay problemas en los que no es aplicable
- B / No hay metodología clara
- C / No existen entornos de desarrollo
- D / Se pasa directamente del nivel del conocimiento al de procesadores (por eso no hay entornos de desarrollo)

Modelos de neuronas

1 . Modelos analógicos

- Suma ponderada (lineal) + función de decisión (no lineal) + retardo

Yj* (t+ t) = wij (t) * xi (t)

Yj (t + t) = Gj (Yj* (t))

- Posibles formas de funciones de decisión:

Abrupta : **Yj (t + t) = 1 si Yj* (t) >= umbral**

Yj (t + t) = 0 si Yj* (t) < umbral

Sigmoide : **Yj (t + t) = (1 + exp (− Yj* / T)) ^ (−1)**

(1 − exp (− Yj* / T))

Tangente hiperbólica: $$Yj(t+t) = \frac{}{(1+\exp(-Yj^*/T))}$$

Expansión del modelo analógico

- El modelo analógico está limitado porque al ser lineal en su función de excitación, sólo es posible clasificar ciertos grupos de datos (aquellos separables linealmente).
- Para resolver esta limitación, se introduce un codificador no lineal para expandir los espacios de entrada y/o salida y los haga no lineales.

2 . Modelos lógicos

- Sólo se modifican del modelo analógico las entradas, que ahora son

variables lógicas (0 o 1).

- La suma ponderada es ahora una unión lógica.
- El producto es ahora una intersección lógica.
- Los campos receptivos Vj y Vj* permanecen iguales al modelo analógico.
- Las estructuras FIFO de entradas y salidas también permanecen

Por lo tanto, una neurona lógica puede calcular cualquier función lógica de sus entradas, sus salidas y las salidas de otras neuronas en tiempos anteriores.

$$Yj(t+t) = wij(t) * mi(t)$$

Con wij entre 0 y 1 . Vale 1 si el término mínimo está presente y 0 en caso contrario.

mi (t) toma la notación : para dos entradas x1 , x2 :

m0 = x1' x2 ' m1 = x1' x2 m2 = x1 x2 ' m3 = x1 x2

- Este modelo es similar a la teoría de autómatas finitos deterministas, por lo que las redes neuronales lógicas están formalizadas en sus aspectos de análisis y síntesis.

Expansión del modelo lógico: modelo probabilístico

- Es similar al modelo lógico salvo en la función de excitación.

- En este caso, los pesos son probabilidades de la existencia de un término mínimo mi que excluye a todos los demás.

$$Yj(t+t) = pij(t) * mi$$

Donde pij (t) es la probabilidad de disparo de la neurona j en el instante t si la entrada es mi .

3 . Modelos inferenciales

- Estos modelos se han introducido para lanzar un puente entre la IA simbólica y la IA conexionista. Esto se hace permitiendo funciones de computación local estructuradas (marcos, guiones) o reglas.

- Las funciones de excitación y de activación (umbral) se sustituyen por la evaluación del antecedente de una regla y la utilización de una tabla LUT (para el caso de la utilización de reglas).

- De esta forma, es posible introducir redes borrosas y tratarlas con computación neuronal.

- Los modelos basados en reglas no pueden simular todo el poder de las reglas en la IA simbólica, por lo que se suelen emplear marcos en vez de reglas.

Tipos de aprendizaje

1 . Reglas correlacionales

- Aprendizaje de tipo no supervisado.
- Empleo de la regla de Hebb.

Regla de Hebb

Toda la información necesaria para el aprendizaje la transporta la propia señal de entrada.

Wij (t + t) = wij (t) + lr * xi (t) * yj (t)

- A veces se utiliza la variante de la regla de Hebb en la que se acota la subida creciente de los pesos en cada ciclo del aprendizaje. Se introduce un factor de olvido en la red.

Wij (t + t) = wij (t) + lr * xi (t) * yj (t) − dr * wij (t)

Aprendizaje asociativo (redes asociativas)

- Se utiliza la regla de Hebb.
- Se trata de descubrir coincidencias o agrupaciones implícitas en los datos de entrada de la red.

- De esta forma: a/ Se disminuye la dimensionalidad del espacio de entrada.

b/ Se extraen características de los datos de entrada.

Aprendizaje competitivo (redes de inhibición lateral)

- Primero se calcula cuál es la neurona cuyo peso está más cerca del vector de entrada.

Distancia (wij , xi) = mínima

- Para esta neurona es para la única que se modifica el peso (las demás permanecen igual).

Wij (t + t) = wij (t) + lr * yj (t) * (xi (t) − wij (t))

- Su función es detectar contrastes.

2 . Minimización de una función del error : retropropagación del gradiente

- Aprendizaje supervisado.
- Se trata de evaluar lo adecuado o inadecuado de una respuesta a un estímulo. Y para ello se necesita conocer la respuesta deseada.

- Se utiliza una función del error para modificar los pesos o bien una función del error cuadrático medio si sólo se sabe el error global de salida de la red.

e (t) = dj (t) − yj (t)

E = 1/2 (dj − yj)2

- Para seleccionar el conjunto de entrenamiento se debe usar todo el conocimiento disponible sobre el problema a resolver.
- Para el caso de que se conozcan todas las respuestas deseadas para cada neurona:

Wij (t + t) = wij (t) + lr * (dj (t) − yj (t) * xi (t)

- Para el caso de que sólo se conozca la respuesta deseada para toda la red, se debe aplicar la retropropagación del error hacia atrás, pasando la parte del error correspondiente a cada capa de neuronas.

W = − a * (J E (w) / J w)

- En este caso, primero se calcula la modificación de los pesos para la última capa (capa de salida) y luego se retropropaga el error a la capa anterior y así sucesivamente.

- **. Funciones de refuerzo**

- Estas funciones son aplicables cuando existe el caso de retropropagación del gradiente pero no conocemos con suficiente precisión el par xi y dj , y por lo tanto no podemos retropropagar el error a capas anteriores puesto que no lo conocemos .

- Para estos casos, se utiliza una función de refuerzo que se encarga de afianzar las conexiones de las neuronas que estaban activas cuando se cumplió un requisito general llamado premio z(t).

- Para el modelo analógico, los pesos varían así:

Wij (t + t) = wij (t) + yj (t) * xi (t) * z (t)

Donde z(t) es la señal de refuerzo.

- Los modelos que han dado mejores resultados son los lógico−probabilísticos y se les suele dotar a las neuronas de una memoria local.

Tipos de neuronas

1 . Lógica−secuencial de MacCulloch y Pitts

- Neurona lógica.
- Umbral + retardo.
- El umbral aumenta tras cada ciclo de aprendizaje.
- Equivalente a un autómata finito de dos estados.

2 . Perceptrón de Rosenblatt

- Neurona analógica.

- Suma ponderada (lineal) + umbral (no lineal) + retardo.
- Aprendizaje con minimización de una función del error: e (t) = dj (t) − yj (t)

3 . Adalina de Widrow

- Neurona analógica.
- Suma ponderada (lineal) + identidad (lineal) + retardo.
- Aprendizaje por descenso en la superficie del gradiente del error. Se toma un umbral del error que se quiere permitir y un número máximo de ciclos de aprendizaje. Para cada ciclo, se calcula el error cuadrático global obtenido a la salida de la red y se compara con el permitido y en caso de que el obtenido sea menor que el permitido o en caso de que se haya llegado al máximo de ciclos permitidos, el proceso de aprendizaje ha finalizado.

Simbiosis entre IA simbólica y IA conexionista

Características de la IA conexionista:

- La computación neuronal es **distribuida sobre procesadores de grano pequeño** (arquitectura paralela tipo SIMD, memoria RAM local y cálculo local de la misma duración que el cálculo de toda la red).
- El **número de procesadores** (neuronas) es **muy elevado** pero con una **capacidad de cálculo muy limitada** (si no fuera limitada, hablaríamos de ordenadores en paralelo).
- La **TG de una neurona es la mínima posible**, y consta del cálculo de la función local y del algoritmo de aprendizaje.
- La **computación de cada procesador es de tipo paramétrico** para que pueda ser **autoprogramable.**
- Al igual que en la IA simbólica, en la IA conexionista también **impera el clasificador como TG más utilizada** para asociar entradas x a salidas y.
- **La IA conexionista bebe de la neurología** y aprende de ella, cosa que no ocurre con la IA simbólica.
- En la IA conexionista **falta metodología y entornos de desarrollo**, cosa que sí existe por lo menos en parte de la IA simbólica.

Diferencias entre la IA simbólica y la IA conexionista:

- En **el ámbito de procesadores son similares**.
- El **salto** del nivel de conocimiento al nivel físico es mayor en redes neuronales que el salto previo al nivel simbólico en la IA simbólica, por lo que hay que realizar **más esfuerzo en el análisis de TG en la IA conexionista.**
- En **la IA conexionista se sustituye la programación por el aprendizaje**. Y por lo tanto, en la IA conexionista se exige computación modular, de grano pequeño, paramétrica y con algoritmos propios de aprendizaje.
- **Cada alternativa es recomendable para un tipo de problema**.

Una forma de conectar la IA simbólica con la IA conexionista es utilizar a la simbólica para, una vez modelado el conocimiento inicial, utilizarlo para seleccionar la arquitectura de red más idónea para resolver el problema utilizando ejemplos de entrenamiento.