REDES COMPETITIVAS Y DE KOHONEN

Redes competitivas

Hasta ahora hemos estudiado redes en las que muchas salidas están normalmente activas a la vez. En el aprendizaje competitivo sólo una unidad de salida está activa en cada momento.

Las unidades de salida compiten entre si para ser la que se activa como respuesta a una entrada determinada.

Un aspecto importante de la competición es que lleva a la optimización a nivel local sin necesidad de utilizar una serie de recursos en un control global. Esto es de especial importancia en los sistemas distribuidos en los que supone un enorme coste.

Redes competitivas

Las neuronas de una red competitiva reciben idéntica información de la entrada, pero compiten por ser la única que se activa.

Cada neurona se especializa en un área diferente del espacio de entradas, y sus salidas se pueden utilizar para representar de alguna manera la estructura del espacio de entradas.

Vamos a introducir topologías y reglas de aprendizaje específicas que están dirigidas a la competición.

La competición se muestra como un método rápido de organización automática de los recursos de la red y se ha mostrado muy efectiva en la práctica.

Redes competitivas

Stephen Grossberg introdujo la mayor parte de las ideas de las redes competitivas. Sus redes trabajan en tiempo continuo y se definen en términos matemáticos complejos.

Teuvo Kohonen introduce un estilo mas próximo a la ingeniería y adopta una serie de principios fácilmente implementables en sistemas digitales. Presentó en 1982 un modelo de red neuronal con capacidad para formar mapas de características de manera similar a como ocurre en el cerebro, a través de una organización matricial de neuronas artificiales.

Redes competitivas

La competición puede ser de dos tipos:

- **Competición dura**, sólo una neurona consigue los recursos
- **Competición blanda**, hay un vencedor claro, pero sus vecinos comparten un pequeño porcentaje de los recursos del sistema.

En neurobiología, la competición blanda está ampliamente extendida y ha sido la inspiración para los modelos desarrollados hasta el momento.

Redes competitivas

Una red de neuronas competitiva típica consiste en una capa de neuronas en la que todas reciben la misma entrada.

La neurona que presenta la mejor salida (la máxima o la mínima según el criterio) es declarada vencedora.

El concepto de elegir un vencedor a menudo requiere un controlador global que compare cada salida con todas las demás.

Deseamos construir una red que encuentre la mayor o la menor salida, sin necesidad de control global. Este sistema se denomina a menudo *winner-take-all* (el ganador lo consigue todo)

Redes *winner-take-all*

Supongamos N entradas x_1, x_2, ..., x_N. Queremos crear una red de N salidas que dé una única salida positiva en la neurona que corresponde a la entrada más grande, mientras todas las demás neuronas presentan salida 0.

$$y_k = \begin{cases} 1 & x_k \text{ es el mayor} \\ 0 & \text{en otro caso} \end{cases}$$

Una posible topología sería:

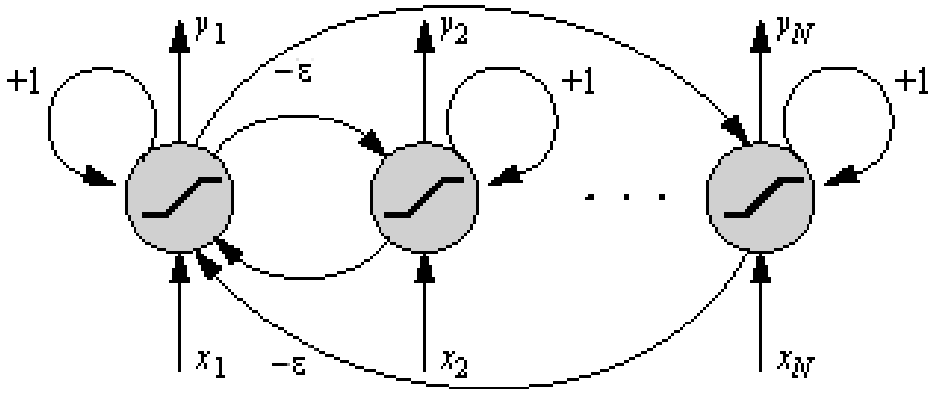


Figure 7-1 Winner-take-all network

Redes *winner-take-all*

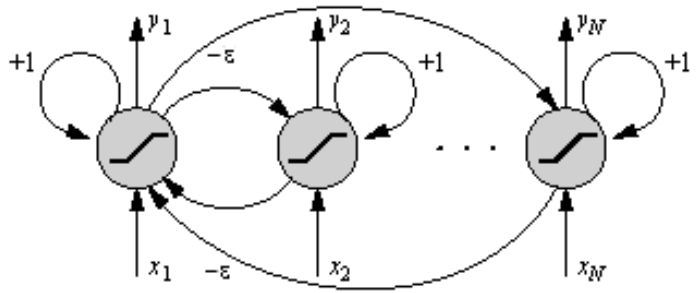


Figure 7-1 Winner-take-all network

Cada neurona tiene una autoalimentación con peso fijo 1 y está conectada lateralmente a todas las demás neuronas por pesos negativos ε con $0 < \varepsilon < 1/N$.

Suponemos todas las entradas positivas, la pendiente de las neuronas igual a 1 y la condición inicial sin entrada es salida 0.

Redes *winner-take-all*

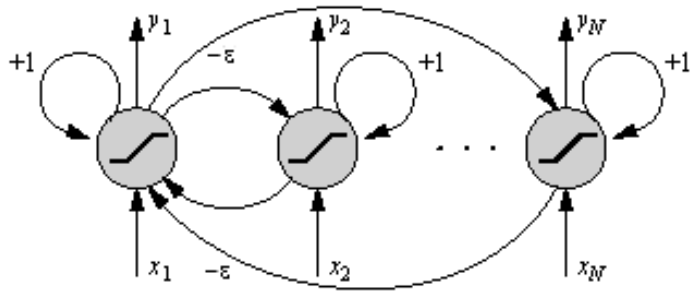


Figure 7-1 Winner-take-all network

Al presentar una entrada, la conexión lateral lleva a todas las neuronas hacia 0. Según las más pequeñas se aproximan a 0, la más grande será menos afectada por las conexiones laterales. A la vez, la autoalimentación eleva su salida, lo que refuerza la tendencia a 0 de las otras neuronas.

Las neuronas compiten por la actividad de salida y sólo una lo consigue.

Redes *winner-take-all*

La salida que queremos obtener de la red es

$$y_i = \begin{cases} 1 & i = i^* \text{ la neurona vencedora} \\ 0 & \text{para todas las otras neuronas} \end{cases}$$

por tanto la regla de aprendizaje será

$$w_{i^*}(n+1) = w_{i^*}(n) + \eta(x(n) - w_{i^*}(n))$$

donde i^* es la neurona ganadora. Todas las demás mantienen sus pesos anteriores.

El tamaño del paso η ($0 < \eta < 1$) controla el tamaño de la actualización en cada paso. Si es grande la red convergerá rápido pero será menos estable. Se puede usar un paso variable comenzando por uno grande y disminuyéndolo progresivamente.

Redes *winner-take-all*

Un **criterio** lógico para seleccionar el ganador en una red de este tipo podría ser la proximidad, así la neurona más próxima a la entrada actual sería la vencedora.

- Mediante el **producto escalar**: $w^t \cdot x$. Cuanto más próximos estén los vectores mayor será el resultado. Este criterio es también sensible al tamaño de los vectores, por lo que habría que normalizarlos, lo que afectaría a las entradas.
- Mediante la **distancia euclídea**: $\|x-w\| = \sqrt{\sum_k (x_k - w_k)^2}$

Es una función cara (por la raíz)

- Mediante la **norma L1** $\|x-w\|_{L1} = \sum_k |x_k - w_k|$

Redes *winner-take-all*

- Sólo una salida está activada en cada momento
- Cada salida se distingue de las demás respondiendo a alguna característica.
- La red amplifica las diferencias creando un mecanismo selector que puede ser aplicado a muy diferentes problemas.
- Una aplicación típica es colocar una red de este tipo en la salida de otra red para seleccionar la salida más grande.
- De esta manera la red toma una decisión basada en la respuesta más probable.

Es trivial implementar esta operación en un sistema digital, por lo que esta arquitectura no se usa en la práctica.

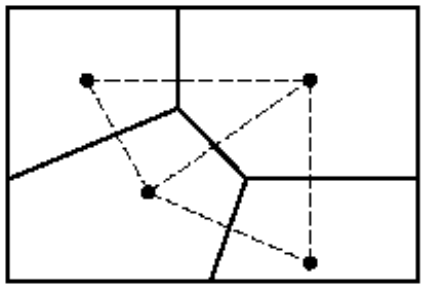
Redes competitivas

- El objetivo del aprendizaje competitivo es crear una regla de aprendizaje que pueda ser aplicada a una topología monocapa de modo que asigne las neuronas a diferentes áreas del espacio de entradas.
- Cada neurona actúa como un codificador de dicho área.
- El aprendizaje competitivo es no supervisado, es decir, no necesita una colección de respuestas deseadas previa al aprendizaje.
- En las redes estudiadas hasta ahora, todos los pesos responden en mayor o menor grado al espacio total de entradas, mientras que en éstas cada neurona se especializa en una porción de dicho espacio.

Redes competitivas

El algoritmo competitivo permite a una red monocapa agrupar y representar datos que están situados en un mismo entorno en el espacio de entradas. Cada entorno se representa por la salida de una única neurona.

Los pesos de cada neurona representan puntos en el espacio de entrada llamados *vectores prototipo.* Uniendo estos vectores y calculando las mediatrices de los segmentos, se forman unas divisiones llamadas teselaciones de Voronoi.



Redes competitivas

En telecomunicaciones, la reducción de los datos es necesaria por razones económicas, con esta técnica en lugar de transmitir los valores de cada dato, éstos son agrupados en celdas cuyos centros son conocidos por el transmisor y el receptor, así sólo se transmite el número de la celda.

Si disponemos de suficiente número de celdas, el error entre el dato y su centro es pequeño, i.e. la fidelidad es alta.

Así se consiguen dos mejoras:

- no es necesario enviar los datos con precisión completa, sino sólo un entero
- los datos más frecuentes pueden ser codificados con números pequeños, lo que hace decrecer el coste total.

Redes competitivas

Para conseguir una salida de buena calidad, es necesario utilizar un buen algoritmo de creación de las celdas que minimice las distancias para cada entrada.

A falta de una respuesta deseada, lo mejor que podemos hacer para agrupar es usar la información sobre la distribución de los datos para separar las entradas en grupos que ocupan la misma región en el espacio de entradas buscando regiones de alta densidad de datos.

Utilizaremos un algoritmo clásico de agrupación de datos en celdas. La idea es encontrar la mejor agrupación de N puntos en K celdas C_i de modo que la distancia total entre los puntos y sus respectivos centros sea mínima.

Redes competitivas

El criterio sería: $J = \sum_{i=1}^{K} \sum_{n \in C_i} \left| x_n - \gamma_i \right|^2$

donde γ_i es el centro de la clase C_i.

El algoritmo asigna aleatoriamente los datos a las clases y calcula los centros según la fórmula:

$$\gamma_i = \frac{1}{N_i} \sum_{n \in C_i} x_n$$

reasigna los datos a la celda más cercana y repite el cálculo.

Se puede demostrar que J decrece en cada paso hasta que alcanza un mínimo.

Redes competitivas

Si consideramos J como criterio de comportamiento, su mínimo se puede obtener utilizando el gradiente respecto de las incógnitas (los centros). Y utilizando la misma técnica de la regresión lineal, un algoritmo de modificación de los centros sería:

$$\Delta \gamma_i(n) = \eta(x(n) - \gamma_i(n))$$

que coincide exactamente con la actualización de una red competitiva si relacionamos el centro de la celda con el peso de la neurona i-esima.

Es claro que las redes competitivas implementan una versión on-line del algoritmo clásico de agrupación en celdas.

Limitaciones de las redes competitivas

- El algoritmo de aprendizaje sólo se puede aplicar a estructuras monocapa i.e. sólo se pueden crear particiones convexas del espacio de entrada.
- El número de celdas ha de ser definido a priori.
- El paso η debe elegirse adecuadamente (mejor alto al principio y decreciendo durante el entrenamiento.
- Si el peso de una neurona está muy alejado de los datos, nunca resultará vencedora, y por tanto su peso nunca se adaptará (*neurona muerta*). Este tipo de neuronas es común cuando se aplica el algoritmo a redes y conjuntos de datos muy grandes.

Limitaciones de las redes competitivas

Para evitar el problema de las *neuronas muertas* se puede hacer una modificación del algoritmo de manera que si una de ellas empieza a vencer más de lo que la corresponde, es penalizada:

cada neurona cuenta el número de veces que resulta vencedora

$$c_i(n+1) = c_i(n) + \beta(o_i(n) - c_i(n))$$

$o_i(n)$ es la salida de la competición actual (1 si gana, 0 si pierde), β es una constante positiva pequeña (0.0001). Así, si la neurona gana c_i aumenta y si pierde disminuye, de modo que si una neurona no gana nunca, c_i es negativo y si gana muchas veces se hace relativamente grande.

Limitaciones de las redes competitivas

Cada neurona actualiza su propio umbral según la fórmula: $b_i = \gamma(1/N - c_i)$

donde γ es una constante positiva (10 puede ser un buen valor).

El b_i es el término de penalización por ganar demasiado a menudo y se resta de la distancia euclídea (si se considera el producto escalar se suma).

$$D(w_i, x) - b_i$$

Si la neurona gana más de lo que le corresponde, b_i será negativo y la distancia a la entrada actual, y por tanto las posibilidades de ganar, disminuyen.

Después de determinar el ganador, los pesos se actualizan según la regla competitiva.

Limitaciones de las redes competitivas

El número de centros, determinado por el número de neuronas, afecta a la calidad de la agrupación. El problema es que nada en el algoritmo nos permite encontrar el mejor número de celdas. El diseñador debe elegir ese número, bien utilizando información sobre el problema o bien por ensayo y error.

Otro problema que no tiene fácil solución es el hecho de que no hay manera de medir hasta que punto es buena la neurona vencedora. Sabemos que la neurona que gana es la más cercana a la entrada, pero no podemos cuantificar si ésta está cerca o lejos de los pesos de la neurona.

Competición blanda

Hasta ahora en la competición sólo había un ganador, una neurona permanece activa y todas las demás están inactivas. La competición blanda permite no sólo a la ganadora, sino también a sus vecinas estar activas.

Crea una bolsa de actividad en el espacio de salida donde la neurona más próxima es la más activa y sus vecinas están menos activas.

Una red de este tipo se puede crear usando alimentación lateral. En este caso los pesos laterales varían según la distancia a la neurona vencedora. Las neuronas cercanas se excitan unas a otras, mientras que inhiben a las alejadas.

Se actualizan varios pesos por cada patrón que se presenta pero atenuándose la actualización según la distancia.

Competición blanda

La competición blanda construye relaciones de vecindad entre las neuronas i.e. establece una especie de métrica en el espacio de salida.

Al contrario que las estructuras anteriores, posibilita la existencia de aplicaciones topológicas, que conservan las relaciones de vecindad.

Una de las aplicaciones importantes es la reducción de datos conservando la información de proximidad entre ellos. Las proyecciones creadas con esta técnica conservan en detalle la estructura celular de los datos. Si dos entradas se transforman en salidas próximas, aquellas deben también ser próximas en el espacio de entrada.

Redes de Kohonen

Teuvo Kohonen presentó en 1982 un modelo de red neuronal con capacidad para formar mapas de características de manera similar a como ocurre en el cerebro, intenta simular los mapas topológicos de los fenómenos sensoriales y motores existentes en el cerebro, a través de una organización matricial de neuronas artificiales.

Una de las propiedades importantes del cerebro es el significativo orden de sus unidades de proceso. Este orden hace que unidades estructuralmente idénticas tengan una diferente funcionalidad debida a parámetros internos que evolucionan de forma diferente según dicha ordenación de las células. Esta propiedad parece ser de fundamental importancia para la representación de imágenes visuales, abstracciones, etc.

Redes de Kohonen

Cuando pensamos realizamos una compresión de la información formando representaciones reducidas con los aspectos más relevantes de las mismas, sin que se produzca por ello ninguna pérdida de conocimiento acerca de las interrelaciones entre ellas. El propósito parece ser la creación de imágenes a varios niveles de abstracción.

La estructura topológica de la red absorbe a su vez aquella que se produce entre las características de los datos, y por tanto el sistema no es solo capaz de realizar una clasificación de los estímulos, sino que además nos pondrá de relieve y conservará las relaciones existentes entre las diferentes clases obtenidas.

Redes de Kohonen

El objetivo de Kohonen era demostrar que un estímulo externo (información de entrada) por sí solo, suponiendo una estructura propia y una descripción funcional del comportamiento de la red, era suficiente para forzar la formación de mapas topológicos de las informaciones recibidas del exterior.

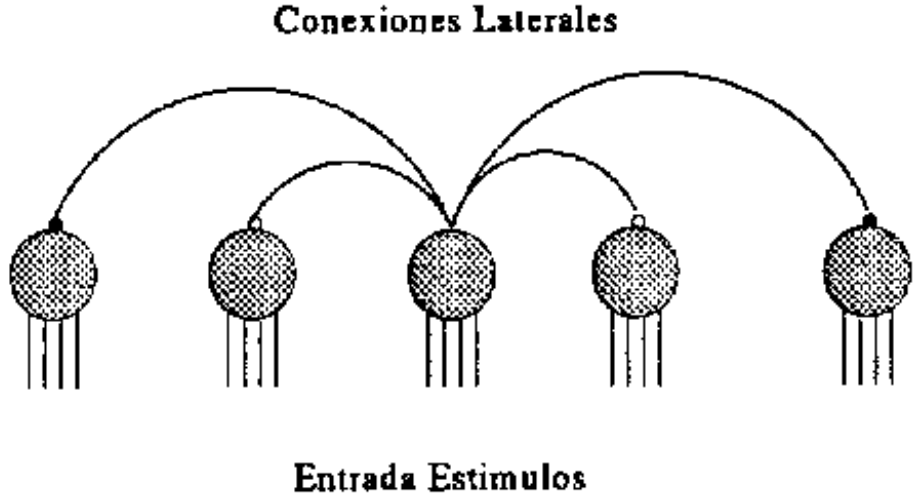
El modelo tiene dos variantes:

- **LVQ** (learning vector quantization)
- **TPM** o **SOM** (topologic preserving map, o self organizing map).

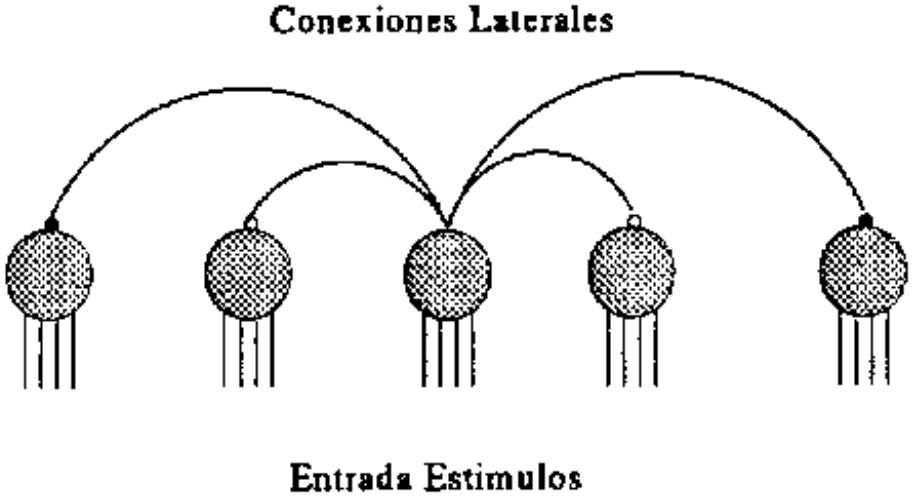
Los vectores de entrada a la red pueden ser de diferentes dimensiones, siendo en el caso de LVQ de una dimensión, y en el caso de TPM bidimensionales o incluso tridimensionales.

Arquitectura LVQ

Se trata de una red de dos capas con N neuronas de entrada y M de salida. Cada una de las N neuronas de entrada se conecta a las M de salida mediante conexiones hacia delante. Entre las neuronas de las capas de salida existen conexiones laterales, ya que cada una de ellas tiene influencia sobre sus vecinas a la hora de calcular los pesos de las conexiones hacia delante entre la capa de salida y la capa de entrada.



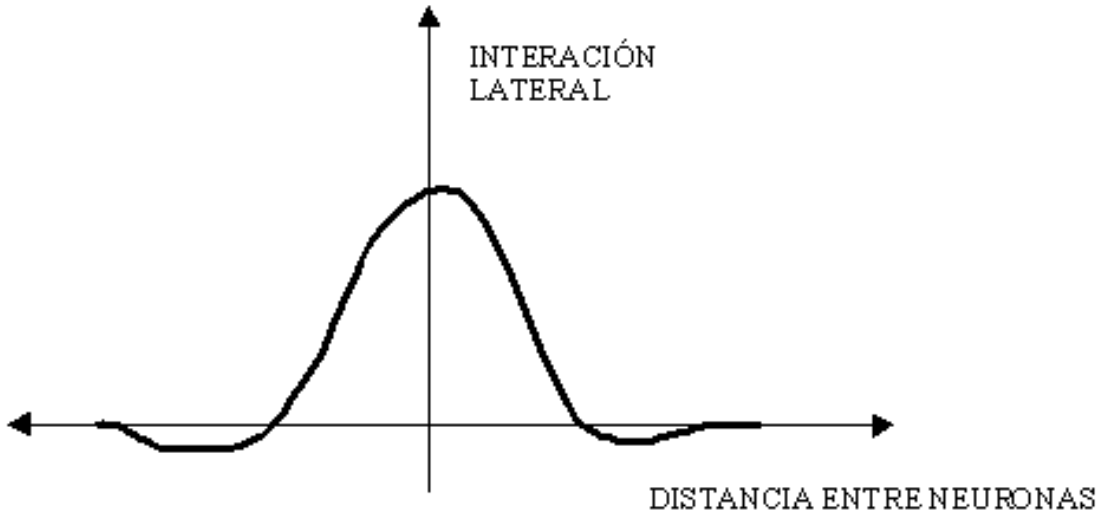
Arquitectura LVQ



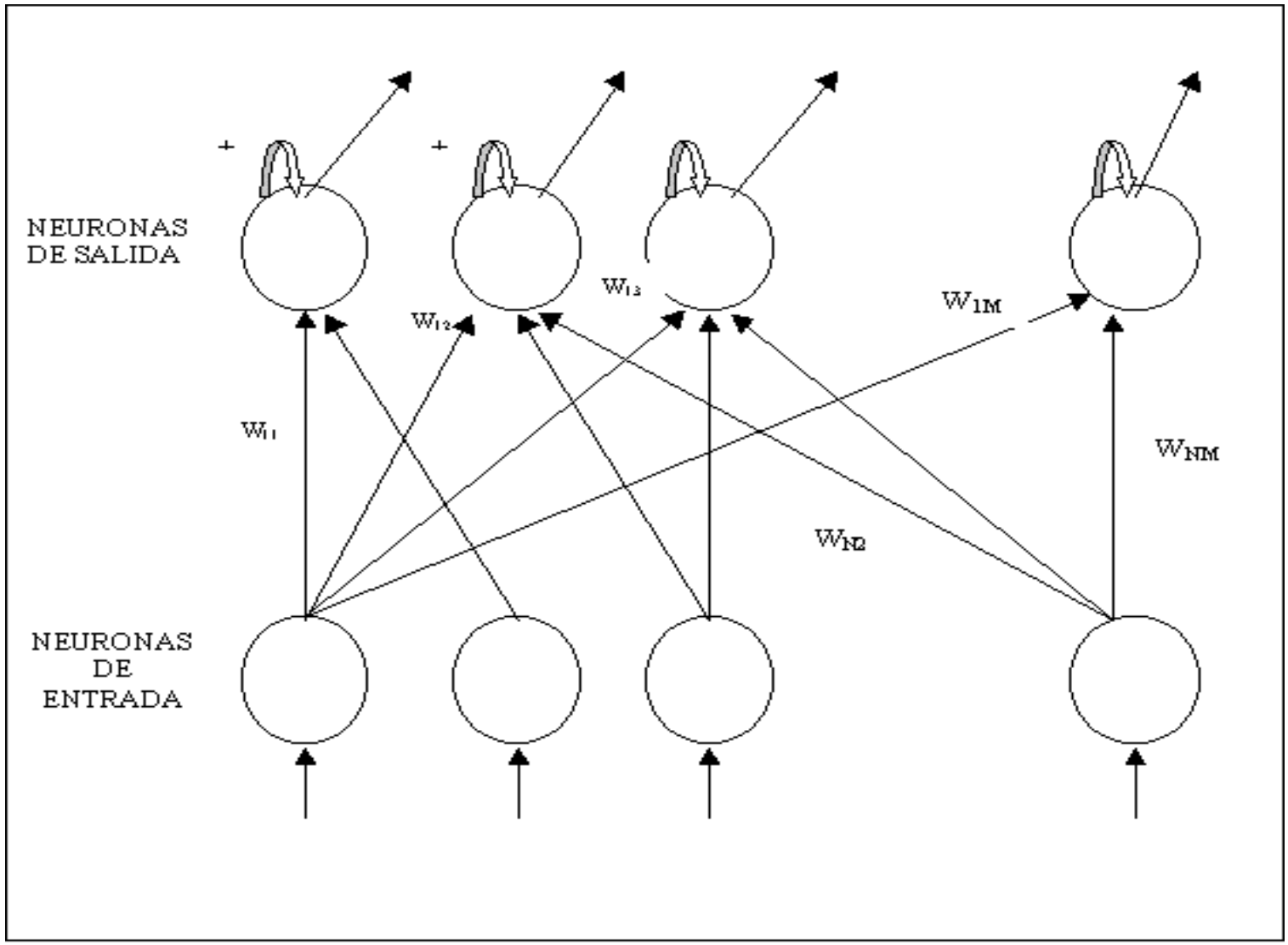
En las neuronas de la capa de salida, cada neurona está conectada con otras de su entorno, de manera que produce una excitación en las más próximas y una inhibición en las más alejadas. Tanto la excitación como la inhibición laterales son gradualmente más débiles a medida que nos alejamos de la neurona en cuestión.

Arquitectura LVQ

Las influencias laterales se suelen representar por una Gaussiana: mientras más cercana más influencia, si la distancia es muy grande, la influencia llega a ser despreciable



Arquitectura LVQ

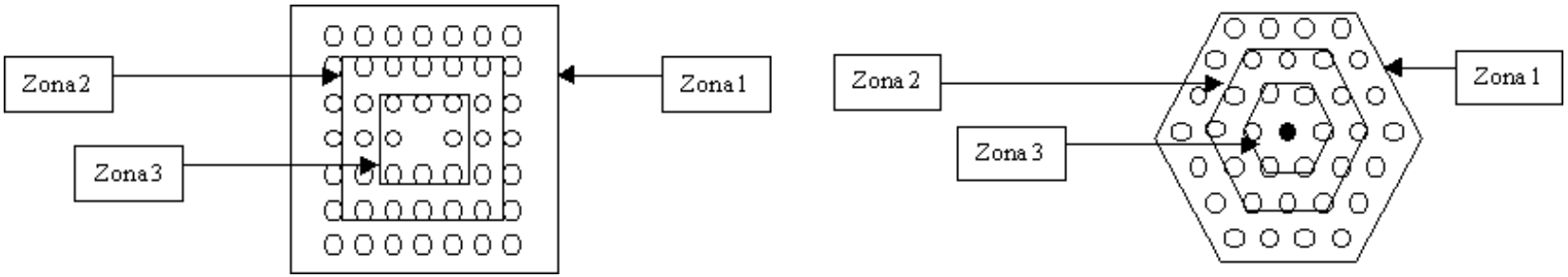


Arquitectura SOM

Trata de establecer una correspondencia entre los datos de entrada y un espacio bidimensional de salida, de modo que ante datos con características comunes se activen neuronas situadas en zonas próximas de la capa de salida.

Cada neurona de la capa de entrada está conectada con cada una de las neuronas de la capa de salida mediante pesos

Las interacciones laterales siguen existiendo en relación con la distancia que se toma como una zona bidimensional alrededor de la neurona.



Funcionamiento de la red de Kohonen

Se presenta a la entrada una información en forma de vector de N componentes (la capa de entrada tiene N neuronas)

Las M neuronas que forman la capa de salida reciben la información a través de conexiones hacia delante con pesos w_{ji} y también a través de las conexiones laterales con el resto de las neuronas de la capa de salida, de manera que la red evoluciona hasta encontrar una situación estable en la que se activa una neurona de salida, esta es la vencedora.

Una vez encontrada la vencedora, los pesos se actualizan, no sólo para la vencedora, sino para todas las de su entorno.

Funcionamiento de la red de Kohonen

Para simplificar los cálculos, se supone que la inhibición lateral produce una distribución gaussiana centrada en la neurona vencedora, así, en lugar de calcular recursivamente la actividad de cada neurona, simplemente encontramos la ganadora y suponemos que las otras tienen una actividad proporcional a la función gaussiana evaluada en la distancia entre dicha neurona y la ganadora:

$$w_i(n+1) = w_i(n) + \Lambda_{i,i^*}(n)\,\eta(n)\,(x(n) - w_i(n))$$

$$\Lambda_{i,i^*}(n) = e^{\left(\frac{-d^2_{i,i^*}}{2\sigma^2(n)}\right)}$$

Donde d es la distancia entre las neuronas y σ disminuye con la iteración

Funcionamiento de la red de Kohonen

La gaussiana comienza por cubrir casi todo el espacio y se reduce progresivamente hasta ser casi exclusivamente la vencedora la que se actualiza.

Para las cercanas a la vencedora, las actualizaciones se reducen exponencialmente según la distancia a aquella.

Según se reduce la gaussiana, la red cambia de competición muy blanda a casi dura.

$$w_i(n+1) = w_i(n) + \Lambda_{i,i^*}(n)\,\eta(n)\,(x(n) - w_i(n))$$

$$\Lambda_{i,i^*}(n) = e^{\left(\frac{-d^2_{i,i^*}}{2\sigma^2(n)}\right)}$$

La red de Kohonen

La red de Kohonen es tan diferente de las otras debido a su capacidad para trasladar la estructura del espacio de entradas al espacio de salidas: crea un espacio de salidas discreto donde se **conservan las relaciones topológicas** entre los elementos del espacio de entrada, lo que quiere decir que **la distribución de los datos de entrada se conserva aproximadamente**.

La conservación de los entornos es importante para aplicaciones que requieren la **conservación de una métrica** en el espacio de entradas y la red lo hace de una **manera no supervisada.**

La red de Kohonen

La regla de competición lleva los vectores de pesos hacia la entrada actual, como la vencedora y sus vecinas se actualizan en cada paso, todas ellas se mueven hacia la misma posición, aunque las vecinas lo hacen más lentamente a medida que la distancia aumenta. Esto organiza las neuronas de modo que las que están en un entorno comparten la representación del mismo área del espacio de entradas, independientemente de su localización inicial

La red de Kohonen

La selección de los parámetros es muy importante para conseguir una buena representación. Normalmente se distinguen dos fases en el periodo de aprendizaje:

- 1.- **organización topológica** de los pesos, i. e. definición de los entornos (supondremos que precisa de N_0 iteraciones)
- 2.- **convergencia** en la que las neuronas se ajustan con precisión a los detalles de la distribución de las entradas

La red de Kohonen

•1.- La función de vecindad (gaussiana u otras) debe mantenerse grande, cubriendo el espacio de salidas completamente para permitir a neuronas que responden a pesos similares que se acerquen entre sí. La variación se suele tomar: $\sigma(n) = \sigma_0(1 - n / N_0)$
y el coeficiente de aprendizaje debe ser alto (mayor que 0.1) para permitir a la red organizarse: $\Delta\eta(n) = \eta_0(1 - n/(N_0 + K))$

2.- La segunda fase (de convergencia) es usualmente mucho más larga, el coeficiente de aprendizaje se debe conservar pequeño (0.01) y los entornos muy pequeños (sólo la neurona vencedora o sus más próximas)

Limitaciones de la red de Kohonen

El número de neuronas se decide de manera experimental. Aumentando el número, se mejora la aproximación del espacio de salidas, pero también incrementa enormemente el tiempo de entrenamiento.

Para aprender nuevos datos, hay que repetir completamente todo el proceso de aprendizaje, con todos los patrones.

Aplicaciones de la red de Kohonen

El modelo de Kohonen es muy utilizado en la computación neuronal, a pesar de sus limitaciones debido a la duración del proceso de aprendizaje.

Las aplicaciones que mas destacan de los mapas de Kohonen son las relacionadas con el reconocimiento de patrones, o extracción de características, por ejemplo

reconocimiento de voz

reconocimiento de texto manuscrito

codificación de datos

compresión de imágenes

resolución de problemas de optimización

Aplicaciones de la red de Kohonen

Reconocimiento de voz: un prototipo ideado por Kohonen para convertir en tiempo real el habla, en texto escrito, a partir de un vocabulario limitado: entrada 15 neuronas a través de las cuales recibe las componentes de frecuencia que caracterizan a los diferentes fonemas que forman las palabras, estas frecuencias se obtienen mediante un micrófono y un preprocesador analógico y digital, ante fonemas parecidos se active en la red neurona próximas de salida, creando al final un mapa fonotópico. En la fase de funcionamiento, al escuchar una palabra la debe dividir secuencialmente en fonemas que se presentan a la red que los recibe y activa una u otra neurona de salida, recorriendo un camino a través del mapa fonotópico que da lugar a la transcripción fonética de la palabra.

Aplicaciones de la red de Kohonen

Reconocimiento de texto manuscrito: Al igual que en el reconocimiento de voz, vamos a construir un mapa grafotópico, que conviete texto manuscrito, en un texto estándar con caracteres en ASCII.

Resolución de problemas de optimización, como el problema del viajante, de establecer el camino mas corto entre una serie de ciudades sin pasar dos veces por la misma ciudad. La red debe constar de tantas neuronas de salida como ciudades del recorrido, y como entrada necesitará dos neuronas para representar las coordenadas en el plano. Lo que hay que hacer es entrenar la red para que aprenda la situación geográfica de cada una de las ciudades