UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ARAGON

"RED NEURONAL QUE IMPLEMENTA EL ALGORITMO BACK-PROPAGATION PARA ANALISIS DE SENSIBILIDAD EN ESTADISTICA"

T E S I S

QUE PARA OBTENER EL TITULO DE :

INGENIERO EN COMPUTACION

P R E S E N T A:

EL C. JOSE DE JESUS RIVERO GARCIA

MEXICO, D. F. A 11 DE OCTUBRE DE 1994

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

DEDICATORIA Y AGRADECIMIENTOS

Dedico ésta tesis a Mis Padres: Sra. Nicolasa García Soto y Sr. Manuel Alfredo Rivero Garduño y todos sus antepasados, por hacerme, por su amor y por su apoyo, no necesariamente en ése órden.

Gracias a Dios de que estoy vivo y muy sano.

Gracias a mis hermanos y compañeros de entrenamiento Manuel y Alex por ser amigos y estar unidos, a pesar de la distancia.

Gracias a mis hermanas Tere y Alma por ser mis hermanas.

A los niños: Leslie, Daniela, Jesús, Enrique, Daniel y Alejandro

Gracias a mis guías: Profr. Rubén Cuevas, Dr. Jesús Figueroa Nazuno, Carlos Castaneda, Príncipe Siddhartha Gautama y a otros, los cuales el tiempo y la distancia ha borrado su nombre pero no su enseñanza.

Gracias a mis amigos y sus respectivas y apreciables familias (por órden cronológico): Fabián y Francisco Meléndez Soria, Beatriz Robles Ronces, Rubén Hinojosa Rodríguez,M. en C. Alex Aristeo Cázares Carrera, Bruno Unna Ruiz, Boris Sergio Torres Valenzuela.

Gracias a mi primo el Dr. Rafael Jiménez Rivero, por mencionarme en su tesis.

Gracias a mis compañeros de carrera, por serlo.

Gracias a mis sinodales: Ing. José González Bedolla
Ing. Juan Gastaldi
Ing. Miguel Comadurán (con admiración y respeto)
Biól. Adrián Martínez
El director de mi Tesis: Ing. Roberto Blanco Bautista

Gracias a los que inventaron la Gimnasia (la pura VIDA) , la disciplina mas dura, bella, enviciante y la mejor forjadora de hombres sobre la tierra.

Y a la bebé Diana por hacerme feliz junto con su madre la Sra. Adriana Aguilar, mi pareja, y a su apreciable familia.

José de Jesús Rivero García
Marina # 11 Unidad Infonavit Iztacalco
Tel. 650-0531
MEXICO

PREFACIO

¿ POR QUE REDES NEURONALES ARTIFICIALES ?

Después de dos décadas de casi desaparecer, el interés en las redes neuronales artificiales ha crecido rápidamente en los pasados años. Profesionales de campos tan diversos como ingeniería, filosofía, fisiología y psicología están intrigados por el potencial ofrecido por su tecnología y están buscando aplicaciones dentro de sus disciplinas.

Este renacimiento del interés ha sido reavivado por éxitos tanto teóricos como prácticos. De repente, parece posible aplicar la computación a campos previamente restringidos a la inteligencia humana; para hacer máquinas que aprendan y recuerden en maneras que parecen ser una reminiscencia de los procesos mentales humanos; y para dar un nuevo significado al sobre-utilizado término de inteligencia artificial.

En éste trabajo se pretende dar una semblanza de la teoría general que sobre las redes neuronales artificiales existe hasta la fecha, haciendo hincapié en el algoritmo Back-Propagation [35, 40].

Se presenta una introducción en donde se explican las características generales de las redes neuronales. En el capítulo 1 se presentan los fundamentos, haciendo asimismo remembranza histórica, basándose fundamentalmente en los trabajos iniciales en éste campo.

El capítulo 2 pretende presentar un análisis de diferentes tipos de redes, el cual no pretende ser un análisis matemático especializado [26], pero asienta las bases para una investigación posterior para aquella persona interesada en él.

El capítulo 3 se dedica expresamente al algoritmo Back-Propagation, tratando de hacer una análisis claro y suficientemente profundo para que sea entendible por el lector.

El capítulo 4 hace una semblanza de lo que es el análisis de sensibilidad en estadística, tema sobre el cual se basa el software que es base de éste trabajo. Se presentan asimismo algunas redes ya entrenadas con impresión de resultados y gráficas. Asimismo, presenta las últimas mejoras efectuadas al algoritmo Back-Propagation [35] [40],sin llegar al análisis matemático. La última parte presenta algunas de las aplicaciones del algoritmo Back-Propagation [13].

El capítulo 7 presenta las conclusiones del trabajo

El apéndice A presenta una semblanza de la red neuronal biológica, que, sin llegar a ser un tratado sobre el tema, trata de dar la mayor información posible al lector.

El apéndice B presenta algunos algoritmos que a juicio del autor, pueden tener gran futuro.

El apéndice C presenta un análisis de los resultados que se pueden obtener con el algoritmo Back-Propagation. Este análisis no es matemático, sino de desempeño.

El apéndice D son las referencias, para el lector que desée ahondar en el tema.

INTRODUCCION

CARACTERISTICAS DE LAS REDES NEURONALES ARTIFICIALES

Las redes neuronales artificiales están inspiradas en la biología, esto es, están compuestas por elementos que se desempeñan de una manera análoga a las funciones más elementales de la neurona biológica. Estos elementos son entonces organizados de una forma que puede (o no) ser similar a la anatomía del cerebro [38]. A pesar de ésta similitud superficial, las redes neuronales artificiales muestran un sorprendente número de las características del cerebro. Por ejemplo, aprenden de la experiencia [3], generalizan de ejemplos previos hacia nuevos ejemplos, y abstraen características esenciales de entradas conteniendo datos irrelevantes.

Estas habilidades, limitadas sin embargo hoy en día, sugieren que un conocimiento mas profundo de la inteligencia humana podría llevarnos a un campo de aplicaciones más revolucionarias.

APRENDIZAJE

Las redes neuronales artificiales pueden modificar su comportamiento en respuesta a su ambiente [46]. Este factor, más que cualquier otro, es responsable del interés que han recibido. Dado un conjunto de entradas (tal vez sean salidas deseadas), se autoajustan para producir respuestas consistentes.

Una amplia variedad de algoritmos de entrenamiento ha sido desarrollada, cada una con sus propias fuerzas y debilidades. Como se apuntó anteriormente, existen cosas que deben ser respondidas sin importar que cosas pueda hacer una red entrenada, y como debe ser hecho el entrenamiento.

GENERALIZACION

Una vez entrenada, la respuesta de una red puede ser, hasta cierto grado, insensible a variaciones menores en su entrada. Esta habilidad para ver a través del ruido [10] [11] y la distorsión hacia el patrón que se encuentra en medio, es vital para el reconocimiento de patrones en un medio ambiente real [7].

Sobrepasando la forma completamente literal de las computadoras convencionales, produce un sistema que puede interactuar en el imperfecto mundo real en el que vivimos. Es importante notar que las redes neuronales artificiales generalizan automáticamente como resultado de su estructura, no por utilizar inteligencia humana apoyada en programas de computadora hechos para ello.

ABSTRACCION

Algunas redes neuronales artificiales son capaces de abstraer la esencia de un conjunto de entradas.Por ejemplo,una red puede ser entrenada con una secuencia de versiones distorsionadas de la letra A. Después de un entrenamiento adecuado, la aplicación de tal ejemplo distorsionado causará que la red produzca una letra perfectamente formada. En un sentido, ha aprendido a producir algo que nunca había visto antes. Esta habilidad de extraer un ideal de entradas imperfectas promueve interesantes tópicos filosóficos; es reminisencia del concepto de los ideales encontrados en La República de Platón. En cualquier caso, el extraer prototipos idealizados es una habilidad grandemente utilizada en los humanos; parece que ahora podemos compartirla con las redes neuronales artificiales.

APLICABILIDAD

Las redes neuronales artificiales no son una panacea. Son claramente inadecuadas para tareas tales como calcular la nómina. Parece ser, sin embargo, que serán la técnica preferida para una gran clase de tareas de reconocimiento de patrones que las computadoras convencionales efectúan pobremente, si es que lo efectúan.

PERSPECTIVA HISTORICA

Los humanos siempre se han preguntado acerca de sus propios pensamientos. Esta pregunta autoreferencial, el pensamiento en uno mismo, puede ser una característica únicamente del género humano. Las especulaciones sobre la naturaleza del pensamiento pleno, van desde el punto de vista espiritual hasta el anatómico. Con filósofos y teólogos contínuamente oponiéndose a las opiniones de fisiologistas y anatomistas, las preguntas han sido debatidas fuertemente con poca ganancia, ya que el sujeto humano es notoriamente difícil de estudiar.

Aquellos que confían en la introspección y la especulación han llegado a conclusiones que carecen del rigor que es demandado por las ciencias físicas. Los experimentadores han encontrado que el cerebro y el sistema nervioso son difíciles de analizar y tienen una organización muy compleja. En pocas palabras, los poderosos métodos de las pesquisas científicas que han cambiado nuestra visión de la realidad física, se han visto lentos en encontrar aplicación al entendimiento de los humanos mismos.

Los neurobiologistas y neuroanatomistas han hecho progresos sustanciales. El mapeo cuidadoso y laborioso de la estructura y función del sistema nervioso humano, nos ha hecho entender mucho del cableado del cerebro, pero poco de su operación. Así como el conocimiento aumenta, la complejidad también. Cientos de millones de neuronas, cada una conectada a cientos de miles de otras, componen un sistema que ridiculiza nuestros mas ambiciosos sueños de supercomputadoras. Aún así, el cerebro está gradualmente dejándonos saber sus secretos para responder a una de las más ambiciosas preguntas de la humanidad: La naturaleza de su propia conciencia.

El conocimiento mejorado del funcionamiento de las neuronas y el patrón de sus interconexiones ha permitido a los investigadores producir modelos matemáticos para probar sus teorías. Los experimentos pueden ahora ser conducidos en computadoras digitales sin envolver sujetos humanos o animales, resolviéndose así muchos problemas prácticos y éticos. De trabajos anteriores es aparente que ésos modelos no sólo imitan funciones del cerebro, sino que son capaces de desempeñar funciones útiles por sí mismas. De aquí, dos objetivos mutuamente apoyados uno en el otro, referentes en modelado neuronal fueron definidos y se mantienen hasta hoy día: primero, entender el funcionamiento fisiológico y psicológico del sistema neuronal humano; y segundo, producir sistemas computacionales (redes neuronales artificiales) que desempeñen funciones cerebrales. Es el último objetivo en el que nos enfocamos.

Junto con los progresos en neuroanatomía y neurofisiología, los psicólogos fueron desarrollando modelos del aprendizaje humano. Uno de tales modelos, el cual fué probado ser fructífero, fué el de D.O. Hebb [17], quien en 1949 propuso una ley de aprendizaje que llegó a ser la norma para los algoritmos de entrenamiento de las redes neuronales artificiales. Aumentado hoy en día por muchos otros métodos, demostró a los científicos de ésa era cómo una red de neuronas puede exhibir comportamiento de aprendizaje.

En los 1950s y en los 1960s, un grupo de investigadores combinaron éstos conocimientos psicológicos y fisiológicos para producir las primeras redes neuronales artificiales. Inicialmente implementadas como circuitos electrónicos, fueron posteriormente convertidas al más flexible medio de la simulación por computadora, la realización más común hoy día. Exitos anteriores produjeron gran actividad y optimismo. Marvin Minsky, Frank Rosenblatt, Bernard Widrow [39] [32], y otros desarrollaron redes consistentes en una sola capa de neuronas artificiales. Comúnmente llamados Perceptrones, fueron aplicadas a diversos problemas tales como predicción atmosférica, análisis de electrocardiogramas, y visión artificial. Por un tiempo pareció que la llave a la inteligencia se había encontrado; reproducir el cerebro humano fué sólo un asunto de construir una red lo suficientemente grande.

Esta ilusión fué pronto deshechada. Las redes fallaron en resolver problemas superficialmente similares a aquellos en los que habían tenido éxito. Estas fallas inexplicables dieron lugar a un periodo de intenso análisis. Marvin Minsky, aplicando cuidadosas técnicas matemáticas, desarrolló rigorosos teoremas sin tomar en cuenta la operación de la red. Su investigación precedió a la aparición de su libro PERCEPTRONS (Minsky & Papert 1969) [32] , en el cual él y Seymore Papert probaron que las redes de una sola capa entonces en uso eran teóricamente incapaces de resolver muchos problemas simples, incluyendo la función desempeñada por una simple compuerta OR. Minsky no estaba optimista acerca del potencial del progreso:

El perceptrón se ha mostrado ser importante para estudiar a pesar de (¡ y aún a causa de !) sus severas limitaciones. Tiene muchas características que llaman la atención: su linealidad; su intrigante teorema de aprendizaje; su clara simplicidad paradigmática como un tipo de computación en paralelo. No hay razón para suponer que cualquiera de ésas virtudes pertenezcan también a la versión multi capa. Aún así consideramos que es un problema de investigación importante para explicar (o rechazar) Tal vez algún poderoso teorema de convergencia será descubierto, o alguna razón profunda para las fallas producirá un interesante teorema de aprendizaje para la máquina multicapa. (pp 231-32) [32]

La brillantez de Minsky, su rigor, y su prestigio dieron al libro gran credibilidad; sus conclusiones fueron irrefutables. Los investigadores desanimados dejaron el campo para áreas de mayor promesa, las agencias del gobierno de EEUU redirigieron sus fondos, y las redes neuronales artificiales cayeron en una oscuridad por cerca de dos décadas. Aún así, unos pocos científicos dedicados como Teuvo Kohonen, Stephen Grossberg, y James Anderson [15] [22] [23] continuaron sus esfuerzos. Continuamente sin fondos y menospreciados, algunos investigadores tuvieron dificultad en hallar editores; de aquí, la investigación publicada durante la los 1970s y comienzos de los 1980s se encontraba ampliamente distribuída entre una gran variedad de revistas, algunas de las cuales eran casi desconocidas.

Gradualmente fueron emergiendo fundamentos teóricos sobre los cuales las mas poderosas redes multicapa de hoy día son construídas. La apreciación de Minsky [32] fué probada ser sumamente pesimista; las redes neuronales están ahora resolviendo muchos de los problemas que expuso en su libro.

En los pasados años, la teoría ha sido trasladada a la aplicación, y nuevas corporaciones dedicadas a la comercialización de la tecnología han aparecido. Ha habido un explosivo crecimiento en la cantidad de actividad investigatoria.

Con cuatro grandes convenciones en 1987 en el campo de las redes neuronales artificiales, y mas de 500 artículos publicados, la tasa de crecimiento ha sido fenomenal. La lección a ser aprendida de ésta historia se encuentra en la ley de Clark. Propuesta por el escritor y científico Arthur C. Clark, establece que *si un respetable científico dice que una cosa puede ser hecha, el o ella casi siempre tiene razón; si dice que una cosa no puede ser hecha, el o ella casi siempre está en un error.*

La historia de la ciencia es una crónica de errores y verdades parciales. El dogma de hoy viene a ser el sinsentido de mañana. La aceptación sin cuestionamiento de "hechos" cualquiera que sea la fuente, puede lesionar la curiosidad científica. Desde un punto de vista, el excelente trabajo científico de Minsky [32] dió paso a un afortunado hito en el progreso de las redes neuronales artificiales. No existe duda, sin embargo, que el campo ha sido inundado por un optimismo sin razón y por una base teórica inadecuada.

Puede ser que el shock proveniente de los perceptrones permitieron un periodo para la necesaria maduración sobre el campo.

LAS REDES NEURONALES ARTIFICIALES HOY DIA

Han habido muchas demostraciones impresionantes de las capacidades de las redes neuronales artificiales. Una red ha sido entrenada para convertir texto a representaciones fonéticas, las cuales son convertidas a voz por otros medios (Sejnowsky & Rosenberg 1987) [43] [5] ; otra red puede reconocer letra manuscrita (Burr 1987); y otro sistema de compresión de imagen basado en redes neuronales ha sido diseñado (Cottrell, Munro y Zipser 1987).

Todos éstos sistemas utilizan la red **Back-Propagation**, tal vez la más exitosa de los algoritmos actuales. El algoritmo Backpropagation, inventado independientemente en tres esfuerzos de investigación separados (Werbos 1974; Parker 1982 [35] y Rumelhart, Hinton y Williams 1986 [40]) da un método sistemático para entrenar redes multicapa, sobrepasando así las limitaciones presentadas por Minsky. [32]

Como se verá más adelante, el algoritmo Backpropagation tiene sus problemas. Primero, no hay garantía de que la red pueda ser entrenada en un periodo finito de tiempo. Muchos esfuerzos de entrenamiento fallan después de consumir gran cantidad de tiempo de computación. Cuando ésto ocurre, el intento de entrenamiento debe ser repetido, sin garantía de que los resultados serán mejores. Tampoco se puede asegurar que la red se entrenará con la mejor configuración posible.

Muchos otros algoritmos de han sido desarrollados con diferentes ventajas específicas. Debe enfatizarse que ninguna de las redes de hoy día es una panacea; todas ellas tienen problemas en su entrenamiento y sus recuerdos.

Estamos frente a un campo que tiene desempeño demostrado, potencial único, muchas limitaciones y muchas preguntas sin contestar. Es una situación que requiere de optimismo temperado con precaución. Los autores tienden a publicar sus éxitos y les dan a sus fallas poca publicidad, creando así una impresión que puede estar lejos de la realidad. A aquellos buscando capital venturoso para iniciar una empresa deben presentar proyectos convincentes con beneficios firmes y substanciales. Existe, sin embargo, un peligro sustancial en de que las redes neuronales artificiales sean vendidas antes de tiempo, prometiendo buen desempeño sin la capacidad de deliberar si ésto ocurre, nuestra área entera sufrirá una pérdida completa de credibilidad, posiblemente regresándonos a la época de obscurantismo de los 70s.

Se requiere de mucho trabajo sólido para mejorar las redes existentes. Nuevas técnicas deben de ser desarrolladas, los métodos existentes deben ser reforzados, y los fundamentos teóricos ampliados antes de que el campo pueda realizar todo su potencial.

PERSPECTIVAS PARA EL FUTURO

Las redes neuronales artificiales han sido propuestas para tareas que van desde estrategia en el campo de batalla hasta atender a los bebés. Las aplicaciones potenciales son aquellas donde las funciones de la inteligencia humana se han esforzado y la computación convencional se ha mostrado inútil o inadecuada. Esta clase de aplicación es al menos tan grande como aquella servida por la computación convencional, siendo la meta que la neurocomputación esté al menos al mismo nivel que la computación convencional. Esto va a ocurrir sólo si la investigación de los fundamentos obtiene resultados en pronto tiempo, ya que los fundamentos actuales son inadecuados para soportar tales proyectos.

REDES NEURONALES ARTIFICIALES Y SISTEMAS EXPERTOS

El campo de la inteligencia artificial ha sido dominado en años recientes por las disciplinas de manipulación simbólica y lógica. Por ejemplo, los sistemas expertos han sido ampliamente aclamados y han logrado muchos éxitos notables, así como muchas fallas. Algunos dicen que las redes neuronales artificiales reemplazarán a la inteligencia artificial actual, pero existen muchos indicadores de que ambas coexistirán y serán combinadas en sistemas en los cuales cada técnica desempeñará la tarea para la cuál fué diseñado.

Este punto de vista es apoyado por la forma en que los humanos operan en el mundo. Las actividades que requieren rápidas respuestas están gobernadas por el reconocimiento de patrones. Ya que las acciones son producidas rápidamente y sin esfuerzo consciente, éste modo de operación es esencial para las respuestas rápidas que se requieren para sobrevivir en un medio ambiente hostil. Considérense las consecuencias si nuestros ancestros tuvieran que razonar la respuesta correcta para responder al ataque de un carnívoro.

Uno puede imaginarse una red neuronal que imita esta división del trabajo. Una red neuronal artificial produciría una respuesta adecuada a su ambiente bajo muchas cirscunstancias. Ya que tales redes pueden indicar el nivel de confianza asociado con cada decisión, podría "saber lo que no sabía", y podría referir ése caso a un sistema experto para su resolución. La decisión hecha a éste nivel podría ser concreta y lógica, pero podría requerir el deducir hechos adicionales antes de encontrar una solución.

La combinación de los dos sistemas podría ser mas robusta si ambos sistemas actuaran solos, y podría seguir el altamente exitoso modelo de la evolución biológica.

CONSIDERACIONES DE CONFIANZA

Antes de que las redes neuronales artificiales puedan ser aplicadas donde están en juego vidas humanas o valores importantes, preguntas acerca de su confianza deben ser repondidas.Así como los humanos a cuya estructura cerebral imitan, las redes neuronales artificiales tienen un grado de impredictibilidad. A menos que toda posible entrada sea probada, no existe forma de que estemos seguros de la salida precisa. En una gran red tales pruebas exhaustivas son imprácticas y la estadística del desempeño debe ser usada. Bajo ciertas circunstancias ésto es intolerable.

Por ejemplo, ¿cuál es un porcentaje de error aceptable para una red controlando un sistema de defensa espacial?. Mucha gente podría decir que cualquier error es intolerable; podría resultar en una inimaginable muerte y destrucción. Esta actitud no cambia por el factor humano en la misma situación; también se podría cometer un error.

Este problema se basa en que se espera que las computadoras son absolutamente perfectas, sin error. Y ya que las redes neuronales cometen algunas veces errores aún cuando están funcionando perfectamente, muchos piensan que ésto se traduce en que son muy propensas a fallar; una característica que hemos encontrado inaceptable en nuestras máquinas.

Una dificultad relacionada es la inhabilidad de las redes neuronales artificiales tradicionales para explicar "como" resuelvan sus problemas. Las representaciones internas que resultan del entrenamiento son a menudo tan complejas y resistentes al análisis excepto en los casos triviales.

Esto está cercanamente relacionado a nuestra falta de habilidad para explicar como reconocemos a una persona sin importar diferencias de distancia, ángulo, iluminación, y el paso de los años. Un sistema experto puede ir hacia atrás a través de su propio proceso de razonamiento de tal modo que un humano puede revisarlo. La incorporación de ésta habilidad en las redes neuronales artificiales ha sido reportada (Gallant 1988) y su desarrollo puede tener un importante efecto sobre la aceptabilidad de éstos sistemas.

CAPITULO 1. FUNDAMENTOS

Las redes neuronales artificiales han sido desarrolladas en una amplia variedad de configuraciones. A pesar de ésta diversidad, los paradigmas de las redes tienen mucho en común. Las redes neuronales artificiales están biológicamente inspiradas; ésto es, los investigadores están usualmente pensando acerca de la organización del cerebro cuando están considerando configuraciones y algoritmos de redes. En éste punto la correspondencia entre redes podría terminar.

El conocimiento acerca de la operación general del cerebro está tan limitada que existe poca guía para aquellos que quisieran emularlo. De aquí, los diseñadores de redes deben ir más allá del conocimiento biológico actual. En muchos casos, es necesario descartar plausibilidades biológicas; el cerebro se convierte en una metáfora; las redes son producidas de tal forma que son orgánicamente imposibles o requieren un conjunto de hechos asumidos muy improbable acerca de la anatomía del cerebro y su funcionamiento.

A pesar de que muy a menudo no existe relación con la biología, las redes neuronales artificiales continúan evocándonos comparaciones con el cerebro. Sus funciones son reminiscencias de la cognición humana, por lo tanto, es difícil el no establecer la comparación. Desafortunadamente, tales comparaciones no son benignas; crean expectaciones falsas que inevitablemente terminan en desilusión. Los fondos para investigación basados en falsas esperanzas pueden evaporarse en la luz de la realidad como pasó en los 1960s, y éste campo promisorio podría otra vez eclipsarse si no se toman medidas.A pesar de la anterior precaución, sigue siendo conveniente el entender algo acerca del sistema nervioso de los mamíferos. Es una entidad que desempeña exitosamente las tareas para las cuales fué diseñado, las cuales sólo son aspiraciones para nuestras actuales redes neuronales. (Ver apéndice A)

Los modelos de redes neuronales están especificados por la topología de la red, características del nodo, y las reglas de aprendizaje o entrenamiento. Estas reglas especifican un conjunto inicial de pesos e indica cómo éstos deben ser adaptados para mejorar el desempeño. Tanto los procedimientos de diseño como las reglas de entrenamiento son el tópico de mucha investigación actual. Los beneficios potenciales de las redes neuronales se extienden más allá de las altas tasas de computación proveídas por el paralelismo masivo. [4]

Las redes neuronales típicamente proveen más alto grado de tolerancia a las fallas que las computadoras secuenciales de Von Neumann porque hay muchos nodos de procesamiento, cada uno con conexiones primarias locales. El daño a unos pocos nodos o ligas en una u otra manera no necesariamente disminuyen el desempeño general significativamente.

Muchos de los algoritmos de redes neuronales también adaptan pesos de conexión para mejorar el desempeño basados en resultados actuales. La adaptación o aprendizaje es uno de los puntos focales de la investigación de redes neuronales actual. La habilidad de adaptarse y continuar aprendiendo es esencial en áreas como reconocimiento del habla [5, 8, 12 donde los datos de entrenamiento son limitados y nuevos hablantes, nuevas palabras, nuevos dialectos, nuevas frases, y nuevos ambientes son contínuamente encontrados. La adaptación también provée un grado de robustez para compensar variaciones pequeñas en características de los elementos procesados.

Las técnicas de estadísticas tradicionales no son adaptativas, pero típicamente procesan todos los datos de entrenamiento simultáneamente antes de ser usados con nuevos datos. Los clasificadores de redes neuronales también son no-paramétricos y hacen mas frágiles suposiciones concernientes a las formas de apoyar distribuciones que los clasificadores estadísticos tradicionales. Estos pueden probar ser mas robustos cuando las distribuciones son generadas por procesos no-lineales y son fuertemente no Gaussianos.

Diseñar redes neuronales artificiales para resolver problemas y estudiar redes biológicas reales puede también cambiar la forma en que pensamos acerca de problemas y mostrar el camino hacia nuevos mejoramientos de algoritmos.

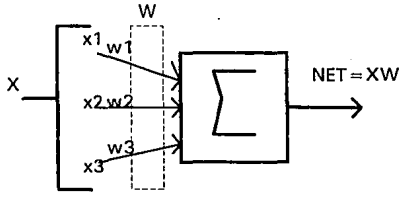


FIGURA 1-1 Neurona Artificial

LA NEURONA ARTIFICIAL

La neurona artificial fué diseñada [30] para imitar las características de primer orden de la neurona biológica. En esencia, un conjunto de entradas son aplicadas, cada una representando la salida de otra neurona. Cada entrada es multiplicada por un correspondiente peso, análogo a una fuerza sináptica, y todas las entradas pesadas son entonces sumadas para determinar el nivel de activación de la neurona. La figura 1-1 muestra un modelo que implementa ésa idea. A pesar de la diversidad de los paradigmas sobre redes, casi todas están basadas en ésta consideración. Aquí, un conjunto de entradas llamadas x1, x2,...,xn es aplicada a la neurona artificial. Estas entradas, colectivamente referidas como el vector **X**, corresponden a las señales dentro de las sinápsis de una neurona biológica.

Cada peso corresponde a la fuerza de una sola conexión sináptica biológica. (El conjunto de pesos es llamado colectivamente como el vector **W**). El bloque sumatorio, correspondiente al cuerpo biológico celular, suma todas las entradas pesadas algebráicamente, produciendo una salida que llamamos **NET**. Esta puede ser escrita compactamente en notación vectorial como sigue:

$$\mathbf{NET=XW}$$

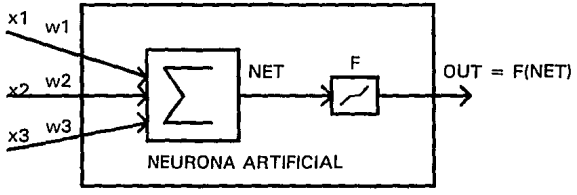


FIGURA 1-2 Neurona artificial con funcion de activación

La señal NET es usualmente procesada más por una función de activación F para producir la señal de salida de la neurona, OUT. Esto puede ser una simple función lineal,

$$OUT = K(NET)$$

donde K es una constante, una función de umbral,

$$OUT = 1 \text{ si } NET > T$$

OUT = 0 de otra forma

donde *T* es una constante de valor de umbral, o una función que simula más exactamente la característica e transferencia de la neurona biológica y permite funciones de red más generales.

En la figura 1-2, el bloque etiquetado como F acepta la salida NET y produce la señal OUT. Si el bloque de procesamiento F comprende el rango de NET, de tal modo que OUT nunca exceda los límites inferiores sin importar el valor de NET, F es llamada una función compresora. La función compresora es comúnmente escogida para ser la función logística del sigmoide (S-sombreada mediana) como se muestra en la figura 1-2. Esta función es expresada matemáticamente como $F(x) = 1/(1+e^{-X})$. Entonces

$$OUT = 1/(1+e^{-NET})$$

Por similitud a los sistemas análogos, podemos pensar en la función de activación como definir la ganancia no lineal de la neurona artificial. Esta ganancia es calculada encontrando el cociente del cambio en OUT a un pequeño cambio en NET. Entonces, la ganancia es la inclinación de la curva a un nivel de excitación específico. Varía de un valor bajo a grandes excitaciones negativas (la curva es casi horizontal), a un gran valor en excitación cero, y decae como la excitación se va haciendo mas grande y positiva.

Grossberg [15] (1973) encontró que ésta característica no lineal de ganancia resuelve el dilema de saturación de ruido que él propuso; ésto es, ¿cómo puede la misma red manejar tanto señales grandes como pequeñas? Las señales de entrada pequeñas requieren alta ganancia a través de la red si van a ser capaces de producir una salida utilizable; sin embargo, un gran número de estados en cascada de gran ganancia saturan la salida con el ruido amplificado (variaciones aleatorias), ésto está presente en cualquier red realizable. También, grandes señales de entrada saturarán estados de gran ganancia, eliminando otra vez cualquier salida utilizable.

La región central de alta ganancia de la función logística resuelve el problema de procesar señales pequeñas, mientras que sus regiones de ganancia decreciente en los extremos positivos y negativos son apropiadas para grandes excitaciones. De ésta manera, una neurona se desempeña con ganancia apropiada para una gran amplitud de niveles de entrada.

Otra función comúnmente usada es la función tangente hiperbólica fig. 1-3. Es similar en forma a la función sigmoidal y es comúnmente usada por biólogos como un modelo matemático de activación de nervios celulares. Usada como una función de activación de una red neuronal es expresada como sigue:

$$OUT = \tanh(x)$$

Como la función logística, la tangente hiperbólica tiene forma de S simétrica al origen, resultando que OUT tiene el valor de 0 cuando NET es 0 . A diferencia de la función logística, la función tangente hiperbólica tiene un valor bipolar para OUT, una característica que ha sido vista que es benéfica para ciertas redes.

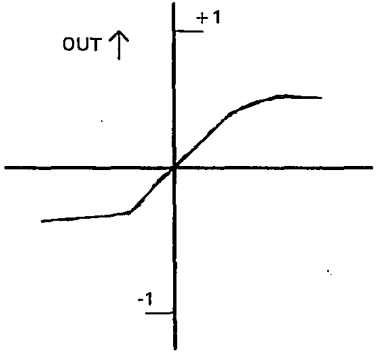


FIGURA 1 - 3 Función Tangente Hiperbólica

Este modelo simple de la neurona artificial ignora muchas de las características de su contraparte biológica. Por ejemplo, no toma en cuenta los retardos de tiempo que afectan la dinámica del sistema; las entradas producen salida inmediata. Más importante, no incluye los efectos de la sincronía o la modulación de frecuencia de la neurona biológica, característica que para algunos investigadores es crucial.

Sin tomar en cuenta éstas limitaciones, las redes formadas de éstas neuronas exhiben atributos que son fuertemente reminiscentes de los sistemas biológicos. Tal vez lo suficiente de la naturaleza esencial de la neurona biológica ha sido capturado para producir respuestas como el sistema biológico, o tal vez la similaridad es coincidencia; sólo el tiempo y la investigación lo dirán.

REDES NEURONALES ARTIFICIALES DE UNA SOLA CAPA

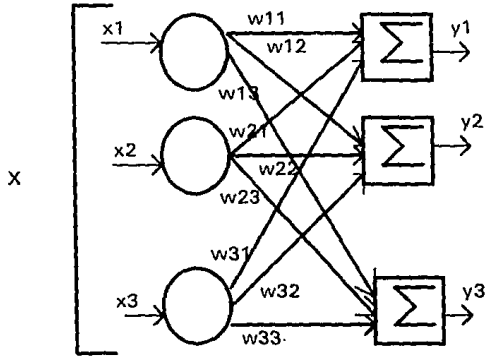


FIGURA 1-4 Red Neuronal de una sola capa

Aunque una sola neurona puede desempeñar ciertas simples funciones de detección de patrones, el poder de la computación neuronal viene de conectar neuronas a redes. La red más simple es un grupo de neuronas arregladas en una capa como se muestra en la figura 1-3. Nótese que los nodos circulares en la izquierda sirven sólo para distribuir las entradas; no desempeñan computación y de aquí que no se considerarán constituyentes de alguna capa.

Por ésta razón, se muestran como círculos para distinguirlas de las neuronas de computación, las cuales se muestran como cuadrados. El conjunto de entradas X tiene cada uno de sus elementos de entrada conectados cada neurona artificial a través de una anchura de separación. Las antiguas redes neuronales artificiales no eran más complejas que ésto. De cada neurona simplemente se obtenía una suma pesada de las entradas de la red. Las redes artificiales actuales y las biológicas pueden tener muchas de las conexiones borradas, pero es mostrada total conectividad por razones de generalidad.También, pueden haber conexiones entre las salidas y las entradas de neuronas en una capa.

Es conveniente considerar que los pesos son elementos de una matriz W. Las dimensiones de la matriz son m renglones por n columnas, donde m es el número de entradas y n el número de neuronas. Por ejemplo, el peso que conecta la tercera entrada a la segunda neurona podría ser De éste modo, se puede ver que calculando el conjunto de N salidas neuronales NET para una capa es una simple multiplicación de matrices. Así, N = XW, donde N y X son vectores de renglón.

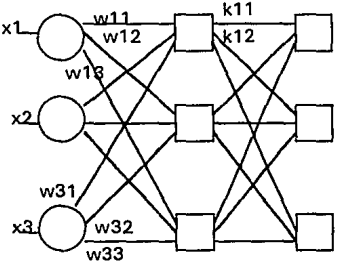


FIGURA 1-5 Red neuronal de dos capas

REDES NEURONALES ARTIFICIALES MULTICAPA

Más aún, las redes más complejas generalmente ofrecen más grandes capacidades de computación. Aunque las redes hayan sido construídas en toda configuración imaginable, el arreglar las neuronas en capas imita las estructura de ciertas porciones del cerebro. Estas redes multicapa han probado tener capacidades más grandes que las de una sola capa, y en años recientes, se han desarrollado algoritmos para entrenarlas.

Las redes multicapa pueden estar formadas por un simple grupo de capas conectadas en cascada; la salida de una capa provée la entrada de la subsecuente. La figura 1-4 muestra tal red, totalmente conectada.

LA FUNCION DE ACTIVACION NO LINEAL

Las redes multicapas no dan incremento de poder computacional sobre las redes unicapa, a menos que exista una función no lineal de activación entre capas. Calcular la salida de una capa consiste en multiplicar el vector de entrada por la primera matriz de pesos, y entonces (si no existe la función no lineal de activación),multiplicar el vector resultante por la segunda matriz de pesos. Esto puede ser expresado como

(XW1)W2

Ya que la multiplicación de matrices es asociativa, los términos pueden ser reagrupados como sigue:

X(W1W2)

Esto demuestra que una red lineal de dos capas es exactamente equivalente a una sola capa teniendo una matriz de pesos igual al producto de las dos matrices de pesos. De aquí, cualquier red multicapa puede ser reemplazada por una red unicapa equivalente. Hacemos incapié en que las redes unicapa son severamente limitadas en sus capacidades de computación; así, las funciones no lineales de activación son vitales para la expansión de la capacidad de la red más allá de una red unicapa.

REDES RECURRENTES

Las redes consideradas hasta éste punto no tienen conexiones de realimentación, ésto es, las conexiones a través de los pesos de las salidas de una capa a las entradas de la misma o de una capa previa. Esta clase especial de redes llamadas no recurrentes o de alimentación en adelanto, es de interés considerable y ampliamente utilizada.

Redes más generales que contienen conexiones de realimentación se les llama recurrentes. Las redes no recurrentes no tienen memoria, su salida es únicamente determinada por las entradas actuales y por los valores de los pesos. En algunas configuraciones, las redes recurrentes recirculan salidas previas hacia las entradas; de aquí, sus salidas son determinadas tanto por las entrada actual como por su salida previa. Por ésta razón, las redes recurrentes pueden exhibir propiedades muy similares a memorias recientes en humanos y el estado de las salidas de la red depende en parte de su entrada previa.

ENTRENAMIENTO DE LAS REDES NEURONALES ARTIFICIALES

De todas las características interesantes de las redes neuronales artificiales, ninguna captura la imaginación como su habilidad para aprender. Su entrenamiento muestra tantos paralelos al desarrollo intelectual de seres humanos que puede parecer que hemos logrado un entendimiento fundamental de éste proceso. La euforia debe ser temperada con precaución; el aprendizaje en las redes neuronales artificiales es limitado, y muchos diferentes problemas quedan por resolver antes de poder determinar si es que vamos en la pista correcta. Aún así, se han hecho demostraciones impresionantes, tales como la de Sejnowski con su NetTalk, y muchas otras aplicaciones prácticas.

EL OBJETIVO DEL ENTRENAMIENTO

Una red es entrenada de tal forma que la aplicación de un conjunto de entradas produce la salida deseada (o al menos consistente). Cada una de tales conjuntos de entrada (o de salida) es referido como un vector. El entrenamiento es llevado a cabo por medio de aplicar secuencialmente vectores de entrada, mientras que se ajustan los pesos de la red de acuerdo a un procedimiento predeterminado. Durante el entrenamiento, los pesos de la red convergen gradualmente hacia valores de tal forma que cada vector de entrada produce el vector de salida deseado.

ENTRENAMIENTO SUPERVISADO

Los algoritmos de entrenamiento están categorizados en supervisado y no supervisado. El entrenamiento supervisado requiere de aparear cada vectorde entrada con un vector objetivo representando la salida deseada; juntos se denominan par de entrenamiento. Comúnmente una red es entrenada sobre un número de tales pares de entrenamiento. Un vector de entrada es aplicado, la salida de la red es calculada y comparada con el vector objetivo correspondiente, y la diferencia (error) es realimentada a través de la red y los pesos son cambiados de acuerdo a un algoritmo que tiende a minimizar el error. Los vectores del conjunto de entrenamiento son aplicados secuencialmente, y los errores son calculados y los pesos ajustados para cada vector, hasta que el error para el conjunto entero de entrenamiento sea puesto a un aceptable nivel bajo.

ENTRENAMIENTO NO SUPERVISADO

Aún con muchos éxitos en aplicaciones, el entrenamiento supervisado ha sido criticado por ser biológicamente implausible; es difícil de concebir un mecanismo de entrenamiento en el cerebro que compare las salidas actuales y las deseadas, alimentando las correcciones procesadas hacia la red. ¿Si éste fuera el mecanismo del cerebro, de dónde vendrían los patrones de salida deseados? ¿Cómo podría el cerebro de un infante llevar a cabo la autoorganización que se ha demostrado ocurre en el desarrollo temprano?.

El entrenamiento no supervisado es un modelo mucho más plausible de aprendizaje en un sistema biológico. Desarrollado por Kohonen (1984) [22] [23] y muchos otros, no requiere de un vector objetivo para las salidas, y de aquí, no hay comparaciones predeterminadas para respuestas ideales. El conjunto de entrenamiento consiste solamente de vectores de entrada. El algoritmo de entrenamiento modifica los pesos de la red para producir vectores de salida que sean consistentes; ésto es tanto la aplicacion de un vector de entrenamiento o un vector que es suficientemente similar a él producirá el mismo patrón de salidas.

El proceso de entrenamiento, así, extrae las propiedades estadísticas del conjunto de entrenamiento y agrupa vectores similares en clases. Aplicando un vector de una clase dada a la entrada producirá un específico vector de salida, pero no hay manera de determinar antes del entrenamiento cuál específico patrón de salida será el producido por una clase dada de vector de entrada. De aquí, las salidas de tal red deben generalmente ser transfomadas a una forma comprensiva subsecuente al proceso de entrenamiento. Esto no representa un problema serio. Es comúnmente un simple asunto de identificar las relaciones de entrada-salida establecidas por la red.

ALGORITMOS DE ENTRENAMIENTO

Muchos de los algoritmos de entrenamiento de hoy día han evolucionado de los conceptos de D.O. Hebb (1961) [17] . El propuso un modelo para aprendizaje no supervisado en el cual la fuerza sináptica (peso) es amplificado si tanto la fuente como el destino (neuronas) son activadas. En ésta forma, los caminos más usados en la red eran reforzados, y el fenómeno del hábito y aprendizaje a través de la representación era explicado.

Una red neuronal artificial utilizando aprendizaje Hebbiano incrementará sus pesos de red de acuerdo al producto de los niveles de excitación de las neuronas fuente y destino. En símbolos:

$$w_{ij(n+1)} = w_{ij(n)} + bOUT_i\,OUT_j$$

donde

$w_{ij(n)}$ = el valor de un peso de una neurona i hacia una neurona j antes del ajuste

$w_{ij(n+1)}$ = el valor del peso desde la neurona i a la neurona j después del ajuste

b = el coeficiente de aprendizaje

OUTi = la salida de la neurona i y la entrada de la neurona j

OUTj = la salida de la neurona j

Se han construído redes que utilizan el aprendizaje Hebbiano, sin embargo, se han desarrollado algoritmos más efectivos en los últimos 20 años. En particular el trabajo de Rosenblatt (1962) [39] , Widrow (1959) [47], Widrow y Hoff (1960) [47] [48] , y muchos otros desarrollaron algoritmos de entrenamiento supervisado, produciendo redes que aprendieron un amplio rango de patrones de entrada, y más altos niveles de aprendizaje, que lo que podrían haber hecho con un simple algoritmo Hebbiano. Existe una tremenda variedad de algoritmos de entrenamiento en uso hoy día; sería necesario más espacio que éste para darnos un tratamiento completo de éste tópico.

PERCEPTRONES

LOS PERCEPTRONES Y LOS ANTIGUOS DIAS DE LAS REDES NEURONALES ARTIFICIALES

La ciencia de las redes neuronales artificiales hizo su primera aparición significante en los años 1940. Los investigadores decidieron duplicar las funciones del cerebro humano desarrollando modelos de hardware simple (y después software) de las neuronas biológicas y su sistema de interconexión. Así como los neurofisiologistas gradualmente ganaron un entendimiento mejorado del sistema neuronal humano, ésos primeros intentos fueron vistos como grandes aproximaciones. Fueron logrados resultados impresionantes que lograron más investigación resultando en redes de más grande sofisticación.

McCulloch y Pitts (1943) [30] publicaron el primer estudio sistemático de redes neuronales artificiales. En trabajos posteriores, (Pitts y McCulloch 1947) exploraron paradigmas de redes para reconocimiento de patrones además de rotación y traslación de objetos . Mucho de su trabajo envolvía el modelo de neurona simple mostrado en la figura 1-2. La unidad multiplica cada entrada x por un peso w, y suma las entradas pesadas. Si ésta suma es mayor que un umbral predeterminado, la salida es 1, de otra manera es 0. Estos sistemas (y sus muchas variaciones) han sido llamados colectivamente Perceptrones. En general, consisten de una sola capa de neuronas artificiales conectadas por pesos a un conjunto de entradas (ver figura 1-4), aunque mas complicadas redes compartan el mismo nombre.

En los 60s los perceptrones crearon un gran interés y optimismo. Rosenblatt (1962) [39] demostró un teorema notable acerca del aprendizaje del perceptron (explicado abajo). Widrow (Widrow 1961, 1963; Widrow y Angell 1962; Widrow y Hoff 1960) [47] [48] efectuó una serie de demostraciones convincentes de sistemas similares a los perceptrones, y los investigadores en todo el mundo exploraron entusiasmadamente el potencial de éstos sistemas. La euforia inicial fué reemplazada por desilusión ya que los perceptrones fallaban en ciertas tareas simples de aprendizaje. Minsky (Minsky y Papert 1969) [32] analizaron éste problema con gran rigor y probaron que existen severas restricciones en lo que un perceptron unicapa puede representar, y de aquí, de lo que puede aprender.

Ya que no habia técnicas conocidas en aquel tiempo para entrenar redes multicapa, los investigadores cambiaron a áreas mas promisorias, y la investigación sobre redes neuronales artificiales cayó en un eclipse. El descubrimiento reciente de métodos de entrenamiento para redes multicapa

ha sido, más que otro factor, responsable del resurgimiento del interés y del esfuerzo de investigación.El trabajo de Minsky pudo retardar el ardor del los entusiastas del perceptrón, pero proveyó de un periodo de consolidación y desarrollo necesario para la teoría de apoyo. Es importante notar que los análisis de Minsky no han sido refutados; continúa siendo un trabajo importante y debe ser estudiado si los errores de los 60s no quieren volver a ser repetidos. Sin tomar en cuenta las limitaciones de los perceptrones, éstos han sido extensivamente estudiados [37] (si no utilizados muy ampliamente). Su teoría es el fundamento de muchas otras formas de redes neuronales artificiales y demuestran importantes principios. Por éstas razones, son un punto lógico de inicio para un estudio de redes neuronales artificiales.

REPRESENTACION DEL PERCEPTRON

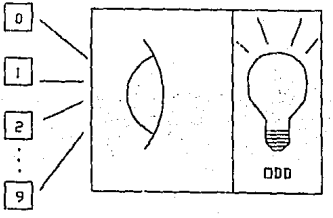


FIGURA 1-6 Sistema de Reconocimiento de Imágenes

La prueba del teorema de aprendizaje del Perceptrón (Rosenblatt 1962) demostró que un perceptrón puede aprender cualquier cosa que puedan representarse. Es importante distinguir entre representación y aprendizaje. La representación se refiere a la habilidad del perceptrón (u otra red neuronal) para simular una función específica. El aprendizaje requiere de la existencia de un procedimiento sistemático para ajustar los pesos de la red para producir tal función.Para ilustrar el problema de representación, supongamos que tenemos un conjunto de cartas conteniendo los números del 0 al 9. Supongamos también que tenemos una máquina hipotética que es capaz de distinguir las cartas con números nones de las cartas con números pares, iluminando un indicador en su panel cuando se muestra una carta con número impar (Ver figura 1-5. ¿Puede tal máquina ser representada por un perceptrón?. Esto es, ¿puede un perceptrón ser construído y sus pesos ajustados (sin importar como sea hecho ésto) para que tenga la misma capacidad discriminatoria ? . Si es así, decimos que el perceptrón puede representar la máquina deseada. Veremos que el perceptrón unicapa está seriamente limitado en sus capacidades de habilidad representativa; hay muchas máquinas simples que el perceptrón no puede representar sin importar cómo se ajusten los pesos.

EL PROBLEMA DE LA O-EXCLUSIVA

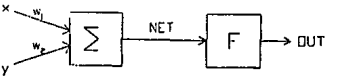


FIGURA 1-7 Sistema de una sola Neurona

Uno de los resultados mas desalentantes muestra que una red unicapa no puede simular una función o-exclusiva. Esta función acepta dos entradas que pueden ser cero o uno. Produce una salida de uno si sólo una de ambas entradas es uno (pero no ambas). El problema puede ser mostrado por considerar un sistema de una sola capa, con una sola neurona con dos entradas como se muestra en la figura 1-6. Llamando una entrada como x y la otra como y, todas sus posibles combinaciones comprenden cuatro puntos en el plano xy mostrado en la figura 1-7. Por ejemplo, los puntos x=0 y=0 están etiquetados como punto A_0 en la figura. La tabla T-1 muestra las relaciones deseadas entre las entradas y las salidas, donde ésas combinaciones de entrada que pueden producir 0 están etiquetadas como A_0 y A_1; aquellas que producen un 1 están etiquetadas como B_0 y B_1.

En la red de la figura 1-6, la función F es un simple umbral que produce un 0 para OUT cuando la red está abajo de 0.5 y 1 cuando es igual o superior. La neurona entonces desempeña el siguiente cálculo:

$$NET = xw_1 + yw_2$$

Ninguna combinación de valores para dos pesos w_1 y w_2 producirá la relación de entrada de la tabla T-1. Para entender ésta limitación considera que NET contendrá una constante en el umbral de valor 0.5. La ecuación E-1 describe la red en éste caso. Esta ecuación es lineal en x y y; ésto es, todos los valores para x y y que satisfacen ésta ecuación caerán sobre alguna línea recta en el plano xy.

$$xw_1 + yw_2 = 0.5 \qquad \text{E-1}$$

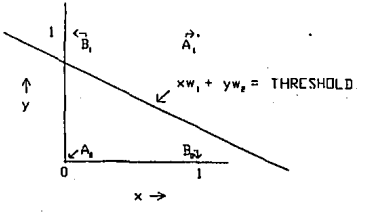


FIGURA 1-8 Problema de la O-Exclusiva como puntos en el plano X-Y

Cualquier valor de entrada para x y y en ésta línea producirá el valor de umbral de 0.5 para NET. Los valores de entrada en una lado de la línea producirán NET más grande que el umbral, de aquí OUT = 1; los valores en el otro lado producirán NET menor que el valor de umbral así que OUT = 0. Cambiando los valores de w_1 y de w_2, y el umbral cambiará la inalineación en la posición de la línea. Para que la red produzca la función o-exclusiva de la tabla T-1, es necesario poner la línea de tal modo que todas las As estén en un lado y todas las Bs en el otro. No es posible dibujar tal línea. Esto significa que no importa que valores sean asignados a los pesos y al umbral, ésta red no puede producir la relación de entrada requerida para representar la función o-exclusiva.

Analizando el problema desde una perspectiva un poco diferente, consideramos a NET como una superficie flotando encima del plano xy. Cada punto sobre ésta superficie está directamente arriba de un punto correspondiente en el plano xy a una distancia igual al valor de NET en tal punto. Se puede demostrar que el inalineamiento de ésta superficie NET es constante sobre el plano xy. Todos los puntos que producen un valor de NET igual al valor de umbral proyectarán un nivel constante en el plano NET (Ver figura 1-8). Claramente, todos los puntos en un lado de la línea de umbral proyectarán valores de NET más altos que el umbral y los puntos en el otro lado deben resultar en valores más bajos de NET. Entonces, la línea de umbral subdivide el plano xy en dos regiones. Todos los puntos en un lado del umbral producen un uno para OUT; todos los puntos del otro lado producen un cero.

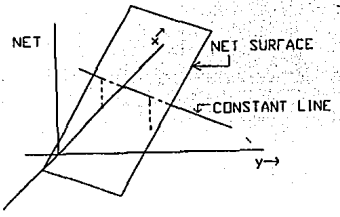


FIGURA 1-9 Plano NET del perceptrón

Hemos visto que no hay manera de dibujar una línea recta subdividiendo el plano xy de tal manera que se represente la función o-exclusiva. Desafortunadamente, éste no es un ejemplo aislado; existe una gran clase de funciones que no pueden ser representadas por una red unicapa. A éstas funciones se les llama linealmente inseparables, y su conjunto define fronteras en las capacidades de las redes unicapa.

La separabilidad lineal limita las redes unicapa a problemas de clasificación en los cuales conjuntos de puntos (correspondientes a valores de entrada) puedan ser separados geométricamente. Para el caso de dos entradas, el separador es una línea recta. Para tres entradas, la separación es desempeñada por un plano cortando el espacio tridimensional resultante de la salida. Para cuatro o más entradas, la visualización es imposible, y debemos generar mentalmente un espacio n-dimensional dividido por un hiperplano, un objeto geométrico que subdivide el espacio en cuatro o más dimensiones.A causa de que la separabilidad lineal limita la habilidad de un perceptrón para hacerse representaciones, es importante saber si una función dada es separable linealmente.Desafortunadamente, no existe una simple forma de hacer ésta determinación si el número de variables es grande.

Una neurona con n entradas binarias puede tener 2^n diferentes patrones de entrada, consistentes en unos y ceros. Ya que cada patrón de entrada puede producir dos diferentes entradas binarias, uno y cero, existen 2 a la 2^n diferentes funciones de n variables.

SOBREPASANDO LA LIMITACION DE LA SEPARABILIDAD LINEAL

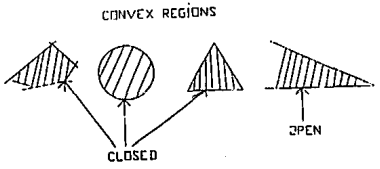


FIGURA 1-10 Regiones convexas, cerradas y abiertas

En lo último de los 60s el problema de la separabilidad lineal fué bien entendido. También se supo que esta limitación representacional de redes unicapa podría resolverse al añadir más capas. Por ejemplo, las redes de dos capas pueden ser formadas por conectar en cascada dos redes unicapa. Estas pueden desempeñar clasificaciones mas generales, separar aquellos puntos que están contenidos en regiones convexas abiertas o cerradas. Una región convexa es una en la cual cualesquiera dos puntos en la región pueden ser unidos por una línea recta que no rebase los límites de la región. Una región cerrada es una en la cual todos los puntos están contenidos dentro de una frontera (por ejemplo,un círculo). Una region abierta tiene algunos puntos que están afuera de cualquier frontera definida. (por ejemplo, la region entre dos líneas paralelas). Para ejemplo de regiones convexas abiertas y cerradas ver la figura 1-9.

Para entender la limitación de complejidad, considérese una red simple de dos capas, con dos entradas hacia dos neuronas en la primera capa, ambas alimentando una simple neurona en la capa 2 (Ver figura 1-10). Asumamos que el umbral de la neurona de salida está puesto a 0.75 y sus pesos están ambos puestos a 0.5. En éste caso, una salida de "1" es requerida de ambas neuronas de la capa 1 para exceder el umbral y producir un "1" a la salida. Entonces, la neurona de salida desarrolla una función "y" lógica. En la figura 2-8 se asume que cada neurona en la capa 1 subdivide el plano xy, una produciendo una salida de "1" para entradas abajo de la línea superior y otra produciendo una salida de uno para entradas arriba de la línea inferior. La figura 1-10 muestra el resultado de la doble subdivisión, donde OUT de la neurona de la capa 2 es "1" sólo sobre una región ashurada.

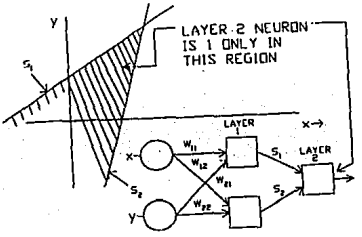


FIGURA 1-11 Región convexa de decisión producida por un perceptrón de dos capas

Similarmente, tres neuronas pueden ser usadas en la capa de entrada, subdividiendo aún más el plano, creando una región en forma de triángulo. Por medio de incluir suficientes neuronas en la capa de entrada, un polígono convexo de cualquier forma deseada puede ser formado. Ya que éstos son formados por la operación AND de regiones definidas por líneas rectas, todos éstos polígonos son convexos, de aquí sólo regiones convexas pueden ser encerradas. Los puntos que no incluyan una region convexa, no pueden ser separados de todos los otros puntos en el plano por una red de dos capas.

La neurona de la capa 2 no está limitada por la función AND; puede producir muchas otras funciones, si los pesos y umbral son escogidos acertadamente. Por ejemplo, puede ser arreglado que ambas neuronas en la capa 1 teniendo su propio nivel OUT en "1" causen que el nivel OUT de la capa sea puesto a "1", formando así un OR lógico. Existen 16 funciones binarias para dos variables. Si los pesos y el umbral son escogidos apropiadamente, una neurona de dos entradas puede simular 14 de ellas (todas menos la "o" ni la "nor" exclusivas).

Las entradas no necesitan ser binarias. Un vector de entradas contínuas puede representar un punto en cualquier lugar del plano xy. En éste caso, entramos en el campo de la habilidad de la red para subdividir el plano en regiones contínuas más que separar conjuntos de puntos discretos.Para todas las funciones, sin embargo, la separabilidad lineal muestra que la salida de una neurona de la capa 2 es sólo una sobre una porción del plano xy encerrada por un polígono convexo. Entonces, para separar regiones P y Q, todos los puntos en P deben estar dentro de un polígono convexo que no contenga puntos de Q (o vice versa).

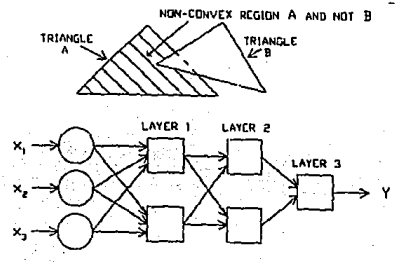


FIGURA 1-12 Región cóncava de decisión formada por la intersección de dos regiones convexas

Una red de tres capas es aún más general; su capacidad de clasificación está limitada sólo por el número de neuronas artificiales y pesos. No existen restricciones de convexitud; la neurona de la capa 3 ahora recibe como entrada un grupo de polígonos convexos, y la combinación lógica de ellos no necesita ser convexa. La figura 1-11 ilustra un caso en el cual dos triángulos, A y B son combinados por la función "A and not B", definiendo así una region no convexa. Como las neuronas y los pesos son añadidos, el número de lados de los polígonos puede incrementarse sin límite. Esto hace posible encerrar una region de cualquier forma a cualquier grado deseado de exactitud.Además, no todas las regiones de salida de la capa 2 necesitan intersectarse.

Es posible, en cambio, encerrar múltiples regiones convexas y no convexas produciendo una salida de "1" cuando el vector de entrada es cualquiera da ellos.

Sin tomar en cuenta la temprana recognición del poder de las redes multicapa, por muchos años no hubo algoritmo de entrenamiento para ajustar sus pesos. Pero por ahora es suficiente entender el problema y darse cuenta que la investigación ha producido soluciones.

EFICIENCIA DE ALMACENAMIENTO

Existen serias dudas acerca de la eficiencia de almacenamiento del perceptrón (y de otras redes neuronales artificiales) relativo a la memoria convencional de computadora y de los métodos de recuperación de los datos. Por ejemplo, podría ser posible almacenar todos los patrones de entrada en una memoria de computadora conjuntamente con los bits de clasificación. La computadora podría entonces buscar el patrón deseado y responder con su clasificación. Varias estrategias bien conocidas podrían ser empleadas para acelerar la búsqueda. Si una correspondencia exacta no fuera encontrada, el criterio del vecino más próximo podría ser usado para regresar como resultado la correspondencia más próxima.

El número de bits requerido para almacenar la misma información en los pesos del perceptrón puede ser substancialmente más pequeño que con el método de la memoria convencional de computadora, si la naturaleza de los patrones permite una representación compacta. Sin embargo, Minsky (Minsky y Papert 1969) han mostrado ejemplos patológicos en los cuales el número de bits para representar los pesos crece más rápido que exponencialmente con el tamaño del problema.

En éstos casos, los requerimientos de memoria se expanden rápidamente hacia niveles imprácticos como el tamaño del problema crece. Si, como él mismo conjetura, ésta situación no es la excepción, los perceptrones podrían contínuamente ser limitados a pequeños problemas. ¿Qué tan comunes son ésos imprácticos conjuntos de patrones? Esto continúa siendo una pregunta abierta que se aplica a todas las redes neuronales. Encontrar una respuesta es una de las áreas críiticas para la investigación de las redes neuronales artificiales.

EL APRENDIZAJE DEL PERCEPTRON

La habilidad de aprendizaje de las redes neuronales artificiales es una propiedad muy intrigante. Como los sistemas biológicos que ellas modelan, éstas redes se modifican a sí mismas como resultado de la experiencia para producir un patrón de comportamiento más deseable. Utilizando el criterio de separabilidad lineal es posible conocer dónde o dónde no una red unicapa puede representar una función deseada. Aún si la respuesta es Sí, no nos sirve de mucho si no tenemos modo de encontrar los valores necesitados para pesos y umbrales. Si la red va a ser un valor práctico, necesitamos un método sistemático (un algoritmo) para computar los valores. Rosenblatt (1962) no dió ésto en el algoritmo de entrenamiento del perceptrón, junto con su prueba de que un perceptrón puede ser entrenado para cualquier función que pueda representar. El aprendizaje puede ser supervisado o sin supervisión. El aprendizaje supervisado requiere un maestro externo que evalúa el comportamiento del sistema y dirija las modificaciones subsecuentes. El aprendizaje sin supervisión, no requiere maestro; la red se autoorganiza para producir los cambios deseados. El aprendizaje del perceptrón es del tipo supervisado.

El algoritmo de entrenamiento del perceptrón puede ser implementado en una computadora digital u otro hardware electrónico y la red viene a ser, en un sentido, autoajustable. Por ésta razón, el acto de ajustar los pesos es comúnmente llamado "entrenamiento", y se dice que la red "aprende". La prueba de Rosenblatt fué un gran progreso y le dió un gran ímpetu a la investigación del campo. Hoy en día, en una forma u otra, los elementos del algortimo de entrenamiento del perceptrón son encontrados en muchos paradigmas de redes modernas.

CAPITULO 2. ALGORITMOS

Los modelos de redes neuronales están especificados por la topología de la red, características del nodo, y las reglas de aprendizaje o entrenamiento.

Estas reglas especifican un conjunto inicial de pesos e indica cómo éstos deben ser adaptados para mejorar el desempeño. Tanto los procedimientos de diseño como las reglas de entrenamiento son el tópico de mucha investigación actual. Los beneficios potenciales de las redes neuronales se extienden más allá de las altas tasas de computación proveídas por el paralelismo masivo [4]. Las redes neuronales típicamente proveen más alto grado de tolerancia a las fallas que las computadoras secuenciales de Von Neumann porque hay muchos nodos de procesamiento, cada uno con conexiones primarias locales. El daño a unos pocos nodos o ligas en una u otra manera no necesariamente disminuyen el desempeño general significativamente.

Muchos de los algoritmos de redes neuronales también adaptan pesos de conexión para mejorar el desempeño basados en resultados actuales. La adaptación o aprendizaje es uno de los puntos focales de la investigación de redes neuronales actual. La habilidad de adaptarse y continuar aprendiendo es esencial en áreas como reconocimiento del habla donde los datos de entrenamiento son limitados y nuevos hablantes, nuevas palabras, nuevos dialectos, nuevas frases, y nuevos ambientes son contínuamente encontrados. La adaptación también provée un grado de robustez para compensar variaciones pequeñas en características de los elementos procesados. Las técnicas de estadísticas tradicionales no son adaptativas, pero típicamente procesan todos los datos de entrenamiento simultáneamente antes de ser usados con nuevos datos.

Los clasificadores de redes neuronales también son no-paramétricos y hacen mas frágiles suposiciones concernientes a las formas de apoyar distribuciones que los clasificadores estadísticos tradicionales. Estos pueden probar ser mas robustos cuando las distribuciones son generadas por procesos no-lineales y son fuertemente no Gaussianos. Diseñar redes neuronales artificiales para resolver problemas y estudiar redes biológicas reales puede también cambiar la forma en que pensamos acerca de problemas y mostrar el camino hacia nuevos mejoramientos de algoritmos.

REDES NEURONALES Y CLASIFICADORES TRADICIONALES

El diagrama de bloque de clasificadores tradicionales y de red neuronal son presentados en la figura 2-1. Ambos tipos de clasificadores determinan cual de las M clases es más representativa de un patrón de entrada estático desconocido conteniendo N elementos de entrada. En un reconocedor de habla, las entradas pueden ser los valores de salida de envolvente de un banco de filtros de un analizador espectral muestreado en un instante de tiempo y las clases pueden representar diferentes vocales. En un reconocedor de imágenes [14], las entradas pueden representar la escala de grises de cada pixel de una imagen y las clases pueden representar diferentes objetos.

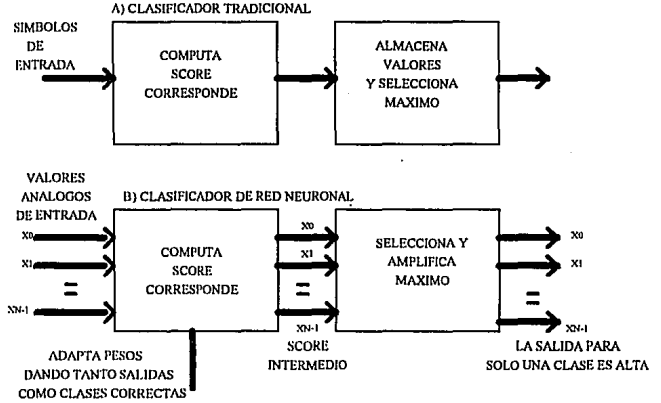


FIGURA 2-1 Diagrama de bloques de un clasificador tradicional y uno de Red Neuronal. Las entradas y salidas del tradicional son pasadas serialmente y las computaciones internas son desempeñadas secuencialmente. Además, los datos de entrenamiento son estimados de los datos de entrenamiento y mantenidos constantes. Las entradas y salidas del clasificador de red neuronales están en paralelo, y las computaciones internas son desempeñadas en paralelo. Los parámetros internos de los pesos son típicamente entrenados o adaptados durante el uso, usando los valores de salida y etiquetas especificando la clase correcta.

El clasificador tradicional en la parte alta de la figura 2-1 contiene dos niveles. El primero computa el número de puntos que corresponden para cada clase y el segundo selecciona la clase con el máximo número de puntos. Las entradas al primer nivel son símbolos representando valores de los N elementos de entrada. Estos símbolos son introducidos secuencialmente y decodificados de su forma simbólica externa a una representación interna útil para desempeñar operaciones aritméticas y simbólicas. Un algoritmo computa el número de puntos para cada una de las M clases lo que indica que tan cercanamente la entrada corresponde al patrón de ejemplo para cada clase. Este patrón de ejemplo es aquel patrón que es más representativo de cada clase.

En muchas situaciones un modelo probabilístico es usado para modelar la generación de patrones de entrada de ejemplos y el número de puntos correspondientes representa la probabilidad de que el patrón de entrada fué generado por cada uno de los M posibles ejemplos. Los parámetros o distribuciones pueden entonces ser estimados usando datos de entrenamiento como se muestra en la figura 2-1. Las distribuciones Gaussianas multivariable son frecuentemente usadas llevándonos a algoritmos relativamente simples para computar número de puntos correspondientes. Los números de puntos correspondientes son codificados en representaciones simbólicas y pasados secuencialmente al segundo nivel del el clasificador. Aquí son decodificadas y la clase con el máximo número de puntos es seleccionada. Un símbolo representando ésa clase es entonces enviada a la salida para completar la tarea de clasificación.

Un clasificador de red neuronal adaptativo es mostrado en la parte de abajo de la figura 2-1. Aquí los valores de entrada son alimentados en paralelo al primer nivel mediante N conexiones de entrada. Cada conexión trae un valor analógico el cual puede tomar uno de dos niveles para entradas binarias o puede variar en un gran rango para entradas de valor contínuo. El primer nivel computa los números de puntos que corresponden y manda a la salida éstos números de puntos en paralelo al siguiente nivel sobre M líneas de salida análogas. Aquí el máximo de éstos valores es seleccionado y amplificado. El segundo nivel tiene una salida para cada una de las M clases. Después que la clasificación es completada, sólo ésa salida correspondiente a la clase más similar estará en nivel alto o "1"; las otras salidas estarán en bajo.

Note que en éste diseño, las salidas existen para todas las clases y ésta multiplicidad de salidas debe ser preservada en posteriores niveles de proceso, tanto tiempo como las clases sean consideradas distintas. En el sistema más simple de clasificación éstas líneas de salida deben ir directamente a luces con etiquetas que especifican identificadores de clases. En casos más complicados, éstas salidas podrían ir a más niveles donde las entradas de otras modalidades o dependencias temporales son tomadas en consideración. Si es dada la clase correcta, entonces ésta información y las salidas del clasificador pueden ser realimentadas al primer nivel del clasificador para adaptar pesos usando un algoritmo de aprendizaje como se muestra en la figura 2-1. La adaptación hará una respuesta correcta mas probable para seguidamente introducir patrones similares al patrón actual.

Las entradas en paralelo requeridas por los clasificadores de red neuronal sugieren que las implementaciones de hardware de tiempo real deben incluír pre-procesadores de propósito especial. Una estrategia para diseñar tales procesadores es construir pre-procesadores psicológicamente basados imitando sistemas sensoriales humanos. Un pre-procesador para clasificación de imágenes, modelado imitando la retina y diseñado usando circuitería análoga VLSI es descrito en [31]. Bancos de filtros de pre-procesadores para recognición del habla que son crudas analogías de la cóclea han sido también construidos[34, 29]. Algoritmos de pre-procesador mas recientes, basados psicológicamente para recognición del habla, intentan dar información similar a aquella disponible en el nervio auditivo. [11, 44, 27, 5]. Muchos de ésos algoritmos incluyen bancos de filtros para análisis espectral, control automático de la ganancia, y procesamiento el cual usa información síncrona o asíncrona en adición a información proveniente de señales de salida filtradas.

Los clasificadores de la figura 2-1 pueden desempeñar tres diferentes tareas. Primero, como de describe arriba, pueden identificar qué clase representa mejor un patrón de entrada, donde se asume que las entradas han sido corrompidas por ruido o algún otro proceso.

Este es un problema clásico de teoría de decisiones. Segundo, los clasificadores pueden ser usados como memorias de contenido direccionable o memorias asociativas [2], donde la clase ejemplar es deseada y el patrón de entrada es usado para determinar cual ejemplar producir. Una memoria de contenido direccionable es útil cuando sólo parte de un patrón de entrada es disponible y el patrón completo es requerido, como en recuperación bibliográfica de referencias de revistas de información parcial. Esto normalmente requiere la adición de un tercer nivel en la figura 2-1. para regenerar el ejemplar de la clase mas probable. Un nivel adicional es innecesario para algunas redes neuronales tales como la red Hopfield [45] la cual está diseñada específicamente como memoria de contenido direccionable.

Una tercera tarea que éstos clasificadores pueden desempeñar es cuantizar vectores [28] o arracimar las N entradas en M racimos. Los cuantizadores de vectores son usados en sistemas de transimsión de habla e imagen para reducir el número de bits necesarios para transmitir datos análogos. En aplicaciones de reconocimiento del habla e imagen son usados para comprimir el conjunto de datos que deben ser procesados sin pérdida importante de información. En ambas aplicaciones, el número de racimos puede ser pre-especificados o pueden ser permitidos de crecer hasta un número determinado por el número de nodos disponible en el primer nivel.

Las redes entrenadas con supervisión tal como la red Hopfield [18] y los perceptrones [39] son usadas como memorias asociativas y como clasificadores. Estas redes son proveídas con información lateral o etiquetas que especifican la clase correcta para nuevos patrones de entrada durante el entrenamiento. Muchos de los clasificadores estadísticos tradicionales, tales como clasificadores Gaussianos [7], son entrenados con supervisión usando datos de entrenamiento etiquetados. Las redes entrenadas sin supervisión, tal como las redes formativas de mapas de características de Kohonen [22],son usadas como cuantizadores vectoriales o para formar racimos.

Ninguna información concerniente a la clase correcta es proveída a éstas redes durante el entrenamiento. Los algoritmos clásicos de K-media [7] y líder [16], ambos de arracimamiento, son entrenados sin supervisión. Una diferencia más entre redes, es cuando es soportado el entrenamiento adaptativo. Aunque casi todas las redes pueden ser entrenadas adaptativamente, la red Hopfield y la red Hamming son usadas generalmente con pesos fijos.

La red Hamming [25] es una implementación de red neuronal del clasificador óptimo para patrones binarios corrompidos por ruido aleatorio [10]. Puede ser mostrada que la estructura del perceptrón desempeña aquellos cálculos requeridos por un clasificador Gaussiano [7] cuando los pesos y umbrales son seleccionados apropiadamente. En otros casos los algoritmos de red neuronal son diferentes de los algoritmos clásicos. Por ejemplo, los perceptrones entrenados con el procedimiento de convergencia del perceptrón [39] se comportan diferente que los clasificadores Gaussianos. También la red de Kohonen no efectúa el algoritmo de entrenamiento de K-media. En vez de ello, cada nuevo patrón es presentado sólo una vez y los pesos son modificados después de cada presentación. La red Kohonen, sin embargo, forma un número preespecificado de racimos como en el algoritmo K-media, donde la K se refiere al número de racimos formados.

LA RED HOPFIELD

La red Hopfield [18] [19] [20] (mostrada en la figura 2-2) y otras redes son usadas normalmente con entradas binarias. Estas redes son mas apropiadas cuando representaciones exactas binarias son posibles como con imágenes en blanco y negro donde los elementos de entrada son valores pixels, o con texto ASCII donde los valores de entrada pueden representar bits en el formato ASCII de 8 bits de cada caracter.

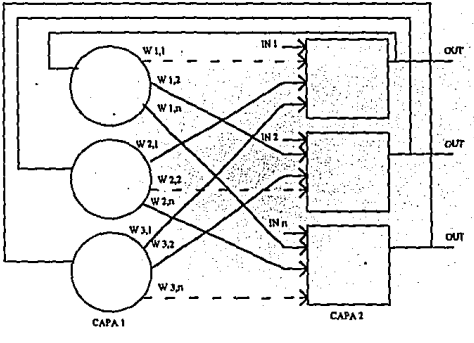


FIGURA 2-2 Red Recurrente de una sola capa

Estas redes son menos apropiadas cuando los valores de entrada son realmente contínuos, porque un problema fundamental de representación debe ser direccionado para convertir cantidades análogas a valores binarios.

Hopfield hizo renacer el interés en las redes neuronales por su extensivo trabajo sobre diferentes versiones de la red Hopfield [18, 19, 20]. Esta red puede ser usada como una memoria asociativa o para resolver problemas de optimización. Una versión de la red original [18] la cual puede ser usada como memoria direccionable es descrita aquí. Esta red, mostrada en la figura 2-2, tiene N nodos conteniendo fuertes no-linealidades limitantes y entradas binarias y salidas tomando los valores de +1 y -1.

La salida de cada nodo es realimentada a todos los otros nodos a través de pesos . la operación de ésta red es descrita en la caja 1. Primero, los pesos son puestos usando los valores recibidos dados por los patrones de ejemplo de todas las clases. Entonces, un patrón desconocido es impuesto a la red en el tiempo 0 mediante forzar a la red a igualar el patrón desconocido. En seguida de ésta inicialización, la red itera en tiempo discreto usando la fórmula dada.

CAJA 1: Algoritmo de Hopfield

1:Asignar pesos	$t_{ij}=\sum_{s=0}^{M-1} x^s_i x^s_j$	$i \neq j$	o $i=1$ si $0 \le i, j \le M-1$
2: inicializa con modelos desconocidos de entrada	$\mu_i(0) = x_i$	$0 \le j \le N-1$	Donde $\mu_i(t)$ es la salida del nodo i en el tiempo t
3: Itera hasta que converga	$\mu_i(t+1)= fn\left[\sum_{j=0}^{N-1} t_{ij}\mu_i(t)\right]$	$0 \le j \le M-1$	
4: Ir al paso 2	El proceso es repetido hasta que los nodos de salida no cambien		

Se considera que la red converge cuando las salidas no cambian en sucesivas iteraciones. El patrón especificado por los nodos de salida después de la convergencia es la salida de la red.

Hopfield [18] y otros [4] han probado que ésta red converge cuando los pesos son simétricos ($t_{ij} = t_{ji}$) y los nodos de salida son actualizados asíncronamente usando las ecuaciones de la caja 1. Hopfield también demostró que la red converge cuando no-linealidades graduadas similares a la no-linealidad sigmoidal es usada. Cuando la red Hopfield es usada como memoria asociativa, la red de salida, después de la convergencia es usada directamente como la memoria completamente restaurada.

Cuando la red Hopfield es usada como clasificador, la salida después de la convergencia debe ser comparada a los M ejemplares para determinara si ésta corresponde a algún ejemplar exactamente. Si lo hace, la salida es aquella clase cuyo ejemplar correspondió al patrón de salida. Por el contrario, entonces ocurre un resultado de "no correspondencia".

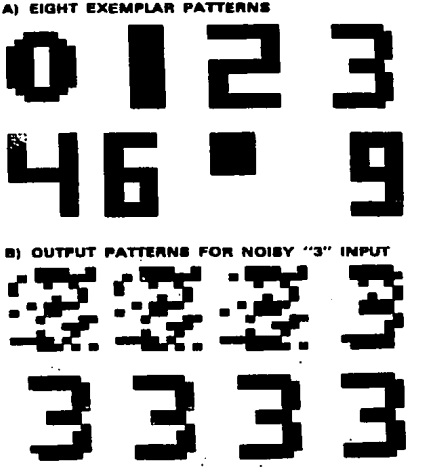


FIGURA 2-3 Ejemplo del comportamiento de una red Hopfield cuando se utiliza como una memoria de contenido direccionable. Una red de 120 nodos fué entrenada utilizando los ocho ejemplares mostrados en (A). El patrón para el dígito "3" fué corrompido mediante aleatoriamente complementar cada bit con una probabilidad de 0.25, y aplicarlo a la red en el tiempo cero. Las salidas en el tiempo cero y después de siete iteraciones son mostradas.

El comportamiento de la red Hopfield es ilustrado en la figura 2-3. Una red Hopfield con 120 nodos y 14,400 pesos fué entrenada para recordar los ocho patrones ejemplares mostrados en la parte de arriba de la figura 2-3. Estos patrones en blanco y negro parecidos a dígitos, contienen 120 pixels cada uno y fueron hechos a mano para proveer de buen desempeño. Los elementos de entrada a la red toman el valor de +1 para pixels negros y de -1 para pixels blancos.

En el ejemplo mostrado, el patrón para el dígito "3" fué corrompido mediante aleatoriamente cambiar el valor de cada pixel independientemente de +1 a -1 y viceversa con una probabilidad de 0.25. Este patrón fué entonces aplicado a la red en el tiempo 0.Los patrones producidos en la salida de la red en las iteraciones de la 0 a la 7 se presentan en la parte de abajo de la figura 2-3. El patrón corrompido de entrada es presentada inalterado en la iteración 0. Según vaya iterando la red, la salida se va pareciendo más y más al patrón ejemplar correcto hasta que en la sexta iteración la red ha convergido al patrón para el dígito 3.

La red Hopfield tiene dos grandes limitaciones cuando es usada como una memoria de contenido direccionable. Primero, el número de patrones que pueden ser almacenados y recordados exactamente está severamente limitado. Si son almacenados demasiados patrones, a un nuevo patrón falso diferente de los patrones ejemplares. Tal patrón falso producirá una salida "no corresponde" cuando la red es usada como clasificador.

Hopfield [18] mostró que ésto no ocurre frecuentemente cuando patrones ejemplares son generados aleatoriamente y el número de clases (M) es menor que 0.15 veces el número de elementos de entrada o nodos en la red (N). El número de clases es así mantenido abajo de 0.15N. Por ejemplo, una red Hopfield para sólo 10 clases podría requerir más de 70 nodos y más de, por decir, 5,000 conexiones de pesos.

Una segunda limitación de la red Hopfield es que un patrón ejemplar es considerado inestable si es aplicado en el tiempo 0 y la red converge a algún otro ejemplar. Este problema puede ser eliminado y el desempeño puede ser mejorado por un cierto número de procedimientos de ortogonalización [14, 46].

LA RED HAMMING

La red Hopfield es probada frecuentemente en problemas donde las entradas son generadas por seleccionar un ejemplar y cambiar los valores de los bits aleatoria e independientemente con una probabilidad dada [18, 12, 46]. Este es un problema clásico en teoría de comunicaciones, que ocurre cuando señales binarias de longitud fija son mandadas a través de un canal simétrico binario sin memoria. El clasificador de mínimo error óptimo en éste caso calcula la distancia Hamming al ejemplar para cada clase y selecciona aquella clase con la mínima distancia Hamming [10]. La distancia Hamming es el número de bits en la entrada que no corresponden a los correspondientes bits ejemplares. Una red, la cual será llamada una red Hamming, implementa éste algoritmo usando componentes de redes neuronales, como se ilustra en la figura 2-4.

La operación de la red Hamming es descrita en la caja 2. Los pesos y los umbrales son primero colocados en la subred inicial de tal modo que los números de puntos correspondientes generados por las salidas de los nodos intermedios de la figura 2-4 son iguales a N menos las distancias Hamming a los patrones ejemplares. Estos números de puntos correspondientes tendrán un rango desde 0 hasta el número de elementos en la entrada (N) y son más altos para aquellos números correspondientes a clases con ejemplares que mejor corresponden a la entrada.

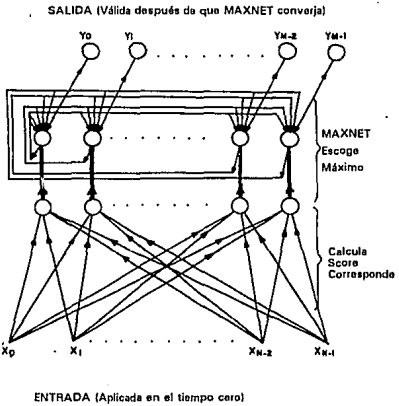


FIGURA 2-4 Red Hamming

CAJA 2: Algoritmo de Hamming

1: Asignar Pesos de Conexión	capa 1: $w_{ij}=x_i^j/2$	$\theta_j=N/2$ $0\le j\le M-1$	w_{ij}, t_{kl}=peso θ=umbral
	capa w: $t_{ij}=\begin{cases}1, & k=l\\ -\varepsilon, & k\neq l\end{cases}$	$\varepsilon < 1/M$ $0 \le k$ $l \le M-1$	x_i^j=elemento i ejemplar j
2: Inicializa con modelo desconocido	$\mu_i(0)=f_t\left[\sum_{i=0}^{N-1}\omega_{ij}x_j-\theta_j\right]$	$0\le j\le M-1$	$\mu_i(t)$=salida del nodo x_i=entrada f_t=funcion
3: Itere hasta que converga	$\mu_j(t+1)=$ $f_t\left[\mu_j(t)-\varepsilon\sum_{k\neq j}\mu_k(t)\right]$	$0\le j$ $k\le M-1$	
4: Ir al paso 2			

Los umbrales y pesos en la subred MAXNET son fijos. Todos los umbrales han sido puestos a cero y los pesos de cada nodo a sí mismos son 1. Los pesos entre nodos son inhibitorios con un valor de $-\varepsilon$ donde ε 1/M.

Después de que los pesos y umbrales han sido puestos, un patrón de entrada binario con N elementos es presentado en la parte de abajo de la red Hamming. Debe ser presentado lo suficiente como para permitir que la salidas con números de puntos correspondientes a la primera subred se pongan en su lugar e inicializar los valores de salida de la MAXNET.

La entrada es entonces quitada y la MAXNET itera hasta que la salida de un sólo nodo es positiva. La clasificación es entonces completa y la clase seleccionada es aquella que corresponda al nodo con una salida positiva. El comportamiento de la red Hamming es ilustrada en la figura 2-5. Los cuatro dibujos en ésta figura muestran que las salidas de los nodos en un MAXNET con 100 nodos en iteraciones 0, 3, 6 y 9. Estas simulaciones fueron obtenidas usando patrones ejemplares aleatoriamente seleccionados con 1000 elementos cada uno.

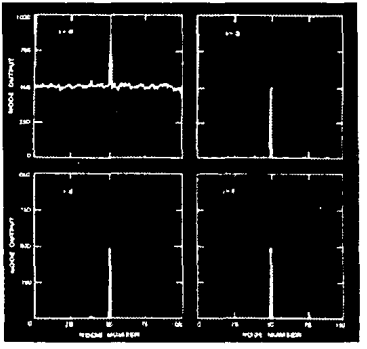


FIGURA 2-5 Comportamiento de la Red Hamming

El ejemplar para la clase 50 fué presentado en el tiempo 0 y quitado. El número de puntos correspondientes en el tiempo 0 es máximo (1000) para el nodo 50 y tiene un valor aletatorio cerca de 500 para otros nodos. Después de sólo 3 iteraciones, las salidas de todos los nodos excepto el nodo 50 han sido grandemente reducidas, y después de 9 iteraciones sólo la salida para el nodo 50 es no-cero. Las simulaciones con diferentes probabilidades de invertir bits en patrones de entrada y con diferentes números de clases y elementos en los patrones de entrada han demostrado que la MAXNET típicamente converge en menos de 10 iteraciones en ésta aplicación [25]. Además, puede probarse que la MAXNET siempre convergerá y encontrará el nodo con el máximo valor cuando $\varepsilon < 1/M$ [25].

La red Hamming tiene un número de ventajas obvias sobre la red Hopfield. Implementa el clasificador de mínimo error cuando los errores de bits son aleatorios e independientes, y así el desempeño de la red Hopfield puede ser peor o equivalente a la red Hamming en tales situaciones. Comparaciones entre las dos redes en problemas tales como reconocimiento de caracteres, reconocimiento de patrones aleatorios, y recuperación bibliográfica han demostrado ésta diferencia en desempeño [25]. La red Hamming también requiere muchas menores conexiones que la red Hopfield. Por ejemplo, con 100 entradas y 10 clases, la red Hamming requiere sólo 1,100 conexiones mientras que la red Hopfield requiere casi 10,000. Aún más, la diferencia en número de conexiones requeridas se incrementa con el número de entradas, porque el número de conexiones en la red Hopfield crece como el cuadrado del número de entradas, mientras que el número de conexiones en la red Hamming crece linealmente.

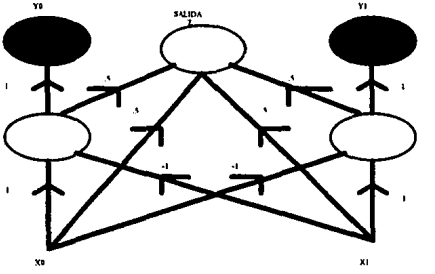


FIGURA 2-6 Subred comparadora con lógica de umbral que selecciona el máximo de dos entradas análogas. La salida Z es el valor máximo, y las salidas Y0 y Y1, indican cual entrada fué máxima. Los umbrales internos en los nodos de lógica de umbral (nodos abiertos) y los nodos de límites duros (nodos cerrados) son cero. Los pesos son como se muestran.

La red Hamming también puede ser modificada para ser un clasificador de mínimo error cuando los errores son generados por invertir elementos de entrada de +1 a -1 y de -1 a +1 asimétricamente con diferentes probabilidades. [25] y cuando los valores de específicos elementos de entrada son desconocidos [2].

Finalmente, la red Hamming no sufre de falsos patrones de salida que pueden producir un resultado de "no corresponde"

SELECCIONANDO O AUMENTANDO LA ENTRADA MAXIMA

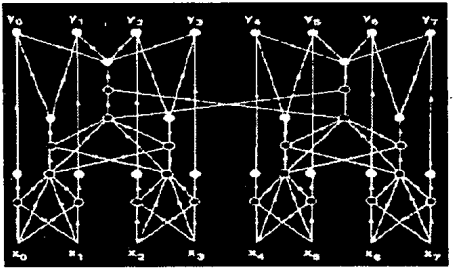


FIGURA 2-7 Una red Feed-Forward que determina cuál de 8 entradas es máxima utilizando un árbol binario y redes comparadoras como las de la fig. 2-6. Los umbrales en los nodos de salida son 2.5.

La necesidad de seleccionar o aumentar la entrada con un máximo valor ocurre frecuentemente en problemas de clasificación. Muchas diferentes redes neuronales pueden desempeñar ésta operación. La red MAXNET descrita arriba usa inhibición lateral pesada similar a aquella usada en otros diseños de redes, donde un máximo fué deseado. [20, 22, 9]. Esos diseños imitan el uso pesado de la inhibición lateral evidente en las redes biológicas naturales del cerebro humano [21]. Otras técnicas para obtener un máximo son también posibles [25]. Una de éstas es ilustrada en la figura 2-5.

Esta figura muestra una subred comparadora la cuál es descrita en [29]. Esta usa nodos de lógica de umbral para obtener el máximo de dos entradas y entonces alimenta esta valor máximo en adelanto. Esta red es útil cuando el valor máximo debe ser pasado inalterado a la salida. Una red que usa éstas subredes para obtener el máximo de 2 entradas es presentada en la figura 2-6.

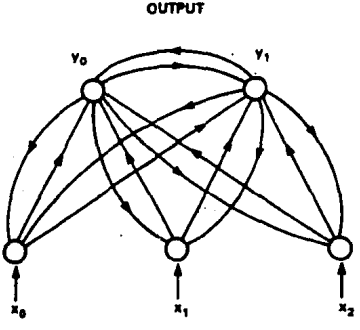
En algunas ocasiones es requerido un máximo y los números de puntos correspondientes deben entonces ser comparados a un umbral. Esto puede ser hecho usando un arreglo de nodos con límites duros con umbrales internos puestos en los valores de umbral deseados. Las salidas de éstos nodos deben ser -1 a menos que las entradas excedan los valores de umbral.

Alternativamente, los umbrales pueden ser puestos adaptivamente usando una entrada inhibitoria común alimentando a todos los nodos. Este umbral puede ser aumentado o disminuído hasta que la salida de un sólo nodo sea positiva.

EL CLASIFICADOR GROSSBERG-CARPENTER

Carpenter y Grossberg [3], en el desarrollo de su teoría de resonancia adaptativa, han diseñado una red la cual forma racimos y es entrenada sin supervisión. Esta red implementa un algoritmo de arracimamiento que es muy similar al algoritmo líder simple secuencial de arracimamiento descrito en [16]. El algoritmo líder selecciona la primera entrada como el ejemplar para el primer racimo. La próxima entrada es comparada al primer ejemplar arracimado. Este "sigue al líder" y es arracimado con el primero si la distancia al primero es menor que un umbral. De otra manera éste es el ejemplar para un nuevo racimo. Este proceso es repetido para todas las siguientes entradas. Así, el número de racimos crece con el tiempo y depende tanto del umbral como de la distancia métrica usa para comparar entradas a ejemplares de racimos.

FIGURA 2-8 Clasificador Carpenter-Grossberg



Los mayores componentes de una red clasificadora Carpenter-Grossberg con tres entradas y dos nodos de salida está presentada en la figura 2-8 La estructura de ésta red es similar a la de la red Hamming. Los puntos que corresponden son computados usando conexiones de alimentación en adelanto y los valores máximos son amplificados usando inhibición lateral formado parte de los nodos de salida. Esta red difiere de la red Hamming en que las conexiones de alimentación en retraso son hechas de los nodos de salida a los nodos de entrada. Los mecanismos también son dados para apagar los nodos de salida con un valor máximo, y comparar los ejemplares a la entrada para la prueba de umbral requerida por el algoritmo líder. Esta red está completamente descrita usando ecuaciones diferenciales no-lineales, incluyendo la realimentación extensiva, y se ha demostrado que es estable [3].

En operación típica, se puede demostrar que las ecuaciones diferenciales implementan el algoritmo de arracimamiento presentado en la caja 3. El algoritmo de la caja 3 asume que el "aprendizaje rápido" es usado como en las simulaciones presentadas en [3] y así los elementos en ambas entradas y ejemplares almacenados toman sólo los valores de 0 y 1. La red es inicializada por poner todos los ejemplares representados por pesos de conexión a cero. Además, un umbral de correspondencia llamado "de vigilancia" cuyos rangos están entre 0.0 y 1.0 debe ser puesto. Este umbral determina que tan cerca un nuevo patrón de entrada se asemeja a un ejemplar almacenado para ser considerado similar.

CAJA 3: Algoritmo Carpenter Grossberg o de Arracimamiento (Clustering)

1: Inicialización	$t_{ij}(0)=1$ $b_{ij}(0)=1/(1+N)$	$0\le i\le N-1$ $0\le j\le M-1$ $0\le p\le 1$	**b_{ij},t_{ij}= pesos** **p=umbral de vigilancia**
2: Aplicar nueva entrada			
3:Calcular Puntajes	$\mu_j=\sum_{i=0}^{N-1} b_{ij}(t)x_i$	$0\le j\le M-1$	**μ_j= salida del nodo** **x_i=i-ésimo elemento de la entrada**
4: Elegir el mejor ejemplar	$\mu_j=i_{max}\{u_i\}$		
5: Prueba de vigilancia	(1) $\lvert X\rvert=\sum_{i=0}^{N-1} x_i$ (2) $\lvert T\bullet X\rvert=\sum_{i=0}^{N-1} t_{ij}x_j$	si $\dfrac{\lvert T\bullet X\rvert}{\lvert X\rvert} > p$	**si: Ir a paso 6** **no: Ir a paso 7**

6: Desabilita el ejemplar mas parecido. La salida del nodo elegido como mas parecido en el paso 4 es temporalmente puesto a cero y no toma parte en la maximizacion del paso 4. Ir al paso 3			
7: Adapta el ejemplar mas parecido	$t_{ij}(t+1)=t_{ij}(t)x_i$ $b_{ij}(t+1)=$ $\frac{t_{ij}(t)x_i}{0.5+\sum_{i=0}^{N-1} t_{ij}(t)x_i}$		
8: Ir al paso 2 (habilitar los nodos desabilitados en 6)			

Un valor cercano a uno requiere una correspondencia cercana y valores menores aceptan una correspondencia mas pobre. Nuevas entradas son presentadas secuencialmente en la parte de abajo de la red, igual que en la red Hamming. Después de la presentación la entrada es comparada a todos los ejemplares almacenados en paralelo como en la red Hamming para producir correspondencias. El ejemplar con el más alto número de puntos de correspondencia es seleccionado usando inhibición lateral. Este es entonces comparado a la entrada mediante el cómputo de dividir el producto punto de la entrada y el ejemplar con mejor correspondencia (número de bits en uno en común) dividido por el número de bits en 1 en la entrada. Si esta división es mas grande que el umbral de vigilancia, entonces se considera que la entrada es similar al mejor ejemplar correspondiente y ése ejemplar es actualizado mediante ejecutar una operación **AND** entre éstos bits y aquellos de la entrada. Si el resultado es menor que el umbral de vigilancia, entonces se considera que la entrada es diferente que todos los ejemplares y se almacena como nuevo ejemplar. Cada nuevo ejemplar adicional, requiere un nodo y 2N conexiones para computar números de puntos correspondientes.

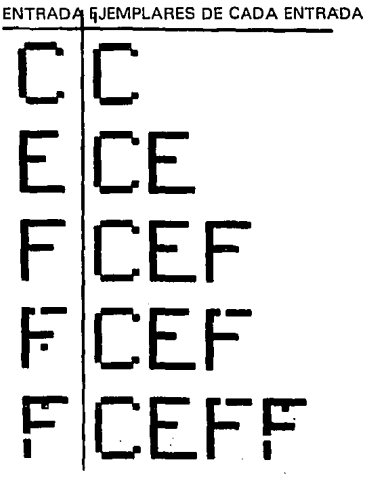


FIGURA 2-9 Un ejemplo del comportamiento de la red Carpenter-Grossberg para patrones de letras.

El comportamiento de la red Carpenter-Grossberg es ilustrado en la figura 2-9. Aquí se asume que los patrones a ser reconocidos son los tres patrones de las letras "**C**", "**E**" y "**F**" mostrados en la parte izquierda de ésta figura.

Estos patrones tienen 64 pixeles cada uno que toman el valor de 1 cuando son negros y el valor de cero cuando son blancos. Los resultados son mostrados cuando el umbral de vigilancia fué puesto a 0.9. Esto forza a separar patrones ejemplares a ser creados para cada letra.

El lado izquierdo de la figura 2-9 muestra la entrada a la red en pruebas sucesivas. El lado derecho presenta patrones ejemplares formados después de que cada patrón ha sido aplicado. En éste ejemplo, "**C**" fué presentado primero, seguido por "**E**", seguido por "**F**", etc. Después que la red es inicializada y una "**C**" es aplicada los pesos de conexión interna son alterados para formar un ejemplar interno que es idéntico a la "**C**". Después que una "**E**" es aplicada, Un nuevo ejemplar "**E**" es añadido. El comportamiento es similar para una nueva "**F**", dejando tres ejemplares almacenados. Si el umbral de vigilancia ha sido ligeramente disminuído, sólo dos ejemplares habrían sido presentados después de la "**F**"; uno para la "**F**", y uno para ambas "**C**" y "**E**" que habrían sido idénticos al patrón de "**C**".

Ahora, cuando una "**F**" alterada es aplicada sin un pixel negro en la parte superior, es aceptada como similar al ejemplar "**F**" y se degrada éste ejemplar debido a la operación **AND** ejecutada durante la actualización. Cuando otra "**F**" alterada es aplicada otra vez con sólo un pixel negro faltante, es considerada diferente de otros ejemplares y un nuevo ejemplar "**F**" alterado es añadido a los demás. Esto ocurrirá para posteriores entradas "**F**" almacenadas dando lugar a un crecimiento de ejemplares "**F**" alterados. Estos resultados ilustran que el algoritmo Carpenter-Grossberg puede desempeñar bien con patrones perfectos de entrada pero que aún una pequeña cantidad de ruido puede causar problemas. Sin ruido, el umbral de vigilancia puede ser puesto de tal manera que los dos patrones que son los más similares, sean considerados diferentes. Con ruido, sin embargo, éste nivel puede ser demasiado alto y el número de ejemplares almacenados puede rápidamente crecer hasta que todos los modelos disponibles hayan sido usados. Son necesarias modificaciones para mejorar el desempeño de éste algoritmo con ruido. Esto puede incluír pesos de adaptación más lentos y el cambio del umbral de vigilancia durante el entrenamiento y pruebas como es sugerido en [3].

El algoritmo LMS o algoritmo Widrow-Hoff [47] es idéntico al procedimiento de convergencia del perceptrón descrito anteriormente, excepto que la linealidad de límite fijo es hecha lineal o reemplazada por una no-linealidad de lógica de umbral. Los pesos son entonces corregidos en cada prueba por una cantidad que depende de la diferencia entre la entrada deseada y la real. Un clasificador que use el algoritmo de entrenamiento LMS puede usar salidas deseadas de 1 para clase A y de 0 para clase B. Durante la operación la entrada puede ser asignada a la clase A sólo si la salida fué arriba de 0.5. Las regiones de decisión formadas por los perceptrones son similares a aquellas formadas por un clasificador de máxima probabilidad el cuál asume que las entradas no tienen correlación y la distribución para diferentes clases difiere sólo en valores medios. Este tipo de clasificador Gaussiano y los pesos Euclidianos o la distancia métrica Euclidiana es frecuentemente usada en reconocimiento del habla donde hay datos de entrenamiento limitado y las entradas han sido ortogonalizadas por una transformación adaptable [36]. La caja 4 demuestra cómo los pesos y el umbral en un perceptrón pueden ser seleccionados de tal forma que la estructura del perceptrón compute la diferencia entre probabilidades logarítmicas requeridas por tal clasificador Gaussiano [7].

Estructuras similares al perceptrón pueden ser también usadas para desempeñar las computaciones lineales requeridas por una transformación Karhunen Loeve [36]. Estas computaciones pueden ser usadas para transformar un conjunto de $N + K$ entradas Gaussianas correlacionadas dentro de un reducido conjunto de N entradas no relacionadas que pueden ser usadas con éste clasificador Gaussiano.

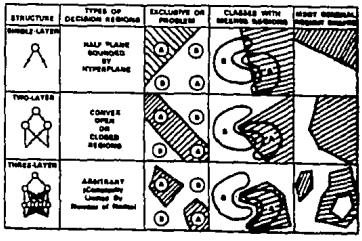


FIGURA 2-10 Tipos de regiones de decisión que pueden ser formados por perceptrones unacapa y multicapa con una y dos capas de unidades escondidas y dos entradas.

Si M_{Ai} y σ^2_{Ai} son la media y la varianza de la entrada x_i cuando la entrada es de la clase A y M_{Bj} y σ^2_{Bi} son la media y la varianza de la entrada x_j para la clase B y $\sigma_{2i}=\sigma^2_{Ai}=\sigma_{Bi}$, entonces los valores requeridos por un clasificador de máximos, son monotónicamente relacionados a:

$$L_A=\sum_{i=0}^{N-1}\frac{(x_i - M_{Ai})2}{\sigma^2_i}=\sum\frac{x^2_i}{\sigma^2_i}+2\sum\frac{M_{Ai}x_i}{\sigma^2_i}-\sum\frac{M^2_{Bi}}{\sigma^2_i}$$

y a:

$$L_B=\sum_{i=0}^{N-1}\frac{(x_i - M_{Bi})2}{\sigma^2_i}=\sum\frac{x^2_i}{\sigma^2_i}+2\sum\frac{M_{Bi}x_i}{\sigma^2_i}-\sum\frac{M^2_{Bi}}{\sigma^2_i}$$

Término I Término II Término III

Un clasificador de máxima semejanza debe calcular L_A y seleccionar la clase con la mayor semejanza. Ya que el término I en éstas ecuaciones es idéntico para L_A y L_B, puede suprimirse. El término II es un producto de las veces de la entrada de los pesos y puede ser calculado por un perceptrón y e término III es una constante que puede ser obtenida del umbral en un nodo del perceptrón. Un clasificador Gaussiano puede entonces ser formado por utilizar el perceptron estandar para calcular $L_A - L_B$ por hacer:

$$\omega=\frac{2(M_{Ai} - M_{Bi})}{\sigma^2_i}$$

y

$$\theta=\sum_{i=1}^{N-1}\frac{M^2_{Ai} - M^2_{Bi}}{\sigma^2_i}$$

CAJA 4. Un clasificador Gaussiano implementado utilizando la estructura del Perceptrón

Es válido generalizar la derivación de la caja 4 para demostrar cómo un clasificador Gaussiano para M clases puede ser construído desde M estructuras similares al perceptrón seguidas por una red que escoge el máximo.

La red requerida es idéntica en estructura a la red Hamming de la figura 2-4.

En éste caso, sin embargo, las entradas son análogas y los pesos y los nodos de umbral son calculados a partir de los términos II y III en las ecuaciones probabilísticas similares a aquellas para L_A en la caja 4.

Es igualmente válido generalizar la variante Widrow-Hoff del procedimiento de convergencia del perceptrón para aplicar para M clases. Esto requiere una estructura idéntica a la red Hamming y una regla de clasificación que seleccione la clase correspondiente al nodo con la máxima salida. Durante la adaptación los valores de salida deseados pueden ser puestos a 1 para la clase correcta y 0 para las otras.

La estructura del perceptrón puede ser usada para implementar ya sea una clasificador máximo probabilístico o clasificador el cuál usa el algoritmo de entrenamiento del perceptrón en una de sus variantes. La elección depende de la aplicación. El algoritmo de entrenamiento del perceptrón no asume nada concerniente a la forma de las distribuciones pero se enfoca en errores que ocurren cuando las distribuciones se sobreponen. Esto puede así ser mas robusto que las técnicas clásicas y trabaja bien cuando las entradas son generadas por procesos no-lineales y es fuertemente ortogonal y no-Gaussiano.

El clasificador Gaussiano hace fuertes conclusiones concernientes a las distribuciones y es más apropiado cuando las distribuciones son conocidas y corresponden a la conclusión Gaussiana. El algoritmo de adaptación definido por el procedimiento de convergencia del perceptrón es simple de implementar y no requiere de almacenar ninguna información extra mas que la presente en los pesos y en el umbral. El clasificador Gaussiano puede ser hecho adaptativo [24], pero debe ser almacenada información extra y las computaciones requeridas son más complejas.

Ni el procedimiento de convergencia del perceptrón ni el clasificador Gaussiano son apropiados cuando las clases no pueden ser separadas por un hiperplano. Dos de tales situaciones son presentadas en la parte superior de la figura 2-9. Los suaves contornos cerrados etiquetados A y B en ésta figura son la distribuciones de entrada para las dos clases cuando hay dos entradas de valores contínuos a las diferentes redes. Las áreas sombreadas son las regiones de decisión creadas por un perceptrón unicapa y otras redes de alimentación en adelanto. Las distribuciones para las dos clases para el problema de O-exclusiva están apartadas y no pueden ser separadas por una sola línea recta.

Este problema fué usado para ilustrar la debilidad del perceptrón por Minsky y Papert [32]. Si el racimo de abajo a la izquierda de clase B se considera el origen de su espacio de dos dimensiones entonces la salida del clasificador debe ser "Alta" sólo una no ambas entradas es "Alta". Una región de decisión posible para la clase A que un perceptron podría crear es ilustrada por la región sombreada en el primer renglón de la figura 2-9. Las distribuciones de entrada para el segundo problema mostradas en ésta figura son entretejidas y tampoco pueden ser separadas por una sola línea recta. Situaciones similares a éstas pueden ocurrir cuando parámetros tales como frecuencias formantes son usadas para reconocimiento del habla.

Como se subrayó arriba, un perceptrón unicapa forma regiones de decisión de medio plano. Un perceptrón de dos capas puede formar cualquier, posiblemente sin límites, region convexa en el espacio entre las entradas. Tales regiones incluyen polígonos convexos algunas veces llamados cáscaras convexas, y las regiones convexas ilimitadas mostradas en el renglón medio de la figura 2-9.

Aquí el término convexo significa que cualquier línea que une puntos en el borde del de una región, va sólo a través de puntos dentro de tal región. Las regiones convexas son formadas de intersecciones de regiones del medio plano formadas por cada nodo en la primera capa del perceptrón multicapa.

Cada nodo en la primer capa se comporta como un perceptrón unicapa y tiene una salida "Alta" sólo para puntos en un sólo lado del hiperplano formado por sus pesos y offset.

Si los pesos a un nodo de salida desde los N nodos de la primera capa son todos 1.0 y el umbral 1 en el nodo de salida es N - ε donde $0 < \varepsilon < 1$, donde el nodo de 1 salida será alto sólo si las salidas de todos los nodos de la primera capa están en "Alto". Esto corresponde a efectuar una operación **AND** en el nodo de salida y resulta en una región de decisión final que es la intersección de todas las regiones de medios planos que forman regiones convexas como se describe arriba. Estas regiones convexas tienen como máximo tantos lados como nodos en la primera capa.

Este análisis se adentro de algún modo en el problema de seleccionar el número de nodos para usar en un perceptron de dos capas. El número de nodos debe ser suficientemente grande como para formar una region de decisión tan compleja como la requerida por un problema dado. Esta no debe, sin embargo, ser tan grande que los muchos pesos requeridos no puedan ser confiadamente estimados de los datos de entrenamiento disponibles.

Por ejemplo, dos nodos son suficientes para resolver el problema de O-exclusiva como se muestra en el segundo renglón de la figura 2-9. Ningún número de nodos, sin embargo, pueden separar las entretejidas regiones de clase en la figura 2-9 con un perceptrón de dos capas. Un perceptrón de 3 capas puede formar regiones de decisión arbitrariamente complejas y puede separar las entretejidas clases como se muestra en la parte de abajo de la figura 2-9. Este puede formar regiones tan complejas como aquellas formadas usando distribuciones mezcladas y clasificadores de mas próximo vecino [7]. Esto puede ser probado por construcción.

La prueba depende de particionar la region de decision deseada en pequeños hipercubos (cuadrados cuando hay dos entradas). Cada hipercubo requiere 2N nodos en la primera capa (cuatro nodos cuando hay dos entradas), uno para cada lado del hipercubo, y un nodo en la segunda capa, que toma la operación **AND** lógica de las salidas de los nodos de la primera capa. Las salidas de los nodos de la segunda capa estarán "Altas" Sólo para entradas dentro de cada hipercubo.

Los hipercubos son asignados a la región de decisión adecuada mediante conectar la salida de cada nodo de la segunda capa sólo al nodo de salida correspondiente a la region de decisión que el nodo del hipercubo se encuentra y entonces hacer una operación lógica OR en cada nodo de salida. Será efectuada una operación lógica OR si éstos pesos de conexión de la segunda capa escondida a la capa de salida son uno, el umbral en los nodos de salida es 0.5. Este procedimiento de construcción puede ser generalizado para usar regiones convexas arbitrariamente sombreadas en vez de pequeños hipercubos y es capaz de generar las regiones no convexas desconectadas mostradas en la parte de abajo de la figura 2-9.

El análisis anterior demuestra que no mas de tres capas son requeridas en redes de alimentación en adelanto tipo perceptrón porque una red de tres capas es capaz de generar regiones de decisión arbitrariamente complejas. Estas también se adentran en el problema de seleccionar el número de nodos para usar en perceptrones de tres capas. El número de nodos en la segunda capa debe ser mayor que uno cuando las regiones de decisión están desconectadas o entretejidas y no puede estar formada de un área convexa. El número de nodos en la segunda área requeridos en el peor de los casos es igual al número de regiones desconectadas en las distribuciones de entrada. El número de nodos en la primera capa debe ser típicamente suficiente para proveer tres o más fronteras para cada área convexa generada por cada nodo de la segunda capa. Esto debe así típicamente ser más de tres veces la cantidad de nodos en la segunda capa que en la primera capa.

La anterior discusión, centrada principalmente en perceptrones multicapa con una entrada cuando se utilizan no-linealidades con límites duros, es usada. Comportamiento similar es mostrado por perceptrones multicapa con múltiples nodos de salida cuando son usadas no-linealidades sigmoidales y la regla de decisión es seleccionar la clase correspondiente al nodo de salida con la salida más grande. El comportamiento de éstas redes es más complejo, porque las regiones de decisión son típicamente rodeadas por curvas suaves en vez de elementos rectos y así es más difícil el análisis. Estas redes, sin embargo, pueden ser entrenadas con el nuevo algoritmo de entrenamiento de back-propagation [40].

El algoritmo de Back-Propagation [40, 41] descrito en la caja 5 es una generalización del algoritmo LMS. Usa una técnica de búsqueda de gradiente para minimizar una función costeada igual a la diferencia cuadrática media entre las salidas de la red deseadas y las reales. La salida deseada de todos los nodos es normalmente "Baja" (0 o 0.1) a menos que ése nodo corresponda a la clase correspondiente a la entrada actual, en cuyo caso es "Alta" (1.0 o 0.9).

La red es entrenada por seleccionar inicialmente pequeños pesos aleatorios y umbrales internos y entonces presentar todos los datos de entrenamiento repetidamente. Los pesos son ajustados después de cada prueba usando información lateral especificando la clase correcta hasta que los pesos converjan y la función costeada es reducida a un valor aceptable. Un componente esencial del algoritmo es el método iterativo descrito en la figura 2-10. Esta figura muestra regiones de decisión formadas por un perceptrón de dos capas con dos entradas, ocho nodos en la capa interna, y dos nodos de salida correspondientes a dos clases.

Fueron usadas no linealidades sigmoidales como en la caja 5, el término de ganancia η fué 0.3, el término de momentum α fué 0.7, muestras aleatorias de las clases A y B fueron presentadas en pruebas alternadas, y las salidas deseadas fueron 1 ó 0. Muestras de la clase A fueron distribuidas uniformemente sobre un círculo de radio 1 centrado en el origen.

Las muestras de la clase B fueron distribuidas uniformemente fuera de éste círculo hasta un radio de 5. La region de decision inicial es un hiperplano ligeramente curvo. Esta cambia gradualmente hacia una region circular que encierra la distribución circular de la clase A después de 200 pruebas (100 muestras de cada clase). Esta region de decision está cerca de la region óptima que sería producida por un clasificador de máxima probabilidad.

CAJA 5: Algoritmo Backpropagation

1: Inicializa pesos con pequeños valores random	
2: Presenta entrada y salida deseada	
3: Calcula salida actual, utilizando función de no-linealidad (compresora)	
4: Adapta los pesos	$\delta=OUT(1-OUT)(Target-OUT)$ $\Delta\omega pq,k=\eta\delta q,kOUTp,j$ $\omega pq,k(n+1)=\omega pq,k(n)+\Delta\omega pq,k$
5: Adapta los pesos de las capas ocultas	$\delta p,j=OUTpj(1-OUT_{pj})(\Sigma\delta_{q,k}\omega pq,k)$
6: Ir al paso 2	

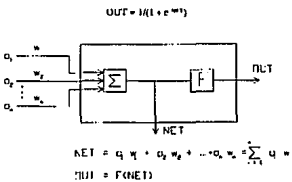


FIGURA 3-1 Red artificial con función de activación

Por mucho años no hubo algoritmo teórico para entrenar redes neuronales artificiales multicapa, ya que las redes sencillas probaron ser severamente limitadas en lo que podrían representar. (y de aquí, en lo que podrían aprender), el campo entero se vió entro de un eclipse virtual. La invención de el algoritmo de backpropagation jugó un papel importante en el resurgimiento del interés en las redes neuronales artificiales. Backpropagation [40, 41] es un método sistemático para entrenar redes neuronales artificiales multicapa. Tiene un fundamento matemático que es sólido pero no es muy práctico. Si tomar en cuenta sus limitaciones, backpropagation ha expandido dramáticamente el rango de problemas en los cuales pueden aplicarse las redes neuronales artificiales, y ha generado muchas demostraciones exitosas de su poder.Backpropagation tiene una historia interesante. Rumelhart, Hinton, y Williams (1986) presentaron una descripción clara y concisa del algoritmo de backpropagation. Parker (1982), a su vez se había adelantado a Rumelhart. Un poco antes de ésto, Werbos (1972) habia ya descrito el método. Rumelhart y Parker podrían haber ahorrado un gran trabajo y esfuerzo si hubieran estado pendientes del trabajo de Werbos.

Aunque duplicación similar de esfuerzos se ha encontrado en virtualmente toda disciplina científica, en las redes neuronales artificiales el problema es particularmente severo debido a la estructura interdisciplinaria del sujeto en cuestión. La investigación en redes neuronales es publicada en libros y revistas de tan diversos campos que aún el investigador mas diligente puede no estar al día en investigaciones de éste campo.

CONFIGURACIONES DE RED PARA EL ALGORITMO BACK-PROPAGATION

La figura 3-1 muestra la neurona usada como el bloque estructural fundamental para las redes Backpropagation. Un conjunto de entradas es aplicado, ya sea desde afuera o desde una capa anterior. Cada uno de éstos conjuntos es multiplicado por un peso, y los productos son sumados. Estas sumas de productos es llamado NET y debe ser calculado para cada neurona de la red. Después de que NET es calculado, una función de activación F es aplicada para modificarlo, produciendo así la señal de salida OUT.

La figura 3-2 muestra la función de activación usualmente utilizada para Backpropagation.

$$OUT = 1 / (1 + e^{-NET}) \quad (3\text{-}1)$$

Como se muestra por la ecuación 3-2, ésta función, llamada una sigmoide, es deseable que posea derivada de primer grado, un hecho que utilizamos en implementar el algoritmo de Backpropagation:

$$\frac{\partial OUT}{\partial NET} = OUT\,(1\text{-}OUT) \quad (3\text{-}2)$$

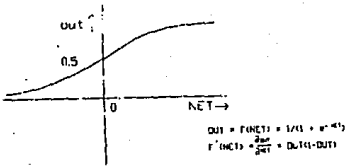


FIGURA 3-2 Función sigmoidal de Activación

Algunas veces llamada función logística o simplemente función compresora, el sigmoide comprime el rango de **NET** de tal forma que **OUT** cae entre 0 y 1.

Como se discutió previamente, las redes multicapa tienen mayor poder representacional que las redes unicapa sólo cuando es introducida una no linealidad. La función compresora produce la no linealidad necesitada.

Existen muchas funciones que pueden ser usadas; el algoritmo de Backpropagation sólo requiere que la función sea completamente diferenciable. El sigmoide satisface éste requerimiento. Tiene la ventaja adicional de que provée una forma de control automático de ganancia. Para señales pequeñas, (**NET** cercano a cero) la tangente de la curva entrada/salida es muy inclinada, produciendo una gran ganancia. Si la magnitud de la señal se hace mas grande, la ganancia decrece. En ésta forma las señales grandes pueden ser acomodadas por la red sin saturación, mientras que las señales pequeñas se les permite pasar a través sin atenuación excesiva.

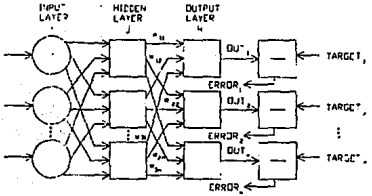


FIGURA 3-3 Red Back-propagation de dos capas

LA RED MULTICAPA

La figura 3-3 muestra una red multicapa hecha para entrenar con Backpropagation. (La figura ha sido simplificada para claridad). El primer conjunto de neuronas (conectadas a las entradas) sirven sólo como puntos de distribución; no desempeñan sumatoria de entrada. La señal de entrada es simplemente pasada a través hacia los pesos en sus salidas. Cada neurona en capas subsecuentes producen las señales **NET** y **OUT** descritas arriba.

La literatura es inconsistente en definir el número de capas en éstas redes. Algunos autores se refieren al número de capas de neuronas (incluyendo la capa de entrada que no suma), otros a las capas de los pesos.

A causa de que la última definición es más funcionalmente descriptiva, es utilizada a través de éste libro.Por ésta definición, la red de la figura 3-3 se considerada que consiste de dos capas. También, una neurona es asociada con los pesos que conectan a su entrada. Así, los pesos en la capa 1 terminan en las neuronas de la capa 1. La entrada o distribución es designada capa 0.

Backpropagation puede ser aplicada a redes con cualquier número de capas; sin embargo, sólo dos capas de pesos son necesarias para demostrar el algoritmo. En éste punto de discusión, sólo redes de alimentación en adelanto son consideradas. También es posible aplicar Backpropagation a redes con retroalimentación; lo cual es discutido después.

UNA INTRODUCCION AL ENTRENAMIENTO

El objetivo de entrenar la red es ajustar los pesos de tal forma que la aplicación de un conjunto de entradas produce el conjunto de salidas deseado. Por razones de brevedad, éstos conjuntos de entrada-salida pueden ser referidos como ***vectores***. El entrenamiento asume que cada vector de entrada es aparejado con un vector objetivo representando la salida deseada; juntos éstos vectores son llamados un ***par de entrenamiento***. Usualmente una red es entrenada sobre un número de pares de entrenamiento. Por ejemplo, la parte de entrada de un par de entrenamiento puede consistir de un patrón de unos y ceros representando una imagen binaria de una letra o del alfabeto. La figura 3-4 muestra un conjunto de entradas para la letra A, dibujada en una cuadrícula.

Si una línea se pasa del cuadrado, la correspondiente entrada de la neurona es uno; de otra manera, ésa entrada de la neurona es cero. La salida pudiera ser un número que represente a la letra A, o tal vez otro conjunto de unos y ceros que pudieran ser usados para producir un patrón de salida. Si uno deseara entrenar la red para reconocer todas las letras del alfabeto, se requerirían 26 pares de entrenamiento. El grupo de pares de entrenamiento es llamado ***grupo de entrenamiento***.Antes de empezar el proceso de entrenamiento, todos los pesos deben ser inicializados a pequeños números aleatorios. Esto asegura que la red no está saturada por grandes valores de los pesos, y previene otras patologías del entrenamiento.Por ejemplo, si todos los pesos empiezan con valores iguales y el desempeño deseado requiere valores diferentes, la red no aprenderá. Entrenar la red de Backpropagation requiere los pasos que siguen:

1. Seleccionar el próximo par de entrenamiento del conjunto de entrenamiento; aplicar el vector de entrada a la entrada de la red.

2. Calcula la salida de la red

3. Calcula el error entre la salida de la red y la salida deseada (el vector objetivo del par de entrenamiento)

4. Ajustar los pesos de la red en una forma que minimize el error.

5. Repetir los pasos del 1 al 4 para cada vector en el conjunto de entrenamiento hasta que el error para el conjunto entero sea aceptablemente bajo

Las operaciones requeridas en los pasos 1 y 2 son similares a la manera en la cual la red entrenada será usada; ésto es, un vector de entrada es usado y la salida es calculada. Los cálculos son desempeñados capa por capa. Refiriéndonos a la figura 3-3, son calculadas primero las salidas de las neuronas en la capa ***j***; éstas son entonces usadas como entradas a la capa ***k***; las neuronas de salida de la capa ***k*** son calculadas y éso constituye el vector de salida de la red.

En el paso 3, cada una de las salidas de la red etiquetada como **OUT** en la figura 3-3, es substraída de su correspondiente componente del vector objetivo para producir un error. Este error es usado en el paso 4 para ajustar los pesos de la red, donde la polaridad y magnitud de los cambios de los pesos es determinada por el algoritmo de entrenamiento. (Ver abajo). Después de las suficientes repeticiones de éstos cuatro pasos, el error entre las salidas reales y las salidas objetivo debe ser reducido a un valor aceptable, entonces se dice que la red está entrenada. En éste punto la red es utilizada para recognición y los pesos no son cambiados.

Puede verse que los pasos 1 y 2 constituyen un "paso hacia adelante" en el que la señal se propaga desde la entrada de la red hacia su salida. Los pasos 3 y 4 son un "paso hacia atrás"; aquí, la señal de error calculada se propaga hacia atrás (backward) a través de la red donde es usada para ajustar los pesos. Estos dos pasos son ahora expandidos y expresadas en una forma un poco mas matemática.

PASO HACIA ADELANTE

Los pasos 1 y 2 pueden ser expresados en forma de vector como sigue: un vector de entrada **X** es aplicado y un vector de salida **Y** es producido. El par vector objetivo **X** y **Y** vienen del conjunto de entrenamiento. El cálculo es efectuado en **X** para producir el vector de salida **Y** .Como hemos visto, el cálculo en redes multicapa es hecho capa por capa, empezando en la capa más cercana a las entradas. El valor **NET** de cada neurona en la primera capa es calculado como la suma pesada de las entradas de sus neuronas. La función de activación *F* entonces "comprime" **NET** para producir el valor **OUT** para cada neurona en la capa. Una vez que el conjunto de salidas de una capa es encontrado, sirve como entrada para la próxima capa. El proceso es repetido, capa por capa, hasta que el conjunto final de salidas de la red es producido.

Este proceso puede ser establecido en notación vectorial. Los pesos entre neuronas pueden ser considerados como la matriz **W**. Por ejemplo, el peso de la neurona 8 en la capa 2 a la neurona 5 en la capa 3 es designado $w_{8,5}$. Mejor que utilizar la suma de productos, el vector **NET** para una capa **N** puede ser expresado como el producto de **X** y **W**. En notación vectorial **N = XW**. Aplicando la función *F* al vector **NET** de **N**, componente por componente, produce el vector de salida **O**. Así, para una capa dada, la siguiente expresión describe el proceso de cálculo:

$$\mathbf{O = F(XW)} \quad (3\text{-}3)$$

El vector de salida para una capa es el vector de entrada de la próxima, entonces el cálculo de las salidas de la capa final requiere la aplicación de la ecuación 3-3 para cada capa, desde la entrada de la red hasta su salida.

PASO HACIA ATRAS

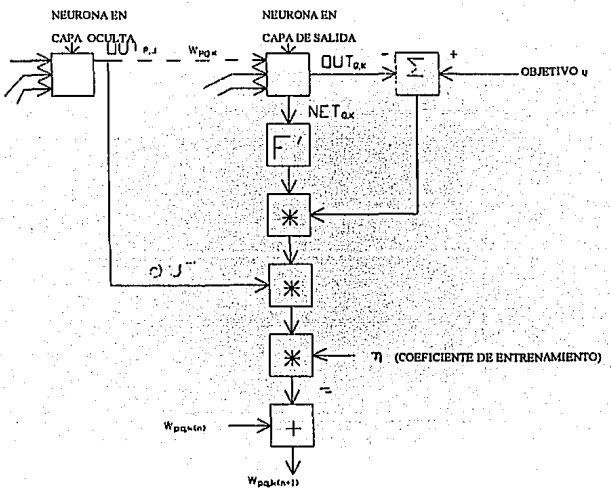


FIGURA 3-4 Entrenamiento de un peso en la capa de salida

Ajustar los pesos de la capa de salida

Ya que un valor objeto es disponible para cada neurona en la capa de salida, ajustar los pesos asociados es efectuado fácilmente, utilizando una modificación de la regla delta.

Las capas interiores son referidas como "capas escondidas", ya que sus salidas no tienen valores objetivo para comparación; de aquí, el entrenamiento es más complicado.La figura 3-5 muestra el proceso de entrenamiento, para un solo peso desde la neurona ***p*** en la capa escondida ***j*** a la neurona ***q*** en la capa de salida ***k***, para producir una señal de **ERROR**. Esto es multiplicado por la derivada de la función compresora **OUT(1-OUT)** calculada para la neurona ***k*** de ésa capa, produciendo entonces el valor δ.

$$\delta = \mathbf{OUT(1\text{-}OUT)(Target\text{-}OUT)} \quad (3\text{-}4)$$

Entonces δ es multiplicado por **OUT** desde una neurona *j*, la neurona fuente para el peso en cuestión. Este producto es a su vez multiplicado por un coeficiente de entrenamiento η (típicamente de 0.01 a 1.0) y el resultado es sumado al peso. Un proceso idéntico es efectuado para cada peso procedente de desde una neurona en la capa oculta hacia una neurona en la capa de salida.

Las siguientes ecuaciones ilustran éste cálculo:

$$\Delta\omega_{pq,k} = \eta\delta_{q,k}\, OUT_{p,j} \quad (3\text{-}5)$$

$$\omega_{pq,k(n+1)} = \omega_{pq,k(n)} + \Delta\omega_{pq,k} \quad (3\text{-}6)$$

donde:

$\omega_{pq,k(n)}$ es el valor de un peso desde la neurona *p* en la capa escondida a la neurona *q* en la capa de salida en el paso *n* (antes del ajuste); note que la *k* indica que el peso es asociado con su capa destino.

$\omega_{pq,k(n+1)}$ es el valor del peso en el paso *n*+1 (después del ajuste)

$\delta_{q,k}$ es el valor de δ para la neurona q en la capa de salida k

$\mathbf{OUT}_{pj}$ es el valor de **OUT** para la neurona p en la capa escondida j.

Note que p y q se refieren a neuronas específicas, donde j y k se refieren a capas.

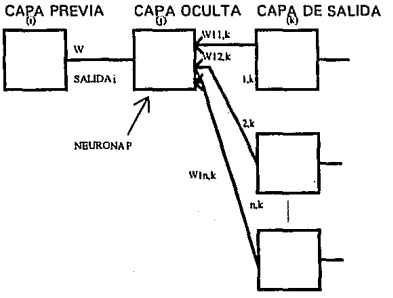
Ajustar los pesos de las capas ocultas

Las capas ocultas no tienen vector objetivo, entonces el proceso de entrenamiento descrito arriba no puede ser usado. Esta falta de un objetivo de entrenamiento impidió los esfuerzos de entrenar redes multicapa hasta que backpropagation proveyó de un algoritmo utilizable.

Backpropagation entrena las capas escondidas mediante propagar el error de salida hacia atrás a través de la red capa por capa, ajustando los pesos en cada capa. La ecuación 3-5 y 3-6 son utilizadas para todas las capas, tanto de salida como escondidas; sin embargo, para capas escondidas δ debe ser generada sin el beneficio de un vector objetivo.

La figura 3-5 muestra como ésto es llevado a cabo.

FIGURA 3-5 Ajustando los pesos de las capas ocultas



Primero, δ es calculado para cada neurona en la capa de salida, como en la ecuación 3-4. Esto es usado para ajustar los pesos que alimentan la capa de salida, entonces se propaga hacia atrás a través de los mismos pesos para generar un valor de d para cada neurona en la primera capa de salida. Estos valores de δ son utilizados a su vez para ajustar los pesos de ésta capa escondida y, en forma similar, es propagada hacia atrás a todas las capas precedentes.

Considére una neurona en la capa oculta justo antes de la capa de salida. En el paso en adelanto, ésta neurona propaga su valor de salida hacia las neuronas en la capa de salida a través de los pesos de interconexión. Durante el entrenamiento, éstos pesos operan en reversa, pasando el valor de d desde la capa de salida hacia la capa oculta. Cada uno de éstos pesos es multiplicado por el valor de d de la neurona a la cuál se conecta en la capa de salida. El valor de d necesario para la neurona de la capa oculta es producido por sumar todos éstos productos y multiplicarlos por la derivada de la función compresora:

$$\delta_{p,j} = OUT_{pj}(1-OUT)(\Sigma_q \delta_{q,k}\, \omega_{pq,k}) \quad (3\text{-}7)$$

(ver figura 3-6) Ya con δ en la mano, los pesos que alimentan la primera capa oculta pueden ser ajustados utilizando las ecuaciones 3-5 y 3-6, modificando los índices para indicar las capas correctas.

Para cada neurona en una capa oculta dada, los valores de δ deben ser calculados, y todos los pesos asociados con ésa capa deben ser ajustados. Esto es repetido, regresando a hacia la capa de entrada, capa por capa, hasta que todos los pesos son ajustados.

Con notación vectorial, la operación de propagar el error hacia atrás puede ser expresada en forma mucho más compacta. Sea D_k el conjunto de δs y el conjunto de pesos para la capa de salida a arreglar W_k. Para llegar a D_k (δ de la capa oculta), el vector δ, es decir D_k, se siguen éstos dos pásos:

1.- Multiplicar el vector δ de la capa de salida D_k por la transpuesta de la matriz de pesos que conecta la capa oculta a la capa de salida (W_k^t)

2.- Multiplicar cada componente del producto resultante por la derivada de la función compresora para la neurona correspondiente en la capa oculta.

Simbólicamente:

$$D_j = D_k W_k^t \$ [O_j \$ (I - O_j)] \text{ (3-8)}$$

donde **$** indica la multiplicación componente a componente de dos vectores. Oj es el vector de salida para la capa oculta j, e **I** es un vector cuyos todos componentes son 1.

Añadiendo una neurona de dirección

En muchos casos, es deseable proveer cada neurona con una dirección de entrenamiento. Esto desfasa el origen de la función logística, produciendo un efecto que es similar al ajustar el umbral de la neurona del perceptrón, permitiendo así una convergencia más rápida del proceso de entrenamiento. Esta característica es fácilmente incorporada dentro del algoritmo de entrenamiento; un peso conectado a +1 es añadido a cada neurona. Este peso es entrenable en la misma manera que lo son los otros pesos, excepto que la fuente es siempre +1 en vez de ser la salida de una neurona de una capa previa.

Momentum

Rumelhart, Hinton y Williams (1986) [40] describen un método para mejorar el tiempo de entrenamiento del algoritmo de Backpropagation, mientras se mejora la estabilidad del proceso.

Llamado Momentum, el método consta de añadir un término al ajuste del peso que es proporcional a la cantidad del cambio de peso previo. Una vez que un ajuste es efectuado, es "recordado" y sirve para modificar todos los ajustes de peso subsecuentes. Las ecuaciones de ajuste son modificadas como sigue:

$$\Delta \omega_{pq,k}(n+1) = \eta(\delta_{q,k} OUT_{pj}) + \alpha[\Delta \omega_{pq,k}(n)] \text{ (3-9)}$$

$$\omega_{pq,k}(n+1) = \omega_{pq,k}(n) + \Delta \omega_{pq,k}(n+1) \text{ (3-10)}$$

donde α, el coeficiente de momentum, es comúnmente puesto a 0.9.

Utilizando el método de momentum, la red tiende a seguir el fin de canales angostos en la superficie de error (si existen) mas que cruzar rápidamente de lado a lado. Este método parece trabajar bien en algunos problemas, pero tiene poco efecto o efecto negativo en otros.

Sejnowski y Rosenberg (1987) [43] describen un método similar basado en un suavizado exponencial que puede ser superior en algunas aplicaciones.

$$\Delta \omega_{pq,k}(n+1)=\alpha \Delta \omega_{pq,k}(n)+(1-\alpha)\delta_{q,k} \mathbf{OUTpj} \quad (3\text{-}11)$$

Entonces el cambio de peso es computado:

$$\omega_{pq,k}(n+1) = \omega_{pq,k}(n) + \eta \Delta \omega_{pq,k}(n+1) \quad (3\text{-}12)$$

donde α es un coeficiente en el rango de 0.0 a 1.0. Si α es 0.0, entonces el suavizado es mínimo; el ajuste entero del peso viene del nuevo cambio calculado. Si α es 1.0, el nuevo ajuste es ignorado y el cálculo previo es repetido. entre 0 y 1 hay una region donde el ajuste del peso es frenado por un coeficiente proporcional a α. Otra vez η es el coeficiente de entrenamiento, sirviendo para ajustar el tamaño del promedio de cambio de peso.

CAPITULO 4. EL ANALISIS DE SENSIBILIDAD, EL ALGORITMO BACK-PROPAGATION MEJORADO Y ALGUNAS APLICACIONES

Una de las áreas principales en las cuales las redes neuronales y los sistemas expertos difieren es en cómo interiorizan la información que contienen. En sistemas expertos tradicionales, la información es mantenida en forma de reglas.

Cuando un sistema experto llega a una conclusión, uno puede ir a su base de conocimiento para ver cuál de sus reglas fué activada para encontrar la conclusión final. En caso de una red neuronal, éste tipo de explicación no es fácil de obtener porque las redes almacenan la información en forma de pesos o "fuerzas sinápticas". Los pesos en una red neuronal tienen valores numéricos más que simbólicos, haciendo difícil el extraer una explicación simbólica.

Sin embargo, es posible aprender más acerca de cómo las varias entradas afectan la salida de una red neuronal. Esto es efectuado mediante la aplicación de una técnica proveniente del campo del análisis estadístico no-paramétrico llamada "análisis de sensibilidad". [24]

Aquí se describe el análisis de sensibilidad y cómo puede ser aplicado para explicar el "pensamiento" de una red neuronal.

¿ QUE ES EL ANALISIS DE SENSIBILIDAD ?

El análisis de sensibilidad [6] inspecciona cada una de las entradas y determina cuanto y de que tipo de impacto un pequeño cambio en la entrada puede tener en la salida. Si un pequeño cambio en la entrada en particular ocasiona un cambio drástico en una salida en particular, ésa entrada debe ser considerada como uno de los factores principales en la salida resultante.

El proceso real de aplicar análisis de sensibilidad consiste en empezar con la primera entrada de la red y añadir una pequeña cantidad, recomputar las salidas, sustraer una pequeña cantidad de la entrada de la red, recomputar la salida, tomar entonces la diferencia entre las dos salidas (el cambio de las salidas) y dividirlo entre las diferencia de las dos entradas (en total). Este resultado es M valores, uno para cada salida. Este proceso es repetido para cada una de las entradas de la red. El resultado es una matriz de MxN donde M es el número de salidas, y N el número de entradas. La cantidad más pequeña de cambio en la entrada es llamada "incertidumbre" (Dither).

Como comentario, ésta técnica básica es usada internamente en una variedad de redes neuronales para determinar cual elemento de proceso debe ser modificado. En particular, es usado en ciertas variantes de Madaline, y también en ciertas técnicas de optimización de redes.

Como un ejemplo de cómo el análisis de sensibilidad trabaja, supongamos que una red neuronal ha sido entrenada para determinar la confianza para otorgar crédito. Las entradas a ésta red son información de una solicitud de crédito, renta o propietario, etc. Para el aplicante actual, digamos que la salida es 0.3 (menor que 0.5 es incierto).

¿ Cuál de las entradas tienen que cambiar menos para cambiar la salida actual (incierta: 0.3) a una aceptación ? Lo mas facil sería el requerir al solicitante incrementar su sueldo en un 10% resultando una "aceptación" (salida = 0.6). La razón por la cual el resultado del crédito fué negado es el sueldo bajo. Casi no hay diferencia si la persona renta o es propietaria de su casa. Por otra parte, una razón secundaria para negarle el crédito pudiera ser que los gastos del solicitante fueran muy altos. Si los gastos fueran reducidos de un 20 a un 30% (preferentemente a un 10%), éso podría ser suficiente para elevar la salida del evaluador del crédito sobre el umbral de rechazo/aceptar de 0.5.

Es muy importante notar que la manera en la cual cada una de ésas variables de entrada afecta la salida depende de sus valores actuales. Para otro solicitante, el factor mas importante podría set el estatus de "renta/propietario".

En muchas maneras, ésto es análogo a cómo las personas enfrentan decisiones. Cuando es difícil el hacer un decisión entre dos caminos de acción, es usualmente porque existe un factor particular (sin embargo, puede haber más), que puede estar justo en la línea de decisión. Si hubiera un poco mas en un camino que en el otro, la decisión sería mas fácil de tomar.

Por ejemplo, la compañia NeuralWare ofrece un paquete de software intoductorio por $ 99.00 usd así como también un sofisticado paquete por $1,895.00 usd ¿ Gastaría $ 99.00 usd por un paquete para aprender redes neuronales ? Sí lo haría, casi sin pensarlo (nivel de compra = 1.0). ¿ Gastaría 1,895.00 usd ? Probablemente no, al menos que tuviera una aplicación específica y quisiera asegurarme de tener la mejor herramienta posible para asegurar el éxito (nivel de compra de educación = 1.0 ; nivel de compra de aplicación = 0.8) ¿ Gastaría $ 200.00 usd tal vez (nivel de compra = 0.5). ¿500.00 ? No, al menos que realmente lo quisiera (nivel de compra = 0.3)

Así, enfrentando la decisión de cuánto pagaría para adquirir un paquete de software para aprender de redes neuronales, mientras más alto el precio, menos estaría dispuesto a pagar. Menos de $ 150.00 usd lo compraría casi sin pensar. Sobre éste precio, necesitaría motivaciones adicionales.

¿ Pero si tuviera una aplicación especial en mente, pagaría $ 1,895.00 usd por el producto ? Sí, basado en capacidades apropiadas (nivel de compra = 0.8). ¿ $ 3,000.00 ? Sí, basado otra ve en capacidades apropiadas (nivel de compra = 0.7). ¿ $ 500.00 ? Tal vez, pero podría querer revisar muy cuidadosamente las opciones de productos. (nivel de compra = 0.5). ¿ Por qué podría sentirme mas confortable pagando $ 2,000.00 o $ 3,000.00 usd por un producto para resolver una aplicación particular ? Porque sé por experiencia que las buenas herramientas cuestan. Las baratas cuestan mas en esfuerzo perdido que el dinero ahorrado. (con ciertas notables excepciones).

La compra de un producto para la educación es muy sensitiva al precio cuando el precio es mayor de cierto umbral. Cuando se compra una buena herramienta para desarrollar aplicaciones, arriba de cierto umbral de precio, otros factores se vuelven mas importantes y el precio es irrelevante. Una observación clave de ésto es que la forma en la que el precio afecta mi voluntad de comprar es dependiendo del propósito para el cual yo efectúe la compra del producto. Esto es verdadero para el análisis de sensibilidad en general. Los cambios resultantes en la salida,son válidos sólo para el actual conjunto de entradas.

Si un pequeño cambio en las entradas pueden no tener efecto en las salidas no significa que ésa entrada en particular no sea importante. Combinada con otro conjunto de entradas, un pequeño cambio en la entrada puede tener gran impacto en la salida.

EL PROGRAMA EJEMPLO

El programa una red neuronal diseñada para entrenar una red Backpropagation y entonces tenerla explicándose a sí misma utilizando análisis de sensibilidad. Una rutina principal selecciona cuál función desempeñar y permite a la red y las explicaciones ser impresas en un archivo para inspección posterior.

El programa está organizado en una forma completamente simple. Las rutinas de utilería para carga, grabado, borrar, crear e imprimir la red están en el inicio. Son invocadas directamente desde el programa principal. La rutina Recall es usada para desempeñar una sencilla ejecución de la red para determinar su respuesta a un conjunto de entradas.

Learn desempeña una actualización simple de los pesos de la red. La regla Delta normalizada con momentum es usada para entrenar los pesos.

Train repetidamente invoca Learn, pasándole ejemplos desde el conjunto estático de entrenamiento (testE[]). Para asegurar que todos los ejemplos de datos están presentes, es usada una estrategia de "Shuffle & Deal". esto es implementado en NextTestN, el cual crea una lista mezclada de ejemplos, lo distribuye al programa de muestra, y lo remezcla cuando la pila está vacía.

ANALIZANDO LOS RESULTADOS

La red fué entrenada hasta que el error de la raíz media cuadrática fuera menos que 0.001. . El conjunto de entrenamiento fué diseñado para hacer aleatoria a la entrada 3. Si ésta fuera completamente aleatoria, los pesos conectados a ella deberían ser cercanos a cero.

La tabla 3 es una impresión de la red. Note que la magnitud de los pesos asociados con la entrada 3 (iluminada en la tabla) son casi todos muy pequeños en comparación de las otras entradas a cada elemento de procesamiento. Esto significa que la entrada 3 típicamente tendrá poco efecto en la salida de la red. El elemento de proceso 0 es el términno "bias" y su salida es siempre 1.0.

Las tablas 4(b) y 4(c) muestran la sensibilidad de la salida a pequeños cambios en la primera y segunda variables, respectivamente. Las tablas tienen rango sobre los posibles valores de entrada (la entrada 1 sobre el eje x, la entrada 2 sobre el eje y). Note de ésas tablas que la salida es mas sensitiva cuando la variable en particular está cercana a una frontera. Si se piensa un momento en ello, tiene sentido.

En una región lejos de la frontera, los cambios pequeños tendrán poco efecto sobre las salidas. Sólo cerca de la frontera, donde las salidas cambian de un estado al otro, tendrá la entrada apropiada considerable impacto. Esto sugiere algunas modificaciones al programa. En particular, si un pequeño cambio en cada una de las entradas produce poco o ningún efecto en las salidas, incremente la magnitud del cambio en la entrada hasta que el cambio sea significante.

Observando las tablas de salida , notará que la salida es siempre sensitiva a la entrada cuando hay una transición. Sin embargo, el conjunto de entrenamiento tiene fronteras muy definidas y sólo una variable cambia a cada una de ellas. Esto puede hacer que se sospeche de problemas. Realmente, el problema resulta de ciertas características básicas de chipotes (hump) y los adapta para construir el mapeo resultante de entradas a salidas. Como resultado, las fronteras están un poco informes haciendo la salida muy sensible a las entradas.

Otro comentario. Nótese en el primer ejemplo de aprobamiento de crédito. Los cambios a las variables de entrada fueron como un porcentaje de las entradas no preprocesadas. Dependiendo de como fueron transformadas éstas variables, ésto puede representar un gran o pequeño cambio en una o mas entradas reales a la red. En situaciones reales, es importante variar los datos de entrada sin entrenar y medir los cambios en la salida basada en ello.

PARA MAYOR INVESTIGACION

Como se describió anteriormente, hacer la incertidumbre dinámicamente mas grande produce resultados interesantes. Hasta cierto grado, un nivel de confianza puede ser derivado de que tan grande sea la incertidumbre. Una incertidumbre pequeña resultando en una gran sensibilidad significa una conclusión menos confiable que una gran incertidumbre con poca sensibilidad. En el programa, los efectos de incertidumbre negativa y positiva han sido promediados. Otra modificación podría ser reportar el cambio en cada dirección separadamente.

Poner un umbral de error mas alto en el entrenamiento resulta en fronteras borrosas (fuzzy). Esto tiende a hacer la red menos sensitiva a todas las entradas. También crea regiones anómalas adicionales, donde los chipotes (bumps) no se acoplan.El conjunto de datos está codificado dentro del programa, fué diseñado específicamente para ilustrar como trabaja el análisis de sensibilidad.

Muchos investigadores han inventado mejoras y extensiones para el algoritmo de backpropagation básico descrito arriba. La literatura es demasiado extensa para cubrirla aquí. Aún más, es demasiado rápido para una evaluación; algunas de éstas técnicas pueden probar ser fundamentales, otras pueden simplemente desaparecer. Para referencia, unos pocos de los desarrollos más promisorios son discutidos en ésta sección.

MEJORAS AL ALGORITMO BACK-PROPAGATION

Parker (1987) describe un método para mejorar la velocidad de convergencia del algoritmo de backpropagation. Llamado backpropagation de segundo orden, utiliza segundas derivadas para producir una estimación más exacta del cambio de peso correcto.

Parker ha demostrado que el algoritmo es óptimo en el sentido de que utilizando derivadas mas altas que de segundo orden no mejorarán el resultado estimado.

Los requerimientos computacionales se incrementan comparados con la backpropagation de primer orden, y mas resultados de pruebas son necesarios para demostrar que el costo adicional es justificado.

Stornetta y Huberman (1987) describen un método decepcionantemente simple para mejorar las características de las redes Backpropagation. Ellos apuntan que el rango dinámico de entradas y salidas convencional de 0 a 1 no es óptimo. Ya que la magnitud de un ajuste de peso $\Delta\ \omega_{pq,k}$ es proporcional al nivel de salida de la neurona que origina OUT_{pj}, un nivel de 0 resulta en no modificar el peso.

Con vectores de entrada binarios, la mitad de las entradas, en promedio, serán 0, y los pesos conectados a ellas no serán entrenados. La solución reside en cambiar el rango de entrada a 0.5 y añadir una direccional a la función compresora para modificar el rango de salida de la neurona a 0.5. La nueva función compresora es como sigue:

$$OUT = \underline{-0.5} + 1/(e^{-NET} + 1) \quad (3\text{-}13)$$

Los tiempos de convergencia fueron reducidos en un promedio de 30 a 50% con éstos cambios fácilmente implementados. esto es un ejemplo de las modificaciones prácticas que pueden traer mejoras sustanciales en el desempeño del algoritmo.

Sin embargo, en la red neuronal del presente trabajo, el desempeño no fué mejorado, en cambio, al ir llegando al límite de 0.001, el algoritmo empieza a dar resultados oscilantes. Este es un ejemplo de dónde no se mejora el back-propagation (Análisis de Sensibilidad)

Pineda (1988) y Almeida (1987) han descrito métodos para aplicar backpropagation a redes recurrentes, ésto es, redes cuyas salidas retroalimentan sus entradas. Ellos han demostrado que el aprendizaje puede ocurrir muy rápidamente en tales sistemas y el criterio de estabilidad es fácilmente satisfecho.

ALGUNAS APLICACIONES EXITOSAS

Backpropagation ha sido aplicado a una amplia variedad de aplicaciones de investigación; unas pocas son descritas para demostrar el poder de éste método.

NEC en Japón ha anunciado recientemente que ha aplicado Backpropagation a un nuevo sistema de reconocimiento óptico de caracteres, mejorando así la eficiencia en un 99%. Esta mejora fué lograda a través de una combinación de algoritmos convencionales con una red Backpropagation proveyéndo verificación adicional.

Sejnowski y Rosenberg (1987) [43] produjeron un éxito sensacional con NetTalk, un sistema que convierte inglés impreso en habla altamente inteligible. Su grabación del proceso de entrenamiento tiene un fuerte parecido con los sonidos de un niño en varias etapas del proceso de aprender a hablar.

Burr (1987) ha utilizado Backpropagation en reconocimiento por máquina del inglés manuscrito. Los caracteres son normalizados por tamaño, son colocados en una cuadrícula, y se hacen proyecciones de las líneas a través de los cuadrados de la cuadrícula. Estas proyecciones forman entonces las entradas a una red Backpropagation. El reporta exactitudes del 99.7% cuando se utiliza con un diccionario filtro

Cotrell, Munro y Zipser (1987) reportan una exitosa aplicación de compresión de imágenes en el cual las mismas fueron representadas con un bit por pixel, una mejora de ocho veces sobre los datos de entrada.

El algortimo de Back-Propagation ha sido probado conun número de problemas determinísticos tales como el problema de la O-exclusiva [40], en problemas relacionados a síntesis de voz y reconocimiento [43, 37, 8] y en problemas relacionados a reconocimiento de patrones visuales [40].

Se ha visto que funcione bien en muchos casos y para encontrar buenas soluciones a los problemas propuestos. Una demostración del poder de éste algoritmo fué hecha por Sejnowski [43]. Entrenó una red de dos capas tipo perceptrón, con 120 unidades escondidas y mas de 20,000 pesos para formar letras para reglas de transcripción fonética. La entrada de ésta red fué un código binario indicando ésas letras en una ventana deslizante de siete letras de largo que fué movida sobre una transcripción de texto hablado. La salida deseada era un código binario indicando la transcripción fonética de la letra en el centro de la ventana.

Después de 50 veces a través de un diálogo conteniendo 1024 palabras, el porcentaje de error de transcripción fué sólo 5%. Este porcentaje se incrementó a 22% para una continuación de ése diálogo que no fué usado para el entrenamiento.

CAPITULO 5. CONCLUSIONES

PRECAUCIONES

A pesar de las muchas aplicaciones exitosas de Backpropagation, no es una panacea. el proceso de entrenamiento es incierto y largo. Para problemas complejos, puede requerir días o semanas para entrenar la red, y puede que no se entrene por completo. El gran tiempo de entrenamiento puede ser el resultado de un tamaño de los pasos no óptimo. Las fallas de entrenamiento generalmente vienen de dos fuentes: parálisis de la red y mínimo local.

PARALISIS DE LA RED

En el entrenamiento de la red, los pesos pueden llegar a ajustarse a valores muy grandes. Esto puede forzar a todas o muchas de las neuronas a operar en grandes valores de **OUT**, en una region donde la derivada (tangente) de la función compresora es muy pequeña.(Sin Ganancia, es decir, sin inclinación en la gráfica)

Ya que el error regresado para el entrenamiento es proporcional a su derivada, el proceso de entrenamiento puede virtualmente detenerse. Existe muy poco entendimiento teórico de éste problema. Es comúnmente evitado por reducir el tamaño del paso , pero aumenta el tiempo de entrenamiento. Varios Heuritmos han sido empleados para prevenir la parálisis, o recobrarse de sus efectos, pero sólo pueden describirse como experimentales.

MINIMO LOCAL

Backpropagation emplea un tipo descendiente de gradiente; ésto es, sigue la tangente de la superficie de error hacia abajo, ajustando constantemente los pesos hasta un mínimo. La superficie de error de una red compleja es altamente convolucionada, llena de crestas, valles, doblajes, y canales en un espacio meta-dimensional. La red puede ser atrapada en un mínimo local (un valle poco profundo) donde existe gran cercanía al mínimo. Desde el punto de vista limitado de la red, todas las direcciones son hacia arriba, y no hay forma de escapar.

Los métodos estadísticos de entrenamiento pueden ayudarnos a evitar ésta trampa, pero tienden a ser lentos. Wasserman (1988a) ha propuesto un método que combina los métodos estadísticos de la máquina de Cauchy con el gradiente descendiente de Backpropagation para producir un sistema que encuentre mínimo global mientras que se retiene el más alto índice de entrenamiento del Backpropagation.

TAMAÑO DEL PASO

Una lectura cuidadosa de la prueba de convergencia de Rumelhart, Hinton y Williams (1986) [40] muestra que se asumen cambios de peso infinitesimales. Esto es claramente impráctico, ya que implica tiempo de entrenamiento infinito.

Es necesario seleccionar un tamaño de paso finito, y existe muy poca guía para la decisión exceptuando la experiencia. Si el paso es muy pequeño, la convergencia puede ser muy lenta; si es muy grande, pueden resultar parálisis o estabilidad continua. Wasserman (1986b) describe un algoritmo de tamaño de paso adaptativo que sirve para ajustar el paso automáticamente en el mismo proceso de entrenamiento.

INESTABILIDAD TEMPORAL

Si una red está aprendiendo a reconocer el alfabeto, no está bien que aprenda B si en el proceso olvida A. Un proceso es necesario para enseñar a la red a aprender un conjunto entero de entrenamiento sin interrumpir lo que ya ha aprendido. La prueba de convergencia de Rumelhart lleva ésto a cabo, pero requiere que la red muestre todos los vectores en el conjunto de entrenamiento antes de ajustar cualquier peso. Los cambios de peso necesarios deben ser acumulados sobre el conjunto entero, requiriendo así almacenaje adicional.

Después de un número de tales ciclos de entrenamiento, los pesos convergirán hacia un error mínimo. Este método puede no ser útil si la red enfrenta a un ambiente continuamente cambiante donde puede que nunca se repita al mismo vector de entrada dos veces. En éste caso, el proceso de entrenamiento puede nunca converger; puede perderse u oscilar ampliamente. En éste sentido Backpropagation falla en imitar los sistemas biológicos. El potencial más grande de las redes neuronales se basa en el procesamiento de alta velocidad que puede ser efectuado a través de implementaciones VLSI masivamente paralelas. Muchos grupos están actualmente explorando diferentes estrategias de implementación VLSI [31, 13, 42].

Los algoritmos para reconocimiento de habla y de imágenes puede ser desempeñada usando redes neuronales apoyando la aplicación potencial de cualquier hardware de red neuronal VLSI que sea desarrollado.

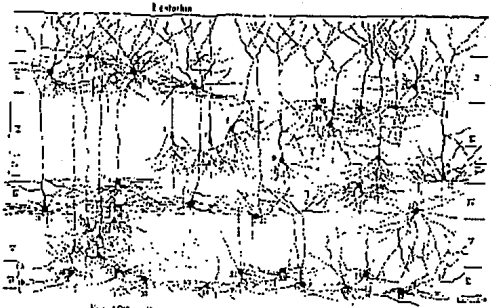


Fig. 189. Diagram of dendritic fields of pyramidal cells.
The numbers of the layers are shown at the side. Numerous individual cells may here be ignored. (*Lorente de Nó, from Fulton, "Physiology of Nervous System"* Courtesy of Oxford University Press.)

figura A-1: Red neuronal Biológica

APENDICE A. LA RED NEURONAL BIOLOGICA

Un cerebro humano contiene más de cien millones de millones de elementos de computación llamados neuronas.

Excediendo el número de estrellas en nuestra galaxia la vía láctea, éstas neuronas se comunican a través del cuerpo por medio de fibras nerviosas que son tal vez cien mil billones de conexiones llamadas sinápsis. Esta red de neuronas es responsable de todos los fenómenos que llamamos pensamientos, emociones y cognición, así como el desempeño de miles de funciones autonómicas y sensomotoras. La manera exacta en la cual es llevada a cabo ésta función, es muy poco entendida, pero mucho de la estructura biológica ha sido mapeada, y ciertas áreas funcionales están gradualmente siendo entendidas.

El cerebro también contiene una densa red de vasos de sangre que proveen de oxígeno y nutrientes a las neuronas y otras entidades cerebrales. Estos vasos están conectados al sistema circulatorio principal por un sistema de filtración altamente efectivo llamado la barrera de sangre del cerebro, un mecanismo de protección que aísla el cerebro de sustancias potencialmente tóxicas encontradas en el torrente sanguíneo.

El aislamiento es mantenido por la baja permeabilidad de los vasos sanguíneos del cerebro, y también por la cubierta de células de glucosa que rodean a las neuronas. Además de sus otras funciones, las células de glucosa proveen la envoltura estructural del cerebro. Virtualmente todo el volumen del cerebro no ocupado por los vasos sanguíneos y por las neuronas es ocupado por las células de glucosa.

La barrera de sangre del cerebro es esencial para la seguridad del cerebro, pero complica grandemente la administración de drogas terapéuticas. También frustra a los investigadores, quiénes quisieran observar los efectos de una amplia variedad de químicos en el funcionamiento del cerebro.

Sólo una pequeña lista de drogas diseñada para afectar el cerebro cruzará la barrera. Estas drogas consisten en pequeñas moléculas capaces de pasar a través de los pequeños poros en los vasos sanguíneos. Para afectar la función cerebral, deben pasar a través de las células de glucosa o ser soluble en su membrana. Pocas moléculas de interés satisfacen éstos requerimientos, las moléculas de muchas drogas terapéuticas son detenidas por ésta barrera.

El cerebro es el consumidor de energía más concentrado en el cuerpo. Conteniendo sólo el 2% de la masa corporal, utiliza el 20% del oxígeno del cuerpo. Aún cuando dormimos, el consumo de energía se mantiene estable. De hecho, existe evidencia de que puede incrementarse durante los periodos de sueño MOR (REM). Consumiendo sólo 20 watts, el cerebro es un órgano increíblemente eficiente en su consumo de energía.

Las computadoras, con sólo una pequeña fracción de la habilidad computacional del cerebro, consuma muchos miles de watts y requiere de elaboradas provisiones de enfriamiento para prevenir su autodestrucción térmica.

LA NEURONA

La neurona es el bloque de construcción fundamental del sistema nervioso. Es una célula similar a otras del cuerpo; sin embargo, ciertas especializaciones críticas le permiten desempeñar todas las funciones computacionales y de comunicación dentro del cerebro.

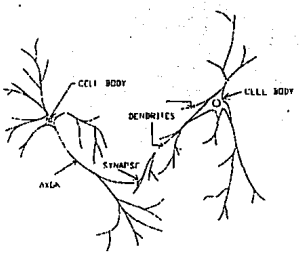


FIGURA A-1 Componentes de una Neurona

Como se muestra en la figura, la neurona consiste de tres secciones: el cuerpo de la célula, las dendritas, y el axón, cada una de ellas con funciones separadas pero complementarias.

Funcionalmente, las dendritas reciben las señales desde otras células en puntos de conexión llamados sinápsis. De aquí, las señales son pasadas al cuerpo de la célula, donde son esencialmente promediadas con otras señales. Si el promedio en un corto intervalo de tiempo es suficiente, la célula "dispara", produciendo un pulso hacia el axón que es pasado hacia células sucesivas, a través de sus respectivas sinápsis. A pesar de su aparente simplicidad, ésta función computacional cuenta para muchas de las actividades conocidas del cerebro. Subrayándo, sin embargo, que es un complejo sistema electroquímico.

EL CUERPO DE LA CELULA

Las neuronas en el cerebro del adulto no se regeneran, tienen un tiempo de vida. Esto significa que todos los componentes deben ser reemplazados contínuamente y los materiales renovados según se vaya necesitando. Muchas de ésas actividades de mantenimiento se llevan a cabo en el cuerpo de la célula, donde una verdadera planta química fabrica una amplia variedad de moléculas complejas.

Además, el cuerpo celular maneja la economía de energía de la neurona y regula un gran número de otras actividades dentro de la célula. La membrana externa del cuerpo de la neurona tiene la capacidad única de generar impulsos nerviosos, una función vital del sistema nervioso donde centra sus capacidades computacionales.

Cientos de tipos de neuronas han sido identificadas, cada uno con distinta forma de cuerpo celular, la cual mide normalmente de 5 a 100 micrones de diámetro. Una vez se pensó que eran meras variaciones aleatorias, pero éstos diferentes tipos morfológicos se ha encontrado que exhiben importantes especializaciones funcionales.

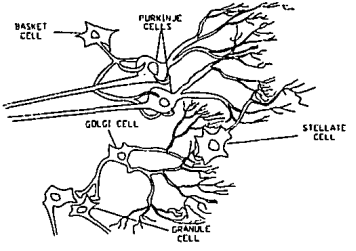


FIGURA A-2 Tipos de Neuronas

La identificación de las funciones de los diferentes tipos de células es actualmente un tópico de intensa investigación y es esencial para un entendimiento de los mecanismos de proceso en el cerebro.

DENDRITAS

Muchas de las señales de entrada de otras neuronas entran a la célula por medio de las dendritas, una estructura arborescente que emana del cuerpo de la célula. En las dendritas están las conexiones sinápticas donde las señales son recibidas, usualmente desde otros axones.

Además, existen un número significante de conexiones sinápticas de axón a axón, de axón a cuerpo celular, y de dendrita a dendrita; la función de éstas conexiones es poco entendida, pero es demasiado notable para ser insignificante.

A diferencia de los circuitos eléctricos, normalmente no existe conexión física o eléctrica en la sinápsis. En vez de ello, una pequeña separación llamada fisura sináptica separa la dendrita del axón transmisor. Sustancias químicas especializadas liberadas por el axón dentro de la fisura sináptica se difunden en la dendrita. Estas sustancias llamadas neurotransmisores, llegan a receptores específicos que existen en las dendritas y entran en el cuerpo celular.

Más de treinta neurotransmisores han sido identificados. Algunos son excitatorios y hacen que la célula "dispare" y produzca un pulso de salida. Otros son inhibitorios y tienden a suprimir tales pulsos. El cuerpo celular combina las señales recibidas en sus dendritas y, si su señal resultante se encuentra arriba de cierto umbral, un pulso es producido y se propaga hacia el axón hacia otras neuronas.

EL AXON

Un axón puede ser tan corto como 0.1 milímetro, o puede exceder 1 metro en longitud, extendiéndose hacia una parte enteramente diferente del cuerpo. Cerca de su final, el axón tiene múltiples ramificaciones, cada una terminando en una sinápsis, donde la señal es transmitida a la otra neurona a través de una dendrita, o en algunos casos, directamente al cuerpo celular.

En ésta forma, una sola neurona puede generar un pulso que activará o inhibirá cientos de miles de otras neuronas, cada una de las cuales puede a su vez (a través de sus dendritas) ser activada por cientos de miles de otras neuronas.

Así, es éste alto grado de conectividad más que la complejidad funcional de la neurona en sí misma lo que la de a la neurona su poder computacional.

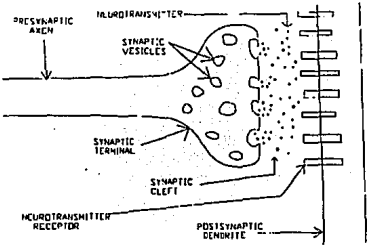


FIGURA A-3 Sinápsis

La conexión sináptica que termina una rama del axón es una expansión pequeña y bulbosa conteniendo estructuras esféricas llamadas vesículas sinápticas, cada una de las cuales contiene un gran número de moléculas neurotransmisoras. Cuando un impulso nervioso llega al axón, algunas de ésas vesículas liberan sus contenidos dentro de la fisura sináptica, iniciándo así el proceso de comunicación interneuronal.

Además del comportamiento de todo-o-nada las neuronas débilmente estimuladas también transmiten señales electroquímicas a través de su interior con una respuesta graduada. De naturaleza local, éstas señales terminan rápidamente con la distancia a menos que sean reforzadas. La naturaleza hace uso de ésta característica celular en el sistema nervioso periférico mediante el envolver los axones con células de Swann, formando así una cubierta aislante conocida como mielina.

La cubierta de mielina es interrumpida cada milímetro por espacios llamados nodos de Ranvier. Los impulsos nerviosos pasándo hacia un axón saltan de nodo a nodo, propagándose por un modo de respuesta graduada entre saltos. De ésta forma, el axón no necesita gastar energía para mantener sus gradientes químicos a través de su longitud. Solamente en los nodos expuestos es regenerada la acción potencial.

Además de ahorrar energía, otras funciones de ésta cubierta han sido descubiertas. Por ejemplo, las fibras nerviosas mielinizadas transmiten las señales más rápidamente. Muchas enfermedades han sido atribuídas a la deteriorización de éste aislante, y se sospecha que contribuye a otras.

ORIGEN DEL IMPULSO NERVIOSO

El estudio de los cambios de potencial que presentan las membranas excitables ha dominado el campo de la fisiología en los últimos años. En la sinápsis, la liberación del transmisor químico está también estrechamente vinculada a la despolarización que presentan las terminales sinápticas.

Es muy antigua la idea de que son los iones en su paso por la membrana los que generan la actividad de la fibra nerviosa. El cambio súbito y reversible en el potencial de la membrana que se produce cuando se la estimula adecuadamente se denomina ***potencial de acción***, y a las membranas que son capaces de generarlo se les denomina excitables; así pues, se consideran células excitables la neurona, los diversos tipos de células musculares y algunas células secretorias.

LA MEMBRANA CELULAR

En el sistema nervioso la información se transmite con un doble carácter: por un lado, es esencialmente eléctrica cuando las señales viajan en una misma neurona a lo largo de su axón, y es esencialmente química, a través de neurotransmisores cuando la señal viaja de una neurona a otra. La señal eléctrica, el impulso nervioso, consiste básicamente en un cambio reversible y breve en el potencial de la membrana denominado potencial de acción, descrito en el párrafo anterior.

Cuando la neurona no presenta impulsos, presenta también un potencial de membrana, que se puede medir con un microelectrodo; el voltaje en el interior celular con respecto al exterior es de algunas decenas de milivoltios, siendo potencial negativo: a éste potencial se le denomina ***potencial de reposo.***

La existencia del potencial de reposo en fibras musculares se prupuso hace muchísimos años, en la primera mitad del siglo XIX, cuando se midió la corriente de lesión con un galvanómetro. Esta corriente se produce entre una zona intacta del músculo y una región que ha sido lesionada; la lesión permite hacer a la membrana totalmente permeable, y exponer el inteior celular. En el siglo XIX Boys-Reymond demostró que la corriente de lesión fluye de la parte dañada a la parte intacta y concluyó que el interior del músculo está cargado negativamente.

La comunicación en el cerebro es de dos tipos: señales químicas a través de las sinápsis, y señales eléctricas dentro de la neurona. Es la maravillosa y compleja acción de la membrana la que crea la habilidad de la célula para producir ambos tipos de señales.

La membrana celular es de alrededor de cinco nanómetros de ancho y consiste de dos capas de moléculas de lípidos. En la membrana existen varios tipos de proteínas específicas que caen en cinco clases: bombas, canales, receptores, enzimas y proteínas estructurales.

Las bombas mueven activamente los iones a través de la membrana celular para mantener los gradientes de concentración. Los canales pasan iones selectivamente y controlan su flujo a través de la membrana. Algunos canales son abiertos o cerrados por el potencial eléctrico prevalente en la membrana, proveiendo así un medio rápido y sensitivo de modular gradientes iónicos. Otros tipos de canales son químicamente controlados, cambiando su permeabilidad hasta recibir mensajeros químicos.

Los receptores son proteínas que reconocen y ajustan muchos tipos de moléculas en el ambiente celular con gran especifidad. Las enzimas sobre o cerca de la membrana aceleran una variedad de reacciones químicas. Las proteínas estructurales interconectan células y ayudan a mantener la estructura de la célula en sí misma.

La concentración interna de sodio de la célula es diez veces mas baja que en los alrededores y su concentración de potasio es diez veces mas alta. Estas concentraciones tienden a igualarse a través de fisuras en la célula debidas a poros en la membrana. Para mantener la concentración necesaria, una proteína de la membrana, llamada bomba de sodio, contínuamente saca sodio de la célula y mete potasio a ella. cada bomba mueve doscientos átomos de sodio y ciento treinta iones de potasio por segundo.

Una neurona puede tener millones de tales bombas,moviendo cientos de millones de iones adentro y afuera cada segundo. La concentración de potasio dentro de la célula es además incrementada por la presencia de un gran número de canales de potasio permanentemente abiertos; ésto es, existen proteínas que pasan iones de potasio dentro de la célula, pero inhiben el paso del sodio. La combinación de éstos dos mecanismos es reponsable de crear y mantener es dinámico equilibrio químico que constituye el estado de descanso de la célula.

El gradiente de concentración iónica a través de la membrana celular causa que el interior de la célula asuma un potencial eléctrico de -70 milivoltios relativos a su entorno. Para que la célula se dispare (produciendo una acción de potencial), las entradas sinápticas deben reducir éste nivel aproximadamente a - 50 milivoltios. Cuando ésto ocurre, los flujos de sodio y de potasio es invertido repentínamente; en un milisegundo el interior de la célula se vuelve 50 milivoltios positivo relativo a su entorno.

Esta cambio de polaridad se extiende rápidamente a través de la célula, causando que el impulso nervioso se propague hacia la longitud del axón hacia sus conexiones presinápticas.

Cuando la señal llega a la terminal de un axón, los canales de calcio controlados por voltaje son abiertos. Esto dispara la liberación de moléculas de neurotransmisores dentro de la fisura sináptica y el proceso continúa hacia las otras células. Después de generar una acción potencial, la célula entra en una periodo de descanso de varios milisegundos, durante los cuales regresa a su potencial de descanso en preparación para la generación de otro impulso.

Examinando este proceso en mayor detalle, la recepción inicial de moléculas de neurotransmisor disminuye el potencial interno de la célula de -70 milivoltios a -50 milivoltios. En éste punto, los canales controlados por voltaje de sodio son abiertos, permitiendo al sodio entrar a la célula. Esto reduce aún más el potencial, aumenta el flujo de sodio, y crea un proceso auto-reforzante que rápidamente se propaga hacia regiones adyacentes, invirtiendo el potencial de la célula local de negativo a positivo.

Un poco después de abrirse, los canales de sodio se cierran y los canales de potasio se abren. Esto permite al potasio fluir fuera de la célula, y el potencial interno de -70 milivoltios es restablecido. Este cambio rápido de voltaje constituye el potencial de acción, el cuál se propaga rápidamente a través del axón como una tira de dominós cayendo.

Los canales de sodio y de potasio responden a los potenciales de las células; de aquí, se dice que son accionados por voltaje. Otro tipo de canal es accionado químicamente. Se abre solo cuando una molécula específica de neurotransmisor se une a un receptor, y es insensible al voltaje. Tales canales se encuentran en las conexiones postsinápticas en las dendritas y son responsables de la respuesta de la neurona a varios tipos de moléculas neurotransmisoras. La proteína de acetilcolina que combinada con un receptor es uno de tales canales accionados químicamente.

Cuando un paquete de moléculas de acetilcolina es liberado dentro de la fisura sináptica, se difunde en los receptores de acetilcolina que se encuentran en la membrana postsináptica. Estos receptores (los cuales también son canales, se abren, permitiendo paso libre tanto al sodio como al potasio a través de la membrana. Esto produce un reducción local breve en el potencial negativo interno de la célula (constituyendo un pulso positivo).

A causa de que son cortos y pequeños, muchos de tales aberturas de canales son requeridas para causar que la célula produzca un potencial de acción aunque cada uno produce una respuesta graduada.

Los canales/receptores de acetilcolina pasan tanto al sodio como al potasio, produciendo así pulso positivos. Tales pulsos son excitatorios, ya que contribuyen a la producción de un potencial de acción. Otros canales activados químicamente pasan solo iones de potasio fuera de la célula, y producen pulsos negativos; son inhibitorios, ya que tienden a impedir el disparo de la célula.

El ácido Gamma-aminobutírico (GABA) es uno de los neurotransmisores inhibitorios más comunes. Encontrado casi exclusivamente en el cerebro y médula espinal, se une a un receptor con un canal que pasa selectivamente iones de cloro. Hasta la entrada, éstos iones aumentan el potencial negativo de la célula, inhibiendo así el disparo de la misma.

La deficiencia de GABA ha sido asociada con la corea de Huntington, un síndrome inherente neurológico que causa movimientos musculares incontrolados. Desafortunadamente, la barrera de sangre del cerebro tiene un mecanismo que previene del aumento de GABA, y no ha sido encontrado otro tratamiento.

Parece probable que otros desórdenes neurológicos y enfermedades mentales se deban a defectos similares en los neurotransmisores o sus precursores químicos.

La razón de disparo de una neurona está determinada por el efecto acumulativo de un gran número de entradas excitatorias e inhibitorias, promediado por el cuerpo de la célula en un corto intervalo de tiempo. La recepción de moléculas neurotransmisoras excitatorias incrementará la razón de disparo; un número mas pequeño, o una mezcla con entradas inhibitorias reducirá la razón de disparo. De ésta forma, la señal neuronal está en razón de pulso, es decir en frecuencia modulada (FM). Este método de modulación ampliamente utilizado en ingeniería de comunicaciones, ha probado tener ventajas significantes en rechazo de ruido sobre otras técnicas.

La investigación ha abierto el camino a una gran complejidad bioquímica en el cerebro humano. Por ejemplo, más de treinta sustancias se piensa que son neurotransmisores, y existen un gran número de receptores con varios modos de respuesta. Aún más, la acción de una molécula de un neurotransmisor en particular depende del tipo de receptor en la membrana postsináptica; el mismo neurotransmisor puede ser excitatorio en una sinápsis e inhibitorio en otra.

También un sistema de "segundo mensajero" trabaja en la célula, donde al recepción de un neurotransmisor dispara la producción de un gran número de moléculas cíclicas de adenosin trifosfato, produciendo así una respuesta fisiológica grandemente amplificada.

Los investigadores siempre esperan encontrar un patrón simple que unifique observaciones complejas y diversas. De todos modos, no ha ocurrido así con los estudios neurobiológicos. Muchos descubrimientos han expuesto mas ignorancia de la que han eliminado.

Un resultado así de la investigación neurobiológica ha sido una rápida proliferación en el número y tipos de actividades electroquímicas reconocidas como trabajo en el cerebro; la tarea consiste en combinarlas dentro de un modelo coherente y funcional.

APENDICE B. OTROS ALGORITMOS

MAPAS AUTO ORGANIZABLES DE KOHONEN

Un principio importante de organización de caminos sensoriales del cerebro es que la situación de las neuronas es ordenada y frecuentemente refleja alguna característica física de los estímulos externos siendo sentidos [21]. Por ejemplo, en cada nivel del camino auditivo, células nerviosas y fibras son posicionadas anatómicamente en relación con la frecuencia la cual elige la más grande respuesta en cada neurona.

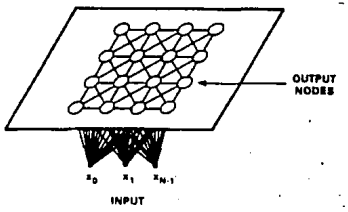


FIGURA B-1 Arreglo bidimensional de nodos de salida utilizados para formar mapas de características. Cada entrada es conectada a cada nodo de salida via una conección de peso variable.

Esta organización fonotópica en el camino auditivo se extiende hasta la corteza cerebral [33, 21]. Aunque mucha de la organización de bajo nivel está predeterminada genéticamente, es similar a alguna de la organización a niveles más altos que es creada durante el proceso de aprendizaje por algoritmos los cuales promueven autoorganización.

Kohonen [22] presenta uno de tales algoritmos el cual produce lo que él llama mapas de autoorganización similares a aquellos que ocurren en el cerebro.

El algoritmo de Kohonen crea un cuantizador de vectores mediante el ajustar pesos de nodos comunes de entrada a M nodos de salida arreglados en una trama bidimensional como se muestra en la figura b-1 . Los nodos de salida están intensivamente interconectados con muchas conexiones locales. Vectores de entrada de valores contínuos son presentados secuencialmente en el tiempo sin especificar la salida deseada.

Después de que suficientes vectores de entrada han sido presentados, los pesos especificarán centros de vectores o de racimos que muestrean el espacio de entrada tal que la función de densidad de punto del centro del vector tiende a aproximar la función de densidad probabilística de los vectores de entrada [22].

Además, los pesos serán organizados de tal manera que los nodos topológicamente cercanos son sensitivos a entradas que son físicamente similares. Los nodos de salida serán así ordenados en una manera natural. Esto puede ser importante en sistemas complejos con muchas capas de procesamiento porque puede reducir longitudes de conexiones entre las capas.

El algoritmo que forma mapas de características requiere una vecindad a ser definida alrededor de cada nodo como se muestra en la figura B-2. Esta vecindad disminuye lentamente en tamaño con el tiempo, como se muestra. El algoritmo de Kohonen es descrito en la caja 6. Los pesos entre los nodos de entrada y de salida es puesto inicialmente en pequeños valores aleatorios y una entrada es presentada. La distancia entre la entrada y todos los nodos es computada como se muestra. Si los vectores de peso son normalizados para tener distancia constante (la suma de los pesos cuadrados de todas las entradas a cada una de las salidas son idénticas) entonces el nodo con la mínima distancia Euclidiana puede ser encontrada usando la red de la figura 17 para formar el producto punto de la entrada y de los pesos.

CAJA 6: Algoritmo de Kohonen

1: Asignar pesos	Inicializa los pesos de los N nodos de entrada para los M nodos de salida. Para pequeños valores aleatorios. Establece el radio inicial de la vecindad topológica.
2: Presenta nueva entrada	
3: Calcula la distancia para todos los nodos	Calcula la distancia entre nodo de entrada y nodo de salida utilizando la fórmula: $d_j=\sum_{i=0}^{N-1}(x_i(t) - \omega_{ij}(t))^2$ donde j es el nodo de salida, xi(t) es la entrada del i-ésimo nodo en el tiempo t y $\omega_{ij}(t)$. es el peso de ése nodo al nodo de salida j en el tiempo t
4: Seleccionar los nodos de salida con la distancia mínima	Seleccionar el nodo de salida j* señelado con el mínimo d_j
5: Actualizar los pesos del nodo j* y su vecindad. Los pesos son actualizados para el nodo j* y todos los nodos en su vecindad definida por **NEj*(t).**	Los nuevos pesos son: $\omega_{ij}(t+1)=\omega_{ij}(t)+\mu(t)(x_i(t) - \omega_{ij}(t))$ Para $j \in NEj^*(t)$ $0\le i\le N-1$ El término $\mu(t)$ es un término de ganancia en el intervalo de 0 a 1, que decrece con el tiempo.
6: ir al paso 2	

La selección requerida en el paso 4 entonces se vuelve un problema de encontrar el nodo con un valor máximo.

Este nodo puede ser seleccionado usando inhibición lateral extensiva como en la MAXNET de arriba de la figura 6. Una vez que es seleccionado éste nodo, los pesos a él y a otros nodos en ésta vecindad son modificados para hacer a éstos nodos más responsivos a la entrada actual.

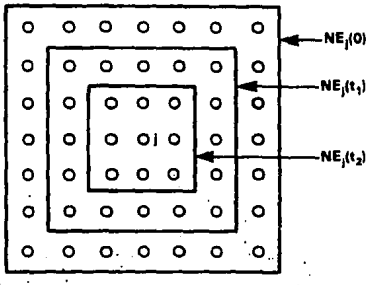


FIGURA B-2 Vecindades topológicas a diferentes tiempos de creación de mapas de características. $NE_j(0)$ es el conjunto de nodos considerados a estar en la vecindad del nodo j en el tiempo t. La vecindad empieza grande y lentamente disminuye en tamaño con el tiempo. En éste ejemplo, $0 < t1 < t2$.

Este proceso es repetido para entradas posteriores. Los pesos eventualmente convergen y son fijados después que el término de ganancia en el paso 5 es reducido a cero.

Un ejemplo del comportamiento de éste algoritmo es presentado en la figura B-3. Los pesos para 100 nodos de salida son dibujados en ésos seis subdibujos donde existen dos entradas aleatorias uniformemente distribuidas sobre la region encerrada por las áreas delimitadas. Las intersecciones de las lineas en ésos dibujos especifican pesos para un nodo de salida.

Los pesos la entrada x_0 son especificados por la posición sobre el eje horizontal y los pesos de la entrada x_1 es especificada por la posición sobre el eje vertical. Las líneas conectan valores pesados para nodos que son topológicamente los vecinos más cercanos. Los pesos empiezan en el tiempo cero arracimados en el centro del dibujo. Entonces se expanden gradualmente en una forma ordenada hasta que su punto de densidad se aproxima a la distribución uniforme de las muestras de entrada.

En éste ejemplo, el término de ganancia en el paso 5 de la caja 6 fué una función Gaussiana de la distancia al nodo seleccionado en el paso 4 con una anchura que decrece con el tiempo.

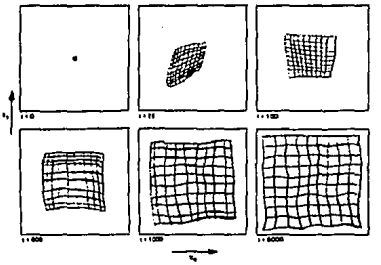


FIGURA B-3 Pesos a 100 nodos de entrada desde dos nodos de entrada, en la formación de un mapa de características. El eje horizontal representa el valor de el peso de la entrada x0 y el eje vertical representa el valor del peso de la entrada x1. Las intersecciones de las líneas especifican los dos pesos para cada nodo. Las líneas conectan pesos para nodos que son los vecinos mas cercanos. Una red ordenada que los nodos topológicamente cercanos son físicamente similares. Las entradas fueron aleatorias, independientes y uniformemente distribuidas sobre el área mostrada.

Kohonen [22] presenta muchos otros ejemplos y pruebas relativas a éste algoritmo. EL también ha demostrado cómo el algoritmo puede ser usado en un reconocedor del habla como cuantizador de vectores [23]. A diferencia del clasificador Grossberg-Carpenter, éste algoritmo puede desempeñarse relativamente bien con ruido, porque el número de clases es fijo, los pesos se adaptan lentamente, y la adaptación se detiene después del entrenamiento.

Este algoritmo es así un cuantizador vectorial secuencial viable cuando el número de racimos deseados puede ser especificado antes de usarlos y la cantidad de datos de entrenamiento es grande con relación al número de racimos deseados. Es similar al algoritmo K-media en éste respecto.

Los resultados, sin embargo, pueden depender del orden de presentación de los datos de entrada para pequeñas cantidades de datos de entrenamiento.

APENDICE C. ANALISIS DEL ALGORITMO BACK - PROPAGATION

El desempeño generalmente bueno encontrado para el algoritmo de Back-Propagation es sorprendente considerando que es una técnica de búsqueda de gradiente que puede encontrar un mínimo local en la función costeada LMS en vez del mínimo global.

Sugerencias para mejorar el desempeño y reducir la ocurrencia de mínimo local incluye permitir unidades escondidas extras, y hacer muchas corridas de entrenamiento empezando con diferentes conjuntos de pesos aleatorios.

Cuando es utilizado con problemas de clasificación, el número de nodos puede ser puesto usando las consideraciones descritas arriba. El problema del mínimo local en éste caso corresponde a arracimar dos o más puntos de desunión en uno sólo. Esto puede ser minimizado por usar múltiples inicios con diferentes pesos aleatorios y baja ganancia para adaptar los pesos. Una dificultad notada con el algoritmo de Back-Propagation es que en muchos casos el número de presentaciones de datos de entrenamiento requeridos para la convergencia ha sido grande [más de 100 pasadas de todos los datos de entrenamiento].

Aunque un número de mas complejos algoritmos de adaptación han sido propuestos para acelerar la convergencia [35], parece improbable que las regiones de decisión complejas formadas por perceptrones multicapa puedan ser generadas en pocas pruebas cuando las regiones de clases están desconectadas.

APENDICE D. REFERENCIAS

[1] Y.S. Abu-Mostafa and D.Pslatis, "Optical Neural Computers" Scientific American 256, 88-95, March 1987.

[2] E.B. Baum, J. Moody, and F. Wilczek, "Internal Representations for Associative Memory", NSF-ITP-86-138 Institute of Theoretical Phisycs, University of California, Santa Barbara, California, 1986.

[3] G.A. Carpenter, and S. Grossberg, "Neural Dynamics of Category Learning and Recognition: Attention, Memory consolidation and Amnesia" in J. Davis R. Newburgh, and E. Wegman (Eds.) Brain Structure, Learning and Memory, AAAS Symposium Series, 1986.

[4] M.A. Cohen, and S. Grossberg, "Absolute Stability of Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks", IEEE Trans. Syst. Man Cybern. SMC-13, 815-826, 1983.

[5] B. Delgutte, "Speech Coding in the Auditory Nerve: II. Processing Schemes for Vowel-Like Sounds"J. Acoustic Soc. Am. 75, 879-886, 1984.

[6] J.S. Denker, AIP Conference Proceedings 151, Neural Networks for Computing, Snowbird Utah, AIP, 1986.

[7] R.O. Duda and P.E. Hart, "Pattern Classification and Scene Analysis", John Wiley & Sons, New York 1973.

[8] J.L.Elman and D.Sipzer. "Learning the Hidden Structure of Speech", Institute for Congnitive Science, University of California at San Diego, ICS Report 8701, Feb, 1987.

[9] J.A. Feldman and D.H. Ballard, "Connectionist Models and Their Properties", Cognitive Science, Vol. 6, 205-254, 1982.

[10] R.G. Gallager Information "Theory and Reliable Communication", John Wiley & Sons, New York (1968).

[11] O. Ghitza, "Robustness against Noise: The Role of Timing-Synchronny Measurement" In Proceedings International Conference on Acoustics Speech and Signal Prossecing, ICASSP-87, Dallas, Texas, April 1987.

[12] B. Gold, "Hopfield Model Applied to Vowel and Consonant discrimination", MIT Lincoln Laboratory Technical Report, TRT-747, AD-A169742, june 1986.

[13] H.P. Graf, L.D. Jackel, R.E. Howard, B. Straughn, J.S. Denker, W. Hubbard, D.M. Tennant and D. Shwartz, "VLSI Implementation of A Neural Network Memory With Several Hundreds of Neurons", in J.S. Denker (Ed.) AIP Conference Proceedings 151, Neural Network for Computing, Snowbird Utah, AIP, 1986.

[14] P.M.Grant and J.P.Sage, "A Comparison of Neural Network and Matched Filter Processing for Detecting Lines in Images", in J.S. Denker (Ed.) AIP Conference Proceedings 151, Neural Network for Computing, Snowbird Utah, AIP, 1986.

[15] S. Grossberg, "The Adaptive Brain I: Cognition, Learning, Reinforcement, and Rythm", and "The Adaptive Brain II: Vision, Speech, Language, and Motor Control", Elsevier/North-Holland, Amsterdam 1986.

[16] J.A. Hartigan, "Clustering Algorithms", John Wiley & Sons, New York 1975.

[17] D.O. Hebb, "The Organization of Behavior", John Wiley & Sons, New York 1949.

[18] J.J. Hopfield, "Neural Networks and Phisycal Systems with Emergent Collective Computational Abilities", Proc. Natl. Acad. Sci. USA Vol 79, 2554-2558, April 1982.

[19] J.J. Hopfield, "Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons", Proc. Natl. Acad. Sci. USA, Vol. 81, 3088-3092, May 1984.

[20] J.J. Hopfield and D.W. Tank, "Computing with Neural Circuits: A Model", Science, Vol. 233, 625-633, August 1986.

[21] E.R. Kandel and J.H. Schwartz, "Principles of Neural Science", Elsevier, New YOrk (1985).

[22] T. Kohonen, "Self-Organization and Associative Memory", Springer-Verlag, Berlin (1984).

[23] T. Kohonen, K. Masisara and T. Saramaki, "Phonotopic Maps-Insightful Representation of Phonological Features of Speech Representation", Proceedings IEEE 7th Inter. Conf. of Pattern Recognition, Montreal, Canada, 1984.

[24] F.L. Lewis, "Optimal Estimation", John Wiley & Sons, New York 1986.

[25] R.P. Lippmann, B. Gold, and M.L. Malpass, "A Comparison of Hamming and Hopfield Neural Nets for Pattern Classification", MIT Lincoln Laboratory Technical Report, TR-769, to be published.

[26] G.G. Lorentz, "The 13th problem of Hilbert", in F.E. Browder (Ed.) Mthematical Developments Arising from Hilbert Problems, American Mathematical Society, Providence, R.I. 1976.

[27] R.F. Lyon and E.P. Loeb, "Isolated Digital Recognition Experiments with a Cochlear Model" in Proceedings International Conference on Acoustics Speech and signal Processing, ICASSP-87, Dallas, Texas, April 1987.

[28] J. Makhoul, S. Roucos, and H. Gish, "Vector Quantization in Speech Coding", IEEE Proceedings, 3, 1551-1558, nov. 1985.

[29] T. Martin, "Acoustic Recognition of a Limited Vocabulary in Continuos Speech", Ph.D. Thesis, Dept Electrical Engineering, Univ. Pennsylvania, 1970.

[30] W.S. McCulloch and W. Pitts, "A Logical Calculus of the Ideas Imminent in Nervous Activity", Bulletin of MAthematical Biophysics, 5, 115-133, 1943.

[31] C.A. Mead, "Analog VLSI and Neural Systems", Course Notes, Computer Science Dept., California Institute of Technology, 1986.

[32] M. Minsky, and S. Papert, "Perceptrons: an Introduction to Computational Geometry", MIT Press 1969.

[33] A.R. Moller, "Auditory Physiology", Academic Press, New York 1983.

[34] P. Mueller and J. Lazzaro, "A Machine for Neural Computation of Acoustical Patterns with Application to Real-Time Speech-Recognition", in J.S. Denker (Ed.) AIP Conference Proceedings 151, Neural Networks for Computing, Snowbird Utah, AIP 1986.

[35] D.B. Parker, "A Comparison of Algorithms for Neuron-Like Cells", in J.S. Denker (Ed.) AIP Conference Proceedings 151, Neural Networks for Computing, Snowbird Utah, AIP 1986.

[36] T. Parsons, "Voice and Speech Processing", McGraw-Hill, New York 1986.

[37] S.M. Peeling, R.K. Moore, and M.J. Thomlinson, "The Multi-Layer Perceptron as a Tool for Speech Pattern Proccesing Research", In Proc. IoA Autumn Conf. on Speech and Hearing, 1986.

[38] T.E. Posch, "Models of the Generation and Processing of Signals by Nerve Cells: A Categorically Indexed Abridged Bibliography", USCEE Report 290, August 1968.

[39] R. Rosenblatt, "Principles of Neurodynamics", New York, Spartan Bookss, 1959.

[40] D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning Internal Representation by Error Propagation" in D.E. Rumelhart & J.L. McClelland (Eds.) Parallel Distributed Processing: Explorations in the Microstructure of Recognition, Vol. 1: Foundations. MIT Press, 1986.

[41] D.E. Rumelhart, and J.L. Mcclelland, "Parallel Distributed Processing: Explorations in the Microstructure of Cognition", MIT Press, 1986.

[42] J.P. Sage, K. Thompson, and R.S. Withers, "An Artificial Neural Network Integrated Circuit Based on MNOS/CD Principles", in J.S. Denker (Ed.) AIP Conference Proceedings 151, Neural Networks for Computing, Snowbird Utah, AIP 1986.

[43] T. Sejnowski and C.R. Rosenberg, "NETtalk: a Parallel Network That Learns to Read Aloud", John Hopkins Univ. Technical Report JHU/EECS-8601, 1986.

[44] S. Seneff, "A Computational Model to Speech Recognition Research", in Proceedings International Conference on Acoustics Speech and Signal Processing, ICASSP-86, 4, 97.8.1-37.8.4, 1986.

[45] D.W. Tank, and J.J. Hopfield, "Simple Neural Optimization Networks: An A/D Converter, Signal Decision Circuit and a Linear Programming Circuit" IEEE Trans. Systems CAS-33, 533-541, 1986.

[46] D.J. Wallace, "Memory and Learning in a Class of Neural Models", in B. Bunk and K.H. Mutter (Eds.) Proceedings of the Workshop on Lattice Gauge Theory, Wuppertal, 1985, Plenum 1986.

[47] Widrow, and, M.E. Hoff, "Adaptive Switching Circuits", 1960 IRE WESCON Conv. record, Part 4, 96-104, August 1960.

[48] B. Widrow and S.D. Stearns, "Adaptive Signal Processing", Prentice-Hall, New Jersey (1985).

glosario

Inteligencia Artificial (AI)

Inteligencia Artificial se refiere a un amplio rango de aplicaciones computacionales que recuerdan al comportamiento e inteligencia humanos. Por ejemplo, máquinas y robots con capacidades sensoriales que reconocen sonido, imágenes y texturas caen dentro de éste rango, asimismo el reconocimiento de patrones de voz, entendimiento del lenguaje natural, y la traducción de lenguajes extranjeros.

Los Sistemas Expertos son una de las grandes categorías de la Inteligencia Artificial. Contienen una base de conocimiento acerca de un área de experiencia y son utilizados para asisitir en el desempeño de tareas tales como diagnóstico médico, reparación de equipos, ejecución de horarios y planeación financiera.

Uno de los mas significantes beneficios de la Inteligencia Artificial será el reconocimiento de voz y la comprensión del lenguaje natural. Antes de fin de siglo se podrán hacer preguntas verbales a las computadoras. Cuando éso ocurra, todo lo que sabemos y pensamos acerca de las computadoras cambiará. Los sistemas operativos del futuro le preguntarán que ayuda necesita y automáticamente efectuará las operaciones adecuadas, también creadas con técnicas de inteigencia artificial, para ayudarle a resolver su problema.

Como todo lo sobresaliente en ésta industria, se abusará de la Inteligencia Artificial, que será referida en toda clase de productos. Sin embargo, la prueba máxima de un sistema con Inteligencia Artificial fué definida hace años por el pionero inglés de la computación, Alan Turing, quien dijo: "Una máquina tiene Inteligencia Artificial cuando no existe diferencia discernible entre la conversación generada por la máquina y la generada por una persona inteligente".

Nota: El término *Inteligencia* por si mismo se refiere a la capacidad de procesar. Por lo tanto, TODA computadora es inteligente, ya que puede seguir la instrucciones de un programa. La Inteligencia Artificial implica inteligencia semejante a la humana.

Autómata

La teoría de *autómatas* relaciona a las operaciones y aplicaciones de máquinas automáticas a conceptos de comportamiento humano.

Binario

Binario significa "dos" y es el principio fundamental detrás de las computadoras digitales. Toda entrada a la computadora es convertida en números binarios hechos de los números 0 y 1, llamados bits (BInary digiTS). Poe ejemplo, cuando se presiona la tecla "A" en el teclado de la computadora, el teclado genera y transmite el número 01000001 a la memoria de la computadora como una serie de pulsos. Los bits 1 son transmitidos como pulsos de electricidad (típicamente 5 volts); los bits 0 son transmitidos como no pulsos (típicamente menos de 2.5 volts). Los 1's y 0's son almacenados en la memoria de la computadora como una serie de cargas (y no cargas).

Los circuitos electrónicos que procesan los números binarios son asimismo binarios en concepto. Son hechos de switches on/off (transistores) que son abiertos y cerrados eléctricamente. El estado actual de un transistor afecta la operación del siguiente, y así sucesivamente. Estos transistores se abren y cierran en nanosegundos y picosegundos (mil millonésimas y billonésimas de segundo).

La capacidad de trabajo de una computadora se basa en su almacenamiento (memoria y disco), y velocidad interna de transmisión y ejecución de datos.

Biónica

Un dispositivo *biónico* es cualquier máquina cuyas bases se encuentran en patrones humanos o de la naturaleza, por ejemplo, robots. La biónica también se refiere a los dispositivos artificiales implantados en humanos que reemplazan o extienden las funciones humanas normales.

Sistema Experto

Un *Sistema Experto* es una aplicación de inteligencia artificial que utiliza una base de conocimiento de expertos humanos para ayudar a resolver problemas. El grado de resolución del problema se basa en la calidad de los datos y reglas obtenidos del experto humano. Los Sistemas Expertos son diseñados para desempeñar tareas a nivel de experto humano. En la práctica, se desempeñan bien tanto debajo como encima del nivel del experto humano.

El sistema experto deriva sus respuestas por consultar la base de conocimiento mediante un ingenio de inferencia, un programa que interactúa con el usuario y procesa los resultados de las reglas y los datos en la base de conocimiento.

Los Sistemas Expertos son utilizados en aplicaciones tales como diagnósticos médicos, reparación de equipo, palneación financiera y de seguros, control de producción, entrenamiento, etcétera.

Red Neuronal

Red Neuronal es una técnica de modelado que es utilizada, entre otras cosas, para aprender cómo imitar el desempeño de un sistema. La Red Neuronal consiste en un conjunto de elementos que comienzan conectados en un estado aleatorio, y, basado en retroalimentación operacional, son moldeados en el complejo patrón requerido para generar los resultados deseados.