Predicción de carga eléctrica a corto plazo en el área occidental de la República Mexicana mediante el uso de redes neuronales artificiales.

Celestino Sánchez Fdz. (csanchez@realtime.com.mx)
Asesor: Dr. Eduardo de Jesús García García. (egarcia@campus.cem.itesm.mx)
I.T.E.S.M. C.E.M.
Atizapán de Zaragoza, Edo. de Mex., México.
Marzo del 2000.

Abstract: Este trabajo trata sobre los resultados obtenidos en la predicción a corto plazo mediante el uso de redes neuronales artificiales, de la carga eléctrica realizados en el área occidental de la República Mexicana. Para ello se usaron datos de carga proporcionados por la C.F.E. correspondientes a los años de 1996 y 1997 tomados en forma horaria.

Los diferentes experimentos se realizaron sobre redes feed-forward mediante el uso de los algoritmos de aprendizaje back propagation, cascade correlation y mediante el uso de redes recurrentes y el algoritmo recurrent cascade correlation, cada uno con diferentes topologías y parámetros de entrenamiento. Al final se presenta el algoritmo más exitoso y los parámetros con los que se llegó al mínimo valor tiempo de procesamiento- error.

PALABRAS CLAVE: Redes Neuronales, Predicción (Forecast), Series de Tiempo, Consumo Eléctrico, Feed-Forward, Redes Recurrentes.

1.0 Introducción.

El manejo de la producción, transmisión y distribución de la energía eléctrica es un reto al que los ingenieros electricistas siempre se han enfrentado. Una predicción precisa de carga eléctrica a corto plazo resulta en ahorros de costos de operación e incrementa las condiciones de seguridad del sistema eléctrico, permitiendo a empresas eléctricas comprometer sus fuentes de energía para optimizar los precios así como el posible intercambio con otras empresas generadoras o de distribución.

Para entender porque existe un gran interés en encontrar y mejorar los métodos para la predicción de carga eléctrica a corto plazo, se puede ver el equilibrio entre la producción/consumo desde dos puntos de vista. En primer plano, el período de la planeación: entre mas corto sea el periodo a predecir, más difícil es realizar modificaciones a los planes de producción de potencia. Existen limitaciones por retardos en el arranque, sistemas fuera de servicio por mantenimiento así como disponibilidad de combustibles fósiles y agua. En ese contexto, empresas aisladas tienen que confiar de sus propios medios de producción considerando todas las limitaciones existentes debiendo predecir con anterioridad la demanda que se tendrá y la forma de suplirla, aunque existen acuerdos entre compañías que permiten en un momento dado el intercambio de energía. En México en que la compañía que genera, transforma, transmite, distribuye y comercializa la energía eléctrica es una sola, este problema también se presenta ya que cada una de las regiones eléctricas deben satisfacer sus necesidades propias con las plantas de su jurisdicción y establecer coordinación con otras regiones para el citado intercambio de energía si es necesario.

El segundo punto de vista, el tipo de organización: para una compañía encargada de la distribución de energía la cual compra energía de un mayorista las tarifas son bien conocidas, pero algunos otros costos marginales dependen en gran medida de la situación del mercado en el momento. En forma particular el sobrepasarse de del consumo máximo contratado puede llegar a ser extremadamente caro. Aunque en México este punto no se aplica directamente por la cuestión tratada anteriormente, no debe descartarse del todo, considerando el sentido que el sector eléctrico nacional va tomando rumbo a su privatización y de la necesidad que las empresas que tomen una parte de este rubro conozcan y prevean las necesidades de su jurisdicción.

La predicción hecha por los seres humanos es muy buena aunque para llegar a un buen grado de exactitud se necesita de un verdadero experto con varios años de experiencia y especialización en el sector que le compete, un sistema automático nos permite no depender tan estrechamente de un experto, permitiendo que casi cualquier persona pueda efectuar tal predicción con una exactitud considerable y en muy corto tiempo, pudiendo usar, también, una gran cantidad de datos asociados a la predicción.

1.1 Estado del arte.

Jeffrey L. Elman en 1990 presentó una arquitectura conexionista novedosa con unidades recurrentes, estas unidades le proporcionaban a la red una característica de memoria que resultaba interesante.

Esta recurrencia fue modelada con la introducción de unidades de contexto al mismo nivel que las unidades de entrada. Elman presentó esta arquitectura en el tratamiento de lenguaje natural.

Scott E. Fahlman y Christian Lebiere en 1990 propusieron "cascade correlation" como un modelo de autogeneración: la red inicia con, únicamente, una capa de unidades de entrada y salida y automáticamente se entrena e introduce nuevas unidades ocultas una a una, creando una estructura multicapas. Esta arquitectura presentó características que la colocaban muy delante de back propagation principalmente por no requerir la retropropagación del error, algo que ha comprobado ser una de las cualidades que alenta el aprendizaje.

Scott E. Fahlman en 1991 presentó una variación al algoritmo cascade correlation: "recurrent cascade correlation" en donde combina la arquitectura de cascade correlation con la operación recurrente propuesta por Elman.

Conserva básicamente las cualidades de cascade correlation en cuanto a rapidez en el aprendizaje, buena generalización y construcción de una red mínima se refiere. Esta red presentó entre otras la posibilidad de realizar un mejor aprendizaje sobre secuencias temporales complejas

Peter C. McCluskey en 1993 usó y llevó a cabo una comparación con las tres arquitecturas usadas en nuestro trabajo: feed-forward back propagation, cascade correlation y recurrent cascade correlation, para hacer una predicción de manchas solares.

Sus resultados mostraron que cascade correlation tuvo un mejor desempeño en cuanto a disminución del error se refiere y solo en algunas pruebas recurrent cascade correlation se comportó mejor.

Thomas Czernichow, Antonio Piras y Marie-Madelein Martin en 1995 trataron sobre la predicción de carga eléctrica mediante el uso de redes neuronales. En su trabajo presentaron características de varias arquitecturas como back propagation , cascade correlation, redes recurrentes, mapas de Kohonen y algunos modelos híbridos sin exponer algún resultado en concreto.

T.Czernichow, A.Germond, B.Dorizzi y P.Caire presentaron un estudio bastante completo al realizar predicciones de carga aprovechando las bondades de los sistemas recurrentes. Tomaron como una de las variables de entrada el tipo de día (entre semana, fin de semana o día festivo particular), sus mejores resultados fueron con series de tiempo con variaciones lentas (series no caóticas).

2.0 Arquitecturas usadas.

2.1 Back propagation.

El algoritmo back propagation es central en muchos de los trabajos actuales sobre aprendizaje en redes neuronales. Este fue inventado independientemente en varias ocasiones, por Bryson y Ho [1969], Werbos [1974], Parker [1985] y Rumelhart, Hinton y Williams [1986]. El algoritmo da un método para cambiar los pesos w_{pq} en cualquier red del tipo feed-forward y "aprender" sobre un conjunto de pares entrada-salida $\{x_k^m, z_i^m\}$. Para este efecto consideremos primero una red de dos capas, mostrada en la figura siguiente.

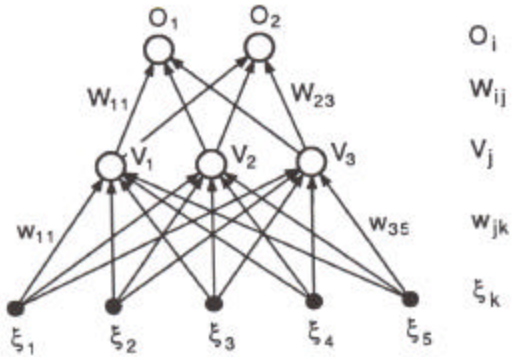

Figura 1. Red feedforward de dos capas.

Las salidas son denotadas por O_i, las unidades ocultas por V_j y las terminales de entrada por x_k. Existen conexiones w_{jk} desde las entradas hacia las unidades ocultas, y W_{ij} desde estas últimas hacia las correspondientes de salida. Nótese que el índice i siempre refiere a una unidad de salida, j a una oculta y k a una terminal de entrada.

Las entradas están siempre sujetas a valores particulares, vamos a etiquetar a diferentes patrones por un superíndice m, así la entrada k se representa como x_k^m cuando se presenta el patrón m. También usamos N para el número de unidades de entrada y p para el número de patrones de entrada $(m = 1, 2, ..., p)$. Dado el patrón m, la unidad oculta j recibe una entrada de red

$$h_j^m = \sum_k w_{jk} x_k^m \qquad \text{(ec. 1)}$$

y produce una salida

$$V_j^m = g\left(h_j^m\right) = g\left(\sum_k w_{jk}\, x_k^m\right) \qquad \text{(ec.2)}$$

La unidad de salida i así recibe

$$h_i^m = \sum_j W_{ij}\, V_j^m = \sum_j W_{ij} g\left(\sum_k w_{jk}\, x_k^m\right) \qquad \text{(ec.3)}$$

y produce para la salida final

$$O_i^m = g\left(h_i^m\right) = g\left(\sum_j W_{ij}\, V_j^m\right) = g\left(\sum_j W_{ij}\, g\left(\sum_k w_{jk}\, x_k^m\right)\right) \qquad \text{(ec.4)}$$

En este caso se han omitido los umbrales; estos pueden ser tomados como una entrada extra con un valor de –1 y conectada a todas las unidades en la red.

La medida del error o función costo

$$E[w] = \frac{1}{2} \sum_{mi} \left[z_i^m - O_i^m\right]^2 \qquad \text{(ec.5)}$$

ahora se obtiene

$$E[w] = \frac{1}{2} \sum_{mi} \left[z_i^m - g\left(\sum_j W_{ij}\, g\left(\sum_k w_{jk}\, x_k^m\right)\right)\right]^2 \qquad \text{(ec.6)}$$

En un sentido esto es todo sobre back propagation, pero existe una gran importancia práctica en la forma de las reglas de actualización de resultados. Para las conexiones desde las capas ocultas a la salida la regla de gradiente descendiente da :

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} = \eta \sum_m \left[z_i^m - O_i^m\right] g'\left(h_i^m\right) V_j^m = \eta \sum_m \delta_i^m\, V_j^m \qquad \text{(ec.7)}$$

donde se define

$$\delta_i^m = g'\left(h_i^m\right)\left[z_i^m - O_i^m\right] \qquad \text{(ec.8)}$$

El resultado es idéntico al obtenido para un perceptrón de una sola capa, con la salida V_j^m de las unidades ocultas jugando ahora el papel de entrada del perceptrón. Para las conexiones desde la entrada a las capas ocultas Δw_{jk} se debe diferenciar con respecto a los w_{jk}. Usando la regla de la cadena, se obtiene

$$\begin{aligned} \Delta w_{jk} &= -\eta \frac{\partial E}{\partial w_{jk}} = -\eta \sum_m \frac{\partial E}{\partial V_j^m} \frac{\partial V_j^m}{\partial w_{jk}} \\ &= \eta \sum_{mi} \left[z_i^m - O_i^m\right] g'\left(h_i^m\right) W_{ij}\, g'\left(h_j^m\right) x_k^m = \eta \sum_{mi} \delta_i^m\, W_{ij}\, g'\left(h_j^m\right) x_k^m = \eta \sum_m \delta_j^m\, x_k^m \end{aligned} \qquad \text{(ec.9)}$$

con

$$\delta_j^m = g'\left(h_j^m\right) \sum_i W_{ij} \delta_i^m \qquad \text{(ec.10)}$$

Nótese que la ecuación 9 tiene la misma forma que la 7, pero con una definición diferente de los $\partial's$. En general, con un arbitrario número de capas, la regla de actualización back propagation siempre tiene la forma

$$\Delta w_{pq} = \eta \sum_{patrones} \delta_{salida} \times V_{entrada} \qquad \text{(ec.11)}$$

donde *salida* y *entrada* se refieren a las dos terminaciones p y q de la conexión en cuestión, y V representa la activación apropiada entrada-fin desde una unidad oculta o una entrada real. El significado de δ depende de la capa de la que se trate; para la última capa de conexiones, este está dado por la ecuación 8, cuando para las otras capas está dado por una ecuación como la 10.

La ecuación 10 nos permite determinar δ para una unidad oculta V_j en términos de los $\delta's$ de las unidades O_i que ésta provee. Los coeficientes son los usuales $W_{ij}'s$, pero aquí ellos están propagando errores $(\delta's)$ hacia atrás en lugar de señales hacia delante: de ahí el nombre de error back propagation o solo back propagation.

Aunque se han escrito reglas de actualización 7 y 9 como sumas sobre todos los patrones m, ellos son usados en forma incremental: un patrón m es presentado en la entrada y entonces todos los pesos son actualizados antes de que el siguiente patrón sea considerado. Esto claramente decrementa la función costo (para una η suficientemente pequeña) en cada paso, haciendo pasos sucesivos para adaptar el gradiente local.

2.2 Cascade correlation.

Cascade correlation es una arquitectura y algoritmo de aprendizaje supervisado relativamente nuevo (1990) para redes neuronales artificiales. En lugar de solo ajustar los pesos en una red de topología fija, cascade correlation inicia con una red mínima, automáticamente entrena y suma nuevas unidades ocultas una por una, creando una estructura multicapas. Una vez que una unidad oculta ha sido dada de alta a la red, su peso del lado de la entrada se fija o "congela".

Cascade correlation combina dos ideas: la primera es la arquitectura de cascada, en la cual unidades ocultas son introducidas a la red una a la vez y no cambian después de haber entrado. La segunda es el algoritmo de aprendizaje, el cual crea e instala la nuevas unidades ocultas.
Para cada nueva unidad oculta, se intenta maximizar la magnitud de la correlación entre la salida de la nueva unidad y la señal del error residual que se está intentando eliminar.

La arquitectura de cascada se ilustra en la figura 2. Esta inicia con algunas entradas y una o más unidades de salida, pero ninguna unidad oculta. El número de entradas y salidas es dictada por el problema y por la representación de entrada-salida que se ha determinado para el experimento. Cada entrada está conectada a cada unidad de salida por una conexión con un peso ajustable.

Existe también una entrada , permanentemente con una señal de +1. Las unidades de salida pueden solo producir una suma lineal de sus entradas ponderadas o pueden usar alguna función de activación no lineal.

Figura 2. Arquitectura Cascade Correlation

Se adicionan unidades ocultas a la red una a una. Cada unidad oculta nueva recibe una conexión desde cada una de las entradas originales de la red y también desde cada unidad oculta ya existente. Los pesos de entrada de las unidades ocultas son fijados en el momento en que la unidad es puesta en la red, únicamente las conexiones de salida son entrenadas repetidamente. Cada nueva unidad por lo tanto suma una nueva capa oculta de una unidad en la red, a menos que algunos de sus pesos de entrada sean cero.

El algoritmo de aprendizaje empieza sin unidades ocultas. Las conexiones directas entre la entrada y la salida son entrenadas tanto como sea posible sobre el conjunto completo de datos de entrenamiento. En virtud de que no existe la necesidad de una propagación del error hacia atrás a través de las unidades ocultas se puede usar cualquier algoritmo de aprendizaje usado para redes de una sola capa, tales como la regla "delta", "quickpropagation", etc.

Cuando no se presenta una reducción significante del error después de un cierto número de ciclos de entrenamiento (controlados por un parámetro de "paciencia" fijado por el usuario), se "corre" la red una última vez sobre el conjunto completo de parámetros de entrenamiento para medir el error. Si se satisfacen los requerimientos de ejecución de la red, el algoritmo para; si no, entonces es porque existe un error residual que queremos aún disminuir. Para lograr esto, se deberá aumentar otra unidad oculta a la red, usando el mecanismo descrito posteriormente. La nueva unidad es introducida a la red, sus pesos en la entrada se fijan y todos los pesos de salida son una vez más entrenados usando un algoritmo como "quickpropagation". Este ciclo se repite hasta que el error sea razonablemente pequeño.

2.3 Recurrent cascade correlation.

Cascade correlation (correlación en cascada), como back propagation (retropropagación) y otras arquitecturas feed-forward, no tienen memoria de corto plazo en la red. Las salidas en un tiempo dado son función solamente de las entradas actuales y los pesos en la red. Por supuesto, muchas tareas del mundo real requieren el reconocimiento de una secuencia de entradas y, en algunos casos, la correspondiente producción de una secuencia de salidas.

Un número de arquitecturas recurrentes han sido propuestas en respuesta a esta necesidad. Quizá la más ampliamente usada en el presente, es el modelo de Elman[10], la cual asume que la red opera en pasos de tiempo discretos. Las salidas de las unidades ocultas de la red en el tiempo t son regresados hacia atrás para usarse como entradas adicionales en el tiempo $t+1$.

Recurrent cascade correlation es una arquitectura que suma la operación recurrente de Elman a la arquitectura cascade correlation. De cualquier forma, algunos cambios fueron necesarios para hacer que los dos modelos trabajaran juntos. En la arquitectura original de Elman hay una conectividad total entre las salidas previas de las unidades ocultas y la propia capa oculta. En cascade correlation, nuevas unidades ocultas son introducidas una a una y son fijadas una vez adicionadas a la red. Aunque se puede violar este concepto al insertar las salidas de unidades ocultas nuevas a unidades ocultas existentes como nuevas entradas. Por otro lado, la red debe ser capaz para formar circuitos recurrentes para retener el estado para un tiempo indefinido.

La solución que se adoptó en recurrent cascade correlation fue adicionarle a cada unidad candidata una entrada recurrente ponderada que regresa de la propia salida de la unidad en el paso de tiempo anterior. El enlace recurrente es entrenado con los pesos de las entradas de las demás unidades para maximizar la correlación de la unidad candidata con el error residual.

Si el enlace recurrente adopta un valor fuertemente positivo, la unidad funcionará como un flip-flop, manteniendo su estado previo a menos que las otras entradas lo obliguen a cambiar; si el enlace adopta un valor negativo, la salida de la unidad tenderá a oscilar entre valores positivos y negativos en cada paso de tiempo a menos que las otras entradas lo mantengan en su lugar; si el peso de este enlace recurrente es cercano a cero, entonces la unidad actuará como una compuerta de alguna clase.

Cuando una unidad candidata es introducida a la red activa como una nueva unidad oculta, el peso de su enlace recurrente se fija o "congela" junto con todos los demás pesos. Cada nueva unidad oculta es en efecto una sola variable de estado en una máquina de estados finitos que es construida específicamente para una tarea en particular. La salida, $V(t)$, de cada unidad recurrente es calculada como sigue:

$$V(t) = \sigma\left(\sum I_i(t)\, w_i + V(t-1)\, w_s\right) \qquad \text{(ec.14)}$$

donde σ es alguna función no lineal aplicada a la suma ponderada de entradas I más la conexión recurrente . Durante la fase de entrenamiento de las candidatas, se ajustan los pesos w_i y w_s para cada unidad a fin de maximizar la correlación. Para ello se requiere calcular la derivada de $V(t)$ con respecto a esos pesos:

$$\partial V(t) \Big/ \partial w_i = \sigma'(t)\left(I_i(t) + w_s\, \partial V(t-1)/\partial w_i\right) \qquad \text{(ec.15)}$$

$$\partial V(t) \Big/ \partial w_s = \sigma'(t)\left(V(t-1) + w_s\, \partial V(t-1)/\partial w_s\right) \qquad \text{(ec.16)}$$

El término de la extrema derecha refleja la influencia del peso en cuestión sobre el estado previo de la unidad. Desde que se calculó $\partial V(t-1)/\partial_w$ sobre el paso de tiempo anterior, se puede solo salvar este valor y usarlo en el paso de tiempo actual. Así que la versión actual del algoritmo de aprendizaje requiere guardar un solo número adicional para cada peso candidato, más $V(t-1)$ para cada unidad. En $t=0$ se asume que el valor previo de la unidad y las derivadas previas son todos cero.

3.0 Etapa de predicción.

3.1 Obtención de datos y normalización.

Se obtuvo por parte de la Comisión Federal de Electricidad (C.F.E.) los valores de carga horaria de dos años (1996 y 1997) , 17,520 valores de carga, del área occidental de la República Mexicana, un 10% de ellos se usó como validación y el resto para la etapa de producción. entrada de longitud 24 y 6 de salida, es decir, se toman las 24 horas anteriores al punto de la predicción y se pretenden obtener las siguientes 6. Debido a la saturación de las unidades por la sigmoide usada como función de transferencia, cada uno de los valores de carga se normalizaron dividiéndolos entre 10,000.

3.2 Experimentación con back propagation.

Al inicio de este trabajo se tomó una arquitectura de 3 capas: 5 datos en la capa de entrada, una sola capa oculta de 10 unidades y una capa de salida compuesta por solo una unidad de salida.

Posteriormente se consideró que esta topología , en forma práctica, no nos serviría de mucho, ya que es más provechoso para una simulación de este tipo una ventana de datos de salida (datos a predecir) mas grande, debido a ello se escogió el tomar 6 datos de salida y 24 datos de entrada.

Entonces se determinó iniciar las etapas de aprendizaje con una topología de 3 capas: la capa de entrada con 24 unidades (simulando las 24 horas previas a la 1/er. hora de predicción) , 1 capa oculta de 10 unidades y la capa de salida con 6 unidades.

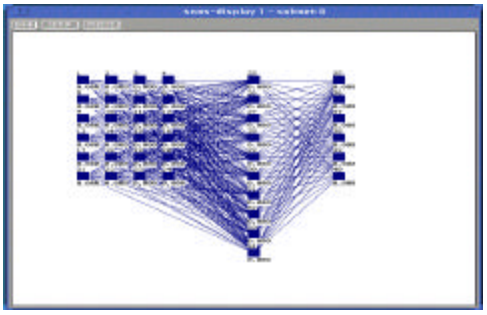

Figura 3. Arquitectura inicial

Se realizaron varias pruebas comparando redes de una sola capa oculta y otras con tres capas de este tipo. El mejor desempeño en cuanto a disminución del error se encontró con las arquitecturas con tres capas ocultas fijándose entonces los siguientes experimentos con esta configuración (una capa de entrada de 24 datos, tres capas ocultas con 10 unidades cada una y otra capa de salida con 6 unidades).

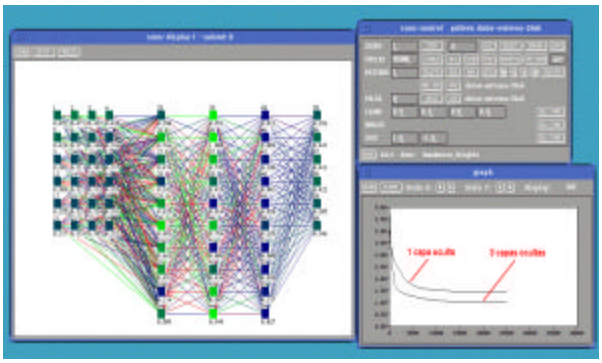

Figura 4. Mejor disminución del error en una red de 3 capas ocultas.

3.2.1 Entrenamiento y resultados.

Con redes de 3 capas ocultas se realizaron varias pruebas variando el paso de aprendizaje y momentum:

Paso de Aprendizaje.	Momentum.
0.6	0.2
1.2	0.4
0.2	0.2
Por etapas: (5000 iter.) 0.9 (10000 iter.) 0.6 (10000 iter.) 0.3	 0.4 0.4 0.4
0.6	0.4
0.002	0.4
0.2	0.4

Ninguna configuración nos entregó mejores resultados en cuanto a tiempo de procesamiento-error que con un paso de aprendizaje de 0.9 y momentum de 0.4

3.3 Experimentación con cascade correlation.

Se iniciaron experimentos bajo este algoritmo con una configuración de número de unidades de entrada y salida igual que con back propagation (24 y 6 respectivamente).

a. Iniciamos esta etapa con la misma configuración en cuanto a cantidad de unidades de entrada y de salida (24 y 6). Los parámetros usados en el algoritmo de aprendizaje usado con esta arquitectura, quickpropagation, son los marcados como típicos en el manual del simulador SNNS (usado para este trabajo), se intentaron algunas pruebas modificando estos valores

pero ninguna configuración se comportó mejor que los citados por ese manual. Dichos parámetros y valores son :

1. Paso de aprendizaje, especifica la amplitud del paso del gradiente descendiente cuando se está minimizando el error de la red y un valor típico es: 0.0001
2. Parámetro de máximo crecimiento, realiza un tipo de momentum dinámico : 2.0
3. Término de decaimiento de pesos : ≤ 0.0001
4. Paso de aprendizaje, el cual especifica la amplitud del paso del gradiente ascendente cuando se está maximizando la covarianza: 0.0007

En cuanto a los valores de configuración propios del algoritmo cascade correlation se tomaron como iniciales los siguientes:

1. Error máximo de salida por unidad : 0.002
2. Cambio mínimo de covarianza : 0.005
3. Máximo número de actualizaciones de la covarianza: 100
4. Máximo número de unidades candidatas: 5

La mayoría de estos valores se mantuvieron constantes durante todos los experimentos por lo que solo se mencionarán aquellos que se hayan modificado. En este primer intento se pusieron a prueba dos redes con un número máximo de unidades ocultas a crear de 30, una de ellas con un número máximo de 5 unidades candidatas y la otra con 10. Ambas redes se comportaron de manera muy similar y llegaron a un error también muy parecido, SSE = 1.7307, los tiempos de cálculo fueron casi los mismos como para considerarlos punto de decisión (aprox. 3 hs.).

b. A continuación se efectuaron dos pruebas más pero se aumentó el número máximo de unidades ocultas a 80, una de las redes nuevamente con 5 unidades candidatas y la otra con 10. Los tiempos de procesamiento fueron nuevamente muy similares, el error obtenido en la red con 5 unidades candidatas fue SSE = 0.8557 y para la de 10 unidades SSE = 0.7932.
c. En una prueba más se usaron dos redes una de ellas con un error máximo de la unidad de salida de 0.02 y el otro con 0.2, dejamos ya fijo el valor del número de unidades candidatas en 10 y además se incrementó la cantidad de unidades ocultas a 90.En la red con un error máximo en las unidades de salida de 0.2 no se alcanzó la cantidad de 90 unidades ocultas creadas, solo se obtuvieron 73 y se llegó a un error SSE = 0.8975. En el correspondiente 0.02 de error máximo en las unidades de salida, se obtuvo un SSE = 0.6726. Como se ha mostrado ya se han obtenido configuraciones con este metalgoritmo que mejoran el desempeño del back propagation.
d. A continuación se compararon las dos redes y sus curvas descritas en el experimento anterior con otra red con posibilidad de generar más unidades ocultas, 100.
El error obtenido con 100 unidades ocultas fue SSE = 0.5652
e. Se realizó una prueba más dando la posibilidad de generar 200 unidades ocultas, el error se disminuyó hasta 0.1822 pero el tiempo de proceso fue alto, 96 hs.

f. En la siguiente prueba se aumentó el número de actualizaciones de la covarianza y de ciclos de presentación de los patrones a 200, fijando, primeramente, a 100 la cantidad de unidades ocultas y posteriormente se corrió con 150 unidades. El error SSE para la red con 100 unidades ocultas fue de 0.4918 pero la red con 150 unidades nos disminuyó el error hasta 0.1864 solo un poco mayor que el error que nos entregó la red con 200 unidades ocultas y 100 iteraciones de aprendizaje. En cuanto al tiempo, este fue mucho menor (solo 24 horas) contra la red en la que se crearon 200 unidades ocultas.

g. Como última prueba se decidió hacer una corrida con 100 unidades ocultas y aumentar el número de iteraciones de aprendizaje a 300. Se tenía una hipótesis que al aumentar el número de repeticiones en el aprendizaje podríamos obtener una mayor disminución del error usando menos unidades ocultas La hipótesis se desechó ya que el error no disminuyó en la magnitud esperada SSE = 0.4312 y el tiempo de procesamiento fue del doble con respecto al experimento con 100 iteraciones y 200 unidades ocultas.

3.4 Recurrent cascade correlation.

Para quickpropagation, se tomaron los mismos parámetros de configuración que en el experimento anterior con cascade correlation. En cuanto a los valores de configuración propios del algoritmo recurrent cascade correlation se tomaron como iniciales los siguientes:

1. Error máximo de salida por unidad : 0.2
2. Cambio mínimo de covarianza : 0.005
3. Máximo número de actualizaciones de la covarianza: 100
4. Máximo número de unidades candidatas: 5
5. Cambio en el error: 0.005
6. Paciencia a la salida: 50
7. Número máximo de iteraciones: 100

a. En la primera prueba se especificó a 30 como la cantidad máxima de unidades ocultas que podría crear la red. Se realizó una comparación con otra con capacidad de generar 50 unidades ocultas. La red de 30 unidades tuvo un error SSE = 1.5723 y la de 50 unidades, SSE = 1.1300

b. La siguiente prueba consistió en aumentar nuevamente la cantidad de unidades ocultas que podía crear la red a 80 y 100 así como el error máximo de salida por unidad a 0.02, tratando de disminuir el error. Los valores de error fueron para la red con 80 unidades: 0.6848 y para la red con 100 unidades: 0.4985. Los resultados antes mostrados en comparación con la arquitectura cascade correlation fueron similares como se muestra a continuación:

ERROR OBTENIDO

Unidades ocultas.	Cascade Correlation	Recurrent Cascade Correlation
30 unidades ocultas	1.7307	1.5723
80 unidades ocultas	0.7932	0.6848
100 unidades ocultas	0.5652	0.4985

La diferencia principal se debió a los tiempos de procesamiento donde, en promedio, fueron 3 veces mayor que para las pruebas con cascade correlation y la gran cantidad de recursos del

hardware necesarios, esto último limitó la generación de más pruebas ya que la memoria RAM de los equipos y la velocidad de procesamiento no fueron suficientes para los cálculos requeridos.

5.0 Conclusiones.

Las pruebas anteriores demostraron un mejor desempeño tanto en tiempo de cálculo como disminución del error en el algoritmo cascade correlation con 150 unidades ocultas. El algoritmo recurrent cascade correlation tuvo un desempeño muy similar en cuanto a la disminución de error mas no en cuanto al tiempo tomado para ello. Los recursos computacionales necesarios para su uso fueron considerables y por ello la cantidad de experimentos fue menor. Así que en general el desempeño del algoritmo simple cascade correlation fue superior.

Diversas investigaciones han encontrado algunos problemas y limitaciones en el algoritmo back propagation que lo han hecho ineficiente con respecto a otros. El más importante es la lentitud con la cual back propagation aprende de ejemplos. De aquí parte la mayor importancia de este trabajo, presentar y experimentar sobre arquitecturas diferentes al back propagation tradicional y como se demostró los resultados sobre arquitecturas incrementales da excelentes perspectivas para seguir trabajando sobre ellas en los diferentes campos en los que la computacional neuronal ha resultado altamente eficiente.

6.0 Bibliografía.

[1] ZAIYONG TANG, et. al., ***Feed Forward Nets as Models for Time Series Forecasting***.

[2] THOMAS CZERNICHOW, et.al., ***Electrical Short Term Load Forecasting with Artificial Neural Networks.***

[3] P. WAGNER WILLIAM, et.al., ***Daily Peak Load Electricity Forecasting Using Artificial Neural Networks.***

[4] JAYENDAR RAJAGOPALAN,et.al., ***Short Term Load Forecasting Using Functional Link Networks: Results of a Feasibility Study.***

[5] THOMAS CZERNICHOW, et.al., ***Improving Recurrent Network Load Forecasting***.

[6] JOHN HERTZ, et.al., ***Introduction to the theory of neural computation***. Santa Fe Institute, 1991.

[7] SCOTT E. FAHLMAN, ***The Recurrent Cascade-Correlation Architecture***, School of Computer Science, Carnegie Mellon University, 1991.

[8] JEFFREY L. ELMAN, ***Finding Structure in Time,*** University of California, San Diego. 1990.

[9] Boletín ***"Prospectiva del Sector Eléctrico 1998-2007"***, Secretaría de Energía, Gobierno Federal, México 1998.

[10] HELGE RITTER, ***Neural Computation and Self-Organizing Maps***, Addison Wesley, 1992.

[11] ERIC WEISSTEIN, ***Encyclopedia of Mathematics***, http://www.treasure-troves.com/math/.

[12] SCOTT E. FAHLMAN / CHRISTIAN LEBIERE, ***The Cascade Correlation Learning Architecture***, School of Computer Science, Carnegie Mellon University, 1990.