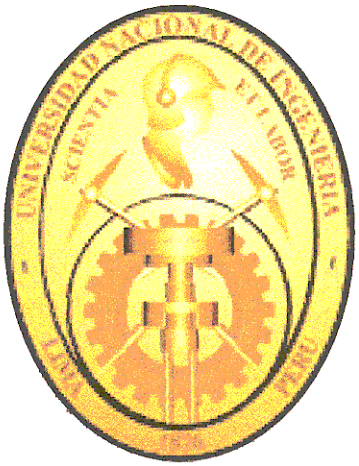
UNIVERSIDAD NACIONAL DE INGENIERIA

FACULTAD DE INGENIERIA DE PETROLEO



"APLICACION DE UNA RED NEURAL ARTIFICIAL PARA LA GENERACION DE CURVAS DE POROSIDAD NEUTRON PARA DOS POZOS EN EL NOROESTE DEL PERU"

PARA OPTAR EL TITULO PROFESIONAL DE INGENIERO DE PETROLEO

JOSE MANUEL CASTRO SOLORZANO

PROMOCION 1990 - I

LIMA - PERU

2003

INDICE

1. INTRODUCCION.
2. OBJETIVO.
3. GENERALIDADES.
    A. COMPARACION ENTRE PROGRAMACION ESTRUCTURADA E INTELIGENCIA ARTIFICIAL.
    B. FUNDAMENTOS DE REDES BIOLOGICAS.
    C. HISTORIA DE LAS REDES NEURALES ARTIFICIALES.
    d. APLICACIONES EN GENERAL Y EN PETROLEOS.
4. REVISION TEORICA Y CONCEPTOS.
    A. RED NEURAL ARTIFICIAL.
    B. NEURONA ARTIFICIAL.
    C. ENLACE.
    D. ESTRUCTURA.
    E. ADAPTABILIDAD Y GENERALIZACION.
    F. APRENDIZAJE DE LA RED NEURAL.
    G. ENTRENAMIENTO.
    H. FUNCION DE ACTIVACION.
5. METODO DE RETROPROPAGACION.
6. ALGORITMO DE PROGRAMACION DEL METODO DE RETROPROPAGACION.
7. APLICACION PRACTICA DE LA TEORIA DE REDES NEURALES.
    A. OBJETIVO.
    B. SITUACION ACTUAL.
    C. COMENTARIO SOBRE LA SITUACION ACTUAL.
    D. DESCRIPCION DEL PROBLEMA PASO A PASO.
    E. DESARROLLO DEL PROBLEMA.
        I. DATOS PARA EL ENTRENAMIENTO.
        II. EL PROGRAMA.
    F. RESULTADOS.

8. EVALUACION ECONOMICA DE LA APLICACION.
9. CONCLUCIONES Y RECOMENDACIONES.
10. BIBLIOGRAFIA.
ANEXO 1. CODIGO DEL PROGRAMA EN MICROSOFT VISUAL BASIC.
ANEXO 2. CURVAS DE LOS POZOS DE PRUEBA.
ANEXO 3. CURVAS DE LOS POZOS H E I CON LA CURVA NPHI GENERADA.
ANEXO 4. MAPAS Y CORRELACIONES.

1. **INTRODUCCION**

El presente estudio esta orientado a demostrar la utilidad del empleo de herramientas de inteligencia artificial, en concreto las redes neurales artificiales, en la resolución de problemas de ingeniería, en especial en la interpretación de información proveniente de campo como alternativa al análisis tradicional basado en la programación secuencial.

De especial interés son los problemas en los cuales se cuenta con una cantidad importante de información que puede ser usada como base sobre la cual establecer las correlaciones que permitan resolver los problemas individuales.

Entre las muchas aplicaciones de las redes neurales artificiales se encuentra la interpretación de registros eléctricos. Dentro de este grupo destacan aquellas dedicadas a la generación de curvas específicas en pozos que carezcan de las mismas partiendo de información proveniente de pozos vecinos cuyas curvas son utilizadas como variables de apoyo para este propósito. La información que se toma como base sobre la cual generar la curva puede provenir de distintas fuentes como otras curvas (porosidad neutrón) o información de testigos (permeabilidad). También se conoce la aplicación de esta tecnología para la generación de registros de propiedades obtenidas de imágenes de resonancia magnética a partir de registros convencionales. Adicionalmente, el empleo de herramientas de inteligencia artificial también ha sido investigado en campos como la interpretación sísmica, y la estimulación de pozos, a tal punto de que se tiene conocimiento de sistemas que pueden generar modelos de fracturamiento hidráulico comparables a los obtenidos por los simuladores más sofisticados del mercado.

Todas estas aplicaciones y muchas otras se encuentran ya en uso en la industria.

Es evidente el potencial de estas herramientas para ser utilizadas en campos que cuenten con abundante información y la posibilidad de añadirle valor a ésta a muy bajo costo sólo aplicando adecuadamente la tecnología que tenemos a nuestro alcance y que a continuación pasaremos a presentar.

2. **OBJETIVO**

El objetivo de este trabajo es la generación de las curvas de porosidad neutrón utilizando redes neurales artificiales en dos pozos ubicados en el lote Z2A-B a partir de sus curvas Rayos Gamma e Inducción Eléctrica así como información de pozos vecinos, todo esto dentro de un marco de viabilidad económica que hacen de esta herramienta una alternativa eficiente y rentable para incrementar la información de pozo destinada a la caracterización del reservorio.

3. **GENERALIDADES**

A. COMPARACION ENTRE PROGRAMACION ESTRUCTURADA E INTELIGENCIA ARTIFICIAL

La programación secuencial implica la utilización de algoritmos sobre los cuales se construyen programas que realizarán una labor especifica basados en el conocimiento preciso de los procesos que se están analizando. Por ejemplo, el análisis de registros eléctricos requiere el conocimiento de las fórmulas y correlaciones que permiten la determinación de la saturación de agua o la permeabilidad a partir de la información obtenida del pozo. La simulación de reservorios es otro ejemplo de programación estructurada.

Por otro lado, el desarrollo de la computación de quinta generación ha conducido al uso de inteligencia artificial y herramientas que complementen a la programación secuencial. Dentro de estas herramientas podemos mencionar a los sistemas expertos, redes neurales artificiales, algoritmos genéticos, etc. Estas herramientas son una alternativa para el análisis de la información y entre ellas destacan notablemente las redes neurales artificiales las cuales presentan características interesantes cuando se las compara con las metodologías tradicionales de análisis y programación (Tabla 1).

	PROGRAMACION ESTRUCTURADA	REDES NEURALES
PROCESAMIENTO	*SECUENCIAL: UN PASO A LA VEZ*	*PARALELO: NO HAY SECUENCIA PROGRAMADA*
METODO	*POR REGLAS*	*POR EJEMPLOS*
APLICACIONES	*MATEMATICAS, PROCESAMIENTO BASADO EN FORMULAS*	*RECONOCIMIENTO DE PATRONES, PREDICCIONES*
AUTO APRENDIZAJE	*SOLO MODIFICA LOS PARAMETROS ALGORITMICOS*	*ADAPTABILIDAD CONTINUA*
TOLERANCIA A FALLOS	*BAJA*	*ALTA*
PROGRAMACION	*DIFICIL*	*FACIL*

Tabla 1. Comparación entre programación estructurada y por redes neurales.

B. FUNDAMENTOS DE REDES BIOLOGICAS

Todos los organismos vivos están conformados por células, y la unidad básica del sistema nervioso es la neurona. Básicamente una neurona recibe información de otras fuentes en forma de pulsos electroquímicos a través de las dentritas. Basados en la naturaleza de esta señal, la neurona realizará un proceso de activación (excitación) o desactivación (inhibición) de la señal y la proyectará a través del axon al resto de neuronas conectadas a ella. El punto de contacto entre dos neuronas, es decir, la conexión del axon de la neurona emisora de la señal y las dentritas de la neurona receptora se llama sinapsis y el grado de interconexión entre ambas se llama fuerza sináptica (Figura 1).

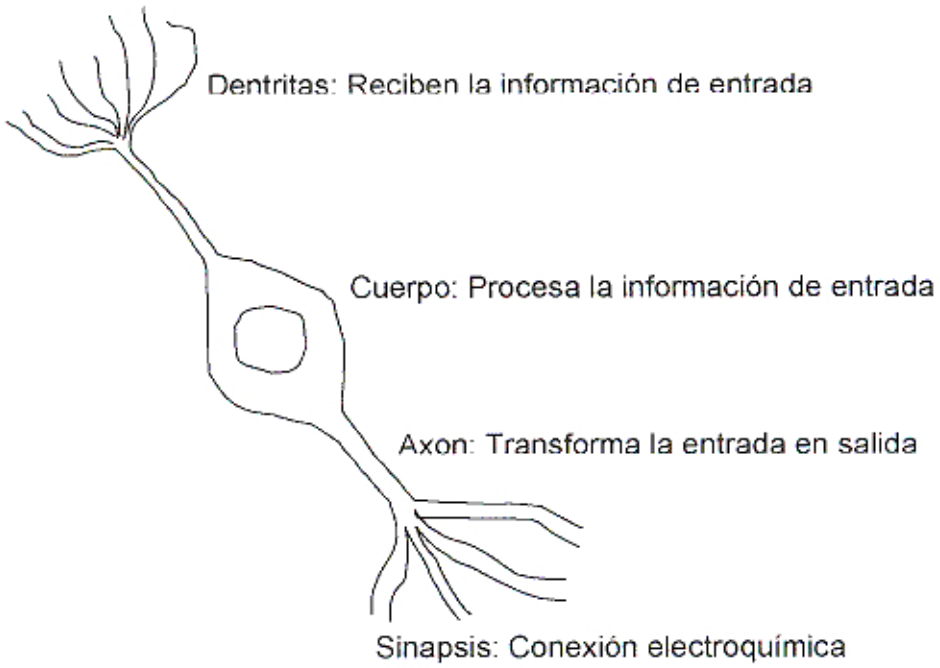


Figura 1. Partes de una neurona biológica típica.

C. HISTORIA DE LAS REDES NEURALES ARTIFICIALES

Los inicios de las redes neurales artificiales se remontan a la década de 1940 con los estudios realizados por McCulloch y Pitts. Mas tarde en 1953 Frank Rosenblatt definió el "Perceptrón" que es una especie de neurona básica y demostró que en un número finito de pasos este Perceptrón podía, en base a prueba y error, desarrollar un vector de enlace entre datos de entrada y salida que permitiera clasificar conjuntos linealmente separables. Además demostró que el resultado final era independiente de los valores iniciales de los enlaces.

Alrededor de la misma época, Minskey y Papert demostraron que problemas mas complejos como la solución de la relación XOR (disyunción exclusiva), que es una función lógica no lineal, no se podían resolver con redes neurales de una sola capa.

Rosenblatt también estudió estructuras con mas de una capa de neuronas pero se encontró con el problema de desarrollar un algoritmo de aprendizaje para todas las conexiones de la red. No fue sino hasta entrada la década de 1980 que, gracias a investigadores como John Hopfield, se pudo al fin desarrollar un algoritmo capaz de entrenar a redes neurales de múltiples capas y desde entonces los campos de aplicación han crecido increíblemente y su éxito esta garantizado para futuras aplicaciones.

). APLICACIONES EN GENERAL Y EN LA INDUSTRIA DEL PETROLEO

Las actuales aplicaciones de las redes neurales artificiales son innumerables, algunos de los campos más prometedores son:

- Reconocimiento de patrones.
- Reconocimiento de caracteres.
- Reconocimiento del habla.
- Análisis de imágenes.
- Predicción del clima.
- Predicción de reacciones nucleares, etc.

En cuanto a la industria del petróleo, también se ha aplicado esta tecnología especialmente en las áreas de exploración y caracterización de reservorios y las siguientes son sólo algunas aplicaciones y estudios de lo que este nuevo enfoque de manejo de información es capaz hacer:

- Predicción de resultados de estimulaciones en pozos.
- Correlación y extrapolación de datos de testigos a todo el pozo vía registros eléctricos (permeabilidad, porosidad, litología, etc.).
- Predicción de resultados de simulación numérica (neurosimulación) para optimizar el manejo de un campo petrolero.
- Predicción de la producción de un campo determinado.
- Procesamiento de datos de sísmica.
- Identificación de zonas en reservorios complejos.

- Generación de registros de resonancia magnética a partir de registros convencionales, etc.

Algunas de estas aplicaciones han sido presentadas en artículos técnicos y verificadas con datos de campo.

Uno de los campos en los cuales las redes neurales artificiales han encontrado una gama de aplicaciones es el referente al manejo de información proveniente de registros eléctricos (well logging) incluyéndose los registros Rayos Gamma, Porosidad, Densidad, y Resistividades. Dentro de este campo destaca la posibilidad de reconstruir curvas específicas en un determinado pozo a partir de información de otros pozos en la vecindad.

El presente estudio demuestra que es posible construir la curva de porosidad Neutrón (NPHI) en pozos en los cuales no se llevó a cabo dicho registro a partir de la información actual del pozo como son las curvas de Rayos Gamma y Resistividad y de la información de pozos vecinos que aporten la correlación entre estas curvas dato y su curva porosidad neutrón respectiva.

Las posibilidades de aplicación de esta tecnología en nuestro medio son muy grandes dada la gran cantidad de información de la que se dispone en áreas como el Noroeste con más de 40 años de historia productiva y por tratarse de una herramienta fácil de implementar y con costos extremadamente bajos comparados con los resultados que ofrece.

4. REVISION TEORICA Y CONCEPTOS

A. RED NEURAL ARTIFICIAL

Una Red Neural Artificial (RNA) es un sistema de procesamiento de información compuesto por unidades llamadas Neuronas Artificiales (NA) las cuales se encuentran interconectadas entre sí y permiten a la red recibir y procesar información. Su nombre proviene del hecho de que este sistema esta inspirado en el cerebro humano y en la forma en que éste procesa la información que recibe de ahí que tenga ciertas características similares a las redes neurales biológicas.

Los componentes principales de una RNA son las Neuronas Artificiales (NA) y los Enlaces entre ellas (Figura 2).

B. NEURONA ARTIFICIAL

Es la unidad principal de procesamiento de información de la RNA. Esta diseñada para imitar las características fundamentales de la neurona biológica. Una NA típica recibe información ya sea del exterior de la red o de otras neuronas de la misma RNA, y luego de procesar dicha información la envía al exterior de la red o a otras neuronas. En esencia, se le aplica un conjunto de datos de entrada, cada uno representando la salida de otra neurona. Cada uno de estos datos de entrada multiplica a un valor correspondiente de interconexión entre las dos neuronas llamado enlace o peso, cuya analogía se encuentra en la fuerza sináptica entre neuronas biológicas. Luego, todos estos productos son sumados para obtener así la señal neta de ingreso o nivel de activación (activation level) de la neurona. Dicho en otras palabras, se calcula el producto escalar entre el vector de entrada y el vector de pesos de entrada de la NA (Figura 3). Adicionalmente, algunas NA tienen la posibilidad de aplicar una función de activación a la señal neta de ingreso obteniéndose así la señal de salida.

C. ENLACE

Las neuronas artificiales de la RNA pueden transmitirse información a través de los enlaces que estén definidos entre ellas. Por supuesto una neurona sólo puede transmitir información a aquellas con las cuales tenga un enlace definido.

El enlace o peso es quizás el componente más importante de la RNA porque de su valor depende el comportamiento final de la red. Las operaciones que se llevan a cabo dentro de las neuronas artificiales son similares en todas ellas, lo que caracteriza en sí a una RNA son los valores de los enlaces sobre los cuales se realizan estas operaciones y de los cuales dependerá finalmente el aprendizaje de la red neural.

El entrenamiento de la red tiene como propósito encontrar los valores de los enlaces o pesos que permitan a la red neural establecer la relación óptima entre los datos de entrada y los datos de salida de la misma.

D. ESTRUCTURA

La estructura de una RNA típica (Figura 2) consta de una serie de NA organizadas en capas ordenadas y conectadas entre si por medio de enlaces lógicos llamados "pesos" por su nombre en inglés (weights). Normalmente hay una capa de entrada de datos (input layer) y una de salida de datos (output layer) y entre estas dos una o mas capas de procesamiento intermedio más conocidas como capas escondidas (hidden layers). En la capa de entrada de datos no se aplica ninguna transformación a los mismos, sólo sirve para recibir la información y distribuirla directamente a la siguiente capa.

Existen muchas arquitecturas de RNA y cada una de ellas se caracteriza por la forma en que las NA están organizadas en la red y la forma en que están

interconectadas dichas neuronas entre sí. Cada tipo de arquitectura tiene aplicaciones específicas pero comparten sus características principales.

E. ADAPTABILIDAD Y GENERALIZACION

Las RNA tienen la capacidad de adaptarse a la información que reciben desarrollando de esta manera asociaciones entre los datos de entrada y los datos de salida. Por otro lado la respuesta de la RNA puede ser hasta cierto punto insensible a cambios pequeños en los datos de entrada, de ahí que tenga habilidad de ver a través del ruido y distorsión de la información que se le presenta lo cual es vital para el reconocimiento de patrones en los datos reales. Esto se debe principalmente a la arquitectura de la RNA que permite un alto grado de paralelismo en el procesamiento de la información. La red neural artificial puede "recordar" la relación entre un conjunto de datos de entrada y un conjunto de datos de salida, y va más allá, puede generalizar y dar resultados acordes con lo "aprendido" cuando se le presentan datos de entrada que no había "visto" anteriormente.

Su arquitectura se caracteriza principalmente por:

- El procesamiento de la información en sí ocurre dentro de las NA.
- Las respuestas (señales) obtenidas de cada NA son transmitidas a las demás a través de conexiones lógicas.
- Cada conexión lógica tiene una magnitud (peso) asociada a ella la cual, en una RNA típica, multiplicará a la señal que por ella transite en determinado momento.
- En cada NA se aplica una función (activación) a la señal neta de ingreso para producir la señal de salida.
- Las NA se encuentran organizadas en capas empezando por la que recibe la información directamente y terminando por la que produce la respuesta final asociada.

- La señal neta de ingreso a cada NA se obtiene al sumar los respectivos productos de señal y peso de las conexiones asociadas. (Figura 3)

F. APRENDIZAJE DE LA RED NEURAL

Esta es la característica más importante de la RNA ya que ella es la base de su capacidad de generalización y clasificación.

El aprendizaje en sí se basa en la corrección iterativa de los pesos (enlaces) entre neuronas llamado entrenamiento de la red y se basa en la comparación de la información que la red recibe y el resultado que se obtiene a partir de ella versus el resultado esperado (entrenamiento supervisado). Los valores de los pesos entre neuronas son los que finalmente determinan el comportamiento de la red y su capacidad de realizar el trabajo para el cual fue entrenada.

G. ENTRENAMIENTO

El entrenamiento consiste básicamente en conseguir que la RNA establezca una correlación entre los datos de entrada y los datos de salida deseados a través de correcciones iterativas de los enlaces o pesos entre las NA de manera que, al aplicársele el conjunto (o vectores) de datos de entrada, se produzca el conjunto de datos de salida deseado, o al menos uno consistente con el mismo.

Una vez que se ha decidido que estructura va a tener la red, ésta se encuentra lista para su entrenamiento. En este punto se asignan los valores iniciales de los enlaces de forma aleatoria.

El entrenamiento se efectúa al aplicar de forma secuencial los vectores de entrada mientras se ajustan los enlaces entre las neuronas de acuerdo a un

procedimiento predeterminado. Durante el entrenamiento los enlaces convergen a un conjunto de valores que eventualmente producirán los vectores de salida deseados para cada vector de entrada.

Los algoritmos de entrenamiento se clasifican en supervisados y no supervisados.

El entrenamiento supervisado, que se usará en este estudio, requiere emparejar cada vector de entrada con su respectivo vector de salida (o resultado deseado); juntos estos dos vectores son llamados un par de entrenamiento (training pair). Durante el entrenamiento, el mismo par de entrenamiento puede ser procesado mas de una vez.

Primero se aplica un vector de entrada a la RNA, se calcula el resultado que ésta produce y se compara con el vector de salida asociado (del par de entrenamiento correspondiente), el error (diferencia) se propaga hacia atrás en el sistema haciendo las correcciones en los enlaces de acuerdo a un algoritmo predeterminado. Este proceso es repetido con la totalidad de los datos y sólo se detendrá cuando se alcance un nivel de convergencia predeterminado inicialmente, es decir cuando la red haya aprendido de los datos presentados. Es importante considerar la cantidad y calidad de los datos a utilizarse para el entrenamiento. Es posible que una red nunca aprenda del conjunto de datos de entrenamiento debido principalmente a dos razones: primero, que los datos de entrada de la red no guardan la información sobre la cual se derivan los datos de salida, y segundo, no hay suficientes datos para lograr la convergencia. Se recomienda tener suficientes datos como para reservar un subconjunto de ellos para verificar el funcionamiento de la red una vez terminado el entrenamiento.

El algoritmo de entrenamiento más empleado es el de retropropagación (backpropagation) para RNA de múltiples capas. Sus fundamentos fueron

descritos en la década de los años 80 y amplió las posibilidades de aplicación de las RNA.

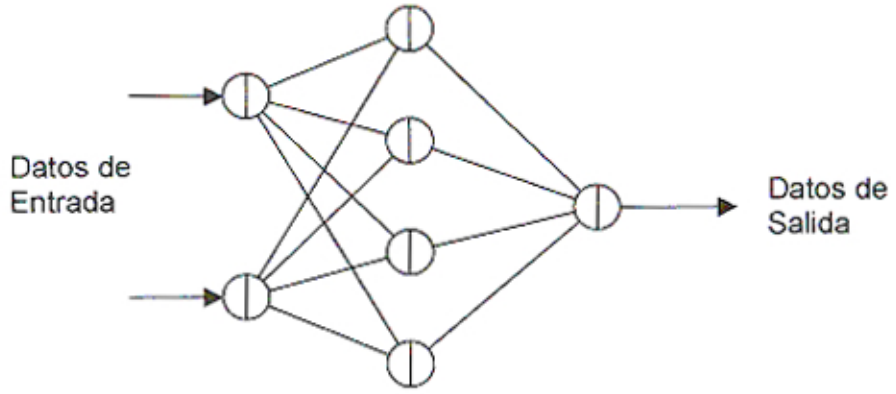


Figura 2: Red Neural Típica

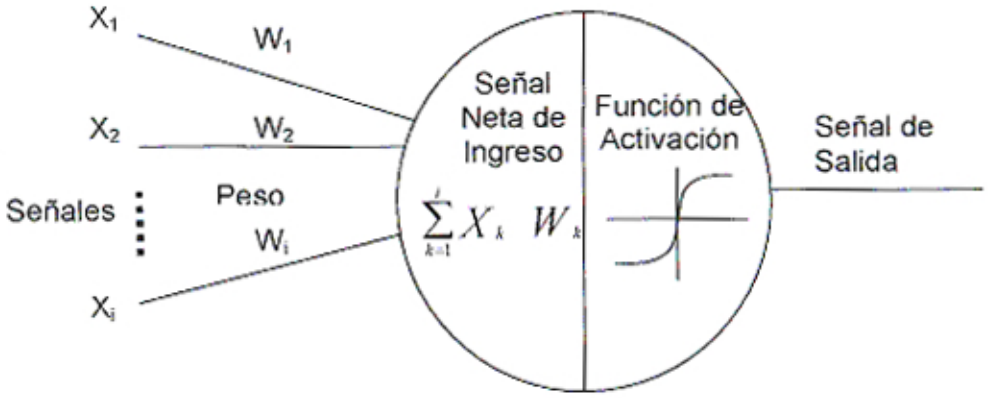


Figura 3: Neurona Artificial

H. FUNCION DE ACTIVACION

Cada neurona evalúa su respectiva señal neta de ingreso y le aplica la función de activación para determinar la señal de salida. Esta función de activación tiene un rol fundamental en la característica de no-linealidad de las RNA que es la responsable de muchas de las aplicaciones de las redes

neurales, especialmente las relativas a clasificación y reconocimiento de patrones.

Entre las muchas posibilidades para la función de activación destaca la función sigmoidal definida por la siguiente ecuación:

$$y = \frac{1}{1 + exp(-kx)}$$

Ecuación 1: Función Sigmoidal

En esta ecuación ***x*** representa la señal neta de entrada de la NA y ***k*** es un parámetro que controla la forma de la "S" sigmoidal. Para valores altos de ***k***, la ecuación mostrada se convierte en una función paso unitario excepto para x=0 donde se obtendría siempre un valor resultante de 0.5. El valor de ***k*** más comúnmente usado es 1 y para este valor obtenemos la variación de la función de activación con respecto a la señal neta de entrada expresada como una función de ***y***:

$$y' = y(1 - y)$$

Ecuación 2: Derivada de la Función Sigmoidal

Además de la función sigmoidal se usan las siguientes funciones de activación: la función Signo, la función Paso, la función Rampa Lineal, y la función Tangente Hiperbólica. Es preferible usar funciones diferenciables ya que, como se verá mas adelante, los algoritmos de aprendizaje de la red se basan en la derivada de esta función para la corrección de los pesos entre las NA.

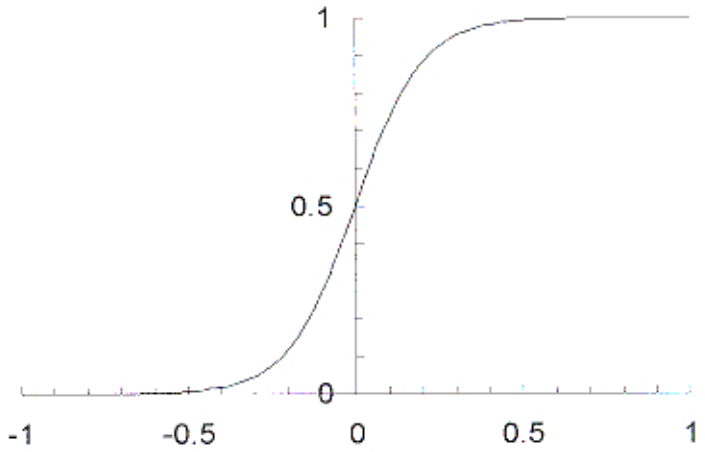


Figura 4: Función de Activación Sigmoidal (k=10)

5. METODO DE RETROPROPAGACION

Hasta este punto ya hemos presentado algunos conceptos de importancia de la teoría de redes neurales así como los distintos componentes que forman una red neural artificial en su arquitectura y en su nivel de procesamiento. A continuación pasaremos a describir el método por el cual la RNA finalmente adquiere la capacidad de resolver los problemas para los cuales fue diseñada.

El nombre proviene del hecho de que este método se basa en encontrar la salida de la red en la última capa y calcular así la diferencia entre las salidas deseadas y las salidas realmente obtenidas para un determinado par de entrenamiento. Dicha diferencia es entonces utilizada para hacer las correcciones a los valores de los enlaces de la red proporcionalmente a esta diferencia o error y en el sentido opuesto, de ahí el nombre de retropropagación del error.

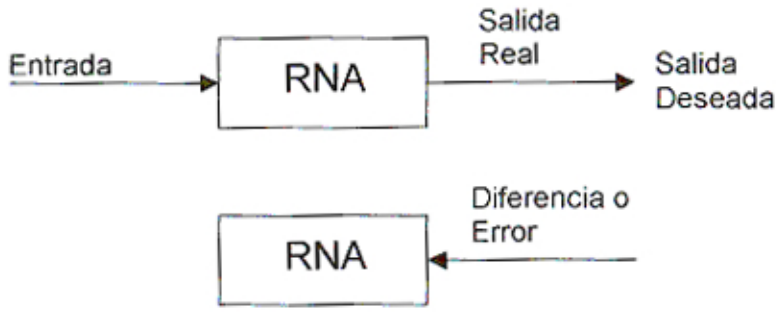


Figura 5: Sentido del procesamiento en el método de retropropagación

La teoría de este método involucra hacer correcciones a los enlaces de la última capa primero y luego utilizar estos cálculos para determinar las correcciones correspondientes a la capa de enlaces anterior, llegando de esta manera y siguiendo este orden, a la capa de entrada de datos.

La capa de entrada de datos sirve únicamente como vía de ingreso del vector de entrada a la red neural, es decir, no realiza ningún tipo de cálculo

sobre los datos que recibe. Este vector de entrada es entregado a la primera capa de pesos de la red, ubicada entre la capa de entrada y la primera (y por lo general la única) capa escondida.

Cada vector de entrada esta acompañado de un vector de salida deseada y a ambos se les conoce como el par de entrenamiento.

En este momento se realiza el primer procedimiento de transformación de la información y es el que determina los valores de las señales netas de entrada de la capa escondida y se calcula de la siguiente forma:

$$Y_j = \sum_{k=1}^{n} X_k \ W_{k,j}$$

Ecuación 3: Cálculo de la entrada neta de una neurona

Donde ***n*** es el número de neuronas de la capa de entrada, Y_j representa la señal neta de entrada de la neurona ***j*** de la capa escondida, X_k representa la salida de la neurona ***k*** de capa de entrada y $W_{k,j}$ representa el enlace entre estas dos neuronas. Este cálculo se realiza a lo largo de la capa escondida para determinar todas las señales de entrada netas de todos los componentes en este nivel. Mas adelante se aplica la misma ecuación para el cálculo de las señales netas de entrada de la capa de salida de la red en función de las salidas de la capa escondida.

Una vez determinado el valor de la señal de entrada neta de todas las neuronas de la capa escondida (o de la capa de salida), se calcula la salida correspondiente aplicando la función sigmoidal a la entrada neta calculada inicialmente.

$$y = \frac{1}{1 + exp(-kx)}$$

Ecuación 4: Función sigmoidal

Donde ***y*** es la salida final de la neurona, ***x*** es la señal de entrada neta y ***k*** es un parámetro que determina la forma de la curva.

Este procedimiento de cálculo de entradas netas y salidas se realiza hasta que alcanzamos la salida de la red en la última capa de la misma. Es aquí donde se compara esta salida obtenida con la respectiva salida deseada y de esta comparación se determina la o diferencia de ambos para cada neurona de esta capa.

$$Diferencia = T - y$$

Ecuación 5: Diferencia o error en la salida de la red

Donde ***T*** es el vector de salida deseado e ***y*** es el vector de salida real, ambos obviamente en la capa de salida de la red.

Una vez obtenidos los componentes de este vector error, es decir los valores de las diferencias para cada neurona de la capa de salida, se calcula el error total de la red sumando los cuadrados de dichas diferencias.

$$Error_p = \sum_{i=1}^{k} \left(T_{p,i} - y_i\right)^2$$

Ecuación 6: Error total de la red

Donde ***k*** es el número de neuronas de la capa de salida que es igual al número de componentes del vector de salida deseado para el par de entrenamiento ***p***.

La matriz ***T*** esta formada por filas de vectores de salida deseados cada uno relacionado con un vector (o fila) de la matriz de entrada correspondiente formando así un par de entrenamiento.

Un paso importante en el proceso de entrenamiento de la red es el cálculo de la variación de la entrada en una neurona en función de la variación de la salida y para esto utilizamos la derivada de la función de activación. Esto se realiza para "pasar" el error que viene de la salida de la red hacia los enlaces previos a la capa de salida a través de las neuronas de esta capa. Como ya se indicó anteriormente, la derivada de la función de activación se expresa en términos de la misma función de esta manera obtenemos la variación total en la entrada de la neurona para cada variación en la salida:

$$\delta = y(1-y)(T-y)$$

Ecuación 7: Vector error en la capa de salida

Donde δ es el vector error, ***T-y*** representa la variación en la salida que ha de propagarse hacia atrás, ***y(1-y)*** es la derivada de la función de activación expresada en términos de ***y***. Todo esto para una iteración. Mas adelante veremos que este vector error se define de distinta manera cuando se trata de corregir los enlaces entre la capa de entrada y la capa escondida.

El proceso de transformación de información que se lleva a cabo en la red neural consiste pues de una serie de operaciones básicas entre matrices siguiendo una secuencia clara. De estas matrices, las que definen el estado de la red en un momento dado son las matrices de pesos, y el entrenamiento consiste en recalcular los valores de los componentes de estas matrices para cada par de entrenamiento presentado, cambiando así el estado de la red en cada iteración.

Para las correcciones de los pesos se empieza por la última capa o capa de salida en función del vector error δ calculado anteriormente, de la salida de las neuronas de la capa anterior o escondida y de un factor o coeficiente de entrenamiento η.

$$\Delta w_{i,j} = \eta \delta_j y_i$$

Ecuación 8: Corrección de los enlaces de la capa de salida

Donde $w_{i,j}$ representa al enlace entre la neurona ***i*** de la capa escondida y la neurona ***j*** de la capa de salida, δ_j es el componente ***j*** del vector error, y_i es la salida de la neurona ***i*** de la capa escondida. Esta ecuación permite calcular la variación del peso en cuestión, de tal forma que el nuevo valor del enlace (***i,j***) se obtiene de sumar esta corrección al valor anterior del mismo de acuerdo a la siguiente ecuación:

$$w_{i,j}(t+1) = w_{i,j}(t) + \Delta w_{i,j}$$

Ecuación 9: Enlaces corregidos

De esta forma obtenemos el valor del enlace entre la neurona ***i*** de la capa escondida y la neurona ***j*** de la capa de salida para la iteración ***t+1*** en función de su valor anterior y de la variación calculada previamente.

En el caso de la capa escondida, es decir, la corrección de la siguiente capa de enlaces, no tenemos un vector ***T*** de salida deseada con el cual comparar la salida de esta capa y calcular el valor correspondiente del vector error δ. Para esto utilizamos los valores de δ de la capa de salida multiplicados por los respectivos enlaces que comunican a todas las neuronas de la capa de salida con la neurona de la capa escondida para la cual hacemos este cálculo. La siguiente ecuación se utiliza para determinar δ para las neuronas de la capa escondida:

$$\delta_i = y_i(1 - y_i)(\sum_{j=1}^{k} \delta_j w_{i,j})$$

Ecuación 10: Vector error en la capa escondida

Donde δ_i es el componente ***i*** del vector error de la capa escondida, y_i es la salida de la neurona ***i*** de dicha capa, δ_j en la sumatoria representa el componente ***j*** del vector error de la capa de salida, $w_{i,j}$ es el enlace entre la neurona ***i*** de la capa escondida y la neurona ***j*** de la capa de salida.

Adicionalmente, se añade un término llamado "momentum" a la ecuación 9 proporcional a la variación de los enlaces en la iteración anterior a la actual para mejorar la tendencia del entrenamiento de la red. La ecuación queda de la siguiente forma:

$$w_{i,j}(t+1) = w_{i,j}(t) + \Delta w_{i,j} + a(w_{i,j}(t) - w_{i,j}(t-1))$$

Ecuación 11: Enlaces corregidos con término de momentum

Donde ***a*** es el coeficiente de momentum y multiplica a la variación de los enlaces de la iteración previa a la actual.

La ecuación 11 es la misma para la corrección de los enlaces tanto para la capa de salida como para la capa escondida.

La anterior secuencia de ecuaciones se aplica cada vez que un par de entrenamiento es presentado a la red neural. Los enlaces se van corrigiendo paulatinamente a medida que iterativamente se van presentando los pares de entrenamiento a la red.

6. ALGORITMO DE PROGRAMACION DEL METODO DE RETROPROPAGACION

A continuación explicaremos como se realiza el entrenamiento de la red neural. Para esto consideraremos una red de retropropagación con una capa intermedia ya que es la que mejor combina la simplicidad de su arquitectura y entrenamiento con una elevada capacidad de generalización y predicción.

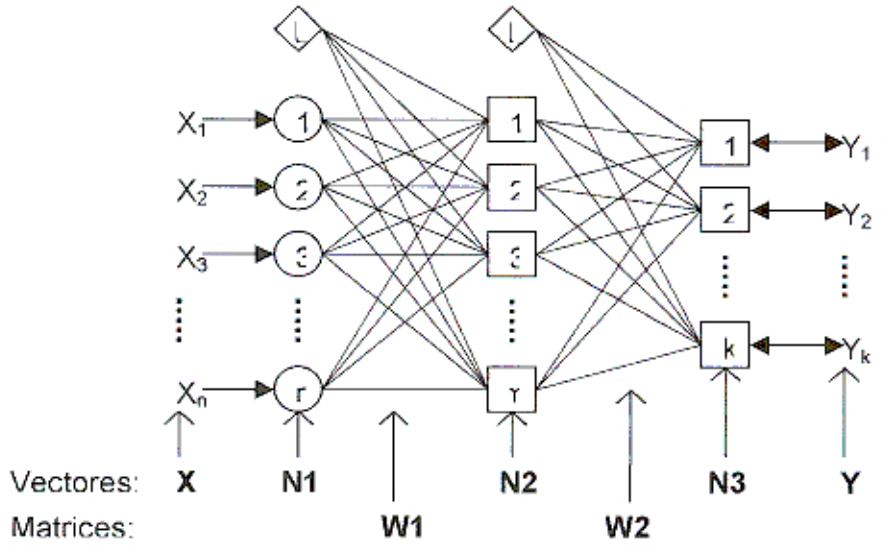


Figura 6: Estructura de la Red Neural

Una red neural de retropropagación se define por lo siguiente:

1. El número de capas.
2. El número de neuronas por capa.

Durante el entrenamiento y para cada iteración, las salidas de las neuronas y las conexiones entre ellas tienen valores determinados, para manejarlos se definen los siguientes vectores y matrices:

1. Vectores de estados de neuronas por capa, ***N1***, ***N2*** y ***N3***. Estos vectores contienen los valores de salida de todas las neuronas de la red en sus tres capas respectivamente.

2. Matrices de valores de enlaces, ***W1*** y ***W2***. Aquí se almacenan los valores de los pesos entre las neuronas de la red.
3. Vectores de entrenamiento ***X*** de entrada e ***Y*** de salida. Cada iteración requiere un vector de entrada y su correspondiente vector de salida, y el número total de pares de entrenamiento ***X,Y*** forman las matrices de entrenamiento totales considerando ***p*** pares de entrenamiento.
4. Otros vectores y matrices se generan a partir de los arriba mencionados como parte del algoritmo de entrenamiento.

$$X=\begin{vmatrix} X_{11} & X_{12} & X_{13} & \dots & X_{1n} \\ X_{21} & X_{22} & X_{23} & \dots & X_{2n} \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ X_{p1} & X_{p2} & X_{p3} & \dots & X_{pn} \end{vmatrix} \qquad Y=\begin{vmatrix} Y_{11} & Y_{12} & Y_{13} & \dots & Y_{1k} \\ Y_{21} & Y_{22} & Y_{23} & \dots & Y_{2k} \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ Y_{p1} & Y_{p2} & Y_{p3} & \dots & Y_{pk} \end{vmatrix}$$

Figura 7: Matrices de entrenamiento y par de entrenamiento resaltado para p=2

Hay ***p*** pares de entrenamiento y para cada uno hay un vector de entrada ***X*** y un vector de salida ***Y***. Nótese que el vector ***X*** tiene ***n*** componentes y el vector ***Y*** tiene ***k*** componentes correspondientes a los números de neuronas en las capas de entrada y salida respectivamente.

Asimismo, los enlaces entre las neuronas de la red neural se ordenan en matrices las cuales están compuestas por los valores de los respectivos pesos para cada iteración durante el entrenamiento.

$$W1 = \begin{vmatrix} W1_{11} & W1_{12} & W1_{13} & \dots & W1_{1m} \\ W1_{21} & W1_{22} & W1_{23} & \dots & W1_{2m} \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ W1_{n1} & W1_{n2} & W1_{n3} & \dots & W1_{nm} \end{vmatrix} \qquad W2 = \begin{vmatrix} W2_{11} & W2_{12} & W2_{13} & \dots & W2_{1k} \\ W2_{21} & W2_{22} & W2_{23} & \dots & W2_{2k} \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ W2_{m1} & W2_{m2} & W2_{m3} & \dots & W2_{mk} \end{vmatrix}$$

Figura 8: Matrices de pesos

Los valores de las salidas (estados) de las neuronas se agrupan también en vectores.

$$N1 = \{ N1_1, N1_2, N1_3, \dots, N1_n \}$$
$$N2 = \{ N2_1, N2_2, N2_3, \dots, N2_m \}$$
$$N3 = \{ N3_1, N3_2, N3_3, \dots, N3_k \}$$

Figura 9: Vectores de estado de las tres capas de neuronas

Los otros vectores y matrices generados a partir de los ya definidos como parte del entrenamiento son:

WA1, WA2: Matrices de pesos de la iteración anterior

WT1, WT2: Matrices de pesos temporales

VD2, VD3: Vectores diferencia

E2, E3: Vectores error

Las constantes usadas para este algoritmo son:

c: coeficiente de entrenamiento

a: momentum

U: potencial de umbral

nsig: constante de la función sigmoidal

El error total de la red se calcula al final de cada aplicación del vector de entrada ***X***, y se determina sumando los cuadrados de las diferencias de los componentes de los vectores ***Y*** y ***N3***.

$$Error = \sum (Y - N3)^2$$

Ecuación 12: Error total de la red

Adicionalmente se define el vector de errores cuyos componentes son los errores de los pares de entrenamiento y se usa sólo para determinar que par de entrenamiento se usará a continuación.

A continuación se detallarán los pasos del algoritmo:

1. Iniciar los valores:
   W1, W2 = Random
   WA1, WA2 = 0
2. Para cada iteración (par de entrenamiento)

a) Leer el par de entrenamiento (X,Y): Para escoger el par de entrenamiento en cada iteración puede optarse por simplemente establecer un lazo recursivo que lea en forma secuencial los pares de entrenamiento. A este lazo puede añadírsele la condición de "saltarse" aquellos pares que tengan el menor error. Finalmente, se optó por usar el par de entrenamiento con mayor error para cada iteración.

b) Asignar N1 = X : Simplemente pasa los valores del vector **X** a las neuronas de la capa de entrada.

c) Calcular N2 = Sigmoid(N1.W1 – U) : Con los valores de los componentes de **W1** y **N1** se determinan las entradas netas de la siguiente capa de la red para luego aplicarle la función de transferencia (sigmoidal) y obtener las salidas de dicha capa de la red. El producto escalar **N1.W1** esta definido como la sumatoria de los productos de los respectivos componentes de ambos vectores:

$\Sigma N_i W_i$. Nótese que para cada componente de **N2** se debe multiplicar escalarmente el vector **N1** por cada columna de la matriz **W1**.

d) Calcular N3 = Sigmoid(N2.W2 – U) : De forma similar al paso anterior. Se le resta el valor de umbral **U** a todas las neuronas de la red para aumentar su capacidad de clasificación.

e) Calcular VD3 = Y – N3 : Obtenido el vector de salida **N3**, éste se compara con el vector de salida deseado **Y** correspondiente por supuesto a su respectivo par de entrenamiento **X** con el cual se inició esta iteración. Simplemente se restan los respectivos componentes de ambos vectores.

f) Calcular el error de la red. Error = $\sum(Y - N3)^2$: Conocidas las diferencias del paso anterior, estas se elevan al cuadrado y se suman para obtener el error total de la red. también se almacenan estos cuadrados en un vector de errores para determinar el orden de lectura de los pares de entrenamiento de acuerdo a su respectivo error.

g) Verificar el máximo componente del vector errores : Se utiliza el par de entrenamiento que tenga el máximo error para la siguiente iteración.

h) Calcular E3 = N3.(1 - N3).VD3 : Se calcula el vector error con cada componente de **N3** y **VD3**. Este error es una función de la salida de la neurona y su respectiva diferencia con la salida deseada.

i) Calcular WT2 = W2 + c.N2.E3 + a.(W2 – WA2) : Este es el paso más importante del entrenamiento ya que aquí se corrigen los valores de los enlaces entre las neuronas. La matriz **WT2** se obtiene del valor actual **W2** y del anterior a este **WA2**. también se utiliza el vector **N2** de **m** componentes y **E3** de **k** componentes, y de acuerdo a esto se realizan los productos **N2.E3** correspondientes a cada componente de la matriz **WT2**(**m** por **k**). La diferencia **W2-WA2** es el vector de variación de pesos y sirve para "guiar" al algoritmo en la dirección de la variación de los pesos.

j) Calcular WA2 = W2 ; W2 = WT2 : Se hacen las asignaciones para la siguiente iteración.

k) Calcular VD2 = E3.W2 : Para la capa intermedia se transmite el error de **E3** de la capa de salida a través de los pesos **W2** pero esta vez en sentido inverso (retropropagación de error). Se efectúa el producto escalar del vector **E3** por la respectiva fila de la matriz **W2** obteniéndose así los m componentes del vector **VD2**.

l) Calcular E2 = N2.(1 - N2).VD2 : El vector error de la capa intermedia se calcula de igual manera que su equivalente en la capa de salida.

m) Calcular WT1 = W1 + c.N1.E2 + a.(W1 – WA1) : Aquí también se combinan los componentes de **N1**(**n**) y **E2**(**m**) para obtener los componentes de la matriz **WT1**(**n** por **m**).

n) Calcular WA1 = W1 ; W1 = WT1

3. Si el error total es mayor que el máximo permitido entonces se realiza la siguiente iteración. En caso contrario se sale del loop de entrenamiento, se graban los archivos con los resultados y fin del programa.

7. APLICACION PRACTICA DE LA TEORIA DE REDES NEURALES

Como se mencionó anteriormente la aplicación de este método es de especial interés en áreas donde se cuenta con abundante información que puede ser utilizada para establecer las correlaciones sobre las cuales determinar alguna propiedad de interés en puntos que carecen de información acerca de la misma. El problema que se plantea en este estudio es una clara aplicación de las redes neurales para el desarrollo de dicha correlación.

1. OBJETIVO

El objetivo de esta aplicación práctica es la generación de la curva de porosidad neutron a partir de las curvas Rayos Gamma e Inducción Eléctrica con un buen grado de efectividad y que pueda ser utilizada satisfactoriamente en el análisis petrofísico de las propiedades del reservorio. Para lograr esto es necesario elaborar un programa basado en la teoría de redes neurales presentada anteriormente y que produzca resultados comparables con los registros tomados directamente por las compañías de servicios.

3. SITUACION ACTUAL

Desde los comienzos del desarrollo de los campos petroleros en el noroeste del Perú se ha recopilado abundante información de todo tipo en cada etapa de su vida productiva, como de producción, completación, geología, geofísica, etc. Un subconjunto de este universo de información es el referente a los registros eléctricos de los pozos (well logging), y uno de los registros más importantes es el de porosidad neutrón (NPHI).

Dicha curva no se encuentra en la mayoría de pozos del noroeste del Perú como sí lo hacen la curva de Rayos Gamma y las de Inducción. Por lo tanto,

se cuenta con suficiente cantidad de información sobre la cual entrenar la red para establecer las correlaciones respectivas.

C. COMENTARIO SOBRE LA SITUACION ACTUAL

Resumiendo lo dicho anteriormente tenemos:

1. Una cantidad apreciable de pozos que tienen Rayos Gamma e Inducción Eléctrica (Resistividad).
2. Una cantidad de pozos que además tienen la curva de Porosidad Neutrón.

Estas dos situaciones que se presentan en los campos del noroeste del Perú hacen de la aplicación de redes neurales para generar la curva NPHI, una excelente alternativa para incrementar la información del pozo y añadir valor a la información total del reservorio con el objetivo de mejorar la calidad de la caracterización y por lo tanto ayudar a la toma de decisiones.

D. DESCRIPCION DEL PROBLEMA PASO A PASO

Resumidamente el proceso para resolver el problema se puede expresar de la siguiente manera:

1. Elección de los puntos para el entrenamiento de la red.
2. Presentación de los puntos uno por uno mientras la red se va entrenando (ajustando los valores de los enlaces).
3. Una vez alcanzado un error máximo permisible, se detiene el entrenamiento. En nuestro caso el error máximo permitido es de 0.01.
4. Se generan las curvas de porosidad neutrón de los pozos de prueba.
5. Se comparan ambas curvas y si el resultado es aceptable se generan las curvas de porosidad neutrón de los pozos incógnita.

E. DESARROLLO DEL PROBLEMA

Se plantea generar la curva de porosidad neutrón en dos pozos adyacentes y para la formación Mogollón en el lote Z2A-B del noroeste del Perú. Los pozos serán llamados H e I. Dichos pozos cuentan con sus respectivas curvas Rayos Gamma e Inducción. Estas curvas serán nuestras variables de apoyo para la solución del problema.

Por otro lado se tiene seis pozos que cuentan no sólo con las curvas Rayos Gamma e Inducción eléctrica sino que también tienen la curva porosidad neutrón. Tres de estos pozos (L, N y O) pertenecen al mismo bloque reservorio de los pozos incógnita, mientras que los restantes tres (D, E y M) pertenecen a bloques contiguos.

Los intervalos correspondientes a la formación Mogollón son respectivamente en pies medidos:

Pozo D: 7570 – 6608
Pozo E: 6045 – 5496
Pozo H: 5025 – 4530
Pozo I: 6250 – 5080
Pozo L: 5850 – 5161
Pozo M: 5527 – 5000
Pozo N: 5390 – 4595
Pozo O: 4950 – 4300

I. DATOS PARA EL ENTRENAMIENTO

Los datos para el entrenamiento se tomaron de los pozos L, N y O sobre la totalidad del intervalo correspondiente a la formación Mogollón.

Dado que el proceso que se lleva a cabo en las neuronas artificiales involucra a la función exponencial, es necesario preparar los datos de tal forma que se evite posibles problemas de desbordamiento (overflow) durante la ejecución del programa. Para resolver esto se procede a normalizar la información de entrada sobre un rango determinado que en nuestro caso estuvo entre 0.1 y 0.9.

La figura 10 muestra el gráfico de los datos de entrada normalizados y sus respectivas proyecciones sobre los tres distintos planos.

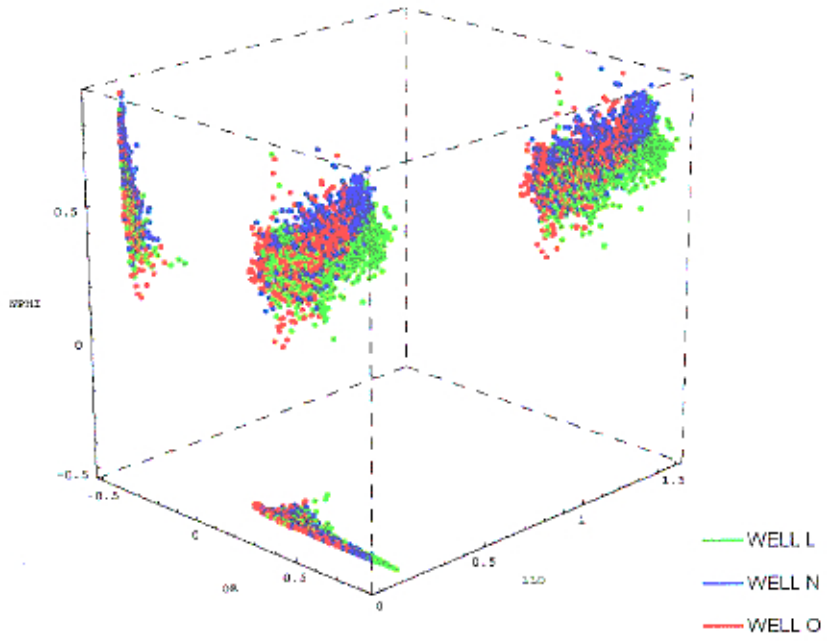


Figura 10: Proyecciones de los datos normalizados de los pozos de entrenamiento

La selección de los puntos para entrenar la red neural debe considerar el representar eficientemente a la totalidad de las curvas y no es necesario utilizar todos los puntos sino que un subconjunto de este puede ser suficiente para lograr un buen entrenamiento. Para mejorar esta selección de puntos se utilizan los gráficos de proyección de la figura 10 para discriminar las tendencias más notorias y que representan mejor a todo el conjunto.

Una vez determinados los puntos para el entrenamiento se preparan los pares de vectores para ser presentados a la red neural. En este caso se prepararon 66 pares de entrenamiento.

El primer paso para entrenar la red neural es la selección de los puntos de entrenamiento. Este total de puntos se subdividió en:

1. Puntos elegidos basándose en las proyecciones de la figura 10.
2. Puntos elegidos aleatoriamente de la nube de puntos.
3. Puntos elegidos basándose en los resultados de las primeras ejecuciones para refinar el resultado de las ejecuciones futuras.

Los puntos elegidos fueron:

Pozo	Prof.
L	5205
L	5232
L	5251
L	5312
L	5313
L	5314
L	5404
L	5487
L	5488
L	5497
L	5596
L	5597
L	5598
L	5629

Pozo	Prof.
L	5645
L	5646
L	5647
L	5662
L	5810
N	4628
N	4649
N	4658
N	4721
N	4782
N	4783
N	4784
N	4785
N	4794

Pozo	Prof.
N	4824
N	4840
N	4874
N	4923
N	4926
N	4927
N	4983
N	5144
N	5163
N	5188
O	4319
O	4320
O	4341
O	4353

Pozo	Prof.
O	4354
O	4355
O	4474
O	4475
O	4519
O	4533
O	4603
O	4604
O	4657
O	4668
O	4669
O	4670
O	4713
O	4722

Pozo	Prof.
O	4723
O	4724
O	4750
O	4826
O	4827
O	4838
O	4839
O	4840
O	4842
O	4843

II. EL PROGRAMA

Se desarrolló la aplicación en Visual Basic de acuerdo al algoritmo presentado en el capítulo anterior. La figura 11 muestra el panel de interacción del programa.

En cada ejecución del programa se leen iterativamente los componentes de las matrices de entrenamiento de acuerdo al criterio de mayor error y se presentan a la red neural, al final de la ejecución actual se calcula el error total de la red. El programa considera múltiples entrenamientos por ejecución, es decir, la red se entrena sobre el conjunto de vectores de entrenamiento y llega a un error total determinado, luego genera los archivos

de salida correspondientes. La ejecución se repite una y otra vez y sólo se generarán (reemplazarán) los archivos si el nuevo error total es menor que el de la ejecución que generó los actuales archivos.

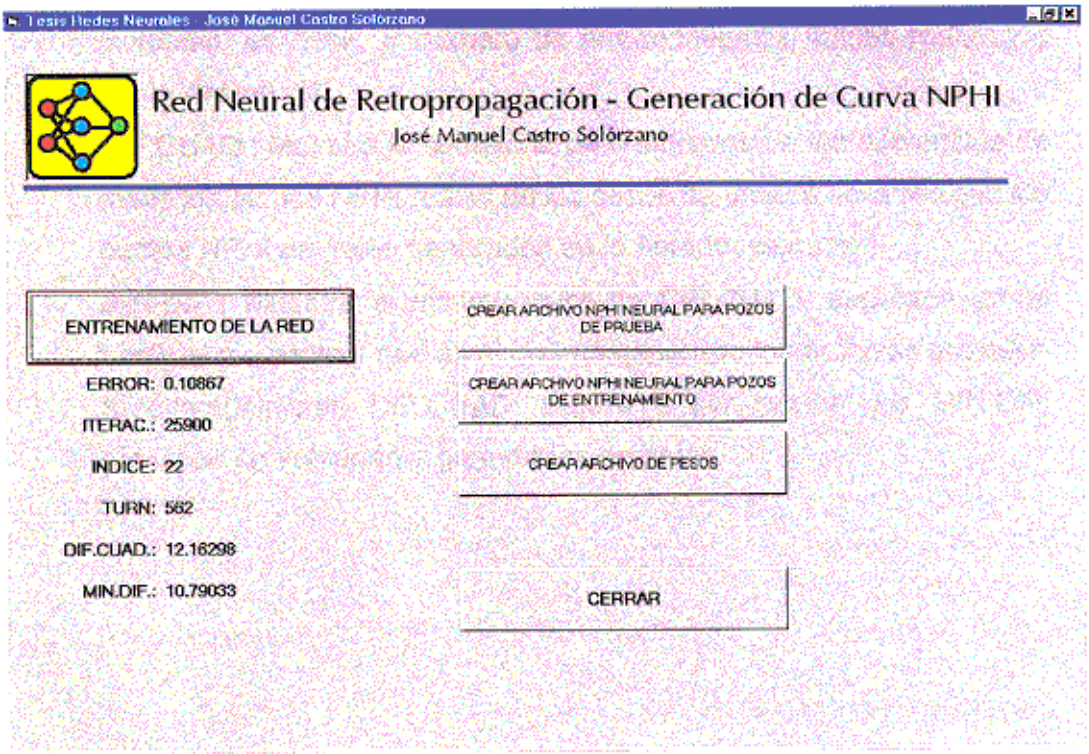


Figura 11: Panel de interacción del programa

El código incluye instrucciones para el ingreso de información ya sea para el entrenamiento de la red como para la generación de archivos con las curvas porosidad neutrón normalizadas.

Asimismo, el panel de interacción del programa muestra información del entrenamiento en tiempo real de ejecución como:

- ERROR: Muestra la magnitud del componente del vector de errores correspondiente al par de entrenamiento que esta siendo presentado a la red neural.
- ITERAC.: Muestra el valor de la iteración para la ejecución actual, en otras palabras, la cantidad de vectores de entrenamiento presentadas a la red durante la actual ejecución.

- INDICE: Muestra el índice del componente de la matriz de entrenamiento que esta actualmente siendo presentada a la red neural.
- TURN: Muestra el número de ejecuciones acumulado hasta el momento, es decir, el número de entrenamientos totales realizados por la red neural.
- DIF.CUAD.: Muestra la sumatoria de cuadrados de las diferencias de todos los puntos NPHI reales de los pozos de prueba de la red con los puntos NPHI neurales generados en la anterior ejecución.
- MIN.DIF.: Muestra el mínimo valor de DIF.CUAD. explicado arriba hasta el momento y que es el que ha generado los archivos actuales. Si eventualmente DIF.CUAD. llegara a ser menor que MIN.DIF. entonces se volverían a generar los archivos.

F. RESULTADOS

El final del entrenamiento de la red neural produce un conjunto de valores para los enlaces de la misma, dichos valores serán usados para generar las curvas de porosidad neutrón tanto para los pozos de prueba como para los pozos H e I.

Los valores de los enlaces son los siguientes:

w1(1, 1) = 0.73179
w1(2, 1) = 3.35966
w1(1, 2) = 0.77351
w1(2, 2) = -1.1702
w1(1, 3) = 0.43704
w1(2, 3) = 2.85044
w1(1, 4) = -0.09224
w1(2, 4) = -0.10917
w1(1, 5) = 0.45826
w1(2, 5) = 1.19691
w1(1, 6) = 1.03292
w1(2, 6) = -1.36531
w1(1, 7) = 0.16664
w1(2, 7) = 1.93725
w1(1, 8) = 0.9755
w1(2, 8) = 0.54817
w1(1, 9) = 0.30384
w1(2, 9) = 0.86088
w1(1, 10) = 0.44025
w1(2, 10) = 2.26277
w1(1, 11) = 0.61436
w1(2, 11) = 0.36024
w1(1, 12) = 0.15263
w1(2, 12) = 0.20075
w1(1, 13) = 0.43289
w1(2, 13) = -0.25584
w1(1, 14) = 1.05025
w1(2, 14) = 0.83964
w1(1, 15) = 0.66896
w1(2, 15) = -2.08101
w1(1, 16) = 0.27005
w1(2, 16) = 1.86372
w1(1, 17) = 1.45788
w1(2, 17) = -1.75488
w1(1, 18) = 1.94831
w1(2, 18) = -2.75603
w1(1, 19) = -0.08993
w1(2, 19) = 0.87726
w1(1, 20) = 0.45592
w1(2, 20) = 0.14942
w2(1, 1) = -2.61601
w2(2, 1) = 1.79171
w2(3, 1) = -2.14837
w2(4, 1) = 0.62702
w2(5, 1) = -0.61628
w2(6, 1) = 1.90184
w2(7, 1) = -1.33008
w2(8, 1) = -0.03386
w2(9, 1) = -0.23746
w2(10, 1) = -1.48952
w2(11, 1) = 0.15105
w2(12, 1) = 0.46094
w2(13, 1) = 0.76108
w2(14, 1) = -0.31207
w2(15, 1) = 2.70257
w2(16, 1) = -1.27814
w2(17, 1) = 2.37389
w2(18, 1) = 3.3401
w2(19, 1) = -0.36732
w2(20, 1) = 0.39822

El primer subíndice hace referencia a la neurona de origen y el segundo a la neurona de destino. W1 es la matriz de pesos entre la capa de entrada de datos y la capa escondida. W2 es la matriz de pesos entre la capa escondida y la capa de salida, es decir, por ejemplo w2(9,1) representa el enlace entre la neurona 9 de la capa intermedia y la neurona 1 (en este caso la única) de la capa de salida.

Con estos valores de enlaces se obtienen los resultados mostrados en el anexo 2 para los pozos de prueba D, E y M. Fueron necesarias 27519 iteraciones sobre los 66 pares de entrenamiento, lográndose un error total (sumatoria de los cuadrados de las diferencias en cada punto de la curva) de 10.7611 para los 2041 puntos de dichos pozos lo que hace un error promedio por punto de 0.0053. Como este error promedio proviene de los cuadrados de las diferencias, el error promedio de la porosidad es de +/- 0.073 ó 7.3% por punto.

Los mismos puntos fueron analizados utilizando un algoritmo de optimización no lineal conocido como GRG2 desarrollado por la Universidad León Lasdon de Austin (Texas) y la Universidad Allan Waren (Cleveland) y que viene incorporado en el programa Microsoft Excel. Los resultados de este método arrojan un error de +/- 13.46% por punto.

Adicionalmente, se hizo una comparación de las curvas reales y generadas basándose en valores límites y los resultados se muestran en las tablas y gráficos siguientes.

	POZO D		POZO E		POZO M	
VALOR LIMITE	REAL	NEURAL	REAL	NEURAL	REAL	NEURAL
>0.05	957	955	547	527	528	527
>0.1	884	929	515	497	523	520
>0.15	663	762	395	417	506	492
>0.2	420	434	215	273	449	453
>0.25	219	54	21	33	267	207
>0.3	30	0	0	0	11	0

Tabla 2: Comparación de Valores Límites entre las curvas NPHI real y generada por la red neural.

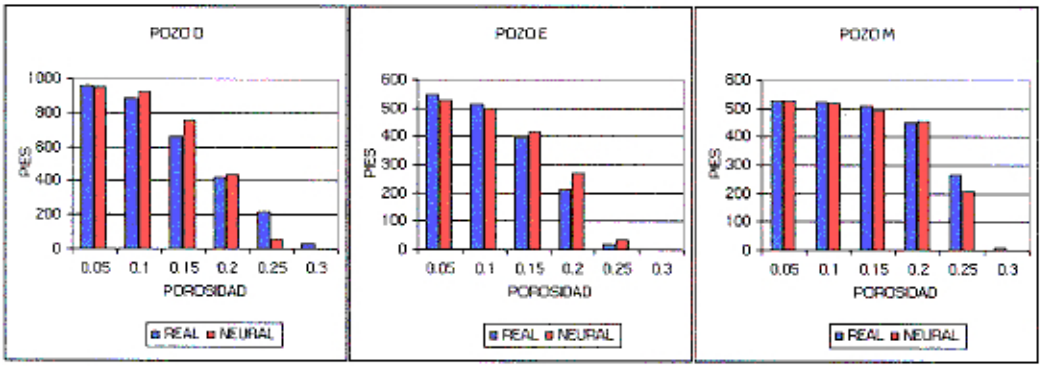


Figura 12: Comparación de Valores Límites entre las curvas NPHI real y generada por la red neural.

Desde el punto de vista cuantitativo de conteo de pies mediante Valores Límites se puede comparar ambas curvas de porosidad neutrón. Los gráficos de la figura 12 muestran que existe una buena afinidad en los resultados de conteos para cada Valores Límites.

El siguiente paso consiste obviamente en generar las curvas de porosidad neutrón para los pozos incógnita H e I. Para esto utilizamos los valores de los enlaces de la red neural finalmente entrenada y tomando los datos Rayos Gamma e Inducción de estos pozos calculamos sus respectivos NPHI neurales. Luego, los datos de salida de la red neural, al estar normalizados, se deben someter al proceso inverso para tener así los valores reales de las porosidades neutrón respectivas. El anexo 2 muestra estos resultados.

8. EVALUACION ECONOMICA DE LA APLICACION

En la definición del objetivo del presente estudio se propone la aplicación de esta herramienta como una alternativa a la obtención de información del pozo. La otra alternativa evidentemente es la toma directa de datos mediante registros radiactivos de porosidad.

Ambas alternativas difieren sustancialmente en sus aproximaciones y por lo tanto en sus esquemas de costos. La siguiente tabla muestra una comparación entre las dos alternativas en lo referente a requerimientos y condiciones para la obtención de la información.

	Red Neural Artificial	Registro de Porosidad Tomados con Herramientas a Cable
Herramientas	• Computadora personal. • Impresora.	• Herramienta radiactiva de porosidad y dispositivos de transmisión y recepción. • Computadora e impresora. • Unidad a Cable. • Unidad de servicio de pozos, lubricador, válvula de baleo, etc.
Condiciones	• Ninguna.	• Las requeridas para el correcto registro.
Información	• Registros de pozos vecinos digitizados.	• Rayos Gamma y CCL para calibrar en profundidad.
Personal	• Un especialista.	• Un especialista. • Personal de Unidad a Cable. • Personal de Servicio de Pozos.

Tabla 3: Composición de costos

Los costos provenientes del alquiler de los servicios de toma de registros, y de Unidad de Servicio de Pozos consideran los puntos arriba mencionados en Herramientas y personal. En el punto correspondiente a condiciones se considera el estado actual del pozo, es decir, si requiere un servicio adicional al registro (sacar instalación de producción, calibrar, etc). Por otro lado también hay que considerar el hecho de que si el pozo es productor o inyector se deberá diferir su status mientras dure la toma del registro.

La composición de los costos mostrada en la figura 14 indica claramente la diferencia en los requerimientos de herramientas y personal entre las dos aproximaciones a favor de la generación de la curva por medio de redes neurales. Traduciendo esta tabla a números obtenemos lo siguiente por pozo:

Registro de porosidad con Unidad a Cable (US $):	
Acondicionamiento del pozo con unidad de servicio (antes)	4200.00
Alquiler del servicio de toma de registro	8000.00
Producción o inyección diferida del pozo (*)	1800.00
Acondicionamiento del pozo después del servicio	4200.00
Total por pozo	**18200.00**

(*) Considerando una producción de 20 bbl/dia, a US $ 30 por barril durante 3 días.

Generación de la curva con redes neurales (US $):	
Alquiler del servicio de generación de curvas	1000.00
Total por pozo	**1000.00**

Tabla 4: Comparación de costos de las dos opciones para la obtención de la curva de porosidad neutrón.

9. CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

a. Es factible la generación de la curva de porosidad neutrón a partir de las curvas Rayos Gamma e Inducción como variables de apoyo utilizando información de pozos vecinos para establecer una correlación que permita cumplir con el objetivo de manera efectiva.
b. Son de especial interés para la aplicación de esta tecnología los campos en donde se dispone de información suficiente y un gran número de pozos, por ejemplo, el nororeste del Perú.
c. Es económicamente factible esta aproximación cuando se la compara con la toma del correspondiente registro utilizando herramientas radiactivas. Además, no se requiere de condiciones especiales de campo para la obtención de la curva. Esto es de especial interés ya que es posible elaborar programas de generación de curvas en cualquier momento y locación independientemente de las actuales condiciones en el campo.
d. Puede mejorarse la aproximación de la curva de porosidad neutrón generada si se dispone de curvas adicionales para ser usadas como variables de apoyo, por ejemplo, efecto fotoeléctrico, densidad. La adición de mas variables de apoyo conduce a tener una red neural de mas dimensiones (mas neuronas de entrada) lo que disminuiría su velocidad de operación, sin embargo esta disminución resulta de poca importancia comparada con la calidad del registro generado.

RECOMENDACIONES

a. Promover el estudio y la investigación de herramientas de procesamiento de información de este tipo las cuales cada vez van teniendo mas aplicación en la industria del petróleo.
b. Incentivar la aplicación de estas herramientas en las empresas operadoras, especialmente aquellas que cuentan con la información de campo necesaria.
c. Promover la investigación y el desarrollo de otras aplicaciones de interés como son la generación de curvas de permeabilidad, identificación detallada de litología, aplicaciones en sísmica, etc.

10. BIBLIOGRAFIA

1) Bharath R., Drosen J. *Neural Network Computing*. Windcrest/Mc Graw-Hill 1994.
2) Wasserman P. D. *Neural Computing. Theory and Practice.* Van Nostrand Reinhold 1989.
3) Rodgers J. *Object-Oriented Neural Networks in C++*. Academic Press 1997.
4) Chata A., Oporto S. *Algoritmos Para Redes Neurales*. Paradigma de Sistemas Vol. 2 N° 2 1994.
5) Winston P. H., *Inteligencia Artificial*. Addison-Wesley Iberoamiricana S.A. 1994.
6) Mohaghegh S., and Ameri S. *Artificial Neural Network As a Valuable Tool For Petroleum Engineers*. SPE 29220. Society of Petroleum Engineers 1995.
7) Mohaghegh S., Arefi R., Ameri S., and Rose D. *Design and Development of an Artificial Neural Network for Estimation of Formation Permeability*. SPE 28237. Society of Petroleum Engineers 1994.
8) Mohaghegh S., Arefi R., Ameri S., Hefner M. H. *A Methodological Approach For Reservoir Heterogeneity Characterization Using Artificial Neural Networks*. SPE 28394. Society of Petroleum Engineers 1994.
9) Mohaghegh S., Richardson M., Ameri S. *Virtual Magnetic Imaging Logs: Generation of Synthetic MRI Logs from Conventional Well Logs*. SPE 51075. Society of Petroleum Engineers 1998.
10) Mohaghegh S., Popa A., Ameri S. *Intelligent Systems Can Design Optimum Fracturing Jobs*. SPE 57433. Society of Petroleum Engineers 1999.
11) White A. C., Molnar D., Aminian K., Mohaghegh S., Ameri S., Esposito P. *The Application of ANN for Zone Identification in a Complex Reservoir*. SPE 30977. Society of Petroleum Engineers 1995.
12) Balan B., Mohaghegh S., Ameri S. *State-Of-The-Art in Permeability Determination From Well Log Data: Part 1 - A Comparative Study, Model Development*. SPE 30978. Society of Petroleum Engineers 1995.

13) Mohaghegh S., Balan B., Ameri S. *State-Of-The-Art in Permeability Determination From Well Log Data: Part 2 - Verifiable, Accurate Permeability Predictions, the Touch-Stone of All Models*. SPE 30979. Society of Petroleum Engineers 1995.