ARTÍCULO DE INVESTIGACIÓN ORIGINAL

REVISTA MEXICANA DE INGENIERÍA BIOMÉDICA

Vol. XXII, Núm. 1
Enero-Marzo 2001
pp 4 - 11

Definición de una red neuronal para clasificación por medio de un programa evolutivo

Alma E. Martínez Licona*
John Goddard Close*

* Universidad Autónoma Metropolitana - Iztapalapa.

Autor responsable: Alma Martínez. Av. Michoacán y Purísima s/n Col. Vicentina México D.F. Tel/Fax: 58044640,
E-mail aaml@xanum.uam.mx

Artículo recibido 26/enero/2001
Artículo aceptado 24/marzo/2001

RESUMEN

Las redes neuronales artificiales llamadas Perceptrones Multicapas (PM) son una herramienta buena para la solución de problemas de clasificación. Se ha demostrado que los PM con una capa oculta en su arquitectura pueden separar satisfactoriamente las clases involucradas en un problema dado; sin embargo el número de nodos ocultos es desconocido ya que no existe un método para definirlos y depende mucho del problema a resolver. En el presente trabajo se explica cómo, por medio de un programa evolutivo (PE) con cromosomas de longitud variable, se puede encontrar el número adecuado de nodos ocultos, así como los pesos de todas las conexiones para definir por completo la arquitectura de un PM empleado en la solución de problemas de clasificación. Se supone únicamente que el número de nodos de entrada y de salida están fijos. Los operadores que se usan en el PE son selección y dos formas de mutación, los cuales se explican detalladamente en el artículo.

Palabras clave:
Perceptrones Multicapas, Programa Evolutivo, Cromosomas de longitud variable.

ABSTRACT

Artificial neural nets called Multi-layer Perceptrons (MP) are an extremely useful tool for solving classification problems. It has been shown that MPs with a single hidden layer can satisfactorily separate the classes involved in a given problem: however the number of hidden units required is unknown as there is no exact method for calculating them. The present work describes a way, using an evolutionary program (EP) with variable length chromosomes, of finding a suitable number of hidden units, as well as the weights on all the connections, thereby defining the architecture of a MP for a given classification problem. Only the number of input and output units are considered given.
The operators used by the EP are selection and two forms of mutation, and these are specified in the paper.

edigraphic.com

Key words:
Multi-layer Perceptrons, Evolutive program, Variable length chromosomes.

INTRODUCCIÓN

Las redes de neuronas artificiales llamadas Perceptrones Multicapas (PM) son una herramienta atractiva para solucionar problemas de clasificación como el reconocimiento de caracteres manuscritos, el reconocimiento de palabras habladas, y el diagnóstico de diferentes enfermedades. Para ilustrar este tipo de aplicación, y así mismo motivar el contenido del presente artículo, consideremos el caso del diagnóstico médico. Suponemos cierta familiaridad con los PM que el lector puede encontrar en[1,2].

Empezamos con un conjunto de pacientes clasificados como sanos o con alguna enfermedad dada, y queremos encontrar un PM que se pueda usar como un clasificador para predecir si un sujeto tiene la enfermedad o no. Por cada paciente tenemos información relevante para detectar la enfermedad en la forma de los valores de ciertas variables, como los valores de pruebas de laboratorio, quizá el sexo, peso, etcétera.

La determinación del PM generalmente involucra la elección de su arquitectura y la identificación de los pesos numéricos en las conexiones entre los nodos que mejor se ajusten a nuestro conjunto de pacientes. La elección de la arquitectura del PM es la decisión de cuántos nodos de entrada, capas ocultas, nodos ocultos de cada capa, y nodos de salida tendrá, así como las conexiones que habrá de una capa a otra. Para nuestro caso, las conexiones están definidas de cada nodo de entrada hacia cada uno de los nodos ocultos y de cada nodo oculto con cada nodo de salida; es decir, cada nodo está conectado a todos los nodos de la capa anterior. Normalmente el número de nodos de entrada está determinado por el número de variables que aparecen en el problema. Así mismo, como se está considerando dos posibles clases, podemos tomar un solo nodo de salida cuyos valores determinen si el paciente está sano o no. Realmente queda entonces la determinación de los nodos ocultos. Como no hay un método para escoger los nodos ocultos, una forma común de proceder es mediante un proceso tardado de acierto y error, intentar con diferentes valores y ver cuál PM da mejores resultados, después de encontrar sus pesos en cada caso. Para encontrar los pesos numéricos, habitualmente se hace uso del algoritmo de retropropagación. Este algoritmo está basado en el método de gradiente descendente y requiere una función de transferencia diferenciable, así como de fijar ciertos parámetros como la tasa de aprendizaje.

Se ha demostrado que los PM con una capa oculta tienen la capacidad de distinguir entre conjuntos de patrones complejos[3]. No obstante, como mencionamos anteriormente, no hay un método para determinar el número de nodos ocultos. El lector podría decir que evitamos dicha dificultad si se escoge un número "grande" de nodos ocultos, sin embargo, esto nos puede llevar a un problema importante.

El propósito del PM es fungir como un clasificador para predecir si un sujeto está sano o no. Nos interesa entonces lo que se llama "su poder de generalización" sobre los casos nuevos o, en otras palabras, qué tan bien clasifica sobre casos no vistos previamente. Cuando el número de nodos ocultos es demasiado grande, puede ocurrir un fenómeno donde el PM clasifica muy bien sobre los pacientes conocidos que se usaron para determinar los pesos, pero no sobre nuevos pacientes. Lo idóneo sería adoptar otro enfoque en la determinación de un PM que podría especificar todos los parámetros involucrados simultáneamente, tanto la arquitectura como los pesos numéricos. Es aquí donde entra la computación evolutiva para la solución a este problema. Hay un buen número de trabajos de investigación dedicados a aplicar ideas evolutivas a los PM y a redes neuronales artificiales más generales que han producido buenos resultados. Por ejemplo, Montana[4] ha usado algoritmos genéticos para escoger únicamente los pesos numéricos, mientras que Gruau[5] y Angeline, Saunders y Pollack[6] han presentado métodos para casos más generales. El lector puede encontrar una introducción al tema en Xin Yao[7].

El presente artículo describe un programa evolutivo (PE) con cromosomas de longitud variable y su aplicación a la determinación de los pesos, y la arquitectura de un PM para la solución de problemas de clasificación. Su definición está motivada por los artículos[4,6]. La organización del artículo es la siguiente: primero se describe el PE, explicando las estructuras que se ocuparon para su implementación así como los operadores empleados, después se presentan los resultados de ésta técnica aplicada a un ejemplo concreto y finalmente las conclusiones.

DESCRIPCIÓN DEL PROGRAMA EVOLUTIVO DE CROMOSOMAS DE LONGITUD VARIABLE

Los algoritmos genéticos y la computación evolutiva están formados por un conjunto de algoritmos que comparten la misma base conceptual de simular la evolución de una población de soluciones potenciales, llamadas cromosomas, usando operadores genéticos tales como selección, recombinación y mutación. En ambos casos hay una función de evaluación definida sobre los cromosomas que indica la aptitud de cada cromosoma, es decir, qué tan bueno es como solución del problema. Sin embargo, los algoritmos genéticos generalmente usan operadores de cruzamiento y mutación, dando mayor importancia al cruzamiento, mientras que la computación evolutiva utiliza diferentes formas de mutación. Otra diferencia importante es la forma de codificación de las soluciones potenciales del problema, las cuales se realizan mediante estructuras más naturales para el problema y no con números binarios. Eso usualmente implica que los operadores genéticos tienen que ser diseñados especialmente para el problema. El lector puede encontrar más sobre el área de la computación evolutiva en Fogel (8). Al igual que en el caso de los PM, el esquema puede parecer excesivamente sencillo comparado con el punto de vista biológico, sin embargo ha permitido soluciones a problemas complejos que no son fáciles de resolver con otras técnicas. En el resto de la sección damos una descripción de el PE comenzando con la forma de codificar los PM como cromosomas.

La codificación

Para aplicar un PE en un caso concreto, primero hay que decidir cómo se va a modelar el problema; esto implica la codificación de las soluciones potenciales del problema. En nuestro caso, la idea del PE es definir una población de cromosomas en donde cada cromosoma represente un PM. Consideremos al PM con una sola capa oculta donde el número de nodos ocultos pueden variar. El número de nodos de entrada y de salida son constantes, dependiendo del problema considerado. Un cromosoma deberá codificar el número de nodos ocultos que tendrá esa capa y los pesos que unen a los nodos de entrada con los de la capa oculta, y los de ésta hacia los nodos de salida, así como los pesos de los sesgos que se definieron en la capa de entrada y en la capa oculta. Esto nos lleva a la necesidad de considerar cromosomas con una longitud variable. Cada cromosoma, o PM, está implementado como una lista donde cada elemento de la lista representa conceptualmente un nodo oculto del PM.

La información que se guarda en un elemento de la lista son los valores de los pesos numéricos sobre las conexiones que llegan al nodo oculto de los nodos de entrada y los que salen de los nodos ocultos hacia los nodos de salida. Un cromosoma típico se muestra en la figura1 junto con el PM que representa.

La población está guardada en un arreglo de estructuras de tamaño *n* que define *n-1* cromosomas. La estructura mantiene información sobre la aptitud de cada cromosoma, lo cual se explicará más adelante, además de un apuntador a la lista del cromosoma. La posición *n* del arreglo se utiliza para dejar el mejor PM encontrado de una generación a otra (Figura 2).

El algoritmo principal

El algoritmo principal del PE se muestra en el cuadro 1.

Cuadro 1. Algoritmo PE.

```
Algoritmo principal

    Inicio
        Generación=0
        Inicializar la población
        Evaluar cada cromosoma
        Guardar el mejor cromosoma
        Mientras (generación<MAXGENS)
        Inicio
            Seleccionar la nueva población
            Aplicar Mutación Estructural
            Aplicar Mutación de los pesos
            Evaluar cada cromosoma
            Guardar el mejor cromosoma
            Generación=Generación+1
        Fin
    Fin
```

El algoritmo comienza con la inicialización de una población de cromosomas y la evaluación de cada uno de ellos. El tamaño de la población es fijo, aunque la longitud de cada cromosoma puede variar. La población evoluciona

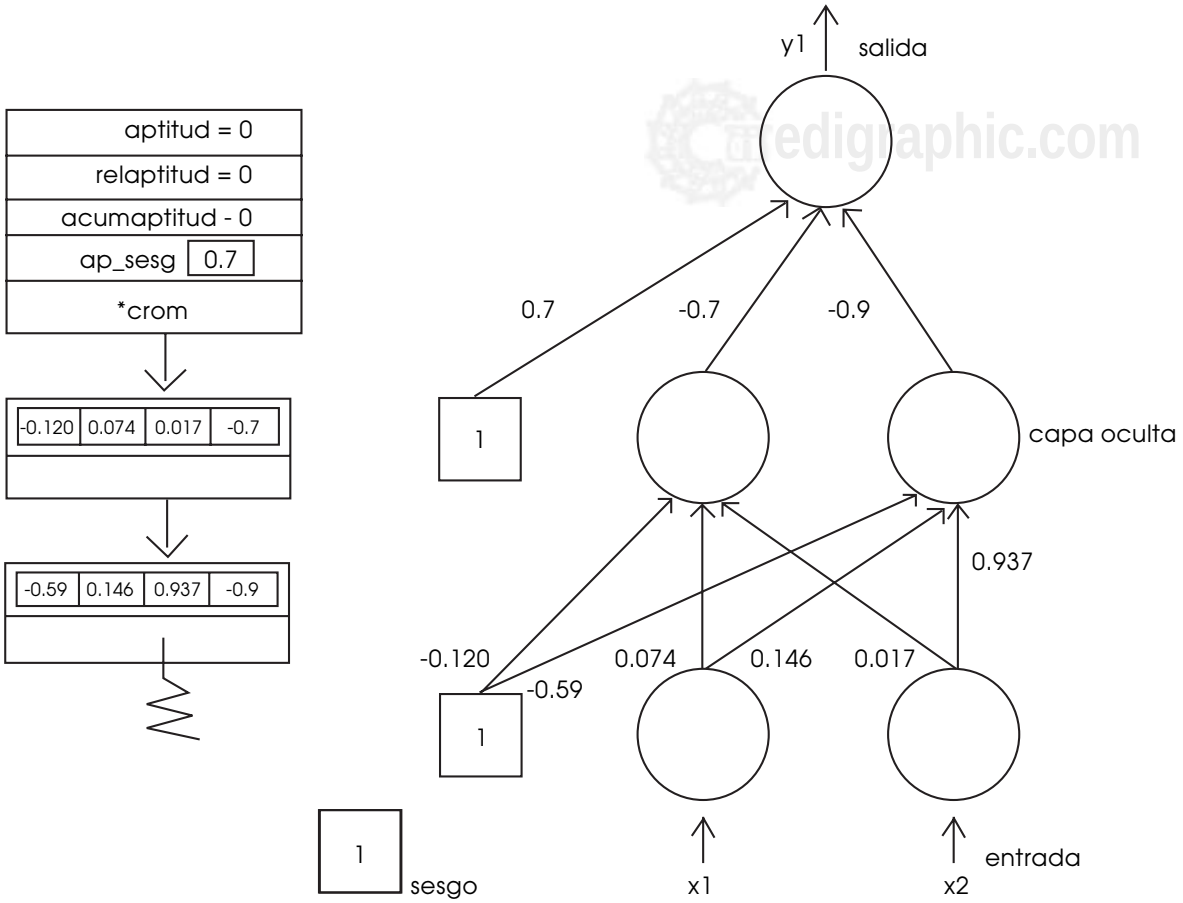

Figura 1. Representación de un PM mediante un cromosoma en el PE.

```
struct Genotype {
                    Nodo_oculto* crom;
                    float   ap_sesg(NSAL);
                    int     nmax;
                    float   aptitud;
                    float   relaptitud;
                    float   acumaptitud;
                    }
struct Genotype*población(tamaño+1)


struct Nodo_oculto  {
                    float
peso(NENT+NSAL);
                    Nodo_oculto*Sig_hor;
                    }
```

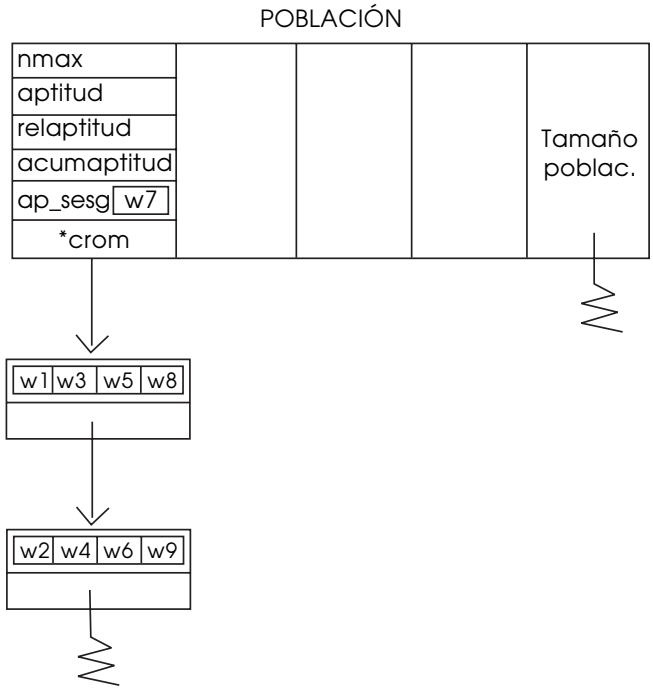

Figura 2. Estructuras de datos del PE.

varias generaciones aplicando los operadores de selección (para escoger la nueva población), mutación estructural (para modificar el número de nodos en un cromosoma), y mutación de los pesos. A continuación se describen los operadores con mayor detalle.

Inicialización de la población

Una función inicializa cada elemento de la población de la siguiente manera: el número de nodos ocultos representados en cada cromosoma se genera de manera aleatoria, cuidando que no exceda un número máximo definido por el usuario. De igual manera los pesos de cada nodo se inicializan de forma aleatoria con valores entre –1 y 1.

Función de evaluación

La función de evaluación debe indicar la aptitud de cada cromosoma, que en nuestro contexto significa qué tan bien el PM correspondiente al cromosoma clasifica un conjunto de datos. Consideremos dos diferentes maneras de calcular esta función, aunque hay otras posibilidades. Para explicarlas hay que definir la manera exacta en la cual un PM calcula su salida. En el caso del PM de la Figura 1, la entrada a cada uno de los nodos ocultos es el producto interno del vector de los pesos entrando en el nodo con el vector (1,x1,x2), dando:

$$-0.12+0.074*x1+0.017*x2$$
$$-0.59+0.146*x1+0.937*x2$$

La función de transferencia que se ocupa es la función sigmoide definida de la siguiente manera: f(u) = 1/(1+exp(-u)). Se aplica esta función de transferencia para calcular la salida de cada uno de los nodos ocultos dando:

$$z1=f(-0.12+0.074*x1+0.017*x2)$$
$$z2=f(-0.59+0.146*x1+0.937*x2)$$

Finalmente, para calcular la salida del nodo en la capa de salida, se vuelve a aplicar la función de transferencia al producto interno del vector de los pesos que entran en el nodo con el vector (1,z1,z2), dando:

$$y1=f((0.7-0.7*z1-0.9*z2)$$

La primera función de evaluación se basa en el error cuadrático medio, ECM, que comúnmente se usa en el algoritmo de retropropagación para determinar el error. El ECM se define de la siguiente manera:

$$ECM=\sum(yi-di)^2$$

donde la suma es sobre todos los patrones de los datos, *yi* es la salida del PM para el patrón *i* calculado como se describió anteriormente, y *di* es la salida deseada, que seria el valor que queremos asociar con el patrón.

La segunda función calcula cuántos errores tiene el PM al clasificar, es decir, cuántos datos están mal clasificados. Una ventaja del PE sobre un PM en este contexto es la posibilidad de usar funciones de aptitud no diferenciables, como esta. Para encontrar el mejor PM para cualquiera de las dos funciones de evaluación, hay que minimizar cada una de ellas. El campo de aptitud en la estructura de datos corresponde al valor de la función de evaluación.

Selección de la nueva población

Para escoger una nueva población se aplica el proceso conocido como la rueda de la ruleta, "sin embargo", es importante mencionar que no es el único método que existe. La idea esencial es particionar un disco en sectores en proporción a la aptitud de cada cromosoma. Se gira el disco, y de acuerdo al sector donde éste para, se escoge el cromosoma correspondiente. Esto se lleva a cabo de la siguiente manera. Primero encuentra la aptitud total de la población sumando todas las aptitudes menos el del último cromosoma. Después calcula la aptitud relativa de cada cromosoma dividiendo la aptitud de ese cromosoma entre la aptitud total. Luego, con la aptitud relativa calcula la aptitud acumulada: se inicializa la aptitud acumulada del primer cromosoma con su aptitud relativa y se calcula la aptitud acumulada del siguiente cromosoma sumando su aptitud relativa más la acumulada del cromosoma anterior y así sucesivamente para cada cromosoma. Esta aptitud acumulada representa la proporción de cada cromosoma (Figura 3).

Para simular el giro del disco se genera un número aleatorio *x*, entre 0 y 1, y se elige el cromosoma cuya aptitud acumulada correspondiente contiene a *x*. Este proceso se repite el número

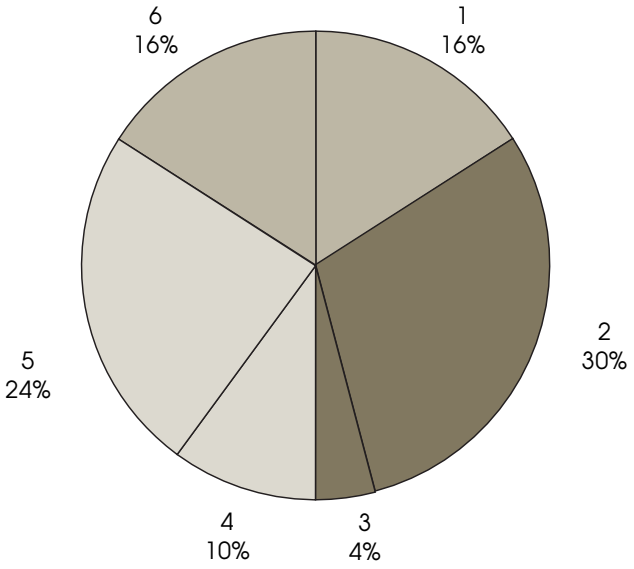

Figura 3. Proceso de selección.

de cromosomas existentes. Los cromosomas que ocupan una porción mayor del disco tienen más probabilidad de ser escogidos para la nueva población, sin embargo, no excluye la elección de cromosomas con peor aptitud. Según la figura 3 el cromosoma 2 tiene mayor probabilidad de sobrevivir y el 3 tiene mayor probabilidad de morir. Al final de este proceso el arreglo que representa la nueva población puede quedar con cromosomas repetidos. La aptitud relativa y la acumulada son campos de la estructura que se guardan en el arreglo (Figura 2).

Mutación estructural

Es necesario tener al menos un operador que pueda cambiar el tamaño de los cromosomas; si no fuera así, el número de nodos ocultos en cada cromosoma quedaría igual al número con el cual fue inicializado. Esto es importante de acuerdo a nuestra meta de encontrar simultáneamente la arquitectura y los pesos más adecuados de un PM para resolver un problema de clasificación. El PE utiliza dos operadores, Adicionar y Eliminar, definidos como sigue:

1. Adicionar, se encarga de aumentar un nodo oculto al cromosoma en cuestión inicializando sus pesos con 0. Se valida que al aumentar este nodo no sobrepase el número máximo de nodos permitidos por el usuario; si ya se tiene ese número máximo no se añade otro nodo. Este nodo se incorpora al final de la lista y se hace de acuerdo a un número generado de forma aleatoria: si este es menor a una probabilidad de adición definida por el usuario se adiciona el nodo, en otro caso no se hace ningún cambio al cromosoma.

2. Para equilibrar el efecto que pudiera tener la función Adiciona, y para reducir el número de nodos ocultos en un cromosoma, se define otro operador de nombre Eliminar. De forma similar se genera un número aleatorio y se compara con la probabilidad de eliminación definida por el usuario, si dicho número es menor se elimina el primer nodo de la lista del cromosoma en cuestión.

En los dos casos se reduce linealmente el parámetro de probabilidad de su valor inicial a un valor pequeño definido por el usuario, conforme evoluciona el algoritmo; con esto, al final de la corrida no es necesario modificar la arquitectura del PM.

Mutación paramétrica

Para modificar los pesos de un cromosoma, se aplica un operador a los pesos de un nodo oculto del cromosoma. Esto quiere decir que para cada nodo oculto de un cromosoma se genera un número aleatorio y si éste es menor a la probabilidad de mutación definida por el usuario, se modifican los pesos agrupados en el nodo; en caso contrario se conserva el mismo valor de los pesos. Si se decide cambiar el valor de un peso se le agrega el valor generado por una gaussiana, con media 0 y varianza definida por el usuario. La varianza está reducida linealmente conforme evoluciona el algoritmo; esto se hace para evitar perturbaciones "grandes" a los pesos lo cual no es benéfico al final de la corrida.

Guardar el mejor cromosoma

En la literatura se han reportado buenos resultados usando un operador llamado elitismo. El cometido de éste es asegurar que el mejor cromosoma encontrado sobreviva de una generación a otra. En el PE, si el cromosoma con mejor aptitud de la población actual supera al seleccionado en la población anterior, éste es reemplazado, conservando siempre al mejor cromosoma de una generación a otra. En otro caso reemplaza el peor cromosoma de la población actual con el mejor de la generación anterior.

Una aplicación

Para ilustrar el PE con un ejemplo concreto, tomamos un conjunto de datos, denominado Iris, bien conocido en el ámbito de la clasificación y usado por el estadístico Fisher (9). El conjunto de datos consta de 3 tipos de la planta iris, con 50 ejemplos cada una. Las clases son: Iris Setosa, Iris Versicolor e Iris Virginica. Para distinguir estas clases se han registrado, para cada ejemplo, los valores de cuatro variables correspondientes a: longitud de sépalo, ancho de sépalo, longitud de pétalo, y ancho de pétalo (todo en cm.).

Una de las razones por su interés es la ubicación física de los miembros de las clases en el espacio de cuatro dimensiones. Como el lector puede observar en la Figura 4, una de las clases (Setosa) se distingue fácilmente de las otras dos; de hecho se puede encontrar un hiperplano que realiza la separación en el sentido de que todos los miembros de ésta clase esten en un lado del hiperplano, mientras que el resto de los datos estén en el otro lado. Sin embargo, en el caso de los miembros de las otras dos clases ya no es tan sencillo separarlos.

Se aplicó el PE a las dos clases "difíciles" de Versicolor y Virginica con los siguientes parámetros (Cuadro 2):

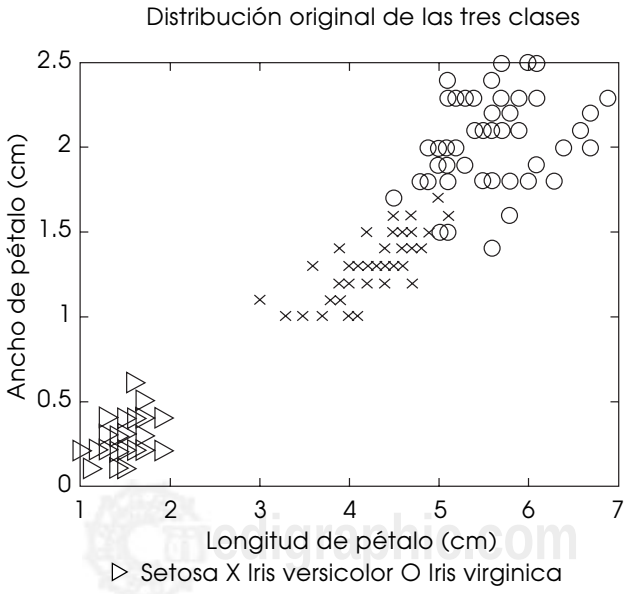

Figura 4. Proyección de los datos de Iris a dos dimensiones.

Cuadro 2. Parámetros del PE.

PARÁMETRO	VALOR
Probabilidad de eliminación de un nodo (Mutación estructural)	1
Probabilidad de adición de un nodo (Mutación estructural)	0.4
Número de nodos de entrada	4
Número de nodos de salida	1
Máximo tamaño de cromosomas	15
Tamaño de la población	20
Número de generaciones	300
Probabilidad de mutación de un nodo (Mutación paramétrica)	0.8

El usuario puede cambiar el valor de cada parámetro y experimentar con ellos.

Los resultados obtenidos se muestran en el cuadro 3:

Cuadro 3. Resultados.

Función de aptitud	Mal clasific.	Bien clasific.	Nodos ocultos
ECM	4	96	6
Por núm. de aciert.	2	98	6

El importante notar que el número de nodos ocultos que dio como resultado el PE en ambas funciones de aptitud fue de 6. Estos resultados concuerdan con el desempeño de PM entrenados de la forma usual.

CONCLUSIONES

En este artículo, se ha descrito un PE con el propósito de simultáneamente encontrar la arquitectura y los pesos de un PM en tareas de clasificación. Debido al problema, el PE utiliza cromosomas de longitud variable, que representan a los PM, y la idea de mutación estructural para variar el número de nodos ocultos en el PM. Vimos que en el ejemplo sencillo de Iris el PE funcionó bien. No obstante, hay un buen número de parámetros involucrados en el PE (Véase cuadro 2), y puede parecer más arte que ciencia cómo definirlos. En realidad, una parte importante es la experimentación "bien dirigida" y el de-

sarrollo de una intuición acerca del problema tratado. También surgen muchas preguntas acerca de los operadores usados, las funciones de evaluación definidas, la implementación adoptada, que seguramente el lector interesado le gustaría modificar. Esperemos que el lector ahora pueda implementar su propio algoritmo e iniciarse en el mundo de los PE.

BIBLIOGRAFÍA

1. Arturo Hernández Aguirre, «Introducción a las Redes Neuronales Artificiales», Soluciones Avanzadas, Año 7, No. 63, pp. 25-34, nov. De 1998.
2. José R. Hilera, Víctor J. Martínez. "Redes Neuronales Artificiales.
   Fundamentos, modelos y aplicaciones", Addison-Wesley Iberoamerica, 1987.
3. Mohamad H. Hassoun, "Fundamentals of Artificial Neural Networks", The MIT Press, 1995.
4. David J. Montana, "Neural Network Weight Selection Using Genetic Algorithms", capitulo 5 de Intelligent Hybrid Systems, Goonatilake and Khebbal (Eds.), John Wiley & Sons, 1995.
5. Frédéric Gruau, "Genetic Programming of Neural Networks: Theory and Practice" capitulo 13 de Intelligent Hybrid Systems, Goonatilake and Khebbal (Eds.), John Wiley & Sons, 1995.
6. Peter J. Angeline, Gregory M. Saunders y Jordan B. Pollack, "An Evolutionary Algorithm that Constructs Recurrent Neural Networks", IEEE Transactions on Neural Networks, 5 (1), pp. 54-65, 1994.
7. Xin Yao, "Evolving Artificial Neural Networks", Proc. of the IEEE, 87(9):1423-1447, September 1999.
8. David B. Fogel, "Evolutionary Computation. Toward a New Philosophy of Machine Intelligence", The Institute of Electrical and Electronic Engineers, New York, 1995.
9. R.A. Fisher,"The use of multiple measurements in taxonomic problems," Annual Eugenics, 7, Part II, pp. 179-188, 1936.

(Apoyo de Conacyt No. 400200-5-31929A)