# Ambiente software de entrenamiento de redes neuronales con ajuste evolutivo de la topología y las funciones de activación*

Edgar A. Méndez Ortiz
*edgar9000@gmail.com*

Juan Sebastián Mariño
*jhusema@gmail.com*

Henry Argüello Fuentes
*henarfu@uis.edu.co*

Fecha de recepción: 07-11-2008 Fecha de selección: 09-02-2009 Fecha de aceptación: 06-02-2009

**ABSTRACT**

This research examines two problems in the optimization in the neural networks used for most real applications: first, architectural design that involves determining the number of layers and neurons by layer, and second, the activation functions that will be should use in each of these layers. For it is developed a software tool based on genetic algorithms to find these parameters of a neural network. The developed tool allows the user to choose the algorithm used for training and also apply techniques to achieve better generalization such as the early stopping, the repetition of training and adjusting the training data to the activation functions used. Finally, the developed tool is tested into a specialized group of users who use the tool to find an optimal neural network architecture to solve a problem of identity verification through the facial image using artificial neural networks.

**KEYWORDS**

Neural networks, genetic algorithms, evolutionary computation, optimization.

* GIIB: Grupo de Investigación en Ingeniería Biomédica. Universidad Industrial de Santander

**RESUMEN**

Este trabajo de investigación estudia dos problemas en la optimización en las redes neuronales utilizadas para la mayoría de aplicaciones reales: primero, el diseño de la arquitectura que involucra determinar el número de capas y neuronas por capa, y segundo, las funciones de activación que se deben usar en cada una de estas capas. Para ello se desarrolla una herramienta software basada en algoritmos genéticos que encuentra estos parámetros de las redes neuronales. La herramienta desarrollada le permite al usuario elegir el algoritmo de entrenamiento usado; además se aplican técnicas para lograr una mejor generalización como son la detención temprana, la repetición del entrenamiento y el ajuste de los datos de entrenamiento a las funciones de activación usadas. Por último, la herramienta desarrollada es probada en un grupo de usuarios especializados que utilizan la herramienta para encontrar una arquitectura de red neuronal óptima para resolver un problema de verificación de identidad a través de la imagen facial mediante redes neuronales artificiales.

**PALABRAS CLAVE**

Redes neuronales, algoritmos genéticos, computación evolutiva, optimización.

**Clasificación Colciencias:** Tipo 1

## 1. INTRODUCCIÓN

Las redes neuronales artificiales, como parte de la teoría de la inteligencia artificial, han mostrado su utilidad en diferentes áreas de la ciencia, donde se aplican a problemas de clasificación y predicción de patrones.[12] Aunque esta técnica ha sido utilizada frecuentemente por muchos investigadores, no se tiene aún una teoría sólida que permita identificar la estructura de una red neuronal de forma adecuada; por lo tanto la selección de esta estructura se busca por medio de prueba y error, o en el mejor de los casos la aplicación de algunas reglas heurísticas para obtener una determinada configuración. En este último caso juega un papel importante la experiencia adquirida por el experto.[11]

Por tanto resulta conveniente desarrollar una herramienta software que encuentre una configuración de red neuronal para un problema específico, de forma tal que sirva como herramienta de decisión para las personas que utilizan estos sistemas en sus investigaciones. Sin embargo, se encuentra el problema que existen infinitas estructuras, por lo cual se hace necesario utilizar alguna técnica, diferente a la fuerza bruta, para hallar de forma rápida esas soluciones.

Se han desarrollado herramientas software[1,2,3,4,5] que utilizan los algoritmos genéticos como método para la búsqueda de las arquitecturas óptimas de la red neuronal artificial para determinados problemas. En estas aplicaciones se obtuvieron buenos resultados; sin embargo, éstas tienen limitaciones ya que sólo se utiliza la variación de un parámetro de la red neuronal.[6]

Con el fin de mostrar una alternativa a estas investigaciones, en el presente artículo se indican las pruebas y conclusiones que se obtuvieron con el software desarrollado, el cual utiliza combinadamente dos de las técnicas para la obtención de configuraciones por métodos evolutivos; la evolución simultánea de la arquitectura y las funciones de activación. Además, se compara el desempeño al utilizar el software desarrollado para obtener topologías de redes neuronales tipo perceptrón multicapa contra las técnicas clásicas para obtener una configuración de red neuronal.

En la primera parte se realiza una descripción del Algoritmo Genético utilizado para hallar las configuraciones de redes neuronales, así como de la herramienta software desarrollada para ser utilizada en encontrar la organización óptima de una red neuronal para una aplicación dada. Seguidamente, se hace una descripción del método de *eigenfaces* utilizado dentro de un sistema de verificación de identidad usando redes neuronales. Luego, la herramienta software es puesta a prueba en un curso en inteligencia artificial de pregrado de una universidad, para que mediante esta herramienta encuentren la configuración óptima de una red neuronal utilizada en la aplicación de verificación de identidad. Por último se muestran los resultados numéricos de las pruebas y se realizan comparaciones entre tres tipos de usuarios.

## 2. PLANTEAMIENTO DEL PROBLEMA

El diseño de la estructura de redes neuronales es un proceso tedioso, que requiere por parte del usuario

un grado de experiencia de diseño/ entrenamiento, junto con un proceso de prueba y error dependiente de la complejidad del problema. Este proceso no se limita a determinar el número de capas y neuronas ocultas que debe tener la red, sino que se involucran muchos más parámetros que se pueden modificar en aras de encontrar una red que produzca una mejor generalización.

La optimización de la estructura de una red neuronal puede ser clasificada de acuerdo con la meta a alcanzar. Algunos esquemas han propuesto la optimización de los pesos sinápticos, otros afirman que lo más importante es la arquitectura y otros acercamientos han incluido funciones de activación y reglas de aprendizaje. Sin embargo, el área más interesante para nuevas investigaciones radica en la combinación de estos enfoques de optimización.[6]

En este proyecto se utilizaron combinadamente dos de las técnicas anteriormente mencionadas: la evolución simultánea de la arquitectura y las funciones de activación.

El sistema propuesto consiste en un algoritmo genético que busca obtener el número de capas, número de neuronas por capa y funciones de activación de las capas de una determinada red neuronal tipo perceptrón multicapa. De esta forma la búsqueda de la mejor estructura $R$ de una red neuronal se puede ver como la maximización de la función $f$ dada por la ecuación 1.

$$R = f(M_x, N_x, G_x, T_x, ES) \qquad (1)$$

Donde $M_x$ representa el número de capas de la red, $N_x$ es el número de neuronas por cada capa, $G_x$ es la función de activación de cada una de las capas, $T_x$ es el algoritmo de entrenamiento y $ES$ son los ejemplos con que se cuenta para entrenar la red neuronal. Además, cada una de estas variables está limitada a un conjunto finito de valores para limitar el espacio de búsqueda, de esta forma $M_x \in \{1, m\}$ donde $m$ es el máximo número de capas; $N_x \in (0, n)$ donde $n$ es el máximo número de neuronas por capa. Los ejemplos de entrenamiento $ES$ permanecen constantes y no serán variados para optimizar la función $f$ dada por (1).

La selección de funciones de activación se realiza de acuerdo con el problema a resolver y a criterio del investigador, en ocasiones por ensayo y error. En la literatura no existe un criterio estándar para la selección de estas funciones de activación en las redes neuronales.[3]

En el sistema se tienen como base cuatro funciones de activación que han sido ampliamente estudiadas.[7,8] Estas funciones serán establecidas por el algoritmo genético buscando los mejores resultados para cada problema.[4] De esta forma:

$G_x \in (\log sig,\ radbas,\ purelin,\ \tan sig)$, donde cada una de estas funciones se muestran en la Figura 1.

Por último, los algoritmos de entrenamiento que se utilizarán para realizar la optimización de la estructura son:

- Traingda: Gradiente descendente con tasa de aprendizaje adaptativa con propagación hacia atrás.
- Traingdx: Gradiente descendente con momento y tasa de aprendizaje adaptativa con propagación hacia atrás.

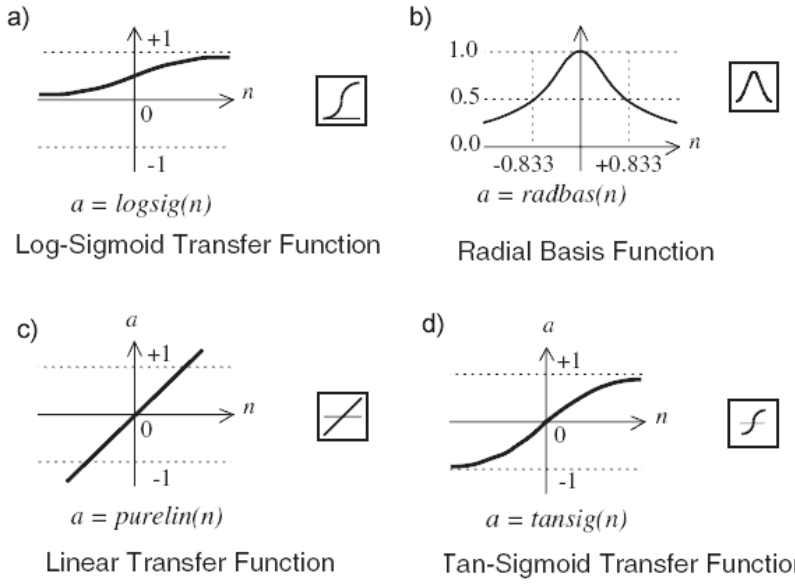


**Figura 1.** Funciones de activación a) Función logística o LogSig. b) Función Gaussiana o RadBas. c) Función lineal o Purelin. d) Función Tangente hiperbólica o Tansig.

- Trainrp: Propagación hacia atrás *Resilient* .
- Trainlm: Propagación hacia atrás Levenberg-Marquardt.
- Trainbfg: Propagación hacia atrás BFGS quasi-Newton.

De esta forma:

$$T_x \in (Traingda, Traingdx, Trainrp, Trainlm, Trainbfg)$$

Definidos los parámetros dados por (1), se procede a resolver el problema usando algoritmos genéticos (AG). Un individuo del AG está formado por cada uno de los parámetros de la ecuación 1.

$$I_x = [M_x\ N_x\ G_x\ T_x] \qquad (2)$$

El AG está sujeto a la función de optimización dado por (3).

$$f_x = \frac{1}{r}\sum_{i=1}^{r} f(I_x) \qquad (3)$$

Donde $f(I_x)$ es una función que toma el individuo $I_x$ que representa una estructura de red neuronal $R$ dada por (1) y realiza el entrenamiento de la misma y devuelve el error cuadrático medio alcanzado por la red neuronal. Debido a que los pesos iniciales tienen una incidencia grande en los resultados que se obtienen, este entrenamiento se repite $r$ veces y se promedian los resultados. Es importante resaltar que si se aplica la función $f$ dos veces sobre la misma estructura $I_x$ , en cada una de las ocasiones se obtendrán resultados diferentes ya que los pesos iniciales serán distintos cada vez.

De esta manera el AG que se diseñó en este proyecto, busca obtener el

mejor individuo $I_M$ sujeto a la función de optimización dada por (3). Al finalizar el proceso de evolución del AG se obtiene la configuración de red que maximiza el desempeño de la red para los datos de entrenamiento *ES* . Es decir se obtiene el número de capas, el número de neuronas por capa y las funciones de activación de cada capa de una red neuronal tipo perceptrón multicapa utilizada en una aplicación específica.

Para verificar el funcionamiento y eficiencia del sistema desarrollado se utilizó un sistema de localización de rostros basado en redes neuronales que es explicado en detalle en la sección 2.1. Por otro lado se colocó a prueba la herramienta en un curso de pregrado en la asignatura Inteligencia Artificial.

Por este motivo y con el objetivo de que un usuario general pueda utilizar fácilmente la técnica basada en AG para encontrar la arquitectura de una red neuronal, usando algoritmos genéticos, se diseñó e implementó una herramienta software. En la siguiente sección se muestran los detalles de esta herramienta.

### 2.1 Herramienta software desarrollada, Wötan Genetics

El sistema automático de entrenamiento desarrollado (Wötan Genetics), es una herramienta computacional para la configuración de redes neuronales artificiales, por medio de algoritmos genéticos, para problemas específicos de clasificación y aproximación. Debido a la facilidad con el manejo de diferentes arquitecturas de redes neuronales artificiales, se utilizó para su desarrollo Matlab® en su versión 2007a.

La herramienta posee una sencilla e intuitiva interfaz gráfica, donde se presentan diferentes ventanas en las cuales se establecen los diversos parámetros necesarios para acotar el espacio de búsqueda del algoritmo genético, cargar los datos a ser utilizados para el entrenamiento y las pruebas de las diferentes configuraciones; finalmente incluye la visualización de las soluciones y el análisis de las mismas.

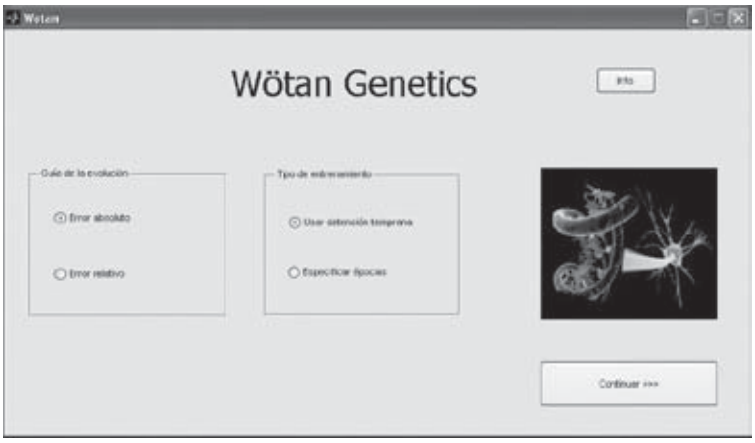


**Figura 2.** Ventana principal del sistema.

Dentro del uso del sistema se tiene la posibilidad de elegir entre realizar el entrenamiento basado en detención temprana, o especificar el número de épocas durante el cual se va a realizar el entrenamiento de cada configuración probable. De acuerdo con la opción seleccionada se mostrará una ventana para la carga de datos correspondiente; en la Figura 3 se observa el menú para carga de datos, realizando un entrenamiento con detención temprana.


**Figura 3.** Ventana para carga de datos de entrenamiento y pruebas.

Después del proceso de búsqueda se muestra un informe detallado de las mejores configuraciones obtenidas para resolver el problema en particular, y un análisis general del problema donde se pueden observar diferentes gráficas que indican la variación del problema al cambiar el número de neuronas y capas ocultas.

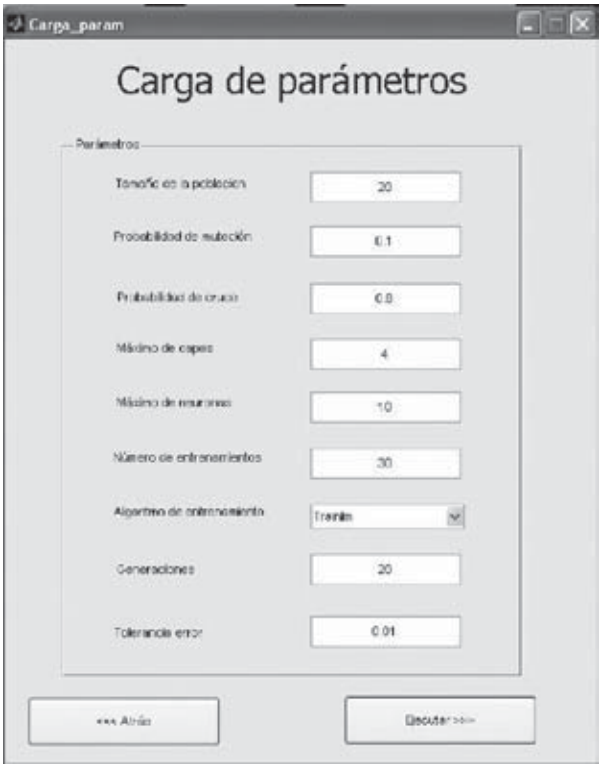


**Figura 4.** Ventana para carga de parámetros.

De esta manera la herramienta implementada permite que un usuario general pueda aplicar el método de AG en la búsqueda de una arquitectura de una red neuronal para una aplicación determinada. Para probar la funcionalidad y utilidad de esta herramienta, se utilizó una aplicación de identificación de rostros para que diferentes usuarios encontraran una configuración de red neuronal óptima para esta


**Figura 5.** Ventana de configuraciones obtenidas.

aplicación. En la siguiente sección se hace una descripción del sistema de verificación de rostros mediante redes neuronales.

### 2.2 Sistema de verificación de rostros

En la aplicación que se analiza, a una red neuronal se le muestra una cantidad de imágenes de las cuales se tiene conocimiento sobre la localización de un rostro (Figura 6.a) y se le muestran imágenes donde no hay un rostro (Figura 6.b). A partir de estos datos se pretende encontrar una estructura de red neuronal que pueda determinar de forma general la presencia de un rostro dentro de una escena.

Para reducir la dimensionalidad de las imágenes que son utilizadas como ejemplos de entrenamiento para la red neuronal se usa la técnica de *eigenfaces*.[9] Para implementar esta técnica se empleó una base de datos de 49 imágenes de rostros, como la que se muestra en la Figura 7.

En cada una de las imágenes de la base de datos se recorta la información de interés de forma manual como se observa en la Figura 7. Todas las imágenes recortadas manualmente están normalizadas en tamaño, posición y orientación.


**Figura 7.** Preprocesado de la base de datos

Cada imagen recortada manualmente de la base de datos es mejorada en cuanto a intensidad y contraste

**Figura 6a.** Subimágenes con rostro

**Figura 6b.** Subimágenes que no son rostros

utilizando técnicas de tratamiento de imágenes como ecualización de histograma y corrección de iluminación. En la Figura 8 se presenta el procedimiento de corrección de gamma y ecualización de histograma en una imagen.

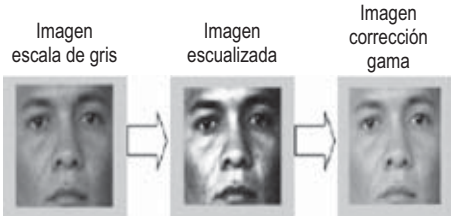


**Figura 8.** Preprocesado de la base de datos

Teniendo las imágenes mejoradas en iluminación y contraste se implementa un filtro sobel para realzar las características extraídas manualmente de cada imagen, esto con el fin de obtener una mejor representación de los eigenvectores de las imágenes. El método de eigenrostros o eigenvectores es una técnica utilizada para reducir el tamaño de los datos en análisis tratando de conservar la mayor cantidad de información posible.

La metodología de eigenrostros se basa en el análisis de componentes principales, una técnica estadística que busca representar un grupo de imágenes faciales como una combinación lineal de los mejores eigenvectores de una matriz de covarianza. Esta matriz está conformada por cada una de las imágenes de la base de datos restándoles el promedio de las mismas.[9] Calculando los mejores eigenvectores del conjunto de imágenes faciales, cualquier rostro puede ser representado como una combinación lineal de estos eigenvectores de la siguiente manera:

$$Y_i = \sum_{k=1}^{R} E_k U_k \tag{4}$$

Donde $Y_i$ es la representación de cada rostro mediante eigenvectores, $U_k$ son los eigenvectores, $E_k$ son los coeficientes de proyección o eigenvalores, los cuales son calculados de la siguiente manera:

$$E_k = U' Q_i \tag{5}$$

Donde $Q_i$ es la diferencia de cada rostro con el promedio de la base de datos. En la Figura 9 se presenta un ejemplo de la representación de una imagen como la combinación lineal de 4 eigenrostros, en este ejemplo se puede observar que la imagen se almacenaría con 4 valores numéricos correspondiente a los eigenvalores, con lo cual se consigue la enorme reducción en la dimensionalidad del problema. Como en esta aplicación se utilizó una base de datos de 49 imágenes con rostros, entonces una imagen en general se puede representar como una combinación lineal de 49 eigenvalores calculados mediante (5).

Se puede encontrar una descripción detallada de cómo calcular los eigenvectores y eigenvalores en los artículos de donde fueron tomadas las pruebas.[9,10]


**Figura 9.** Representación mediante eigenvectores.

Una vez completados estos procedimientos, se seleccionaron 1.000 imágenes con rostros y otras que no tienen rostros como los de la Figura 6.a y 6.b. Mediante la técnica de eigenfaces cada una de estas 1.000 imágenes se puede representar como un vector columna de 49 valores que corresponden a los 49 eigenvalores.

De esta forma se obtiene una matriz de dimensiones 49x1000 que constituye la matriz de entrenamiento para la red neuronal. En la Figura 10 se muestra un ejemplo de la metodología usada para entrenar la red neuronal.

De los 1.000 ejemplos seleccionados, 800 son usados para entrenamiento y 200 para probar la eficiencia de las redes neuronales entrenadas. De los 800 ejemplos de entrenamiento 500 son usados para el aprendizaje de las redes neuronales y los 300 restantes para implementar la técnica de parada anticipada.[13]

Una vez entrenada la red neuronal, esta debe estar en capacidad de, dada una imagen, determinar si contiene o no un rostro.

En las secciones anteriores se describió la técnica de AG para encontrar la arquitectura de una red neuronal para una aplicación determinada, luego se describió una herramienta software para que un usuario en general pueda utilizar fácilmente esta técnica. Además, se indicaron los detalles de un sistema de verificación mediante la imagen facial usando redes neuronales. Por último, en la siguiente sección se colocará a prueba la técnica implementada suministrándoles a estudiantes universitarios que cursan la asignatura inteligencia artificial la herramienta Wötan Genetics para que hallen una configuración óptima de una red neuronal para ser utilizada en un sistema de verificación mediante rostros que previamente fueron mostrados.

### 2.3 Metodología de las pruebas

Una vez establecidos los conceptos teóricos de la aplicación de verificación de rostros, se procede a encontrar una arquitectura de red neuronal que solucione este problema. El objetivo de este proyecto es encontrar una arquitectura de red mejor que la utilizada en el trabajo original que usa esta técnica[11,2].

La herramienta desarrollada se probó mediante un curso de treinta

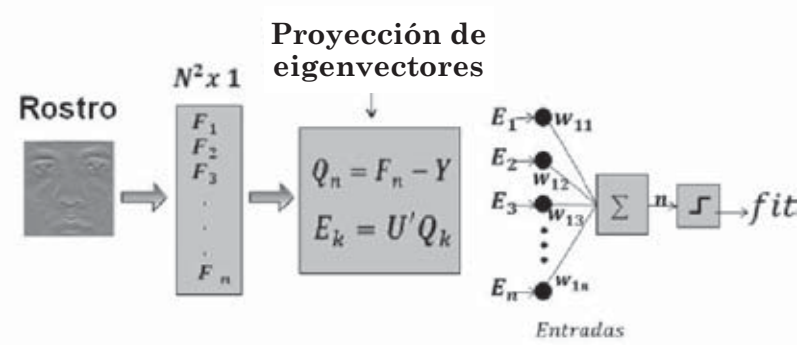


**Figura 10.** Entrenamiento de la red neuronal mediante eigenvalores.

alumnos de últimos semestres de ingeniería de sistemas que estaban cursando la materia de inteligencia artificial, y que por tanto tuvieran conocimientos sobre el tema que se estaba tratando.

Para estas pruebas, a los voluntarios se les entregó la herramienta de configuración Wötan Genetics y la matriz de entrenamiento de la red neuronal para verificación mediante la imagen facial mencionada en la sección anterior. Los parámetros de la búsqueda que necesita la herramienta software debían ser escogidos por los estudiantes para buscar diferentes comportamientos en la solución del problema.

Los parámetros establecidos en cada una de las búsquedas realizadas por los estudiantes son las mostradas en la Tabla 2.

**Tabla 1.** Variables de entrada usadas por el algoritmo genético.

| Número | Notación | Descripción |
|---|---|---|
| 1 | P1 | Número máximo de capas |
| 2 | P2 | Número máximo de neuronas |
| 3 | P3 | Tolerancia al error |
| 4 | P4 | Número de generaciones |
| 5 | P5 | Algoritmo de entrenamiento |
| 6 | P6 | Número de entrenamientos |
| 7 | P7 | Tamaño de la población |
| 8 | P8 | Probabilidad de cruce en porcentaje |
| 9 | P9 | Probabilidad de mutación en porcentaje |

Para evaluar el desempeño de las redes neuronales obtenidas se utilizaron el error promedio absoluto dado por (7) y la Tasa de ajuste dada por (8).

**Tabla 2.** Parámetros utilizados por los estudiantes en las pruebas de inteligencia artificial.

| | Prueba 1 | Prueba 2 | Prueba 3 | Prueba 4 | Prueba 5 |
|---|---|---|---|---|---|
| P1 | 3 | 1 | 2 | 2 | 2 |
| P2 | 20 | 30 | 30 | 30 | 30 |
| P3 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| P4 | 20 | 20 | 20 | 20 | 25 |
| P5 | Traingda | Traingdx | Trainrp | Trainlm | Trainbfg |
| P6 | 30 | 35 | 60 | 50 | 40 |
| P7 | 20 | 20 | 20 | 20 | 20 |

$$E_a = \frac{\sum_{i=1}^{n} |S_i - S_{si}|}{n} \tag{7}$$

$$Ta = 100 - 100 \frac{\sum_{i=1}^{n} \frac{|S_i - S_{si}|}{S_i}}{n} \tag{8}$$

$S_i$ = Salida deseada.

$S_{si}$ = Salida obtenida por la red neuronal.

$n$ = Número de datos usados para la prueba.

Estos valores suministran una idea de la calidad de los resultados obtenidos por cada arquitectura de red neuronal encontrada. Los valores de las ecuaciones (7) y (8) son calculados con los 200 ejemplos de entrenamiento guardados para las pruebas tal y como se explicó en la sección 2.2.

## 3. ANÁLISIS DE RESULTADOS

Las diferentes pruebas realizadas utilizando la herramienta software basada en AG, arrojaron como resultado configuración de redes neuronales artificiales que se adaptan en mayor o menor medida a los datos de entrenamiento seleccionados.

En cada una de las pruebas de simulación la herramienta software suministra la configuración de red neuronal que haya producido los mejores resultados en cuanto a su error cuadrático medio en el proceso de entrenamiento.

Esta red es evaluada luego con los 200 datos de prueba que fueron guardados para evaluar la eficiencia de las redes neuronales encontradas y mencionados en la sección 2.2. Esta eficiencia se calcula mediante la tasa de ajuste dada por (8) y el error promedio dado por (7).

### 3.1 Pruebas de inteligencia artificial realizadas por los estudiantes

De los parámetros de búsqueda mostrados en la Tabla 2, se obtuvieron diversas topologías para dar solución al problema de verificación mediante la imagen facial, los mejores resultados son los mostrados en la Tabla 3.

**Tabla 3.** Mejores configuraciones de inteligencia artificial halladas por los estudiantes.

| Prueba | Capas | Número de neuronas | Función de activación | Tasa de ajuste (%) |
|---|---|---|---|---|
| 1 | Oculta 1 | 9 | Purelin | 57.5 |
| | Oculta 2 | 10 | Purelin | |
| | Salida | 1 | Radbas | |
| 2 | Oculta 1 | 29 | Tansig | 91.4 |
| | Salida | 1 | Radbas | |
| 3 | Oculta 1 | 30 | Tansig | 95.6 |
| | Oculta 2 | 21 | Radbas | |
| | Salida | 1 | Radbas | |
| 4 | Oculta 1 | 2 | Logsig | 99.9 |
| | Salida | 1 | Tansig | |
| 5 | Oculta 1 | 5 | Purelin | 97.4 |
| | Oculta 2 | 1 | Logsig | |
| | Salida | 1 | Tansig | |

Aplicando el método de validación cruzada,[9] cada una de estas topologías fue evaluada con los datos de prueba, un conjunto de datos que nunca formó parte del conjunto de entrenamiento, pero que presenta el mismo comportamiento de los datos, esto es de gran utilidad para determinar el nivel de generalización de cada configuración de acuerdo con los errores (calculados con la ecuación 7) producidos en la simulación.

A partir de los errores promedio de las simulaciones, calculados con la ecuación 7, se puede realizar una comparación del desempeño de cada una de las mejores configuraciones halladas en las pruebas anteriores; en la Figura 11 se muestra esta comparación.

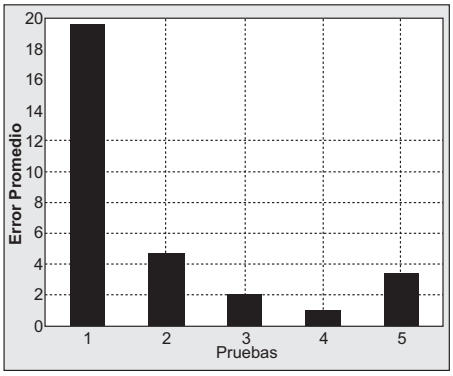


**Figura 11.** Error promedio de las mejores configuraciones obtenidas en cada simulación.

La configuración que mostró mejores resultados (prueba 4) fue la entrenada por medio del algoritmo Levenberg-Maquard (trainlm) y es la escogida para ser comparada con las configuraciones halladas en las siguientes etapas de la prueba.

Es importante resaltar que el tiempo promedio para que un usuario obtenga los resultados de las arquitecturas

usando la herramienta Wötan Genetics está entre cuatro y doce horas. Los usuarios utilizaban equipos de cómputo de propósito general con procesador Pentium IV con dos núcleos de 1.8GHz y 2GB de memoria RAM. Sin embargo, puede mencionarse que si un usuario intenta a prueba y error de simular el mismo número de arquitecturas que esta herramienta software podría demorar semanas o meses en esta tarea.

### 3.2 Configuración hallada por los desarrolladores de la herramienta software

Después de tener el suficiente conocimiento sobre el comportamiento del problema de encontrar una arquitectura de red neuronal para la aplicación de verificación de identidad, se decidió ajustar un espacio de búsqueda basado en los resultados de inteligencia artificial obtenidos por los estudiantes. En este punto, el número de neuronas y capas aproximado para solucionar el problema eran conocidas, y se decidió utilizar un algoritmo que explotara el espacio de búsqueda de la mejor manera. Los parámetros utilizados son los mostrados en la Tabla 4:

**Tabla 4.** Parámetros utilizados en las pruebas por los autores del proyecto.

| Parámetro | Valor |
|---|---|
| Número máximo de capas | 2 |
| Número máximo de neuronas | 30 |
| Tolerancia al error | 0.1 |
| Algoritmo de entrenamiento | Trainlm |
| Número de entrenamientos | 60 |
| Número de generaciones | 20 |
| Tamaño de la población | 20 |
| Probabilidad de mutación | 0.1 |
| Probabilidad de cruce | 0.8 |

En esta simulación, el mejor resultado fue obtenido al usar una sola capa oculta con seis neuronas, con funciones de activación Radbas y Tansig en la capa oculta y de salida, respectivamente. Al evaluar con los datos de prueba se comprobó una tasa de ajuste del 100%, de acuerdo con la tolerancia establecida.

### 3.3 Comparación de resultados

De los resultados anteriores se tiene conocimiento de las mejores configuraciones halladas por diferentes usuarios y con diferentes niveles de conocimiento del problema de verificación de rostros mediante redes neuronales. Estos usuarios son: (a) los resultados encontrados por los autores de la referencia bibliográfica,[11] donde originalmente se abordó el problema de verificación de rostros mediante redes neuronales artificiales, (b) los resultados encontrados por los estudiantes de inteligencia artificial utilizando la herramienta software Wötan Genetics y por último (c) los resultados obtenidos por los autores de esta investigación con la observación que ya se conocía, una solución aproximada del problema dada por (a) y (b).

Con estos tres tipos de usuarios se procede a realizar una comparación de sus resultados, para esto se evaluarán las configuraciones obtenidas con la totalidad de los datos del problema para calcular el error cometido. Además, se calcularán los tiempos de simulación para determinar la configuración que produzca resultados más rápidamente.

En este caso, los resultados obtenidos por los estudiantes de inteligencia artificial o (b) se denominarán red 1,

la hallada por (c) se llamará red 2, y la encontrada por (a) red 3.

Al simular las tres redes obtenidas en cada fase de prueba y evaluar su error promedio por medio de la ecuación 7, se puede observar el desempeño de cada una de estas, como es mostrado en la Figura 12.

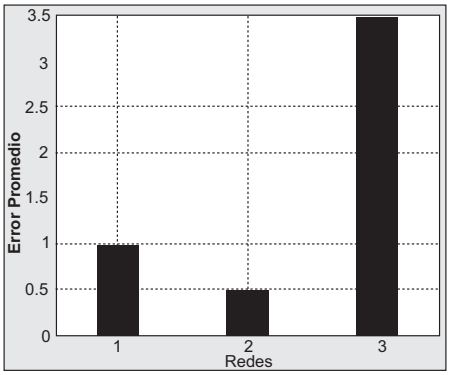


**Figura 12.** Error promedio de las mejores configuraciones obtenidas en cada fase de prueba.

Los tiempos de simulación de cada una de las configuraciones de red, en segundos, fueron:

Red 1: 0.008305

Red 2: 0.015592

Red 3: 0.010988

Las características de cada una de las topologías son las mostradas en la Tabla 5.

Por medio de esta comparación se puede observar cómo la configuración encontrada por los autores de esta investigación logra el mejor nivel de acierto, al ajustarse totalmente a los datos usados para las pruebas de cada configuración.

De igual manera, la mejor configuración obtenida por los estudiantes de inteligencia artificial también logra un excelente nivel de ajuste y el menor tiempo de simulación; esto dado que solo usa una capa oculta para resolver el problema.

**Tabla 5.** Configuraciones halladas en cada etapa de prueba.

| Nombre | Capa | Número de neuronas | Función de activación | Tasa de ajuste (%) |
|---|---|---|---|---|
| Red 1 | Oculta 1 | 2 | Logsig | 99.6 |
| | Salida | 1 | Tansig | |
| Red 2 | Oculta 1 | 6 | Radbas | 100 |
| | Salida | 1 | Tansig | |
| Red 3 | Oculta 1 | 20 | Tansig | 90.40 |
| | Salida | 1 | Tansig | |

En general, el desempeño de las configuraciones halladas por medio de la herramienta software basada en AG para la selección de la arquitectura de una red neuronal en la aplicación de verificación de rostros logró superar el nivel de acierto y el tiempo de simulación que los obtenidos por los autores en el artículo original.[11] Aunque las pruebas se hicieron con una aplicación específica, la herramienta Wötan Genetics sirve para cualquier tipo de aplicación basada en redes neuronales ya que lo que cambia es la matriz de entrenamiento usada.

## 4. CONCLUSIONES

Con la herramienta desarrollada se logra mejorar algunos de los problemas que existen en la búsqueda tradicional de la arquitectura de redes neuronales tipo perceptrón multicapa. Por una parte, las configuraciones probadas se generan de manera automática y son guiadas por el comportamiento del algoritmo genético; y por otro lado, la herra-

mienta permite hacer una búsqueda de la arquitectura de forma más exhaustiva al permitir observar el efecto de cambiar algunos de los parámetros que intervienen en la selección de esta arquitectura. De esta forma se obtienen resultados que tengan en cuenta una gran cantidad de posibilidades, y una base de conocimiento para decidir qué arquitectura usar.

Los resultados obtenidos mediante la aplicación del algoritmo han demostrado ser muy satisfactorios, superando los alcanzados al encontrar una arquitectura de red neuronal a prueba y error, donde el éxito está determinado en gran parte por el azar. Mediante este trabajo de investigación se encontró que una herramienta software basada en AG puede efectivamente facilitar el proceso de búsqueda de una arquitectura de red neuronal para una aplicación determinada.

La herramienta puede ser utilizada en cualquier otra aplicación de redes neuronales y, como se indicó en este trabajo, puede ser fácilmente empleada por usuarios que tengan conocimientos básicos de redes neuronales. De esta forma los usuarios disminuyen el trabajo de buscar una arquitectura de red neuronal y les permite invertir más tiempo en el análisis de sus investigaciones.

Aunque el tiempo medio para encontrar una arquitectura mediante la herramienta Wötan Genetics está entre cuatro y doce horas, en la actualidad se está desarrollando el mismo sistema pero que opera mediante un clúster de computadores, de forma tal que estos tiempos disminuyan considerablemente.

## 5. BIBLIOGRAFÍA


1. T.Sasaki and M Tokoro. *Evolving learnable neural networks under changing environment with various rates of inheritance of acquired characters: Comparison between darwinian and lamarckian evolution.* Artifical Life, pags 203-223, 1999
2. Richard K. Belew, Jhon McInerney, and Nicol N. Schraudolph. *Evolving networks: Using the genetic algoprithm with connectionist learning.* In Christopher G. Langton, Charles Taylor, J Doyne Farmer, and Steen Rasmussen, editors, Artificial Life II, pages 511-547, Addison-Wesley, Redwood City, CA, 1992.
3. J. Branke. *Evolutionary algorithms for neural network design and training.* Technical report no. 322, University of Karlsruhe, Institute AIFB, 1995
4. K. Stanley and R. Miikkulainen. *Evolving neural networks through augmenting topologies, technical report ai01-290.* Technical report, department of computer science, university of Texas as Austin 2001
5. H. Kitano. *Designing neural networks using genetic algorithm with graph generation system.* Complex Systems, pages 461-476, 1990.
6. D. Curran, C. O'Riordan**.** *Applying Evolutionary Computation to Designing Neural Networks: A Study of the State of the Art, technical report* NUIG-IT-111002, National University of Ireland, Galway, 2002

7. L. Llano, A. Hoyos, F. Arias, J. Velásquez. *Comparación del desempeño de funciones de activación en redes Feedforward para aproximar funciones de datos con y sin ruido,* 2007.
8. Hara, K.; Nakayamma, K. *Comparison of activation functions in multilayer neural network forpattern classification.* IEEE World Congress on Computational Intelligence, 1994. IEEE International Conference on Volume 5, Issue , 27 Jun-2 Jul 1994 Page(s):2997 – 3002 vol.5.
9. M. Turk and A. Pentland. "Eigenfaces For Recognition", Journal of cognitive Neuroscience, Vol. 3, pp. 71-86.1991.
10. George Bebis, Satishkumar Uthiram and Michel Georgiopoulos. "Genetic Search for Face Detection and verification", department of Computer Science, University of Nevada, Reno. 1999.
11. Fuentes, Henry Argüello, Ángel Rodríguez José, Caicedo Bravo Eduardo, Fabián Jaimes, Jairo. *Faces Identification using a hybrid model based on genetic algorithm and neural network*. In: 4th Conference Iberoamerican on System, Cybernetic and Informatics, 2005, v.1. p.101 – 106. Orlando, Florida, EE.UU.
12. Argüello, Fuentes Henry. *Applications of Genetic Algorithms in imaging processing*. I International seminary in electronic engineering, Universidad Pontificia Bolivariana, Bucaramanga, Faculty of electronic engineering. August 2006. Bucaramanga, Colombia.
13. Stone M**.** *Cross-validatory choice and assessment of statistical predictions.* Journal of the Royal Statistical Society, vol. B36, pp. 111-133, 1974.

## CURRÍCULOS

**Edgar Alberto Méndez Ortiz**: Estudiante de último semestre de ingeniería de sistemas Universidad Industrial de Santander. Miembro del grupo de investigación en ingeniería biomédica.

**Juan Sebastián Mariño**: Estudiante de último semestre de Ingeniería de Sistemas Universidad Industrial de Santander. Miembro del grupo de investigación en ingeniería biomédica.

**Henry Argüello Fuentes**. Nacido en julio de 1976 en Simacota, Santander, Colombia. Graduado como ingeniero electricista en el 2000 en la Universidad Industrial de Santander (UIS). Graduado como magíster en potencia eléctrica en el 2003 en la UIS. Actualmente se desempeña como profesor asistente con dedicación de tiempo completo en la escuela de Ingeniería de Sistemas e Informática de la UIS. Sus áreas de interés son el procesamiento de señales digitales, la inteligencia artificial y las telecomunicaciones.