Ingeniería Energética Vol. XXXIV No. 2/2013 p. 137-150, Mayo/ Agosto ISSN 1815 - 5901

APLICACIONES INDUSTRIALES

Clasificación de fallas con redes neuronales para grupos electrógenos

Faults classification with neuronal networks for electrogen groups

Luis Benigno – Corrales Barrios
Alexei - Ramírez Vázquez

Recibido: Julio del 2012
Aprobado: Diciembre del 2012

Resumen/ Abstract
Con el incremento del grado de dependencia de la sociedad moderna de los sistemas y procesos tecnológicos complejos, su disponibilidad y correcto funcionamiento se han convertido en una cuestión estratégica, donde las tareas de diagnóstico y clasificación de fallos juegan un rol muy importante con el fin de garantizar y mantener en operación continúa y confiable al proceso, los fallos pueden provocar desde una reducción del desempeño hasta un daño que provoque paradas en la producción. La generación distribuida de energía eléctrica a través de los grupos electrógenos instalados, no está ajena a sufrir fallas. Este trabajo tiene como objetivo el desarrollar un sistema para el diagnóstico y clasificación de fallos para la unidad de motores Diesel (MTU) del Emplazamiento de Grupo Electrógeno Camagüey 1. La solución propuesta constituye una herramienta para evaluar la aplicación de mantenimiento preventivo antes de la ocurrencia de un fallo.
Palabras clave: centrales eléctricas, confiabilidad, detección de fallas, diagnóstico, inteligencia artificial.

With the increment of the grade of dependence of the modern society of the systems and complex technological processes, their readiness and correct operation they have become a strategic question, where the tasks of diagnostic and classification of shortcomings plays a very important list with the purpose of to guarantee and to maintain in operation it continues and reliable to the process. The shortcomings can cause from a reduction of the acting until a damage that causes stopped in the production. The distributed generation of electric power through the groups installed, is not unaware to suffer shortcomings. This work has as objective the development of a system for the diagnosis and classification of shortcomings for the Diesel unit of motors (MTU) of the Location of Grupo Electrógeno Camagüey 1. The proposed solution constitutes a tool to evaluate the application of preventive maintenance before the occurrence of a failure.
Key words: *electric central, reliability, detection of faults, diagnostic, artificial intelligence.*

INTRODUCCIÓN

Una de las misiones en una red neuronal consiste en simular las propiedades observadas en los sistemas neuronales biológicos a través de modelos matemáticos recreados mediante mecanismos artificiales (como un circuito integrado o un ordenador).Una red neuronal, según Freeman y Skapura [1], es un sistema de procesadores paralelos conectados entre sí en forma de grafo dirigido.

Esquemáticamente cada elemento de procesamiento (neuronas) de la red se representa como un nodo. La idea básica consiste en encontrar un modelo del sistema, el cual está basado en redes neuronales.

Modelos de redes y arquitectura

Dentro del campo de las redes neuronales artificiales (RNA), existen varios modelos de redes y variadas arquitecturas, entre ellas se tienen:
El modelo neuronal de McCulloch y Pitts [2] fue el primer modelo neuronal moderno, y ha servido de inspiración para el desarrollo de otros modelos neuronales, este modelo por sí mismo está retomando importancia debido a que es uno de los pocos modelos digitales en tiempo discreto y tiene utilización en implantaciones electrónicas o computacionales por lo que en la actualidad se utilizan en sistemas digitales.
El modelo "ADALINE" y "MADALINE", son un tipo de red neuronal artificial desarrollada por BernieWidrow y MarcianHoff [3] en la Universidad de Stanford en 1959. ADALINE proviene de ADAptiveLINearElement y MADALINE de MultipleADAptiveLINearElement (muchas Adalines).
El MultiLayerPerceptron es reconocido por sus resultados en comparación con otros modelos como la mejor red neuronal para solucionar un problema de clasificación a partir de ejemplos.
La selección del tipo adecuado de arquitectura de la Red Neuronal (número de neuronas en cada capa y el número de capas) para cada caso concreto, es un problema empírico (prueba y error).
Las RNA, y más concretamente el perceptron multicapa, son presentadas como una estructura con capacidad de aproximación universal [4] y una estructura interna que permite su eficaz realización tanto utilizando software o hardware. Este mismo punto de vista es planteado por reconocidos investigadores Barron [5]. Sin embargo, ¿qué tienen de especial las RNA, aparte de sus ventajas de implantación, que les ha dado un protagonismo no compartido por otras estructuras de caja negra preexistentes? [5]. El investigador Ljung propone dos razones: la primera es que los sistemas que se suelen encontrar en la práctica presentan características de saturación para entradas elevadas. La propia estructura de las RNA incorpora este efecto de forma "natural", en contraposición, por ejemplo, a los aproximadorespolinomiales. La segunda razón, está relacionada con la redundancia interna de las RNA. Esta redundancia explica la relativa facilidad con que se maneja un elevado número de parámetros o pesos en el ajuste de RNA, cuando suele ser una tarea tan costosa con otros tipos de aproximadores.
Se puede concluir que las RNA se están aplicando al diagnóstico de procesos industriales, tanto a nivel de detección y aislamiento como de identificación de anomalías.

Arquitectura *Multilayer Perceptron*

Para la implementación del método se utilizó una red neuronal *Multilayer-Perceptron* (MLP).
Las redes MLP entrenadas por el algoritmo *BackPropagation*, son consideradas quizás las más generales de las Redes Neuronales Artificiales. Son usadas en problemas de predicción, clasificación, reconocimiento de patrones, estimación de parámetros y resolución de señales.
Este algoritmo de entrenamiento de las Redes Neuronales, tiene la ventaja de no necesitar un conocimiento previo de la forma de la señal analítica tratada, o sea, puede ser usado para modelar el sistema, que es de gran utilidad en los casos en que el modelo matemático que describe el sistema es desconocido. Además, el trabajo con este tipo de redes puede utilizarse con información heterogénea, o sea, los vectores de entrada pueden contener variables de diferentes naturaleza, lo que en principio permite su uso en problemas de naturaleza muy diferentes.

Desarrollo

Flujo de información del grupo electrógeno

Los métodos de detección de anomalías pueden dividirse según sus principios de operación en cuatro grandes categorías: el alcance de umbrales, los métodos basados en redundancia física, los métodos basados en redundancia analítica, y los métodos basados en criterios estadísticos.

Alcance de umbrales

Es la solución más extendida que consiste simplemente en comprobar la permanencia de variables individuales dentro de unos límites preestablecidos o umbrales. En caso de sobrepaso de estos límites se activan de forma automática las alarmas correspondientes, quedando en manos del personal encargado la interpretación de las mismas. Esta solución, pese a su extrema sencillez, tiene las siguientes desventajas (Gertler, [5]): en primer lugar el establecimiento de los límites permitidos suele realizarse con criterios muy conservadores debido al amplio rango de variación que pueden tener las variables medidas. En segundo lugar la aparición de una falla simple en un componente puede provocar el que varias variables excedan sus límites permitidos, complicando la tarea de identificación de la anomalía. La primera consecuencia de la aplicación práctica de esta metodología es que las indisponibilidades se detectan una vez que sus efectos han causado daños importantes. Por otro lado deja en manos del usuario la difícil tarea de interpretar la secuencia de alarmas que se produce con el fin de emitir un diagnóstico sobre el estado de los equipos y poder tomar acciones correctoras.

Las deficiencias descritas plantearon la necesidad de incorporar un conocimiento más profundo del proceso bajo estudio con vistas a realizar la acción del diagnóstico. Este conocimiento ha de incluir las ligaduras intrínsecas entre las distintas variables así como una perspectiva histórica de la evolución del proceso.

El SCADA tiene mecanismos que permiten la generación de eventos a partir de las señales ya procesadas. Una forma inmediata de generar dichos eventos es a partir del sobrepaso de umbrales establecidos convenientemente, este es el principio de funcionamiento de estos sistemas SCADA.

Descripción del flujo tecnológico

Los grupos electrógenos tienen instalado un SCADA (ver figura 1), que es supervisado en la sala de control remoto por el operador de turno, este monitorea un grupo de variables; mediante su interface gráfica le informa al operador si algunas de las variables está fuera de parámetro. Cuando el valor del parámetro pasa del límite permitido, el sistema muestra una alarma visual y sonora del problema ocurrido, entonces el operador realiza acciones correctivas para devolver o restablecer el régimen de operación del equipo e informar al nivel superior del fallo: al Centro de Control de Generación radicado en la empresa y al Departamento Técnico del Grupo de la Central.

En muchas ocasiones es necesario un mantenimiento para la solución del problema, este mantenimiento que es correctivo es una práctica muy costosa, ya que en ocasiones implica paradas no programadas, más los daños provocados por las fallas de los subsistemas.
Hay otro grupo de variables que en el Manual de Gestión GDECU, de la Unión Nacional eléctrica, indican su lectura y registro en el modelo de control de régimen con el objetivo de chequear el proceso de producción y mantener la disponibilidad.

Estas son verificadas por el operador de turno cada una hora, en caso de encontrarse con alguna fuera de parámetros, se le informa al departamento técnico y el operador vuelve a realizar las acciones de comprobación para detectar si hay desviación de régimen.

Con el sistema de diagnóstico de fallos se pretende que el especialista ingrese manualmente a un episodio del sistema las variables leídas para esperar un resultado más acertado de lo que puede ocurrir, así reducir el riesgo de paro por una alarma provocada por un fallo.

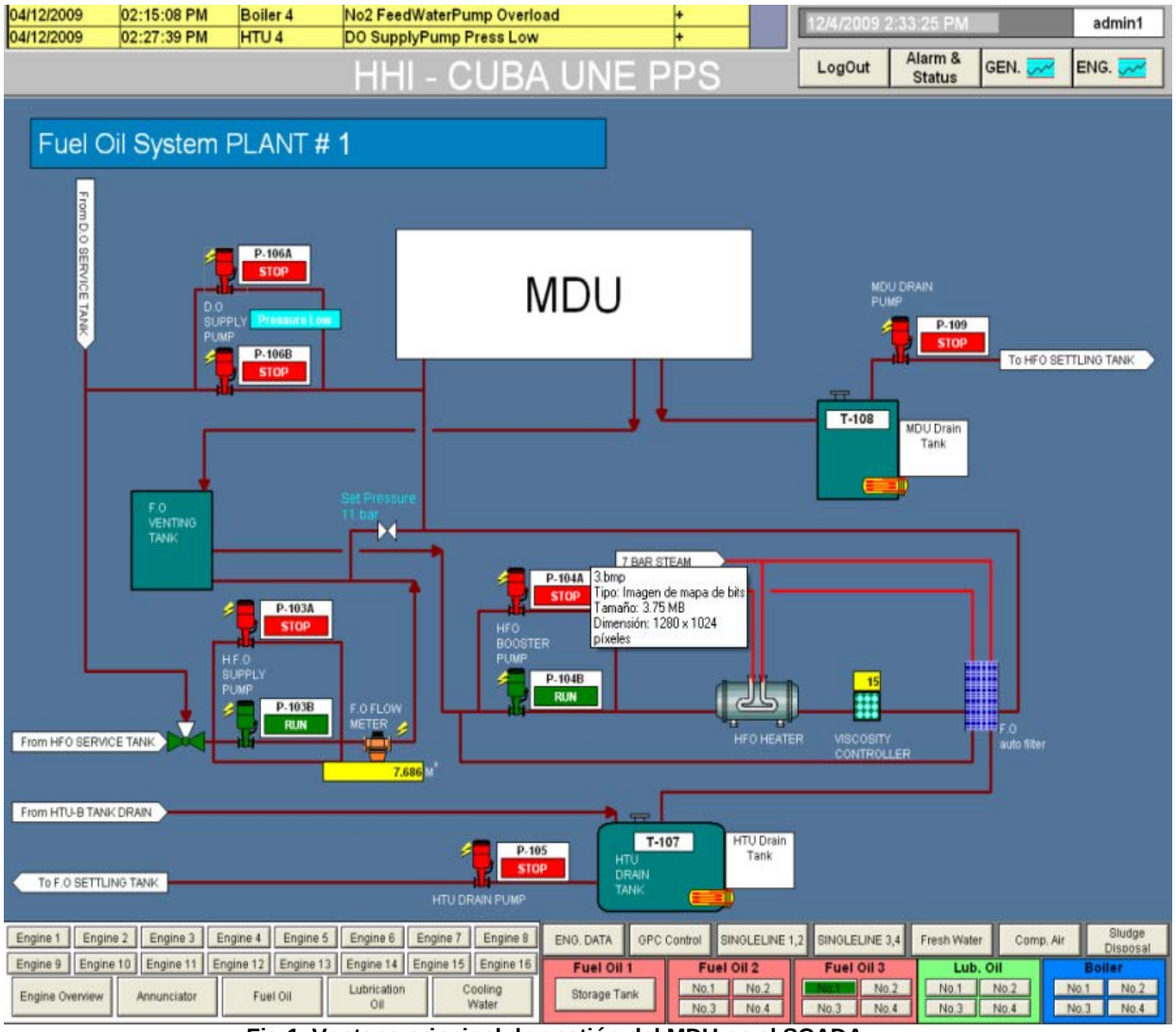


Fig.1. Ventana principal de gestión del MDU en el SCADA.

En la tabla 1, se relacionan las variables que permiten leer cada un tiempo determinado el estado de los sistemas y subsistemas del MDU por los operadores de turno, los cuales son recogidos en el *Book Instruction* y en la carta régimen del fabricante Hyundai.

Los sensores instalados en el motor (MDU), registran en tiempo real el valor de cada una de ellas.

Tabla 1. Variables del sistema medidas por los operadores.

Nombre de la variable	Registrada por SCADA	Rango valor	U/M
Potencia activa	X	50~100	kW
Velocidad motor		891~909	rpm
Velocidad turbo		<4000	rpm
Índice gobernador		3.5~6	unidad
Combustible presión	X	7~10	bar
Combustible temperatura	X	110~140	°C
Aceite lubricante nivel		50~75	%
Aceite lubricante temperatura	X	60~70	°C
Aceite lubricante presión motor	X	4~5	bar
Aceite lubricante presión llenado		5~6	bar
Aceite lubricante presión diferencial		0.1~1	bar
Aceite lubricante presión turbo	X	2~4	bar

Uno de los aspectos fundamentales a la hora de utilizar herramientas de Inteligencia Artificial para el análisis de situaciones en procesos industriales es el tratamiento de las señales provenientes del proceso. Se trata, en definitiva, de extraer y codificar, a partir de las señales, aquella información útil sobre los fallos que deben detectarse o diagnosticarse.
Si el objetivo es evaluar estas señales para decidir sobre el estado del proceso, deben establecerse los mecanismos que permitan tratar los diversos problemas que pueden afectarlas, como pueden ser la imprecisión, la incertidumbre, la ausencia o la cantidad excesiva de información.
En esta aplicación se evalúan las primeras 12 variables del sistema motor (MDU). Las cuales corresponden con los subsistemas:

- Variables generales del motor.
- Subsistema combustible.
- Subsistema aceite de lubricación.

Estas son las variables de entradas al sistema, el valor prescrito de cada variable cuenta con un rango de operación definido por un valor máximo y un valor mínimo. Estos valores se utilizan construir un umbral fijo el cual disminuye la complejidad del trabajo y los cálculos. Para la generación de los residuos se tiene en cuenta la comparación del valor de la variable con el umbral fijo de la misma.

Creación de la matriz teórica de firmas de fallo:
1. Cada columna corresponde a un posible fallo
2. Cada fila corresponde a un síntoma
3. La matriz está formada por 1 y 0
4. Un 1 indica que cuando ocurre el fallo al que se refiere la fila se produce el síntoma que indica la columna. Un 0 indica lo contrario.

En la tabla 2, se observa un ejemplo de firma de fallo.

Tabla 2. Ejemplo de firma de fallo.

	f1	f2	f3	f4	f5
residuo1	1	0	1	0	0
residuo2	0	1	0	1	0

El documento revisado para estudiar las fallas ocurridas, es el libro de incidencias de turno, que recoge todas las averías que ocurren diariamente. A partir de esos datos, tomando un período de 1 año, se clasificaron las averías del MDU, encontrando las siguientes características:
1. Hay fallas encontradas que no tienen incidencia en ninguna de las variables que se leen, en estos casos se tienen fallos por salideros y que son detectados por inspección visual del operador.
2. Las fallas se presentan después de que se ha alcanzado el estado estable de operación del motor.
3. Las fallas encontradas a priori,para las cuales las variables han modificado su valor son las que se relacionan a continuación:
1. Alta presión diferencial
2. Alta temperatura salida cilindros
3. Alta temperatura entrada en el turbocompresor
4. Alta temperatura salida en el turbocompresor
5. Alta velocidad en el turbocompresor
6. Baja temperatura salida cilindros
7. Falla en el gobernador
8. Disparo por sobre-velocidad
9. Falla en motor de arranque
10. Oscilaciones de potencia
11. Bajo nivel aceite del cárter
12. Alta temperatura aceite lubricante
13. Alto índice del gobernador
14. Falla sincronización
15. Falla por sobre-velocidad

Se diseñó una encuesta para ser aplicada a los expertos de la OBE y especialistas del Emplazamiento de Grupo Electrógeno, con el objetivo de determinar su criterio sobre las fallas que ocurren cuando las lecturas sobrepasan los umbrales.
A partir de esta información se creó la matriz de incidencia o matriz teórica de fallos. (Ver figura 3, Matriz T).

Diseño del sistema de clasificación basado en las redes neuronales artificiales

El proceso del diseño consiste en los siguientes pasos:
1. Construcción de los conjuntos de entrenamiento y prueba.
2. Selección de la arquitectura de la Red Neuronal Artificial.
3. Entrenamiento.
4. Evaluación de la red entrenada usando patrones de prueba.

Construcción de los conjuntos de entrenamiento y prueba

La red primero debe ser entrenada con un conjunto de entrenamiento el cual incluye ejemplos de datos en buen funcionamiento y otros con fallo. Al finalizar el entrenamiento, la red deberá estar lista para reconocer los ejemplos aprendidos, y clasificar otros nuevos basándose en las generalizaciones hechas a partir del aprendizaje [6-7].
Para usarla se emplea como una función, la cual se evalúa y da un resultado. En la evaluación, la red recibe un vector de entrada de componentes reales que identifica a un patrón determinado, y luego de “analizarlo”, devuelve la clase o patrón al cual debe pertenecer dicho vector.
El entrenamiento es el proceso durante el cual la red MLP aprende los ejemplos que se le enseñen y durante el mismo clasifica a cada ejemplo del conjunto de entrenamiento y, en dependencia del error que comenta, se rectificará ella misma, para cuando vuelva a evaluar a ese ejemplo, intentar hacerlo mejor. Luego de clasificar a todos los ejemplos, se considera que la red ha efectuado un paso del algoritmo de entrenamiento. Este proceso se ejecutará hasta que se cumpla una condición de parada.

Caracterización del conjunto de entrenamiento

Usualmente se evalúa la calidad del conocimiento resultante de aplicar algún método de aprendizaje usando el conjunto de control; es decir, la evaluación es post-aprendizaje. A partir de los datos disponibles se aplican diferentes métodos de aprendizaje para determinar cuál produce un mejor conocimiento. El estudio de la relación entre el conjunto de entrenamiento y la eficiencia y eficacia lograda en el proceso de aprendizaje se realiza usando el método de prueba y error de una forma experimental, tal y como se realizó este trabajo. Es decir, se realizan sucesivos procesos de entrenamiento y se validan los resultados alcanzados.
De modo que resultaría de gran interés poder estimar la calidad de los datos antes de proceder al aprendizaje para evitar trabajos innecesarios.
Para el tipo de red que se implementó, cuando los conjuntos de entrenamientos son grandes se recomienda aplicar las funciones de pre-procesamiento y post-procesamiento a las entradas y salidas de la red, que tienen como objetivo que el entrenamiento pueda ser más eficiente y rápido. Estas funciones se relacionan a continuación:

Antes del entrenamiento

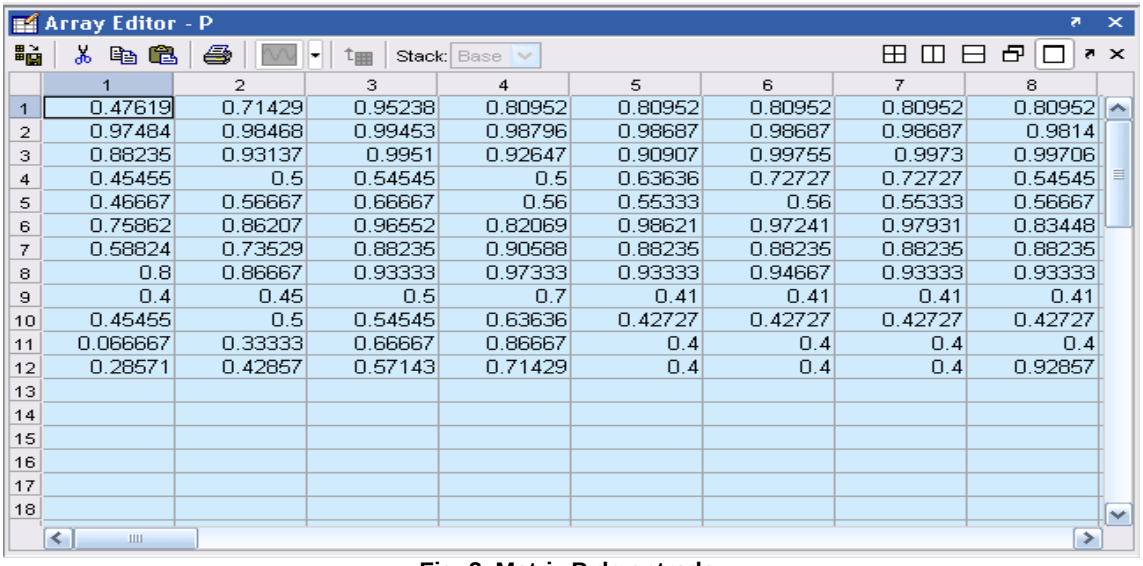
En muchas ocasiones es útil normalizar las entradas y salidas para que siempre caigan dentro de un rango específico, las funciones premnmx, postmnmx y tramnmx son implementadas en MatLab para realizar esta tarea.
Otra forma de disminuir las entradas y salidas de una red es normalizar la media y la desviación estándar del conjunto de entrenamiento. Se normalizan las entradas y salidas de modo que ellas tendrán cero medias y desviación estándar uno. Este procedimiento es implementado con las funciones prestd, poststd y trastd.
En muchas ocasiones la dimensión del vector de entrada es grande, pero los componentes de los vectores están altamente correlacionados (hay redundancia). Es útil en esta situación reducir la

dimensión de los vectores de entrada. Las funciones prepca y trapca son implementadas para realizar esta tarea.
Se dividió el conjunto de entrenamiento en 3 subconjuntos: subconjunto de entrenamiento con 60%, validación con 20% y prueba con el otro 20% y se observó el gráfico de cómo se comporta durante en el entrenamiento la tendencia de los 3 subconjuntos.

Después del entrenamiento

El desempeño de una red adiestrada puede ser medido hasta cierto punto por los errores en el entrenamiento, validación y conjuntos de prueba, pero es a menudo útil investigar la respuesta de la red en más detalle. Una opción es realizar un análisis de regresión entre la respuesta de la red y la correspondiente salida esperada. La rutina diseñada para realizar este análisis es postreg.
Probar a = sim(net,p);
[m,b,r] = postreg(a,t)

En este caso no se realizó ninguna de las tareas de pre-procesamiento debido a que el conjunto de entrenamiento no es grande y esta normalizado dentro de en un rango de 0 a 1.
La matriz P (figura 2), es la entrada con lecturas en condiciones normales de operación y en condiciones de fallos, la salida esperada es la matriz T con valores de 0 y 1. (figura 3).
La base de datos consta con un total de 1000 datos, de los cuales 850 fueron escogidos para el entrenamiento de la red y 150 fueron tomados para la prueba, todos fueron escogidos de manera aleatoria.

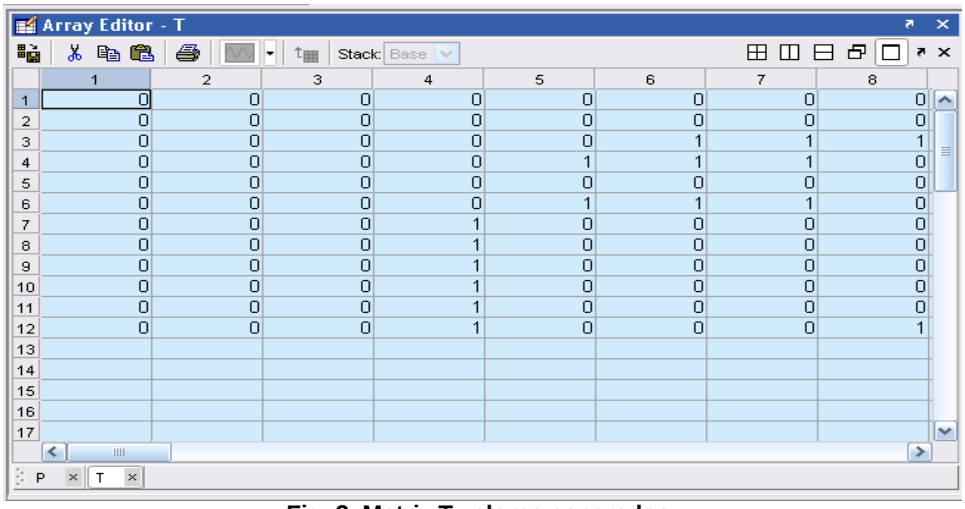


	1	2	3	4	5	6	7	8
1	0.47619	0.71429	0.95238	0.80952	0.80952	0.80952	0.80952	0.80952
2	0.97484	0.98468	0.99453	0.98796	0.98687	0.98687	0.98687	0.9814
3	0.88235	0.93137	0.9951	0.92647	0.90907	0.99755	0.9973	0.99706
4	0.45455	0.5	0.54545	0.5	0.63636	0.72727	0.72727	0.54545
5	0.46667	0.56667	0.66667	0.56	0.55333	0.56	0.55333	0.56667
6	0.75862	0.86207	0.96552	0.82069	0.98621	0.97241	0.97931	0.83448
7	0.58824	0.73529	0.88235	0.90588	0.88235	0.88235	0.88235	0.88235
8	0.8	0.86667	0.93333	0.97333	0.93333	0.94667	0.93333	0.93333
9	0.4	0.45	0.5	0.7	0.41	0.41	0.41	0.41
10	0.45455	0.5	0.54545	0.63636	0.42727	0.42727	0.42727	0.42727
11	0.066667	0.33333	0.66667	0.86667	0.4	0.4	0.4	0.4
12	0.28571	0.42857	0.57143	0.71429	0.4	0.4	0.4	0.92857
13								
14								
15								
16								
17								
18								

Fig. 2. Matriz P de entrada.

Selección de la arquitectura de la Red Neuronal Artificial

Sin lugar a dudas, la topología de red neuronal mejor conocida y más utilizada en la actualidad para la solución de problemas de clasificación es la *MultiLayerPerceptron* (MLP). La literatura, para soluciones del tipo de problema, hace referencia a la utilización de redes multicapa con conexiones hacia delante (*feed-forward*), aprendizaje supervisado y corrección de error aplicando el algoritmo de propagación de los errores hacia atrás (*backpropagation*).



	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1	1
4	0	0	0	0	1	1	1	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	1	1	1	0
7	0	0	0	1	0	0	0	0
8	0	0	0	1	0	0	0	0
9	0	0	0	1	0	0	0	0
10	0	0	0	1	0	0	0	0
11	0	0	0	1	0	0	0	0
12	0	0	0	1	0	0	0	1
13								
14								
15								
16								
17								

Fig. 3. Matriz T valores esperados.

Las redes *Feedforward* a menudo tienen una o más capas ocultas de neuronas sigmoid seguidas por una capa de salida de neuronas lineales. Múltiples capas de neuronas con funciones de transferencia no lineales permite que la red aprenda relaciones no lineales y lineales entre los vectores de entrada y salida. La figura 4, muestra la topología de esta red.
Esta red puede ser usada como un aproximador general de funciones, puede aproximar cualquier función con un número finito de discontinuidades, arbitrariamente con un número suficiente de neuronas en la capa oculta.

Partiendo de lo anterior definimos una red MLP de 2 capas, una capa oculta con 30 neuronas y 12 en la salida, la primera capa con función de transferencia tansig y la segunda purelin.
Esta red*Perceptron* con dos capas, puede formar cualquier región convexa en el espacio. El número de neuronas en la capa oculta debe ser lo suficientemente grande como para que se forme una región compleja que pueda resolver el problema, es decir, que la red sea capaz de generalizar, aunque como se mencionó anteriormente es un problema de prueba y error encontrar una topología adecuada. La red que se seleccionó primeramente es muy utilizada, aunque se ha encontrado en la bibliografía revisada [8], una red *perceptron* con una capa oculta de 40 o más neuronas, también para la solución a problemas de clasificación.

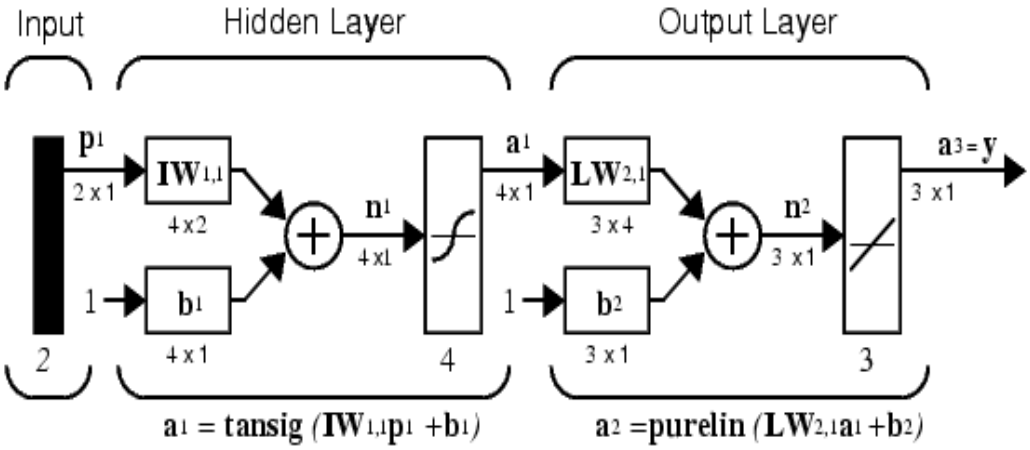


Fig. 4. Ejemplo de topología de red Feedforward.

La topología propuesta para la red neuronal se implementó en MatLab, a continuación se detalla la definición, los parámetros de la red y el entrenamiento.
Es posible que una arquitectura particular de Red Neuronal Artificial seleccionada no pueda ser entrenada a satisfacción del diseñador, por lo que su estructura y parámetros deben cambiarse y la red debe ser reentrenada. De igual forma una red ya entrenada puede no ofrecer buenos

resultados ante patrones de prueba y entonces los parámetros y la red deben cambiarse, reentrenarse y volver a someterse a prueba.
El método a seguir para desarrollar la red neuronal multicapa, fue basado en los siguientes pasos:

Para la primera red, que funciona como generador de residuos, se tiene:
Primer paso: Tener un punto de partida, empezar las pruebas a partir de la topología propuesta 30 neuronas en la capa oculta.
Inicialmente se hicieron las pruebas cambiando las funciones de entrenamiento, la primera prueba fue con el **algoritmo de entrenamiento "*BatchGradientDescent*"** ("traingd"),
Modelo 1: Algoritmo de entrenamiento (traingd), arquitectura 12:30:12 (12 neuronas en la capa de entrada, 30 en la capa oculta y 12 en la capa de salida).
El resultado, al probar la red no fue el esperado.

Segundo paso: una primera idea fue reiniciar la red y volver a entrenarla, obteniendo así resultados diferentes, pues, por ejemplo, los valores iniciales de las ganancias y las matrices de pesos, usados como punto de partida, son diferentes, ofreciendo el método del gradiente descendiente otra convergencia.
Para reiniciar la red con nuevos pesos se usa el comando init:
>> red1 = init(red1);
Y así volver a repetir el proceso anterior, si no encontramos un resultado satisfactorio entonces probar modificando el modelo, en este caso cambiar la función de entrenamiento.
Durante el entrenamiento los pesos y los *bias* de la red son iterativamente ajustados para minimizar la función de desempeño net.performFcn. La función de desempeño por defecto para redes *feedforward* es el error cuadrático, el error cuadrático promedio entre la salida de la red y la salida esperada.
Hay 2 formas diferentes fundamentales en las cuales el algoritmo del gradiente descendiente puede ser implementado: modo incremental y modo *batch*. En el modo incremental el gradiente es calculado y los pesos son actualizados después de que cada entrada es aplicada a la red. En el modo *batch* todas las entradas son aplicadas a la red antes que los pesos sean actualizados.
Hay varios algoritmos de mejor rendimiento que han sido implementados sobre el modo *batch* que pueden converger desde 10 a 100 veces más rápido que el algoritmo de entrenamiento probado anteriormente, por lo que se prueba con otros algoritmos de entrenamiento:

Modelo 2: Algoritmo de entrenamiento trainbfg, arquitectura 12:30:12.
El entrenamiento arrojó el siguiente resultado: El número de iteraciones alcanzó el máximo=1000 por lo que finaliza el entrenamiento por cumplirse esta condición.
Las pruebas realizadas al simular la red con nuevos valores, no se corresponden con el resultado esperado, la generalización no es buena en todos los casos.

Modelo 3: Algoritmo de entrenamiento trainlm, arquitectura12:30:12.
El entrenamiento arrojó el siguiente resultado: El número de iteraciones alcanzó el máximo=1000 por lo que finaliza el entrenamiento por cumplirse esta condición.
Las pruebas realizadas al simular la red con nuevos valores aún no se corresponden con el resultado esperado, hay algunos ejemplos que la salida no se corresponde con la esperada.

Tercer paso: Un método para mejorar la generalización de la red, es incrementar el número de neuronas en la capa oculta, esto puede permitir obtener mejores resultados en la simulación, pues se cuenta con más parámetros para obtener una mejor optimización. En este caso se incrementa el número de neuronas en la capa oculta a 40.Sin embargo, existe una cota superior para el número de neuronas en las capa ocultas, demasiadas neuronas podrían conseguir que los patrones de entrada fueran memorizados ("*overfitting*"), de manera que ningún tipo de generalización fuera posible, haciendo que la red fuera inútil con nuevos datos de entrada. Aunque el error cuadrático medio obtenido es muy pequeño, para los patrones de entrenamiento, resulta muy grande cuando nuevos datos son presentados a la red. La red ha memorizado las muestras de entrenamiento, pero no ha aprendido el comportamiento a seguir en situaciones nuevas (patrones de entrada no empleados en la fase de entrenamiento).

Modelo 4: Algoritmo de entrenamiento trainlm, arquitectura 12:40:12.

El entrenamiento arrojó el siguiente resultado: En esta red se puede observar como el error medio cuadrático ha encontrado el mínimo, es decir, como la red neuronal va aprendiendo el comportamiento a seguir.
Se realizó un análisis de las variables y la incidencia de los fallos y se determinó realizar el proceso con las primeras 4 variables (un solo subsistema), ya que de los 13 tipos de fallos analizados sólo uno tiene incidencia en las variables de 2 subsistemas a la vez; otros 3 fallos sólo inciden en las variables de la 6 a la 12. Se concluye que en estas 4 variables tienen incidencia el 70% de los fallos analizados, por lo que se pasó a probar con una red de 4 variables de entrada y 4 en la salida.
Modelo 5: Algoritmo de entrenamiento rainlm, arquitectura4:16:4.
Los resultados del entrenamiento con conjuntos de prueba y validación se muestran en la figura 5.

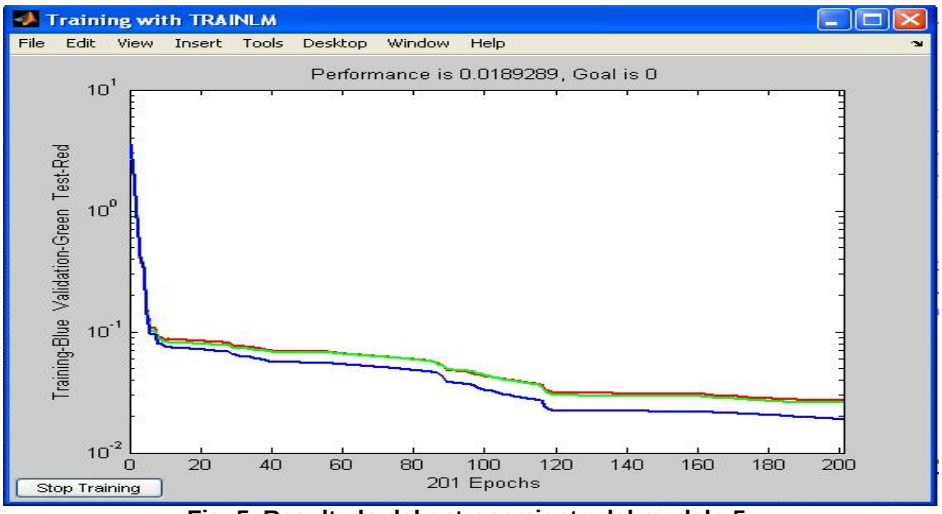
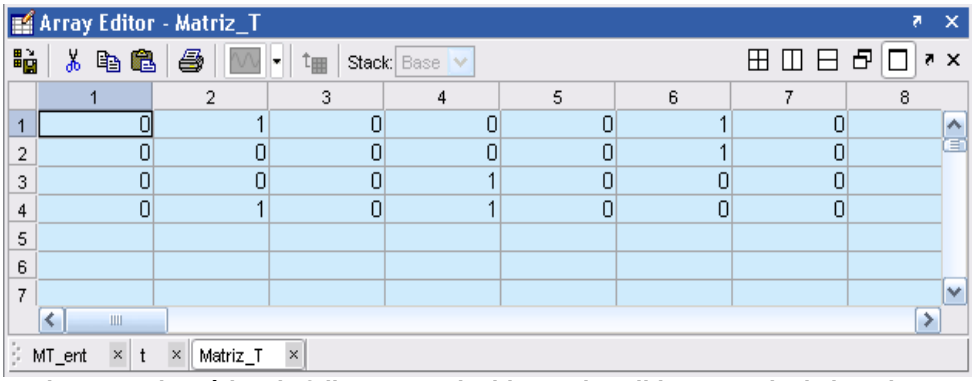


Fig. 5. Resultado del entrenamiento del modelo 5.

La simulación de la red con un conjunto de prueba, mostró resultados satisfactorios, comprobándose que esta red si clasifica correctamente un conjunto de datos nuevos para ella, con un índice de 99.9% de certeza.
Para la segunda red, que funciona como un clasificador de residuos, se tiene:
Para la creación de la segunda red se tiene la experiencia de la anterior, por lo que se selecciona una red con 16 neuronas en la capa oculta y con la función de entrenamiento trainlm.
Primer paso: comenzar el entrenamiento y valorar la convergencia. Los resultados se muestran en la figura 6.



	1	2	3	4	5	6	7	8
1	0	1	0	0	0	1	0	
2	0	0	0	0	0	1	0	
3	0	0	0	1	0	0	0	
4	0	1	0	1	0	0	0	
5								
6								
7								

Fig. 6. Matriz teórica de fallos, que coincide con la salida esperada de la red # 2.

En la figura 7, se muestra el resultado del entrenamiento realizado a la red # 2.

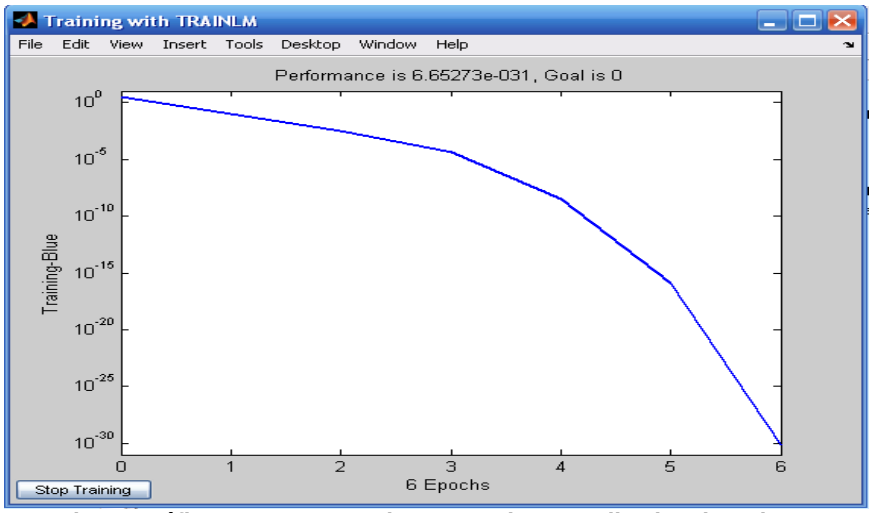


Fig. 7. Gráfica que muestra el entrenamiento realizado a la red # 2.

Evaluación de la red entrenada usando patrones de prueba.

Para evaluar la capacidad de generalización de las 2 redes se utilizó un conjunto de patrones de prueba (150 patrones) y se calculó el porcentaje de clasificaciones incorrectas ante estos nuevos vectores de entrada.

Para la selección de la red a usar se tuvo en cuenta los resultados de:

1. Convergencia de la red en el entrenamiento (figura 8).
2. Capacidad de generalización (figura 9).

En la figura 8, se muestra un resumen de los resultados obtenidos durante el entrenamiento y con el conjunto de prueba en la figura 9.

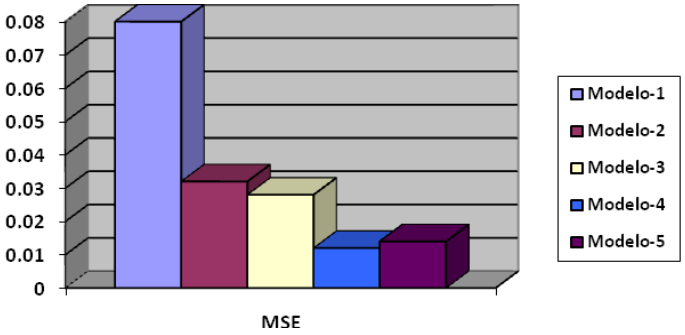


Fig. 8. Resultados del entrenamiento por la función MSE.

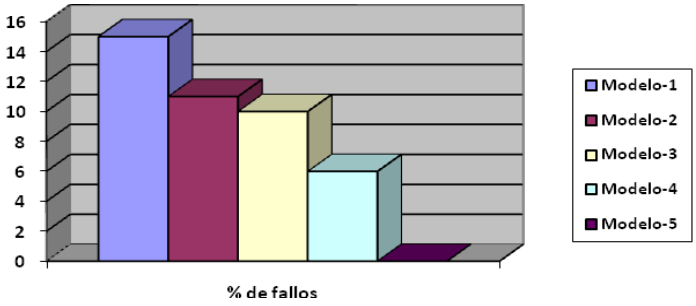


Fig.9.Resultados del entrenamiento por él % de clasificaciones incorrectas del conjunto de pruebas.

De acuerdo a los resultados obtenidos se escoge el modelo 5 que es el que mejores resultados ofrece.

Características del Software desarrollado

La aplicación diseñada es una herramienta para apoyar el trabajo del especialista principal del grupo electrógeno, se programó en MatLab 7.0.0.1, que ofrece muchas ventajas para el trabajo con matrices y redes neuronales, se dividió en 5 módulos, el principal y 2 procedimientos que se corresponden con cada una de las funciones que realiza la aplicación y los otros 2 son los módulos para la creación y entrenamiento de las redes.
Modulo principal: es el que se le presenta al usuario, este le muestra todas las posibles acciones a realizar. Ver figura 10.

Fig. 10. Ventana del módulo principal.

Opción Introducir episodio: Le permite al usuario la introducción de un episodio, durante este proceso se hace una validación de los datos introducidos; el episodio es guardado en un fichero texto con un nombre seleccionado por el usuario para que posteriormente sea utilizado. Se implementó de esta forma con el propósito de que en futuras versiones se puedan obtener los datos del proceso en línea. Figuras 11 y 12.

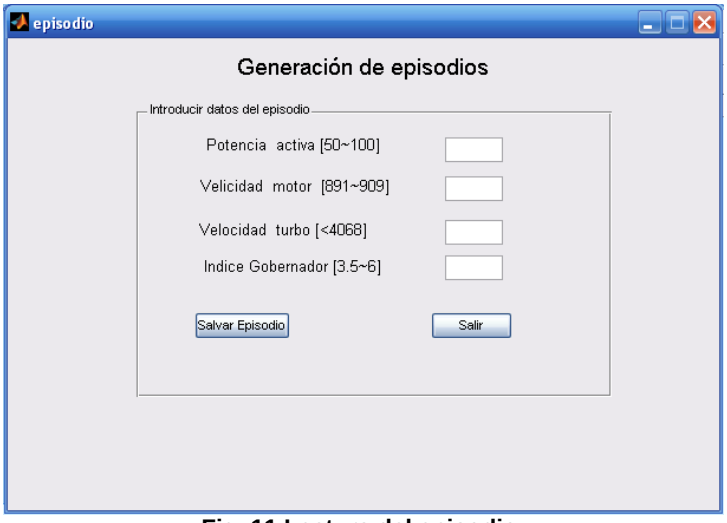


Fig. 11.Lectura del episodio.

Fig. 12. Resultado del diagnóstico.

CONCLUSIONES

1. La red neuronal utilizada fue la *MultiLayer Perceptron*. Para lograr la arquitectura deseada se realizaron pruebas con diferentes funciones de entrenamiento, con varios conjuntos de entrenamiento y con diferentes números de neuronas en la capa oculta hasta llegar a una arquitectura adecuada constituida por 4 neuronas en la capa de entrada, 16 en la oculta y 4 en la salida.
2. De acuerdo al resultado anterior se desarrolló la aplicación para el diagnóstico y clasificación de fallos para el MDU de los Grupos Electrógenos con motores Hyundai, para el subsistema de variables generales.
3. El sistema presentado está dirigido a prevenir un fallo en el MDU, basado en la evaluación de las variables de entrada. Genera una salida que le muestra al operador si es necesario proponer un mantenimiento preventivo. Fue probado con un resultado de 99,95% de clasificaciones correctas.

RECOMENDACIONES

Se recomienda para futuros desarrollos acoplar el sistema al SCADA del grupo electrógeno para obtener las lecturas en línea de las variables que se van a procesar.
Otra línea de desarrollo es la ampliación del diagnóstico de fallos al resto de los subsistemas que operan en la central ya que el actual sólo se subscribe a 12 variables del MDU.

REFERENCIAS

[1] Freeman, J.A.; Skapura, DM. “Redes Neuronales. Algoritmos, aplicaciones y técnicas de propagación”. México : Addison-Wesley, 1993.
[2] Wipedia. “Neurona de McCullosh y Pitts”. [En línea]. [consulta: 22 de Julio de 2011]. Disponble en: http://es.wikipdia.org/wiki/neuro.
[3] Yañez Márquez, León. “Introducción a las memorias asociativas”. Cuba. Santiago: Editorial Universitaria, 2008.
[4] Sorsa, T.; Kolvo, H. N. “Application of artificial neural networks in processes fault diagnoses”. En: presentado en IFAC/IMAC Symposium on fault detection supervision and safety for technical Processes Safeprocess’91. Baden-Baden : s.n., 10-13 Sept. 91, 1991.
[5]. San Roque, Antonio Muñoz.“Aplicación de técnicas de redes neuronales artificiales al diagnóstico de procesos industriales” [tesis doctoral]. Escuela técnica superior de ingenieros industriales, Universidad Pontificia Comillas Madrid, 2008.

[6] Mota Caballero, Yaile.“Teoría de los Conjuntos Aproximados, la selección de rasgos y la edición de conjuntos de entrenamiento. Métodos de aprendizaje a partir de ejemplos. Camagüey” : [en formato digital]. Cuba. Universidad de Camagüey, Facultad de Informática, 2007.
[7] Caballero Y, Arco L, Bello R, Salgado Y, Márquez Y, León P, et al. “Nuevas medidas de la Teoría de los Conjuntos Aproximados para la evaluación de sistemas de información en Bioinformática”. En: presentado en II Congreso Internacional de Bioinformática y Neuroinformática Informática 2007. La Habana, 2007.
[8] de la Fuente, María Jesús.“Detección y Diagnóstico de fallos usando redes neuronales”. [Tesis de doctorado]. Valladolid: Dpto. Ingeniería de sistemas y automática. Universidad de Valladolid, España, 2008.

AUTORES

Luis Corrales Barrios
Ingeniero electricista. Doctor en Ciencias Técnicas. Profesor titular. Dpto. Ingeniería Eléctrica. Facultad de Electromecánica. Universidad de Camagüey, Cuba.
e-mail: luis.corrales@reduc.edu.cu

Alexei Ramírez Vázquez
Ingeniero en Máquinas computadoras. Máster en informática aplicada. Oficial del MININT, Delegación Provincial del MININT Camagüey, Cuba.
e-mail: alexei.ramirez@reduc.edu.cu