

Sistema de medición y control de proceso industrial de embotellado con algoritmo de redes neuronales

César E. Martínez Reinoso, Mario Alvarez Sifontes, Francisco J. Arteaga Bravo
*Unidad de Investigación en Automatización Industrial, Escuela de Ingeniería Eléctrica,
Universidad de Carabobo, Valencia, Venezuela*
Email: cmartinez79@hotmail.com, marioalvarez95@hotmail.com, farteaga@uc.edu.ve

Resumen

Las redes neuronales artificiales son algoritmos capaces de procesar y almacenar conocimiento experimental y tenerlo disponible para usarlo lo cual representa una gran ventaja en control y automatización. Este trabajo presenta un algoritmo de redes neuronales para el control de un proceso de embotellado. Se define la función y estado de los sensores y equipos, y las salidas de los equipos en función de los estados de los sensores, determinándose como apropiada una red perceptron. Su arquitectura y sus parámetros se muestran durante la simulación. La red clasifica correctamente los patrones de entrada y aprende exitosamente el proceso, lo que evidencia su importancia y aplicabilidad en automatización.

Palabras claves: Red neuronal, perceptron, control, algoritmo de aprendizaje, sensores.

Measurement and control system for a bottling industrial process using a neural network algorithm

Abstract

Artificial neural networks are algorithms capable of processing and storing experimental knowledge and make it available for its use, and this represents a great advantage in process control and automation. This work presents a neural network algorithm for the control of a bottling process. The function and status of sensors and equipment and the equipment output in terms of sensor status are defined, and a perceptron network is determined to be appropriate. Its architecture and its parameters are shown during simulation. The network classifies correctly the input patterns and successfully learns the process, which evidences its importance and applicability in automation.

Keywords: Neural network, perceptron, control, learning algorithm, sensors.

1. INTRODUCCIÓN

El uso de las redes neuronales artificiales está ampliamente difundido en aplicaciones tales como adquisición y procesamiento de datos en tiempo real, modelación de sistemas no lineales, reconocimiento de patrones, sensores inteligentes, etc [1-4]. Sin embargo, en el medio industrial actual, donde la automatización de procesos es la principal tendencia, las herramientas más utilizadas son los controladores lógicos programables (PLC). Estos son dispositivos muy eficaces y

de relativa facilidad para el manejo por parte del usuario, pero poseen una desventaja. El programador debe conocer todo el universo de entradas posibles, para poder introducir en el programa del controlador los comandos necesarios que debe ejecutar para obtener la salida deseada para cada entrada al sistema.

Es posible desarrollar un algoritmo con redes neuronales en Matlab que realice la función del PLC, con la ventaja de que, por un lado, el programador necesita conocer sólo una parte del universo total de entradas al sistema para que el algoritmo sea capaz de suministrar la salida deseada aún cuando la entrada

nunca se haya introducido. Por otra parte, se tiene el ahorro económico que significa la adquisición de un dispositivo PLC, teniendo la posibilidad de realizar la misma función con un computador con Matlab, bajo el cual corra la aplicación para el control de embotellado en línea.

2. DESCRIPCIÓN DEL PROCESO

Las botellas que se van a llenar y tapar llegan a través de una banda transportadora, la cual es accionada por un motor (el cual gira en sentido horario). Dicho motor también es el que acciona el movimiento circular de la ruleta que transporta a las botellas. Las botellas van llegando a la ruleta y se introducen en unas ranuras para ser transportadas. El motor hace que la ruleta gire, haciendo que las botellas vayan ocupando las ranuras de la ruleta. A medida que la ruleta gira, las botellas se van llenando y tapando por medio de la surtidora y la enchapadora respectivamente.

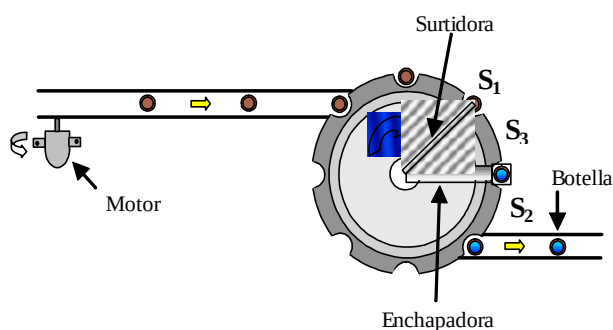


Figura 1. Vista superior del proceso.

Finalmente las botellas salen de este proceso llenas y tapadas a través de una segunda banda transportadora hacia otra zona de la planta. Para realizar este proceso se cuenta con tres sensores de estados discretos (S_1 , S_2 y S_3), los cuales son las entradas a la red neuronal a implementar.

Además de los sensores se tienen los equipos que servirán para mover, llenar y tapar las botellas como lo son: el motor, la surtidora y la enchapadora, los cuales servirán como salidas de la red. Dichos equipos también se activan con entradas discretas (0 ó 1) y funcionan como se describe en la Tabla 2.

Las condiciones del proceso son las siguientes:

Tabla 1. Función y estado de los sensores del proceso.

Sensor	Función	Edo.	Acción
S_1	Detecta si hay botella en la ranura para que la llene la surtidora.	0	No hay botella para llenar.
		1	Si hay botella para llenar.
S_2	Detecta si la botella fue tapada por la enchapadora.	0	No le fue colocada la chapa a la botella.
		1	Chapa colocada correctamente a la botella.
S_3	Detecta si la botella fue llenada correctamente por la surtidora.	0	Botella vacía o llenada incorrectamente.
		1	Botella llenada correctamente.

El motor debe estar en marcha sólo cuando: 1) No exista una botella para llenar, la botella no haya sido llenada correctamente, o si se le colocó mal la chapa a la botella ubicada en la enchapadora (si existe). 2) La botella ubicada en la surtidora se haya llenado correctamente, o si se le colocó mal la chapa a la botella ubicada en la enchapadora (si existe). 3) La botella ubicada en la surtidora se haya llenado correctamente y se le haya colocado correctamente la chapa a la botella ubicada en la enchapadora.

La surtidora debe llenar sólo cuando: 1) Exista botella en la surtidora, la botella no haya sido llenada correctamente, o si se le colocó mal la chapa a la botella ubicada en la enchapadora (si existe). 2) Exista

Tabla 2. Función y estado de los equipos del proceso.

Equipo	Función	Edo.	Acción
Motor	Mueve bandas transportadoras y ruleta.	0	Motor detenido.
		1	Motor en marcha.
Surtidora	Llena botellas con el producto líquido.	0	Surtidora parada.
		1	Surtidora llenando la botella.
Enchapadora	Coloca chapas a botellas completamente llenadas.	0	Enchapadora parada.
		1	Enchap. colocando chapa a la botella.

botella en la surtidora, la botella no haya sido llenada correctamente, y se le haya colocado correctamente la chapa a la botella ubicada en la enchapadora.

La enchapadora debe tapar sólo cuando: 1) No exista una botella para llenar, la botella no haya sido llenada correctamente, y se le haya colocado correctamente la chapa a la botella ubicada en la enchapadora. 2) Exista botella en la surtidora, la botella no haya sido llenada correctamente, y se le haya colocado correctamente la chapa a la botella ubicada en la enchapadora. 3) La botella ubicada en la surtidora se haya llenado correctamente y se le haya colocado correctamente la chapa a la botella ubicada en la enchapadora.

Tabla 3. Salida de los equipos en función de los estados de los sensores.

S ₁	S ₂	S ₃	Enchap.	Surtidora	Motor
0	0	0	0	0	1
0	0	1			
0	1	0	1	0	0
0	1	1			
1	0	0	0	1	0
1	0	1	0	0	1
1	1	0	1	1	0
1	1	1	1	0	1

Los estados sin salidas (sombreados), no son físicamente posibles [4].

3. DEFINICIÓN DEL TIPO DE RED A IMPLEMENTAR

Una red tipo Perceptron puede ser entrenada con patrones binarios de cualquier dimensión en la entrada y en la salida. Debido a que los sensores y los accionamientos operan con señales binarias, este tipo de red es la más adecuada. Para garantizar que el problema pueda ser resuelto por una red perceptron, se debe comprobar que los patrones de entrenamiento son linealmente separables. Para ello se deben plantear las desigualdades generadas por cada patrón de entrenamiento [5]. Cada patrón de tres dimensiones generará tres desigualdades (una por cada salida). Estas desigualdades no deben contradecirse, si esto ocurriera el problema no podría ser resuelto por la red monocapa. Debido a la naturaleza discreta de las salidas, la fun-

ción de transferencia a utilizar es hardlim [6].

$$p_1 = \begin{cases} b_1 < 0 \\ b_2 < 0 \\ b_3 < 0 \end{cases} \quad (1)$$

$$p_2 = \begin{cases} w_{12} + b_1 \geq 0 \\ w_{22} + b_2 < 0 \\ w_{32} + b_3 < 0 \end{cases} \quad (2)$$

$$p_3 = \begin{cases} w_{11} + b_1 < 0 \\ w_{21} + b_2 \geq 0 \\ w_{31} + b_3 < 0 \end{cases} \quad (3)$$

$$p_4 = \begin{cases} w_{11} + w_{13} + b_1 < 0 \\ w_{21} + w_{23} + b_2 < 0 \\ w_{31} + w_{33} + b_3 \geq 0 \end{cases} \quad (4)$$

Las ecuaciones planteadas se satisfacen plenamente, lo que implica que el problema es linealmente separable y puede ser resuelto por una red perceptron.

4. ARQUITECTURA DE LA RED

La red perceptron a implementar tiene una capa con tres neuronas, una para cada salida. Una neurona para cada uno de los equipos que se desean controlar (motor, surtidora y enchapadora) y tres entradas, correspondientes a cada uno de los sensores del proceso (S₁, S₂ y S₃) [7].

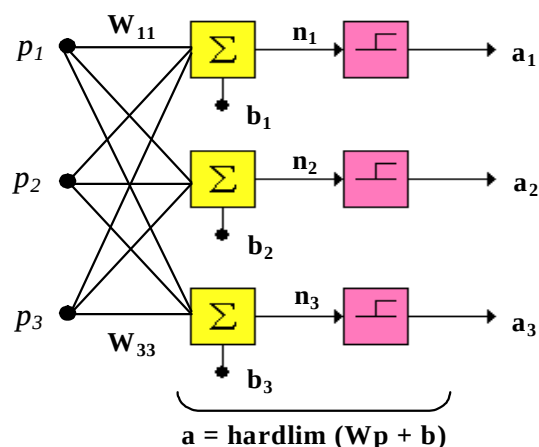


Figura 2. Arquitectura de la red a implementar.

5. ENTRENAMIENTO DE LA RED

A la red se le presentan cuatro patrones de entrada, de un universo total de seis, para entrenarla, y luego del entrenamiento se deben introducir los dos patrones restantes para verificar que la red clasificó correctamente todos los patrones de entrada. A continuación se explica la implementación en lenguaje Matlab. Inicialmente se debe crear la red perceptron, los valores de las tres entradas (sensores) estarán comprendidos en un rango entre 0 y 1; además la red presenta tres salidas (motor, surtidora y enchapadora). Se usa el comando `net = newp([0 1; 0 1; 0 1], 3)`. Los patrones de entrada provenientes de los sensores vienen dados por el vector fila `p`:

`p = [0 0 1 1 1; 0 1 0 0 1; 0 0 0 1 1];`

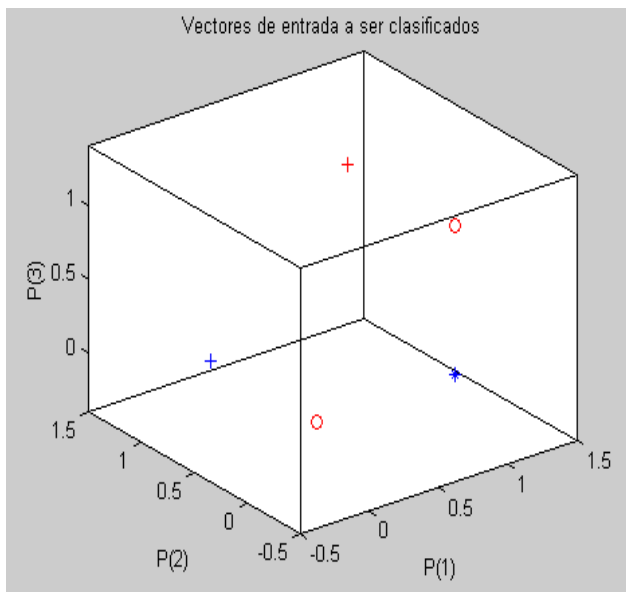


Figura 3. Patrones de entrada al sistema.

Las salidas deseadas (targets) correspondientes a las acciones que los equipos deben tener cuando se aplican las entradas anteriores son las siguientes:

`t = [0 1 0 0 1; 0 0 1 0 0; 1 0 0 1 1];`
`plotpv(p, t); title('Vectores de entrada a ser clasificados');`
`pause;`

Inicializando el vector de pesos y el vector de bias con ceros:

`w1 = net.iw ; wts1 = w1{1, 1}; b1 = net.b; bias1 = b1{1};`
`pause; net.adaptParam.passes = 100;`

Se utiliza la función `Adapt` para realizar la segunda presentación de los patrones de entrada:

`[net, a, e] = adapt(net, p, t); pause;`

Una vez entrenada la red, se procede a verificar que la misma ha sido entrenada correctamente. Para ello, se le introduce un patrón de entrada aleatorio que no se usó para entrenarla y se calcula el error entre el target y la salida correspondiente. Si dicho error es cero, la red suministra la salida deseada al introducir una entrada distinta a la esperada. En la Figura 4 se observa que los vectores han sido clasificados correctamente y que la red produce la salida correcta.

`pFaltante1 = [1; 1; 0]; tFaltante1 = [1; 1; 0];`
`a1 = sim(net, pFaltante1); e1 = tFaltante1 - pFaltante1`
`pause; plotpc(net.iw{1, 1}, net.b{1});`
`title('Vectores de entrada clasificados correctamente y Frontera de decision');`

6. CONCLUSIONES

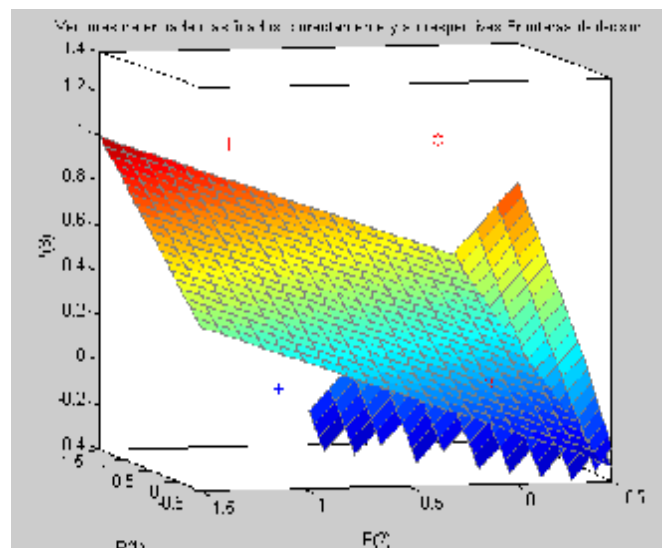


Figura 4. Vectores correctamente clasificados.

Se ha demostrado que se puede implementar un algoritmo de redes neuronales, para automatizar un proceso industrial de embotellado, con Matlab. El algoritmo se desarrolló con una red perceptron, con patrones de entrada a la red clasificados correctamente ya que son linealmente separables. Si el proceso posee entradas y salidas discretas, el algoritmo generalmente se puede desarrollar utilizando una red perceptron. Si las entradas o salidas son de otro tipo, se puede implementar un algoritmo basado en una red Backpropagation o Competitiva o una red híbrida,

combinación de diferentes tipos de redes.

7. REFERENCIAS

- [1] Al-Duwaish, H., Ghouti, L., Halawani, T., and Mohandes, M. (2002), "Use of Artificial Neural Networks Process Analyzers: A Case Study", ESANN'2002 Proceedings European Symposium on Artificial Neural Networks, Bruges (Belgium), pp. 465-470.
- [2] Krichmar, Jeffrey L and Snook, James A., "A Neural Approach to Adaptive Behaviour and Multi-Sensor Action Selection in a Mobile Device" (2002), Proceedings of the 2002 IEEE International Conference on Robotics & Automation, Washington DC, pp. 3864-3869.
- [3] Keller, P. E., Kouzes, R. T., Kangas L. J. and Hashem, S. (1994), "Neural Networks Based Sensor Systems for Manufacturing Applications", Advanced Information Systems and Technology Conference, Williamsburg, VA, USA.
- [4] Martínez César y Alvarez Mario, (2002). "Software didáctico para la enseñanza de las Redes Neuronales Artificiales", Tesis de Grado, Universidad de Carabobo, Bárbula, Venezuela.
- [5] Freeman James A. y Skapura David M. (1993). "Redes Neuronales. Algoritmos, aplicaciones y técnicas de programación". Addison-Wesley, Wilmington, USA.
- [6] Demuth Howard and Beale Mark. (1996). "Neural Network Design", Universidad de Colorado. Colorado, USA.
- [7] Hilera José R. y Martínez Víctor J. (1995). "Redes Neuronales Artificiales. Fundamentos, modelos y aplicaciones", Ra-Ma. USA.