



Revista científica

ISSN: 0124-2253

ISSN: 2344-8350

Universidad Distrital Francisco José de Caldas

Ladino-Moreno, Edgar-Orlando; García-Ubaque, César-Augusto; García-Vaca, María-Camila
Estimación de fugas en tuberías a presión para sistemas de
agua potable mediante redes neuronales artificiales y Epanet
Revista científica, vol. 43, núm. 1, 2022, Enero-Abril, pp. 2-19
Universidad Distrital Francisco José de Caldas

DOI: <https://doi.org/10.14483/23448350.18275>

Disponible en: <https://www.redalyc.org/articulo.oa?id=504371975001>

- Cómo citar el artículo
- Número completo
- Más información del artículo
- Página de la revista en redalyc.org



Sistema de Información Científica Redalyc
Red de Revistas Científicas de América Latina y el Caribe, España y Portugal
Proyecto académico sin fines de lucro, desarrollado bajo la iniciativa de acceso
abierto



Estimación de fugas en tuberías a presión para sistemas de agua potable mediante redes neuronales artificiales y Epanet

Leak Estimation in Pressure Pipes for Drinking Water Systems through Artificial Neuronal Networks and Epanet

Estimación de vazamentos na pressão de tuberías para sistemas de água potável através de redes de neuronales artificiales e Epanet

Edgar-Orlando Ladino-Moreno¹ 

César-Augusto García-Ubaque² 

María-Camila García-Vaca³ 

Recibido: julio de 2021

Aceptado: octubre de 2021

Para citar este artículo: Ladino-Moreno, E. O., García-Ubaque, C. A. y García-Vaca, M. C. (2022). Estimación de fugas en tuberías a presión para sistemas de agua potable mediante redes neuronales artificiales y Epanet. *Revista Científica*, 43(1), 2-19. <https://doi.org/10.14483/23448350.18275>

Resumen

Este trabajo trata de la estimación de una fuga para un sistema de tubería principal sin ramificaciones. Se propone un algoritmo y una red neuronal con cuatro variables de entrada, una capa oculta con 25 neuronas y tres variables de salida. La obtención de los datos se realizó mediante un *bucle* anidado en Visual Basic (Excel®) estableciendo 35.837 escenarios de fuga para una tubería de 30 m que conduce agua con viscosidad cinemática de 0,000001 (m²/s), un diámetro igual a 0,15222 m, rugosidad de 0,0000015 m, pérdida de carga de 3,5 m y dos accesorios (k_1 , k_2) que suma 1,5. Se instalaron en el sistema hidráulico dos caudalímetros y dos manómetros virtuales al inicio y al final de la tubería. Asimismo, se utiliza Epanet® e Hydroflo®

(Tahoe Design Software) para estructurar el modelo hidráulico y validar los datos iniciales. Se utilizó MatLab R2021a para analizar los algoritmos de aprendizaje de retropropagación y regularización bayesiana, adoptando la función de transferencia log sigmoide. Como función de control se implementó el error medio cuadrático y el coeficiente de determinación R^2 . El modelo neuronal obtenido presentó un error medio cuadrático de 1,44E-06 y un error relativo igual a 0,0055 % para los datos de entrenamiento. La validación cruzada de la red neuronal se realizó a partir de 5.973 datos de entrada independientes.

Palabras clave: agua potable; fugas; gestión de los recursos hídricos; Levenberg-Marquardt; red neuronal artificial.

1. Ph. D. (c) Universidad Distrital Francisco José de Caldas. Bogotá, Colombia. eoladinom@correo.udistrital.edu.co.
2. Ph. D. Universidad Distrital Francisco José de Caldas. Bogotá, Colombia, cagarciau@udistrital.edu.co.
3. M. Sc. Universidad Católica de Colombia. Bogotá, Colombia., mgarciav@ucatolica.edu.co.

Abstract

This work deals with the estimation of a leak for a main pipe system without branches. An algorithm and a neural network with 4 input variables are proposed, a hidden layer with 25 neurons and 3 output variables. The data was obtained through a nested loop in Visual Basic (Excel®) establishing 35,837 leak scenarios for a 30 m pipe that conducts water with a kinematic viscosity of 0.000001 (m^2/s), a diameter equal to 0.15222 m, roughness of 0.0000015 m, pressure drop of 3.5 m and two accessories (k_1 , k_2) that add up to 1.5. Two flowmeters and two virtual pressure gauges were installed in the hydraulic system at the beginning and end of the pipeline. Also, Epanet® and Hydroflo® (Tahoe Design Software) are used to structure the hydraulic model and validate the initial data. Matlab R2021a was used to analyze the Backpropagation and Bayesian Regularization learning algorithms adopting the log sigmoid transfer function. The mean square error and the coefficient of determination R^2 were implemented as a control function. The neural model obtained presented a mean square error of 1.44E-06 and a relative error equal to 0.0055% for the training data. The cross-validation of the neural network was carried out from 5,973 independent input data.

Keywords: artificial neural network; drinking water; leakage; Levenberg-Marquardt; water resources management.

Resumo

Este trabalho trata da estimativa de vazamento para um sistema de tubulação principal sem ramificações. Um algoritmo e um vermelho neural com 4 variáveis de entrada são propostos, uma camada oculta com 25 neurônios e 3 variáveis de saída. A coleta de dados é realizada por meio de um loop aninhado no Visual Basic (Excel®) estabelecendo 35.837 cenários de fuga para uma tubulação de 30 m que transporta água com viscosidade cinemática de 0,000001 (m^2/s), diâmetro igual a 0,15222 m, rugosidade de 0,0000015 m, perda de carga de 3,5 m dos acessórios (k_1 , k_2) que soma 1,5. É instalado no sistema hidráulico dos medidores de vazão e manômetros virtuais no início e no final da tubulação. Asimismo, Epanet® e Hydroflo® (Tahoe Design Software) são usados para estruturar o modelo hidráulico e validar

os dados iniciais. Use o Matlab R2021a para analisar os algoritmos de aprendizagem de retropropagação e regularização bayesiana adotando a função de transferência log sigmóide. A função de controle implementa o erro quadrático médio e o coeficiente de determinação R^2 . O modelo neuronal obtido apresenta um erro quadrático médio de 1,44E-06 e um erro relativo igual a 0,0055% para os dados de entrada. A validação cruzada do vermelho neuronal foi realizada a partir de 5.973 dados de entrada independentes.

Palavras-chaves: água potável; gestão de recursos hídricos; Levenberg-Marquardt; rede neural artificial; vazamento.

Introducción

Las detecciones de fugas en tuberías pueden ser abordadas desde dos enfoques: uno basado en hardware (monitoreo) y otro centrado en el software (Lu et al., 2020). La complejidad de los sistemas hidráulicos y especialmente en lo referente a la localización y cuantificación de las fugas en sistemas de acueducto radica en la no linealidad de las ecuaciones diferenciales que gobiernan el flujo a presión. La localización de fugas, la estimación de presiones y los caudales generados por la pérdida de agua constituyen un problema Np-Duro, es decir, no existe un método determinístico que resuelva el problema en un tiempo polinomial establecido y con un costo computacional aceptable. Así, las ecuaciones de flujo a resolver corresponden a la ecuación de conservación de masa en el nodo donde se presenta la fuga y las ecuaciones de conservación de energía. La determinación de la cantidad de masa y la posición de una fuga para un sistema hidráulico simple en el cual la sección transversal de la tubería permanece constante dependen directamente del comportamiento del gradiente hidráulico (Figura 1). Al generarse la fuga se evidencia un cambio de velocidad dentro del sistema, por lo cual se establece una mínima variación en el coeficiente de fricción antes y después de la fuga. Si se considera la existencia de varias fugas en el sistema hidráulico conformado

por diferentes mallas o *bucles* se estructura un problema hidráulico de alta complejidad, debido a la sensibilidad de las variables de entrada. Este problema puede ser abordado desde el campo de la inteligencia artificial a partir de algoritmos híbridos de clasificación, cuyo objetivo es reducir el área donde se generan las fugas en el sector hidráulico. La arquitectura de red neuronal parte de las variables de entrada como son las presiones y los caudales de entrada y salida. Estos valores son afectados por los pesos sinópticos y pasan a través de la función de activación, este proceso se repite en forma de bucle hasta alcanzar la capacidad de generalización de la estructura neuronal. El desarrollo de redes neuronales implementados en sistemas hidráulicos ha demostrado alta eficiencia en términos de variables de salida (Burton *et al.*, 1999). Si bien es cierto que existen otros tipos de metaheurísticas para la detección de fugas como máquinas de soporte vectorial, algoritmos genéticos, lógica difusa y esquemas meméticos, las redes neuronales, debido a su capacidad de aprendizaje, presentan resultados óptimos con respecto a la cuantificación y la localización de fugas en los sistemas de distribución de agua potable y transporte de hidrocarburos, entre otros, por lo cual se puede afirmar que las redes neuronales han demostrado resultados robustos para problemas de alta complejidad (Caputo y Pelagagge, 2003).

Metodología

Hidráulica de la fuga

Para un sistema hidráulico con una única fuga es posible determinar la cantidad de masa y la localización de la fuga mediante la igualación de la pérdida de carga antes y después de la fuga. Al aceptar la ecuación que gobierna el comportamiento del flujo a presión dada por (1) se establece un sistema subdeterminado debido a la existencia de más incógnitas que ecuaciones (9). Esto puede ser resuelto a través de un *bucle* anidado para el

cálculo de los coeficientes de fricción y la posición lineal de la fuga. Para un sistema presurizado la pérdida por fricción está dada por la ecuación de Darcy-Weisbach.

$$h_f = f \frac{L}{D} \frac{V^2}{2g} \quad (1)$$

Donde, h_f : pérdida de carga (m); f : coeficiente de fricción; L : longitud tubería (m); D : diámetro (m); V : velocidad media del flujo (m/s); g : gravedad (m/s²). Para las pérdidas menores o por accesorios, se tiene,

$$h_i = \sum k \frac{V^2}{2g} \quad (2)$$

Donde, h_i : pérdidas menores (m); k : coeficiente de pérdidas menores (adimensional); V : velocidad media del fluido (m/s); g : aceleración de la gravedad (m/s²). Para el cálculo del coeficiente de fricción (f) se implementó la ecuación propuesta por Colebrook-White (3).

$$\frac{1}{\sqrt{f}} + 2 \log \left[\frac{\varepsilon/D}{3.7} + \frac{2.51}{R_e \sqrt{f}} \right] = 0 \quad (3)$$

Donde, f : coeficiente de fricción; ε : rugosidad del tubo (m), D : diámetro (m); R_e : número de Reynolds. La presión para el punto de la fuga (P_f) está dada por la diferencia entre la presión inicial (P_i) y las pérdidas por fricción antes de la fuga (h_{f_i}) (Figura 2). Aceptando la ecuación de Darcy-Weisbach (1) se obtiene la pérdida de carga en el tramo antes de la fuga, donde el coeficiente de fricción se determina de forma iterativa implementado el modelo propuesto por Colebrook-White (3). La Figura 1 presenta el esquema hidráulico para una tubería principal sin derivaciones en la cual no se evidencia la presencia de fugas en el sistema. La diferencia de nivel entre los tanques establece la pérdida de carga total del sistema (H). El modelo hidráulico presenta dos accesorios (k_1 , k_2) a la entrada y a la salida

del sistema respectivamente. De igual forma, el diámetro permanece constante a lo largo de la longitud de la tubería. Conociendo el diámetro, la pérdida de carga, la rugosidad, la longitud del tubo, la viscosidad y la sumatoria producida por los accesorios es posible determinar el caudal a partir de la ecuación (4). Esta ecuación resuelve la mayoría de problemas de tuberías simples, debido a que la única incógnita corresponde a la pérdida por fricción (h_f).

$$Z_1 - Z_2 - h_f - \frac{\Sigma k}{2g} \left[-2 \sqrt{\frac{2gDh_f}{L}} \log \left[\frac{\epsilon/D}{3.7} + \frac{2.51\nu}{D \sqrt{\frac{2gDh_f}{L}}} \right] \right]^2 = 0 \quad (4)$$

La solución para h_f se puede obtener a partir de la herramienta análisis de hipótesis (Excel®), donde la función objetivo corresponde a la ecuación (4). Esta ecuación resuelve un sistema hidráulico para tuberías conectadas por una sola tubería con un solo diámetro y diferentes accesorios, estableciendo un depósito al inicio y al final del sistema. El comportamiento de la línea de gradiente hidráulico para un sistema de tuberías simple está dado por la [Figura 1](#).

La suma de la elevación con respecto a un datum y la altura de presión corresponden a la línea de gradiente hidráulico (LGH), esta línea indica la presión en términos de columna de agua a lo largo de la tubería. Por encima de la LGH y paralela a esta se establece la línea de energía (LE) separada por la altura de velocidad ($V^2/2g$).

Una vez se genera la fuga ([Figura 2](#)), se origina un cambio significativo del gradiente hidráulico (J) ([Gupta, Kishore y Jain, 2017](#)). Las presiones (P_1, P_2) cambian con respecto a las presiones determinadas por el sistema sin fuga. Rojas, Verde y Torres (2021) proponen un polinomio de segundo orden para modelar el gradiente hidráulico ($J(Q)$) a partir de diferentes rangos de velocidades. En consecuencia, la ley de potencia N1, una versión aproximada del concepto FAVAD (*Fixed And Variable Area Discharges*) se ha utilizado desde 1994 para la evaluación de fugas en concordancia con el concepto simplificado de Factor Noche-Día (NDF) ([Lambert, Fantozzi y Shepherd, 2017](#)). Al presentarse un cambio de velocidades en el sistema debido a la aparición de la fuga se establecen tres incógnitas a determinar: el coeficiente de fricción antes de la fuga (f_1); la localización longitudinal de la fuga (L_f) y el coeficiente de fricción después de la fuga (f_2), resuelto esto es posible cuantificar la cantidad de masa y la presión en el punto de la fuga. Es decir, teniendo en cuenta los vectores de entrada netos (Q_1, Q_2, P_1, P_2) es viable obtener una solución para la ubicación de la fuga, el caudal y la presión en ese punto (Q_f, P_f, L_f) a partir de la ecuación (11).

La ecuación parametrizada está dada por (12), estableciendo tres incógnitas (f_1, f_2, L_f), las cuales pueden ser encontradas a partir de las condiciones hidráulicas de entrada y la ejecución iterativa de un *bucle* anidado.

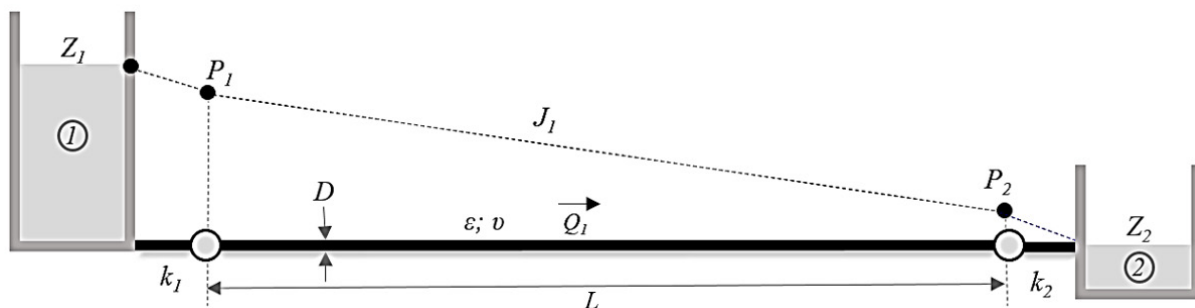


Figura 1. Esquema hidráulico sin fuga

Fuente: elaboración propia

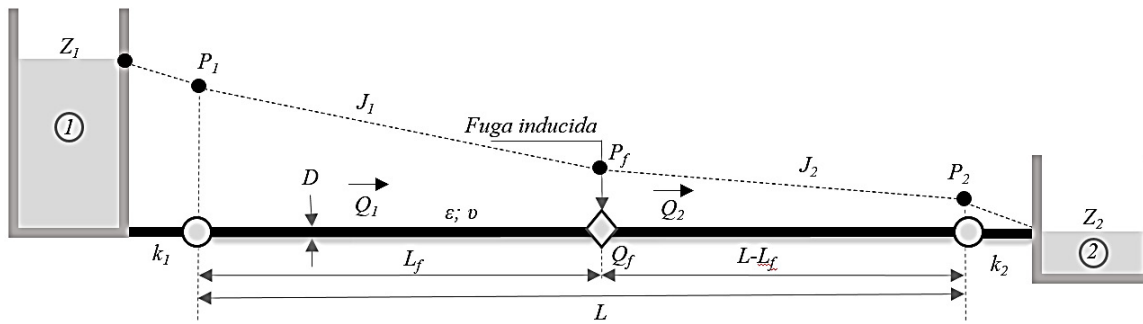


Figura 2. Esquema hidráulico para una fuga
Fuente: elaboración propia

Al igualar la presión antes (P_1) y después de la fuga (P_2), se tiene:

$$P_f = P_1 - hf_1 \quad (5)$$

$$P_f = P_1 - f_1 \frac{L_f V_1^2}{D 2g} \quad (6)$$

$$P_f = P_1 - f_1 \frac{L_f Q_1^2}{D 2gA^2} \quad (7)$$

$$P_f = P_2 + f_2 \frac{(L - L_f) Q_2^2}{D 2gA^2} \quad (8)$$

$$P_1 - f_1 \frac{L_f Q_1^2}{D 2gA^2} - P_2 - f_2 \frac{(L - L_f) Q_2^2}{D 2gA^2} = 0 \quad (9)$$

$$P_1 - P_2 - \frac{[f_1 L_f Q_1^2 + f_2 (L - L_f) Q_2^2]}{2DgA^2} = 0 \quad (10)$$

$$\theta_1 - f_1 L_f \theta_2 - f_2 \theta_3 + L_f \theta_4 = 0 \quad (11)$$

Para establecer la capacidad del algoritmo (11), se propone una tubería de 30 m que conduce agua con una viscosidad cinemática de 0,000001 (m^2/s), un diámetro igual a 0,15222 m y con una rugosidad de 0,0000015 m. Se instalaron en el sistema hidráulico dos caudalímetros y dos manómetros virtuales al inicio y al final de la tubería, las lecturas obtenidas aparecen en la [Tabla I](#).

Tabla I. Input: Datos de entrada

P_1 (mca)	Q_1 (m^3/s)	P_2 (mca)	Q_2 (m^3/s)
3,02082497	0,07891233	0,81815652	0,07291

Fuente: elaboración propia

La [Tabla II](#) muestra el proceso iterativo realizado, se estableció la función objetivo (FO $_f$) para el coeficiente de fricción antes de la fuga a partir de Colebrook-White (3). De igual forma, se obtuvo el coeficiente de fricción después de la fuga (f_2). Finalmente, la función objetivo está dada por (FO $_L$) donde el objetivo es minimizar la ecuación (11) modificando la localización de la fuga a lo largo de la tubería partiendo del tanque 1 (L_f). Es decir, encontrar el valor de L_f que minimice (11).

Implementado Visual Basic (Excel®) se realiza un *bucle* anidado para la función objetivo (11).

```

Sub Fuga()
For i = 1 To 6
Range("G12").Select
Do Until ActiveCell = ""
ActiveCell.Offset(0, 1).GoalSeek Goal:=0,
ChangingCell:=ActiveCell
ActiveCell.Offset(1, 0).Range("A1").Select
Loop
Range("K12").Select
Do Until ActiveCell = ""
ActiveCell.Offset(0, 1).GoalSeek Goal:=0,
ChangingCell:=ActiveCell
ActiveCell.Offset(1, 0).Range("A1").Select
Loop
Range("M12").Select
Do Until ActiveCell = ""
ActiveCell.Offset(0, 1).GoalSeek Goal:=0,
ChangingCell:=ActiveCell
ActiveCell.Offset(1, 0).Range("A1").Select
Loop
Next i
End Sub
    
```


Tabla II. Iteración cálculo para una fuga (Estado estacionario)

V_1 (m/s)	Re_1	f_1	FO_{f_1}	V_2 (m/s)	Re_2	f_2	FO_{f_2}	L_f (m)	FO_{L_f}
4,3362	660060	0,012689	0	4,0065	609873	0,012858	0	12	0

Fuente: elaboración propia

La solución encontrada a partir de las condiciones de frontera preestablecidas y la función objetivo (11), indican que el sistema presentará una fuga a 12 m medidos desde el tanque 1 (Figura 2). La fuga presenta un caudal de 6 lps y una presión manométrica en ese punto igual a 2,06 mca.

Tabla III. Output: Solución para una fuga tubería 30 m

Q_f (m ³ /s)	P_f (mca)	L_f (m)
0,006	2,06	12

Fuente: elaboración propia

La validación de estos resultados se realizó a través del software Epanet® desarrollado por la Agencia de Protección Ambiental de Estados Unidos, este software es utilizado para modelar y diseñar sistemas a presión, de igual forma, es posible realizar el seguimiento de sustancias en la red, también es viable para la detección de fugas a partir de emisores (Mashford et al., 2009). Epanet® resuelve diferentes ecuaciones no lineales simultáneamente utilizando métodos iterativos con el objetivo de alcanzar el equilibrio de red (Jasper et al., 2013). Si la red no presenta fuga el valor del coeficiente del emisor será igual a cero. El caudal generado por el emisor es igual al producto del coeficiente del emisor y la presión en el nodo elevada a una potencia. La potencia recomendada es 0,5, que normalmente se aplica a rociadores y boquillas.

$$q = \alpha P^\beta \quad (12)$$

Donde, q : caudal en la fuga (m³/s); P : presión en el nodo (mca); α : coeficiente del emisor; β : potencia. En el caso de las condiciones establecidas

en la Tabla I, el caudal generado por la fuga está dado por:

$$q = 0,0041804 * 2,06^{0,5} \\ = 0,006 \text{ m}^3/\text{s}$$

Bajo esta consideración el coeficiente del emisor utilizado en Epanet® corresponde a 4,18, el cual garantiza un caudal en la fuga de 6 lps. La simulación hidráulica se realizó bajo condiciones estacionarias del flujo con una pérdida de carga de 3,5 m, longitud de la tubería igual a 30 m, un accesorio a la entrada con k_1 igual a 0,5 y uno a la salida con k_2 igual a 1.

De igual forma, a partir del software Hydroflo® (Tahoe Design Software) se modeló el tramo de tubería colocando tres manómetros, uno a la entrada (M1) y a la salida (M2) de la tubería, y otro en la ubicación de la fuga inducida (M3), los resultados obtenidos se presentan en la Figura 4.

La Tabla IV muestra la comparación entre las cuatro metodologías con base en los datos de entrada (Tabla I). De igual forma, se validaron 45 sistemas hidráulicos independientes bajo las mismas condiciones hidráulicas de entrada (pérdida de carga, diámetro, rugosidad y longitud del tubo), modificando la posición (L_f) y el caudal de la fuga (Q_f). Los resultados obtenidos con la ecuación (11) presentan una aproximación a la décima de milímetro con respecto a la posición virtual de la fuga obtenida en los modelos simulados y una aproximación a la décima de litro con respecto al caudal generado por la fuga. Una vez validado el algoritmo se construyó el conjunto de datos de entrenamiento para la RNA (red neuronal artificial).

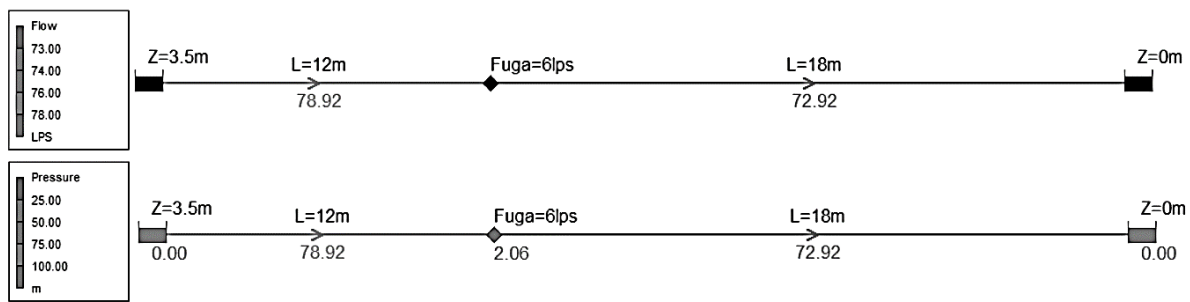


Figura 3. Modelo Epanet® (tubería 30 m)

Fuente: elaboración propia

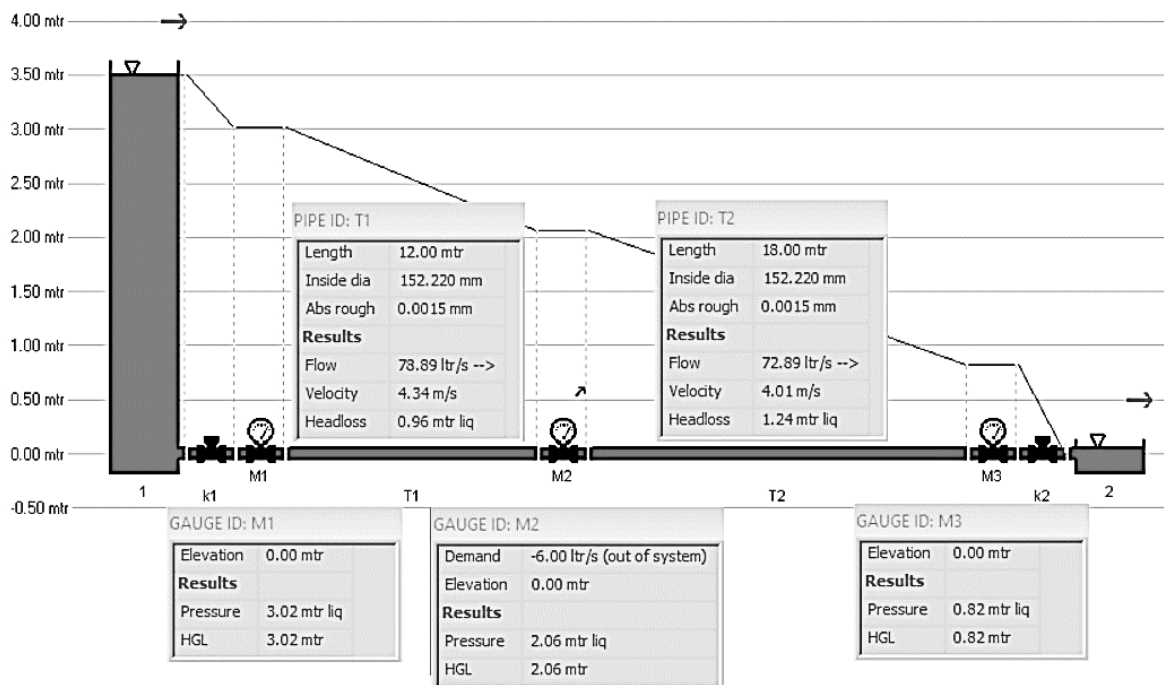


Figura 4. Modelo hidráulico Hydroflo® (Tubería 30 m)

Fuente: Hydroflo® (Tahoe Design Software)

Tabla IV. Comparación soluciones (Tubería 30 m)

Metodología	Q_i (m ³ /s)	P_i (mca)	L_i (m)
Algoritmo (11)	0,006	2,0621	12
RNA	0,00601	2,0622	11,9998
Hydroflo®	0,006	2,06	12
Epanet®	0,006	2,06	12

Fuente: elaboración propia

Procesamiento de datos

Los datos se establecieron a partir de la función objetivo dada por (11) con base en un algoritmo anidado realizado en Visual Basic (Excel®), para lo cual se determinó el caudal generado por la fuga (Q_f), la presión manométrica en la fuga (P_f) y la posición longitudinal (L_f) para 35.837 escenarios diferentes. Estos datos de entrada se generaron de manera aleatoria y se ajustan a la distribución normal para el caudal al inicial del tramo (Q_1), presión manométrica inicial (P_1), caudal al final del tramo (Q_2), presión manométrica al final de la tubería (P_2). Los datos están disponibles en el siguiente enlace ([descarga datos](#)). El Dominio de las variables de entrada se muestran en la [Tabla V](#).

Tabla V. Rangos para variables de entrada (*inputs*)

	Caudal (m ³ /s)	Longitud (m)
Mínimo	0,0005	0,01
Máximo	0,05	30

Fuente: elaboración propia

Arquitectura de la red neuronal artificial (RNA)

Para definir la estructura de una red adecuada es necesario determinar el número de neuronas ocultas. De hecho, para describir relaciones complejas no lineales, aumentar la cantidad de neuronas y capas ocultas puede mejorar la capacidad de la RNA ([Fan, Zhang y Yu, 2021](#)). Esta cantidad afecta de manera significativa el potencial de rendimiento de la red ([Bishop, 2005](#)). La [Tabla VI](#) presenta seis arquitecturas diferentes para abordar el cálculo de la fuga y la localización en una tubería principal sin ramificaciones en términos de un modelo estacionario. Es posible realizar simulaciones sintéticas transitorias a partir del cierre de una válvula utilizando una simulación numérica ([Bohórquez et al., 2020](#)). [Sabu et al. \(2021\)](#)

proponen un modelo inteligente para el mapeo de varios nodos de fuga en función de los valores de presión con una precisión significativa. Para este estudio el esquema que presentó mejores resultados con base en el MSE y el estadístico R correspondió a la arquitectura con cuatro variables de entrada, una capa oculta con 25 neuronas y tres variables de salida [4-25-3], esto bajo tiempos computacionales aceptables.

La [Tabla VI](#) muestra cómo el número de neuronas y la cantidad de capas ocultas inciden sobre el MSE y el R^2 , además la precisión del modelo determinado depende del número de neuronas en el nivel topológico ([Rojek y Studzinski, 2019](#)). No obstante, este estudio evidenció que al aumentar el número de neuronas y de capas ocultas el MSE se estabiliza en valores cercanos a 1,03E-06. Esto indica que al aumentar el número de neuronas o de capas ocultas el estadístico MSE no va a mejorar y se comporta de forma asintótica con respecto al número de épocas ([Figura 7a](#)). De hecho, los tiempos computacionales aumentan considerablemente. Por ejemplo, para alcanzar convergencia en la arquitectura [4-30-30-3] se requieren 3 horas y 32 minutos de procesamiento computacional.

La [Tabla VI](#) presenta diferentes arquitecturas neuronales para la optimización de la ubicación de la fuga.

Entrenamiento de la red neuronal

El desarrollo de la RNA requiere tres etapas bien definidas: la arquitectura de la red, las funciones de activación y el algoritmo de aprendizaje ([Simpson, 1991](#)). En el proceso de entrenamiento se considera aleatorios a los pesos iniciales, una vez se ha producido la primera época, estos valores cambian y se ajustan de forma continua. Para este estudio los pesos se optimizaron con el algoritmo de regularización bayesiana teniendo en cuenta el MSE y el coeficiente de determinación R^2 , el cual explica la variación de las variables respuesta (Q_f , P_f , L_f) explicadas por las variables independientes (Q_1 , Q_2 , P_1 , P_2) ([Tabla VII](#)). Se utilizó

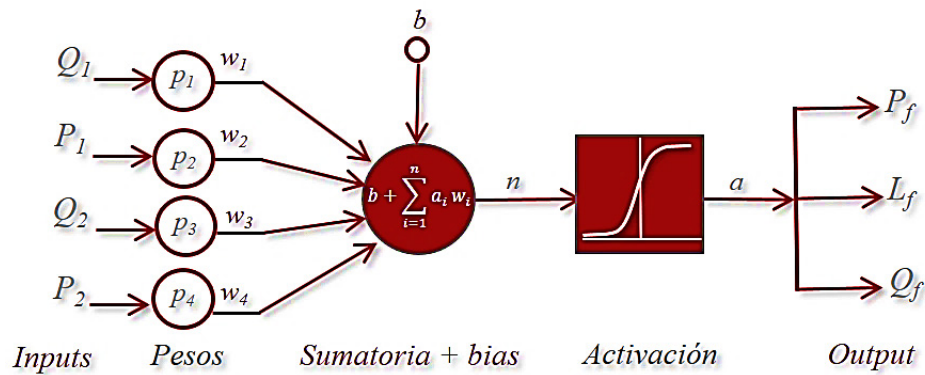


Figura 5. Esquema arquitectura neuronal [4-25-3]

Fuente: elaboración propia

Tabla VI. Arquitecturas realizadas para el cálculo del coeficiente de fricción

Nº capas	Nº de neuronas	Arquitectura	R ²	MAE	MSE	SSE	SAE	BCE
1	2	4-2-3	9,74E-01	5,64E-01	1,74E+00	1,87E+05	6,06E+04	3,43E+00
1	5	4-5-3	9,98E+00	1,58E-01	9,57E-02	1,02E+04	1,70E+04	1,13E+04
2	5	4-5-5-3	9,98E+00	1,07E-02	3,26E-04	35,1E+00	1,15E+03	1,27E+03
1	25	4-25-3	1,00E+00	1,88E-04	1,44E-06	1,55E-01	20,27E+0	89,06E+0
1	30	4-30-3	1,00E+00	1,97E-04	1,50E-06	1,62E-01	21,2E+00	56,2E+00
3	25	4-30-30-3	9,99E+00	1,41E-04	1,03E-06	1,11E-01	15,2E+00	1,50E+00

Fuente: elaboración propia

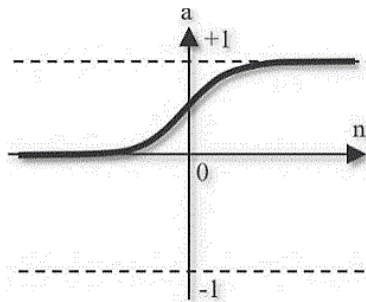


Figura 6. Función de transferencia log sigmoide

Fuente: elaboración propia

MatLab (2021a) para analizar los algoritmos de aprendizaje de retropropagación y regularización bayesiana. La RNA se entrena dividiendo en tres conjuntos los datos de entrada a partir de índices aleatorios, esto para validar la predicción de la posición, la presión y el caudal en la fuga. De

igual forma, la función de transferencia log sigmoide “logsig” presentó los mejores resultados y los mínimos estadísticos, esto se evidencia en la [Tabla VIII](#), donde la función de transferencia calcula la salida de la capa oculta a partir del vector de entradas netas.

La función de activación log sigmoide “logsig” está definida por la siguiente expresión,

$$a = \frac{1}{1 + e^{-n}} \quad (13)$$

$$n = wp + b \quad (14)$$

Donde n está definido como el vector que contiene los pesos sinápticos (w) multiplicados por los patrones (p) de entrada más un valor del sesgo de la red (b); a corresponde a las señales de salida (Q_p, P_p, L_p).

La función de transferencia log sigmoide “logsig” se implementó para la red multicapa entrenada a partir del algoritmo de retropropagación, esto debido a que esta función es diferenciable.

La [Figura 7a](#) presenta el rendimiento de la red graficando el MSE con respecto a la época para la etapa de entrenamiento, la validación y los

rendimientos de prueba de los datos de entrenamiento. Durante una época, la función de pérdida se calcula para todos los datos estableciendo el valor de la pérdida cuantitativa en la época determinada. Lo recomendable para estas tres funciones es que adopten una tendencia similar de forma descendente. Dado el caso en el cual las gráficas de las funciones comiencen a alejarse una de otra, el modelo estará sobre-entrenado, afectando su capacidad de generalización.

Al aumentar el número de épocas la función de control se reduce. Sin embargo, para este estudio el MSE comenzó a comportarse de forma asintótica después de las 6.000 épocas, tomando el error de validación el MSE más bajo. La Figura 7b presenta la variación del coeficiente del gradiente con respecto al número de épocas. El valor final del coeficiente del gradiente para la época 6.000 es 3,179E-05, que es aproximadamente cercano a cero.

Tabla VII. Comparación funciones de entrenamiento [4-25-3]

Función de entrenamiento	R ²	MAE	MSE	SSE	SAE	BCE
Bayesian Regularization	1,00E+00	1,88E-04	1,44E-06	1,55E-01	20,27E+0	89,06E+0
Levenberg-Marquardt	9,99E-01	2,96E-04	1,62E-06	1,74E-01	31,9E+00	81,26E+00
Gradient Descent	5,56E-01	3,03E-01	35,6E+00	3,82E+00	1,97E+03	1,26E+02
BFGS Quasi-Newton (Foresee y Hagan, 1997)	9,98E-01	1,83E-02	1,11E-03	1,19E+02	3,25E+00	32,7E+00
Gradient Descent with Momentum	1,55E-01	4,58E+00	73,8E+00	7,94E+06	4,93E+05	67,1E+00
Scaled Conjugate Gradient	9,94E-01	2,07E-01	3,52E-01	3,79E+04	2,22E+04	1,19E+00
Variable Learning Rate Gradient Descent	9,54E-01	8,44E-01	3,19E+00	3,43E+05	9,07E+04	3,54E+00
One Step Secant	9,86E-01	3,65E-01	9,76E-01	1,05E+05	3,93E+04	1,57E+00
Polak-Ribière Conjugate Gradient	9,93E-01	2,27E-01	4,52E-01	4,86E+04	2,44E+04	1,15E+00

Fuente: elaboración propia

Tabla VIII. Comparación funciones de transferencia [4-25-3]

Función de transferencia	R ²	MAE	MSE	SSE	SAE	BCE
Log sigmoide (logsig)	1,00E+00	1,88E-04	1,44E-06	1,55E-01	20,27E+0	89,06E+0
Tangente sigmoidea (tansig)	9,98E-01	1,93E-01	1,41E-01	1,52E+04	20,7E+04	27,93+00

Fuente: elaboración propia

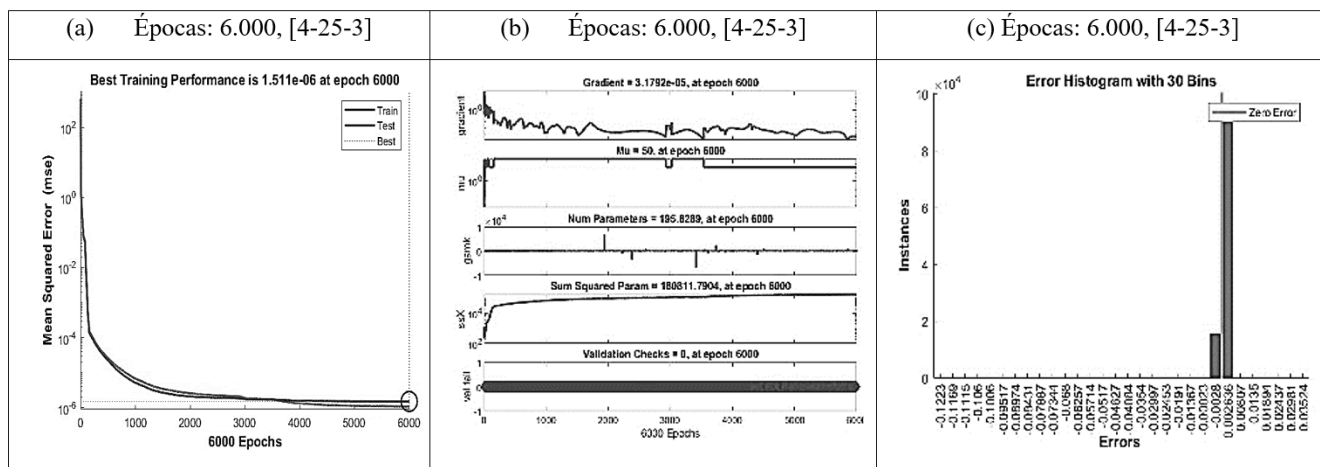


Figura 7. Estado del entrenamiento [4-25-3]

Fuente: elaboración propia

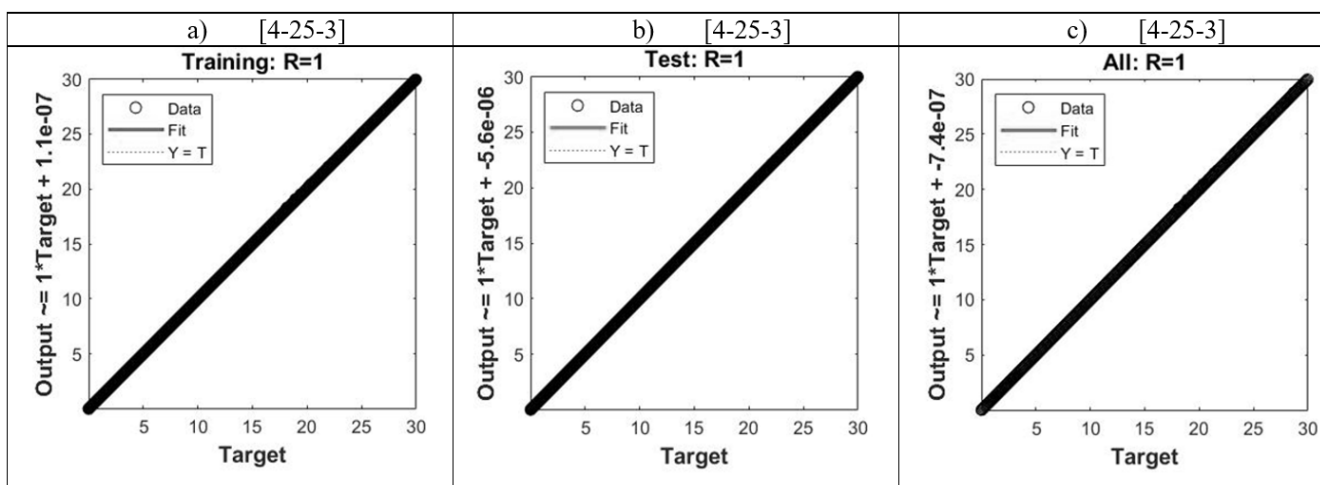


Figura 8. Regresión de entrenamiento [4-25-3]

Fuente: elaboración propia

Debido a que a MatLab R2021a divide el conjunto de datos de forma aleatoria, se origina la [Figura 8](#), esta indica la relación lineal que existe entre los valores objetivo de los datos de entrenamiento y los valores estimados por la red neuronal. Si la entrada es igual a la respuesta se formará una recta a 45° con respecto a la horizontal, indicando que el valor del coeficiente de determinación es igual a 1. La [Figura 8a](#) establece la correlación lineal para los datos de entrenamiento, mientras que la [Figura 8b](#) indica la correlación lineal para el conjunto de datos de prueba

seleccionados aleatoriamente por MatLab R2021a a partir de la función "dividerand" (división del conjunto de datos objetivo en tres conjuntos utilizando índices aleatorios).

Resultados

Este estudio implementó el método de propagación hacia atrás "backpropagation", en el cual se establece un patrón en la entrada de la red afectado por los pesos sinópticos y el sesgo, estos valores son transferidos a la función log sigmoide

originando las señales de respuesta. Al comparar estas señales de salida con los valores de entrenamiento se genera una matriz de errores en el que el objetivo del método es minimizar la magnitud de la superficie de error (Figura 9).

La [Tabla IX](#) presenta de forma explícita el cálculo de la presión, el caudal y la posición de la fuga para ocho escenarios diferentes, comparando los resultados obtenidos con el algoritmo (11) y los valores estimados por la RNA [4-25-3].

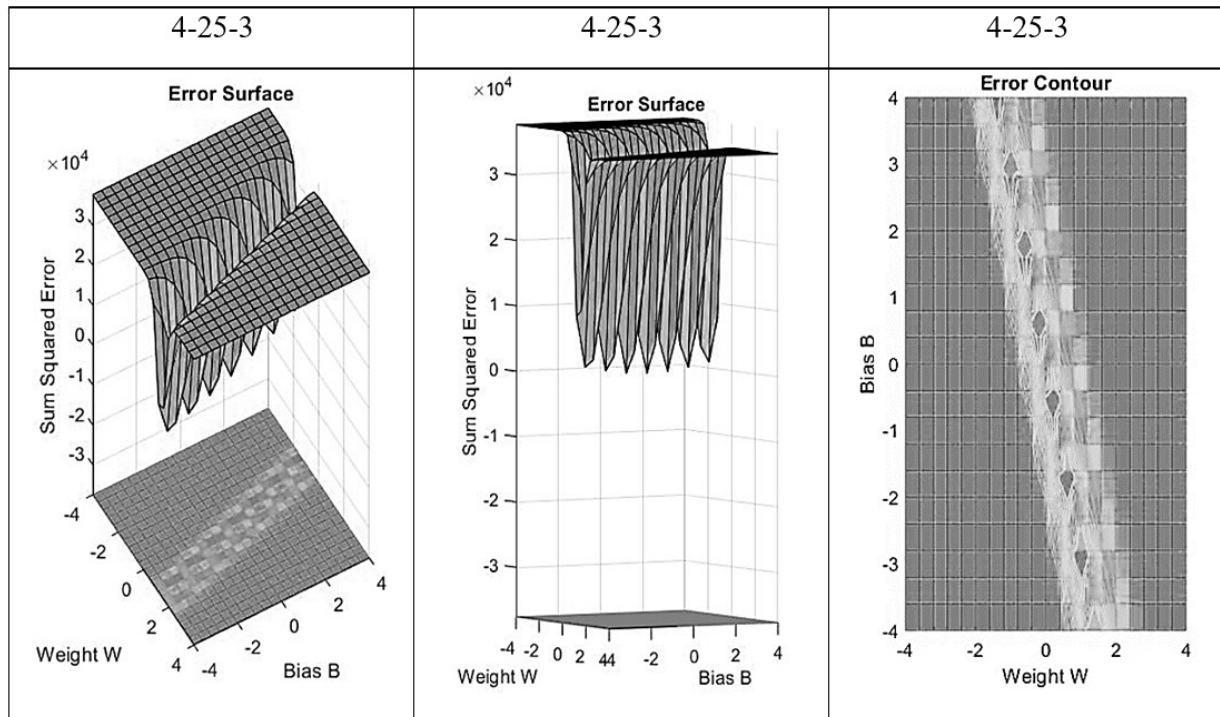


Figura 9. Superficie de error [4-25-3]

Fuente: elaboración propia

Tabla IX. Validación red neuronal [4-25-3]

Input				Output (Algoritmo) (11)			Output (RNA) [4-25-3]		
Q_1 (m ³ /s)	Q_2 (m ³ /s)	P_1 (mca)	P_2 (mca)	P_f (mca)	L_f (m)	Q_f (m ³ /s)	P_f (mca)	L_f (m)	Q_f (m ³ /s)
0,07709	0,07459	3,043	0,856	2,622	5,500	0,00250	2,621	5,503	0,00242
0,07732	0,07192	3,040	0,796	1,254	23,200	0,00540	1,255	23,197	0,00549
0,07992	0,06972	3,008	0,748	1,426	19,350	0,01020	1,427	19,348	0,00975
0,07805	0,07455	3,031	0,855	2,832	2,550	0,00350	2,832	2,548	0,00361
0,08776	0,06926	2,907	0,738	2,106	8,250	0,01850	2,106	8,250	0,01864
0,07898	0,07448	3,020	0,854	2,920	1,250	0,00450	2,920	1,249	0,00468
0,07632	0,07232	3,052	0,805	0,873	29,000	0,00400	0,874	29,002	0,00449
0,07659	0,07389	3,049	0,840	1,725	17,500	0,00270	1,7249	17,497	0,00269

Fuente: elaboración propia

La capacidad de estimación de una RNA se mide a partir del potencial de generalización. Es decir, la capacidad que tiene la red para estimar el vector respuesta a partir de un conjunto de datos nuevos, datos que la red no conocía y con los cuales no ha sido entrenada. Para este estudio se estableció un conjunto de 5.973 datos de prueba con el objetivo de realizar el proceso de validación cruzada. La [Figura 10](#) ilustra el comportamiento estadístico de la serie de entrenamiento. Como función de control se implementó el MSE obteniendo un valor de 0,995. Los estadísticos obtenidos a partir del conjunto de datos de prueba se presentan en la [Tabla X](#) demostrando la capacidad de generalización de la RNA [4-25-3].

El histograma de error mostrado en la [Figura 10](#) indica la forma como se distribuyen los tamaños de error. Por lo cual, la mayoría de los errores para los datos de prueba independientes están cerca de -0,9376.

Conclusiones

El objetivo de este trabajo fue comprobar la capacidad de las redes neuronales para predecir el comportamiento de una fuga en una tubería principal. Los resultados obtenidos demuestran el potencial de la inteligencia artificial aplicado a la detección de fugas. Este estudio demostró que es posible localizar, cuantificar y establecer el nivel de presión de una posible fuga para cualquier sistema de tubería principal sin ramificaciones, a partir de la consolidación de un conjunto de datos de entrenamiento calculados con base en el algoritmo (11). La estructura neuronal corresponde a un vector de entrada con cuatro variables, una capa oculta con 25 neuronas y tres variables de salida [4-25-3] ([Figura 5](#)).

Se evidenció que al aumentar el número de neuronas y de capas ocultas el MSE se estabiliza en valores cercanos a $1,03\text{E}-06$. Esto indica que al incrementar el número de neuronas o de capas

Tabla X. Validación cruzada

Datos	N° datos	Arquitectura	R ²	MAE	MSE	SSE	SAE	BCE
Entrenamiento	35,837	4-25-3	1,00E+00	1,88E-04	1,44E-06	1,55E-01	20,27E+0	89,1E+00
Independientes	5,973	4-25-3	9,95E+00	9,29E-03	3,14E-01	5,64E+03	1,66E+02	88,2E+00

Fuente: elaboración propia

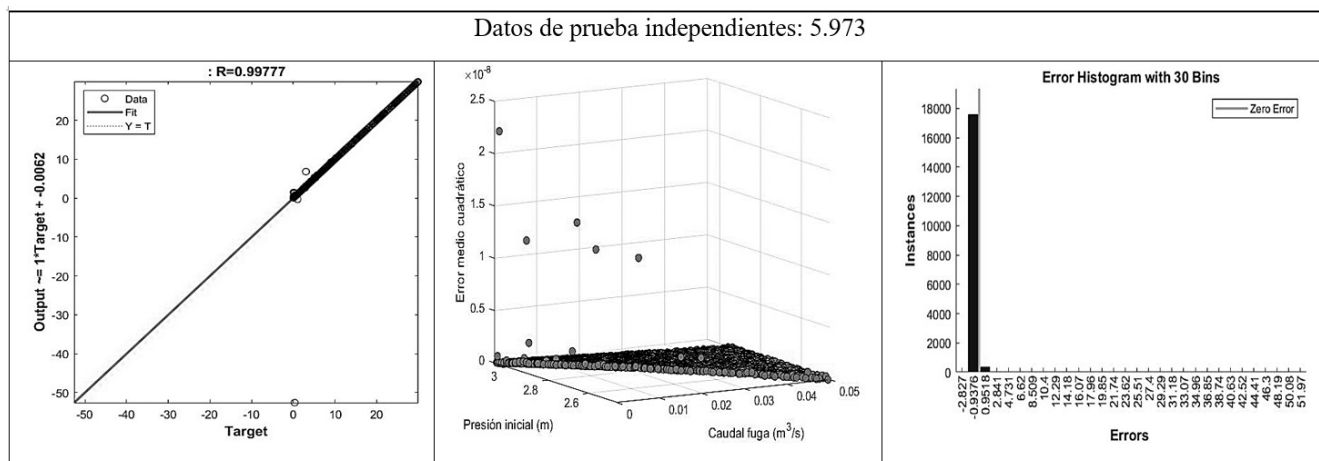


Figura 10. Validación cruzada [4-25-3]

Fuente: elaboración propia

ocultas el estadístico MSE no va a mejorar y se comporta de forma asintótica con respecto al número de épocas. La capacidad de generalización de la red neuronal se demuestra a partir de los estadísticos obtenidos en el proceso de validación cruzada de los 5.973 datos de prueba independientes. Finalmente, la red neuronal tiene la capacidad de localizar la fuga a partir del caudal inicial, el caudal final, la presión inicial y la presión final obtenidos por los sensores virtuales ubicados en el sistema hidráulico.

Referencias

- Bishop, C. (2005). *Neural Networks for Pattern Recognition*. Oxford University Press.
- Bohórquez, J., Alexander, B., Simpson, A., Lambert, M. (2020). Leak detection and topology identification in pipelines using fluid transients and artificial neural networks. *Journal of Water Resources Planning and Management*, 146(6), e0001187. [https://doi.org/10.1061/\(ASCE\)WR.1943-5452.0001187](https://doi.org/10.1061/(ASCE)WR.1943-5452.0001187)
- Burton R. T., Ukrainetz, P. R., Nikiforuk, P. N., Schoenau, G. J. (1999). Neural networks and hydraulic control—from simple to complex applications. En *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*. <https://doi.org/10.1243/0959651991540205>
- Caputo, A., Pelagagge, P. (2003). Using neural networks to monitor piping systems. *Process Safety Progress*, 22(2), 119-127. <https://doi.org/10.1002/prs.680220208>
- Fan, X., Zhang, X., Yu, X. (2021). Machine learning model and strategy for fast and accurate detection of leaks in water supply network. *Journal of Infrastructure Preservation and Resilience*, 2, e10. <https://doi.org/10.1186/s43065-021-00021-6>
- Foresee, F., Hagan, M. (1997). Gauss-Newton approximation to Bayesian learning. En *Proceedings of International Conference on Neural Networks (ICNN'97)*
- Gupta, K., Kishore, K., Jain, S. (2017). Modeling and simulation of CEERI's Water distribution network to detect leakage using HLR approach. En *6th International Conference on Reliability, Infocom Technologies and Optimization*. <https://doi.org/10.1109/ICRITO.2017.8342440>
- Jasper, M., Mahinthakumar, G., Ranjithan, S., Brill, E. (2013). Leak detection in water distribution systems using the dividing rectangles (DIRECT) search. En *World Environmental and Water Resources Congress*. <https://doi.org/10.1061/9780784412947.078>
- Lambert, A., Fantozzi, M., Shepherd, M. (2017). Pressure: Leak flow rates using FAVAD: An improved fast-track practitioner's approach. En *CCWi2017: Computing and Control in the Water Industry*
- Lu, H., Iseley, T., Behbahani, S., Fu, L. (2020). Leakage detection techniques for oil and gas pipelines: State-of-the-art. *Tunnelling and Underground Space Technology*, 98, e103249. <https://doi.org/10.1016/j.tust.2019.103249>
- Mashford, J., De Silva, D., Marney, D., Burn, S. (2009). An approach to leak detection in pipe networks using analysis of monitored pressure values by support vector machine. En *Third International Conference on Network and System Security*. <https://doi.org/10.1109/NSS.2009.38>
- Rojas, J., Verde, C., Torres, L. (2021). Estimation of hydraulic gradient for a transport pipeline. *Journal of Pressure Vessel Technology*, 143(3), e031801. <https://doi.org/10.1115/1.4048322>
- Rojek, I., Studzinski, J. (2019). Detection and localization of water leaks in water nets supported by an ICT system with artificial intelligence methods as away forward for smart cities. *Sustainability*, 11(2), e518. <https://doi.org/10.3390/su11020518>
- Sabu, S., Mahinthakumar, G., Ranjithan, R., Levis, J., Brill, D. (2021). Water leakage detection using neural networks. En *World Environmental and Water Resources Congress 2021: Planning a Resilient Future along America's Freshwaters*. <https://doi.org/10.1061/9780784483466.096>
- Simpson, P. (1991). *Artificial Neural Systems: Foundations, Paradigms, Applications, and Implementations*. McGraw-Hill



Nomenclatura

Símbolo	Unidad	Parámetro
D	m	Diámetro
ε	m	Rugosidad
f	Factor adimensional	Coeficiente de fricción antes de la fuga
f_1	Factor adimensional	Coeficiente de fricción después de la fuga
f_2	Factor adimensional	Coeficiente de fricción
g	m/s ²	Aceleración de la gravedad
H	m	Pérdida de carga total
h_f	m	Pérdida de carga por fricción
hl	m	Pérdida de carga por accesorios
J	m/m	Pérdida unitaria
J_1	m/m	Pérdida unitaria antes de la fuga
J_2	m/m	Pérdida unitaria después de la fuga
k	Factor adimensional	Coeficiente pérdida por accesorio
L	m	Longitud de la tubería
L_f	m	Posición lineal de la fuga
LGH	m	Línea de gradiente hidráulico
LE	Nm/N	Línea de energía
m.c.a	m	Metro columna de agua
P	mca	Presión
P_1	mca	Presión inicial
P_2	mca	Presión final
P_f	mca	Presión en la fuga
Q_1	m ³ /s	Caudal inicial
Q_2	m ³ /s	Caudal final
Q_f	m ³ /s	Caudal en la fuga
Re	Factor adimensional	Número de Reynolds
V	m/s	Velocidad
z	m	Cota
ν	m ² /s	Viscosidad cinemática

Abreviaturas

BCE	Entropía cruzada binaria
MAE	Error absoluto medio
MSE	Error medio cuadrático
LE	Línea de energía
lps	Litros por segundo
LGH	Línea de gradiente hidráulico
R	Coeficiente de correlación de Pearson
RNA	Red neuronal artificial
SAE	Suma absoluta de errores
SSE	Suma de errores al cuadrado
Logsig	Función de activación log sigmoide
FAVAD	Descargas de área fija y variable

Apéndice A. Codificación del modelo RNA para MatLab (2021a)

```

close all; clear all; clc; format long
%=====
% Edgar O. Ladino M. | César A. García U. | María Camila García V.
% Universidad Distrital Francisco José de Caldas | Facultad Tecnológica
% Ingeniería Civil
% Bogotá | Colombia
%=====

% ===== Cálculo de fuga tubería simple =====
% ===== Artificial neural networks =====
%
% 1. Importar dataset 35,838 datos
% Matriz (35838 x 7)
d = csvread('Dataset_35838_Fuga_30_m.csv'); %Archivo plano (csv)

%
% 2. Definición matriz p (Inputs); Vector t (Outputs)
p = d(:,1:4)'; % Matriz p: Transpuesta de la columna 1, 2, 3 y 4 de la matriz d
t = d(:,5:7)'; % Matriz t: Transpuesta de la columna 5, 6 y 7 de la matriz d
t1 = d(:,7)'; % Matriz t: Transpuesta de la columna 7 de la matriz d
%
% 3. Importar dataset matriz de prueba 5973 datos(Test)
test = csvread('DataTest_5973_una_Fuga_30_m.csv'); % Archivo plano (csv)

%
% 4. Definición matriz test_p (Inputs) 5,973 datos
test_p = test(:,1:4)'; % Matriz p: Transpuesta de la columna 1, 2, 3 y 4 de la matriz test
test_fuga = test(:,5:7)'; % Matriz p: Transpuesta de la columna 5, 6 y 7 de la matriz test
test_P1 = test(:,3)'; % Matriz p: Transpuesta de la columna 3 de la matriz test
test_Q3 = test(:,7)'; % Matriz p: Transpuesta de la columna 7 de la matriz test

%
% 5. Arquitectura de la red neuronal
net = fitnet(25); % feedforwardnet; patternnet; network; # neuronas
net.layers{1}.transferFcn = 'logsig'; % logsig; hardlim, tansig, purelin
net.performFcn = 'mse'; % mse; crossentropy; mae; msereg
net.trainFcn = 'trainbr'; % Entrenamiento: trainbr; trainlm; traingd; trainbfg; traingdm; trainscg; traingdx; trainoss;
traincgp;
net.divideFcn = 'dividerand'; % División: dividerand; divideblock; divideint; divideind
net.trainParam.epochs = 6000; % Controla el número de épocas
[net, tr] = train(net,p,t); % Entrenamiento de la red
view(net) % Grafica esquema de la red
y = net(p); % Función de la red
y1 = d(:,3)';
y_test = net(test_p); % Función de la red test 2,100 datos
y_test1 = y_test(3,:);
mse_test = 1/1991 * (test_fuga - y_test).^2;

%
% 6. Gráficas rendimiento de la red
figure;
plottrainstate(tr)
figure;
plotperform(tr)
figure;
plotregression(t,y)
figure;
plotregression(test_fuga,y_test)

%
```

```

% 7. Estimación del error
e = t-y; %y (entrenamiento) - y (Estimado)
figure;
ploterrhist(e,'bins',30)% Histograma de errores
R = corrcoeff(t,y)% Coeficientes de correlación
MAE = mae(e)% mae: Error absoluto medio
MSE = immse(t,y)% Error medio cuadrático
SSE = sse(net,t,y,1)% sse: Error de suma cuadrada
SAE = sae(net,t,y)% Suma absoluta de errores
BCE = crossentropy(net,t,y,{1},'regularization',0.1)% Entropía cruzada

%
% 8. Estimación del error datos de prueba (Test 350)
e_test = test_fuga-y_test; %y (entrenamiento) - y (Estimado)
figure;
ploterrhist(e_test,'bins',30)% Histograma de errores
R_test = corrcoeff(test_fuga,y_test)% Coeficientes de correlación
MAE_test = mae(e_test)% mae: Error absoluto medio
MSE_test = immse(test_fuga,y_test)% Error medio cuadrático
SSE_test = sse(net,test_fuga,y_test,1)% sse: Error de suma cuadrada
SAE_test = sae(net,test_fuga,y_test)% Suma absoluta de errores
BCE_test = crossentropy(net,test_fuga,y_test,{1},'regularization',0.1)%Entropía cruzada

%
% 9. Pesos y bias
w1 = net.IW{1};% Pesos de la capa de entrada a oculta
w2 = net.LW{2};% Pesos de la capa oculta a salida
b1 = net.b{1};% Sesgo de entrada a la capa oculta
b2 = net.b{2};% Sesgo de la capa oculta a la salida

%
% 10. Validación
input = [0.0780543012835326;0.0745543012835325;3.03118872394703;0.855419326975464]% Datos de prueba
output_fuga = sim(net, input);% Dato estimado
Presion_Fuga= output_fuga(1,1)
Localizacion_Fuga= output_fuga(2,1)
Caudal_Fuga= output_fuga(3,1)
%P3 (m)= 2.83
%L3 (m)= 2.55
%Q3 (m^3/s)= 0.0035

%
% 11. Gráfica 3D: Superficie de error
figure;
wv = -4:0.4:4;% Límites de la grilla; Tamaño del cuadrante
bv = wv;
ES = errsurr(y1,t1,wv,bv,'tansig');% y(Datos predecidos); t(Datos objetivos)
plotes(wv,bv,ES,[60 30])

%
% 12. Grafica 3D: Superficie de error MSE
x = test_Q3;
y1 = test_P1;
z = mse_test(3,:);
figure;
scatter3(x',y1',z','MarkerEdgeColor','k','MarkerFaceColor',[0 .75 .75])
view(-30,10)
xlabel('Caudal fuga (m^3/s)')
ylabel('Presión inicial (m)')
zlabel('Error medio cuadrático')

%

```

% 13. Grafica puntos de dispersión

```
figure;  
H50= test(:,5);  
D50=test(:,7);  
sz = 90;  
scatter(H50,D50,sz,'o')  
xlabel('Caudal fuga (m^3/s)')  
ylabel('Presión fuga (m)')  
hold on  
H50_test= test(:,5);  
D50_test=(y_test1)';  
sz = 70;  
scatter(H50_test,D50_test,sz,'+')  
hold on
```