Experimental study and modeling of hydraulic jump for a suddenly expanding stilling basin using different hybrid algorithms

Enes Gul, O. Faruk Dursun and Abdolmajid Mohammadian

ABSTRACT

Hydraulic jump is a highly important phenomenon for dissipation of energy. This event, which involves flow regime change, can occur in many different types of stilling basins. In this study, hydraulic jump characteristics such as relative jump length and sequent depth ratio occurring in a suddenly expanding stilling basin were estimated using hybrid extreme learning machine (ELM). To hybridize ELM, imperialist competitive algorithm (ICA), firefly algorithm (FA) and particle swarm optimization (PSO) metaheuristic algorithms were implemented. In addition, six different models were established to determine effective dimensionless (relative) input variables. A new data set was constructed by adding the data obtained from the experimental study in the present research to the data obtained from the literature. The performance of each model was evaluated using k-fold cross-validation. Results showed that ICA hybridization slightly outperformed FA and PSO methods. Considering relative input parameters, Froude number \( Fr \), expansion ratio \( B \) and relative sill height \( S \), effective input combinations were \( Fr-B-S \) and \( Fr-B \) for the prediction of the sequent depth ratio \( Y \) and relative hydraulic jump length \( Lj/h1 \), respectively.

Key words | cross-validation, evolutionary algorithm, extreme learning machine, hydraulic jump, machine learning, optimization

HIGHLIGHTS

- Suddenly expanding stilling basins were examined both experimentally and using AI.
- Hydraulic jump characteristics were estimated using hybrid extreme learning machine.
- New laboratory data was modeled using novel machine learning algorithms.
- Among optimization algorithms, ICA was superior to PSO and FA.
- The performance of each model was evaluated using \( k \)-fold cross-validation.

INTRODUCTION

Hydraulic jump is an important phenomenon that is widely used in hydraulic engineering. It is a rapid transition from a high-velocity flow to a slower stream movement. It usually occurs downstream of dam spillways, in streams and rivers and in industrial channels. There are many different types of stilling basins. According to plan geometry, they are classified as gradually expanding and suddenly expanding.

In the literature, hydraulic jump has been widely studied. Bakhmeteff & Matzke (1956) proposed dimensionless free-surface profiles and presented experimental data...
for the sequent depths and the length of jump. Bradley & Peterka (1957) developed a stilling basin type (i) where hydraulic jump occurs. Rajaratnam & Murahari (1971) presented an experimental study of forced hydraulic jumps formed with two-dimensional baffles or baffle walls. Hager (1985) studied different channel types dealing with hydraulic jumps in horizontal, rectangular and non-prismatic channels and U-shaped channels (Hager 1989). Gharangik & Chaudhry (1991) examined the one-dimensional Boussinesq equations, which were solved to simulate a hydraulic jump in a rectangular channel. Bremen & Hager (1993) comprehensively studied T-jump, which is one of the hydraulic jump types in abruptly expanding stilling basins. In addition, Bremen & Hager (1994) studied expanding stilling basins and the use of sills for energy dissipation. They proposed the use of a central sill as the most efficient method for energy dissipation. But they tested a range of expansion ratios $B = 0.33–0.67$ and suggested estimated values for $B = 0.25$. Zare & Doering (2011) examined forced hydraulic jump in the abrupt stilling basin.

Artificial intelligence studies have proved to be attractive modeling methods (Ebtehaj et al. 2016; Saghebian 2019; Dutta et al. 2020). Kisí (2005) examined the streamflow–suspended sediment relationship with artificial neural network (ANN) and neuro-fuzzy system (NF). Machine learning methods are widely used for the estimation of hydrological parameters. Kisí (2006) modeled daily pan evaporation process using NF and ANN techniques. Artificial intelligence is also widely used in the modeling of hydraulic structures. Paoli et al. (2010) used the multi-layer perceptron (MLP) network, which is the most widely used method among ANN architectures both for renewable energy and for time series forecasting. Roushangar et al. (2014) applied different methods to model energy dissipation in nappe and skimming flow regimes through a stepped spillway with ANNs and genetic expression programming (GEP) techniques. Azimi et al. (2018) modeled hydraulic jump characteristics in a rough channel bed using optimized firefly algorithm (FA) ANFIS. Ebtehaj & Bonakdari (2016) used evolutionary algorithms, particle swarm optimization (PSO) and imperialist competitive algorithms (ICA) to predict non-deposition sediment transport. In particular, ELM has many significant features that are distinct from the other aforementioned machine learning algorithms. Azimi et al. (2017) used ELM for sensitivity analysis to investigate the factors affecting the discharge coefficient in trapezoidal channels. They stated that ELM has advantages in terms of training speed and generalization performance. Ebtehaj et al. (2017) used an extreme learning machine method, self-adaptive extreme learning machine, to model maximum scour depth around bridge piers.

In the context of hydraulic jump, Güven et al. (2006) modeled pressure fluctuations beneath a type of hydraulic jump (B-jump) using a multilayer feed-forward neural network with a back-propagation learning algorithm. Karbasi & Azamathulla (2016) used GEP to predict characteristics of a hydraulic jump over a rough bed. They compared the performance of the GEP model, traditional equations and common artificial intelligence techniques (ANN and support vector regression). Roushangar et al. (2017) examined sudden expanding stilling basins. They modeled hydraulic jump characteristics such as sequent depth ratio ($h_2/h_1$) and relative jump length ($L_j/y_1$). The data used in their modeling was obtained from Bremen (1990). They compared GEP and existing empirical equations in the literature and found that GEP had the best performance. Senthil Kumar et al. (2015) studied modeling of streamflow at Kasol in India using decision tree, MT, fuzzy logic and ANN. They concluded that the decision tree algorithm (REPTree) consistently performed better in terms of selected performance criteria.

Prediction of hydraulic jump characteristics such as hydraulic jump length, width, roller length, and sill height is of great importance in hydraulic engineering. The aim of this study is to estimate characteristics of the hydraulic jump occurring in a suddenly expanding stilling basin with central sill with many different variable combinations using hybrid algorithms. Some of the data used in the modeling were obtained from the literature, and others were from a new experimental study performed in this research. The new experimental data has expansion ratio ($B = 0.25$) for the first time. Novel algorithms were used for modeling, such as ELM-PSO, ELM-FA and ELM-ICA.

The current study presents two significant novelties. First, evolutionary algorithms, PSO, FA and ICA are employed to investigate the modeling performances and second, new experimental data collected by the authors are combined with existing data in the literature for the
machine learning procedure. The data are divided into ten subsets for \( k \)-fold cross-validation, i.e. ten-fold cross-validation is performed. The new experimental data is added to the tenth fold. The performance of this fold is very important because it has a new data range. It should be noted that the machine learning algorithm may not learn accurately in the training phase. This makes it important to perform \( k \)-fold cross-validation especially with the broad range of data considered in the study, including the new experimental data added to the available literature data, which is an innovative aspect of the present research.

**MATERIALS AND METHODS**

**Data set**

A total of 165 data points were used to model the suddenly expanding stilling basin, ten of which were new. These data were obtained from Bremen & Hager (1994) and a new experimental study conducted in the Hydraulic Laboratory of Inonu University. The experiments of Bremen were carried out at the Laboratoire de Constructions Hydrauliques of the Ecole Polytechnique Federale de Lausanne (EPFL), which were designed for expanding channels with a central sill (Figure 1(a)). The new experiments were carried out as summarized in Table 1 (\( Q \) is the flow rate) with the experimental setup determined as \( b_1 = 30 \text{ cm} \), \( b_2 = 120 \text{ cm} \) and \( x_s = 18 \text{ cm} \) created by pumps controlled by a PLC automation system (Figure 1(b)). Discharge was measured by an electromagnetic flow meter with an accuracy of \( \pm 0.01 \text{ m}^3/\text{s} \). The water level was measured with a Mitutoyo digital meter with an accuracy of \( \pm 0.01 \text{ mm} \).

In Figure 1, \( b_1 \) is the width of the first section before sudden expansion, \( b_2 \) is the width of the second section with sudden expansion, \( b_s \) is the width of the central sill, \( x_1 \) is the length of the hydraulic jump before sudden expansion and \( x_j \) is the length of the hydraulic jump that occurs within a sudden expansion. Adding this value to \( x_1 \) gives the jump length \( L_j \) and \( Y \) is the ratio of the water depth after the hydraulic jump (\( h_2 \)) to the water depth before the hydraulic jump (\( h_1 \)); \( s \) is the height of the step or sill.

Hydraulic jump and dissipation of energy are influenced by the following parameters:

\[
f(L_j, \mu, g, b_1, b_2, s, h_1, h_2, V_1) = 0
\]  

where \( \mu \) is the dynamic viscosity of water, \( g \) is gravity acceleration, \( s \) is the sill height and \( V_1 \) is the velocity before

**Table 1** | New experimental results for modelings

| Run | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|-----|---|---|---|---|---|---|---|---|---|----|
| \( Q \) (l/s) | 40 | 35 | 30 | 25 | 20 | 40 | 35 | 30 | 25 | 20 |
| \( s \) (cm) | 5 | 5 | 5 | 5 | 5 | 7.5 | 7.5 | 7.5 | 7.5 | 7.5 |
| \( L_j \) (cm) | 211.3 | 200.3 | 186.4 | 175.4 | 168.6 | 190.3 | 184.6 | 174.7 | 156.9 | 143 |
| \( h_1 \) (cm) | 3.98 | 3.55 | 3.05 | 2.61 | 2.22 | 3.98 | 3.55 | 3.05 | 2.61 | 2.22 |
the hydraulic jump. With dimensional analysis, using parameters $\mu$, $g$ and $h_1$ as repeating variables, Equation (2) can be represented as follows:

$$f\left(\frac{h_2}{h_1}, \frac{L_j}{h_1}, \frac{b_1}{h_1}, \frac{b_2}{h_1}, \frac{\mu}{gh_1}, \frac{\rho V_1}{\mu}, \frac{s}{h_1}\right) = 0$$

Equation (3) can be expressed as:

$$f\left(Y, \frac{L_j}{h_1}, B, Fr_1, Re, S\right) = 0$$

where $Y$ is sequent depth ratio, $L_j/h_1$ is dimensionless jump length, $Fr_1$ is Froude number at the pre-jump, $Re$ is Reynolds number and $S$ is dimensionless sill height. In this study, $Y$ and $L_j/h_1$ values were estimated using dimensionless input values of $Fr$, $B$ and $S$ because of reports that they cannot be affected by $Re$ (Elevatorski 1959; Ranga Raju et al. 1980; Roushangar & Ghasempour 2018). Bremen & Hager (1994) tried many types of energy dissipator blocks, but they were convinced that the most effective type is the central sill for suddenly expanding stilling basins. Therefore, the hydraulic jump characteristics of the proposed central sill are modeled using new experimental data. The dimensionless parameters are expansion ratio, dimensionless sill height, sequent depth ratio, Froude number and dimensionless jump length $B = b_1/b_2$, $S = s/h_1$, $Y = h_2/h_1$, $Fr = \sqrt{gh_1}$ and $L_j/h_1$, respectively, as shown in Table 2. Six different model input combinations were examined to analyze the sensitivity of the variables. The established models are summarized in Table 3.

**Extrem learning machine**

ELM is a neural network with a single hidden layer (Huang Zhu & Siew 2006). The learning structure of ELMs has many advantages over the classical BP algorithm. While the conventional BP algorithm, a gradient-based learning process, is calculated by tuning, ELM begins with the generation of random weight and bias values for the network. ELM consists of three layers, an input layer, an output layer and a hidden layer. These layers form a single hidden layer forward network where linear algebra is used for calculating the equations to obtain optimum output layer weights. The training process is extremely fast and the generalization potential is high in ELM (Li et al. 2019).

Figure 2 displays the typical architecture of the optimized ELM. Mathematical representation of the $N$-sample

| Model Name | Selected Parameters | Predicted Parameter |
|------------|---------------------|---------------------|
| Model 1    | $Fr$, $B$, $S$      | $Y = \frac{h_2}{h_1}$ |
| Model 2    | $Fr$, $S$           |                     |
| Model 3    | $Fr$, $B$           |                     |
| Model 4    | $Fr$, $B$, $S$      | $L_j/h_1$           |
| Model 5    | $Fr$, $S$           |                     |
| Model 6    | $Fr$, $B$           |                     |

**Table 2 | Statistical indices for experimental tests**

| Data Type | Minimum | Maximum | Mean   | Std Deviation | Skewness |
|-----------|---------|---------|--------|---------------|----------|
| $Fr$      | 2.97    | 9.02    | 6.05   | 1.96          | -0.026   |
| $B$       | 0.25    | 0.67    | 0.48   | 0.15          | 0.014    |
| $S$       | 0.66    | 3.33    | 1.44   | 0.73          | 1.08     |
| $Y$       | 2.89    | 10.63   | 6.29   | 2.14          | 0.12     |
| $L_j/h_1$ | 34.43   | 118.25  | 71.99  | 23.37         | 0.1873   |

| Figure 2 | Structure of optimized ELM.
ELM is as follows:

$$\sum_{i=1}^{L} \beta_i g(x_j, w_i, b_i) = y_j, \quad j \in \mathbb{R}^N$$

(4)

where \( \beta \) is output weights, \( L \) is the number of hidden nodes, \( w \) is the input weights, \( b \) is the bias values and \( g(x) \) is the activation function. Expressions of the hidden layer output matrix and compact view are as follows, respectively:

$$G(x) = \begin{pmatrix} g(x_1, w_1, b_1) & \cdots & g(x_1, w_L, b_L) \\ \vdots & \ddots & \vdots \\ g(x_N, w_1, b_1) & \cdots & g(x_N, w_L, b_L) \end{pmatrix}_{N \times L}$$

(5)

$$G\beta = T$$

(6)

where \( T \) is the target vector. The hidden layer output weight vector \( \beta \) is calculated with the Moore-Penrose generalized inverse of matrix \( G \) as in Equation (7):

$$\beta = G^+ T$$

(7)

**Particle swarm optimization**

The particle swarm optimization (PSO), meta-heuristic algorithm was inspired by the actions of fish and birds, and was developed by Kennedy & Eberhart (1995). The PSO, a population-based approach to stochastic optimization, starts with a random solution or a particle population in the search area and updates optima iteratively. The consequence of this simulation of social behavior is a search mechanism by which particles travel to appropriate locations. Particles learn from each other in the community based on information gained; they move towards better neighbors. At any moment, a particle changes its location in the search space to the best position by far and the best position in the neighborhood. Particle \( i \) is considered to be a vector and position vector in an area of the \( n \)-dimensional space. Growing particle update uses two demonstrative particles. First, the best solution to date, called ‘pbest’, is found by particles. Another is the best ever between all particles in the ‘gbest’ group. The PSO algorithm structure is shown in Figure 3.

The first step in Figure 3 is an arbitrary distribution of speeds and sites to begin the initial population. The next step is to test this particle using a statistical approach in a regression analysis. One can stop the scheme and export the parameters specified, once the best fitness standard of particulates meets the stop criterion. If the level of operation is insufficient for interruptions, the particle speed and position will be changed in two cases (Kennedy & Eberhart 1995; Shi & Eberhart 1999).

Every particle in PSO is a candidate solution in the \( n \)-dimensional search space. The position of a particle \( i \) at any iteration is \( x_i = [x_1, x_2, \ldots, x_n] \) and the velocity is \( v_i = [v_1, v_2, \ldots, v_n] \) in \( n \)-dimensional space. The new velocity of each particle in the search space is calculated as follows:

$$v_{ji}(t+1) = \omega v_{ji}(t) + c_1 r_1 (y_j(t) - x_{ji}(t)) + c_2 r_2 (y_j(t) - x_{ji}(t))$$

(8)

where \( j \) is the dimension of the search space \( i \in [1, 2, \ldots, n] \), \( i \) is the number of iterations, \( \omega \) is the inertia weight, \( y_j(t) \) is the
Every firefly emits. The mathematical representation of the intensity produced by the firefly is directly related to the radiation it emits. The mathematical representation of the intensity and attraction of a firefly is given in Equations (10) and (11). Every firefly’s ability to attract another firefly depends on its sequence similarity $I$ (Yang 2010; Tao et al. 2018):

$$w(r) = w_0 e^{-yr^2}$$  \hspace{1cm} (10)$$
$$I = I_0 e^{-yr^2}$$  \hspace{1cm} (11)$$

where attraction at a distance is represented by $w(r)$ and the light intensity is represented by $I$. $I_0$ and $w_0$ represent the intensity of the emitted light and the attraction at a distance $r = 0$ from the firefly and $y$ represents the light absorption coefficient. Equation (12) provides the distance $r$ between any two fireflies $j$ and $m$ (Yang 2010):

$$r_{jm} = x_j + x_m = \sqrt{\sum_{k=1}^{d} (x_{j,k} - x_{m,k})^2}$$  \hspace{1cm} (12)$$

where $d$ is the population of the fireflies, and $x_j$ and $x_m$ are the location of the fireflies in the Cartesian coordinate system. As explained, each firefly is attracted by the others and vice-versa so the movement of fireflies is expressed in Equation (13) for the $j$th firefly by the $m$th firefly:

$$\overline{x}_m(t + 1) = \overline{x}_m(t) + \beta_{jm}(r_{jm})(\overline{x}_j(t) - \overline{x}_m(t)) + \alpha \overline{e}_m$$  \hspace{1cm} (13)$$

where $\alpha$ is a constant between 0 and 1 and $\overline{e}_m$ is a random number vector obtained from the Gaussian distribution.

### Imperial competitive algorithm

Through this newly introduced meta-heuristic methodology (Atashpaz-Gargari & Lucas 2007), ICA imitates imperialist social political activity and imperialist competition between countries in order to solve optimization problems. Once the colonies are divided into imperialists, they start with their own territory. As a result, a whole empire’s power is defined by the colonial country’s power plus a proportion of its colonies. For imperial competition, empires attempt to increase their overall strength by attracting settlements from other empires. As a result, the power of weaker empires will decline gradually and the power of more powerful empires will increase. Within the colonial competition, there is a competition between empires that control the weaker colonies belonging to the weaker empires. The stronger an empire, the more likely they are to attract the weakest colonies. Weak empires that are unable to compete with others and will not increase their total power will be excluded after some algorithm iterations. This will slowly increase the overall strength of large empires, while small empires will increasingly lose control and ultimately disappear. The associated colonies will be divided among other empires until an empire is removed. This cycle brings all the colonies together in a community in which only one empire remains in the world and all the others are their colonies (Figure 4). Each state is characterized as a $1 \times n$ array in an $n$-dimensional optimization problem:

$$Country = [p_1, p_2, \ldots, p_n]$$  \hspace{1cm} (14)$$

Among the countries with the best values, imperialists are chosen and the rest of the countries serve as colonies of these imperialists. Initial empires are generated by distributing colonies among imperialists proportionally based on imperialist normalization ($p_n$):

$$p_n = \frac{c_n}{\sum_{i=1}^{N_{imperial}} c_i}$$  \hspace{1cm} (15)$$

where $c_n$ is the imperialist $n$th cost. The normalized imperialist cost ($C_n$) is determined using the formula:

$$C_n = c_n - \text{max} (c_i)$$  \hspace{1cm} (16)$$

where \( \max(c_i) \) is the imperialist with maximum cost (weakest imperialist).

**k-fold cross validation**

Cross-validation tests the performance of a predictive model and is applied to a specific data set in statistical analyses. Many kinds of cross-validations are available, including repeated random sub-validations, \( k \)-fold cross-validation, Monte Carlo test, etc. as seen in Figure 5 (Bengio & Grandvalet 2004; Rohani et al. 2018).

In the first stage, the data set to be evaluated is divided into subsets equal to \( k \). Up to \( k-1 \) subsets are selected as training data for the model. The fold-\( t \) subset is selected as test data. The calculated accuracy value for the fold-\( t \) subset is compared to the other subsets to ensure that the fold-\( t \) subset is not used in training. This process is repeated \( k \) times, with each subset serving as the test data once. The average accuracy over all \( k \) iterations is used as the final accuracy value.

![Figure 4](image1)  
**Figure 4** | Flowchart and schematic view for ICA.

![Figure 5](image2)  
**Figure 5** | Cross-validation types.
subset is added to the cross validation (CV) array. This process is repeated for the number of subsets \((k)\). All accuracies calculated in the final process are averaged. Either this average or the lowest accuracy is used to indicate the performance of the model (see Figure 6).

RESULTS AND DISCUSSION

Hybridization

The aforementioned optimization algorithms (PSO, FA and ICA) were implemented for the hybridization of ELM. A number of hidden layer neurons between ten and 20 was selected for each model to provide the optimum performance. With bias and weight values collected in a vector, firstly, the initial population was created for metaheuristic algorithms. Secondly, the population was searched for the best solution, and according to the best weight and bias values, the test data was investigated. This process was performed in each fold. Table 4 shows the initial parameters of the evolutionary algorithms.

Evaluation of model performance

The models were compared using standard statistical performance evaluation measures such as correlation coefficient \((R)\), root mean square error (RMSE), mean absolute error (MAE), variance (VAF), scatter index (SI), mean absolute relative error (MARE), mean relative error (MRE), BIAS and Nash–Sutcliffe model efficiency coefficient (Nash), which are represented in Equations (17)–(25), respectively. As the RMSE approaches 0 and the \(R\) value gets close to 1, the model performs better. The MAE value may range from 0 to \(\infty\). Negative-oriented scores, i.e. those with lower values, perform better. Nash is used generally in hydrological studies and can range from \(\infty\) to 1. If the index value is close to 1, the model performs better. Also, if the MRE and SI values are close to 0, the model performs better. Through this, a ten-fold CV was applied to conduct a performance evaluation.

Table 4 | Initial parameters for PSO, FA and ICA evolutionary algorithms

| Parameters of Algorithms | PSO                   | FA                   | ICA                   |
|--------------------------|-----------------------|----------------------|-----------------------|
|                          | Maximum Number        | Population Size     | Light Absorption      | Maximum Number       | Population Size | Number of Empires | Maximum Number    | Population Size | Number of Empires |
|                          | Iterations            |                      | Coefficient           | Iterations            |                      |                    | Iterations        |                      |                    |
|                          | 100                   | 200                  | 0.1                   | 100                   | 20                   | 10                  | 100                 | 100                 | 10                  |
|                          | 1.0                   | 2.0                  | 1.5                   | 0.1                   | 0.9                  | 1.5                 | 0.5                 | 0.5                 | 0.5                 |
|                          | Inertia Weight         |                      | Inertia Weight Damp   | Personal Learning     | Global Learning      |
|                          | 0.1                   |                      | Rate                  | Coefficient           | Coefficient          |
|                          | 0.009                 |                      | 0.05                  | 2                     |                      |
|                          |                       |                      |                       |                       |                      |
|                          | All except fold-t are |                      |                       |                       |                      |
|                          | used as training data |                      |                       |                       |                      |
|                          | sets.                 |                      |                       |                       |                      |
|                          | Test model with fold-t|                      |                       |                       |                      |
|                          | subset                 |                      |                       |                       |                      |
|                          | fold-t predictions    |                      |                       |                       |                      |
|                          | fold-K                |                      |                       |                       |                      |
|                          | CV array              |                      |                       |                       |                      |

Figure 6 | Schematic for k-fold cross-validation.
for all data.

\[
R = \frac{\sum_{i=1}^{n} (y_i - \bar{y}_{i,m})(\tilde{y}_i - \hat{y}_{i,m})}{\sqrt{\sum_{i=1}^{n} (y_i - \bar{y}_{i,m})^2 \sum_{i=1}^{n} (\tilde{y}_i - \hat{y}_{i,m})^2}}
\]  

(17)

\[
RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \bar{y}_i)^2}
\]  

(18)

\[
MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \bar{y}_i|
\]  

(19)

\[
VAF = \left(1 - \frac{\text{variance}(y_i - \bar{y}_i)}{\text{variance}(y_i)}\right) \times 100
\]  

(20)

\[
SI = \frac{RMSE}{\bar{y}_{i,m}}
\]  

(21)

\[
MARE = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - \bar{y}_i|}{|y_i|}
\]  

(22)

\[
MRE = \frac{1}{n} \sum_{i=1}^{n} \frac{y_i - \bar{y}_i}{y_i}
\]  

(23)

\[
BIAS = \frac{1}{n} \sum_{i=1}^{n} y_i - \bar{y}_i
\]  

(24)

\[
NASH = 1 - \left[ \frac{\frac{1}{n} \sum_{i=1}^{n} (y_i - \bar{y}_i)^2}{\frac{1}{n} \sum_{i=1}^{n} (\tilde{y}_i - \hat{y}_{i,m})^2} \right]
\]  

(25)

In Equations (17)–(25), \(y_i\) is the observed value, \(\hat{y}_i\) is the predicted value and subscript \(m\) indicates mean value.

**Comparison of models**

All the simulations were conducted in the MATLAB 2016 environment running on a PC with 2.67 GHz CPU and 4 Gb memory space. ELM was tuned using PSO, FA, and ICA during the training phase. RMSE was used as the best objective function in the process. The evaluation was continued using 100 iterations. The stability of the machine learning models highly depends on the properties of the data. As can be seen in Figure 7, stability of folds for \(Y\) is better than \(L_i/h_1\). As can be seen in the VAF results in Table 5, this may be due to the lower variance of \(Y\). ELM-PSO, ELM-FAn and ELM-ICA became almost stable after the 20th iteration. When the behavior of each fold is analyzed, almost all folds during the training lead to very close error rates for the \(Y\) prediction, while there are differences in the behavior of the folds for the \(L_i/h_1\) prediction (Figure 7). In Table 6, CPU time is summarized for each of the folds for ELM-PSO, ELM-FA and ELM-ICA during the training phase. ELM-ICA outperformed ELM-PSO and ELM-FA for every fold and every model considering time consumption.

In Table 5, the results are summarized for models that predicted \(Y\) and \(L_i/h_1\). Results for the prediction of \(Y\) indicate that Model 1 was superior to other models including ELM-PSO, ELM-FA and ELM-ICA, with RMSE = 0.19, RMSE = 0.19 and RMSE = 0.24, respectively. When comparing the machine learning algorithms, ELM-FAn outperformed ELM-PSO and ELM-ICA for Model 1, with Nash = 0.97, Nash = 0.96 and Nash = 0.97. Results for relative hydraulic jump length, \(L_i/h_1\), indicate that Model 6 was superior to other models among the machine learning algorithms, ELM-PSO, ELM-FAn and ELM-ICA, with RMSE = 10.56, RMSE = 10.44 and RMSE = 10.35, respectively. ELM-ICA was slightly better than ELM-PSO and ELM-FAn. Scatter plots of all models are presented in Figure 8. As can be seen in Figure 8, \(Y\) is in the shaded area (±10% confidence intervals) while \(L_i/h_1\) is distributed. Despite the addition of experimental data with a new range, very good results were obtained. Training and testing data are very important for machine learning methods. These methods can better estimate the range of data learned during the testing phase. Therefore, the distribution of training and test data is very important. In this study, unlike the suddenly expanding energy dissipation models in the literature, both existing experimental data in the literature and the collected data with a new range were used and evaluated with the reliable \(K\)-fold cross validation method. Despite all these difficulties, hybrid models, ELM-PSO, ELM-FAn and ELM-ICA generated almost the same results as SVM and GEP. Roushangar et al. (2017) and Roushangar & Ghasempour (2018) modeled data from Bremen & Hager (1994) via SVM and...
Figure 7 | RMSE error for each iteration in the training phase. (Continued.)
Figure 7 | Continued.
Figure 7  | Continued.
In this study, the percentage of error (RMSE_p) was considered for each of the folds as follows:

$$RMSE_p = \frac{\sum_{i=1}^{CV} RMSE_i}{CV}$$

where CV is the total fold number. Figure 9 displays the RMSE_p for each of the models for ELM-PSO, ELM-FA and ELM-ICA. Fold 10 consists of new experimental data. The results indicate that for prediction of $Y$, fold 10 adapted well. When considering Model 1, which is the best model, ELM-ICA was superior to ELM-PSO and ELM-FA for fold 10 with $RMSE_p = 17\%$, $RMSE_p = 17\%$ and $RMSE_p = 15\%$, respectively. On the other hand, the results indicate that for prediction of $L_j/h_1$, fold 10 adapted better than in Model 6 for ELM-PSO ($RMSE_p = 8\%$ and $RMSE_p = 8\%$), ELM-FA ($RMSE_p = 9\%$ and $RMSE_p = 8\%$) and ELM-ICA ($RMSE_p = 8\%$ and $RMSE_p = 7\%$), respectively (Figure 9). As can be seen in Table 5 and Figure 9, Model 1 and Model 6, which are the best models, had worse performance for fold 10 in ELM-PSO ($RMSE_p = 17\%$ and $RMSE_p = 23\%$), ELM-FA ($RMSE_p = 17\%$ and $RMSE_p = 23\%$) and ELM-ICA ($RMSE_p = 17\%$ and $RMSE_p = 23\%$), respectively. When the best models, Model 1 and Model 6, were investigated, fold 7 and fold 10 showed almost the same performance for all machine learning algorithms.

### Table 5 | Performance criteria for modeling with different models

| Models    | $R$ | VAF | RMSE | SI  | MAE | MARE | MRE  | BIAS | Nash |
|-----------|-----|-----|------|-----|-----|------|------|------|------|
| ELM-PSO   |     |     |      |     |     |      |      |      |      |
| Model 1   | 0.99| 97.99| 0.19 | 0.03| 0.15| 0.03 | 0.002| 0.005| 0.96 |
| Model 2   | 0.99| 97.95| 0.19 | 0.03| 0.15| 0.03 | 0.0009| − 0.004| 0.96 |
| Model 3   | 0.99| 96.18| 0.51 | 0.08| 0.44| 0.07 | 0.010| 0.034| 0.90 |
| Model 4   | 0.92| 72.13| 16.42| 0.23| 14.81| 0.21 | 0.056| 0.899| 0.48 |
| Model 5   | 0.92| 74.99| 15.92| 0.23| 14.46| 0.20 | 0.055| 0.861| 0.50 |
| Model 6   | 0.92| 71.32| 10.56| 0.15| 8.85 | 0.12 | 0.034| 0.828| 0.72 |
| ELM-FA    |     |     |      |     |     |      |      |      |      |
| Model 1   | 0.99| 97.99| 0.19 | 0.03| 0.15| 0.03 | 0.002| 0.007| 0.97 |
| Model 2   | 0.99| 97.83| 0.21 | 0.03| 0.17| 0.03 | 0.004| 0.008| 0.96 |
| Model 3   | 0.99| 96.26| 0.50 | 0.08| 0.44| 0.07 | 0.010| 0.029| 0.91 |
| Model 4   | 0.92| 75.48| 16.12| 0.23| 14.67| 0.21 | 0.056| 0.791| 0.48 |
| Model 5   | 0.92| 75.54| 15.62| 0.22| 14.07| 0.19 | 0.051| 0.662| 0.50 |
| Model 6   | 0.92| 71.99| 10.44| 0.15| 8.71 | 0.12 | 0.034| 0.769| 0.72 |
| ELM-ICA   |     |     |      |     |     |      |      |      |      |
| Model 1   | 0.99| 97.67| 0.24 | 0.04| 0.19| 0.03 | 0.0008| 0.004| 0.97 |
| Model 2   | 0.99| 97.60| 0.24 | 0.04| 0.19| 0.03 | − 0.0006| − 0.008| 0.96 |
| Model 3   | 0.99| 96.47| 0.50 | 0.08| 0.44| 0.08 | 0.011| 0.028| 0.91 |
| Model 4   | 0.92| 77.62| 15.19| 0.21| 15.70| 0.19 | 0.051| 0.502| 0.50 |
| Model 5   | 0.92| 76.21| 14.67| 0.21| 15.16| 0.18 | 0.042| 0.143| 0.53 |
| Model 6   | 0.92| 71.05| 10.35| 0.15| 8.60 | 0.12 | 0.033| 0.702| 0.72 |
Experimental validation

The design of a suddenly expanding stilling basin depends on different hydraulic jump types such as (a) repelled jump, (b) spatial jump (S-jump), (c) transitional jump (T-jump) and (d) submerged spatial jump. In a symmetrical expansion, Herbrand (1973) investigated the S-jump and used the momentum equation to derive a simple formula (Equation (27)):

\[ \frac{Y}{Y^*} = \sqrt{B} \]  

with

\[ Y^* = \frac{h_2}{h_1} = \frac{1}{2} \left[ \sqrt{8Fr_1^2 + 1} - 1 \right] \]  

where \( Y^* \) is sequent depth ratio for classical hydraulic jump (CHJ). Also, Herbrand (1973) suggested an equation for hydraulic jump length (\( L_{j} \)) as follows:

\[ \frac{L_{j}}{L_{j}^*} = 0.8, \quad B < 0.64 \]  

in which \( L_{j}^* \) is hydraulic jump length for CHJ. Bremen & Hager (1993) and Bremen & Hager (1994) investigated the T-jump in a suddenly expanding stilling basin. They suggested an empirical equation for sequent depth ratio and hydraulic jump length, respectively, as:

\[ \frac{L_{j}}{L_{j}^*} = 1 + \psi \]  

with

\[ \psi = \frac{Y'}{Y^*} - 1 = \left( \frac{1}{\sqrt{B}} \right) [1 - \tanh (1.9X_1)] \]  

\[ Y^* = \frac{h_2}{h_1} = \sqrt{2Fr_1} - (1/2) \]
Figure 8 | Scatter for all models with ±10% confidence intervals (shaded area).
Figure 9 | Error pie charts for ELM-PSO, ELM-FA and ELM-ICA.
where $X_1 = x_1/L^*_1$; $x_1$ is the distance between the toe of the jump and the expansion; and $L^*_1$ is the roller length for CHJ. Zare & Doering (2011) investigated asymmetric and symmetric expansion and suggested an empirical equation for sequent depth ratio and hydraulic jump length, respectively, as:

$$Y/Y^* = 0.65B + 0.35$$

and

$$L^*_1 = [B^{0.5} \exp(1 - B)^{0.75}] + [(1 - B)(0.1 + B)(0.4 - 0.2)]$$

where $0.4 - 0.5$.
where $\delta = S/LS$; $\delta$ is a dimensionless parameter to control sill height and sill location; $LS = L_s/h_0$; $L_s$ is the distance from the initial hydraulic jump length to the sill, $h_0$ is the opening of the gate and $S = s/h_0$, respectively.

The performance of various models for the new experimental data obtained from the experiments in the present research for sequent depth ratio and relative jump length is presented in Table 7. Model 3 and Model 5, which are superior models in machine learning algorithms, were used for new experimental data fitness (Figure 10). ELM-PSO, ELM-FA and ELM-ICA outperformed conventional regression equations, Herbrand (1973), Bremen & Hager (1994), and Zare & Doering (2011) for estimation of $Y$ with RMSE $= 0.63$, RMSE $= 0.64$, RMSE $= 0.61$, RMSE $= 1.77$, RMSE $= 0.68$ and RMSE $= 1.67$, respectively (Table 7). Measurement of the hydraulic jump length is very difficult because it has a dynamic length. Therefore, the variance of the values of $L_j/h_1$ was higher than the value of $Y$ (Table 7).

CONCLUSIONS

In this study, the hydraulic jump characteristics for a suddenly expanding stilling basin were estimated with sequent depth ratio and relative hydraulic jump length $Y$ and $L_j/h_1$ using novel hybrid machine learning algorithms. This study presents a novel method to examine the suitability of new experimental data with those previously presented in the literature. New experimental data ranges ($B = 0.25$) that are not in the literature were modeled using novel machine learning algorithms, ELM-PSO, ELM-FA and ELM-ICA, by adding them to the existing data. The classical approach examines the suitability of new experimental data with the regression equations proposed in the literature. In this study, the suitability of new data with previous studies was examined with both classical regression equations and the $k$-fold cross validation method. As a result, it was shown that the tenth fold of the new experimental data presented in this study fits very well with the folds containing the data previously presented in the literature. The best input combinations according to the $Y$ and $L_j/h_1$ modeling results for ELM-ICA, which is superior among the modeling methods, are Model 1 and Model 6, $Fr-B-S$ and $Fr-S$, respectively. Machine learning model results were better than classical regression results. Compared with optimization algorithms, ICA was superior to PSO and FA.

ACKNOWLEDGEMENTS

This research was supported by IUBAP (Inonu University Scientific Projects Unit) under the project numbers FCD-2018-1324 and FBG-2018-1474.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest.
DATA AVAILABILITY STATEMENT

All relevant data are included in the paper or its Supplementary Information.

REFERENCES

Atashpaz-Gargari, E. & Lucas, C. 2007 Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In: 2007 IEEE Congress on Evolutionary Computation. IEEE, Piscataway, NJ, USA, pp. 4661–4667.

Azimi, H., Bonakdari, H. & Ebtehaj, I. 2017 Sensitivity analysis of the factors affecting the discharge capacity of side weirs in trapezoidal channels using extreme learning machines. Flow Measurement and Instrumentation 54, 216–223.

Azimi, H., Bonakdari, H., Ebtehaj, I. & Michelson, D. G. 2018 A combined adaptive neuro-fuzzy inference system–firefly algorithm model for predicting the roller length of a hydraulic jump on a rough channel bed. Neural Computing and Applications 29 (6), 249–258.

Bakhmeteff, B. M. & Matzke, A. E. 1936 The hydraulic jump in terms of dynamic similarity. Trans. ASCE 101, 630–647.

Bengio, Y. & Grandvalet, Y. 2004 No unbiased estimator of the variance of K-fold cross-validation. Journal of Machine Learning Research 5, 1089–1105.

Bradley, J. N. & Peterka, A. J. 1957 The hydraulic design of stilling basins: hydraulic jumps on a horizontal apron (basin I). Journal of the Hydraulics Division 83 (3), 1–24.

Bremen, R. 1990 Expanding Stilling Basin. EPFL-LCH, Lausanne, Switzerland.

Bremen, R. & Hager, W. H. 1993 T-jump in abruptly expanding channel. Journal of Hydraulic Research 31 (1), 61–78.

Bremen, R. & Hager, W. H. 1994 Expanding stilling basin. Proceedings of the Institution of Civil Engineers – Water Maritime and Energy 106 (3), 215–228.

Dutta, D., Mandal, A. & Afzal, M. S. 2020 Discharge performance of plan view of multi-cycle W-form and circular arc labyrinth weir using machine learning. Flow Measurement and Instrumentation 73, 101740.

Ebtehaj, I. & Bonakdari, H. 2016 Assessment of evolutionary algorithms in predicting non-deposition sediment transport. Urban Water Journal 13 (5), 499–510.

Ebtehaj, I., Bonakdari, H., Shamshirband, S. & Mohammadi, K. 2016 A combined support vector machine–wavelet transform model for prediction of sediment transport in sewer. Flow Measurement and Instrumentation 47, 19–27.

Ebtehaj, I., Sattar, A. M. A., Bonakdari, H. & Zaji, A. H. 2017 Prediction of scour depth around bridge piers using self-adaptive extreme learning machine. Journal of Hydroinformatics 19 (2), 207–224.

Elevatorski, E. A. 1959 Hydraulic Energy Dissipators. McGraw-Hill, New York, USA.

Gharangik, A. M. & Chaudhry, M. H. 1991 Numerical simulation of hydraulic jump. Journal of Hydraulic Engineering 117 (9), 1195–1211.

Güven, A., Günal, M. & Çevik, A. 2006 Prediction of pressure fluctuations on sloping stilling basins. Canadian Journal of Civil Engineering 33 (11), 1379–1388.

Hager, W. H. 1985 Hydraulic jump in non-prismatic rectangular channels. Journal of Hydraulic Research 23 (1), 21–35.

Hager, W. H. 1989 Hydraulic jump in U-shaped channel. Journal of Hydraulic Engineering 115 (5), 667–675.

Herbrand, K. 1973 The spatial hydraulic jump. Journal of Hydraulic Research 11 (3), 205–218.

Huang, G.-B., Zhu, Q.-Y. & Siew, C.-K. 2006 Extreme learning machine: theory and applications. Neurocomputing 70 (1–3), 489–501.

Karbasi, M. & Azamathulla, H. M. 2016 GEP to predict characteristics of a hydraulic jump over a rough bed. KSCE Journal of Civil Engineering 20 (7), 3006–3011.

Kennedy, J. & Eberhart, R. 1995 Particle swarm optimization. In: Proceedings of ICNN’95 – International Conference on Neural Networks. IEEE, Piscataway, NJ, USA, pp. 1942–1948.

Kisi, O. 2005 Suspended sediment estimation using neuro-fuzzy and neural network approaches/Estimation des matières en suspension par des approches neurofleuses et à base de réseau de neurones. Hydrological Sciences Journal 50 (4), 683–696.

Kisi, Ö. 2006 Daily pan evaporation modelling using a neuro-fuzzy computing technique. Journal of Hydrology 329 (3–4), 636–646.

Li, L.-L., Wen, S.-Y., Tseng, M.-L. & Wang, C.-S. 2019 Renewable energy prediction: a novel short-term prediction model of photovoltaic output power. Journal of Cleaner Production 228, 359–375.

Paoli, C., Vuyant, C., Muselli, M. & Nivet, M.-L. 2010 Forecasting of preprocessed daily solar radiation time series using neural networks. Solar Energy 84 (12), 2146–2160.

Rajaratnam, N. & Murahari, V. 1971 A contribution to forced hydraulic jumps. Journal of Hydraulic Research 9 (2), 217–240.

Ranga Raju, K. G., Kitaal, M. K., Verma, M. S. & Ganeshan, V. R. 1980 Analysis of flow over baffle blocks and end sills. Journal of Hydraulic Research 18 (3), 227–241.

Rohani, A., Taki, M. & Abdollahpour, M. 2018 A novel soft computing model (Gaussian process regression with K-fold cross validation) for daily and monthly solar radiation forecasting (part: I). Renewable Energy 115, 411–422.

Roughangir, K. & Ghasempour, R. 2018 Explicit prediction of expanding channels hydraulic jump characteristics using gene expression programming approach. Hydrology Research 49 (3), 815–830.

Roughangir, K., Akhgar, S., Salmasi, F. & Shiriri, J. 2014 Modeling energy dissipation over stepped spillways using machine learning approaches. Journal of Hydrology 508, 254–265.

Roughangir, K., Valizadeh, R. & Ghasempour, R. 2017 Estimation of hydraulic jump characteristics of channels with sudden...
diverging side walls via SVM. *Water Science and Technology* 76 (7), 1614–1628.

Saghebian, S. M. 2019 Predicting the relative energy dissipation of hydraulic jump in rough and smooth bed compound channels using SVM. *Water Supply* 19 (4), 1110–1119.

Senthil Kumar, A. R., Goyal, M. K., Ojha, C. S. P., Singh, R. D. & Swamee, P. K. 2013 Application of artificial neural network, fuzzy logic and decision tree algorithms for modelling of streamflow at Kasol in India. *Water Science and Technology* 68 (12), 2521–2526.

Shi, Y. & Eberhart, R. C. 1999 Empirical study of particle swarm optimization. In: *Proceedings of the 1999 Congress on Evolutionary Computation - CEC99*. IEEE, Piscataway, NJ, USA, pp. 1945–1950.

Tao, H., Diop, L., Bodian, A., Djaman, K., Ndiaye, P. M. & Yaseen, Z. M. 2018 Reference evapotranspiration prediction using hybridized fuzzy model with firefly algorithm: regional case study in Burkina Faso. *Agricultural Water Management* 208, 140–151.

Yang, X.-S. 2010 Firefly algorithm, stochastic test functions and design optimisation. *arXiv* preprint, arXiv:1003.1409.

Zare, H. K. & Doering, J. C. 2011 Forced hydraulic jumps below abrupt expansions. *Journal of Hydraulic Engineering* 137 (8), 825–835.