




## Article

# Obtaining Bricks Using Silicon-Based Materials: Experiments, Modeling and Optimization with Artificial Intelligence Tools

Costel Anton <sup>1</sup>, Florin Leon <sup>2,\*</sup>, Marius Gavrilescu <sup>2</sup>, Elena-Niculina Drăgoi <sup>1</sup>, Sabina-Adriana Floria <sup>2</sup>,  
Silvia Curteanu <sup>1</sup> and Cătălin Lisa <sup>1,\*</sup>

<sup>1</sup> Faculty of Chemical Engineering and Environmental Protection, “Gheorghe Asachi” Technical University of Iasi, Bd. Mangeron, No. 73, 700050 Iasi, Romania; costel.anton@gmail.com (C.A.); elena-niculina.dragoi@academic.tuiasi.ro (E.-N.D.); silvia.curteanu@academic.tuiasi.ro (S.C.)

<sup>2</sup> Faculty of Automatic Control and Computer Engineering, “Gheorghe Asachi” Technical University of Iasi, Bd. Mangeron, No. 27, 700050 Iasi, Romania; marius.gavrilescu@academic.tuiasi.ro (M.G.); sabina-adriana.floria@academic.tuiasi.ro (S.-A.F.)

\* Correspondence: florin.leon@academic.tuiasi.ro (F.L.); catalin.lisa@academic.tuiasi.ro (C.L.)

**Abstract:** In the brick manufacturing industry, there is a growing concern among researchers to find solutions to reduce energy consumption. An industrial process for obtaining bricks was approached, with the manufacturing mix modified via the introduction of sunflower seed husks and sawdust. The process was analyzed with artificial intelligence tools, with the goal of minimizing the exhaust emissions of CO and CH<sub>4</sub>. Optimization algorithms inspired by human and virus behaviors were applied in this approach, which were associated with neural network models. A series of feed-forward neural networks have been developed, with 6 inputs corresponding to the working conditions, one or two intermediate layers and one output (CO or CH<sub>4</sub>, respectively). The results for ten biologically inspired algorithms and a search grid method were compared successfully within a single objective optimization procedure. It was established that by introducing 1.9% sunflower seed husks and 0.8% sawdust in the brick manufacturing mix, a minimum quantity of CH<sub>4</sub> emissions was obtained, while 0% sunflower seed husks and 0.5% sawdust were the minimum quantities for CO emissions.

**Keywords:** bricks; artificial neural networks; optimization algorithms; biologically inspired methods; modeling

**MSC:** 97R40; 90C26; 92Exx



**Citation:** Anton, C.; Leon, F.; Gavrilescu, M.; Drăgoi, E.-N.; Floria, S.-A.; Curteanu, S.; Lisa, C. Obtaining Bricks Using Silicon-Based Materials: Experiments, Modeling and Optimization with Artificial Intelligence Tools. *Mathematics* **2022**, *10*, 1891. <https://doi.org/10.3390/math10111891>

Academic Editor: Jüri Majak

Received: 19 April 2022

Accepted: 22 May 2022

Published: 31 May 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The process of obtaining burnt bricks involves a significant consumption of energy. Given the current context in which the cost of energy has become a critical issue, there is growing concern among researchers to find solutions to reduce energy consumption [1]. Most of the existing studies in the literature are carried out at the laboratory scale and involve the inclusion in the manufacturing mix of bricks containing waste products, such as agricultural waste [2] and textile sludge [3], as well as wastes resulting from the steel industry, containing mainly slag, dust, mud [4], cotton micro-waste [5], sawdust [6], coffee grounds [7], brick powders and residual ceramics [8–10], ash [11] and others. After obtaining the finished products at the laboratory scale, they are evaluated in terms of their mechanical properties, thermal conductivity and water absorption. Beshah and others [3] obtained very good compression resilience results (approximately 30 MPa) and energy savings of 26 and 50% when used in combination with 10% clay and 20% textile sludge to obtain burned bricks. At the same laboratory scale, Cultrone and others [6] established that the addition of sawdust to the composition of bricks in different proportions from 2.5 to 10% does not cause significant changes in the color of the bricks or in their mineralogy. Manni and others [7] also found that bricks do not change color even when coffee grounds

are added to their composition in proportions ranging from 10 to 30%. In the context of the significant reduction in clay sources involved in the manufacture of bricks, Khitab et al. [8] established that 27% of clay can be replaced with ceramic waste powder and waste brick powder without affecting the quality of the bricks. The same conclusion was reached by Wiriyikfu and others [9] when they used 20% waste brick powder. The introduction of coffee grounds to the composition of burnt bricks led to decreased thermal conductivity, resistance to bending and increased porosity and water absorption. Very good results were obtained for burnt bricks when using granite sludge at the laboratory scale by Hamid [10]. The optimum composition was 70% clay, 25% silica fume and 5% granite sludge waste at a firing temperature of 700 °C.

The production of burnt bricks by adding waste to the manufacturing mix must be assessed both in terms of preserving the properties of the finished products, which must be comparable to the classic ones, and taking into account other important factors. Among them, the distance from the place where the waste is located can be mentioned, which can lead to additional production costs and also to an increase in exhaust gases emissions in the furnace chimney. Sani and Nzihou evaluated the impacts of adding agricultural waste to the manufacturing mix on the thermal and mechanical properties of bricks at the laboratory scale, as well as on the net consumption of energy and gases emitted during the combustion process [1]. These authors established that the introduction of 4% olive core flour in the brick manufacturing mix led to a 36% reduction in energy consumption. They also found that CO<sub>2</sub> emissions increased by a maximum of 4.38% when using agricultural waste [1].

Analyzing the influence of adding waste to industrial-scale brick manufacturing mixes is a challenge for process engineers. These installations are designed and put into operation for a certain type of manufacturing mix and a certain type of combustion process. The tests performed at the industrial scale involve high costs and many hours of production in test mode. The use of various artificial intelligence tools in the processing and optimization of databases obtained at the industrial scale leads to significant savings of time and money. Information can be obtained to change the operating limits without affecting the performance during the manufacturing process and to improve production costs and environmental impact.

Different optimization algorithms inspired by human and virus behaviors were applied in this approach. These algorithms were deliberately chosen because they are inspired by organisms with very different degrees of complexity, i.e., very simple or very complex, but with complicated, unpredictable or unknown behaviors. By analyzing these organisms, new ideas can emerge for optimization research. Additionally, few applications have been addressed so far using these algorithms; therefore, another original contribution is to test them in the chemical engineering field, more precisely to optimize the process of obtaining bricks using silicon-based materials.

Mainly, these optimization algorithms can be organized into three categories: algorithms inspired by the human behaviors of both cooperation and competition, and algorithms inspired by virus behavior.

*Optimization algorithms inspired by the human behaviors of learning and cooperation.*

It is thought that human learning is based on three main strategies: random learning, individual learning and social learning. Accordingly, the following algorithms have been developed: Simplified Human Learning Optimization (SHLO) [12], Social Learning Optimization (SLO) [13] and Teaching–Learning-Based Optimization (TLBO) [14]. Not many improvements have been proposed so far for SHLO and SLO. Still, one can mention adaptive SHLO [15,16] and a combination between SLO, differential evolution and improved Social Cognitive Optimization [17]. On the other hand, TLBO is a simple, efficient algorithm [18], which has been used in several hybrid variants, e.g., including an error correction strategy and Cauchy distribution [19], genetic crossover and mutation strategies [20], a combination with a Bird Mating Optimizer [21] and Differential Learning [22].

*Optimization algorithms inspired by human competitive behavior.*

The specific algorithms based on human competition are sports-inspired algorithms, e.g., the Football Game Algorithm (FGA) [23,24] and Volleyball Premier League (VPL) algorithm [25], as well as the Imperialist Competitive Algorithm (ICA) [26], which was applied for a robust PID controller [27] and elsewhere in the engineering domain [28]. So far, FGA and VPL have not been the subject of a great number of studies. Among the proposed improvements, one can mention a modified VPL approach that uses the sine cosine algorithm [29] and the Multi-Objective Volleyball Premier League algorithm [30]. On the other hand, the ICA has been extensively studied, and many variants have been published, e.g., ICA combined with different chaotic maps [31], k-means clustering [32] and neural networks [33,34].

*Optimization algorithms inspired by virus behavior.*

Some examples that belong to this class are: Viral System (VS) [35], Virulence Optimization Algorithm (VOA1) [36] and Virus Colony Search (VCS) [37] approaches. VCS was applied to find the optimal placement of distributed generators with regard to a reliability assessment [38], and for unit commitment in smart grids with wind farms [39]. Another algorithm is the Virus Optimization Algorithm (VOA2) [40–42], for which only one modified variant [43] is available in the literature. In general, one cannot find many reported applications for these algorithms.

In most studies, these optimization algorithms have been tested on benchmark problems, with several applications in the field, especially for the TLBO algorithm [18]. After an in-depth literature study, no applications were found in the fields addressed in this paper or in related fields in which real industrial processes are considered.

In this approach, some optimization algorithms are applied to minimize the gas emissions in an industrial process, whereby burnt bricks are obtained through the introduction in the manufacturing mix of sunflower seed husks and sawdust. The novelty of this research paper is the study via simulation of the industrial process using optimization procedures, including neural networks and biologically inspired algorithms, to provide the percentage compositions of seed husks and sawdust and dry product mass, as well as the amounts of clay, ash and organic raw materials to be used in the manufacturing mix, so that the amounts of CO and CH<sub>4</sub> discharged to the furnace chimney are minimal. The combination of neural networks with algorithms inspired by human and virus behaviors has not been studied in the literature, which is another contribution of the present article.

## 2. Materials and Methods

### 2.1. Experimental Determinations

The experimental tests were performed in an industrial system used for obtaining bricks. Here, 100 batches of bricks were evaluated, for which the manufacturing mix was modified via the introduction of sunflower seed husks and sawdust in proportions ranging between 0 and 3.5%. The impact assessment of the addition of these auxiliary materials on the exhaust emissions resulting from the manufacturing process was performed by analyzing the exhaust gases in the furnace chimney with a Testo 350 flue gas analyzer. This was equipped with detection and measurement cells specific to the gases of interest (CO, NO and CH<sub>4</sub>) and metrologically calibrated. This analyzer provided CO measurement resolutions of 0.1 ppm and 1 ppm for NO and CH<sub>4</sub>.

The experimental determinations allowed the construction of a database containing information on the mass percentage composition of sunflower seed husks (SSH) and sawdust (S), dry product mass (DPM), the amount of clay (C), the amount of ash (A) and the amount of organic raw material (ORM) used, as well as the amounts of CO, NO and CH<sub>4</sub> discharged to the furnace chimney in the manufacturing process of 100 loads of bricks.

### 2.2. Modeling Methodology

Given the current energy crisis, process engineers are trying to maintain and improve productivity in industrial brick-making plants that are designed and put into operation for a certain type of manufacturing mix and a certain type of combustion process by adding

auxiliary materials such as sawdust and sunflower seed husks. The manufacturing mix must follow the same combustion curves due to the constructive characteristics of the installation, and the quantity of exhaust emissions in the furnace chimney must also be carefully monitored.

In this study, it is proposed to build neural models that make predictions about the changes in quantity of exhaust emissions when different percentages of auxiliary materials are introduced into the manufacturing mix, thereby helping to reduce the number of experimental tests with significant economic impacts, because these tests are carried out on large-capacity industrial plants (approximately 90,000 bricks per day) and involve significant consumption of materials. The predictions based on the constructed neural models have the advantage of reducing the consumption of materials, energy and time. Furthermore, the neural models are then introduced into an optimization procedure with the goal of minimizing the quantities of exhaust emissions.

The input data used for the neural models were: the mass percentage compositions of sawdust (S) and sunflower seed husks (SSH), dry product mass (DPM) expressed in kg and the amount of clay (C), the amount of ash (A) and the amount of organic raw material (ORM) expressed in tons. The output parameters considered were: the amounts of CO and CH<sub>4</sub> present in the flue gases in the furnace chimney. Neural networks with forward propagation with 6 inputs, one or two layers with 6 to 30 hidden neurons and an output were developed. The neural models with the best results in the training stage were tested in the validation stage on new series of experimental data (unseen data).

In the modeling stage, feed-forward neural networks with one or two hidden layers were tested because in many applications it has been shown that this type of neural model provides satisfactory results as a universal approximator. The method applied for the development of the network was trial and error, and in order to obtain the best topology, many variants were tested (neural networks with one or two hidden layers and different numbers of intermediate neurons), following the errors at the end of a relatively large number of training epochs.

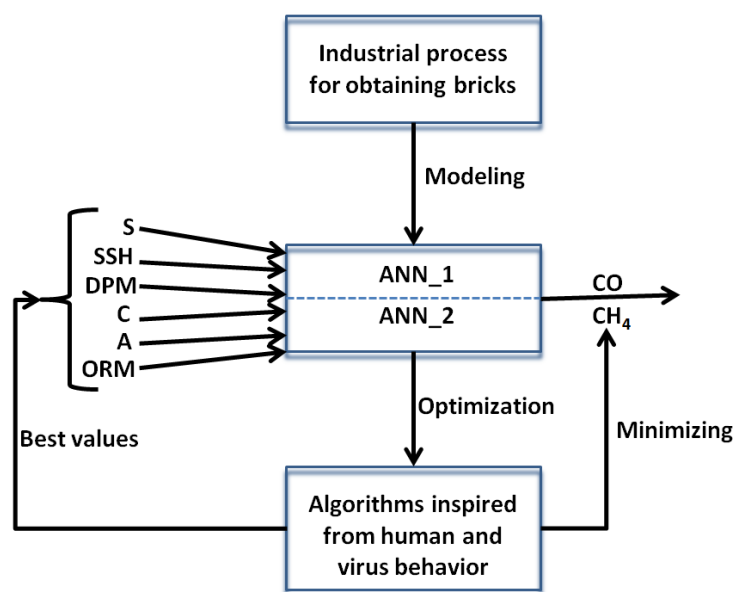
The feed-forward multilayer perceptron neural networks were built using the NeuroSolutions commercial simulator produced by the company NeuroDimension. The back-propagation training algorithm was applied, since it is the most commonly used algorithm for training neural networks, which aims to minimize the error via gradient descent. The transfer function used was the hyperbolic tangent. The experimental data were randomly ordered and divided into 87 instances used for training and 13 used for validation.

### 2.3. Optimization Methodology

The optimization problem aims to determine the percentage compositions of seed husks and sawdust and the amounts of clay, ash and organic raw materials to be used in the manufacturing mix so that the amounts of CO and CH<sub>4</sub> discharged to the furnace chimney are minimal. The previously determined neural models were included in the optimization procedure.

The methodology implemented for modeling and optimizing the industrial process for obtaining bricks is presented in Figure 1.

For single-objective optimization, considering separate problems for minimizing CO and CH<sub>4</sub>, the following algorithms inspired by the human and virus behavior were tested: Simple Human Learning Optimization Algorithm, Teaching–Learning-Based Optimization Algorithm, Social Learning Optimization, Football Game Algorithm, Volleyball Premier League Algorithm, Imperialist Competitive Algorithm, Viral System, Virulence Optimization Algorithm, Virus Colony Search and Virus Optimization Algorithm.



**Figure 1.** Methodology for modeling and optimizing experimental data (S = sawdust; SSH = sun-flower seed husks; DPM = dry product mass; C = the amount of clay; A = the amount of ash; ORM = the amount of organic raw materials).

All the algorithms have original implementations in C#. They were included in a unified optimization framework with a flexible architecture that allows the loose coupling of the modules implementing various algorithms and optimization problems [43]. New algorithms and new optimization problems can easily be added because they are based on predefined interfaces and the main program only uses *IProblem* and *IAlgorithm* objects, respectively. This is one of the few optimization frameworks available for .NET (more specifically, .NET framework 4.7.2). The resulting software is a Windows-based, self-contained application where the user can experiment with different combinations of algorithms, problems and parameter values.

Because these algorithms are less known and applied, a short description will be made for each of them. Additionally, we applied these algorithms and their performances were compared.

The Simple Human Learning Optimization (SHLO) algorithm [12] is inspired by a human learning model in which three learning mechanisms, similar to those of humans, are addressed: random learning, individual learning and social learning. The random learning operator is used to mimic random events that may occur during learning. At the beginning of the learning process, one can expect the process to be random because people do not have previous experience of the problem. Moreover, during learning, people cannot fully replicate previous experiences due to forgetting or partial knowledge of information. The individual learning operator is based on the principle that people avoid mistakes by using their own experiences. Therefore, this type of learning can improve an individual's performance. The SHLO algorithm mimics individual learning through the use of an individual knowledge database (IKD). Each individual  $i$  in the population has an  $IKD_i$  knowledge database that other individuals cannot access. However, in a social environment, people can effectively learn from shared experience [44,45]. The social learning operator attempts to eliminate the possible disadvantage of a slow and inefficient learning process of individual learning, especially for complicated problems. In the SHLO algorithm, a *social knowledge database* (SKD) is used to mimic learning from collective experience. The SKD database is used to store the best experience of the population. Although in SHLO there is no direct interaction between individuals, cooperative learning is indirectly achieved because (1) the SKD can be updated with better knowledge from better individuals and (2)



each individual in the population can access the SKD database. An updating operation is also used in SHLO to update the IKD and SKD during the search.

The Teaching–Learning-Based Optimization (TLBO) algorithm proposed by Rao et al. [14,46–48] is inspired by the process of teaching and learning. The basic principle of this process is the effect that a teacher has on the results obtained by a class of learners. Learning based on cooperation between learners is also considered. Therefore, the algorithm has two learning phases known as the teacher phase and the learner phase. TLBO is a population-based algorithm with the following analogies: the class of students represents the population, the marks of students for different subjects taught are correlated with the values for the variables of the solution in the optimization problem and the result of a student represents the fitness of the individual. In the teacher phase, the best student in the class is chosen in the algorithm to have the role of teacher. The goal of the teacher is to influence the marks of other learners in each subject. However, the teacher has a certain level of knowledge for each subject taught. The teacher’s level of knowledge in a particular subject is given by the difference between the teacher’s mark and the average mark of the class in that subject. When the teacher has a better level of knowledge, he or she has a more significant influence on the marks of each learner. The teacher always influences the marks of all learners in the class, but a learner accepts the new knowledge only if he or she has a better result than the previous one. In the second phase of TLBO, the learners attempt to improve their knowledge by interacting with one another. When some learner has a superior amount of knowledge, this can help a peer to improve their own knowledge. The interactions between learners are random, and each learner in the class has to interact with any other learner [49]. When two learners interact, their result identifies which one is better, while the difference between their marks influences the amount of knowledge transferred. The TLBO algorithm has been tested on many constrained and unconstrained optimization problems in various fields of engineering [50].

The Social Learning Optimization (SLO) algorithm [17] is based on the idea that intelligence in humans is determined both by genes and by social and cultural influence. It can be improved by learning and by acquiring new knowledge from participating in collective actions. When enough knowledge has been accumulated, a form of culture is established, and this can accelerate the development of intelligence. SLO includes three co-evolution spaces, organized by layers. The first (bottom) layer is the micro-space that supports individual evolution, which is in fact based on genetic evolution. SLO does not impose any constraints on the type of evolution and does not suggest a particular algorithm; for example, differential evolution can be used here. The next layer is the learning space that fosters imitation learning and observational learning by individuals, with the goal of improving their intelligence by learning from peers. The third (top) layer is the belief space, where knowledge extracted from the middle layer is conveyed. In this space, the accumulation of knowledge creates culture, which is further used to guide the individual genetic evolution in the micro-space. The purpose of the top layer is to simulate the phenomenon by which culture can accelerate the speed of evolution of human intelligence. In the upper co-evolution spaces, the SLO algorithm uses a library of knowledge points. These knowledge points are potential solutions, and better solutions can be found in their vicinity. The best individuals in the learning space replace the poor individuals in the belief space, and those in the belief space replace the poor individuals in the micro-space.

The Football Game Algorithm (FGA) [23] is based on an idealized version of the football game. The initial population defines the initial team of players on the field. Then, each player moves around his last position with a random walk procedure combined with a motion towards the ball. The ball is passed between players, and the players in better positions, i.e., with lower values of the objective function, are more likely to receive the ball. The part related to the coach represents the local search aspect of the algorithm. In order to increase the solution quality, a hypersphere is considered around the nearest best position in the vicinity of a player, whose radius decreases as the algorithm evolves. Members farther from good solutions are pushed toward the closest good positions. The coach may

also employ the change option to replace weaker players with other players around the closest good position, depending on the coach's memory.

The Volleyball Premier League (VPL) algorithm [25] is inspired by the interactions between volleyball teams during a competition season, along with the coach's decisions during a match. The members of a team are the players, the reserves and the coach. The representation of the solution has two segments called the active and passive parts. The active part, represented by the actual players of the team, includes six active players. The fitness of each solution is calculated based on this part. The passive part contains certain variables used in special rules such as the strategy for changing players, where a replacement can occupy the position of a player who was removed at the decision of the coach. In the VPL, the term "league" is used to represent the concept of a population. The teams play against each other and the winning team in each match is determined based on a power index. Furthermore, several strategies are employed for knowledge sharing, similar to crossovers, and for learning by exploiting the best solutions from the team population. Changes may also occur between active formations and passive reserves. Additionally, there is a mechanism for exchanging players between teams, as well as a mechanism for promoting and relegating teams from the league.

The Imperialist Competitive Algorithm (ICA) [26] is a metaheuristic inspired by socio-political behaviors, based on the phenomena of imperialism and colonialism [28]. An empire consists of an imperialist and one or more colonies. An imperialist is a developed country that attempts to expand its power by spreading its cultural values to other less developed countries called colonies. As the colonies adopt the cultural values of an imperialist, they are said to be assimilated or gradually absorbed by that imperialist. The existence of several empires implies the existence of competition for power between these empires. The ICA is a population-based algorithm with the following analogies: a country represents an individual, the socio-political elements of a country are the variables of the solution and the cost of a country represents the fitness of the individual. The set of countries, i.e., the population of individuals, is divided into several empires that compete for power. The power of an empire is determined using the cost of its imperialist and a fraction of the cost of its colonies. The following main steps can describe the ICA algorithm: assimilation, revolution, intra-empire competition and inter-empire competition. The assimilation operation is applied within each empire and imitates the effect of imperialist influence on the colonies. When assimilation is completed, the colonies of an empire will be closer to their imperialist. The revolution operation imitates the resistance of some colonies to be absorbed by their imperialists. Revolution in ICA is similar to the mutation operation of a genetic algorithm, and it randomly changes the positions of certain colonies. After the operations of assimilation and revolution, intra-empire competition follows. In this type of competition, the imperialist of an empire exchanges positions with one of the colonies if that colony has a better cost than that of the imperialist. Thus, a colony can become the new imperialist. In the last step, inter-empire competition imitates the competition between empires. Weaker empires will gradually lose their colonies to stronger empires until they collapse, and the ideal convergence of the search process is identified when only one empire remains.

The Viral System (VS) [35] is an optimization method based on an analogy of how a biological system reacts to a viral infection. The population of potential solutions is considered to be an organism composed of multiple cells. When a cell is infected, the virus starts replicating nucleus capsids inside the cell body. Once a threshold is reached and sufficient capsids have been generated, the virus is able to use the cell's DNA to replicate itself, at which point the cell is destroyed and the resulting viruses can spread to other cells from its neighborhood. The number of nucleus capsids increases over multiple iterations to a binomially distributed amount. Fitter solutions have higher upper thresholds for the number of capsids required for the infection to manifest. Furthermore, the organism may develop an antigenic response, in which case fitter cells may resist the infection altogether. The virus replication process may be either lytic, when the infection manifests suddenly,

or lysogenic, when the virus lies dormant until triggered. The duration of the lysogenic cycle is determined from binomially distributed variables, with healthier cells having longer thresholds in terms of the accumulated delay until the infection starts to manifest. Consequently, fitter cells will have a higher chance to resist the infection and longer delays until the manifestation or the replication of the virus, while less fit cells will be removed from the population more easily and will have a greater contribution to the spread of the infection.

The Virulence Optimization Algorithm (VOA) [36] is based on the behavior of viruses when attempting to spread through a host organism. The solution population is composed of both cell and virus instances, and the approach is based on the tendency of the viruses to seek regions from the problem space where there are more resources, i.e., fitter cells which offer more potential for replication. For each generation of potential solutions, the cell and virus populations are clustered together via k-means. After a mutation and crossover phase, the viruses migrate through the problem space by moving toward the best member of the best cluster. In an attempt to better generalize the algorithm, the viruses are translated toward the best solution part-way, along a percentage of the total distance, while the translation direction is deviated by a certain angle. Of the resulting virus population, the best viruses are cloned and the least fit ones are removed. Repeated clustering, migration and selection of the viruses eventually causes the population to converge towards forming a large, dense cluster containing the fittest solution.

Similar to the VOA algorithm, the Virus Colony Search (VCS) algorithm [37,43] simulates how viruses survive and propagate by attacking living cells. Although the terminologies used by the authors of the VOA and VCS are different, the general principles are the same. The major difference between the two algorithms is given by an additional step present in the VCS (virus diffusion) and by the way in which the common steps are implemented. The implementation of the VCS algorithm is based on a set of five rules: (a) two groups are simulated, the colony of viruses and the colony of host cells; (b) during the diffusion process, each virus randomly generates a new individual; (c) each virus infects a single cell; (d) the reproduction of each virus is based on the host cell destruction; (e) after the application of the immune response process, only some of the best individuals remain in the population. Thus, the steps taken by the VCS algorithm are:

1. *Initialization*. Similar to VOA, this step is designed to generate the initial population. In the VCS this is achieved through a random sampling of the search space;
2. *Virus diffusion*. This step simulates the process by which a virus searches for a host cell. The mechanism used in the VCS is based on the Gaussian Random Walk:

$$V'_{popi} = \text{Gaussian}\left(G_{best}^g, \tau\right) + \left(r_1 G_{best}^g - r_2 V_{popi}\right) \quad (1)$$

where  $i$  is the index of the current individual,  $V'_{popi}$  is the newly created individual,  $G_{best}^g$  is the best individual from a generation and  $r_1$  and  $r_2$  are random values generated in  $[0, 1]$  interval. For the Gaussian parameters, the standard deviation  $\tau$  is computed using the following relation:  $\log(g)/g \cdot (V_{popi} - G_{best}^g)$ ;

3. *Host cell infection*. After the cell has been attacked, the viruses begin to multiply using its resources. This process is simulated using the CMA-ES algorithm consisting of three sub-stages: the host cell colony changes relative to the individual represented by the arithmetic mean of the virus population; the best individuals in the virus population are identified and its center is calculated; the parameters for the calculation of the center and the applied covariance matrix are updated;
4. *Immune response*. At this stage, viruses evolve and the best-performing ones are selected for the next generation. Evolution is carried out on the basis of a performance order.

The Virus Optimization Algorithm (VOA) is also a population-based metaheuristic algorithm that simulates the attacking behavior of viruses [41]. The algorithm has three main steps: (1) initialization; (2) replication; (3) updating and selection.



1. *Initialization*. Like other metaheuristics, the initialization consists of generating the initial population of individuals. In this step, the control parameters are also set. In the initial version, a three-level factorial design is used to determine them [41].
2. *Replication*. In this step, new individuals are created. This is performed using the common and strong members of the population and is based on 2 sub-steps that include classification and replication. In the classification sub-step, the best individuals (which correspond to the strong member groups) and the rest (which correspond to the common members group) are identified. In the replication sub-step, new strong and common individuals, determined through Equations (2) and (3), are added to the population:

$$nv_{ij} = sv_{ij} \pm \frac{rand()}{intensity} sv_{ij} \quad (2)$$

$$nv_{ij} = cv_{ij} \pm rand() cv_{ij} \quad (3)$$

where  $nv$  indicates the new virus,  $sv$  refers to the strong virus and  $cv$  to the common virus. In the case of strong viruses, the replication is directed by an intensity parameter that is modified in the updating and selection step of the algorithm if the right conditions occur.

3. *Updating and selection*. In this case, two sub-steps are encountered: (i) updating of the exploitation mechanisms; (ii) population maintenance. In the first sub-step, the population convergence and the algorithm evolution are checked. If the average performance did not improve, the exploitation is intensified. In the population maintenance, the population undergoes a reduction phase, where if the number of viruses is higher than 1000 (value determined on the relation encountered in nature, where the average size of a virus is usually around 1000 times that of a cell), then the worst identified individuals are eliminated.

### 3. Results

#### 3.1. Neural Network Modeling

The evaluation of the topology of artificial neural networks (ANN) was performed by testing the performances of several networks with 6 inputs (the percentage compositions of sawdust and sunflower seed husks, dry product mass, the amount of clay, the amount of ash and the amount of organic raw material expressed in tons), with one or two intermediate layers with 6 to 30 hidden neurons and an output for predicting the amounts of CO and CH<sub>4</sub>, respectively, resulting in the gas discharge chimney. The selection criteria for the best topology were the mean square error (MSE), the coefficient of determination ( $r^2$ ) and the percent error  $E_p$  (%). The coding of the topology of the neural networks used was ANN( $m:n:p$ ), where  $m$  represents the number of nodes in the input layer,  $n$  is the number of neurons in the hidden layer and  $p$  is the number of neurons in the output layer.

In order to ensure the good generalization capability of the networks, they were trained while following the evolution of the MSE error on the validation set. Although the exact values differed from network to network, as an intuitive estimation, it was empirically found that after about 80,000 epochs for CO and about 50,000 epochs for CH<sub>4</sub>, the performance no longer improved and the network training was ended at that moment.

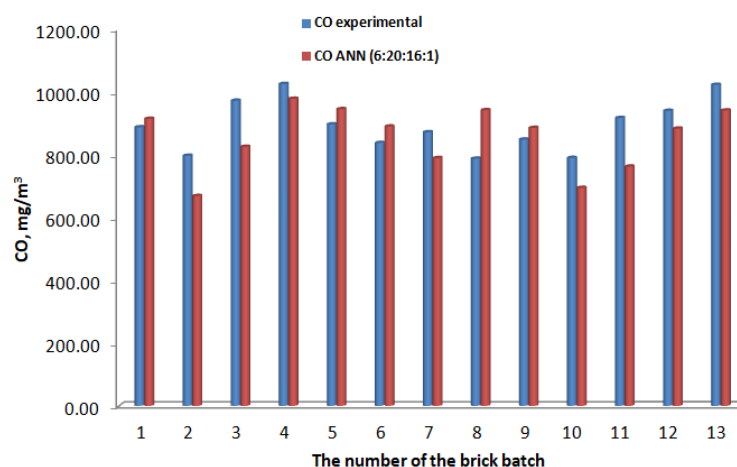
Table 1 presents the topologies of neural models constructed for predicting the amount of CO together with the selection criteria and the time required to obtain them. The best performances in the training stage were obtained with the ANN(6:20:16:1) model. The mean square deviation calculated for the training stage was  $\pm 34.4 \text{ mg/m}^3$ .

**Table 1.** Topology and performance of different ANNs constructed and trained for CO prediction.

No.	Topology	MSE	$r^2$	$E_p$ (%)	Time (min)
1.	ANN(6:6:1)	0.0388	0.870	4.99	4.19
2.	ANN(6:12:1)	0.0321	0.894	4.13	5.04
3.	ANN(6:18:1)	0.0276	0.910	3.49	5.28
4.	ANN(6:20:1)	0.0261	0.915	3.36	5.38
5.	ANN(6:24:1)	0.0292	0.905	3.68	5.45
6.	ANN(6:20:6:1)	0.0251	0.918	3.35	5.48
7.	ANN(6:20:12:1)	0.0148	0.953	2.57	5.55
8.	<b>ANN(6:20:16:1)</b>	<b>0.0128</b>	<b>0.959</b>	<b>2.19</b>	<b>5.49</b>
9.	ANN(6:20:18:1)	0.0242	0.921	3.21	5.66

(MSE = mean square error;  $r^2$  = coefficient of determination;  $E_p$  = percent error).

The results obtained with the ANN(6:20:16:1) model in the validation stage are shown in Figure 2.

**Figure 2.** Experimental values for CO compared to those obtained with the ANN(6:20:16:1) model.

The mean square deviation calculated by comparing the experimental data with those provided by the ANN (6:20:16:1) model was  $\pm 89.7$  mg/m<sup>3</sup>.

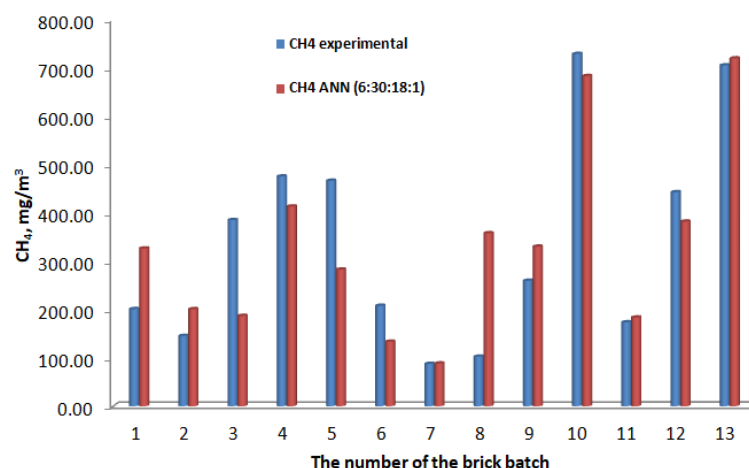
Neural models constructed to predict the amount of CH<sub>4</sub> resulting in the gas discharge chimney and their performance in the training stage are presented in Table 2. The results obtained indicate that the best performance for CH<sub>4</sub> is obtained with the ANN (6:30:18:1) model. The value of the mean square error calculated for CH<sub>4</sub> is  $\pm 43.8$  mg/m<sup>3</sup> and is slightly higher than that obtained for CO.

**Table 2.** Topology and performance of different ANNs constructed and trained for CH<sub>4</sub> prediction.

No.	Topology	MSE	$r^2$	$E_p$ (%)	Time (min)
1.	ANN(6:6:1)	0.0173	0.946	18.28	2.02
2.	ANN(6:14:1)	0.0122	0.962	15.38	2.25
3.	ANN(6:20:1)	0.0106	0.967	12.16	2.42
4.	ANN(6:24:1)	0.0105	0.968	11.48	2.55
5.	ANN(6:30:1)	0.0093	0.971	10.44	3.05
6.	ANN(6:36:1)	0.0100	0.970	11.29	3.32
7.	ANN(6:30:12:1)	0.0091	0.972	9.81	3.33
8.	<b>ANN(6:30:18:1)</b>	<b>0.0085</b>	<b>0.973</b>	<b>9.59</b>	<b>3.58</b>
9.	ANN(6:30:24:1)	0.0095	0.971	10.25	3.64

(MSE = mean square error;  $r^2$  = coefficient of determination;  $E_p$  = percent error).

Figure 3 compares the values obtained with the ANN (6:30:18:1) model in the validation stage with the experimental ones and shows a mean square deviation of  $\pm 108.8$  mg/m<sup>3</sup>.



**Figure 3.** Experimental values for CH<sub>4</sub> compared to those obtained with the ANN (6:30:18:1) model.

Neural models built to assess the impacts of the addition of auxiliary materials (sawdust and sunflower seed husks) on the amounts of exhaust gases in the furnace chimney in an industrial brick-making plant offer the possibility of making predictions that can help to reduce the number of test batches, with significant savings in time and money. Even if the mean square deviations are slightly larger than what other authors report in the literature [51–56], it must be taken into account that the modeled data are obtained experimentally in an industrial installation where the flow of gases discharged to the furnace chimney is of about 40,000 m<sup>3</sup>/h. In order to evaluate the influence of composition and density on the compressive strength of AAC lightweight brick, Zulkifli et al. [51] obtained a correlation coefficient of 0.984 in the training stage for the neural models with feed-forward backpropagation architecture and the Levenberg–Marquardt training algorithm. Recently, Shaban et al. [54] proposed an adaptive neuro-fuzzy inference system (ANFIS) with particle swarm optimization (PSO) to predict the compressive strength of concrete aggregate bricks (BACs). The correlation coefficient obtained in the training stage was 0.955. The importance of the study by Shaban et al. [54] derives from the fact that built models can help reduce demolition and construction (D&C) wastes and support efficient construction management. Members of our research group obtained correlation coefficients higher than 0.999 in the training stage for the neural models with feed-forward backpropagation architecture when modeling the thermal stability of some materials [56].

### 3.2. Single-Objective Optimization

Two optimization problems were formulated to determine the percentage compositions of seed husks and sawdust and the amounts of clay, ash and organic raw materials to be used in the manufacturing mix so that the amounts of CO and CH<sub>4</sub> discharged to the furnace chimney are minimized. Optimization algorithms inspired by human and virus behavior were used.

Thus, the optimization procedure has the following characteristic elements:

- The objective function is represented by the amounts of CO and CH<sub>4</sub> discharged to the furnace chimney (they are distinct problems, meaning the optimization process has a single objective);
- The decision variables are the inputs of the neural network, i.e., the percentage compositions of seed husks and sawdust, the dry product mass, as well as the amounts of clay, ash, and organic raw materials, respectively;
- The optimization process included the best models that had been previously determined, i.e., ANN (6:30:18:1) for CH<sub>4</sub> and ANN (6:20:16:1) for CO;
- The purpose of the optimization process was to determine the working conditions (the values of the five inputs of the neural networks) that lead to the minimum amount of exhaust gas.

The results obtained for optimizing the amount of CO discharged to the furnace chimney are presented in Table 3. The runtime is expressed in milliseconds (ms). In this table, for each algorithm (column 1) the best solution is specified (column 2), together with the corresponding values of the 6 decision variables and the value of the objective function, i.e., the amount of CO that was subjected to minimization. Additionally, in columns 4 and 5 the performance recorded by the algorithm for 100 simulations is presented.

**Table 3.** Results obtained with biologically inspired optimization algorithms for CO (mg/m<sup>3</sup>).

Optimization Algorithm	Best Solution ([Inputs] $\Rightarrow$ Output)	Performance (100 Simulations)	
Simple Human Learning Optimization Algorithm	[0.000, 0.523, 14.310, 729.676, 133.477, 8.564] $\Rightarrow$ 627.567	Runtime mean: 169.9 Runtime st. dev.: 33.783 No. evaluations mean: 100,100 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0
Teaching-Learning Based Optimization Algorithm	[0.000, 0.529, 14.305, 729.677, 133.477, 8.547] $\Rightarrow$ 627.567	Runtime mean: 723.3 Runtime st. dev.: 3.466 No. evaluations mean: 200,100 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0.002
Social Learning Optimization	[0.056, 0.565, 14.268, 375.746, 240.888, 1.403] $\Rightarrow$ 627.567	Runtime mean: 70 Runtime st. dev.: 11.747 No. evaluations: 16,030 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0
Football Game Algorithm	[0.000, 0.472, 14.611, 729.677, 133.477, 10.382] $\Rightarrow$ 627.567	Runtime mean: 55.3 Runtime st. dev.: 13.252 No. evaluations mean: 10,000 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.568 St. dev. solution: 0.001
Volleyball Premier League Algorithm	[0.578, 2.425, 13.460, 682.426, 86.488, 4.182] $\Rightarrow$ 627.582	Runtime mean: 1492.3 Runtime st. dev.: 13.077 No. evaluations mean: 333,938 No. evaluations st. dev.: 1358.68	Best solution: 627.582 Mean solution: 627.59 St. dev. solution: 0.007
Imperialist Competitive Algorithm	[0.000, 0.529, 14.305, 729.677, 133.477, 8.547] $\Rightarrow$ 627.567	Runtime mean: 305.7 Runtime st. dev.: 147.901 No. evaluations mean: 79,520 No. evaluations st. dev.: 39,005.3	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0.002
Viral System	[1.809, 0.268, 11.717, 723.748, 98.529, 5.989] $\Rightarrow$ 627.583	Runtime mean: 6.7 Runtime st. dev.: 4.562 No. evaluations mean: 175.1 No. evaluations st. dev.: 23.927	Best solution: 627.583 Mean solution: 627.583 St. dev. solution: 0
Virulence Optimization Algorithm	[3.673, 0.302, 12.475, 391.974, 183.682, 33.514] $\Rightarrow$ 627.571	Runtime mean: 442.3 Runtime st. dev.: 130.741 No. evaluations mean: 5662.5 No. evaluations st. dev.: 654.742	Best solution: 627.571 Mean solution: 627.682 St. dev. solution: 0.179
Virus Colony Search	[0.002, 0.595, 14.249, 729.497, 133.256, 8.370] $\Rightarrow$ 627.567	Runtime mean: 0.9 Runtime st. dev.: 2.7 No. evaluations mean: 33,110 No. evaluations st. dev.: 17,291.1	Best solution: 627.567 Mean solution: 627.568 St. dev. solution: 0.002
Virus Optimization Algorithm	[0.264, 0.494, 14.517, 715.361, 130.378, 11.070] $\Rightarrow$ 627.567	Runtime mean: 0.1 Runtime st. dev.: 0.03 No. evaluations mean: 6550.8 No. evaluations st. dev.: 802.434	Best solution: 627.567 Mean solution: 627.578 St. dev. solution: 0.008

The Simple Human Learning Optimization Algorithm, Teaching-Learning Based Optimization Algorithm, Social Learning Optimization, Football Game Algorithm, Imperialist Competitive Algorithm, Virus Colony Search and Virus Optimization Algorithm reach the same optimum, but the corresponding combinations of inputs slightly vary.

Excluding the algorithms with input values rather far from the input values provided by the majority, and taking the average of the input values provided by the majority of algorithms, it follows that the manufacturing mix composed of 0% sunflower seed husks, 0.5% sawdust, 14.4 kg of dry product, 729.6 tons of clay, 133.4 tons of ash and 8.9 tons of organic raw materials leads to the minimum amount of CO being discharged into the furnace chimney following the combustion process.

The results obtained for the second optimization problem, i.e., the input data necessary so that the flow of CH<sub>4</sub> discharged to the furnace chimney is minimal, are presented in Table 4. The best results were obtained with the Simple Human Learning Optimization Algorithm, Teaching–Learning-Based Optimization Algorithm and Imperialist Competitive Algorithm. The result that does not contain inputs close to their extreme values indicates that a minimum amount of CH<sub>4</sub> can be obtained if 1.9% sunflower seed husks, 0.8% sawdust, 14.9 kg of dry product, 510.8 tons of clay, 18.2 tons of ash and 26.2 tons of organic raw materials are used in the manufacturing mix.

**Table 4.** Results obtained with biologically inspired optimization algorithms for CH<sub>4</sub> (mg/m<sup>3</sup>).

Optimization Algorithm	Best Solution ([Inputs] ⇒ Output)	Performance (100 Simulations)	
Simple Human Learning Optimization Algorithm	[3.165, 1.707, 13.038, 428.125, 91.827, 14.837] ⇒ 5.847	Runtime mean: 176.6 Runtime st. dev.: 5. No. evaluations mean: 100,100 No. evaluations st. dev.: 0	Best solution: 5.847 Mean solution: 5.848 St. dev. solution: 0.001
Teaching–Learning Based Optimization Algorithm	[0.568, 3.500, 10.210, 62.521, 11.507, 28.932] ⇒ 5.847	Runtime mean: 1021.5 Runtime st. dev.: 6.454 No. evaluations mean: 200,100 No. evaluations st. dev.: 0	Best solution: 5.847 Mean solution: 5.847 St. dev. solution: 0
Social Learning Optimization	[2.827, 1.029, 17.527, 1175.909, 213.194, 53.036] ⇒ 5.848	Runtime mean: 82.2 Runtime st. dev.: 1.72 No. evaluations mean: 16,030 No. evaluations st. dev.: 0	Best solution: 5.848 Mean solution: 5.867 St. dev. solution: 0.025
Football Game Algorithm	[2.366, 3.076, 12.711, 436.729, 17.956, 28.932] ⇒ 5.861	Runtime mean: 61.6 Runtime st. dev.: 1.114 No. evaluations mean: 10,000 No. evaluations st. dev.: 0	Best solution: 5.861 Mean solution: 5.946 St. dev. solution: 0.086
Volleyball Premier League Algorithm	[3.500, 3.500, 18.670, 651.084, 133.477, 28.932] ⇒ 5.859	Runtime mean: 2208.4 Runtime st. dev.: 16.421 No. evaluations mean: 358,402 No. evaluations st. dev.: 1609.16	Best solution: 5.859 Mean solution: 5.907 St. dev. solution: 0.059
Imperialist Competitive Algorithm	[1.916, 0.758, 14.865, 510.807, 18.213, 26.201] ⇒ 5.847	Runtime mean: 188.8 Runtime st. dev.: 48.004 No. evaluations mean: 34,537 No. evaluations st. dev.: 9325.13	Best solution: 5.847 Mean solution: 8.386 St. dev. solution: 2.221
Viral System	[2.836, 3.123, 16.306, 290.118, 113.866, 24.425] ⇒ 6.253	Runtime mean: 5.7 Runtime st. dev.: 3.662 No. evaluations mean: 100.6 No. evaluations st. dev.: 0.8	Best solution: 6.253 Mean solution: 9.764 St. dev. solution: 4.481
Virulence Optimization Algorithm	[2.541, 0.700, 12.646, 261.597, 84.943, 19.883] ⇒ 6.464	Runtime mean: 3054 Runtime st. dev.: 594.252 No. evaluations mean: 18,614.9 No. evaluations st. dev.: 18,336.1	Best solution: 6.464 Mean solution: 13.97 St. dev. solution: 8.994

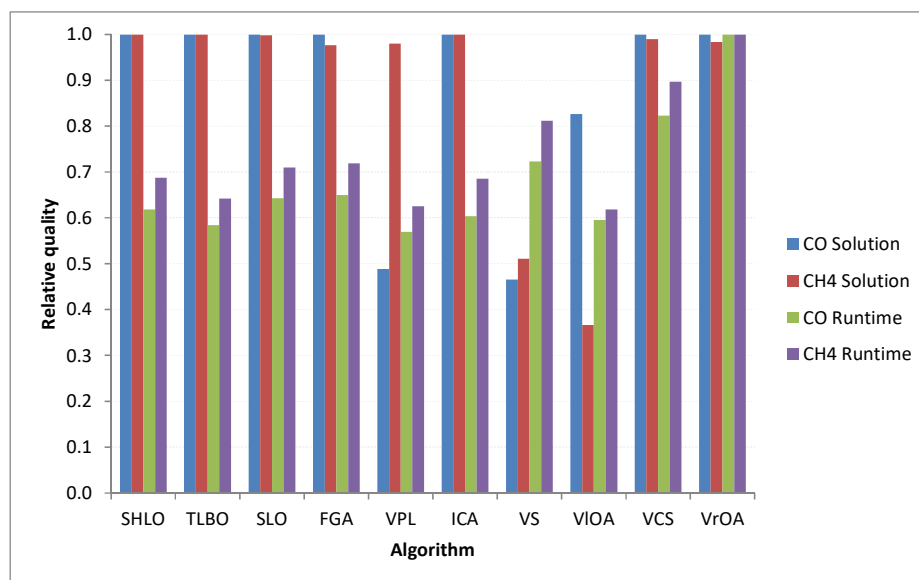


Table 4. Cont.

Optimization Algorithm	Best Solution ([Inputs] ⇒ Output)	Performance (100 Simulations)	
Virus Colony Search	[3.094, 3.426, 14.746, 176.624, 117.014, 27.545] ⇒ 5.853	Runtime mean: 1.2 Runtime st. dev.: 1.327 No. evaluations mean: 33,110 No. evaluations st. dev.: 17,291.1	Best solution: 5.853 Mean solution: 5.917 St. dev. solution: 0.059
Virus Optimization Algorithm	[3.149, 2.484, 14.615, 358.040, 112.675, 21.226] ⇒ 5.857	Runtime mean: 0.3 Runtime st. dev.: 0.458 No. evaluations mean: 4589 No. evaluations st. dev.: 2368.11	Best solution: 5.857 Mean solution: 6.095 St. dev. solution: 0.276

The results in Tables 3 and 4 depend on the values of the parameters for each algorithm. They are included in Appendix A (Table A1). Different values would naturally lead to different results.

Figure 4 presents an overview related to the performance of the optimization algorithms in terms of the best solution quality and execution time. However, the scales of the problems are very different. The best solutions are very close for all algorithms, while the runtimes can greatly vary. Therefore, some artificial performance indicators were computed independently for each component. For solution quality, these are based on the ratio between the solution provided by an algorithm and the minimum (best) solution found by all algorithms. The runtime indicators are based on the ratios between the logarithms of actual execution times. Thus, the graph gives more of an intuitive view of the solution quality and runtime. The best values are those close to 1; smaller values refer to lower-quality indicators.



**Figure 4.** Performance indicators for solution quality and execution time for the ten algorithms considered for the two optimization problems (SHLO = Simple Human Learning Optimization; TLBO = Teaching–Learning–Based Optimization; SLO = Social Learning Optimization; FGA = Football Game Algorithm; VPL = Volleyball Premier League; ICA = Imperialist Competitive Algorithm; VS = Viral System, VIOA = Virulence Optimization Algorithm; VCS = Virus Colony Search; VrOA = Virus Optimization Algorithm).

The following explanations based on exemplification will clarify the significance of Figure 4, highlighting the accuracy and usefulness of the results obtained, while also highlighting some of the advantages and disadvantages of the applied methods. Four

rectangles appear next to each algorithm—the first two reflect the quality of the results (CO and CH<sub>4</sub>) and the next two are for the execution time. The ideal case corresponds to the situation in which very good results would be obtained over a short execution time. Figure 4 shows that satisfactory results are accompanied by relatively long lead times. The best situation corresponds to the SLO algorithm, for which the result indicators equal 1, but those for the running time are at the values of 0.6–0.7. This is an acceptable compromise to obtain very good results by having a relatively long running time.

We also have good results reflected by values of 1 for CO and CH<sub>4</sub> provided by SHLO, TLBO and ICA, but with longer execution times (lower values for the following two bars). Other information provided by the figure shows unacceptable results for CO and CH<sub>4</sub> given by VPL, VS and VIOA and short execution times (values of 1 for bars 3 and 4) corresponding to the VrOA and VCS algorithms.

By analyzing Figure 4, the main conclusion is that most optimization results are satisfactory from a technological point of view.

### 3.3. Comparison with Classic Population-Based Algorithms

Although promising, the use of the algorithms applied so far is not yet widespread in the optimization community. This is why we also include a comparison with well-established algorithms such as the classic real-valued Genetic Algorithm (GA), Differential Evolution (DE) and Particle Swarm Optimization (PSO). The results obtained for different configurations for the two optimization problems addressed in our work are given in Tables 5 and 6.

**Table 5.** Results obtained with classic optimization algorithms for CO (mg/m<sup>3</sup>).

Optimization Algorithm	Best Solution ([Inputs] ⇒ Output)	Performance (100 Simulations)	
GA no. chromosomes = 50 no. generations = 100	[0.051, 0.590, 14.275, 727.648, 133.476, 8.707] ⇒ 627.567	Runtime mean: 428.596 Runtime st. dev.: 21.858 No. evaluations mean: 5000 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0
GA no. chromosomes = 50 no. generations = 500	[0.002, 0.548, 14.290, 729.461, 133.430, 8.561] ⇒ 627.567	Runtime mean: 2013.860 Runtime st. dev.: 104.082 No. evaluations mean: 25,000 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0.001
GA no. chromosomes = 100 no. generations = 1000	[0.000, 0.528, 14.308, 729.612, 133.474, 8.534] ⇒ 627.567	Runtime mean: 13856.667 Runtime st. dev.: 720.981 No. evaluations mean: 100,000 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0
DE no. chromosomes = 30 no. generations = 100	[0.083, 0.208, 13.480, 786.487, 217.001, 12.477] ⇒ 627.567	Runtime mean: 90.300 Runtime st. dev.: 4.628 No. evaluations mean: 3000 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0
DE no. chromosomes = 50 no. generations = 500	[0.323, 0.877, 13.383, 746.862, 123.897, 10.662] ⇒ 627.567	Runtime mean: 750.877 Runtime st. dev.: 38.574 No. evaluations mean: 25,000 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.567 St. dev. solution: 0
PSO no. particles = 30 no. iterations = 100	[0.000, 0.529, 14.305, 729.677, 133.477, 8.547] ⇒ 627.567	Runtime mean: 92.281 Runtime st. dev.: 4.679 No. evaluations mean: 3000 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.568 St. dev. solution: 0.001
PSO no. particles = 50 no. iterations = 1000	[0.000, 0.529, 14.305, 729.677, 133.477, 8.547] ⇒ 627.567	Runtime mean: 1415.965 Runtime st. dev.: 73.474 No. evaluations mean: 50,000 No. evaluations st. dev.: 0	Best solution: 627.567 Mean solution: 627.569 St. dev. solution: 0.002

(GA = genetic algorithm; DE = differential evolution; PSO = particle swarm optimization).

**Table 6.** Results obtained with classic optimization algorithms for CH4 (mg/m<sup>3</sup>).

Optimization Algorithm	Best Solution ([Inputs] $\Rightarrow$ Output)	Performance (100 Simulations)	
GA no. chromosomes = 50 no. generations = 100	[2.435, 1.747, 15.333, 436.009, 17.800, 27.772] $\Rightarrow$ 5.850	Runtime mean: 460.526 Runtime st. dev.: 23.858 No. evaluations mean: 5000 No. evaluations st. dev.: 0	Best solution: 5.850 Mean solution: 6.766 St. dev. solution: 1.600
GA no. chromosomes = 50 no. generations = 500	[2.478, 0.797, 12.962, 423.188, 18.531, 21.379] $\Rightarrow$ 5.847	Runtime mean: 2237.895 Runtime st. dev.: 116.928 No. evaluations mean: 25,000 No. evaluations st. dev.: 0	Best solution: 5.847 Mean solution: 5.858 St. dev. solution: 0.022
GA no. chromosomes = 100 no. generations = 1000	[2.155, 1.339, 13.413, 518.764, 18.863, 27.093] $\Rightarrow$ 5.847	Runtime mean: 14829.123 Runtime st. dev.: 773.507 No. evaluations mean: 100,000 No. evaluations st. dev.: 0	Best solution: 5.847 Mean solution: 5.847 St. dev. solution: 0.002
DE no. chromosomes = 30 no. generations = 100	[1.347, 2.970, 12.083, 418.237, 18.604, 25.541] $\Rightarrow$ 5.847	Runtime mean: 121.404 Runtime st. dev.: 6.960 No. evaluations mean: 3000 No. evaluations st. dev.: 0	Best solution: 5.847 Mean solution: 5.857 St. dev. solution: 0.013
DE no. chromosomes = 50 no. generations = 500	[1.562, 1.430, 12.652, 418.746, 17.961, 22.133] $\Rightarrow$ 5.847	Runtime mean: 986.667 Runtime st. dev.: 51.830 No. evaluations mean: 25,000 No. evaluations st. dev.: 0	Best solution: 5.847 Mean solution: 5.847 St. dev. solution: 0.001
PSO no. particles = 30 no. iterations = 100	[1.673, 3.500, 13.908, 462.521, 17.188, 27.932] $\Rightarrow$ 5.847	Runtime mean: 115.088 Runtime st. dev.: 5.640 No. evaluations mean: 3000 No. evaluations st. dev.: 0	Best solution: 5.847 Mean solution: 7.296 St. dev. solution: 4.335
PSO no. particles = 50 no. iterations = 1000	[1.448, 2.378, 18.670, 485.228, 17.477, 27.932] $\Rightarrow$ 5.847	Runtime mean: 1890.877 Runtime st. dev.: 98.453 No. evaluations mean: 50,000 No. evaluations st. dev.: 0	Best solution: 5.847 Mean solution: 5.847 St. dev. solution: 0

(GA = genetic algorithm; DE = differential evolution; PSO = particle swarm optimization).

The main parameters that influence the execution speed and solution quality are the population size and the number of generations or iterations. Other parameters have their own influence, but in order to keep the number of experiments manageable, they were given typical values. Moreover, in our case the difference in solution quality was not large, while the runtime mainly depends on the number of individuals generated throughout the execution of the algorithm. Thus, for GA we used tournament selection with two individuals (chromosomes), elitism with one individual, arithmetic crossover with 0.9 probability and mutation by gene resetting with 0.1 probability. For DE, the amplification factor was 0.8 and the crossover rate was 0.9. For PSO, we used the global best method, whereby the inertia weight was set to 0.729 and the cognitive and social coefficients were both set to 1.494.

The results show that the runtime greatly depends on the combination of parameters, although the algorithms obtain the optimal solutions most of the time. When the parameters considered have lower values, thereby decreasing the runtime, the solution may be suboptimal but still good.

However, some biologically inspired algorithms, e.g., Virus Colony Search or Virus Optimization Algorithm, are much faster, even by two orders of magnitude, and the quality of their results is still very good.

It must also be mentioned that the execution time of the algorithms can be further reduced by parallelizing some operations. Since some algorithms are easier to parallelize than others, in order to allow a fair comparison all of the implementations were sequential.

#### 4. Discussion

The main goal of this approach was to develop a complex, efficient methodology based on artificial intelligence tools for modeling and simulation. The methodology was applied on an industrial process, i.e., obtaining bricks from materials with different added ingredients, aiming to streamline the process. In this respect, artificial neural networks proved to be good models. Thus, for the prediction of CO emissions, the best model was ANN(6:20:16:1) with  $MSE = 0.0128$ ,  $r^2 = 0.959$  and  $E_p = 2.19\%$ ; while for  $CH_4$ , the ANN(6:30:18:1) model gave values of  $MSE = 0.0085$ ,  $r^2 = 0.973$  and  $E_p = 9.59\%$ . The neural models were integrated into an optimization procedure solved with different algorithms inspired by human behavior (learning, cooperation and competition) and virus behavior. For the same value of the objective function, the algorithms give different results for the 6 decision variables, which is a real advantage in practice, allowing the user to choose the most convenient solution.

A comparison between the performance of the optimization algorithms graphically illustrated in Figure 4 highlights the algorithms that obtained the best results regarding the CO and  $CH_4$  values (SLO, SHLO, TLBO, ICA) or the best (short) execution times (VrOA, VCS).

Most solutions are of the same quality as the solutions found by other commonly used population-based algorithms such as GA, DE and PSO.

Regarding the advantages and disadvantages of the applied algorithms, it must be emphasized that an optimization algorithm depends very much on the specifics of the approached problem. This was also the reason why in this approach we used 10 algorithms, tracking the quality of the results and execution times. Obviously, the accessibility of the method can be added to the aspects discussed. However, once the implementation is complete, in the version with the user-friendly interface, the handling of the program is no longer a disadvantage. In addition, it can be easily adapted to other processes and systems.

The optimization problems solved with algorithms inspired by the behavior of viruses indicate that the addition of sunflower seed husks and sawdust to the manufacturing mix contributes to increasing the amount of CO in the exhaust gases to the furnace chimney. However, the heat generated during the burning of sunflower seed husks and sawdust can supply the heat required during the manufacturing process, thereby reducing energy consumption. Ibrahim and others [57] in a laboratory-scale study established that the use of sawdust as an alternative to clay to produce bricks helps to reduce energy consumption. This aspect was also highlighted by Kurmus and Mohajerani [58] for bricks incorporating 1% waste (cigarette butts), whereby energy savings of at least 8% can be achieved. Sani and Nzihou established that the introduction of 4% olive core flour in the brick manufacturing mix leads to a 36% reduction in energy consumption [1]. Other recent studies [59,60] have shown the efficiency of using waste in the manufacture of bricks.

#### 5. Conclusions

The industrial process of obtaining bricks was studied here, first experimentally by evaluating the influence of adding sunflower seed husks and sawdust on the exhaust emissions resulting from the manufacturing process. Then, using experimental data sets, artificial intelligence tools were developed and applied for modeling and optimization actions.

In the modeling step, neural network models were determined and used to make predictions about the changes in quantity of exhaust emissions when different percentages of auxiliary materials were introduced into the manufacturing mix, there helping to reduce the number of experimental tests, having a significant economic impact.

Feed-forward neural networks were developed in various configurations, with 6 input variables (percentage compositions of sawdust and sunflower seed husks, dry product mass, amount of clay, amount of ash and amount of organic raw materials), one or two intermediate layers and a single output variable (amounts of CO and  $CH_4$ , respectively, present in the flue gases in the furnace chimney). The best models, selected using the mean

square error, coefficient of determination and percent error, were ANN(6:20:16:1) for CO prediction and ANN (6:30:18:1) for CH<sub>4</sub> prediction.

The best neural networks were included in an optimization procedure designed to minimize the gas emissions. Algorithms from three categories, inspired by the human behaviors of learning and cooperation, human competitive behavior and virus behavior, were applied comparatively to provide the best working conditions associated with the minimum energy consumption.

An overview related to the performance of the optimization algorithms in terms of the best solution quality and execution time was provided based on certain artificial performance indicators computed independently for each component. The main conclusion was that most optimization results were acceptable from a technological point of view.

The optimization results indicated that the addition of sunflower seed husks and sawdust to the manufacturing mix contributes to increasing the amount of CO in the exhaust gases in the furnace chimney. However, the heat generated during the burning of sunflower seed husks and sawdust can supply the heat required during the manufacturing process, thereby reducing the energy consumption.

In addition to these good results, what is important in this approach is the simulation methodology, including the neural networks and optimization biologically inspired algorithms, which provide satisfactory results. In addition, the methodology developed in this approach is generalizable and flexible, meaning it can be easily adapted to other processes, in association with different types of models.

**Author Contributions:** Conceptualization, S.C., C.L. and F.L.; methodology, S.C., C.L. and F.L.; software, F.L., M.G., E.-N.D. and S.-A.F.; validation, S.C., C.L. and F.L.; formal analysis, C.A.; investigation, C.A. and C.L.; resources, C.A. and F.L.; writing—original draft preparation, S.C., C.L., C.A., M.G., E.-N.D., S.-A.F. and F.L.; writing—review and editing, S.C. and C.L.; supervision, S.C.; funding acquisition, C.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by a publication grant of TUIASI, project number GI/P31/2021.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Appendix A

In this section, the parameter values of the algorithms whose results are included in Tables 3 and 4 are succinctly presented in Table A1.

**Table A1.** The parameters values of the algorithms considered for the optimization of CO and CH<sub>4</sub>.

Optimization Algorithm	Parameters and Values
Simple Human Learning Optimization	number of iterations = 100 population size = 100 number of bits = 32
Teaching–Learning-Based Optimization	number of iterations = 1000 population size = 100
Social Learning Optimization	population size = 30 number of generations = 250 amplification factor = 0.8 crossover rate = 0.9 lambda = 5 delta = 5
Football Game Algorithm	number of iterations = 200 number of players = 50 max strategies = 25 flv reduction = 0.85 hrlv reduction = 0.95



Table A1. Cont.

Optimization Algorithm	Parameters and Values
Volleyball Premier League	number of iterations = 100 fall rate = 0.15 transportation rate = 0.36 $g = 2$ league size = 10 number of players = 10
Imperialist Competitive Algorithm	number of iterations = 1000 population size = 100 number of empires = 10 $\alpha = 1.0$ $\beta = 1.5$ probability of revolution = 0.05 $\mu = 0.1$ $\zeta = 0.2$
Viral System	number of iterations = 1000 number of cells = 100 $plt = 0.5$ $pi = 0.4$ $pr = 0.7$ $pan = 0.2$ $lnr\ init = 10$ $lit\ init = 10$ max neighborhood distance = 10 number of max converge solutions = 3 convergence epsilon = 0.001
Virulence Optimization Algorithm	number of iterations = 100 number of cells = 100 number of initial viruses = 10 probability to mutate = 0.75 probability to recombine = 0.2 mutation sigma = 1.0 max angle offset = 1.57079633 number of best viruses from cluster = 3 number of best virus clones = 2 number of max converge solutions = 3 convergence epsilon = 0.001
Virus Colony Search	number of iterations = 100 population length = 20
Virus Optimization Algorithm	number of iterations = 10 population length = 20 strong members = 5 strong growth rate = 10 common growth rate = 2 intensity = 1

## References

1. Sani, R.; Nzihou, A. Production of clay ceramics using agricultural wastes: Study of properties, energy savings and environmental indicators. *Appl. Clay Sci.* **2017**, *146*, 106–114. [\[CrossRef\]](#)
2. Srisuwan, A.; Phonphuak, N.; Saengthong, C. Improvement of thermal insulating properties and porosity of fired clay bricks with addition of agricultural wastes. *Suranaree J. Sci. Technol.* **2018**, *25*, 49–58.
3. Beshah, D.A.; Tiruye, G.A.; Mekonnen, Y.S. Characterization and recycling of textile sludge for energy-efficient brick production in Ethiopia. *Environ. Sci. Pollut. Res.* **2021**, *28*, 16272–16281. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Vieira, C.M.F.; Amaral, L.F.; Monteiro, S.N. Recycling of Steelmaking Plant Wastes in Clay Bricks. In *Current Topics in the Utilization of Clay in Industrial and Medical Applications*; Zoveidavianpoor, M., Ed.; IntechOpen: London, UK, 2018; pp. 25–43.

5. Mebrahtom, T.; Haregeweyni, H.; Tamrat, T. Manufacturing of ecofriendly bricks using microdust cotton waste. *J. Eng.* **2021**, *2021*, 8815965.
6. Cultrone, G.; Aurrekoetxea, I.; Casado, C.; Arizzi, A. Sawdust recycling in the production of lightweight bricks: How the amount of additive and the firing temperature influence the physical properties of the bricks. *Constr. Build. Mater.* **2020**, *235*, 117436. [\[CrossRef\]](#)
7. Manni, A.; El Haddar, A.; El Hassani, I.-E.E.A.; El Bouari, A.; Sadik, C. Valorization of coffee waste with Moroccan clay to produce a porous red ceramics (class BIII). *Bol. Soc. Esp. Cerám. Vidr. V* **2019**, *58*, 211–220. [\[CrossRef\]](#)
8. Khitab, A.; Riaz, M.S.; Jalil, A.; Khan, R.B.N.; Anwar, W.; Khan, R.A.; Arshad, M.T.; Kirgiz, M.S.; Tariq, Z.; Tayyab, S. Manufacturing of clayey bricks by synergistic use of waste brick and ceramic powders as partial replacement of clay. *Sustainability* **2021**, *13*, 10214. [\[CrossRef\]](#)
9. Wiryikfu, N.C.; Fokam, C.B.; Kenmeugne, B.; Tchotang, T. The influence of burnt clay brick waste addition on recycled brick. *Int. J. Pavement Res. Technol.* **2021**, *14*, 482–486. [\[CrossRef\]](#)
10. Abdel Hamid, E.M. Investigation of using granite sludge waste and silica fume in clay bricks at different firing temperatures. *HBRC J.* **2021**, *17*, 123–136. [\[CrossRef\]](#)
11. Kadir, A.A.; Sarani, N.A. An overview of wastes recycling in fired clay bricks. *Int. J. Integr. Eng.* **2012**, *4*, 53–69.
12. Wang, L.; Ni, H.; Yang, R.; Fei, M.; Ye, W.A. Simple Human Learning Optimization Algorithm. In *Communications Computer and Information Science*; Book Series CCIS; Springer: Berlin/Heidelberg, Germany, 2014; Volume 462, pp. 56–65.
13. Leon, F.; Curteanu, S. *Regression Algorithm Based on Nearest Neighbors with Adaptive Distance Metrics and Multiple-Point Hill Climbing Training on a Lot of Noise-Affected Training*; Register of Works, No. 6573/9.10.2018; Romanian Copyright Office, ORDA: Bucharest, Romania, 2018.
14. Rao, R.V.; Savsani, V.J.; Vakharia, D.P. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Comput. Aided Des.* **2011**, *43*, 303–315. [\[CrossRef\]](#)
15. Wang, L.; Ni, H.; Yang, R.; Pardalos, P.M.; Du, X.; Fei, M. An adaptive simplified human learning optimization algorithm. *Inf. Sci.* **2015**, *320*, 126–139. [\[CrossRef\]](#)
16. Cao, J.; Yan, Z.; Xu, X.; He, G.; Huang, S. Optimal power flow calculation in AC/DC hybrid power system based on adaptive simplified human learning optimization algorithm. *J. Mod. Power Syst. Clean Energy* **2016**, *4*, 690–701. [\[CrossRef\]](#)
17. Liu, Z.-Z.; Chu, D.H.; Song, C.; Xue, X.; Lu, B.Y. Social learning optimization (SLO) algorithm paradigm and its application in QoS-aware cloud service composition. *Inf. Sci.* **2016**, *326*, 315–333. [\[CrossRef\]](#)
18. Venkata, R.R. Review of applications of TLBO algorithm and a tutorial for beginners to solve the unconstrained and constrained optimization problems. *Decis. Sci. Lett.* **2016**, *5*, 1–30. [\[CrossRef\]](#)
19. Zhai, Z.; Jia, G.; Wang, K. A novel teaching-learning-based optimization with error correction and cauchy distribution for path planning of unmanned air vehicle. *Comput. Intell. Neurosci.* **2018**, *3*, 5671709. [\[CrossRef\]](#)
20. Kumar, Y.; Dahiya, N.; Malik, S.; Khatri, S. A new variant of teaching learning based optimization algorithm for global optimization problems. *Informatica* **2019**, *43*, 65–75. [\[CrossRef\]](#)
21. Zhang, Q.; Yu, G.; Song, H. A hybrid bird mating optimizer algorithm with teaching-learning-based optimization for global numerical optimization. *Stat. Optim. Inf. Comput.* **2015**, *3*, 54–65. [\[CrossRef\]](#)
22. Zou, F.; Wang, L.; Chen, D.; Hei, X. An improved teaching-learning-based optimization with differential learning and its application. *Math. Probl. Eng.* **2015**, *2015*, 754562. [\[CrossRef\]](#)
23. Fadakar, E.; Ebrahimi, M. A New Metaheuristic Football Game Inspired Algorithm. In Proceedings of the 1st Conference on Swarm Intelligence and Evolutionary Computation, CSIEC 2016—Proceedings, Higher Education Complex of, Bam, Bam, Iran, 9–11 March 2016.
24. Djunaidi, A.V.; Juwono, C.P. Football game algorithm implementation on the capacitated vehicle routing problems. *Int. J. Comput. Algorithm* **2018**, *7*, 45–53.
25. Moghdani, R.; Salimifard, K. Volleyball premier league algorithm. *Appl. Soft Comput.* **2018**, *64*, 161–185. [\[CrossRef\]](#)
26. Atashpaz-Gargari, E.; Lucas, C. Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 4661–4667.
27. Shabani, H.; Vahidi, B.; Ebrahimpour, M. A robust PID controller based on imperialist competitive algorithm for load-frequency control of power systems. *ISA Trans.* **2013**, *52*, 88–95. [\[CrossRef\]](#) [\[PubMed\]](#)
28. Hosseni, S.; Khaled, A. A survey on the imperialist competitive algorithm metaheuristics: Implementation in engineering domain and directions for future research. *Appl. Soft Comput.* **2014**, *24*, 1078–1094. [\[CrossRef\]](#)
29. Moghdani, R.; Abd Elaziz, M.; Mohammadi, D.; Neggaz, N. An improved volleyball premier league algorithm based on sine cosine algorithm for global optimization problem. *Eng. Comput.* **2021**, *37*, 2633–2662. [\[CrossRef\]](#)
30. Moghdani, R.; Salimifard, K.; Demir, E.; Benyetton, A. Multi-objective bolleyball premier meague algorithm. *Knowl.-Based Syst.* **2020**, *196*, 105781. [\[CrossRef\]](#)
31. Talatahari, S.; Farahmand Azar, B.; Sheikholeslami, R.; Gandomi, A.H. Imperialist competitive algorithm combined with chaos for global optimization. *Commun. Nonlinear Sci. Numer. Simul.* **2012**, *17*, 1312–1319. [\[CrossRef\]](#)
32. Niknam, T.; Fard, E.T.; Pourjafarian, N.; Roust, A. An efficient hybrid algorithm based on modified imperialist competitive algorithm and K-means for data clustering. *Eng. Appl. Artif. Intell.* **2011**, *24*, 306–317. [\[CrossRef\]](#)
33. Ahmadi, M.A.; Ebadi, M.; Shokrollahi, A.; Majidi, J.M.S. Evolving artificial neural network and imperialist competitive algorithm for prediction oil flow rate of the reservoir. *Appl. Soft Comput.* **2013**, *13*, 1085–1098. [\[CrossRef\]](#)

34. Hajihassani, M.; Armaghani, D.J.; Marto, A.; Mohamad, E.T. Ground vibration prediction in quarry blasting through an artificial neural network optimized by imperialist competitive algorithm. *Bull. Eng. Geol. Environ.* **2015**, *4*, 873–886. [\[CrossRef\]](#)
35. Cortés, P.; García, J.M.; Muñuzuri, J.; Onieva, L. Viral systems: A new bio-inspired optimisation approach. *Comput. Oper. Res.* **2008**, *35*, 2840–2860. [\[CrossRef\]](#)
36. Jaderyan, M.; Khotanlou, H. Virulence optimization algorithm. *Appl. Soft Comput.* **2016**, *43*, 596–618. [\[CrossRef\]](#)
37. Li, M.D.; Zhao, H.; Weng, X.W.; Han, T. A novel nature-inspired algorithm for optimization: Virus colony search. *Adv. Eng. Softw.* **2016**, *92*, 65–88. [\[CrossRef\]](#)
38. Hosseini, S.J.D.; Moradian, M.; Shahinzadeh, H.; Ahmadi, S. Optimal placement of distributed generators with regard to reliability assessment using virus colony search algorithm. *Int. J. Renew. Energy Res.* **2018**, *8*, 714–723.
39. Shahinzadeh, H.; Gharehpetian, G.B.; Moazzami, M.; Moradi, J.; Hosseini, S.H. Unit Commitment in Smart Grids with Wind Farms Using Virus Colony Search Algorithm and Considering Adopted Bidding Strategy. In Proceedings of the 2017 Smart Grid Conference (SGC), Tehran, Iran, 20–21 December 2017; pp. 1–19.
40. Liang, Y.C.; Cuevas Juarez, J.R. Multilevel Image Thresholding Using Relative Entropy and Virus Optimization Algorithm. In Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2012 IEEE World Congress on Computational Intelligence, Brisbane, Australia, 10–15 June 2012; pp. 1–8.
41. Liang, Y.C.; Cuevas Juarez, J.R. A novel metaheuristic for continuous optimization problems: Virus optimization algorithm. *Eng. Optim.* **2015**, *48*, 73–93. [\[CrossRef\]](#)
42. Liang, Y.C.; Cuevas Juarez, J.R. Harmony Search and Virus Optimization Algorithm for Multi-Objective Combined Economic Energy Dispatching Problems. In Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2016, Vancouver, BC, Canada, 24–29 July 2016; pp. 3947–3954.
43. Lu, C.; Li, X.; Gao, L.; Liao, W.; Yi, J. An effective multi-objective discrete virus optimization algorithm for flexible job-shop scheduling problem with controllable processing times. *Comput. Ind. Eng.* **2017**, *104*, 156–174. [\[CrossRef\]](#)
44. Leon, F.; Curteanu, S. The Architecture of a Software Framework for Biologically-Inspired Optimization Algorithms. In Proceedings of the 10th International Conference on Intelligent Systems and Applications (INTELLI 2021), Nice, France, 18–22 July 2021; pp. 10–15.
45. Andrews, K.M.; Delahaye, B.L. Influences on knowledge processes in organizational learning: The psychosocial filter. *J. Manag. Stud.* **2002**, *37*, 797–810. [\[CrossRef\]](#)
46. McEvily, S.K.; Chakravarthy, B. The persistence of knowledge-based advantage: An empirical test for product performance and technological knowledge. *Strat. Manag. J.* **2002**, *23*, 285–305. [\[CrossRef\]](#)
47. Rao, R.V.; Savsani, V.J.; Vakharia, D.P. Teaching–learning-based optimization: An optimization method for continuous non-linear large scale problems. *Inf. Sci.* **2012**, *183*, 1–15. [\[CrossRef\]](#)
48. Rao, R.V.; Savsani, V.J.; Balic, J. Teaching–learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems. *Eng. Optim.* **2012**, *44*, 1447–1462. [\[CrossRef\]](#)
49. Rao, R.V.; Savsani, V.J. *Mechanical Design Optimization Using Advanced Optimization Techniques*; Springer Series in Advanced Manufacturing; Springer: London, UK, 2012.
50. Rao, R.V. *Teaching Learning Based Optimization Algorithm: And Its Engineering Applications*; Springer International Publishing: London, UK, 2016; Available online: [www.springer.com](http://www.springer.com) (accessed on 20 October 2021).
51. Zulkifli, D.S.; Absa, M.; Musyafa, A. Prediction of mechanical properties of light weight brick composition using artificial neural network on autoclaved aerated concrete. *Asian J. Appl. Sci.* **2017**, *5*, 556–567.
52. Goel, G.; Kalamdhad, A.S.; Agrawal, A. Parameter optimisation for producing fired bricks using organic solid wastes. *J. Clean. Prod.* **2018**, *205*, 836–844. [\[CrossRef\]](#)
53. Utomo, D.P.; Perdana, B.W.; Pamungkas, A.; Syaiin, M.; Adhitya, R.Y.; Munadhif, I.; Endrasmono, J.; Soeprijanto, A.; Soelistijono, R.T. CLC (Cellular Lightweight Concrete) Brick Making Process Using Neural Network and Extreme Learning Method Based on Microcontroller and Visual Studio. In Proceedings of the International Symposium on Electronics and Smart Devices, Yogyakarta, Indonesia, 17–19 October 2017; pp. 79–84.
54. Shaban, W.M.; Yang, J.; Elbaz, K.; Xie, J.; Li, L. Fuzzy-metaheuristic ensembles for predicting the compressive strength of brick aggregate concrete. *Resour. Conserv. Recycl.* **2021**, *169*, 105443. [\[CrossRef\]](#)
55. Apreutesei, N.A.; Tircoveanu, F.; Cantemir, A.; Bogdanici, C.; Lisa, C.; Curteanu, S.; Chiselita, D. Predictions of ocular changes caused by diabetes in glaucoma patients. *Comput. Methods Programs Biomed.* **2018**, *154*, 183–190. [\[CrossRef\]](#)
56. Lisa, G.; Wilson, D.A.; Curteanu, S.; Lisa, C.; Piuleac, C.G.; Bulacovschi, V. Ferrocene derivatives thermostability prediction using neural networks and genetic algorithms. *Thermochim. Acta* **2011**, *521*, 26–36. [\[CrossRef\]](#)
57. Ibrahim, J.-E.F.M.; Tihihi, M.; Gömze, L.A. Environmentally-friendly ceramic bricks made from zeolite-poor rock and sawdust. *Constr. Build. Mater.* **2021**, *297*, 123715. [\[CrossRef\]](#)
58. Kurmus, H.; Mohajerani, A. Energy savings, thermal conductivity, micro and macro structural analysis of fired clay bricks incorporating cigarette butts. *Constr. Build. Mater.* **2021**, *283*, 122755. [\[CrossRef\]](#)
59. Erdogmus, E.; Harja, M.; Gencel, O.; Sutcu, M.; Yaras, A. New construction materials synthesized from water treatment sludge and fired clay brick wastes. *J. Build. Eng.* **2021**, *42*, 102471. [\[CrossRef\]](#)
60. Harja, M.; Gencel, O.; Sari, A.; Sutcu, M.; Erdogmus, E.; Hekimoglu, G. Production and characterization of natural clay-free green building brick materials using water treatment sludge and oak wood ash. *Arch. Civ. Mech. Eng.* **2022**, *22*, 79. [\[CrossRef\]](#)