EAI Endorsed Transactions
on Smart Cities

Research Article EAI.EU

A Case Based Reasoning Coupling Multi-Criteria Decision Making with Learning and Optimization Intelligences: Application to Energy Consumption

Naomi Dassi Tchomté[1,*], Sohail Asghar[2], Nadeem Javaid[2], Paul Dayang[1], Duplex Elvis Houpa Danga[1], Dieudoné Lucien Bitom Oyono[3]

[1]Mathematics and Computer Science, Faculty of Science, University of Ngaoundéré, Cameroon
[2]Department of Computer Science, COMSATS University Islamabad, Pakistan
[3]Faculty of Agronomy and Agricultural Sciences, University of Dschang, Cameroon

Abstract

Optimization energy is a technique helpful to manage electricity consumption of home devices according to the electric system. CBR is used to predict consumption but lacks to be generic. This paper intends to design a more generic CBR approach by relying on various intelligences. The retrieve process includes four steps. The first step is weight evaluation of attributes based on AHP. The second step exploits an adapted cosine model for distance similarity. The third and fourth steps use k-Means and k-NN to identify the most similar cases. The reuse process is defined as a linear programming problem solved by PSO. During revise, an algorithm based on the reuse model and SVR, derives the revised solution. Experiments on a dataset of 1096 samples are made for forecasting energy electricity consumption. CBR revise process is 99.35% accurate, improving the reuse accuracy by 11%. The proposed architecture is a potential in energy management as well as for other prediction problems.

Received on 07 July 2019; accepted on 25 November 2019; published on 19 December 2019

Keywords: AHP, CBR, Forecasting, PSO, Supervised Learning, Support Vector Regression

doi:10.4108/eai.26-6-2018.162292

1. Introduction

Case-Based Reasoning (CBR) is a methodology which makes use of solutions of past problem cases to solve new problems [1] [2]. It includes four steps. The retrieve process aims to retrieve cases that are similar to the new case of the problem. The reuse process aims to reuse a solution that has been suggested by measuring the similarity in the retrieve phase. The revise process aims to adapt the solution to fit the new problem. The retain process is used to retain the new solution when it is confirmed. CBR is a promising artificial intelligence methodology as compared to machine learning techniques for two reasons [3] and [4]. First, it is able to adapt new data which is automatically stored into the knowledge base of the model and become part of the solution for predicting the future solution. Second, it is able to provide interesting performance independently to the dataset. On the contrary, supervised learning algorithms require large volume of data and need to be re-calculated when there is new data. It is not representative of the general trends of the modelling data. However, CBR lacks specifications on how these processes should be fulfilled. As a consequence, a CBR system can differ from another. CBR has been largely applied in energy consumption prediction to increase consumer awareness of the forecasted energy consumption. It is done to stimulate the shift their appliance consumption there by improving energy usage. Different approaches exist as using CBR to adjust the energy consumption of a house according to the actual state of consumption of appliances [5] and [6], predicting energy consumption

*Corresponding author. Email: naomitdassi@gmail.com

according to internal and external factors as well as people habits and agendas in the building [6], and selecting optimal algorithms exploited in different processes in resource scheduling [7] and [8]. However, these works lack to be generic because they only solve specific problems. The suggested aim is to make CBR functionalities more generic based on well structured data for energy forecasting problems. This study proposes **C**ase based reasoning **A**rchitecture **B**ased on machine **LE**arning (CABLE), a novel CBR approach coupling various intelligences to efficiently forecast a solution based on historical cases. It relies on a modified version of cosine for distance similarity and associates k-NN and k-Means for the retrieve phase. CABLE structures in the reuse, the determination of solution as an optimization problem from which the solution is determined using Particle Swarm Optimization(PSO). The retrieve process combines the reuse model and the SVM regression model based on a threshold. Experiments were made on real data samples about energy loads in buildings to assess the performance of forecasting using CABLE. This work is expected to be relevant to people willing to predict their energy consumption for making adequate decisions. And therefore to be further exploited as a support to determine appliances which increase consumptions.
The rest of the paper is organized as follows. The first section presents related works about predicting energy consumption using CBR. The second section presents different intelligent techniques required to build the proposed approach. The third section details the different components of the resulted architecture. The fourth section describes a case study exploiting CABLE to predict the electricity energy consumption in market buildings. The last section concludes and specifies future works.

2. Related works

Several works have been proposed to improve energy consumption using case-based reasoning. Platon *et al.* [7] study the accuracy of CBR and Artificial Neural Networks (ANN) models in predicting the hourly building electricity use. They investigate Artificial Neural Networks(ANN) and CBR methods for to the development of accurate models. This work uses Principal Component Analysis (PCA) to optimally reduce the number of variables by identifying the significant variables containing significant information in the dataset. Xiao *et al.* [9] and Shen *et al.* [10] provide a case-based reasoning model to build green buildings based on past knowledge and experience. They use text mining techniques to translate the experience in the format of text to be systematically descriptive. Faia *et al.* [8] propose a case-based reasoning scheme to determine optimal algorithms selected in previous cases to solve a new problem and the problem of optimal resource scheduling (ORS). They define requirements and related meta-heuristics to be applied in each process (retrieve, reuse, revise and retain). Faia *et al.* [5] propose an approach aiming to adjust the instant consumption of a house according to the required reduction values in each moment. This approach couples CBR and an intelligent house management to obtain suggested reduction values for house energy management. Authors analyze the history of previous cases of energy reduction in buildings, and using them to provide a suggestion on the ideal level of energy reduction that should be applied in the consumption of houses. González-Briones *et al.* [6], [11] aim to prevent temperature jump of the Heating, Ventilation and Air-Conditioning (HVAC) system by anonymously analyzing parameters influencing energy consumption. While, considering the presence of people in the building as well as forecasted in-door and out-door temperature fluctuations. They propose a multi-agent system to optimize energy usage relying on information related to the inhabitants captured by sensors and a CBR system, in conjunction with the current indoor and outdoor temperature together with the future temperature. Based on CBR, the system recommends the best possible energy saving techniques in the building and propose changes in the habits of the inhabitants. Minor and Max [12] investigate the replacement of traditional proportional-integral-derivative loops (PID) controller by a CBR system for the experience-based control of inert systems and demonstrated the feasibility of the approach for the reduction of energy wastage. Kadir *et al.* [13] show that energy consumption of a building can be predicted by focusing on the properties of the data and their pretreatment methods. Authors use automatic learning algorithms for the prediction.

Problem Statement The aforementioned works are limited in three points. Firstly, several cases with similarity between each others are required to reach good results and existing cases in the case base should cover every possible option or at least similar cases. These facts are difficult to verify, especially in early stages of the management and execution. Concerning this issue, this research supposes to have a consistent dataset as input. Secondly, the proposed CBR approaches are so far specific and problem-oriented. And the application of these models in other area, or slightly different problems, require significant changes. This research intends to provide functionalities in each CBR process so that they can be parameterized to every problem instance. Thirdly, the rules obtained in the revise stage are subjective. This research relies on supervised learning to produce knowledge to adapt the solution to fit to the problem's new case.

3. Background

This section presents different concepts required for the CBR proposal.

3.1. Case-based reasoning

The case-based reasoning is the result of the work of Roger Schank and his students at Yale University in 1980s [2]. The purpose of his work was based on dynamic memory and learning for the story set in natural language [14]. The first CBR system called CYRUS was developed by Janet Kololner using Schank's ideas [5]. This system was fundamentally focused on a question-and-answer system concerning the various trips and meetings of former Secretary of State Cyrus Vance [15]. Bruce Porter and his team later developed the PROTOS system which is a CBR system used in the classification of stains [16]. CBR is a knowledge-based system where the most suitable solution to a decision-making problem based on the most similar cases stored in a database [17]. Based on its methodology, case-based reasoning means using old cases from the case database to find the solution to a new case, adapting old solutions to new solution requests or using old cases of the case database to compare and or criticize new situations [18]. Case-based reasoning is based on artificial intelligence tools used by designers to build knowledge-based systems. It requires little effort in terms of acquiring bias judgements from experts. The choice of CBR will therefore provide the final system with memory and history. Figure 1 illustrates the process of CBR including four steps : Retrieve, Reuse, Revise and Retain.

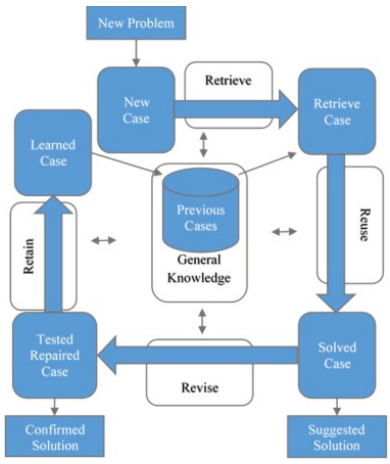

Figure 1. CBR steps [2]

Retrieve. This step extracts knowledge, relevant cases that solve the given target problem [19]. The retrieve phase is very important for CBR utility as shown in Figure 1. It finds similar cases to the new case in the case base and compares them with the new case. This stage has some classical methods like k-Nearest neighbours(k-NN) [19], or methods based on artificial intelligence like artificial neural networks [18] as well as genetic algorithms. Park *et al.* [20] demonstrate that statistical tools can also be exploited to recover best similar cases, by recovering the optimal number of neighbours according to their probability of similarity.

Reuse. As shown in Figure 1, the reuse phase is the second stage of CBR. This step is responsible for proposing a solution to the new problem [21]. It is easy when the similar case of the retrieve step is sufficiently similar to the new case because the solution of the new case will be just that of the similar case found. The following approaches can be exploited in this case [22].

Majority rule: The solution is the class c_i with the larger number of votes($maxcount$), i.e., the class that has solved the larger number of cases in the set of retrieved cases (RC).

$$s_q = arg_i maxcount(c_i, RC) \tag{1}$$

Probabilistic scheme: This procedure assigns probabilities to each possible outcome or class c_i. The class with the higher probability is selected.

$$p(s_q = c_i) = \frac{\sum_{j \in RC} Sim(P_q, P_j)\lambda^j}{\sum_{j \in RC} Sim(P_q, P_j)} \tag{2}$$

where $\lambda^j = 1$ if $s_j = c_i$, 0 otherwise.

Class-based scheme: This procedure determines, for each class c_i, the mean distance of the query case with the cases belonging to c_i, and takes the class with the minimum average ($minmeant$).

$$s_q = arg_i minmean_{c_i} Sim(p_q, p_j), \forall j, \tag{3}$$

The Reuse process becomes more difficult when the retrieve cases have significant differences. One must therefore be able to adapt the solutions of the retrieved cases to obtain new solutions. According to Kolodner [18], adaptation can be done in two ways: substitution and transformation. Garza & Maher used evolutionary algorithms in [23] and Suganthan used nature-inspired metaheuristics optimization algorithms in [24]. This step aims to map the solution from the previous cases to the target problem [25]. There are two major steps involved in adaptation: figuring out what it is needed to be adapted and applying the adaptation [21].

Revise. The revision process begins when the reuse phase is finished (see Figure 1). The purpose of this part is to evaluate the proposed solution in the reuse process that is normally done by simulations. It is noted that simulations often neglect important aspects of reality [5] because one can not always formulate all

the aspects that can occur in the real world [26].This step simulates the new solution in the real world and revises it if necessary [27].

Retain. This is the last phase and it keep the right solution in the case base. It is often good to record the specification of problem and the solution for future use. Veloso *et al.* [28] shows that it is recommended to store the final solution along with its knowledge structure. This knowledge is exploited to build the new solution to have a decision-making process. This step stores the resulting experience as a new case in memory after the solution has been successfully adapted to the target problem [29].

3.2. Analytic Hierarchy Process

Analytic Hierarchy Process (AHP) is a Multi-criteria Decision Making (MCDM) technique developed by Saaty [30]. It models an unstructured problem into a hierarchical structure of elements. The components of the hierarchy include the main goal: criteria that affect the overall goal and that can be used for selection of the final solution and sub-criteria that influence the main-criteria and the alternatives to solve the problem [31]. A pairwise comparison matrix is made by expert's opinions under the Saaty's preference scale as given in Table 1.

Table 1. Saaty's scales

Scale	Compare factor of i & j
1	Equally important
3	Weakly important
5	Strongly important
7	Very strongly important
9	Extremely important
2, 4, 6, 8	Intermediate value between adjacent scales

This work uses only the three first steps concerning the determination of weights [32]. The first step builds the pairwise comparison matrix.

$$A_{nn} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \quad (4)$$

Where n is the number of attributes from data, a_{ij} is the Saaty's comparison value between $attribute_i$ and $attribute_j$.

$$a_{ij} = 1 \; when \; i = j \; and \; a_{ji} = 1/a_{ij}.$$

The first step costs about $n^2 + \alpha, \alpha \in \mathbb{R}$. The second step constructs the normalized decision matrix.

$$c_{ij} = \frac{a_{ij}}{\sum_{j=1}^{n} a_{ij}}, \quad i, j = 1, \dots, n \quad (5)$$

This step requires $n^2 + \beta, \beta \in \mathbb{R}$. The third step determines the criteria weight vector w by averaging the entries on each row of the normalized matrix. This step requires about $n^2 + \sigma, \sigma \in \mathbb{R}$.

$$w_i = \frac{\sum_{j=1}^{n} c_{ij}}{n} \quad i = 1, 2, \dots, n \quad (6)$$

At the end, performing the vector of attributes weights is solvable in polynomial time proportionally to the square of the number of attributes, i.e $(n^2 + \alpha) + (n^2 + \beta) + (n^2 + \sigma) = 3n^2 + \alpha + \beta + \sigma, with \;\; \alpha, \beta, \sigma \in \mathbb{R}$.

3.3. Cosine similarity model

Cosine is the most popular Euclidean distance based metric for text classification [33] and [34]. It is determined as in Equation (7) and its value is between 0 and 1. More its value is closest to 1, more the both vectors are similar.

$$\cos ine(A, B) = \frac{\sum_{i=1}^{N} A_i \times B_i}{\sqrt{\sum_{i=1}^{N} A_i^2} \sqrt{\sum_{i=1}^{N} B_i^2}} \quad (7)$$

A and B are two vectors.

3.4. Particle Swarm Optimization

PSO is developed by Kennedy and Eberhart in 1995 [35]. According to [36] and [24] it is considered as the most popular nature-inspired metaheuristic optimization algorithm. The PSO algorithm is directed by personal experience ($Pbest$), overall experience ($Gbest$) as well as the current movement of the particles to decide the next positions in the search space. Additionally, experiences are accelerated by two factors C_1 and C_2, and two random numbers generated between $[0, 1]$ whereas the current movement is multiplied by an inertia factor w varying between $[w_{min}, w_{max}]$. The initial population (swarm) of size N and dimension D is denoted as $X = [X_1, X_2, ..., X_N]^T$, where '$T$' denotes the transpose operator. Each individual (particle) X_i $(i = 1, 2, ..., N)$ is given as $X_i = [X_{i,1}, X_{i,2}, ..., X_{i,D}]$. Also, the initial velocity of the population is denoted as $V = [V_1, V_2, ..., V_N]^T$. Thus, the velocity of each particle $X_i, (i = 1, 2, ..., N)$ is given by $V_i = [V_{i,1}, V_{i,2}, ..., V_{i,D}]$. The index i varies from 1 to N whereas the index j varies from 1 to D. The pseudo-code of PSO is given in Algorithm 1 [37].

Complexity This part evaluates the time complexity of PSO algorithm.

- Line 1 contributes for one (01) operation;
- Line 2 contributes for N operations;
- Line 3 contributes for one (01) operation;

Algorithm 1: Pseudocode of PSO.

1 Set parameter w_{max}, w_{min}, C_1 and C_2 of PSO ;
2 Initialize population of particles having positions X and velocities V ;
3 Set iteration $k = 1$;
4 Calculate fitness of particles $F_i^k = f(X_i^k), \forall i$ and find the index of the best particle b ;
5 Select $Pbest_i^k = X_i^k, \forall i$ and $Gbest^k = X_b^k$;
6 $w = w_{max} - k * (w_{max} - w_{min})/Maxite$;
7 Update velocity and position of particles ;
8 $V_{i,j}^{k+1} = w * V_{i,j}^k + C_1 * rand() * (Pbest_{i,j}^k - X_{i,j}^k) + C_2 * rand() * (Pbest_{i,j}^k - X_{i,j}^k)$; $\forall j$ and $\forall i$;
9 $X_{i,j}^{k+1} = X_{i,j}^k + V_{i,j}^{k+1}$; $\forall j$ and $\forall i$;
10 Evaluate fitness $F_i^{k+1} = f(X_i^{k+1}), \forall i$ and find the index of the best particle $b1$;
11 Update $Gbest$ of population ;
12 **if** $F_i^{k+1} < F_i^k$ **then**
13 $\quad Gbest^{k+1} = X_j^{k+1}$
14 $\quad$ else
15 $\quad Pbest_j^{k+1} = Pbest_j^k$
16 Update $Pbest$ of population ;
17 **if** $F_{b1}^{k+1} < F_b^k$ **then**
18 $\quad Gbest^{k+1} = Pbest_{b1}^{k+1}$ and $b = b1$
19 $\quad$ else
20 $\quad Gbest^{k+1} = Gbest^k$
21 **if** $k < Maxite$ **then**
22 $\quad k = k + 1$ and goto step 6
23 $\quad$ else
24 $\quad$ goto step 25
25 Print optimum solution as $Gbest^k$;

- Line 4 contributes for $1 + \log N$ operations with a binary search to look for index of the best particle;
- Line 5 contributes for two (02) operations;
- Line 6 - 7 perform N iterations in outer loop and use bubble sorting algorithm with six (06) operations;
- Line 8 performs N iterations in outer loop for nine (09) operations;
- Line 9 performs N iterations in outer loop for one (01) operation;
- Line 10 performs $\log N$ operations to look for the best element (binary search);
- Line 11 performs one (01) operation;
- Line 12 to line 15 perform one (01) operation;
- Line 16 performs one (01) operation;
- Line 17 to line 20 perform one (01) operation;
- Line 21 to line 24 perform one (01) operation;
- Line 25 performs one (01) operation.
- N.B: The goto operation indicates k iterations in a outer loop from line 6 to line 22.

In summary, $t(n) = 1 + N + 1 + (1 + \log N) + 1 + k(6 + N \log N + N(9 + 1) + (\log N + 1 + 1 + 1 + 1 + 1 + 1)) = N + \log N + k(10N + 12 + \log N + N \log N) + 4$. PSO includes two inner loops based on the population n, and one outer loop for iteration k (from line 6 to line 22). The complexity time $t(n)$ is linear in terms of k, which is in fact, the number of iterations of the particle movements. Higher, the number of iterations, higher is $t(n)$. The main computational constraint concerns evaluating complexity time related to the objective function (#obj). Therefore, $t(n) = N + \log N + k(10N + 12 + \log N + N \log N) + 4 + \#obj$

3.5. k-Nearest Neighbors

k-Nearest Neighbours (k-NN) is a supervised learning technique used for classification based on majority of k nearest neighbours [38] and [39]. It relies on Euclidean distance based metrics to determine minimum distance from the query instance to the training samples to determine the k-NN. The pseudocode of k-NN is given in Algorithm 2 [40] and is illustrated in Figure 2.

Algorithm 2: Pseudocode of k-NN

1: The data is loaded.
2: The value k is initialized.
3: For every point in the training data
4: DistanceVector <- Calculate the distance between testing data and each row of training data. Here, Euclidean distance or other metrics such as Chebyshev or cosine can be used.
5: vectorSorted <- Sorting DistanceVector in ascending order based on distance values;
6: topK <- Get top k rows from vectorSorted;
7: frequentClass <- Get the most frequent class of these rows;
8: Return frequentClass

3.6. k-Means Clustering

k-Means [41] is an unsupervised learning technique used for clustering. The following steps [42] are involved in k-Mean clustering and is illustrated in Figure 3.

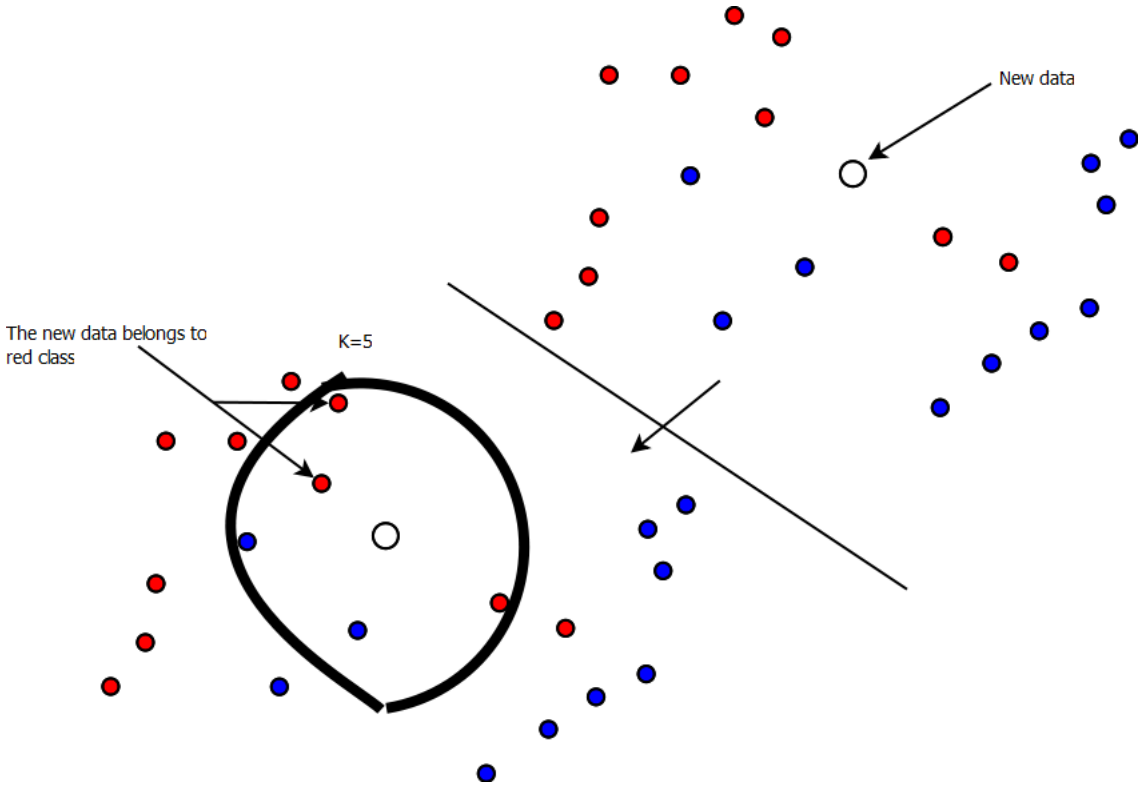

Figure 2. k-NN

- k observations are randomly selected (they are called centroids);
- Each instance in the dataset is affected to a cluster represented by nearest (Euclidean distance) centroid;
- Once the clusters are constituted, the centroid is updated for each cluster to the mean of all cluster members. And the cluster constitution reinitializes with new centroids. This step is repeated until the centroids become themselves mean of clusters. It means that the centroid become unchanged;
- The prediction of a test sample is done based on the nearest centroid calculated based on Euclidean distance.

3.7. Support Vector Machine

Support Vector Machine (SVM) [43] is a discriminative classifier defined by a separating hyperplane. The algorithm outputs an optimal hyperplane given labelled training data (supervised learning) to classify new data. In two dimensional space, this hyperplane is a line dividing a plane in two parts, where each class belongs to either side. As shown in Figure 4, points in star and in circle can be separated with an infinite number of hyperplanes. The best hyperplane maximizes the margin. The margin is the distance between the hyperplane and a few close points. These close points are the support vectors because they control the hyperplane.

The support vector classifier fails if the data is not linearly separable as in Figure 5. In this case, the aforesaid problem is solved through transformation of original feature space. It requires a nonlinear separation to a feature space that can be separated by a linear function. These approaches include linear kernel, polynomial kernel and gaussian kernel.

3.8. Regression

Regression refers a model which relates dependent attributes to independent attributes as shown in Figure 5. It is used to predict a variable in terms of other variables. Simple linear regression models relationships between a dependent variable and an independent variable. Multiple linear regression is an approach to model the relationship between a dependent variable and several independent variables. SVM Regression [44] is used within the scope of this work. The version of SVM for regression was proposed in 1996 by Vapnik *et al.* and baptised as Support Vector Regression (SVR) [45]. SVR is illustrated in Figure 6. It differs from SVM in some aspects. SVR deals with continuous classes by setting a margin of tolerance

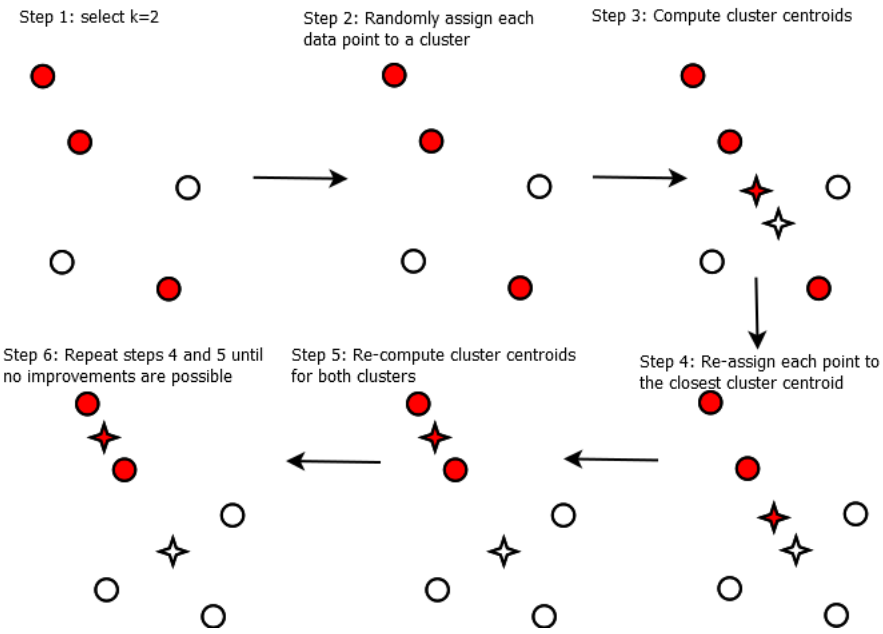

Figure 3. k-Means

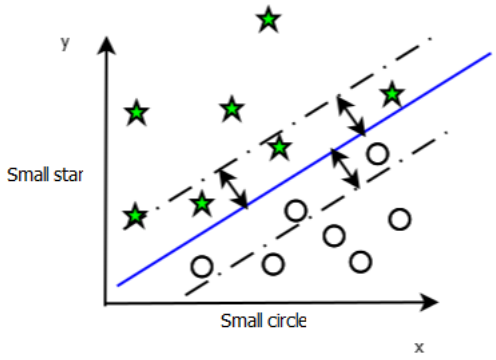

Figure 4. SVM

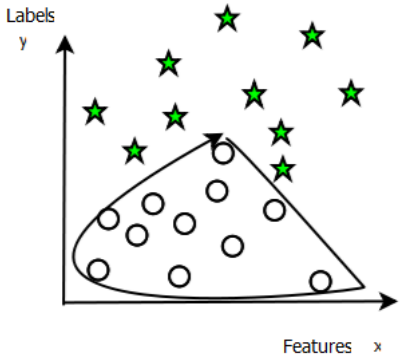

Figure 5. Non Linear Separation

(epsilon) in approximation to the SVM. The main idea remains the same: minimize and tolerate error, and determine the hyperplane which maximizes the margin. A SVR problem is formulated as [46] $min\frac{1}{2} \parallel w \parallel^2$, where $\parallel w \parallel^2$ is the magnitude of the normal vector to the surface being approximated.

Subject to,

$y_i - wx_i - b \leq \varepsilon$ and $wx_i + b - y_i \leq \varepsilon$

where x_i is the a training sample with target value y_i.

ε is a free parameter which is used as a thresold: true positives are predicted within a ε range.

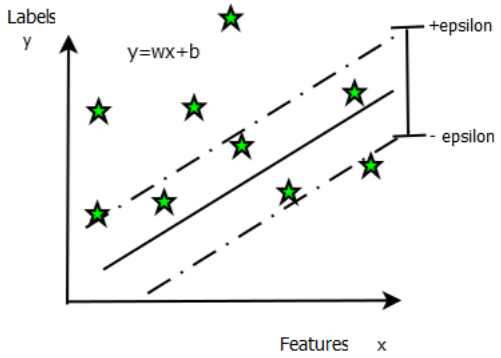

Figure 6. Linear SVR

4. Model development

The proposed approach, namely Case based reasoning Architecture based on machine LEarning (CABLE) is structured in three layers as shown in Figure 7. The retrieve layer relies on an adapted cosine similarity combined to AHP, k-NN and k-Means to determine the most similar cases to the query case. The reuse layer formulates the prediction problem as a linear programming problem on the similar cases and exploits an optimization technique to obtain the solution of the query case. The revise layer improves the detection efficiency based on a certain threshold, by using SVR. Methodologies for predicting a case solution are presented in the following sections.

4.1. Retrieve

The aim of retrieve is to provide the most similar cases to a new incoming case. There are some components to describe to achieve this mission.

Case base. The case base is represented as relational matrix depicted in Table 2 where attribute i refers to characteristics of the case and solution i is the solution value obtained for the problem case i. $Value_{ij}$ is the value of the attribute, which could be of type continuous, discrete, ordinal or nominal. The solution can be discrete or continuous.

Generation of attribute weights. Considering the importance of an attribute is relevant to improve performance of CBR systems [47–49]. Literature exploits [47] the whole costly process of AHP to determine the weights of attributes. On the contrary, some steps are considered to obtain the normalized vector of weights (see section 4.1. Each attribute refers to criteria. The pairwise comparisons are determined through interviews with domain experts under the Saaty's scale to compare attributes with each others. Consistency (lower to 0.1) is checked according to responses [30]. The details on similar case retrieval are described in the following section.

Determination of similar cases. This step takes as input a query case (CQ) and finds similar cases based on distance measures. At this stage, the query case only includes of attributes associated to weights with a solution to be determined. Three steps are performed to get similar cases. First, the similarity measure is computed between the query case and every case in the case base. This operation is realized with the adapted cosine, which considers the importance of attributes as proposed in Equation (8).

$$CosineMod(A,B) = \frac{\sum_{i=1}^{N} w_i A_i B_i}{\sqrt{\sum_{i=1}^{N} w_i A_i^2}\sqrt{\sum_{i=1}^{N} w_i B_i^2}} \quad (8)$$

Therefore, it takes each case with its pairs (attribute, weight) as determined by AHP. There are therefore as much similarity values as the number of cases. For instance, if the case base has hundred cases with solutions, then there will be hundred similarity measures. Each case is possibly similar to the query case. At this point, it is important to define the criteria to select cases considered as similar. We adopt to realize it by experimentally finding a value (α) varying in the range [0.1, 0.9], which provides the higher number of cases closer to CQ. The logic exploited here is that more experiences, a person has, more that person can make a decision on a future similar problem. It is justified by the fact that the person has enriched with enough knowledge. In this regards, more there are cases, much precise is the determination of the query case solution. Cases with a similarity greater than (α) are consequently taken.

Determination of the most similar cases. The first activity here is to transfer the similar cases (without their solutions) to k-Means to create cluster of cases that are similar to each other. The intention is to put together cases with similar characteristics trend. The algorithm k-NN is then taken into account to match the query case into the closer cluster. This cluster is determined based on calculation of Euclidean distance between previous cluster centroids and the query case. The selected cluster includes the most similar cases. For instance, k-Means provides the following output: 1, 1, 2, 2, 2, 2, 2, 2, 1, 1, 2, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 2, 2. The position represents the case index and the value is the cluster identifier to which the case belongs. In this example, case 1, case 2, case 9, case 10 belong to cluster 1. Case 3, case 4, case 5 belong to cluster 2. The next step consists to apply k-NN to determine in which cluster the query case fits better: it is a classification problem. In other words, this step

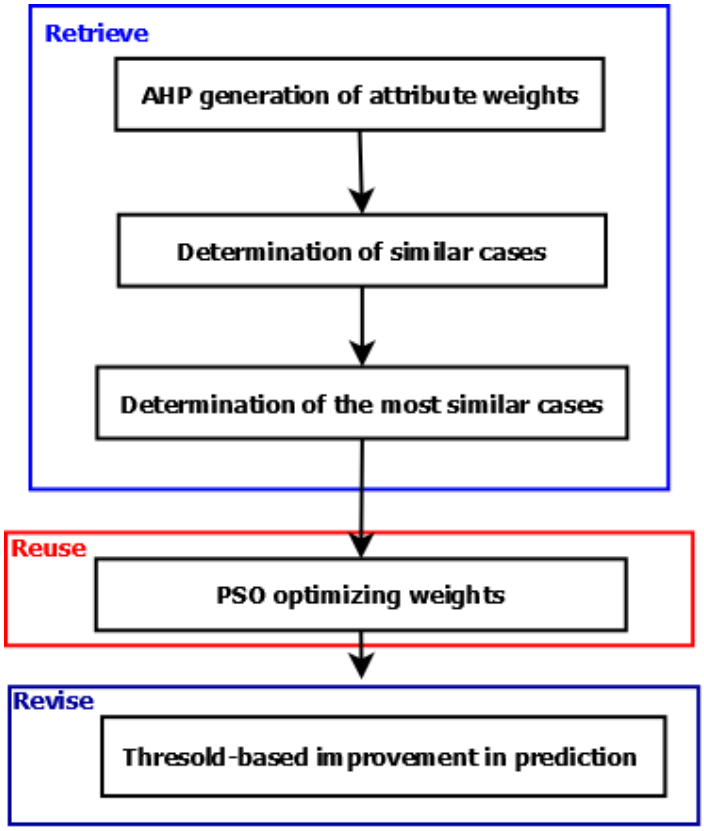

Figure 7. CABLE

Cases	Attributes					Solutions
	$Attribute_1$	$Attribute_2$	...	...	$Attribute_m$	
$Case_1$	$Value_{11}$	$Value_{12}$	...	...	$Value_{1m}$	$Solution_1$
$Case_2$	$Value_{21}$	$Value_{22}$	...	...	$Value_{2m}$	$Solution_2$
...	...	...	...	...	...	...
...	...	...	...	...	...	...
$Case_n$	$Value_{n1}$	$Value_{n2}$	...	...	$Value_{nm}$	$Solution_n$

Table 2. Case base

determines the centroid most closer to the query case, based on Euclidean distance. In case, several clusters are equally distant, the operation is restarted based on cosine distance. At the end, there are two outputs.

- The cluster with the centroid with the smaller Euclidean distance is selected;
- The cases belonging to that cluster are elected as the most similar cases for the problem.

4.2. Reuse

This stage aims to take the set of most similar cases to deduce the solution of the query case. For that, as described in section 3.1, there are three main possibilities. First, the attributes are modified; second, the problem structure is modified; and third, solution of the retrieved cases are together merged to the query case while changing some parameters such as attribute weights. The adopted method is the last option because the two first options modify either the original problem structure. This situation is represented as a model of relationships between different cases. That is, to determine a linear function between the solutions and case attributes, based on the actual structure of the case base. This problem can be modelled as a linear programming model as shown in Equation 9, where

- x_{ij} is the attribute i's value of the case j;
- w_i is the i^{th} AHP weight;
- R_j is the solution value of the case j.

$$R_j = w_1 x_{j1} + w_2 x_{j2} + w_3 x_{j3} + ... + w_i x_{ji} \quad (9)$$

Equation (10) is obtained after applying Equation (9) to the set of retrieved cases.

$$\begin{bmatrix} x_{11} & \dots & x_{1i} \\ \vdots & \ddots & \vdots \\ x_{j1} & \dots & x_{ij} \end{bmatrix} * \begin{bmatrix} w_1 \\ \vdots \\ w_i \end{bmatrix} = \begin{bmatrix} R_1 \\ \vdots \\ R_j \end{bmatrix} \quad (10)$$

Variables called errors in Equation (11) are the gap between the expected solution and the scalar of $case_i$ by the vector of weights.

$$\begin{bmatrix} x_{11} & \dots & x_{1i} \\ \vdots & \ddots & \vdots \\ x_{j1} & \dots & x_{ij} \end{bmatrix} * \begin{bmatrix} w_1 \\ \vdots \\ w_i \end{bmatrix} - \begin{bmatrix} R_1 \\ \vdots \\ R_j \end{bmatrix} = \begin{bmatrix} e_1 \\ \vdots \\ e_j \end{bmatrix} \quad (11)$$

The objective function in Equation (12) has been adopted from [5] with the main goal of minimizing the sum of square root of the error e_j. This function reaches a minimum value when the minimum combination of weights is obtained by solving each of the equations resulting from the matrix calculation.

$$minf(e) = \sqrt[2]{\sum_{i=1}^{maxj} (e_j^2)} \quad (12)$$

Particle Swarm Optimization (PSO) is used to find the optimal weights in Equation 12. The query case's solution is simply calculated by substituting the attribute's optimal weights and its attribute values. Evaluating the objective function costs about $\#attribute * \#cases$ for solving the linear system. $\#attribute$ is the number of attributes and $\#cases$ is the number of cases.

4.3. Revise

This activity is about improving the solution of the case query so that to minimize errors. It includes different steps.

Step 1 the case base C is splitted in possible x% training cases (training) and y% testing cases (testing) such that x belongs to 50%, 55%, 60%, 65%, 70%, 80%, 85%, 90%, 95% and y is 100% – x.
For each splitting, perform step 2 to step 4

Step 2 the SVR algorithm is applied to the training cases to obtain the model of regression $SVRModel$.

Step 3 for each testing case,
(a) Compute $\mu_{testingcase}$ such that

$$\mu_{testingcase} = 1 - \frac{mean(solution_{retrieve}(testingcase))}{solution_{reuse}(testingcase)} \quad (13)$$

$solution_{retrieve}(testingcase)$ provides solutions of the most similar cases for the testing case and $solution_{reuse}(testingcase)$ gives the solution of the testing case obtained in the reuse. The set of training cases is considered as the case base to obtain $solution_{retrieve}$ and $solution_{reuse}$.
(b) Determine μ_{min} as the smallest of $\mu_{testingcase}$.

Step 4 Determine the solution of the query case ($newcase$) such that

$$solution_{revise}(newCase) = \begin{cases} SVRModel(newcase) & \mu_{newcase} > \mu_{min} \\ solution1_{reuse}(newcase) & \text{otherwise} \end{cases}$$

$solution1_{reuse}(newcase)$ is computed on the original C to obtain the solution in the reuse. $SVRModel(newcase)$ predicts the solution of the query case based on the regression model.

5. Tests and validation

This research focused on proposing a reliable scheme to estimate solutions of new case problems. Various experiments using the collected cases were conducted to test the reliability of CABLE. All the scripts have been written using Matrix Laboratory (Matlab) version 2018a because this environment basically includes pre-integrated machine learning and optimization packages. The hardware consists to a machine with a processor Core i5 of 3.20 GHz and a memory of 8 GigaBytes with Microsoft Windows 8 as the hosted operating system.

5.1. Experiment design and process

The experiment design has chronological phases as illustrated in Figure 8. The first phase is the collection of the dataset, which includes different cases related to energy consumption. The second phase concerns the application of CABLE on the gathered data.

This phase includes application of different layer processes involved in CABLE. Experiments based on t-test are exploited to determine which reuse algorithm provides best prediction results as well as to examine which whether reuse or revise is precise in prediction. The final performance of CABLE is obtained by computing the prediction accuracy based on Root Mean Square Error (RMSE) since the target variable is continuous. The last phase aims to measure effects of varying k-NN and PSO parameters on the retrieve and reuse results. This phase is validated by studying consistency of the retrieved cases obtained for each parameter value.

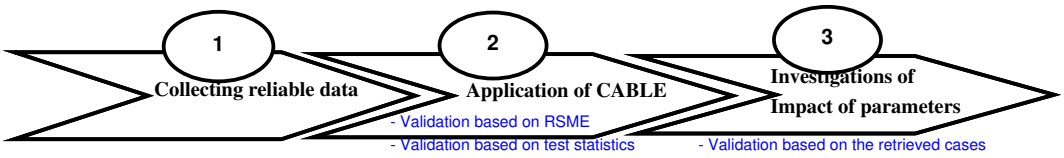

Figure 8. Experiment design

Table 3. Description of dataset attributes

Attribute	Description
DA_DEMAND(A1)	Day-Ahead Cleared Demand
RT _DEMAND(A2)	Real-Time Demand
DA_LMP(A3)	Locational Marginal Price One Day Ahead
DA_EC(A4)	Energy Part of the Day Ahead Price
DA_CC(A5)	Congestion Part of the Day Ahead Price
DA_MLC(A6)	Marginal Loss Part of the Day Ahead Price
RT_LMP(A7)	Real-Time Locational Marginal Price
RT_EC(A8)	Energy Component of Real-Time
RT_CC(A9)	Congestion Component of Real-Time
RT_MLC(A10)	Marginal Loss Component of Real-Time
DRY_BULB(A11)	Dry-bulb temperature in °F for the weather station
DEW_Point(A12)	Dewpoint temperature in °F for the weather station
Reg_Service_Price(A13)	Regulation Market Service clearing price
Reg_Capacity_Price (A14)	Regulation Market Capacity clearing price
System_Load(A15)	Real load corresponding to the energy demands

5.2. Collecting of data

The dataset is collected from of NYISO [50]. It has 1096 cases gathered day-by-day during two years from the first of January 2015 to the 31 December 2017. It concerns electrical consumption of different appliances in market buildings. This dataset has fourteen (14) attributes (A1 to A14) required for assessing energy load (A15), which is the target variable to forecast. They are described in Table 3.

Table 4 presents the types of attributes and some descriptive statistics such as the minimum, the maximum, the mean and the standard deviation of each attribute.

Figure 9 depicts the distribution of the target variable. This figure illustrates that System_Load follows a normal distribution with a skewness of 0.755.

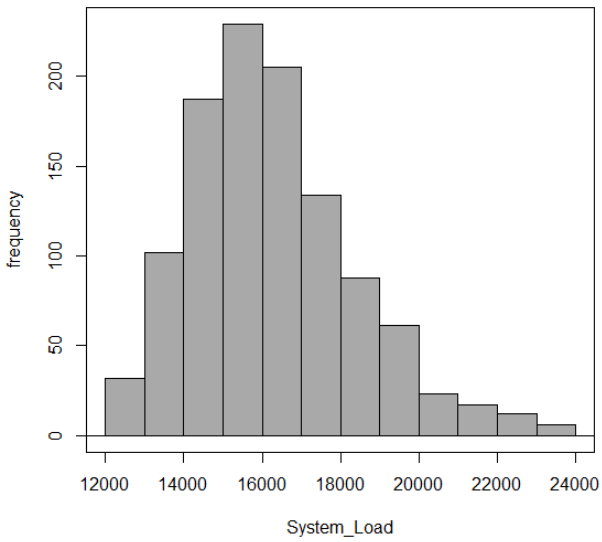

Figure 9. Distribution of the system_load variable

5.3. Application of CABLE

This experiment focused on applying CABLE methodologies (see section 4) on the collected dataset. The dataset has been splitted into training cases and testing cases such that training possibly belongs to 50%, 55%, 60%, 65%, 70%, 75%, 80%, 85%, 90%, 95% and testing is 100% – x. Different experiments take the training set as case base and the testing set includes the query cases to estimate solutions. The overall aim was to estimate the testing solutions through CABLE schemes and to evaluate prediction performance.

Retrieve. Table 5 presents an excerpt of the case base and a new case. As presented in section 4, this layer aims to find the most similar cases.

Attributes are weighting with AHP based on 10 surveyed experts in energy domain, who provided comparisons under Saaty's scale. Based on the answers from expert, the following attribute weights have been obtained based on the methodology 4.1. $w1 = 0.185, w2 = 0.11, w3 = 0.14, w4 = 0.090, w5 = 0.093,$

Table 4. Data types and descriptive statistics

	Type	Min	Max	Mean	Standard deviation
DA_CC	Continuous	-29	2	-0.221	1.436
DA_Demand	Continuous	11419	22928	15718.270	1987.782
DA_EC	Continuous	10	241	44.06204380	28.139
DA_LMP	Continuous	10	242.00	44.032	28.405
DA_MLC	Continuous	0.00	3.000	0.120	0.375
Dew_Point	Continuous	-17.00	74.00	38.917	19.592
Dry_Bulb	Continuous	1.00	90.00	53.683	19.399
Reg_Capacity_Price	Continuous	0.00	435.00	22.977	34.652
Reg_Service_Price	Continuous	0.00	10.00	0.498	1.750
RT_CC	Continuous	-15.00	23.00	0.029	1.758
RT_Demand	Continuous	11955	23633	16020.553	2082.461
RT_EC	Continuous	0.00	334.00	42.283	33.008
RT_LMP	Continuous	0.00	336.00	42.486	33.536
RT_MLC	Continuous	-1.00	3.00	0.108	0.360
System_Load	Continuous	12167	23970	16264.166	2112.897

Table 5. Case base representation

Attributes	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	Solution
Weights	0.185	0.11	0.14	0.090	0.093	0.057	0.048	0.056	0.049	0.044	0.043	0.036	0.024	0.015	
Case 1	16679	16391	63	62	0	1	45	45	0	0	30	9	6	0	16688
Case 2	16924	16649	57	56	0	1	22	22	0	0	31	15	7	0	16934
Case 3	15999	16624	53	52	0	0	58	58	0	1	27	25	12	1	16958
Case 4	14404	15765	41	40	0	0	77	76	0	1	41	40	21	1	16058
Case 5	18036	18557	83	82	0	1	103	111	0	1	19	30	25	3	18867
Case 6	18504	19259	82	82	0	1	130	130	0	1	15	10	28	0	19554
Case 7	19409	19883	115	114	0	1	127	126	0	1	10	20	24	0	20159
New case	18602	19901	81	82	0	1	99	114	0	1	18	25	26	2	

$w6 = 0.057, w7 = 0.048, w8 = 0.056, w9 = 0.049,$
$w10 = 0.044, w11 = 0.043, w12 = 0.036, w13 = 0.024,$
$w14 = 0, 015$

The adapted cosine provides the similarity results depicted in Table 6.

Table 6. Similarity results

Cases	Cosine values
case 1	0.63
case 2	0.71
case 3	0.52
case 4	0.48
case 5	0.900
case 6	0.911
case 7	0.902

Next, the optimal threshold is selected to provide the best similar cases. For that, the following steps are executed while looping from 0.1 to 0.9 with a pace of 0.1.

Step 1 k-Means is used then to form the clusters provided in Table 7.

Step 2 k-NN classifies the new case in different classes. For every alpha, k-NN selects the class {5,6,7} to assign the new case. Case 5, Case 6 and Case 7 are the best similar cases to be retrieved.

An experiment has been realized on different dataset splitting to examine similarity trends. For each splitting, the most similar cases are obtained as previously presented and then the average between their similarity measures is calculated. As presented in Table 8, the overall similarity scores ranged from approximately 70% to 91%. These similarities show that cases similar to the given case were extracted with a similarity of approximately 80.5%, which ensures the reliability of the retrieved cases for prediction. Furthermore, the subdivision (75%, 25%) seems to provide consistent retrieval.

Reuse. Experiments here were achieved based on the dataset subdivision (75%, 25%) because it provided best similarity values (see table 8), therefore optimal retrieved cases. Reuse can be performed by copying solution values from the retrieved cases or by adapting historical cases with a mathematical function. Two procedures has been used to identify which one fits the best to the problem, adapting and copying.

Table 7. k-Means clustering

Alpha	Cases	Clusters
0.1	empty	empty
0.2	empty	empty
0.3	empty	empty
0.4	Case 1, Case 2, Case 3, Case 4, Case 5, Case 6, Case 7	cluster 1={1,2} cluster 2={3,4} cluster 3={5, 6, 7}
0.5	Case 1, Case 2, Case 3, Case 5, Case 6, Case 7	cluster 1={1,2} cluster 2={3} cluster 3={5, 6, 7}
0.6	Case 1, Case 2, Case 5, Case 6, Case 7	cluster 1={1,2} cluster 2={5, 6, 7}
0.7	Case 2, Case 5, Case 6, Case 7	cluster 1={2} cluster 2={5, 6, 7}
0.8	Case 5, Case 6, Case 7	cluster 1={5, 6, 7}
0.9	Case 5, Case 6, Case 7	cluster 1={5, 6, 7}

Table 8. Experiments about dataset splitting vs. similarity measures

	Subdivisions (training, testing)									
	(50,50)	(55,45)	(60,40)	(65,35)	(70,30)	(75,25)	(80,20)	(85,15)	(90,10)	(95,5)
Similarity measures	0.79	0.75	0.86	0.84	0.905	**0.913**	0.77	0.85	0.79	0.70

Reusing by adapting The parameters of PSO are the following.

- Inertia weight: 0.9 to 0.4
- Acceleration factors (c_1 and c_2): 2 to 2.05
- Population size: 10 to 100
- Maximum iteration (*Maxite*): 100 to 500
- Initial velocity: 10% of position

In this appraoch, testing solutions are obtained when PSO converges. Figure 10 shows the gap between the solutions obtained after PSO convergence (graph in red) and the expected solutions (graph in blue).

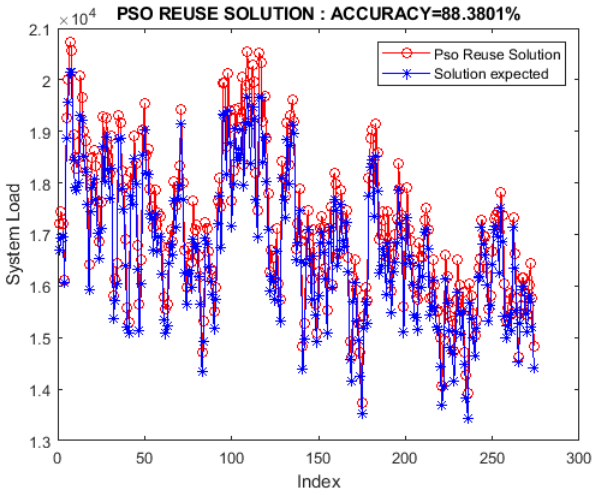

Figure 10. Reuse results

Figure 10 reveals some gaps between the expected solutions and predicted solutions. Moreover, CABLE is able to optimize testing solutions through PSO with an accuracy of 0.88.

Reusing by copying In this scheme, solutions of testing cases are determined by applying the Majority Rule (MR), the Probabilistic Method (PM) and the Class-based Method (CM) as described in section 3.1. Table 9 is an excerpt of the first nine case solutions calculated in each technique (according to section 3.1).

The first remark is that MR, PM and CM have values that are sometimes repeated across cases. MR has its majority value 15153 copied in all cases; PM has either values 15192 or 15409 that are repeated across cases and CM has either values 17480 or 16086 obtained in different cases. This situation is explained by the fact that these models compute solutions independently from the case base structure. It is therefore observed that these estimations can be efficient only in some cases.

The next section investigates which reuse model (MR, PM, CM or PSO) makes solution estimations more closest to the expected testing solutions (see the column "ExpectedSolution" in Table 10). To achieve this objective, ExpectedSolution is compared indepedently to MR, then to PM, then to CM and to PSO. The statistical technique t-Test is used to check if the means in each case are significantly different from each other.

t-Test The aim here is to examine, for example, whether ExpectedSolution and PSO values, are different on average, then PSO could not be chosen as the one getting closer to the expected values. The hypotheses are the following.

Ho(Null hypothesis) The sample means are equal or they do not have any significant difference. For instance, ExpectedSolution's mean and PSO's mean do not have any significant difference.

H1(Alternate hypothesis) The sample means are different or they have significant difference. For instance, ExpectedSolution and PSO means have significant difference.

Table 10 presents the results from the statistical test.

Concerning PSO, $P(T <= t) = 0.64 > 0.05$, Ho is therefore accepted. It means that PSO and ExpectedSolution samples have statistically the same means. PSO is appropriate to approximate ExpectedSolution. Concerning MR, the difference between MR and ExpectedSolution means is significant because we have evidence to reject the null hypothesis ($P(T <= t) = 5.3971E-50 < 0,05$). MR differs or moves away from ExpectedSolution. This conclusion is the same for PM and CM. In view of results, only the test between the ExpectedSolution and PSO reveals that they do not differ on average (from a statistical point of view), and that there is only 5% chance to fail in this assertion. In addition, mean values of PSO and ExpectedSolution are too close as well as their correlation equals 0.995, close to 1. PSO is therefore the model which is more appropriate to the expected values.

Revise. The revision of reuse solutions is made as described in section 4.3. First we experimented different dataset subdivisions as presented in Table 11. Then we investigated the subdivision which offers the high revise accuracy related to dataset.

The subdivision (75%, 25%) offers the best accuracy. Figure 11 illustrates results of revise (graph in red) compared to expected solutions (graph in blue). We observe that the accuracy has improved from 88.38% in the reuse to 99.35% in the revise. Therefore, some mispredicted testing solutions in reuse has been correctly classified in the revise. The SVM regression model effectively improves the determination of solutions.

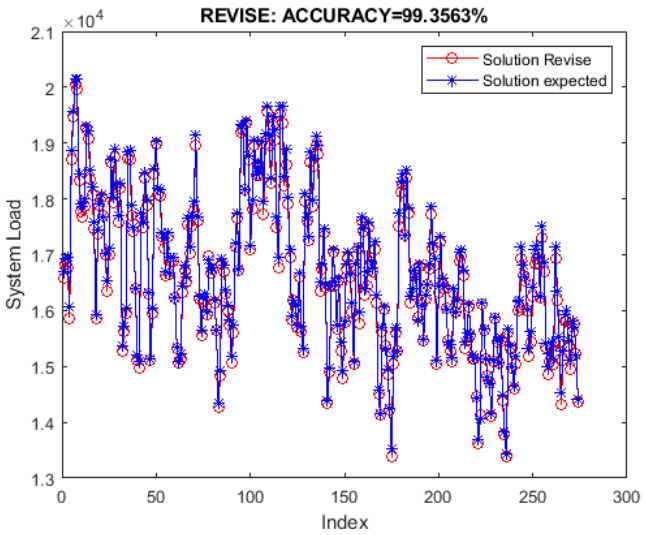

Figure 11. Revise result

We performed a statistical test to investigate whether the reuse approach is better than the revise approach.

t-Test The aim here is see, for example, whether ExpectedSolution and Reuse's estimated values, are statistically different on average, then Reuse could not be chosen as the one getting closer to the expected values. The hypotheses are the following.

Ho(Null hypothesis) The sample means are equal or they do not have any significant difference. For instance, ExpectedSolution's mean and Revise's mean do not have any significant difference.

H1(Alternate hypothesis) The sample means are different or they have significant difference. For instance, ExpectedSolution and Revise means have significant difference.

Table 12 presents the results from the statistical test.

Concerning Reuse, $P(T <= t) = 0.64 > 0.05$, Ho is therefore accepted. It means that PSO and ExpectedSolution samples have statistically the same means. PSO

Table 9. First nine case solutions

	ExpectedSolution	PSO	MR	PM	ClassBasedMethod
Case 1	16688	16798.60	15153	15192	17480
Case 2	16934	17032.85	15153	15192	17480
Case 3	16958	16833.86	15153	15409	16086
Case 4	16058	15809.71	15153	16911	16086
Case 5	18867	18835.71	15153	15409	16086
Case 6	19554	19572.17	15153	15409	16086
Case 7	20159	20278.36	15153	15409	16086
Case 8	20143	20127.56	15153	15409	16086
Case 9	18454	18510.97	15153	15192	16086

Table 10. t-Test: Expected solutions mean vs. PSO mean (resp. MR, PM, CM) , significance level = 0.05. ES: Expected solutions

t-Test: Paired Two Samples for Means, alpha = 0.05								
	ES	PSO	ES	MR	ES	PM	ES	CM
Mean	16714.4	16769.6	16714.4	15153	16714.4	15588.8	16714.4	16503.1
Variance	1957287.0	2051434.9	1957287.0	0	1957287.0	401251.5	1957287.0	409003.8
Observations	274	274	274	274	274	274	274	274
Pearson correlation	0.99	-	-	-	0.006	-	-0.10	-
Hypothesized Mean Difference	0	-	0	-	0	-	0	-
df	546	-	273	-	380	-	382	-
t-stat	-0.45	-	18.47	-	12.13	-	2.27	-
P(T<=t) one tail	0,32	-	2.69855E-50	-	3.72486E-29	-	0.01	-
T critical one tail	1.64	-	1.650454303	-	1.64	-	1.64	-
P(T<=t) two tail	0.64	-	5.3971E-50	-	7.44973E-29	-	0.02	-
T critical two tail	1.96	-	1.96	-	1.96	-	1.96	-

Table 11. Experiments about dataset splitting vs. revise accuracy

	Subdivisions (training, testing)									
	(50,50)	(55,45)	(60,40)	(65,35)	(70,30)	(75,25)	(80,20)	(85,15)	(90,10)	(95,5)
Regression accuracy	69.24	86.24	75.24	76.23	90.13	**99.35**	88.20	79.99	95.12	84.6

Table 12. t-Test: Expected solutions mean vs. Reuse mean (resp. Revise mean) significance level = 0.05. ES: Expected solutions

	t-Test: Paired Two Samples for Means, alpha = 0.05			
	ES	ReuseSolutions	ES	ReviseSolutions
Mean	16714.4	16769.64	16714.4	16713.18
Variance	1957287.0	2051434.904	1957287.0	1938675.069
Observations	274	274	274	274
Pearson correlation	0.995	-	0.999	-
Hypothesized Mean Difference	0	-	0	-
df	546	-	546	-
t-stat	-0.45	-	0,01	-
P(T<=t) one tail	0.32	-	0,49	-
T critical one tail	1.64	-	1,64	-
P(T<=t) two tail	0.64	-	0,991	-
T critical two tail	1,96	-	1,96	-

is appropriate to approximate ExpectedSolution. Concerning revise, $P(T <= t) = 0.991 > 0.05$, so the difference between both samples is not significant. In terms of mean, Revise is statistically equals to ExpectSolution. Revise is also appropriate to approximate the expected solutions with 5% chance to be mistaken. However,

to choose the model that best approximates, we refer to mean and correlation coefficient values. The results illustrate that ExpectedSolution and Revise means are very close (16713 and 16714). Additionnally, the correlation coefficient between ExpectedSolution and Revise is closest to 1 (0.9998) compared to ExpectedSolution and Reuse (0.995). The Revise model is therefore the model that best approximates the expected values.

5.4. Study of k-NN and PSO parameters on the determination of solutions

This step investigates the incidence of variability of PSO and k-NN parameters, on the determination of similar cases.

Number of neighbors Table 13 provides the similar cases for case 1 based on the number of neighbours in k-NN. For each k, the similar cases are ordered based on the distance considering the fourteen attributes. As k is reduced, the k-NN algorithm selects the most similar cases. The most similar cases are performed until the selection is done for $k = 1$. Table 13 reveals that case 57 is the neighbour closest to case 1 for $k = 1$. Figure 12 shows different case values for $k = 10$ and case 57 is found as the most closest to case 1 in terms of *system_Load* prediction. This experiment demonstrates that while varying k, k-NN remains consistent in the selection of most similar cases. This result means that the approach used to determine similar cases is effective.

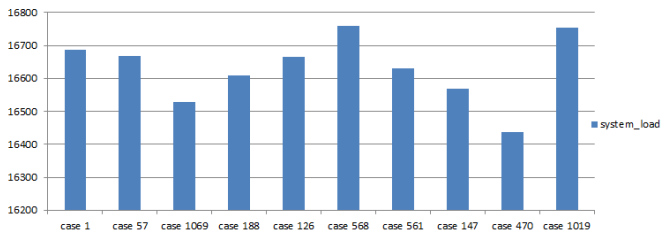

Figure 12. Similar cases for case 1

PSO executions Case 5 is investigated to analyze the relation between the number k of neighbours and the objective function value. Table 14 provides results. The estimation of case 5's solution with PSO is independent to the number of executions of PSO. It is justified by the fact that the estimated solution remains the same across k and the objective function is constant. A better value of the objective function is attained when k takes the value of 10.

5.5. Discussions

Determination of similar cases. Determination of similar cases. Existing works determine similar cases in the retrieve process based on Euclidean distance models by randomly selecting the number of similar cases. This work proposes a new scheme based not only on Euclidean distance but also on k-NN and k-Means for two reasons: the first reason is to take into account continuous solutions using k-Means to create clusters. The second reason is that the hyper-parameter k is very useful to set the number of similar cases taken here as the number of voted neighbors. However, the problem still exists. There should be a number of cases at the beginning. Therefore, a considerable amount of experiences should be gathered for the same problem. Otherwise, k-Means and k-NN will lose their performance. Collecting such information is sometimes very hard. This scenario is a limitation for CABLE since similar cases required for reuse and revise would possibly not exits due to dataset size.

Genericity. CABLE is composed of different components in different stages. In the first stage, CABLE includes AHP means of computing attribute weights. The latter takes as input a matrix of pair comparisons between attributes by experts. Therefore, it is configurable and not fixed to a particular problem. The other functionalities such as the adapted cosine, k-Means and k-NN are adaptable to the dataset taken as input. In the second stage, CABLE uses PSO, which computes the set of weights to apply to each attribute in other to find the solution of the new case. PSO depends on the dataset since it takes as input the subset of known cases from the dataset. In the third stage, CABLE includes an algorithm, which depends on the result in reuse and the SVM Regression. This algorithm is based on two parameters: the number of training cases (x), and the number of testing cases (y). The former is used to derive the regression model and the latter is used to obtain the threshold which indicates whether the reuse solution is applied or the reuse solution is revised. These overall descriptions demonstrate that CABLE is configurable according to the dataset provided as input. It is therefore, generic-dependent to the dataset provided for the forecasting problem to solve. More important it deals with continuous solutions since in the retrieve, k-Means is used to make cluster of retrieve cases before transferring it to k-NN. However, it is limited when one cannot provide enough data at start. In this case, an alternate learning algorithm must be determined, whose performance is not dependent to data size.

Association of intelligences. This research contributes in the way that CABLE federates various intelligences and computational approaches: an adapted version of cosine similarity, a multi-criteria decision making, (un-)supervised algorithms such as k-NN and k-Means, an optimization algorithm and the supervised SVM regression. The CBR architecture proposed so far is able to accurately forecast the electricity energy

Table 13. Similar cases selected for case 1

k	Index of cases
10	57, 1069, 188, 126, 568, 561, 147, 470, 1019
5	57, 1069, 188, 126
3	1, 57, 1069
1	57

Table 14. PSO executions for case 5

PSO executions	Measures	k=1	k=3	k=5	k=10
100	Minimum	18867	18758	18758	18624
	Mean	18867	18823.66	18813.6	18735
	Std	0	57.83	44.89	89.70
10	Minimum	18867	18758	18758	18624
	Mean	18867	18823.66	18813.6	18735
	Std	0	57.83	44.89	89.70

consumption in home buildings and can be generic to other type of dataset with non-discrete class. The efficiency of forecasting after the reuse and revision stages is due to the good measure of approach of similarity implemented in the retrieve stage and the optimization in the determination of solutions. Nonetheless, this process of association induces a complexity to deal with in case one does not have enough resources for computing. Since input data can be huge, approaches should be designed to avoid unuseful and repeated operations.

Performance. The CABLE's revise varies sizes of training and testing sets. The splitting (75%, 25%) has been found accurate in improving the determination of solutions since accuracy has improved by 11% from reuse to revise. Although, this process provides interesting results, it lacks to generalize results to the data structure. It is unsure that the performance remains acceptable if one changes the structure of the dataset: for instance, does the performance remain when the first 20% samples are used for training and the 80% remaining samples are used for testing –or when the first 20% samples are used for testing and 80% remaining for training.

6. Conclusion and perspectives

IoT supports the smart city paradigm in collecting information for making decisions. Smart city requires to connect appliances and equipments that consume energy. Optimization of energy consumption is therefore a concern for reduce monthly expenses. To address this problem, this paper proposed CABLE, a three-layer novel architecture of CBR with more generic functionalities. CABLE combines optimization and machine learning algorithms to forecast energy consumption in buildings. The first layer is made of two ways. The first extends the cosine model to consider attribute weighting while evaluating similarity between cases. The second applies k-Means then k-NN to find the most similar cases to the new case. The second layer designs the determination of the solution as a linear programming problem solved with PSO on the retrieved cases. The third layer improves the solution with the SVM regression model. Experiments realized with 1096 samples about electricity energy consumption revealed that the proposed scheme is accurate using the adaptation manner with PSO in the reuse. In fact, the revise approach is able to improve accuracy by 11% while reaching around 99% of prediction of expected solutions. The proposed architecture is a big potential to predict continuous solutions for the energy problems. The proposed architecture has two advantages. The first is that it optimizes the prediction of solution based on the association of machine learning and optimization processes. The second is its capacity to adapt to energy problems. As future work, we aim to design efficient retain process concerning integration and indexing of solved cases.

References

[1] D. B. Leake and D. B., *Case-based reasoning : experiences, lessons & future directions*. AAAI Press, 1996.

[2] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches," *AI communications*, vol. 7, no. 1, pp. 39–59, 1994.

[3] I. Watson, "An introduction to case-based reasoning," in *UK Workshop on Case-Based Reasoning*, pp. 1–16, 1995.

[4] A. J. C. B. de Sousa, "Application of knowledge acquisition methods in a case-based reasoning tool," 2018.

[5] R. Faia, T. Pinto, O. Abrishambaf, F. Fernandes, Z. Vale, and J. M. Corchado, "Case based reasoning with expert system and swarm intelligence to determine energy

reduction in buildings energy management," *Energy and Buildings*, vol. 155, pp. 269–281, nov 2017.
[6] A. González-Briones, J. Prieto, F. D. L. Prieta, E. Herrera-Viedma, and J. Corchado, "Energy Optimization Using a Case-Based Reasoning Strategy," *Sensors*, vol. 18, p. 865, mar 2018.
[7] R. Platon, V. R. Dehkordi, and J. Martel, "Hourly prediction of a building's electricity consumption using case-based reasoning, artificial neural networks and principal component analysis," *Energy and Buildings*, vol. 92, pp. 10–18, apr 2015.
[8] R. Faia, T. Pinto, T. Sousa, Z. A. Vale, and J. M. Corchado, "Automatic Selection of Optimization Algorithms for Energy Resource Scheduling using a Case-Based Reasoning System," *undefined*, 2017.
[9] X. Xiao, M. Skitmore, and X. Hu, "Case-based Reasoning and Text Mining for Green Building Decision Making," *Energy Procedia*, vol. 111, pp. 417–425, mar 2017.
[10] L. Shen, H. Yan, H. Fan, Y. Wu, and Y. Zhang, "An integrated system of text mining technique and case-based reasoning (tm-cbr) for supporting green building design," *Building and Environment*, vol. 124, pp. 388–401, 2017.
[11] A. González-Briones, J. Prieto, F. D. L. Prieta, E. Herrera-Viedma, and J. M. Corchado, "EnerVMAS: Virtual Agent Organizations to Optimize Energy Consumption Using Intelligent Temperature Calibration," *Sensors*, vol. 18, p. 865, 2018.
[12] M. Minor and L. Marx, "Case-Based Reasoning for Inert Systems in Building Energy Management," 2017.
[13] K. Amasyali and N. M. El-Gohary, "A review of data-driven building energy consumption prediction studies," *Renewable and Sustainable Energy Reviews*, vol. 81, pp. 1192–1205, 2018.
[14] N. Nouaouria, "Una approche d'optimisation par essaim de particules pour le recherche en mémoire de cas," 2013.
[15] R. C. Schank, *Dynamic memory: A theory of reminding and learning in computers and people*, vol. 240. 1982.
[16] E. R. Bareiss, B. W. Porter, and C. C. Wier, "Protos: An exemplar-based learning apprentice," in *Machine Learning, Volume III*, pp. 112–127, 1990.
[17] J. Daengdej, D. Lukose, and R. Murison, "Using statistical models and case-based reasoning in claims prediction: experience from a real-world problem," *Knowledge-Based Systems*, vol. 12, pp. 239–245, oct 1999.
[18] J. L. Kolodner, "An introduction to case-based reasoning," *Artificial intelligence review*, vol. 6, no. 1, pp. 3–34, 1992.
[19] J. W. Schaaf, "Fish and shrink. a next step towards efficient case retrieval in large scaled case bases," in *European Workshop on Advances in Case-Based Reasoning*, pp. 362–376, 1996.
[20] Y.-J. Park, B.-C. Kim, and S.-H. Chun, "New knowledge extraction technique using probability for case-based reasoning: application to medical diagnosis," *Expert systems*, vol. 23, no. 1, pp. 2–20, 2006.
[21] R. L. De Mantaras, D. McSherry, D. Bridge, D. Leake, B. Smyth, S. Craw, B. Faltings, M. L. Maher, M. T COX, K. Forbus, *et al.*, "Retrieval, reuse, revision and retention in case-based reasoning," *The Knowledge Engineering Review*, vol. 20, no. 3, pp. 215–240, 2005.
[22] B. Löpez, "Case-based reasoning: a concise introduction," *Synthesis lectures on artificial intelligence and machine learning*, vol. 7, no. 1, pp. 1–103, 2013.
[23] A. G. D. S. Garza and M. L. Maher, "A process model for evolutionary design case adaptation," in *Artificial Intelligence in Design'00*, pp. 393–412, 2000.
[24] P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3, pp. 1958–1962, 1999.
[25] C. Rodrigues and F. Freire, "Adaptive reuse of buildings: Eco-efficiency assessment of retrofit strategies for alternative uses of an historic building," *Journal of Cleaner Production*, vol. 157, pp. 94–105, 2017.
[26] J. McCarthy, "The frame problem today," in *The Frame Problem in Artificial Intelligence*, p. 3, 1987.
[27] C. K. Riesbeck and R. C. Schank, *Inside case-based reasoning*. Psychology Press, 2013.
[28] M. Veloso and J. Carbonell, "Case-based reasoning in prodigy," *Machine Learning: A Multistrategy Approach. Morgan Kaufmann*, pp. 523–548, 1994.
[29] M. T. Cox, H. Muñoz-Avila, and R. Bergmann, "Case-based planning," *The Knowledge Engineering Review*, vol. 20, no. 3, pp. 283–287, 2005.
[30] T. L. Saaty, "Decision making with the analytic hierarchy process," *International journal of services sciences*, vol. 1, no. 1, pp. 83–98, 2008.
[31] O. Ghorbanzadeh, B. Feizizadeh, and T. Blaschke, "An interval matrix method used to optimize the decision matrix in ahp technique for land subsidence susceptibility mapping," *Environmental Earth Sciences*, vol. 77, no. 16, p. 584, 2018.
[32] R. Karim and C. Karmaker, "Machine selection by ahp and topsis methods," *American Journal of Industrial Engineering*, vol. 4, no. 1, pp. 7–13, 2016.
[33] B. Li and L. Han, "Distance Weighted Cosine Similarity Measure for Text Classification," 2013.
[34] C. Luo, J. Zhan, X. Xue, L. Wang, R. Ren, and Q. Yang, "Cosine normalization: Using cosine similarity instead of dot product in neural networks," in *International Conference on Artificial Neural Networks*, pp. 382–391, 2018.
[35] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE.
[36] M. N. Alam, "Particle Swarm Optimization: Algorithm and its Codes in MATLAB," 2016.
[37] M. N. Alam, B. Das, and V. Pant, "A comparative study of metaheuristic optimization approaches for directional overcurrent relays coordination," *Electric Power Systems Research*, vol. 128, pp. 39–52, nov 2015.
[38] J. Han, M. Kamber, and J. Pei, *Data mining : concepts and techniques*. Elsevier Science, 2011.
[39] S. Zhang, D. Cheng, Z. Deng, M. Zong, and X. Deng, "A novel knn algorithm with data-driven k parameter computation," *Pattern Recognition Letters*, vol. 109, pp. 44–54, 2018.
[40] P. Goel and R. Thareja, "Analysis of various data mining techniques using novel ratings prediction," *IARS'International Research Journal*, vol. 7, no. 2, 2017.

[41] J. Macqueen, "Some methods for classification and analysis of multivariate observations," *in 5th Berkeley Symposium on Mathematical Statistics and Probabilities*, p. 281—297, 1967.
[42] M. Sarstedt and E. Mooi, "Cluster analysis," in *A concise guide to market research*, pp. 301–354, 2019.
[43] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
[44] D. Basak, S. Pal, and D. C. Patranabis, "Support Vector Regression," 2007.
[45] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in neural information processing systems*, pp. 155–161, 1997.
[46] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.
[47] N. Kwon, J. Lee, M. Park, I. Yoon, and Y. Ahn, "Performance evaluation of distance measurement methods for construction noise prediction using case-based reasoning," *Sustainability (Switzerland)*, vol. 11, feb 2019.
[48] T. W. Liao, Z. Zhang, and C. R. Mount, "Similarity measures for retrieval in case-based reasoning systems," *Applied Artificial Intelligence*, vol. 12, no. 4, pp. 267–288, 1998.
[49] H. Núñez, M. Sànchez-Marrè, U. Cortés, J. Comas, M. Martínez, I. Rodríguez-Roda, and M. Poch, "A comparative study on the use of similarity measures in case-based reasoning to improve the classification of environmental system situations," in *Environmental Modelling and Software*, vol. 19, pp. 809–819, sep 2004.
[50] NYISO, "NYISO Electricity Market Data." http://www.nyiso.com/ Accessed 01 October 2019, 2019.