Using Biologically Inspired Computing to Effectively Improve Prediction Models

Anna Bou Ezzeddine[*], Marek Lóderer, Peter Laurinec, Petra Vrablecová, Viera Rozinajová, Mária Lucká, Peter Lacko and Gabriela Grmanová

Faculty of Informatics and Information Technology,
Slovak University of Technology in Bratislava, Ilkovičova 2,
842 16 Bratislava, Slovak Republic
{anna.bou.ezzeddine, marek_loderer, peter.laurinec, petra.vrablecova, viera.rozinajova, maria.lucka, peter.lacko, gabriela.grmanova}@stuba.sk

Abstract. The complexity of certain problems causes that classical methods for finding exact solutions have some limitations. In this paper we propose an incremental heterogeneous ensemble model for time series prediction where biologically inspired algorithms offer a suitable alternative. Ensemble learning techniques are advantageously used for improving performance of various prediction methods. The quality of this kind of machine learning approaches depends on proper combination of used methods. The influence of each of the used method can change on the fly and is determined by proper choice of its weights. Finding optimal weights in prediction methods represents typical optimization problem

[*] corresponding author: email: anna.bou.ezzeddine@stuba.sk, tel. no.: +421 (2) 21 022 315, fax no.: +421 2 654 20 587

The final publication is available at IOS Press through http://dx.doi.org/10.3233/HIS-160228.

with objective function reflecting error minimization, where biologically inspired algorithms can be used. In the proposed paper we study several biologically inspired algorithms in the process of weights optimization. We investigate and compare ensembles using base models and ensembles optimized by biologically inspired algorithms. We demonstrate, that the ensemble learning prediction models optimized by biologically inspired algorithms outperformed the base prediction methods. We present performance and accuracy results of proposed ensemble models that were evaluated on power load datasets with concept drifts.

Keywords: ensemble learning, time series prediction, biologically inspired algorithms, power load prediction.

1 Introduction

Nowadays, the biologically inspired computing is becoming very popular. This is primary due to the fact that the nature and its wise organization has astonished us for a long time. Another motivating factor today is the progress of technology which allows us to simulate these processes much more effectively. There are many areas where natural computing brings new and successful approach. We have found it useful in the problem of creating prediction models, particularly in the ensemble model, where a set of diverse base models is used. The rationale is to combine the advantages of different base algorithms. The essential part of the ensemble learning is the method for combining the base models. The weights of the base models are computed by statistical or different biologically inspired methods. We have concentrated on finding the most suitable method for setting up these weights by investigating six methods of weight computation in this paper. Particular attention was paid to the selection of biologically inspired (BI) algorithms. BI algorithms are one of the main categories of the nature inspired metaheuristic algorithms. The efficiency of the biologically inspired algorithms is due to their significant ability to imitate the best features in nature. These algorithms are based on the selection of the fittest in biological systems which have evolved by natural selection over millions of years. Various biologically inspired optimization algorithms have been developed during the recent decades and according to [6] they can be divided into three main categories: Evolution-based algorithms, Swarm Intelligence and Ecology inspired algorithms. In this work we have used algorithms: Artificial Bee Colony (ABC), Grey Wolves Optimizer (GWO), Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Differential Evolution (DE) and Biogeography-Based Optimization (BBO). They cover all the mentioned categories.

We focused on one application domain – power demand forecasting. With the emergence of smart metering more data are available for a power distributor. The accurate forecasts are essential for his power scheduling and regulation. The improvement of current power demand forecasts in Slovakia by 1 % would lower the total regulation fees by approximately 6.7 million euros a year. The smart metering data arrive continuously each quarter-hour. The prediction model becomes outdated over time with arrival of new data due to changes in data distribution. Changes are caused by various factors, e.g. the power demand can be affected by the day of the week, season, weather, demographic or economic factors. The phenomenon is called *concept drift* [11]. To achieve more targeted experimental cases we have identified some typical concept drift patterns in smart metering data from Slovakia and explored which method fits the best for the particular shape of the drift. The achieved results let us assume that we have found the most appropriate weighting method for specific patterns.

This paper is organized as follows: section 2 contains a summary of the related work. In section 3 we describe our proposed approach (the incremental heterogeneous ensemble model for time series prediction and its improvement in terms of new ways of weighting base models). Experimental evaluation and results are presented in section 4 and the conclusion is in section 5.

2 Related Work

To compute time series predictions, classical approaches are mainly applied. Recently, the incremental and ensemble approaches were also introduced to create methods that are able to adapt to changes.

Classical approaches to time series prediction are represented mainly by regression and time series analysis. Regression approaches model the dependencies of target variable on independent variables. For power demand prediction, the independent variables are the day of the week, hour of the day, temperature, etc. Plenty of different regression models were presented in the literature, such as a stepwise regression model, neural network and decision tree [48]. Because of the strong seasonal periodicities in power demand data, time series models are often used to make predictions. The most often, the models of Box-Jenkins methodology [3] are applied, such as AR, MA, ARMA, ARIMA and derived models.

However, the classical approaches are only able to model seasonal dependencies. They are not able to adapt to incoming data comprising different types of concept drift.

Thus, they are not suitable to make predictions of power demand evolving in time and incremental learning algorithms are needed.

Incremental algorithms process new data in chunks of appropriate size. They can possibly process the chunks by classical algorithms. Most of the incremental learning algorithms, presented in the literature are based on machine learning approaches, e.g. incremental support vector regression [52] and extreme learning machines [14]. Recently, the incremental ARIMA was proposed for time series prediction [34].

Usually, the incremental learning algorithms alone cannot treat changes in the target variable sufficiently well. In order to capture the different types of concept drifts, ensemble models are used to achieve better predictions (see [32] for introduction to ensemble learning). Ensemble model uses a set of diverse base models, where each base model provides an estimate of a target variable – a real number for a regression task. The estimates are combined together to make a single ensemble estimate. The combination of base estimates is usually made by taking a weighted sum of base estimates. The idea behind it is that the combination of several models has the potential to provide more accurate estimates than single (even incremental) models, namely in the presence of concept drifts [54]. Some ensemble models use additional pruning phase to achieve more accurate results by dropping redundant and useless models from the base set [46]. In addition, they have several more advantages over single models, such as the scalability, the natural ability to parallelize.

The accuracy of the ensemble depends on the accuracy of base models and on their diversity [26]. The diversity of base models may be accomplished either by homogeneous or by heterogeneous ensemble learning approaches [54]. In homogeneous approach, the ensemble is comprised by models of the same type learned on different subsets of available data [21], [39], [40]. The heterogeneous learning process applies different types of base models [41], [57]. From the literature several combinations are also known of heterogeneous and homogeneous learning [56].

Ensemble learning was also used to predict values of time series [20], [29], [42], [44], [50]. All of these approaches use ensembles of regression models for generating time series predictions. They do not take explicitly any seasonal dependence into account and do not use time series analysis methods to make predictions.

The essential part of the ensemble learning process is the approach used to combine predictions of base models. For regression problems, this is done by their linear combinations. Weights used in the combination sum to 1, and are computed by various statistical methods and different biologically inspired (BI) methods.

BI algorithms get the inspiration from living organisms and nature. It can be said that nature always finds the strategy to solve various problems even the complex ones. This strategy involves interaction among organisms, balancing the environment, adaptations to changes, species diversity, effective discovering with no or little knowledge of the search space. Nature has been solving these problems for million years, therefore strategies that are seen nowadays can be considered as effective. The evolutionary algorithms are inspired from the genetic evolution process. Genetic Algorithm (GA) [12], and Differential Evolution (DE) [45] are the well-known paradigms of the evolutionary algorithms. The evolutionary algorithms have been remarkably improved over the last decades. Simon [43] has proposed a new evolutionary algorithm, namely Biogeography-Based Optimization (BBO). The BBO algorithm is used for global recombination and uniform crossover which are inspired from GA literature. The most well-known paradigms in the area of Swarm Intelligence are Particle Swarm Optimization (PSO)

[9], Cuckoo Search [51] and Ant Colony Optimization (ACO) [8]. These algorithms are based on the simulation of the collective behavior of animals. PSO is a population-based method inspired from the social behavior of bird flocking or fish schooling. Foraging behavior of honey bees have also been an inspiration for metaheuristic algorithms. For example, the Artificial Bee Colony (ABC) [22] consist of three types of bees and each bee type has different task in food (nectar) collecting. There are also new Swarm Intelligence algorithms like Grey Wolf Optimizer [33] that implements hunting strategy of grey wolves pack where omega wolves are directed by three dominant wolves (alpha, beta and delta).

3 Incremental Heterogeneous Ensemble Model for Time Series Prediction

We propose the *incremental heterogeneous ensemble model for time series prediction*. The main goal is to investigate different biologically inspired algorithms for computing weights that are used to combine predictions of base models. The ensemble approach was chosen for its ability to adapt quickly to changes in the distribution of a target variable and its potential to be more accurate than a single method. As base models we use several chosen regression and time series models to capture seasonal dependencies.

3.1 Incorporation of Different Types of Prediction Models

Our ensemble model incorporates several types of models for capturing different seasonal dependencies. The models differ in *algorithm*, *size of data chunk* and *training period* (see Fig. 1). We chose the heterogeneous approach that assumes different algorithms for particular base models what brings the diversity into the ensemble. The size of each data chunk is chosen in order to capture particular seasonal variation, e.g. data from the last 4 days for daily seasonal dependence. Training period represents the period, after which a particular base model is re-trained. The model that is trained on a data chunk containing data from the last 4 days, can be re-trained the next day (using a 1-day training period) on a data chunk that overlaps with the previous one.

--- Fig. 1 ---

The ensemble model is used to make one-day ahead predictions. Let *h* be the number of observations that are available daily. Hence *h* is also the number of next predictions the model is going to compute in each step by weighted sum of predictions made by *m* base models. After the observations for the current day are available, the prediction errors are computed. Based on computed errors, the weights are updated by one of weighting scheme. Then, each base model $i = 1, \dots, m$, for which *t* fits its training period p_i, is retrained on data chunk of size s_i.

Let $\hat{\mathbf{Y}}^t$ be the matrix of predictions of *m* base methods for next *h* observations at day *t* (see equation 1) and $\mathbf{w}^t = (w_1^t \quad \dots \quad w_m^t)^T$ is a vector of weights for *m* base methods at day *t* before observations of day *t* are available. Weights and particular predictions are combined to make an ensemble prediction $\hat{\mathbf{y}}^t = (\hat{y}_1^t \quad \dots \quad \hat{y}_m^t)^T$.

$$\hat{\mathbf{Y}}^t = \begin{pmatrix} \hat{y}_{11}^t & \cdots & \hat{y}_{1m}^t \\ \vdots & \ddots & \vdots \\ \hat{y}_{h1}^t & \cdots & \hat{y}_{hm}^t \end{pmatrix} = (\hat{\mathbf{y}}_1^t \quad \dots \quad \hat{\mathbf{y}}_m^t) \tag{1}$$

The ensemble prediction for *k*th $(k = 1, \dots, h)$ observation is calculated by equation 2.

$$\hat{y}_k^t = \frac{\sum_{j=1}^m \hat{y}_{kj}^t \overline{w}_j^t}{\sum_{j=1}^m \overline{w}_j^t} \tag{2}$$

From the prediction matrix $\widehat{\mathbf{Y}}^t$and vector of *h* current observations$\mathbf{y}^t = (y_1^t \quad ... \quad y_h^t)^T$, the vector of errors $\mathbf{e}^t = (e_1^t \quad ... \quad e_h^t)^T$ for *m* methods is computed. The error of each method is given by $e_j^t = \text{median}(|\hat{\mathbf{y}}_j^t - \mathbf{y}^t|)$. A vector of errors $\mathbf{e}^t$ is used to update the weights vector of base models in the ensemble by one of selected weighting schemes. The integration method is robust since it uses the median absolute error that is not sensitive to outliers.

3.2 Base Models

In the ensemble model we included the following 8 base models that incorporated short-term (7 models) and long-term (1 model) seasonal dependencies. According to prediction approach, our selected base models can be divided into three groups: regression-based models, time series analysis models and AI based models.

Regression-Based Models. *Multiple linear regression* (MLR) attempts to model the relationship between two or more explanatory variables and a response variable by fitting a linear equation to observed data.

Time Series Analysis. *Random walk* (RW) is a naive method appropriate for time series data that uses ARIMA seasonal method for forecasting. We have used the $ARIMA(0,0,0)(0,1,0)_h$.

Method STL (Seasonal and Trend decomposition using Loess) is *a combination with four different predicting methods.* The main idea in STL is to decompose a seasonal time series into three parts: trend, seasonal and remaining [7].

We combined the STL with ARIMA model (STL+ARIMA) for the prediction of the decomposed parts. ARIMA was introduced by Box and Jenkins [4] and is one of the most popular approaches in forecasting [18].

The second combination of STL involved the Holt-Winters exponential smoothing for the prediction of the seasonal part, the neural network for the trend part and the multiple regression model for the noisy part (STL+HW+ANN+MLR).

Moving average model (MA) can extrapolate the non-seasonal patterns [37] and trends. It is supposed, that the time series is locally stationary and has a slowly varying mean. The moving (local) average is taken for the current value estimation of the mean and used as the forecast for the near future.

AI Based Models. *Artificial Neuronal Networks* (ANN) are trained to learn the relationships between the input variables (past demand, temperature, etc.) and historical load patterns. Proper construction of the ANN model is based on the exploratory data analysis, where traditional time series analysis methods for estimation of the lag dependence in the data are used.

The *SVM regression* method (SVR) [55] that utilize an epsilon-regression that uses the Gaussian radial basis function kernel is employed and tested. We have used the same explanatory variables as in the multiple linear regression model.

Random forests (RF) [5] are an ensemble learning method based on constructing of decision trees at training time and outputting the class (classification) or mean prediction (regression) of the individual trees. Random forests correct for decision trees so that they prevent over-fitting to their training set.

3.3 Biologically Inspired Base Models Weighting

To solve complex problems, scientists have often been inspired by the behavior of animals and plants observed in nature. At first glance, the behavior of individuals may seem very simple, but as a whole, they can solve complex problems simply and very effectively.

In this study, six weighting schemes based on biologically inspired algorithms for heterogeneous ensemble learning model are proposed. These algorithms are used to minimize prediction error of *m* base time series models for power demand forecasting. In the next section, we describe principles and behavior of six biologically inspired algorithms used in proposed ensemble model.

Swarm Intelligence Inspired Algorithms. Swarm Intelligence (SI) inspired algorithms is a branch of biologically inspired algorithms focused on collective social behavior of organisms. SI mimics the way how organisms collectively perform specific tasks like food foraging, hunting, environment exploration or finding a suitable mate. SI algorithms are very popular and used to solve complex problems. There are many reasons for that. First of all, these algorithms usually share information among multiple agents, so the self-organization, co-evolution and learning during iterations may help to provide high efficiency of most SI-based algorithms [10]. Next reason is that they can solve the problems in a flexible and robust way at the same time. The flexibility allows adaptation to changes in environment or population. The robustness ensures that the solution can be found even when some individual agents fail to perform their task. Another reason is that multiple agents can run parallel, so the optimization task can be performed faster.

Artificial Bee Colony (ABC). ABC simulates the foraging behavior of a honeybee colony [22]. The algorithm is used to solve constrained as well as unconstrained global optimization problems.

The bee colony in the implementation of ABC algorithm consists of three types of bees: employed bees, onlooker bees and scout bees. Usually half of the colony is composed by employed bees and the second half is composed by onlooker bees. Each employed bee is associated with one specific food source. Each food source represents a solution (*d*-dimensional vector $\boldsymbol{X}_i = \left(x_1^i, x_2^i, x_3^i, \dots, x_d^i\right)$ of values). The task of an employed bee is to search for new food source $\boldsymbol{V}_i$ with more nectar (better solution) in the neighborhood of the current food source. The search is defined by the equation:

$$v_k^i = x_k^i + \phi_k^i \times \left(x_k^i - x_k^j\right)$$

$\boldsymbol{X}_j$ is randomly selected candidate solution $(i \neq j)$, $k \in \{1,2,\dots,d\}$ is randomly chosen index and ϕ_k^i is a random value from a range $[-1,1]$.

If the fitness of the new source $\boldsymbol{V}_i$ is better than the fitness of current source $\boldsymbol{X}_i$ then the employed bee immediately forgets the old food source and puts the new source in its memory. After all employed bees return to the hive, they share information about the amount of nectar and position of their food sources with the onlooker bees. Each onlooker bee then chooses one food sources depending on the provided information. The food sources with higher quality are more likely to be selected by the onlooker bees than the ones with lower quality. The probabilistic selection is based on a roulette wheel selection mechanism which can be defined by the following equation.

$$P_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n}$$

fit_i is the fitness value of the solution i, which is proportional to the nectar amount of the food source in the position i and SN is the number of food sources.

After an onlooker bee has performed the selection, then the bee visits the source and search in the neighborhood for new better solutions and acts exactly like an employed bee. If a food source is not improved over a predefined number of trials (called limit), then the food source is considered as exhausted and it is abandoned. The employed bee which abandons its food source turns into a scout bee. The scout bee searches for new source regardless of existing sources.

Assuming that the exhausted source is $\boldsymbol{X}_i$ then the scout bee can discover a new food source by the following equation.

$$x_i^j = x_{min}^j + rand(0,1)\left(x_{max}^j - x_{min}^j\right)$$

$j \in \{1,2,\dots,N\}$ are randomly chosen indexes. The algorithm proceeds until predefined number of iterations is reached. x_{min}^j and x_{max}^j are lower and upper boundaries of the jth dimension.

Grey Wolves Optimizer (GWO). GWO is inspired by social leadership and hunting behavior of grey wolves [33]. Grey wolves live in small packs with a strict hierarchy that help them to survive in a very rough environment where they hunt prey. Each pack of wolves consists of four types of wolves: alpha (α), beta (β), delta (δ), and omega (ω). Alpha is the leader of whole pack. It decides which prey is going to be hunted. Beta and delta are subordinate wolves that help the alpha in decision-making or other pack activities. Omegas are the lowest ranking grey wolves and their main task is to help with hunting. During the hunt, they are guided by alpha, beta and delta wolves toward good preys (promising areas of the search space). The hunt consists of three steps: searching for prey, encircling prey, and attacking prey.

The algorithm implements the wolves' hierarchy found in nature and the hunt technique consisting of searching for prey (exploration of the search space), encircling and attacking prey (exploitation the search space). Each wolf represents a solution in high-dimensional space.

The algorithm starts with number of N wolves and runs predefined number of iterations. At the beginning of each iteration three best wolves (depending on their fitness) are chosen as alpha, beta and delta. The rest wolves are omegas. During the iteration omega wolves update their position around the recently chosen leader wolves. The position of each omega wolf at iteration *t* is updated by the following six equations.

$$\mathbf{D}_\alpha = |\, \mathbf{C}_1 . \mathbf{X}_\alpha - \mathbf{X} \,| \qquad \mathbf{X}_1 = \mathbf{X}_\alpha - \mathbf{A}_1 . \mathbf{D}_\alpha$$

$$\mathbf{D}_\beta = |\, \mathbf{C}_2 . \mathbf{X}_\beta - \mathbf{X} \,| \qquad \mathbf{X}_2 = \mathbf{X}_\beta - \mathbf{A}_2 . \mathbf{D}_\beta$$

$$\mathbf{D}_\delta = |\, \mathbf{C}_3 . \mathbf{X}_\delta - \mathbf{X} \,| \qquad \mathbf{X}_3 = \mathbf{X}_\delta - \mathbf{A}_3 . \mathbf{D}_\delta$$

The final position of the wolf is calculated as follows:

$$\mathbf{X}_{t+1} = \frac{\mathbf{X}_1 + \mathbf{X}_2 + \mathbf{X}_3}{3}$$

$\boldsymbol{D}$, $\boldsymbol{C}$, and $\boldsymbol{X}$ are *d*-dimensional vectors, $\boldsymbol{X}_\alpha$ is the position of the alpha, $\boldsymbol{X}_\beta$ is the position of the beta, $\boldsymbol{X}_\delta$ is the position of delta. The $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, $\boldsymbol{C}_3$, $\boldsymbol{A}_1$, $\boldsymbol{A}_2$ and $\boldsymbol{A}_3$ are randomly chosen coefficient vectors, and $\boldsymbol{X}$ is the current position of the wolf. $\boldsymbol{A}$ and $\boldsymbol{C}$ vectors are used to balance the exploration and exploitation in GWO. $\boldsymbol{C}$ and $\boldsymbol{A}$ coefficient vectors are calculated by the following equations.

$$\mathbf{A} = 2\mathbf{a} \,.\, \mathbf{r}_1 - \mathbf{a}$$

$$\mathbf{C} = 2.\,\mathbf{r}_2$$

Components of $\boldsymbol{a}$ are linearly decreased from 2 to 0 over the course of iterations and $\boldsymbol{r}_1, \boldsymbol{r}_2$ are random vectors from range [0,1].

The optimization stops when the algorithm reaches the predefined number of iterations. The best solution is held by the alpha wolf.

Particle Swarm Optimization (PSO). PSO can be used to find approximate solutions to extremely difficult or impossible numeric maximization and minimization problems. PSO is a population based stochastic optimization technique. It is one of the most widely used Evolutionary Computation Techniques; it takes inspiration from the natural swarm behavior of birds and fishes. It was firstly introduced in 1995 by Kennedy and Eberhard [25]. PSO has roots in two main component methodologies - artificial life in general and swarming theory in particular [24]. It is based on the collective intelligence of a swarm of particles. Each particle explores a part of the search space looking for the optimal position and adjusts its position according to two factors: the first is its own experience and the second is the collective experience of the whole swarm.

The concept of algorithm is very simple. Each particle (a member of population) represents a solution in a high-dimensional space. Each particle has four vectors: current particle position, best position found so far (pbest), the global best position found by neighbors so far (gbest) and velocity of the particle.

At the beginning, a number of particles are set into searching space. Each particle is attracted towards its own best position found so far and the global best position found by neighbors. If particle finds a new better position it will updates its current best position (pbest). At the beginning of each iteration a particle with the best position is selected as the global best position (gbest). Algorithm proceeds until predefined number of iterations is reached or the objective is no longer improved. Each particle updates its position x_{k+1}^i and velocity v_{k+1}^i during each iteration.

$$x_{k+1}^i = x_k^i + v_{k+1}^i$$

$$v_{k+1}^i = v_k^i + c_1 r_1 \left(p_k^i - x_k^i\right) + c_2 r_2 \left(p_k^g - x_k^i\right)$$

x_k^i is the current particle position, v_k^i is the particle velocity, $c_1$and c_2 are cognitive and social parameters, $r_1$and r_2 are random numbers from range $(0{,}1)$.

PSO is widely used in many optimization applications, because it can be easily understand and it is computationally and memory effective. It has simple concept of calculation and has only few initial parameters which need to be set. It is simpler than GA and able to obtain better solutions [16]. It may cause by different philosophy of the algorithms. In PSO, all particles share the information about the global best solution in order to improve themselves. The GA does not share such information. It only removes the worst solutions and saves the good ones to next generation.

Evolution Inspired Algorithms. Evolution-based algorithms are inspired by the ability of living organisms to adapt to their environment by evolving. This is one of the most powerful problems solving technique found in nature. It is responsible for the design of all living organisms on the earth, and for various strategies they use to survive.

Evolution-based algorithms consist of population of individuals where each one represents a solution to a given problem. Evolution-based algorithms are iterative. At each iteration a predefined set of operators is applied to individuals of population to generate new individuals of next generation. There are three types of operators. First, recombination operators used to recombine usually two individuals (parents) to produce new individual or individuals. Next operators are mutations used to modify an individual which introduces new features to the population. The most important operator is selection. It is usually based on fitness of each individual (a quality measurable capability to survive and perform a given task). Individuals with a higher fitness have a higher probability to be chosen to the population of the next iteration or as parents for the generation of new individuals.

Genetic Algorithm (GA). GA [17] is a stochastic optimization method. Its basic principle is inspired by real biological evolution process. The fundamental thesis of Darwin theory of evolution proposes that only the best adapted individuals of a population will survive and reproduce. After reproduction of individuals with high fitness (assessment of the individual adaptation in environment) new individual is generated, such a descendant will reach high fitness with high probability. The goal of the evolution is the emergence of an individual with maximum possible fitness in a given environment. When minimizing errors, higher fitness corresponds to lower error.

GA works with a set of individuals who make up the population. Individual features are encoded using vectors (chromosomes in biology). Each individual is rated with its fitness – value which expresses its success in a given environment. Genetic algorithm consists of following steps (there are different selection, crossover and mutation methods, more in [27]):

1. New randomly generated population is created.

2. In every evolution epoch create new population from the old one. Elitism is used to ensure that best individuals will always survive, in the end a small fraction of the best individuals is copied to the new generation.
   (a) Two parent individuals are chosen from population, individuals with higher fitness should have higher probability to be chosen.
   (b) Reproduction takes place only with a certain probability. If individuals do not pass reproduction they are returned with no changes to the population. Reproduction takes place in two steps - crossover and mutation:
      (i) Crossing is exchange of parts of the chromosomes of individuals between themselves. A random point u is chosen as a crossing point where the chromosome divides and parents mutually exchange chromosome parts, thus producing new offspring. The new offspring can be simply calculated by the equation:

$$OFFSPRING = PARENT_1 * u + PARENT_2 * (1 - u)$$

      where u is a random number from (0, 1) representing the portion of the parent values transmitted to the offspring.
      (ii) A mutation is a change in the chromosome of an individual with a certain probability (e.g. when using real numbers encoding, addition of random number with normal distribution).
   (c) Individuals are inserted into new population and their fitness is computed.
3. If best individual fitness is higher than selected threshold (solution was found) or max population epoch is reached then stop, else return to step 2.

Differential Evolution (DE). DE [45] is an evolution strategy for optimization. It was designed to be a stochastic direct search method. It is able to handle non-differentiable, nonlinear and multimodal cost functions. It can be parallelized. A low number of parameters is required to use it and it has good convergence properties. DE borrows the idea from Nelder & Mead method and employs information from within the vector population to alter the search space. DE's self-organizing scheme takes the difference vector of two randomly chosen population vectors to mutate an existing vector. It is done for every population vector in contrast to traditional evolution strategies.

Formally, it can be described as a parallel direct search method that utilizes *NP* *D*-dimensional parameter vectors $\boldsymbol{x}_i^G, i = 1,2,\dots,NP$ as a population for each generation *G*. The first generation is chosen randomly to cover the entire parameter space. Then each next generation is created by this procedure:

1. *mutation:* For each target vector $\boldsymbol{x}_i^G, i = 1,2,\dots,NP$, a mutant vector is created as $\boldsymbol{v}_i^{G+1} = \boldsymbol{x}_{r1}^G + F \cdot (\boldsymbol{x}_{r2}^G - \boldsymbol{x}_{r3}^G)$ with random indices r_1, r_2, r_3 that are different from i. F is a weight from [0,2] that controls the amplification of the differential variation.
2. *crossover:* Each target vector $\boldsymbol{x}_i^G$ is introduced to crossover with the mutant vector $\boldsymbol{v}_i^{G+1}$ to increase the diversity and a trial vector $\boldsymbol{u}_i^{G+1}$ is created as:

$$u_{ji}^{G+1} = \begin{cases} v_{ji}^{G+1} & if(randb(j) \leq CR) or\, j = rnbr(i) \\ x_{ji}^{G} & if(randb(j) > CR)\, or\, j \neq rnbr(i) \end{cases}$$

$$j = 1,2,\dots,D$$

randb(j) is the *j*th evaluation of a uniform random number generator with outcome from [0,1], CR is the crossover constant from [0,1] determined by the user, *rnbr(i)* is a randomly chosen index from 1,2,…,*D* which ensures that the trial vector gets at least one parameter from the mutant vector.

3. *selection:* Each trail vector $\boldsymbol{u}_i^{G+1}$ is compared to the original target vector $\boldsymbol{x}_i^{G}$. The one that yields a smaller cost function is selected to the next generation *G+1*.

Ecology Inspired Algorithms. Ecology inspired optimization algorithms are the third branch of BI algorithms. It is the latest branch so it is not as numerous as the previous ones. The main characteristic of these algorithms is that they are inspired not only by intra-species but also the inter-species interaction, symbiosis and interaction with the environment.

Biogeography-Based Optimization (BBO). BBO is a population-based evolutionary algorithm that is based on the mathematics of biogeography. Biogeography is the study of the geographical distribution of biological organisms.

The theory builds on the first principles of population ecology and genetics to explain how distance and area combine to regulate the balance between immigration and extinction in island populations. Geographical areas that are well suited to life for biological species are said to have a high habitat suitability index (HSI). Features that correlate with HSI include such factors as rainfall, temperature diversity of vegetation, diversity of topographic features and land area. The variables that characterize habitability are called suitability index variables (SIVs). SIVs can be considered the independent variables of the habitat, and HSI can be considered the dependent variable [43,30].

Mathematical modeling of BBO is given as follows: if $P_S(t)$ denotes the probability that a habitat contains exactly *S* species at time *t*, at time *t+Δt* the probability is:

$$P_S(t+\Delta t) = P_S(t)(1-\lambda_S \Delta t - \mu_S \Delta t) + P_{S-1}\lambda_{S-1}\Delta t + P_{S+1}\mu_{S+1}\Delta t$$

where λ_S is immigration rate, μ_S emigration rate, when there are *S* species in the habitat. The largest possible number of species that the habitat can support is *S_{max}*, at which point the immigration rate is zero. If there are no species on an island, then the emigration rate is zero. As the number of species on the island increases, it becomes more crowded, more species representatives are able to leave the island, and the emigration rate increases. When the island contains the largest number of possible species S_{max}, the emigration rate reaches its maximum possible value *E*.

If $\Delta t \to 0$ then the following equation results:

$$P_S = \begin{cases} -(\lambda_S + \mu_S)P_s + P_{S+1}\mu_{s+1} & S = 0 \\ -(\lambda_S + \mu_S)P_S + P_{S+1}\mu_{S+1} + P_{S-1}\lambda_{S-1} & 1 \le S \le S_{max} - 1 \\ -(\lambda_S + \mu_S)P_S + P_{S-1}\lambda_{S-1} & S = S_{max} \end{cases}$$

From [43]

$$\mu_k = \frac{Ek}{n}$$

$$\lambda_k = I\left(1 - \frac{k}{n}\right)$$

where *E* is maximum emigration rate, *I* is maximum immigration rate, *k* represents the rank of a habitat after sorting them according to their HSI and *n* is total number of species in that habitat.

The BBO migration strategy is similar to the global recombination approach of the greedier GA [35] and evolutionary strategies [1]. In evolutionary strategies, global recombination is used to create new solutions. BBO migration is used to change existing solutions. Global recombination in evolutionary strategy is a reproductive process, while migration in BBO is an adaptive process. It is used to modify existing islands.

4 Experimental Evaluation

We performed a series of experiments to evaluate the performance of our ensemble model that incorporates 8 previously mentioned base models (see section 3.2). Particularly, we examined the behaviour of various biologically inspired weighting algorithms and their ability to react to the concept drifts. We identified four concept drift patterns and further investigated what type of weighting is appropriate for the particular patterns.

We evaluated the performance of our ensemble model in comparison to its individual base models. In our previous research [13] the ensemble model consisted of 13 base models. We optimized the ensemble accuracy by reduction of the number of prediction base models. We excluded prediction models with similar results to increase the diversity of the ensemble model, which is the key property for good ensemble results [31].

Our goal was to find out which biologically inspired weighting algorithm is the most suitable (i.e. shows the smallest prediction error) for data with the occurrence of sudden or gradual concept drifts. We also studied the convergence properties and time complexity of each approach.

In the next sections we describe the data we used, the experiments and their results.

4.1 Data

For the experiments we used the Slovak smart metering data. The quarter-hourly measurements of power load for 20 regions of Slovakia were available. We observed concept drifts in data mainly on the holidays or summer leave. We identified the four common patterns of drifts (see Fig. 2) and prepared three datasets for each type of drift.

--- Fig. 2 ---

4.2 Measures

We utilized the *mean absolute percentage error* (MAPE) to evaluate the prediction accuracy. MAPE is based on relative (percentage) errors. This enables us to compare errors for time series with different absolute values. It is computed by the following equation.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|\hat{y}_t - y_t|}{y_t} \times 100 \qquad (3)$$

4.3 Experiment Setup

To design the experiments, the best data chunk sizes for particular models were found experimentally. The most precise predictions for regression (MLR) and AI based methods (SVR, RF) were for 4 days. The other short-term models (RW, STL+ARIMA, STL+HW+ANN+MLR, ANN) coping with daily seasonality performed the best with a data chunk size equal to 10 days. MA need 1-year data chunks. We observed that models which incorporate trend (e.g., STL-based models), perform better with a greater size of data chunk.

Training periods for methods working with short-term seasonal dependency were 1 day and for the long-term methods were 1 year. Since we had less than 2 years' data available, they were trained only once and were not further retrained. There were available data for training (both for the previous 10 days and previous 1 year observations) only for the period July 1, 2014 – February 15, 2015. Only non-holiday Tue-Fri days were assumed, because they contain similar daily consumption patterns. The base models were incrementally trained based on particular sizes of data chunks and their periods on synthetic and real data, while subsequently adding new data and ignoring old data. At each day, ensemble weights are recomputed by the chosen weighting method.

Biologically inspired algorithms, which we have used, have several characteristics in common. All of them minimize the mean absolute percentage error that stands for the fitness function (see section 4.2). They use populations of individuals (known as particles in PSO, bees in ABC or wolves in GWO) to represent the solutions. The dimension of those individuals is given by the number of base methods *m*.

New weights are optimized based on results from *k* last days (*k*-window). At the beginning, when there are available measurements from only 1 day, the parameter *k* is set to 1. As there are more days available, *k* increases to its maximal value. The maximal value was experimentally set to 4. We implemented dynamic change of window length that sets k to value 1, when e^t is larger than given threshold *T*. The dynamic change of window length is applied only if it decreases objective function. The best value of $T =$ 5 was found experimentally. Weights computed by biologically inspired algorithms are rescaled that the smallest value is 1 and largest value is 10.

For GA we set the population size to 80 and the number of generations to 200. Each next population is composed of the best 50% individuals (natural selection). The next half of the population is generated by crossover. All but the best individual are mutated. Mutation adds noise from $\mathcal{N}(0, 0.05^2)$ to 2 randomly selected weights in the individual. The number of best (mutation-free) individuals is 2.

The population size of the PSO was set to 35. The algorithm ends when the objective function is less than 0.5 or if maximal number of 35 iterations is reached. The local exploration constant c_1 was set to $0.5 + \log(2)$ and the global exploration constant was set to $0.8 + \log(2)$.

In the ABC algorithm, the number of bees in swarm was set to 50. Half of the swarm consisted of worker bees, the other half consisted of onlooker bees. The limit of food source was set dynamically depending on swarm size and number of problem dimensions. The formula for the food source calculation is:

$$limit = (\text{swarm size}\ /2) * dimensions$$

Maximum number of iterations was set to 75.

The number of individuals in DE was set to 80 respectively 130. The number is calculated as a decuple of the problem dimensions. The crossover constant CR was 0.9 and selecting differential weighting factor F was 0.8. The algorithm tested each new mutated individual with the individual from the previous generation. Only individual with the best value proceeded into the next generation. The maximum number of generation was 100.

The GWO algorithm operated with population of 50 wolves. The exploration and exploitation constants A and C were chosen randomly for every wolf movement. The A was chosen from range [0.3, 1]; C was chosen from range [0.6, 1]. Algorithm stopped after 40 iterations were reached.

In the BBO, the population size was set to 35. The habitat modification probability was 1 and the mutation probability was 0.6. The best 5 habitats were kept from one generation to the next without change (elitism). The maximum number of generation was set to 120.

The experiments were provided in the R environment. We used methods implemented in *stats*, *forecast* [19], *rpart* [47], *kernlab* [23], *randomForest* [28] *pso* [2], *DE-optim* [36], *ABCoptim* [53] and *bbo* [38] packages.

4.4 Results

Table 1 (base models) and

Table 2 (methods of model weighting) contains average daily MAPE prediction errors for four selected types of power load change. Each type of concept drift pattern has three experimental datasets. The values written in bold represent minimal prediction errors for each of the test cases. In

Table 2 the results of model weighting methods are presented. These values are the averages of five executions of the experiments.

Average daily results (see Table 1 and

Table 2) show that biologically inspired algorithms achieved better prediction results than base models methods in every case except one (the second dataset in concept drift of type 1), where the base model STL+ARIMA was more accurate than the best weighting method (GA), but only by 0.11 %.

--- Table 1 ---

--- Table 2 ---

Fig. 3 shows the average prediction error of every base method and every weighting method of all datasets. In average, all biologically inspired algorithms were better than all base prediction models by 1.33 %. According to the results, we cannot say which biologically inspired algorithm is better for weighting base models, because differences in prediction errors among them are not significant.

--- Fig. 3 ---

Other important characteristics of biologically inspired algorithms is the number of required iterations needed for obtaining good results. In our experiments we set the best found input parameters for each algorithm individually. The number of iteration was set to 100 in all algorithms. During the experiment the best value of objective function in each iteration was stored. In this test case the objective function was MAPE. The six biologically inspired algorithms were tested on all concept drift patterns. Each experiment was executed 5 times. Fig. 4 presents the relationship between the number of iterations and the average MAPE prediction error for each of the biologically inspired algorithms.

The results show some similarities among algorithms. For example PSO and GWO converge fast, but after a small number of iterations they stopped to converge. DE and BBO converge also fast but they need more iterations than previous methods. On the other hand GA and converge slowly but gradually. ABC offers the best tradeoff between number of iterations and obtained results.

The differences in convergence and required number of iterations are caused by the mechanisms of particular algorithms. The evolution-based algorithms use a strong survival mechanism to select the best individuals of population and iteratively converge to solution. The bad individuals are rejected from population. The Swarm Intelligence based algorithms seem to have an advantage over evolutionary based algorithms because they are able to share information among certain or all individuals in population. The information about search space and existing solutions are used to improve position of the individuals and accelerate the convergence to the optimal solution. The Ecology inspired algorithms can be considered as extension of Swarm Intelligence based algorithms. They share information on intra and inter species level. In some cases the accuracy of these algorithms may be better than accuracy of SI algorithms [43].

--- Fig. 4 ---

ABC and GA yield good results when compared to other tested algorithms during the training phase, but their prediction error on test data is bigger or equal to the prediction errors of other tested algorithms which indicates overtraining. It should be noted that all algorithms exhibit small overtraining. It is caused by the fact that algorithms calculate optimal weights for one day ahead prediction based on set of base models. Incoming concept drift increases the prediction error of base models which is transmitted to ensemble before biologically inspired algorithms calculate new appropriate model weights. The difference between the prediction errors of training and testing cases is about 0.9-1.8% depending on the used algorithm.

We also tested various scenarios with different values of movement, selection, mutation and communication variables used to modify individuals' behavior with the aim to balance the exploration and exploitation, avoiding to fall to local minima. Tested scenarios affected mainly the prediction error, not the convergence speed or the shape of the convergence line.

--- Fig. 5 ---

The results show that the Swarm Intelligence based algorithms outperformed evolution and Ecology based algorithms. The best results were obtained by PSO. It is 2.5 times faster than the second best algorithm in the experiment. ABC and GWO achieved similar results. Compared to PSO, both algorithms contain more information sharing and interactions among individuals in the population, which increases the algorithm execution time.

Evolution-based algorithms DE and GA needed approximately four times more time to calculate the solution comparing to the best algorithm in the experiment. Because Evolution-based algorithms do not use any additional information about search space or best solutions found so far, and use only selection, mutation and crossover operations, they need more time to converge to the optimal solution.

BBO needed the most time to find the optimal solution. In general Ecology inspired algorithms are computational and memory expensive because of big amount of intra-species interaction as well as inter-species information sharing.

5 Conclusion

The presented paper describes the results in the area of exploring new ways of predictive base models weighting in ensemble learning. Our goal was to minimize the error of the proposed prediction model. The main concern was to examine the behavior of biologically inspired algorithms as weighting methods in their ability to react to concept drifts and to find out the most suitable algorithm for data where sudden or gradual changes occur. We identified four concept drift patterns and investigated which type of weighting fits best to each particular pattern. During the evaluation we have considered various types of concept drift and explored how the precision of the predictive model is affected by individual methods (i.e. swarm intelligence, evolution and ecology inspired algorithms). The evaluation showed that the ensemble model performed better than its base models individually. The BI algorithms provided very similar accuracy but manifested different convergence properties. The best results were achieved by the weighting computed by the PSO algorithm, which converged to the solution faster than the other five algorithms. In power demand forecasting the accuracy is essential but the timely prediction is also of great significance. Our experiments, similar to the works [49] and [15], proved that PSO can be very successfully employed to solve various complex problems.

Acknowledgement. This work was partially supported by the Research and Development Operational Programme for the project "International Centre of Excellence for Research of Intelligent and Secure Information-Communication Technologies and Systems", ITMS 26240120039, co-funded by the ERDF and the Scientific Grant Agency of The Slovak Republic, grant No. VG 1/0646/15 and VG 1/0752/14.

References

[1] T. Back, *Evolutionary Algorithms in Theory and Practice*, Oxford University

Press, Oxford, UK, 1996.

[2] C. Bendtsen, pso: Particle Swarm Optimization. R package version 1.0.3, (2012), 14.

[3] G.E.P. Box, G.M. Jenkins and G.C. Reinsel, *Time Series Analysis: Forecasting and Control*, John Wiley & Sons, New Jersey, 2008.

[4] G.E.P. Box, G.M. Jenkins and G.C. Reinsel, *Time Series Analysis: Forecasting and Control*, Holden-Day, San Francisco, CA, USA, 1970.

[5] L. Breiman, J. Friedman, C.J. Stone and R.A. Olshen, *Classification and Regression Trees*, Chapman and Hall, 1984.

[6] P. Civicioglu, Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm, *Comput. Geosci.* **46** (2012), 229–247.

[7] R.B. Cleveland, W.S. Cleveland, J.E. McRae and I. Terpenning, STL: A Seasonal-Trend Decomposition Procedure Based on Loess, *J. Off. Stat.* **6** (1990), 3–73.

[8] M. Dorigo, V. Maniezzo and A. Colorni, Ant system: optimization by a colony of cooperating agents, *IEEE Trans. Syst. Man Cybern. Part B* **26** (1996), 29–41.

[9] R. Eberhart and J. Kennedy, A new optimizer using particle swarm theory, in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, IEEE, 1995, pp. 39–43.

[10] I. Fister Jr., X.-S. Yang, I. Fister, J. Brest and D. Fister, A Brief Review of Nature-Inspired Algorithms for Optimization, *Elektroteh. Vestn.* **80** (2013), 1–7.

[11] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy and A. Bouchachia, A survey on concept drift adaptation, *ACM Comput. Surv.* **46** (2014), 1–37.

[12] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Boston, MA, 1989.

[13] G. Grmanová, V. Rozinajová, A.B. Ezzedine, M. Lucká, P. Lacko, M. Lóderer, P. Vrablecová and P. Laurinec, Application of Biologically Inspired Methods to Improve Adaptive Ensemble Learning, in *Proceedings of the 7th World Congress on Nature and Biologically Inspired Computing (NaBIC2015) in Pietermaritzburg, South Africa, held December 01-03, 2015*,, Pillay N, Engelbrecht AP, Abraham A, du Plessis MC, Snášel V, Muda AK, eds., Springer International Publishing, Cham, 2016, pp. 235–246.

[14] L. Guo, J. Hao and M. Liu, An incremental extreme learning machine for online sequential learning problems, *Neurocomputing* **128** (2014), 50–58.

[15] E. Hadavandi, A. Ghanbari and S. Abbasian-Naghneh, Developing a Time Series Model Based on Particle Swarm Optimization for Gold Price Forecasting, in *2010 Third International Conference on Business Intelligence and Financial Engineering*, IEEE, 2010, pp. 337–340.

[16] A. Halim, Hanif and I. Ismail, Bio-Inspired Optimization Method: A Review, *Int. J. Artif. Intell. (NNGT 2014)* **1** (2014), 6.

[17] J. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, MA, 1992.

[18] W.-C. Hong, *Intelligent Energy Demand Forecasting*, Springer, 2013.

[19] R.J. Hyndman, G. Athanasopoulos, S. Razbash, D. Schmidt, Z. Zhou, Y. Khan, C. Bergmeir and E. Wang, *Package "forecast,"* 2015.

[20] A. Chitra and S. Uma, An Ensemble Model of Multiple Classifiers for Time Series Prediction, *Int. J. Comput. Theory Eng.* **2** (2010), 454–458.

[21] E. Ikonomovska, J. Gama and S. Džeroski, Learning model trees from evolving data streams, *Data Min. Knowl. Discov.* **23** (2011), 128–168.

[22] D. Karaboga and B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *J. Glob. Optim.* **39** (2007), 459–471.

[23] A. Karatzoglou, A. Smola, K. Hornik and A. Zeileis, kernlab - An S4 Package for Kernel Methods in R, *J. Stat. Softw.* **11** (2004), 1–20.

[24] J. Kennedy, R.C. Eberhard and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann Publishers, 2001.

[25] J. Kennedy and R. Eberhart, Particle swarm optimization, in *Proceedings of ICNN'95 - International Conference on Neural Networks*, IEEE, 1995, pp. 1942–1948.

[26] L.I. Kuncheva and C.J. Whitaker, Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy, *Mach. Learn.* **51** (2003), 181–207.

[27] V. Kvasnčka, J. Pospíchal and P. Tiňo, *Evolučné algoritmy*, STU Bratislava, 2000.

[28] A. Liaw and M. Wiener, Classification and Regression by randomForest, *R News* **2** (2002), 18–22.

[29] T.P.F. Lima and T.B. Ludermir, An automatic method for construction of ensembles to time series prediction, *Int. J. Hybrid Intell. Syst.* **10** (2013), 191–203.

[30] R.H. MacArthur and E.O. Wilson, *The Theory of Island Biogeography*, Princeton University Press, 1967.

[31] G. Martinez-Muoz, D. Hernandez-Lobato and A. Suarez, An Analysis of Ensemble Pruning Techniques Based on Ordered Aggregation, *IEEE Trans. Pattern Anal. Mach. Intell.* **31** (2009), 245–259.

[32] L.L. Minku, *Online ensemble learning in the presence of concept drift*, University of Birmingham, Birmingham, UK, 2011.

[33] S. Mirjalili, S.M. Mirjalili and A. Lewis, Grey Wolf Optimizer, *Adv. Eng. Softw.* **69** (2014), 46–61.

[34] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira and L. Damas, On predicting the taxi-passenger demand: A real-time approach, in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013, pp. 54–65.

[35] H. Mühlenbein and D. Schlierkamp-Voosen, Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization, *Evol. Comput.* **1** (1993), 25–49.

[36] K. Mullen, D. Ardia, D. Gil, D. Windover and J. Cline, DEoptim : An R Package for Global Optimization by Differential Evolution, *J. Stat. Softw.* **40** (2011), 10.

[37] R. Nau, Moving average and exponential smoothing models, http://people.duke.edu/~rnau/411avg.htm, Last updated 2016, Accessed on February 23, 2016.

[38] S. Nikumbh, bbo: Biogeography-Based Optimization. R package version 0.2, (2014), 7.

[39] N.C. Oza and S. Russell, Experimental comparisons of online and batch versions of bagging and boosting, in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '01*, 2001, pp. 359–364.

[40] N.C. Oza, Online bagging and boosting, in *2005 IEEE International Conference on Systems, Man and Cybernetics*, 2005.

[41] S. Reid, *A Review of Heterogeneous Ensemble Methods*, University of Colorado at Boulder, Boulder, CO, 2007.

[42] W. Shen, V. Babushkin, Z. Aung and W.L. Woon, An ensemble model for day-ahead electricity demand time series forecasting, in *Proceedings of the the fourth international conference on Future energy systems - e-Energy '13*, 2013, p. 51.

[43] D. Simon, Biogeography-Based Optimization, *IEEE Trans. Evol. Comput.* **12** (2008), 702–713.

[44] J. Soto, P. Melin and O. Castillo, Time series prediction using ensembles of ANFIS models with genetic optimization of interval type-2 and type-1 fuzzy integrators, *Int. J. Hybrid Intell. Syst.* **11** (2014), 211–226.

[45] R. Storn and K. Price, Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *J. Glob. Optim.* **11** (1997), 341–359.

[46] Z.S. Taghavi and H. Sajedi, Ensemble pruning based on oblivious Chained Tabu Searches, *Int. J. Hybrid Intell. Syst.* **12** (2016), 131–143.

[47] T. Therneau, B. Atkinson and B. Ripley, rpart: Recursive Partitioning and Regression Trees. R package version 4.1-8, (2014), 34.

[48] G.K.F. Tso and K.K.W. Yau, Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks, *Energy* **32** (2007), 1761–1768.

[49] Y. Wang, J. Lv, L. Zhu and Y. Ma, Crystal structure prediction via particle-swarm optimization, *Phys. Rev. B* **82** (2010), 094116.

[50] J.D. Wichard and M. Ogorzałek, Time series prediction with ensemble models, in *IEEE International Conference on Neural Networks - Conference Proceedings*, 2004, pp. 1625–1630.

[51] L. Xiao, J. Wang, R. Hou and J. Wu, A combined model based on data pre-analysis and weight coefficients optimization for electrical load forecasting, *Energy* **82** (2015), 524–549.

[52] W. Xie, S. Uhlmann, S. Kiranyaz and M. Gabbouj, Incremental Learning with Support Vector Data Description, in *2014 22nd International Conference on Pattern Recognition*, 2014, pp. 3904–3909.

[53] G.V. Yon and E. Muñoz, ABCoptim: Implementation of Artificial Bee Colony (ABC) Optimization. R package version 0.13.11, (2013), 5.

[54] W. Zang, P. Zhang, C. Zhou and L. Guo, Comparative study between incremental and ensemble learning on data streams: Case study, *J. Big Data* **1** (2014), 5.

[55] F. Zhang, H. Dai and D. Tang, A Conjunction Method of Wavelet Transform-Particle Swarm Optimization-Support Vector Machine for Streamflow Forecasting, *J. Appl. Math.* **2014** (2014), 1–10.

[56] P. Zhang, X. Zhu, Y. Shi, L. Guo and X. Wu, Robust ensemble learning for mining noisy data streams, *Decis. Support Syst.* **50** (2011), 469–479.

[57] P. Zhang, X. Zhu and Y. Shi, Categorizing and mining concept drifting data streams, in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 08*, ACM Press, New York, New York, USA, 2008, p. 812.

Table 1. Average daily MAPE (%) of base prediction models (regression based model, time series analysis models and AI based models). Each type of concept drift pattern has 3 experimental datasets.

type	MLR	RW	STL +ARIMA	STL+HW +ANN+MLR	MA	ANN	SVR	RF
1	5.768	4.179	**3.991**	4.358	5.899	4.633	6.171	5.769
	5.941	5.221	**4.231**	4.915	7.092	9.201	6.072	5.843
	3.825	4.146	4.006	3.858	6.703	4.416	3.951	3.826
2	4.296	4.494	4.828	4.592	4.367	4.973	**3.972**	4.289
	5.152	4.701	5.132	**4.584**	4.586	4.849	5.511	5.154
	8.091	5.686	6.198	6.007	**5.581**	6.400	8.476	8.097
3	4.846	3.764	3.963	**3.688**	3.776	3.876	4.865	4.845
	7.046	5.899	5.860	6.129	9.919	**5.847**	7.257	7.049
	7.875	**5.744**	6.087	6.097	7.696	13.315	8.118	7.357
4	6.076	5.034	5.315	4.908	**4.784**	5.157	6.203	6.072
	6.032	5.227	**4.803**	5.076	7.224	11.035	6.228	5.900
	8.646	8.712	10.585	8.963	12.759	13.364	**8.502**	10.969

Table 2. Average daily MAPE (%) of ensemble using different methods of model weighting (swarm intelligence, evolution and ecology inspired methods). Each type of concept drift pattern has 3 experimental datasets.

type	ABC	GWO	PSO	GA	DE	BBO
1	4.082	3.923	4.012	**3.822**	4.106	4.070
	4.493	4.463	4.483	**4.341**	4.498	4.545
	3.577	**3.535**	3.582	3.581	3.569	3.597
2	4.235	4.237	4.230	**4.203**	4.231	4.229
	3.813	3.893	**3.799**	3.976	3.812	3.874
	5.251	5.243	5.290	5.614	5.268	**5.195**
3	3.072	**3.064**	3.086	3.185	3.105	3.208
	5.500	5.555	5.511	5.708	**5.495**	5.581
	5.275	5.338	5.285	5.292	5.278	**5.263**
4	**4.419**	4.442	4.424	4.441	4.412	4.375
	4.401	4.386	4.407	4.530	**4.380**	4.418
	8.566	**8.347**	8.418	8.782	8.524	8.677

Fig. 1. Schematic of ensemble learning.

Fig. 2. Examples of concept drift patterns.

Fig. 3. Average MAPE of base prediction methods and biologically inspired algorithms of all datasets.

Fig. 4. Relationship between prediction error and number of iterations in various biologically inspired algorithms.

Fig. 5. Average runtime of BI algorithms.

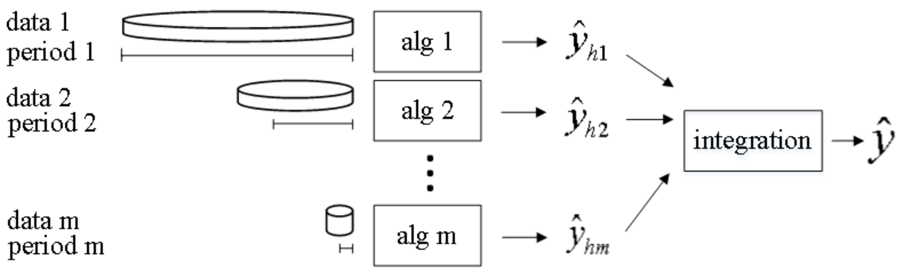

Fig. 1. Schematic of ensemble learning.

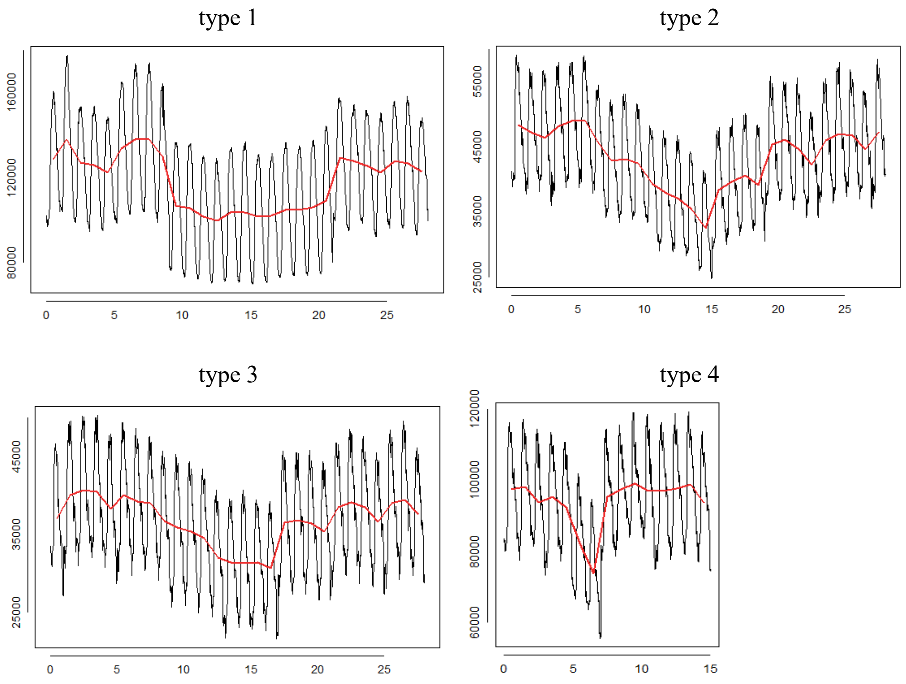

Fig. 2. Examples of concept drift patterns.

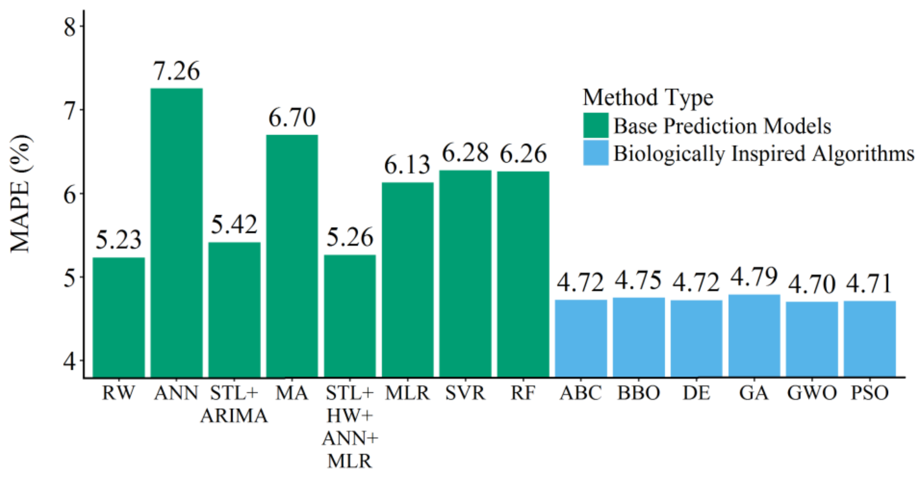

Fig. 3. Average MAPE of base prediction methods and biologically inspired algorithms of all datasets.

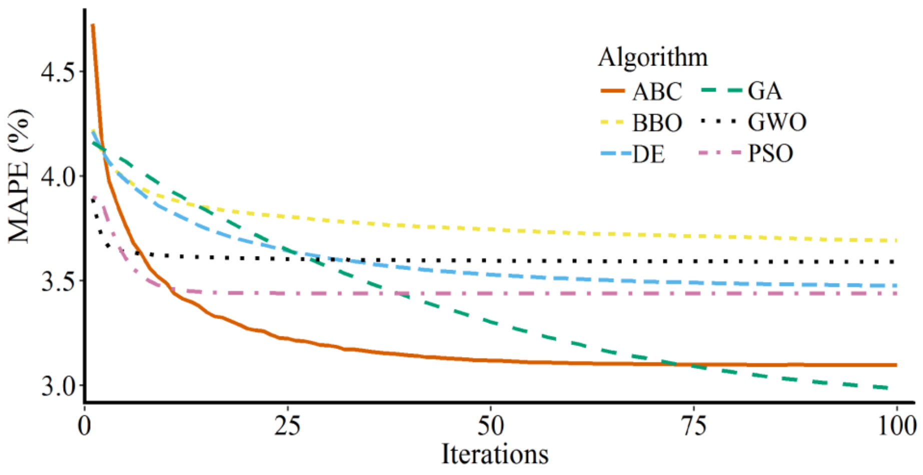

Fig. 4. Relationship between prediction error and number of iterations in various biologically inspired algorithms.

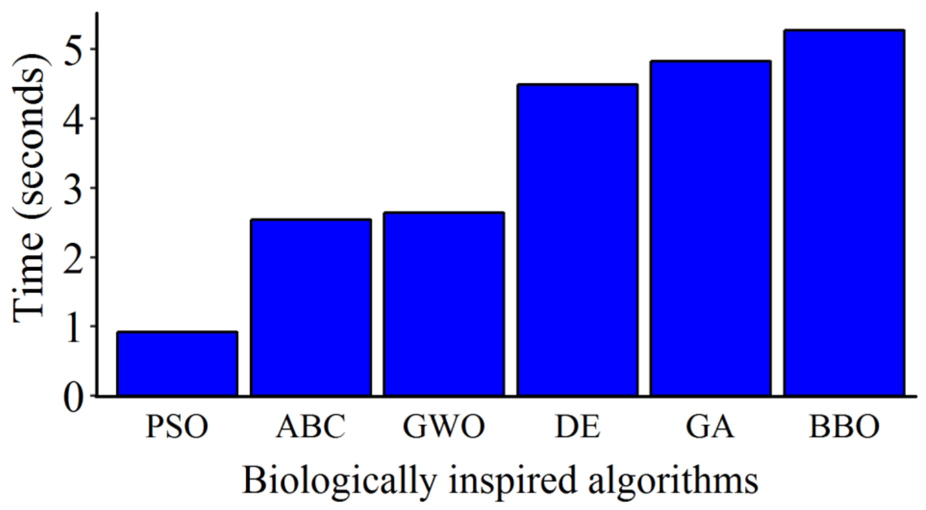

Fig. 5. Average runtime of BI algorithms.