A Reuse-based Environment to Build Ensembles for Time Series Forecasting

Claudio V. Ribeiro
EMGEPRON/Engineering Department, Rio de Janeiro/RJ, Brazil
Email: claudiovas02@yahoo.com.br

Ronaldo R. Goldschmidt
UFRRJ/Technology and Languages Department, Nova Iguaçu/RJ, Brazil
Email: ronaldo.rgold@gmail.com

Ricardo Choren
IME/Computer Engineering Department, Rio de Janeiro/RJ, Brazil
Email: choren@ime.eb.br

***Abstract*—Several works show that ensembles improve the performance of time series forecasting solutions. However, developing an ensemble is not an easy task. Usually, the analyst has to develop each ensemble as a separate project, designing, implementing and configuring the individual and the ensemble methods for each experiment. This paper proposes a change to this common view. It argues that it is possible and necessary to also look from a reuse perspective. Combining ideas from reuse and time series forecasting requirements, this paper proposes an environment to enable reusability for ensemble development. The environment intends to provide a flexible tool for the analyst to include, configure and execute individual methods and to build and execute ensemble experiments.**

***Index Terms*—ensembles, time series forecasting, reuse, environment**

I. INTRODUCTION

The ability to model and perform decision modeling and analysis is an essential feature of many applications, e.g. financial trading, energy and water distribution, and military command systems. In such systems, almost all managerial decisions are based on forecasts: the decision-maker uses forecasting models to assist in decision-making process. Indeed, forecasts are needed continually and their impact on actual performance should be regularly measured.

In this sense, time series forecasting methods intend to predict, with the best accuracy possible, future unknown data values based on historical patterns in the existing data. A number of techniques have been developed in an attempt to predict time series from a simple linear Autoregressive Moving Average models (ARMA) [1] through conditional models like ARCH or GARCH [1] up to the complex nonlinear models [2]

A number of machine learning techniques have been widely used as a promising alternative approach to time series forecasting and on a number of occasions showed considerable improvement compared to traditional regression models [3]. Neural networks are particularly good at capturing complex nonlinear characteristics of time series [4]. The most popular neural networks used in forecasting are the single multi-layer feed forward model or multi-layer perceptron (MLP) [5]. However, since there is no guideline for choosing the appropriate MLP model structure for practical applications. A trial-and-error approach or cross-validation experiment is often adopted to help find the 'best' model. Typically a large number of neural network models are considered: the one with the best performance in the validation set is chosen as the winner, and the others are discarded [5].

Moreover, it is generally accepted that every time series forecasting technique has its weaknesses at some aspects. In this sense, individual models that work alone are usually regarded as unable to produce satisfactory results [6]. Thus it is natural to consider combining multiple models to generate better data forecasting. Such a combined system is commonly referred as an ensemble. Ensemble methods aim at leveraging the performance of a set of models to achieve better forecasting accuracy than that of the individual models [7]. Ensembles have been explored by many researchers such as [8, 9, 10].

Nonetheless, the trial-and-error approach is still adopted when using ensemble methods, with another drawback: it is also necessary to develop the ensemble model. Thus the analyst will need to develop not only a large number of individual methods, but possibly a number of combination methods to experiment with ensembles. When an analyst wants to experiment with particular algorithms to create ensembles, it is usually necessary to build each ensemble separately, viewing each one as a different analysis project.

Software reuse is the process of building or assembling software applications from previously developed software parts designed for reuse (assets). Therefore, almost any software application can be assembled from predefined assets, provided those assets were designed to be "plug compatible". Usually, software reuse is practiced to save time and money, and to improve quality [11].

This paper presents an environment for ensemble configuration and execution that uses a reuse perspective. The idea is to indicate the features of an ensemble application for time series forecasting and to use reusable assets to implement those features. The use of features is motivated by the fact that engineers often speak of product characteristics in terms of "features the product has and/or delivers" [12]. They expose requirements or functions in terms of features and, to them, features are distinctively identifiable functional abstractions that must be implemented.

The environment allows for individual methods development, configuration, selection and execution. Then the environment allows for combination method development, configuration and execution. Besides, the environment allows for result analysis, providing a way to compare different ensemble experiments. The idea is to provide a flexible tool that can reuse method components to experiment with several configurations of ensembles for time series forecasting.

The remainder of the paper is organized as follows. In the next section, the theory underlying ensembles and time series forecasting is briefly reviewed. In the section that follows, the research methodology is covered. This includes description of the features and a detailed discussion of the reuse approach employed. Section V contains the experimental results of two simple ensemble exercises designed to use the current method tool and the final section presents the conclusions.

II. Ensembles and Time Series Forecasting

Time-series forecasting is an important research and application area. Much effort has been devoted over the past several decades to develop and improve the time-series forecasting models [13]. Recently a tendency for combining of linear and nonlinear models for forecasting time series has been an active research area [14].

Most ensembles for time series forecasting are based on artificial neural networks. For example, Kuan and Liu [15] examined the forecasting ability of neural networks on five exchange rates against the US dollar. Borisov and Pavlov [16] applied neural networks to forecast the Russian ruble exchange rate. Both neural networks and exponential smoothing models were used to predict the exchange rates. Zhang, [14] proposed to use a hybrid ARIMA (Box–Jenkins model) and neural network model for time-series forecasting problem, applying it to the Mackey-Glass time series.

Such works show the potential of using ensembles for time series forecasting, but they only investigate the potential of combining multiple neural network models for time series forecasting. Thus they try to show that, by appropriately combining different neural forecasters, forecasting performance of the individual network can be improved.

The ensemble idea is to combine several forecasters. However, these forecasters may not all be instances of the same algorithm. In this sense, the ensemble is used for fine-tuning, since small changes in the training set and/or parameter selection of a neural network, for example, can produce large changes in the predicted output.

III. The Reuse-Based Environment

This section describes the proposed reuse-based environment for time series forecasting using ensembles. The environment divides the ensemble development in two levels: individual and combination (Fig. 1). It provides five main features: data set division; individual method development, configuration and execution; method selection; combination strategy configuration and execution, and; forecasting analysis.

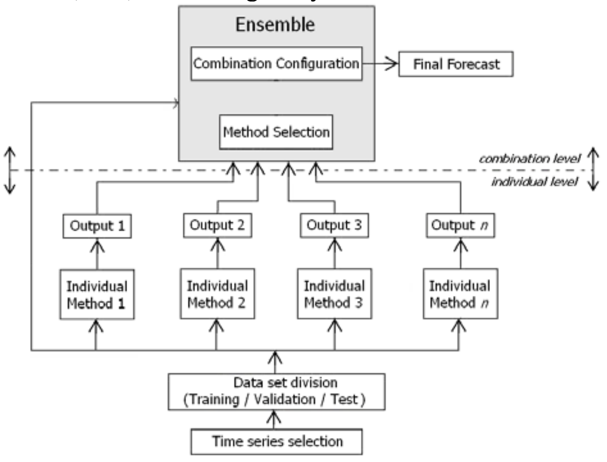


Figure 1. The environment general architecture.

A. Time Series Selection and Data Set Division

To start developing an ensemble using the proposed environment, the analyst should select the time series. The environment requires the time series to be stored in a database so that it can read its schema. Since the environment was developed using Java, any DBMS which has a Java-based connection driver can be used. The environment reads the schema and presents the series attributes. The analyst needs to select at least two attributes: one that will indicate the time ordering, and; one that will be predicted.

After selecting the time series, the analyst should indicate the data sets that will be used by both the individual and combination strategy methods. These methods can use two or three sets to execute: a training set; a validation set (optional), and; a test set. The training set (seen data) is used to build the model (i.e. to determine its parameters) and the test set (unseen data) to measure its performance, holding the parameters constant. Sometimes, the analyst also needs a validation set to tune the model. The validation set cannot be used for testing (as it's not unseen). All three data set have to be representative samples of the data that the model will be applied to.

The environment allows the analyst to divide the time series data set into three different strategies, as seen in Fig. 2. The first strategy divides the available data into two sets: training and test. The second strategy defines the three possible sets, but uses the same set of data for the training and the validating phases. Finally, the third strategy defines different sets for each phase.

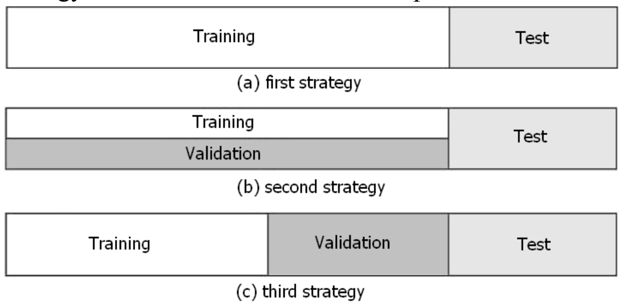


Figure 2. Data set division strategies.

B. Individual Method Features

The proposed environment intends to provide a reuse approach for ensemble development. As an ensemble is composed of individual methods, these methods should be developed for reuse. This is mostly because it would be almost impossible to provide a complete list of individual method implementations. Thus, such methods should be designed and implemented as configurable components.

The re-usability of components depends on the support of component abstraction in order to make components available through libraries and on the support of adaptation techniques to adapt library components to actual requirements, i.e. to glue service provider and user together [17]. The proposed environment provides an abstract interface to deliver a component abstraction. Every individual method should be developed according to this interface in order to be able to be included in the environment library and thus be reused in different ensemble projects.

This feature was developed using the dependency injection (or inversion of control) approach [18]. In this approach, the developer of the method implementation should extend a predefined interface (i.e. the environment uses interface injection). Without dependency injection, a consumer (combination) component that needs a particular provider (individual method) in order to accomplish a combination task will depend not only on the interface of the service, but on the details of a particular implementation of it as well. The user component has to handle both its use and the life cycle of that service – creating an instance, opening and closing streams, etc. Using dependency injection, however, the life-cycle of a service is handled by a dependency provider rather than the consumer. The dependency provider is an independent, external component that is part of the proposed environment and that links the consuming component and the providing component. The consumer would thus only need a reference to an implementation of the service that it needed in order to accomplish the necessary task.

Since the environment is the dependency provider and any individual method must implement a predefined interface, it is possible to populate an environment method library. Whenever an analyst needs a method to create an ensemble, it should be possible to select and configure a method component of the library.

The analyst can select any method in the library to be an individual method in an ensemble. After selecting the method, the analyst needs to configure its parameters. Although the methods have the same interface, which will be used by the environment to invoke its execution (inversion of control), the configuration is particular for each method. In this sense, besides implementing the same interface, the method developer should provide a configuration interface for the method.

Once the individual methods are configured, they can be executed. The environment controls this execution and it stores the execution results. The environment provides an interface for the analyst to view the execution results of the individual methods in a graphical way. These results are also available in a table format for analysis *a posteriori*

Storing the execution of a particular configuration of an individual method execution also fosters reusability. If an analyst wants to perform ensembles forecasting experiments with a time series, the prior results can be directly reused, without the need to re-execute a particular method configuration.

C. Individual Method Selection

The individual method results should have their performance analyzed so that the analyst can make an informed decision as to whether a particular method will be a seed for the ensemble (combination) or not. In this sense, the environment provides five approaches to help an analyst to select which individual methods will be used in the ensemble. The approaches are:

- *All methods selection*: in this approach, the results of all the executed individual methods will be used in the ensemble combination method;
- *Particular method selection*: in this approach, the analyst will specifically indicate which individual methods will be used in the ensemble combination method;
- *Maximum error index selection*: in this approach, the analyst indicates, to the environment, a value for a maximum error index. Then, the environment will automatically select those individual methods that presented results with an error smaller than the value provided by the analyst. This approach can only be used if the time series data set was divided with a validation set (second and third division strategies – see Fig. 2) since the error index of an individual method will be calculated using the data in this set;
- *Percentage error index selection*: this approach is very similar to the above. The difference is that the analyst indicates a percentage error index, instead of indicating an absolute error index;
- *Automatic selection*: in this approach, the analyst will allow the environment to automatically select the individual methods that should be used in the ensemble. The environment executes a series of simulations, using simple averaging combination,

to assess the combination of the available results. The simulation uses the following process: (i) a temporary result set is created by combining, with simple averaging, the results of all individual methods; (ii) iteratively, the result of an individual method is removed and a new combination is done, generating another temporary result set. If this result set is better than the created in step (i), this method is discarded definitely. Otherwise, the method remains and another method is chosen to re-execute step (ii). The iteration continues until every method has been tested for removal.

D. Combination Strategy Features

The environment provides two types of combination strategies which the analyst can select: linear and nonlinear combination. The provided linear combination strategies implement simple combination methods, based on averaging. There are two linear combination strategies in the environment: simple and weighted averaging. Simple averaging calculates an average of the results computed by the selected individual methods for a given time index. This calculation is the result of the ensemble forecast.

Weighted averaging weights each selected individual method result in the average computation. Weights are calculated using a prediction error metric selected by the analyst. The environment provides six prediction error metrics for selection: U-Theil coefficient, mean squared error (MSE); root mean squared error (RMSE); sum of squares error (SSE); mean absolute error (MAE), and; mean absolute percentage error (MAPE). The weights are calculated using:

$$W_i = \frac{1/V_i}{\sum_1^n 1/V_i}. \quad (1)$$

Where:

- V_i: is the value calculated by the metric for the ith individual method result;
- *n:* is the number of selected individual methods selected to be used in the ensemble;
- *Wi:* is the weight for individual method *i*.

Table 1 shows an example of weight computation.

TABLE 1.

SAMPLE WEIGHT COMPUTATION.

Method	V_i	Method weight (for weighted averaging)
Method A	0,2	5,0 / (5,0 + 2,5 + 0,5) = 0,625
Method B	0,4	2,5 / (5,0 + 2,5 + 0,5) = 0,3125
Method C	2,0	0,5 / (5,0 + 2,5 + 0,5) = 0,0625

The ensemble forecast for time index *k* will then be:

$$f(k) = \sum_1^n S_i(k) * W_i. \quad (2)$$

Where:

- $S_i(k)$: is the prediction for time index *k* calculated by individual method *i*.

If the analyst decides to use a nonlinear combination strategy, any method in the environment method library can be used in the ensemble. In addition, the environment allows the analyst to configure the combination training and input strategies. Since the combination method can be any method from the environment library, the analyst can configure the training strategy, just like it was done for the individual methods (see Fig. 2).

There are two input strategies offered in the environment: simple input and sophisticated input. In the simple input, only the results of the selected individual methods will be used as inputs for the combination method. In the sophisticated input, the time series is also used as input (so the combination method can make an association with the individual results and the series pattern).

Then, the environment executes the combination method. While executing the combination method the environment may need to make some adjustments if some individual method used the prediction window concept (e.g. a regression method). This is because the prediction windows of the individual methods should be aligned to indicate the initial time index to be used in the training phase of the combination method. If the combination method itself uses a prediction window, this window should be considered in the alignment as well. Fig. 4 shows an example of the window alignment phase in the environment.

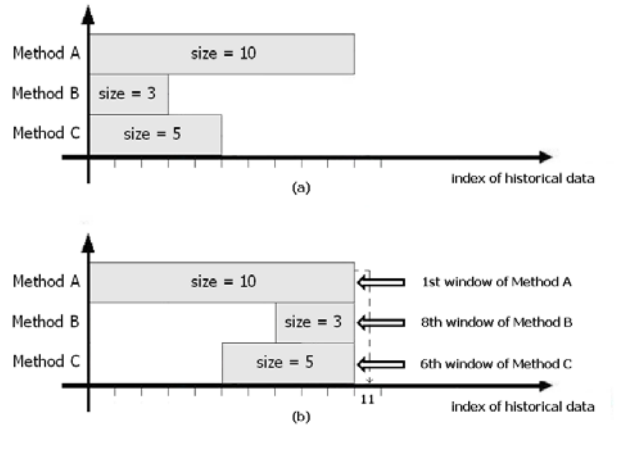


Figure 3. Prediction window alignment.

In Fig. 3(a), the prediction windows of the methods (that could include the combination method) are shown. The prediction window sizes are 10, 3 and 5 for methods A, B and C respectively. Then the first target element method A predicts is the eleventh element. In this sense, methods B and C do not need to use its prediction windows until the eleventh element. Fig. 3(b) shows the final alignment.

E. Ensemble Forecasting Analysis

The results of the combination method execution are stored, similarly to what was done for each individual methods. These results are used for analysis and the environment provides a graphical interface for the analyst to perform the analysis.

The analysis interface is a separate feature of the environment. This means that the analyst can experiment with several ensembles and then open the analysis interfaces of all desired experiments to verify their performance.

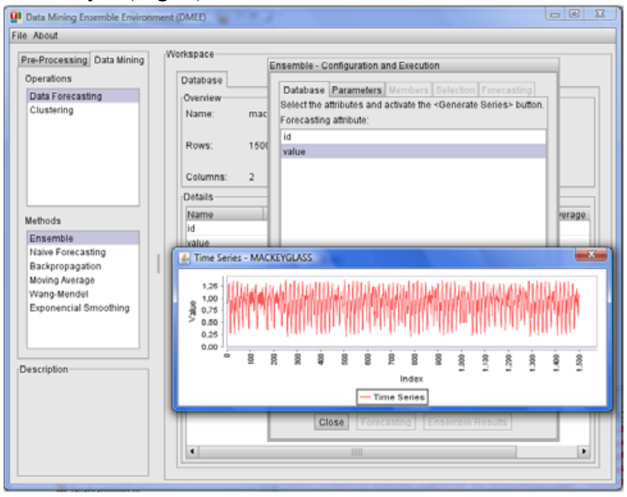
IV. The Environment Prototype

The current version of the environment prototype was developed using Java. There are several frameworks that provide a Java implementation, such as neural networks. Moreover, the popularity of object-oriented development (OOD) brought the notion of reuse to the forefront.

Currently, the method library is not a whole separate component of the environment. Thus it is not possible to create an integrated library of the new components. The component interface is provided and the developer should develop the method according to the provided interface. Then it is necessary to build the environment again so that it can recognize the new added method. There is a build mechanism that helps the developer to add the method component into the environment. Moreover, the environment already has some 'pre-installed' methods that include Naïve Forecasting, Moving Average, Exponential Smoothing, Wang-Mendel Method [19] and Backpropagation.

To begin using the environment prototype for building an ensemble for time series forecasting, the analyst should create an ensemble project. This project is a space that will hold all configuration information about the experiment, such as the time series data, the individual methods and their configuration, and so on.

Once defined the ensemble project, the analyst should select the time series (i.e., indicate the database that stores the time series data). Once the analyst selected the database containing the time series, the environment presents a list of attributes. The analyst must indicate which attribute is to be predicted and which one should be considered the time index. After this initial configuration the prototype shows the time series data to the analyst (Fig. 4).


Data Mining Ensemble Environment (DMEE)
File About
Pre-Processing Data Mining
Operations
Data Forecasting
Clustering
Methods
Ensemble
Naive Forecasting
Backpropagation
Moving Average
Wang-Mendel
Exponencial Smoothing
Description
Workspace
Database
Overview
Details
Name
Ensemble - Configuration and Execution
Database Parameters Members Selection Forecasting
Time Series - MACKEYGLASS
Value
Index
Time Series
Close Forecasting Ensemble Results
Name: mac
Rows: 150
Columns: 2

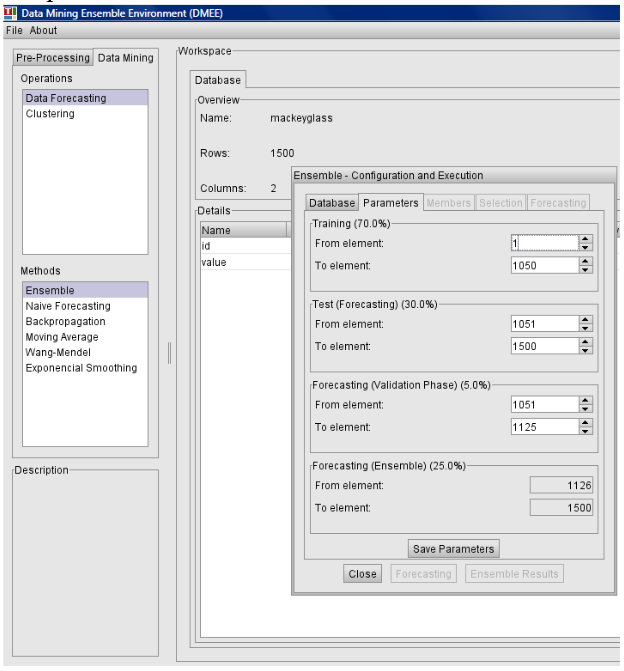
Select the attributes and activate the <Generate Series> button.
Forecasting attribute:
id
value

Figure 4. Selecting a time series.

Then, it is necessary to divide the data set. The analyst must indicate a training set and a test set and, optionally, indicate a validation set. To do so, the analyst should indicate the indexes stating the start and end tuples of the database that describe each subset. These indexes refer to the time index attribute selected previously. Fig. 5 shows the subset configuration in the prototype. The percentages are automatically calculated according to the record spam of the provided indexes.

Both features (time series data import and data set division) are provided directly by the environment. This information is stored and it will be used by all individual methods that will compose the ensemble.

Then, the analyst should select and configure the individual methods to be used. As mentioned before, method configuration interfaces should be done along with the method development. In this paper, we will discuss only the interfaces of the pre-installed method components.



Name: mackeyglass
Rows: 1500
Columns: 2

Training (70.0%)
From element: 1
To element: 1050

Test (Forecasting) (30.0%)
From element: 1051
To element: 1500

Forecasting (Validation Phase) (5.0%)
From element: 1051
To element: 1125

Forecasting (Ensemble) (25.0%)
From element: 1126
To element: 1500

Figure 5. Data set division.

For the Naive Forecasting, there are no parameters to configure since it predicts the value of k_{i+1} element as being the same as the k_i element. For the Moving Average, the analyst should configure the prediction window. For the Exponential Smoothing, the analyst should configure the smoothing factor (α). The formula for the Exponential Smoothing component method in the environment prototype implementation is the following:

$$\begin{aligned} S_0 &= X_0 \\ S_t &= S_{t-1} + \alpha * (X_t - S_{t-1}) \end{aligned} \quad (3)$$

The analyst must configure the number of sets and the size of the prediction window for the Wang-Mendel method. In the available implementation of the method, the membership function is triangular. Finally, for the Backpropagation method, the analyst should configure the parameters for a multilayer perceptrons neural network, including number of epochs, type of activate functions, momentum term, learning rate, and so on.

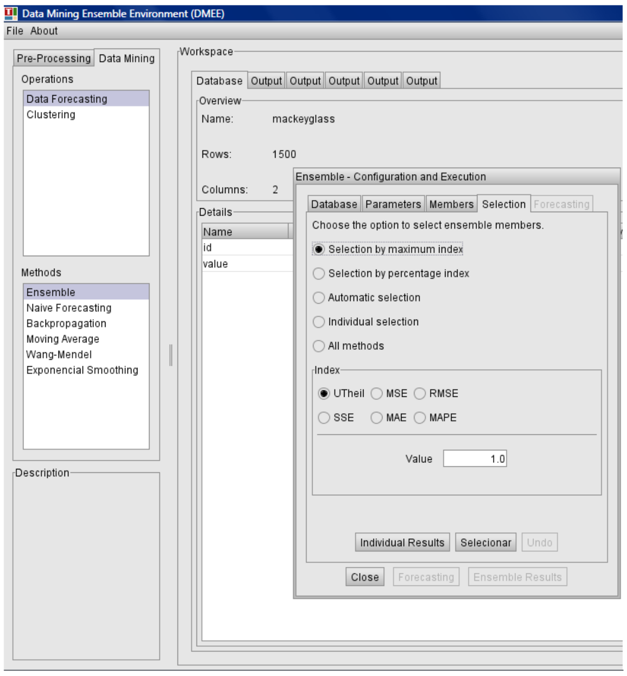
The analyst can configure as many individual methods as it is necessary, regardless of their type. For instance, the analyst can configure two backpropagation methods to be used in the ensemble experiment. This configuration is stored at the ensemble project.

After configuring the individual methods, the analyst can have the environment execute them. Using the inversion of control mechanism, the environment starts the execution of each selected and configured individual method and stores its results. Currently, the environment does not present a multi-thread execution mechanism, which means that each individual method will be executed serially. The results are stored and they can be used in other ensemble experiments. For instance, the analyst creates an ensemble project with two individual methods. These methods are configured and executed. Then the whole ensemble is executed. If the analyst wants to make another experiment using the two previous individual methods and add another one, it will only be necessary to configure and execute this distinct individual method. The ensemble method can then be re-executed using this new individual method and the results stored for the previous ones, reusing their execution results.

Before executing the ensemble method, the analyst can make a selection of the individual methods that will be used by the combination method. As mentioned in subsection 3.C, the analyst can use five different strategies to select the subset of individual methods that will be used. In the current version of the prototype, the metrics that can be used if the analyst chooses the selection by a maximum index or the selection by percentage index strategies are: U-Theil; mean square error (MSE); root mean square error (RMSE); sum of squares error (SSE); mean absolute error (MAE), and; mean absolute percentage error (MAPE). Other metrics are foreseen to be included by the analyst in the new prototype version. Fig. 6 and 7 show the selection by a maximum index and the individual selection approach, respectively.

Following the selection of the individual methods, the analyst must configure and execute the combination method. The configuration is done in a similar fashion to the individual methods configuration. At this moment, the analyst can indicate if the time series data will be used as an input to the combination method or if it will only use the results of the selected individual methods. This feature is available in the nonlinear combination strategy.

Then, the environment executes the combination method. If necessary, the environment adjusts the prediction windows before executing. After executing the combination method, the environment stores the results that were generated for analysis.


Data Mining Ensemble Environment (DMEE)
File About
Pre-Processing Data Mining
Operations
Data Forecasting
Clustering
Methods
Ensemble
Naive Forecasting
Backpropagation
Moving Average
Wang-Mendel
Exponencial Smoothing
Description
Workspace
Database Output Output Output Output Output
Overview
Details
Name
id
value
Ensemble - Configuration and Execution
Database Parameters Members Selection Forecasting
Individual Results Selecionar Undo
Close Forecasting Ensemble Results

Name: mackeyglass

Rows: 1500

Columns: 2

Choose the option to select ensemble members.

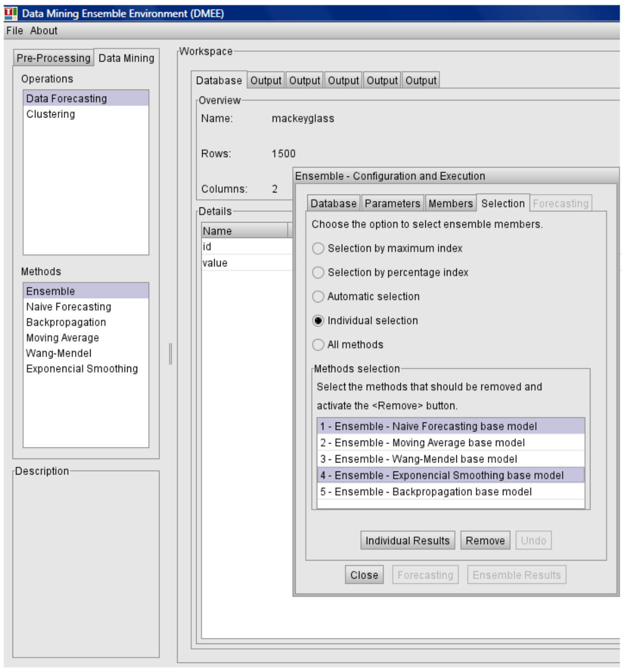
[✓ Selection by maximum index]
Selection by percentage index
Automatic selection
Individual selection
All methods

Index

[✓ UTheil] MSE RMSE
SSE MAE MAPE

Value 1.0

Figure 6. Selection by a maximum index strategy.


Data Mining Ensemble Environment (DMEE)
File About
Pre-Processing Data Mining
Operations
Data Forecasting
Clustering
Methods
Ensemble
Naive Forecasting
Backpropagation
Moving Average
Wang-Mendel
Exponencial Smoothing
Description
Workspace
Database Output Output Output Output Output
Overview
Details
Name
id
value
Ensemble - Configuration and Execution
Database Parameters Members Selection Forecasting
Individual Results Remove Undo
Close Forecasting Ensemble Results

Name: mackeyglass

Rows: 1500

Columns: 2

Choose the option to select ensemble members.

Selection by maximum index
Selection by percentage index
Automatic selection
[✓ Individual selection]
All methods

Methods selection

Select the methods that should be removed and activate the <Remove> button.

1 - Ensemble - Naive Forecasting base model
2 - Ensemble - Moving Average base model
3 - Ensemble - Wang-Mendel base model
4 - Ensemble - Exponencial Smoothing base model
5 - Ensemble - Backpropagation base model

Figure 7. Individual selection strategy.

V. Usage Results

This section discusses the empirical results of the effectiveness of ensemble methods using the proposed environment. It presents the results from two experiments designed to determine whether the ensembles created in the environment are better, or worse, than the individual methods currently provided by the environment (see section 4). The first example shows an ensemble for the forecasting of the Mackey-Glass time series [20]. The second illustrates an example of the Sun Spots time series [20] forecasting.

A. *The Mackey-Glass Experiment*

The data used in the Mackey-Glass experiment encompassed 1500 records: the lowest series value was 0.212559300 and the highest value was 1.378507200. In this experiment, all available individual methods were used, with the following configurations:

- *Naïve Forecasting*: no configuration needed;
- *Moving Average*: prediction window size = 3;
- *Exponential Smoothing*: smoothing factor = 0,9;
- *Wang-Mendel Method*:
  prediction window size = 7;
  number of fuzzy sets = 7;
- *Backpropagation*:
  learning rate = 0,6;
  momentum factor = 0,3;
  number of epochs = 5000;
  input layer:
    number of neurons = 9;
    activation function = linear;
  hidden layer:
    number of neurons = 9;
    activation function = sigmoid;
  output layer:
    number of neurons = 1;
    activation function = linear;

For the individual methods, the series data were divided into the following sets (according to strategy depicted in Fig. 2(b)): 70% training; 70% validation, and; 30% test. For the combination method, the series data were divided into the following sets (according to strategy depicted in Fig. 2(a)): 70 % training and; 30% test.

Table 2 reports the detailed results of the individual methods. Overall, from Table 2, it is observed that the backpropagation method provided the best individual result, while the moving average method provided the worst result.

TABLE 2.

INDIVIDUAL LEVEL METHOD FORECASTING RESULTS.

	Sets - Training: 70%; Test: 30%							
	Validation				Test			
Method	U-Theil	MSE	MAE	MAPE	U-Theil	MSE	MAE	MAPE
Naive	1	0,0296	0,1437	18,83%	1	0,0294	0,1425	18,88%
Exponential Smothing	1,0509	0,0327	0,1516	20,07%	0,0540	0,0327	0,1509	20,18%
Moving Average	1,5305	0,0695	0,2209	30,92%	1,5564	0,0713	0,2237	31,70%
Wang-Mendel	0,2466	0,0018	0,0344	4,37%	0,2909	0,0025	0,0392	4,99%
Backpropagation	0,0602	1,0794E-4	0,0081	1,01%	0,0645	1,2231E-4	0,0085	1,05%

For the ensemble experiment, it was decided to use the nonlinear combination (backpropagation algorithm). This decision is due to the fact that the Linear combination will not produce good results when the most individual results are weak.. It was also decided to use all individual method results as input to the ensemble. The results of the three different configurations of the backpropagation ensemble models are given in Table 3. Although the differences between the ensembles and the best individual method are relatively small, the ensemble method consistently gives better performance.

TABLE 3.

ENSEMBLE FORECASTING RESULTS.

	Sets - Training: 70%; Test: 30% (E = 5000; T = 0,6; M = 0,3)			
	Results			
Ensemble	U-Theil	MSE	MAE	MAPE
5L-5S-5S-1L	0,0635	1,1861E-4	0,0082	1,02%
6L-5S-5S-1L	0,0618	1,1239E-4	0,0079	0,99%
14L-5S-5S-1L	0,0560	9,2312E-5	0,0069	0,85%

B. *The Sun Spots Experiment*

The second experiment used the Sun Spots time series and was used to experiment with only backpropagation methods (both at individual and combination levels). Given the apparent autocorrelations of time series data, the predictions from backpropagation methods built from the same data are often highly correlated, which reduces the effectiveness of the ensemble method [5]. Therefore, this experiment was used to verify if the ensemble is still a good solution.

This experiment used five different configurations of the backpropagation method as individuals. Their configurations were the following:

- *Backpropagation*: configuration number 1
  learning rate = 0,9;
  momentum factor = 0,1;
  number of epochs = 3000;
  input layer:
    number of neurons = 4;
    activation function = linear;
  hidden layer:
    number of neurons = 5;
    activation function = sigmoid;
  output layer:
    number of neurons = 1;
    activation function = linear;
- *Backpropagation*: configuration number 2
  learning rate = 0,9;
  momentum factor = 0,1;
  number of epochs = 3000;
  input layer:
    number of neurons = 6;

activation function = linear;
hidden layer:
number of neurons = 5;
activation function = sigmoid;
output layer:
number of neurons = 1;
activation function = linear;

- *Backpropagation*: configuration number 3
  learning rate = 0,9;
  momentum factor = 0,1;
  number of epochs = 3000;
  input layer:
  number of neurons = 7;
  activation function = linear;
  hidden layer:
  number of neurons = 8;
  activation function = sigmoid;
  output layer:
  number of neurons = 1;
  activation function = linear;
- *Backpropagation*: configuration number 4
  learning rate = 0,9;
  momentum factor = 0,1;
  number of epochs = 3000;
  input layer:
  number of neurons = 6;
  activation function = linear;
  hidden layer number 1:
  number of neurons = 7;
  activation function = sigmoid;
  hidden layer number 2:
  number of neurons = 4;
  activation function = sigmoid;
  output layer:
  number of neurons = 1;
  activation function = linear;
- *Backpropagation*: configuration number 5
  learning rate = 0,9;
  momentum factor = 0,1;
  number of epochs = 10000;
  input layer:
  number of neurons = 4;
  activation function = linear;
  hidden layer number 1:
  number of neurons = 6;
  activation function = sigmoid;
  hidden layer number 2:
  number of neurons = 3;
  activation function = sigmoid;
  output layer:
  number of neurons = 1;
  activation function = linear;

For the individual methods, the series data were divided into the following sets (according to strategy depicted in Fig. 2(b)): 85% training; 85% validation, and; 15% test. For the combination method, the series data were divided into the following sets (according to strategy depicted in Fig. 2(a)): 85 % training and; 15% test.

Three backpropagation method ensembles were created for this experiment. The first used only the individual method results. The second one used the individual method results that and another input configured with the smallest individual prediction window size (in this experiment, size 4). Finally, the third one used the individual method results and another input configured with the largest individual prediction window size (in this experiment, size 7).

Table 4 reports the results of the individual methods. Some observations can be made from Table 4. First, the results are very similar in nearly all cases. Secondly, the methods using two hidden layers performed slightly better than the others.

TABLE 4.
INDIVIDUAL LEVEL METHOD FORECASTING RESULTS.

	Sets - Training: 85%; Test: 15%							
	Validation				Test			
Method	U-Theil	MSE	MAE	MAPE	U-Theil	MSE	MAE	MAPE
Backpropagation 01	0,9640	459,29	18,16	138,39%	0,5359	281,45	12,74	34,82%
Backpropagation 02	0,6471	215,76	12,03	90,61%	0,5366	282,18	12,91	40,01%
Backpropagation 03	0,7388	280,56	14,12	114,80%	0,5229	258,01	13,15	33,54%
Backpropagation 04	0,6149	194,84	11,12	66,88%	0,5011	246,10	12,02	26,77%
Backpropagation 05	0,6293	203,51	11,32	67,16%	0,4821	227,79	11,74	23,93%

The results of ensemble models created with the three different configurations of the backpropagation method are given in Table 5. When compared to the individual methods, the first ensemble did not present any increase in the performance, although performed better than the majority. The second and third ensembles performed a little better in almost all the metrics.

TABLE 5.
ENSEMBLE FORECASTING RESULTS.

	Sets - Training: 85%; Test:15% (E = 10.000; T = 0,9; M = 0,1)			
	Results			
Ensemble	U-Theil	MSE	MAE	MAPE
5L-6S-4S-1L	0,5234	268,49	12,92	25,49%
9L-10S-6S-1L	0,5130	257,89	12,41	23,62%
12L-10S-6S-1L	0,4588	206,27	10,77	21,43%

C. Some Discussion

It is likely that ensembles can achieve better results than the best individual found in the forecasting task (as happens in classification), but this needs to be confirmed empirically [21]. Yet, the results in the two experiments indeed show that the ensemble models can have a significant improvement over individual methods.

Although the examples described above intend to evaluate the environment it shows that ensemble models to rime series forecasting have potential. The results obtained from the environment are consistent with the current state of the art:

- *Individual level performance*: individual nonlinear methods present better performance when compared to individual linear methods, especially in non-stationary times series;
- *Linear method performance*: linear combination ensembles present better performance when compared to individual methods which results are near and have small errors;
- *Combination level performance*: nonlinear combination methods show an improvement over linear combination methods despite the kind of the individual level methods used;
- *MLP combination*: MLP combination methods, usually, are the most applicable combination method.

VI. Conclusions

Time series analysis and prediction is an important task in all fields of science for applications like financial forecasting, weather forecasting, electricity power demand forecasting, process monitoring and control, research, medical sciences, etc [22]. There have been several approaches showing that ensembles of neural networks can improve the performance over any one individual.

Results show that by appropriately combining different neural networks, forecasting accuracy of individual networks can be largely improved [5]. However, it is not easy to determine the number of individual forecasting models that should participate in an ensemble [22]. It was reported that, “the more, the best” rule cannot be applied effectively for all circumstances [23]. The number of individual models to be combined for forecasting and their parameters has to be determined. Moreover, in some circumstances, the linear combination approach can be appropriate – thus, it is necessary to experiment with it.

Nonetheless, experimenting is not an easy task. The analyst needs to try several combinations of individual methods, with another number of ensemble methods. Usually, each ensemble experiment is devised from scratch, requiring an extra amount of work. Although several studies evaluates the benefits and potentials of ensemble models for time series forecasting research efforts should also be devoted to the techniques and tools that can further reduce the workload required to experiment with ensembles.

The main idea of this paper is not to investigate several strategies to form ensemble models neither to actually measure their performance improvements. The idea is to show an environment to help the analyst to experiment with ensembles, following a reuse-based approach. Reuse is identified as a promising technology that uses the cumulative experience of previous designs to enhance later designs. In this case, reuse approach is the process of implementing or updating ensemble systems using existing software assets that can be both at individual or combination levels.

This paper describes an approach to the design of tools to help analysts build repositories of method components, locate potentially reusable methods in those repositories and facilitate the development of ensemble experiments. Thus the environment presented here provides features for creating and managing a method implementation library, method configuration and selection, execution results storage and result analysis. All these features have the purpose of providing a tool to help the analyst work of devising ensemble experiments.

There are points for improvements in the proposed environment. The library of methods needs to become more extensible and it could provide retrieval tools to help the analyst to find the needed components. The prototype presented is also just a first approximation and it requires some investment to become a stable usable tool.

Acknowledgment

This work was supported, in part, by a grant from The Brazilian Council for Scientific and Technological Development (CNPq) for Ricardo Choren.

References

[1] R. S. Tsai, "Analysis of financial time series", John Wiley & Sons, 2002.

[2] M. Casdagli, "Nonlinear prediction of chaotic time series", Physica, vol. 35, pp. 335–356, 1989.

[3] F. E. H. Tay and L. J. Cao, "Modified support vector machines in financial time series forecasting," Neurocomputing, vol. 48, n. 1, pp. 847–861, 2002.

[4] Z. Vojinovic, V. Kecman, and R. Seidel, "A data mining approach to financial time series modelling and forecasting," International Journal of Intelligent Systems in Accounting, Finance & Management, vol. 10, n. 4, pp. 225–239, 2001.

[5] G. P. Zhang and V. L. Berardi, "Time series forecasting with neural network ensembles: An application for exchange rate prediction," The Journal of the Operational Research Society, vol. 52, n. 6, pp. 652–664, 2001.

[6] W. Wang, G. Richards, and S. Rea, "Hybrid data mining ensemble for predicting osteoporosis risk," in Proceedings of the 27th IEEE Engineering in Medicine and Biology Conference, 2005, pp. 886–889.

[7] A. Bouchachia and S. Bouchachia, "Ensemble learning for time series prediction," in Proceedings of the International Workshop on Nonlinear Dynamic Systems and Synchronization, 2008.

[8] K. Brazier and W. Wang, "Implicit fitness sharing speciation and emergent diversity in tree classifier ensembles," in Proceedings of the 5th Conference on

Intelligent Data Engineering and Automated Learning, 2004, pp. 333–338.
[9] T. G. Dietterich, "Ensemble methods in machine learning," in Proceedings of the 1st International Workshop on Multiple Classifier Systems, LNCS 1857, 2000, pp. 1–15.
[10] L. K. Hansen and P. Salamon, "Neural network ensembles," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, n. 10, pp. 993–1001, 1990.
[11] IEEE, "IEEE Standard for Information Technology—Software Life Cycle Processes—Reuse Processes", IEEE Standard 1517—1999, 1999.
[12] K. C. Kang, S. Kim, J. Lee, K. Kim, G. J. Kim, and E. Shin, "FORM: A feature-oriented reuse method with domain-specific reference architectures," Annals of Software Engineering, vol. 5, pp. 143-168, 1998.
[13] Y. Chen, B. Yang, J. Dong, and A. Abraham, "Time-series forecasting using flexible neural tree model," Information Sciences: an International Journal, vol. 174, n.3–4, pp. 219–235, 2005.
[14] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," Neurocomputing, vol. 50, pp. 159–175, 2003.
[15] C. M. Kuan and T. Liu, "Forecasting exchange rates using feedforward and recurrent neural networks," Journal of Applied Econometrics, vol. 10, pp. 347–364, 1995.
[16] A. N. Borisov and V. A. Pavlov, "Prediction of a continuous function with the aid of neural networks," Automatic Control and Computer Sciences, vol. 29, pp. 39–50, 1995.
[17] G. T. Leavens and M. Sitamaran, "Foundations of component-based systems," Cambridge University Press, 2000.
[18] M. Fowler, "Inversion of control containers and the dependency injection pattern," available at: http://martinfowler.com/articles/injection.html, 2004.
[19] L. X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," IEEE Transactions on Systems, Man and Cybernetics, vol. 22, n. 6, pp. 1414–1427, 1992.
[20] E.A. Wan, "Time series data", Department of Computer Science and Electrical Engineering at Oregon Health & Science University, available at http://www.cse.ogi.edu/~ericwan/data.html, 2001.
[21] V. M. Landassuri-Moreno and J. A. Bullinaria, "Neural network ensembles for time series forecasting," in Proceedings of the 11th Annual conference on Genetic and evolutionary computation, 2009, pp. 1235-1242.
[22] A. Chitra and S. Uma, "An ensemble model of multiple classifiers for time series prediction," International Journal of Computer Theory and Engineering vol. 2, no. 3, pp. 454-458, 2010.
[23] K. V. G. Rao, P. P. Chand, and M. V. R. Murthy, "Soft computing-neural networks ensembles", Journal of Theoretical and Applied Information Technology, vol. 3, n. 4, pp. 45-50, 2007.

Claudio V. Ribeiro is an Electronic Engineer at the Naval Design Company. His main research interests are in data fusion algorithms and time series forecasting. He has published two papers in international conferences. He has a Master of Science degree in Systems and Computing at the Military Institute of Engineering of Rio de Janeiro, Brazil.

Ronaldo R. Goldschmidt is an associate professor at the Multidisciplinary Institute of the Federal Rural University of Rio de Janeiro, Brazil. His main research interests are in artificial intelligence and data mining. He has published over 40 papers in journals, book chapters, conferences, and workshops, and has co-edited 6 books on artificial intelligence, data mining, computer science and education. He is also a member of the Brazilian Computer Society.

Ricardo Choren is an associate professor at the Computer Engineering Department of the Military Institute of Engineering. His main research interests are in software engineering techniques for autonomous software systems, particularly multi-agent systems. He has published over 70 papers in journals, book chapters, conferences, and workshops, and has co-edited 3 LNCS volumes on software engineering for multi-agent systems. He has been PC or senior PC member for a number of events. He is a member of the ACM and of the Brazilian Computer Society.