



Munich Personal RePEc Archive

Developing a short-term comparative optimization forecasting model for operational units' strategic planning

Filippou, Miltiades and Zervopoulos, Panagiotis

School of Electrical and Computer Engineering, National Technical University of Athens, Department of Economic and Regional Development, Panteion University of Athens

20 April 2011

Online at <https://mpra.ub.uni-muenchen.de/30766/>

MPRA Paper No. 30766, posted 08 May 2011 07:05 UTC

Developing a Short-Term Comparative Optimization Forecasting Model for Operational Units' Strategic Planning

Miltiades Filippou

School of Electrical and Computer Engineering, National Technical University of Athens,
9 Iroon Polytechniou Street, 157 80, Athens, Greece

Email: miltfil@central.ntua.gr

Panagiotis Zervopoulos

Department of Economic and Regional Development, Panteion University of Athens,
136 Syngrou Avenue, 176 71, Athens, Greece

Email: panagiotis.zervopoulos@gmail.com

ABSTRACT

Data drain for peer active units operating in the same sector is a major factor that prevents policy makers from developing flawless strategic plans for their organisation. This study introduces a hybrid model that incorporates a purely deterministic method, Data Envelopment Analysis (DEA), and a semi-parametric technique, Artificial Neural Networks (ANNs), to provide a strategic planning tool for efficiency optimization applicable to short-term lag of data availability. For consecutive time instances, t and $t+1$, the developed DEANN model returns optimum "regression-type" input and output levels for every sample operational unit, even for the fully efficient ones, that may decide to alter the levels of the efficiency determinants, respecting the t -time efficiency frontier.

Keywords: Forecasting, Optimization, Efficiency, Data Envelopment Analysis (DEA), Artificial Neural Networks (ANN), Adaptive Techniques

1. INTRODUCTION

Efficiency optimization is a primary profitability strengthening driver for private companies (Banker et al., 1984). It also is a prerequisite for public organisations that have adopted the New Public Management concept (3Es: Efficiency, Effectiveness, and Economy). Although relative efficiency measurement is important in the strategic planning of an operational unit, a time lag of data availability exists (e.g., level and cost of resources engaged, level of outputs produced, and revenues obtained). For instance, companies' balance-sheet reports are released a minimum of six months after the end of the fiscal year. Due to this delay, there is a financial and production data drain. This drain delays policymakers from finalising completed economic analyses and strategic plans for their organisation that take into account the decisions of its counterparts. As a result, during the 'idle time', only ceteris paribus analyses can be conducted in which the policymakers of just one player make crucial decisions for the operational unit regarding the peer units as inactive.

We tackle the issue at stake by developing a semi-parametric tool to optimize the input-output or cost-revenue mix. Concurrently, the tool firstly estimates a

stochastic best-practice frontier, or a stochastic production function, and secondly identifies the link between input and output data of the sample units or efficiency optimization. The stochastic reference set is more tolerant to data perturbations and short-term modifications, hence, more robust for short-term forecasting than the frontier determined by non-parametric methods such as DEA.

The novelty of the developed short-term comparative optimization forecasting model relaxes the future period prediction of the optimum output levels, introducing the feasible input levels for the selected operational unit (output-oriented approach), or vice versa (input-oriented approach), anticipating the short-period production frontier variation. An important factor is that data of the sample operational units for one period are adequate for the future period input-output value determination.

The paper is organised as follows. In the first section, we analyse the DEA (Variable Returns to Scale) and the ANNs (feed forward and recurrent neural network architectures), as well as discuss a review of the literature on hybrid DEA-ANN applications. In the following section, we apply the DEANN model to real data. In the last section, we elaborate on the managerial implications of the DEANN model, the concluding remarks and potential for future research.

2. LITERATURE REVIEW

Discussion of the DEA and ANNs, the two components of the DEANN method, follow, as well as studies related to joint DEA and ANNs applications. The scope of this review is the understanding of the DEANN technique's functional underpinnings and potentials towards optimization forecasting.

2.1 Data Envelopment Analysis (DEA)

Based on the seminal paper of Charnes, Cooper and Rhodes (1978), a non-parametric relative efficiency evaluation technique has been developed. DEA relies on linear programming to identify the best-practice or efficient operational units within a sample of homogenous counterparts; consequently the optimum input-output transformation process, and also the ability to estimate the target input or output values for every inefficient unit in order to mimic its best-practice reference peers. DEA relaxes the comparative efficiency assessment, discharging any assumption for the underlying production function of the sample active units, to estimate the best-practice reference set comprised solely of the relatively best input-output transformers.

The sample under evaluation consists of homogenous units, or Decision Making Units (DMUs), that perform common operations and engage uniform inputs to produce uniform outputs (Athanasopoulos & Curram, 1996). The differences between the sample DMUs concentrate on the level of the resources used and the level of goods and services produced.

DEA models have a twofold interpretation, depending either on the orientation of the transformation process decided by the units' policy makers or on the controllability of the operational unit over the resources or the outputs. To be

more precise, the production process may be input oriented or output oriented. In the former case, aim of the comparative analysis is the estimation of the appropriate reduction on input levels (target inputs) holding the output levels fixed, and in the output oriented approach, we seek to reveal the maximum relative output values (target outputs), respecting the current input levels towards the efficiency attainment of every operational unit's production process.

CCR (Charnes et al., 1978) and BCC (Banker et al., 1984) are the two source models which differ in the returns to scale assumption underlying the production process. The former model assumes Constant Returns to Scale (CRS) and the latter Variable Returns to Scale (VRS). As a result, the BCC model leads to better fitting efficiency reference set to the sample data, comparing to the CCR best-practice frontier, while they are returns to scale sensitive. Additionally, BCC is deemed more appropriate than CCR in case various size DMUs comprise the evaluation sample (Cooper et al., 2007).

The formulas developed to apply the BCC model are presented below:

$$\begin{aligned}
 e^* &= \min e \\
 \text{subject to } & \sum_{j=1}^n \lambda_j x_{ij} \leq e x_{i0} \quad i = 1, \dots, m \\
 & \sum_{j=1}^n \lambda_j y_{rj} \geq y_{r0} \quad r = 1, \dots, s \\
 & \sum_{j=1}^n \lambda_j = 1 \\
 & \lambda_j \geq 0 \quad j = 1, \dots, n
 \end{aligned}$$

where DMU_0 stands for one of the sample DMUs under assessment, x_{i0} and y_{r0} represent the i th input and r th output of DMU_0 respectively, and λ_j are the input and output non-negative weights.

2.2 Artificial Neural Networks (ANN)

A neural network (simply called NN) is a large-scale system which contains large numbers of special, non-linear processors. These processors are called 'neurons'. Each neural network is characterised by a state, an input set the weights of which come from other neurons and an equation, which describes the dynamical function of the NN. The weight factors are renewed (their values are changed through the periods of time) with the help of a learning process which takes place along with the minimisation of a cost function (error). As a result, the weight factors are renewed gradually. The optimal values of the weight factors¹ then are stored and used during the solution of a problem which requires the presence of the NN.

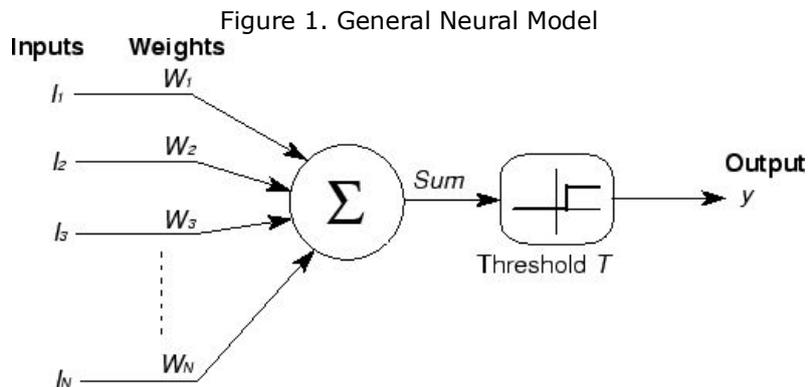
¹ Weight factor is a measure of the connection strength between two neurons.

2.2.1 General form of an artificial neuron – the perceptron

Basic definition

The general form of an artificial neuron can be described in two stages (Figure 1). In the first stage, the *linear combination* of inputs is calculated. Each value of input array is associated with its weight value, which is normally between 0 and 1. The summation function (Function 1) often takes an extra input value θ with weight value equal to unity to represent the threshold or *bias* of a neuron. The summation function will be then performed as,

$$x = \sum_{i=1}^N A_i W_i + \theta \quad (1)$$

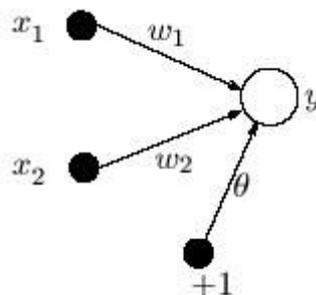


The above McCulloch-Pitts Threshold Logic Unit (TLU) mode is also called "perceptron" (Tzafestas, 2001). The perceptron was the result of merger between two concepts from the 1940s, McCulloch-Pitts model of an artificial neuron and Hebbian learning rule of adjusting weights.

The perceptron algorithm

A single layer feed-forward network consists of one or more output neurons o , each of which is connected with a weighting factor with every i input variable. In the simplest case, the network has only two inputs and a single output (Figure 2).

Figure 2. Single Perceptron Network



The input of the neuron is the weighted sum of the inputs plus the bias term θ . The output of the network is formed by the activation of the output neuron, which is the outcome of an input-dependent functional form (Function 2).

$$y = F\left(\sum_{i=1}^2 w_i x_i + \theta\right) \quad (2)$$

The activation function F can be linear or nonlinear so that we have a linear or nonlinear network respectively. At this point, we consider the threshold, alternatively, *Heaviside* or *sgn*, function.

$$F(s) = \begin{cases} 1, & \text{if } s > 0 \\ -1 & \text{otherwise} \end{cases} \quad (3)$$

The output of the network thus is either +1 or -1 depending on the input. The network now can be used for a classification task: "*it can decide whether an input pattern belongs to one of two classes (linear separability)*". If the total input is positive, the pattern will be assigned to class +1; if the total input is negative, the sample will be assigned to class -1. In this case, the separation between the two classes is expressed by a straight line, given by the following equation:

$$w_1 x_1 + w_2 x_2 + \theta = 0 \quad (4)$$

The above equation is the dot product $W \cdot x$ to which a bias θ is added.

It should be noted that the learning methods are weight-adjustment iterative procedures.

2.2.2 Multilayer perceptrons (MLP) – backpropagation algorithm (BP)

The MLPs are perhaps the most popular and widely applied models of the many existing ANN types. Hornik et al. (1990) have shown that, subject to mild regularity conditions, these models can approximate any function and its derivatives to any degree of accuracy.

The MLPs basic properties are summarised in the triplet: multi-layer, feed-forward and supervised neural network. Their processing elements, known as '*neurons*', are organised in at least three layers: the input layer, the output layer and the hidden layer(s) in between. These *neurons* all are fully connected between adjacent layers. MLPs are feed-forward networks (e.g., all connection points in one direction, from the input towards the output layer). Finally, they are supervised networks since all patterns of inputs and outputs must be provided. The development of a neural network model requires the specification of a '*network topology*' and a '*training strategy*'.

To add layers, we need to do an additional step than just connect up some new weights. To be more precise, we need to introduce a *non-linearity* ($g(a)$) assumption. In general, the non-linearity we will use works to make the outputs from each layer crisper. This is accomplished by using a *sigmoidal activation function*.

There are two basic commonly used *sigmoidal activation functions*:

- The Logistic Sigmoid (*logsig*) that is the integral of the statistical Gaussian distribution.

$$g(a) = \frac{1}{1 + e^{-a}} \quad (5)$$

- The Tangential Sigmoid (*tansig*) that is derived from the *hyperbolic tangent*. It has the advantage over the *logsig* of being able to deal directly with negative numbers.

$$g(a) = \tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}} \quad (6)$$

The backpropagation (BP) algorithm is a method for computing partial derivatives in a network. In short, it is nothing more (nor less) than the chain rule from calculus. One of its virtues is that it is an extremely efficient computation due to its recursive formulation. Note that while it is used most commonly to compute first derivatives, it may also be used to compute second derivatives (or derivatives of any order), albeit at additional computational expense.

For instance, let's assume that we are training a network with three layers: an input layer which is connected to a hidden layer which, in turn, is connected to an output layer. The first thing to do is to select a log likelihood function that is appropriate for the nature of the task. To be more precise, let's assume that the task is regression-type in which the target output $y^*(t)$ is a noise-corrupted version of some function of the input $x(t)$:

$$y^*(t) = f(x(t)) + \varepsilon \quad (7)$$

where $\varepsilon \sim N(0,1)$ that is:

$$y^*(t) \square N(f(x(t)), 1)$$

The log likelihood function is:

$$\begin{aligned} \log L &= \sum_t -\frac{1}{2} \|y^*(t) - y(t)\|^2 \\ &= \sum_i \sum_t -\frac{1}{2} [y_i^*(t) - y_i(t)]^2 \end{aligned} \quad (8)$$

where the index t ranges over all training patterns, and the index i ranges through all output units.

We want to compute the derivatives of the log likelihood with respect to the network's weights.

Let's assume that we are doing "batch" training, meaning that the weights remain constant within an epoch (a pass through all data patterns) but change between epochs. For notational convenience, two time indexes will be used: n expresses epochs and t steps (pattern presentations within an epoch). Using the chain rule, computation of the desired derivatives may be broken up into three stages. It is easier if we consider output units and hidden units separately. For an output unit, the stages are represented by the three partial derivatives on the right-hand side of the equation:

$$\frac{\partial \log L(n)}{\partial w_{ij}(n)} = \sum_t \frac{\partial \log L(t)}{\partial y_i(t)} \frac{\partial y_i(t)}{\partial s_i(t)} \frac{\partial s_i(t)}{\partial w_{ij}(n)} \quad (9)$$

where $s_i(t)$ is the weighted sum of the unit i 's inputs at step t , and $w_{ij}(n)$ stands for the weight on the connection between the hidden unit j and output unit i at epoch n . For a hidden unit, the stages are:

$$\frac{\partial \log L(n)}{\partial w_{ij}(n)} = \sum_t \frac{\partial \log L(t)}{\partial h_i(t)} \frac{\partial h_i(t)}{\partial s_i(t)} \frac{\partial s_i(t)}{\partial w_{ij}(n)} \quad (10)$$

2.2.3 Recurrent Neural Networks – Elman Networks

Typical NN models are closely matched to statistical models and have gained promising results in a large range of applications. For noisy time series prediction, neural networks typically take a delay embedding of previous inputs which is mapped into a prediction.

In the *Elman network*, positive feedback is used to construct memory in the network. The network is consisted of input, hidden and output layers. Special units called *context units* save previous output values of hidden layer neurons. Context unit values then are fed back fully connected to hidden layer neurons and thus serve as additional inputs to the network. Networks output layer values are not fed back to the network. The *Elman network* has a high depth and low resolution memory, since the context units keep exponentially decreasing trace of past hidden neuron output values.

2.3 DEA and ANN models

Both DEA and ANNs are non-parametric techniques that make no assumptions about the production function that links inputs with outputs. However, unlike DEA, which is deemed an extreme method as it estimates a production function based on the relatively best performing sample operational units, ANNs uncover an adaptive functional form with stochastic underpinnings (Wang, 2003). Thus, the ANNs' outcomes are less sensitive than the DEA results to data perturbations due to the noise inherited by their regression-type architecture.

Acknowledging DEA and ANN methods' properties, there are several studies that introduce joint application. The scope of such a combination is the efficiency assessment or prediction (Yaghoobi et al., 2010; Wu et al., 2006; Wu et al., 2004; Pendhakar & Rodger, 2003; Wang, 2003), the efficiency assessment and the comparison of the accuracy of the outcomes obtained by the two methods (Athanasopoulos & Curram, 1996). In these studies, DEA is used in first-stage analysis to preprocess actual input and output data for the following stage in which ANNs are applied. DEA in conjunction with ANNs lead to a semi-parametric method development.

The limitations of the existing DEA-ANN methods are concentrated on the selection of the "efficient" sample DMUs, discarding the remaining "inefficient" units from the training process of the ANN model (second-stage analysis). In other words, these approaches ignore the magnitude of a significant portion of

the sample on the efficiency assessment. The current DEA-ANN methods' classification between efficient and inefficient units does not adopt the traditional DEA concept. For instance, for the ANNs' training phase, efficient units are deemed those imputed efficiency scores less than unity in order to identify an adequate number of DMUs that meets the minimum ANNs' processing requirements. According to Trout et al. (2003), the number of units used for ANNs' training purposes should be greater than ten times the number of input variables.

Additionally, the DEA-ANN papers omit the input and output levels' estimation in future periods, which is a major concern of policy makers. They solely analyse the efficiency parameter.

3. DEANN METHODOLOGY

The DEANN model for short-term optimization forecasting is a hybrid DEA and ANN stochastic technique. The developed model applied DEA in a first-stage comparative analysis for "filtering" the input and output data. To be more precise, the first-stage analysis leads to sample units' efficiency scores and target inputs (input orientation) or target outputs (output orientation) as a roadmap for fully efficiency attainment (efficiency score = 1.000). The second-stage analysis, where the appropriate ANN model (e.g., feed forward, recurrent) and topology (e.g., number of hidden layers and number of nodes per hidden layer) are applied, identifies the stochastic functional form that links the optimum input-output mix. The functional form expresses the "regression-type" efficiency frontier of the sample DMUs that is adaptive to short-term data variations, unlike the pure DEA reference set that is deterministic. Based on the DEANN efficiency frontier, we predict the optimum input-output mix for efficiency attainment for every sample DMU, anticipating the input-output games the peer units may play during the "idle-time".

The phases of the DEANN model implementation are epitomized into the subsequent algorithm:

- Step 1:* Run DEA (BCC) in order to identify sample DMUs efficiency scores and target input or output values (filtered values), depending on the orientation preferred.
- Step 2:* Apply the filtered values to the appropriate ANN model and topology to reveal the stochastic efficiency reference set (functional form).
- Step 3:* Impute new input or output values to the DEANN functional form for selected sample DMUs to predict the optimum solutions for efficiency attainment.

The orientation of the DEANN model depends on the disposability of resources, the market dynamic and structure, and the units' controllability over input or output variables. In case the input-oriented approach is selected, the DEANN model predicts the optimum input values for given outputs (different from the actual output levels) selected by the policymaker. The output values are estimated when the output-oriented approach is applied. The selected values are feasible variable levels for the unit(s) regarding the period t to $t+1$.

By applying filtered values for the stochastic functional form estimation instead of actual values, as recommended by previous studies (e.g., Yaghoobi et al., 2010; Hu et al., 2008; Wu et al., 2006; Wu et al., 2004; Pendhakar & Rodger, 2003; Wang, 2003; Costa & Markellos, 1997; Athanassopoulos & Curram, 1996), we improve the DEA and ANN synergy for identifying the efficiency reference set. The DEA target dataset that represents the optimum input-output mix for every sample DMU to full efficiency attainment increases the flexibility and applicability of the DEANN outcomes to real conditions. To be more precise, we do not exclude the inefficient or the arbitrarily deemed “inefficient”, based on the researchers’ criteria, units from the training process, expanding the applicability of the DEANN model to a smaller sample size and enhancing the accuracy of the estimated stochastic efficiency frontier.

A peculiarity of the DEANN model is the sample size used for the training phase of the second step of the algorithm; namely, the number of DMUs should be greater than ten times the number of input variables (Troutt et al., 1995).

Additionally, by using efficient and potentially efficient DMUs (using DEA target values) for the stochastic best-practice frontier estimation, we respect a major economic assumption: monotonicity (Pendhakar & Rodger, 2003).

4. NUMERICAL EXAMPLE

4.1 Data description

The DEANN model is applied to data from the Citizen Service Centers (CSCs), decentralised governmental one-stop-shops. One hundred SUs comprise the sample, out of 1020 operating in Greece, serving about 70% of the citizens who apply to CSCs for administrative issues. There are five input variables in the dataset (number of full-time employees, weekly working hours, number of PCs, number of fax machines and number of printers), and three output variables (number of electronic protocol registered services provided, number of manual services provided and number of served citizens).

4.2 DEANN application

By implementing Step 1 of the DEANN algorithm, applying the BCC DEA model, the efficiency scores of the sample hundred DMUs and the target input or output values (filtered values), depending on the orientation selected, are assigned. In this study, we load the DEANN model for both-sided orientations.

The DEA application reveals that 51% and 31% of the sample operational units are fully efficient in case of input and output orientation, respectively (Table 1).

Table 1. DEA Statistics

| Units | Orientation | |
|-------------|-------------|--------|
| | Input | Output |
| Efficient | 51 | 31 |
| Inefficient | 49 | 69 |
| Total | 100 | 100 |

In order to exhaust the available data and limit the sample size requirements for applying ANNs, we select the DEA target input and output levels, that potentially lead the DMUs to the non-parametric efficiency frontier for the second stage analysis. By experimenting with ANN models and architectures on the filtered dataset, we identify the most statistically significant functional form that minimises the mean square error. Based on the input-output distribution, the training function applied to the second phase of the input-oriented DEANN analysis is the *Levenberg-Marquardt* (Figure 3), and of the output-oriented, the *Conjugate Gradient with Polak-Ribiere Restarts* (Figure 4). The former is a feed forward and the latter a recurrent (Elman) ANN model. The topology of the input and output-oriented DEANN data processing consists of one hidden layer with six and fifteen nodes respectively. For training of the ANN, for both orientations, seventy DMUs were selected. The remaining thirty operational units were used to cross-validate the adaptability of the network.

Figure 3. Feed-forward ANN Architecture Applied to the Input-Oriented Analysis

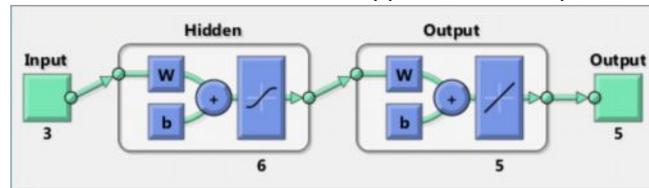


Figure 4. Recurrent (Elman) ANN Architecture Applied to the Output-Oriented Analysis

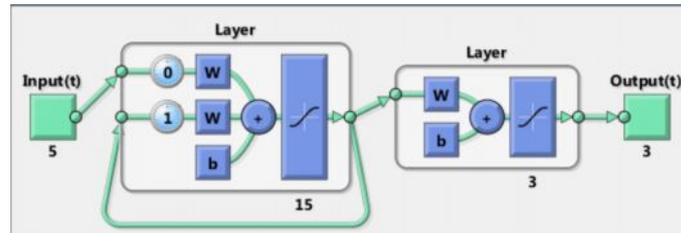


Table 2. ANN Properties

| Properties | Orientation | |
|-------------------|---------------------|--|
| | Input | Output |
| Data Selection | Random | |
| Inputs | 3 | 5 |
| Hidden Layer(s) | 1 | 1 |
| Neurons | 6 | 15 |
| Outputs | 5 | 3 |
| Training Function | Levenberg-Marquardt | Conjugate Gradient with Polak-Ribiere Restarts |
| Mean Square Error | $<10^1$ | $10^1 < \text{MSE} < 10^2$ |
| R | 0.9818 | 0.9927 |

Subsequent to revealing the functional form, a distinct function for each orientation, that expresses the stochastic best-practice frontier, the DEANN optimum input and output levels are estimated. For instance, the deviation between the DEA filtered input and their ANN counterparts, in case of input-oriented analysis, depicted in Table 3, expresses the correction based on the

stochastic reference set. To be more precise, sixty eight out of the one hundred fifty ANN-obtained input values were not altered more than 10% from the equivalent DEA filtered levels. Additionally, 68% of the ANN optimum input values, holding the outputs fixed, were moved upward compared to the DEA target values (Table 3).

Table 3. DEA Filtered and ANN Input Values Deviation (Input-Oriented Analysis)

| DMUs | Deviation | | | | | DMUs | Deviation | | | | |
|------|---------------------|---------------|--------|--------|----------|------|---------------------|---------------|--------|--------|----------|
| | Full-Time Employees | Working Hours | PC | Fax | Printers | | Full-Time Employees | Working Hours | PC | Fax | Printers |
| 71 | 0.117 | 0.101 | 0.075 | -0.216 | 0.246 | 86 | -0.147 | -0.106 | 0.462 | -0.543 | -0.553 |
| 72 | -0.042 | 0.081 | 0.441 | -0.392 | 1.273 | 87 | 0.150 | 0.070 | 0.090 | -0.012 | 0.625 |
| 73 | 0.150 | 0.031 | 0.268 | -0.108 | 0.607 | 88 | 0.000 | 0.201 | 0.456 | -0.105 | 0.400 |
| 74 | -0.038 | 0.391 | 0.094 | 0.007 | 0.113 | 89 | 0.037 | -0.042 | -0.043 | 0.001 | -0.057 |
| 75 | -0.016 | 0.144 | 0.126 | -0.125 | 0.881 | 90 | 0.099 | -0.058 | 0.103 | 0.001 | 0.825 |
| 76 | 0.135 | 0.049 | 0.040 | 0.001 | 0.354 | 91 | 0.109 | -0.002 | 0.298 | -0.226 | -0.041 |
| 77 | -0.079 | -0.085 | 0.523 | 0.001 | -0.054 | 92 | 0.070 | 0.151 | -0.029 | -0.286 | 0.476 |
| 78 | 0.033 | -0.191 | 0.265 | 0.001 | 0.420 | 93 | -0.201 | 0.059 | -0.087 | 0.024 | 0.959 |
| 79 | 0.117 | -0.034 | 0.207 | 0.001 | 0.882 | 94 | 0.048 | 0.043 | 0.280 | -0.386 | 0.473 |
| 80 | 0.166 | 0.048 | -0.189 | -0.264 | 0.602 | 95 | 0.085 | 0.130 | -0.022 | 0.024 | 0.127 |
| 81 | 0.073 | -0.128 | 0.645 | -0.303 | 0.365 | 96 | 0.052 | 0.260 | -0.043 | 0.001 | 0.399 |
| 82 | 0.233 | -0.096 | 0.105 | -0.008 | 0.896 | 97 | -0.100 | 0.351 | -0.231 | 0.743 | 0.038 |
| 83 | 0.072 | 0.056 | -0.047 | 0.001 | 0.003 | 98 | 0.038 | 0.066 | 0.900 | -0.399 | 0.406 |
| 84 | -0.043 | 0.288 | 0.067 | 0.138 | 0.046 | 99 | -0.049 | -0.069 | 0.365 | -0.279 | 0.152 |
| 85 | 0.099 | 0.036 | 0.181 | -0.009 | 0.142 | 100 | 0.038 | 0.199 | 0.901 | -0.399 | 0.407 |

The respective deviations yielded from the output-oriented analysis are included in Table 4 (Appendix).

By imputing input or output values, depending on the output or input approach adopted, respectively, the optimum DEANN output or input levels are projected for the period starting at point t and ending at $t+1$. The inputs or outputs imputed are the outcome of a relative stochastic efficiency assessment.

5. CONCLUDING REMARKS AND FURTHER RESEARCH

The scope of this paper is the development of a stochastic optimization tool for short-term forecasting without requesting assumption of the production function underlying the input-output transformation process. The developed DEANN model incorporates the virtues of a pure deterministic technique (DEA) and a noise-embedded adaptive method (ANN) in order to estimate the best fitting frontier for the efficient and potentially efficient sample DMUs. Due to the parametric properties of the frontier, it is tolerant to short-term data perturbations. Based on the stochastic efficiency frontier, the DEANN model provides an input-output optimization roadmap to policy makers for feasible scenarios towards efficiency attainment, and consequently profit increase.

By using filtered input and output values, we respect the monotonicity assumption dominating the economic theory. The selected ANN for the stochastic

optimum input-output functional form estimation is not predetermined but customised to the properties of the dataset. To be more precise, the ANN model selection, and the particular architecture and topology of the network are the outcomes of a resampling computational statistical process in which the minimisation of the Mean Square Error is pursued.

Further research is needed to obtain a generalised stochastic efficiency frontier that embraces simulated population data, as well as to introduce a stochastic long-term optimization forecasting model without need to arbitrarily select particular production functional form.

REFERENCES

Book

Cooper, W.W. et al, 2007. *Data Envelopment Analysis: A Comprehensive Text with Models, Applications, References and DEA-Solver Software* (2nd ed.). Springer, New York, USA.

Tzafestas, S., 2001. *Computational Intelligence in Systems and Control Design and Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Journal

Athanassopoulos, A. and Curram, S., 1996. A Comparison of Data Envelopment Analysis and Artificial Neural Networks as Tools for Assessing the Efficiency of Decision Making Units. *In The Journal of the Operational Research Society*, Vol. 47, No. 8, pp. 1000-1016.

Banker, R.D. et al, 1984. Some Models for Estimating Technical and Scale Inefficiencies in Data Envelopment Analysis. *In Management Science*, Vol. 30, No. 9, pp. 1078-1092.

Charnes, A. et al, 1978. Measuring Efficiency of Decision Making Units. *In European Journal of Operational Research*, Vol. 2, pp. 429-444.

Costa, A. and Markellos, R.N., 1997. Evaluating Public Transport Efficiency with Neural Network Models. *In Transportation Research Part C: Emerging Technologies*, Vol. 5, No. 5, pp. 301-312.

Hornik, K. et al, 1990. Universal Approximation of an Unknown Mapping and its Derivatives Using Multilayer Feedforward Networks. *In Neural Networks*, Vol. 3, No. 5, pp. 551-560.

Pendhakar, P. and Rodger, J., 2003. Technical Efficiency-Based Selection of Learning Cases to Improve Forecasting Accuracy of Neural Networks under Monotonicity Assumption. *In Decision Support Systems*, Vol. 36, pp. 117-136.

Troutt, M.D. et al, 1995. The Potential Use of DEA for Credit Applicant Acceptance Systems. *In Computers and Operations Research*, Vol. 4, pp. 405-408.

Wang, S., 2003. Adaptive Non-Parametric Efficiency Frontier Analysis: A Neural-Network-Based Model. *In Computers & Operations Research*, Vol. 30, pp. 279-295.

Wu, D. et al, 2006. Using DEA-Neural Network Approach to Evaluate Branch Efficiency of a Large Canadian Bank. *In Expert Systems with Applications*, Vol. 31, pp. 108-115.

Conference paper or contributed volume

Hu, S.C. et al, 2008. Using Hopfield Neural Networks to Solve DEA Problems. *Proceedings of the Cybernetics and Intelligence Systems Conference*, IEEE International.

Wu, C. et al., 2004. Decision-Making Modeling Method Based on Artificial Neural Network and Data Envelopment Analysis. *In Geoscience and Remote Sensing Symposium*, IEEE International.

Yaghoobi, R. et al, 2010. Application of Multi-Layer Recurrent Neural Network and Fuzzy Time Series in Input/Output Prediction of DEA Models: Real Case Study of a Commercial Bank. *Proceedings of the 40th International Conference*, IEEE International.

APPENDIX

Table 4. DEA Filtered and ANN Output Values Deviation (Output-Oriented Analysis)

| DMUs | Deviation | | | DMUs | Deviation | | |
|------|--------------------|-----------------|-----------------|------|--------------------|-----------------|-----------------|
| | eProtocol Services | Manual Services | Served Citizens | | eProtocol Services | Manual Services | Served Citizens |
| 71 | -0.026 | -0.250 | -0.095 | 86 | -0.225 | 0.524 | -0.299 |
| 72 | 0.031 | 0.493 | 0.632 | 87 | -0.077 | -0.075 | -0.083 |
| 73 | -0.101 | -0.184 | -0.215 | 88 | -0.193 | -0.169 | -0.167 |
| 74 | -0.080 | -0.125 | 0.107 | 89 | 0.032 | 0.151 | 0.102 |
| 75 | -0.032 | 0.069 | -0.890 | 90 | -0.013 | 0.035 | -0.052 |
| 76 | -0.131 | -0.042 | -0.126 | 91 | 0.208 | 0.163 | 0.115 |
| 77 | -0.566 | -0.246 | -0.624 | 92 | 0.223 | 0.886 | 0.750 |
| 78 | 1.513 | 1.045 | 1.427 | 93 | -0.036 | 0.301 | 0.215 |
| 79 | -0.168 | -0.045 | -0.101 | 94 | 0.728 | 1.092 | 1.485 |
| 80 | 0.101 | 0.127 | 0.199 | 95 | 0.545 | 1.235 | 0.970 |
| 81 | 0.227 | -0.271 | -0.195 | 96 | 0.386 | 0.768 | 0.492 |
| 82 | -0.148 | -0.068 | -0.067 | 97 | 0.058 | 0.747 | 0.180 |
| 83 | -0.062 | 0.081 | 0.008 | 98 | 0.780 | 0.544 | 0.795 |
| 84 | -0.075 | 0.309 | 0.186 | 99 | 0.094 | -0.243 | -0.081 |
| 85 | -0.006 | -0.201 | -1.000 | 100 | 0.445 | 0.536 | 0.564 |