Forecasting electricity consumption in South Africa:

ARMA, Neural networks and Neuro-fuzzy systems

Lufuno Marwala
School of electrical and electronic engineering
University of Johannesburg
Johannesburg, South Africa
marwalal@gmail.com

Bhekisipho Twala
School of electrical and electronic engineering
University of Johannesburg
Johannesburg, South Africa
Btwala@uj.ac.za

Abstract*—This paper presents an experiment that consists of constructingauto-regressive moving average (ARMA), neural networks and neuro-fuzzy models with historical electricity consumptiontime series data to create models that can be used to forecastconsumption inthe future. The data was sampled on a monthly basis from January 1985 to December 2011.An ARMA,multilayer perceptron neural network with back propagation and neuro-fuzzy modelling technique which combines Takagi-Sugeno fuzzy models and neural networks were used to create the models for one step ahead forecasting. The results of the three techniques were compared and the results show that neuro-fuzzy models outperformed the neural network and ARMA models in terms of accuracy.

Keywords: forecasting; neuro-fuzzy; electricitity consumption; auto-regressive moving average, neural networks, Takagi-Sugeno

I. INTRODUCTION

In 2007, South Africa experienced a shortage in power supply.Inglesi and Pouris argue that part of the crisis was exacerbated by the inadequacy of the demand forecasting models used by Eskom, South Africa electricity utility company responsible for 95% of the power supply that led to poor planning and therefore, power shortage [1]. It has long been established that electricity demand forecasting is important for electricity utility planning.Electricity demand forecasting is divided into short-term forecasting which covers hourly to weekly forecasting, medium-term forecasting which covers from monthly to quarterly forecasting and lastly, long-term which covers years [2].

Medium to long-term demand forecasting are useful in determining the capacity of generation required in the future to meet the forecasted demand and also to plan for transmission or distribution system additions and the type of facilities that are required in transmission planning and maintenance planning. Short-term forecasts are used for control and scheduling of the power system and also as inputs to the load flow study or contingency analysis [3]. Since the beginning of the electricity supply crisis in 2007 in South Africa, the system has been operating at a tight reserve margin. It is, therefore important to conduct medium term forecasts so that maintenance of the generation plants can be planned properly.

A modelling technique that has the ability to comprehend nonlinearity and seasonality is required to be applied to systems that exhibit these dynamics.This study looked at the forecasting of monthly electricity demand using neural networks and neuro-fuzzy systems. South Africa's electricity consumption pattern and forecasting has been understudied and there are a few studies [1][4][5][6]. This could be because South Africa is a developing economy. In addition, the use of artificial intelligence (AI) to study the electricity load in South Africa is almost non-existent.This paper contributes by adding to the diversity of forecasting tools and exploration of the accuracy of the tools by comparison. In addition, the work adds to the electricity consumption studies in a developing economy [7] such as South Africa.

This paper focuses on univariate forecasting. It starts by outlining AI tools used in the study, neural networks and neuro-fuzzy systems in section 2. Section 3 outlines forecasting and has aliterature review of univariate load forecasting studies followed by the explanation of how the experiment was conducted, the resultsand the discussion of the results in section 4. Section 5 presents the conclusion.

II. ARMA, NEURAL NETWORKS AND NEURO-FUZZY SYSTEMS

A. Autoregressive Moving Average (ARMA)

Autoregressive Moving Average (ARMA) model is used to model time series with the purpose of using the model for forecasting. ARMA model is constructed such that the current value of a time series is estimated using prior values of the same time series. The model is a linear combination of the prior values and the coefficients of the linear combination are the parameterswhich are computed during the modelling process. The determination of the parameters is the training step of the modelling process and the estimation is the prediction step.

The autoregressive (AR) model includes lagged terms on the time series itself, and that the moving average (MA) model includes lagged terms on the noise or residuals [8]. By including both types of lagged terms, wearrive at what are called autoregressive-moving-average, or ARMA, models. The order of the ARMAmodel is included in parentheses as ARMA(p,q), where p is the autoregressive order and q

themoving-average order. The ARMA model can be represented as follows:

$$X_t = \theta + \sum_{i=1}^{p} \rho_i X_{t-i} + \sum_{i=1}^{q} \delta_i \varepsilon_{t-i} \quad (1)$$

Where X_t is the estimated value, θ is a constant, ρ_i and δ_i are ARMA model parameters for AR and MA respectively, ε_{t-i} is white noise sequence.

If ε_t is a random variable with mean zero and variance σ^2 then for every $t, \tau \geq 0$, with $t \neq \tau$, ε_t and ε_τ are uncorrelated. This can be represented formally as:

$$E(\varepsilon_t) = 0, E(\varepsilon_t^2) = \sigma^2, E(\varepsilon_t \varepsilon_\tau) = 0 \quad (2)$$

Least squares minimization is used to estimate the parameters of the ARMA models. Residuals of the model have to be random, andthe estimated parameters have to be statistically significant. Usually the fitting process isguided by the principle of parsimony, by which the best model is the simplest possible model, themodel with the fewest parameters that adequately describe the data.

B. *Neural networks*

Neural networks are a network of nonlinear mathematical processing units designed to work like a human brain [9]. A neural network has the ability to learn patterns from a data sample. The learning process uses available data to determine the neural network parameters similar to curve fitting in linear modelling methods. Simply put, neural networks are a data driven mathematical modelling method. Neural network differs from linear curve fitting in that it is a nonparametric learning method. This means that the number of parameters in the model is not determined a priori but it is determined during the learning process.

The most popular type of neural network used for system modelling is the multilayer perceptron (MLP) with back propagation. An MLP has layers of units namely, Input layer, Hidden layer and output layer. A neural network can only have one input layer, one output layer and multiple hidden layers. The number of hidden layers is determined by the level of complexity suitable for the system under investigation to be properly modelled. The number of neurons in the input layer is determined by the number of input features (dimensionality of the input space) used to create and test a neural network model. The number of neurons in the hidden layer or layers is determined during the training process and thus far, a scientific or statistical method has not been developed for choosing the number neurons in this layer. The number of neurons in the output layer is determined by the number of outputs of the model. The graphical structure of the MLP with three layers is illustrated in figure 1.

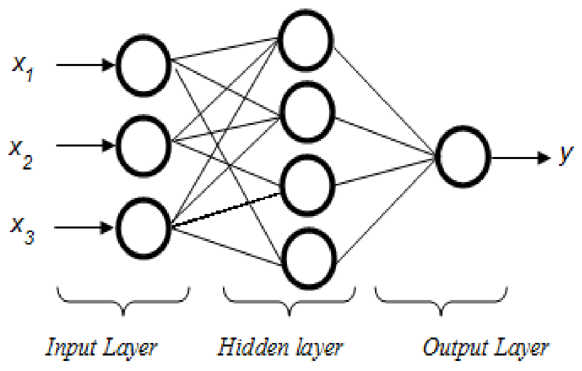

Figure 1: MLP architecture

The mapping of the inputs to the outputs using an MLP neural network can be expressed as follows:

$$f(x_t, w, \omega) = g_2\left(\sum_{j=0}^{N} w_j g_1\left(\sum_{i=0}^{k} \omega_{ji} x_i\right)\right) \quad (3)$$

where$g_1(.)$ and $g_2(.)$ are activation functions which can be sigmodal, linear or tangential.$w$andω are weight parameters.

The activation function can also be called the transfer function of a neural networks system. This function mathematically defines the relationship between the inputs and the output of a node and a network. The activation function introduces non-linearity that is important to the neural networks applications. It is the non-linearity or the ability to model a non-linear function that makes MLP so powerful. A study was conducted by Chen and Chen [10] to identify general conditions for a continuous function to qualify as an activation function. In practice they found that only a small number on bounded, monotonically increasing and differentiable activation functions are used. These include the sigmoidal function, the hyperbolic tangent function, the sine or cosine function and the linear function.MLP with a sigmoid transfer function in the hidden layer and linear transfer functions in the output layer can approximate any function provided a sufficient number of hidden units are available [11].The sigmoid activation functionandfunction is defined as:

$$g(x) = \frac{1}{1+e^x} \quad (4)$$

where$x$ is the input to the neuron.

1) Neural Network Training: Given a training set comprising a set of input x_n, where n = 1, ...N, together with a corresponding set of target vectors t_n, the objective is to minimise the error function $\epsilon(\omega)$.

$$\epsilon(\omega) = \frac{1}{2} \sum_{n}^{N} ||(y(x_n, \omega) - t_n)||^2 \quad (5)$$

whereϵ is the total error all patterns, the index *n* ranges over the set of input patterns. The variablet_n is the desired output for the nth output neuron when the *nth* pattern is presented, and $y(x_n, \omega)$ is the actual output of the *nth* output neuron when pattern h is presented.

This type of learning is called supervised learning, where every input has an associated target output. After the computation of the error the weight vector is then updated as follows:

$$\omega_k^{t+1} = \omega_k^t - \gamma \nabla \epsilon^t(\omega_k^t) \quad (6)$$

where$t$ indexes the iteration steps, k the weights, γ is the learning rate and $\nabla \epsilon(\omega)$ is the gradient:

$$\nabla \epsilon(\omega) = [\frac{\partial \epsilon}{\partial \omega_0}, \frac{\partial \epsilon}{\partial \omega_1}, \dots \dots, \frac{\partial \epsilon}{\partial \omega_k}] \quad (7)$$

The goal is to find a vector of weights such that ϵ takes its smallest value. A minimum that corresponds to the smallest value of the error function for any weight vector is said to be a *globalminimum*. Any other minima corresponding to higher values of the error function are said to be *local minima* [11].

The learning algorithm and number of iterations determines how good the error on the training data set is minimized meanwhile the number of learning samples determines how good the training samples represent the actual function. The perceptron learning rule is a method for finding the weights in a network. The perceptron learning rule is a method for finding the weights in a network. The perceptron has the property that if there exist a set of weights that solve the problem, then the perceptron will find these weights. This rule follows a linear regression approach, that is, given a set of inputs and output values, the network finds the best mapping from inputs to outputs. Given an input value which was not in the set, the trained network can predict the most likely output value. This ability to determine the output for an input the network was not trained with is known as generalization. These hidden units make use of non-linear activation functions.

C. *Neuro-fuzzy systems*

The concepts of fuzzy models and neural network models can be combined in various ways. This section covers the theory of fuzzy models and shows how they combine with neural network concepts to give what is called the neuro-fuzzy model. The most popular neuro-fuzzy model is the Takagi-Sugeno (TS) model which is widely used in data driven modelling [12].

Fuzzy logic concepts provide a method of modelling imprecise models of reasoning, such as common sense reasoning, for uncertain and complex processes [10]. Fuzzy set theory resembles human reasoning in its use of approximate information and uncertainty to generate decisions. In fuzzy systems, the evaluation of the output is performed by a computing framework called the fuzzy inference system (FIS). There are two popular fuzzy models: the Mamdani model and the TS model. The TS model is more popular when it comes to data-driven identification and has been proven to be a universal approximator [12].

1) *Takagi-Sugeno Model*

The TS model is used in data driven system identification. This model defines the antecedent in the same manner as the Mamdani model while the consequent is an affine linear function of the input variables:

$$R_i: if\ x\ is\ A_i\ then\ y_i = a_i^T x + b_i \quad (8)$$

whereA_i is the antecedent, $\boldsymbol{a_i}$ is the consequent parameter vector, b_i is a scalar bias value and $i = 1, \dots \dots k.$. What differentiates the TS model from the Mamdani model is that the former combines the linguistic description with standard functional regression while the latter only uses the linguistic description. The antecedents describe the fuzzy regions in the input space in which the consequent functions are valid. The y output is computed by taking the weighted average of the individual rules' contributions

$$y = \frac{\sum_{i=1}^{k} \beta_i(x) y_i}{\sum_{i=1}^{k} \beta_i(x)} = \frac{\sum_{i=1}^{k} \beta_i(x)(a_i^T x + b_i)}{\sum_{i=1}^{k} \beta_i(x)} (9)$$

whereβ_i is the degree of fulfilment of the ith rule. The antecedent's fuzzy sets are usually defined to describe distinct, partly overlapping regions in the input space. In the TS model the parameters are local linear models of the considered nonlinear system. The TS model can thus be considered as a smooth piece-wise linear approximation of a nonlinear function.

2) *Fuzzy to Neurofuzzy*

The application neural networks technique to FIS makes the FIS parameters adaptive. In orderto optimize parameters in a fuzzy system, gradient-descenttraining algorithms used for training neural networks models can be employed. Hence, this approach is usually referredto as neuro-fuzzy modeling [14]. Under certain minor constraints the neuro-fuzzy architecture is also equivalent to a radial basis function network [13].

Fig. 2illustrates an architectural representation of the network with two rules. The nodes in the first layer compute the membership degree of the inputs in the antecedent Gaussian fuzzy sets. The product nodes $\prod$in the second layer represent the antecedent connective. In the second layer the fuzzy logic operator *and* is applied. The normalization node N and the summation node $\sum$realize the fuzzy-mean operator (6). This system is called adaptive neuro-fuzzy inference system(ANFIS).

The Gaussian membership function of the form shown in (8) is most commonly used [13].

$$\mu A_{ij}(x_j, c_{ij}, \sigma_{ij}) = \exp\left(-\frac{(x_i - c_j^i)^2}{2\sigma_{ij}^2}\right) (10)$$

where$c$ is the centre of the Gaussian function and σ describes the variance of the Gaussian membership function.

The input-output relationship for TS model is defined as:

$$y = \sum_{i=1}^{K} \gamma_i(x)(a_i^T x + b_i) (11)$$

with

$$\gamma_i(x) = \frac{\prod_{j=1}^{p} exp(-(x_i - c_j^i)^2/2\sigma_{ij}^2)}{\sum_{i=1}^{K}\prod_{j=1}^{p} exp(-(x_i - c_j^i)^2/2\sigma_{ij}^2)} \quad (12)$$

In a neuro-fuzzy system, there are two types of model tuning which are required, namely structural and parametric tuning. Structural tuning is a procedure that finds the suitable number of rules and the proper partitioning of the input space. Upon finding the suitable structure, the parametric tuning searches for the optimal membership functions together with the optimal parameters of the consequent models. The problem can be formulated as that of finding the structure complexity which will give the best performance in generalisation.Too many rules may lead to an overly complex model with redundant fuzzy rules which compromises the integrity of the model [14].

The output of the entire inference system is computed by taking a weighted average of the individual rules' contributions as shown in (9).The parameters obtained from the training are then used to approximate models of the non-linear system under consideration [15].

III. Forecasting and forecasting studies

A. Forecasting

If*Y*represents the past values of consumption and*X* is future consumption value to be predicted, then forecasting framework is defined as:

$$X_t = f(Y_{t-1}, \ldots Y_{t-n}) \quad (13)$$

where$X_t$is the electricity consumption in month *t* andisdependent variableand *f(.)* is a function with lagged monthly electricity consumption values as the independent variables of the model.

B. Forecasting studies

Univariate forecasting is mostly used for short-term forecasting. Multiple linear regression methods have been used for modelling systems with a single variable. Box-Jenkins methods have been used frequently in univariate load forecasting. ARIMA method has also been used widely for univariate electricity demand forecasting [16][17]. Christiananseused exponential smoothing as a forecasting method [18]. In the past decade, neural networks have emerged as the preferred tool for modelling non-linear systems such as electricity demand.

Darbellay and Slamaused a neural network, created by using historical hourly load, and the results matched that of an ARIMA model for prediction from an hour ahead to 36 hours ahead [19].James W. Taylor et al,compared various forecasting methods in a short-term univariate study of electricity demand forecasting [17].

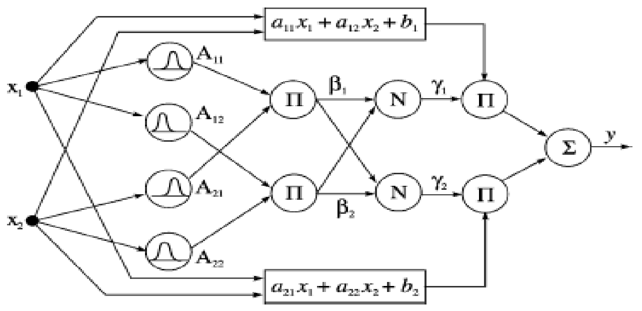

Figure 2: ANFIS architecture

The paper compared the performance of Principal component analysis (PCA), double seasonal ARMA modelling, Exponential smoothing, neural networks and two simplistic benchmark methods.

Load forecasting has been modelled with hybrid models based on artificial intelligence such as wavelet fuzzy neural network and fuzzy neural network [20]. Piras et al.used heterogeneous artificial neural network models for short term electrical load forecasting [21]. There are multiple other studies that have done a comparison between neuro-fuzzy and neural networks [22][23][24].

IV. Experiment

A. Data analysis

The data used in this experiment was sampled on a monthly basis from January 1985 to December 2011 in South Africa. The demand for electricity has been on the increase over the years in South Africa as illustrated in Figure 3.

The data was normalised to lie between 0 and 1 and then partitioned into windows of sizes [n] from the time series data. The window is then evaluated from month [t - (n - 1)] to month [t]. The training example consisted of a sequence of window size [n] + 1 demand values. The earlier window size [n] demand values make up the attributes for that example, and the latest demand is the realised target example.

As noted by Kaastraet al, and by Kolariket al, that a popular method is to use a sliding window approach [25, 26]. To obtain *n* examples, we have to slide the window *n* steps, extracting an example at each step. Window sizes of 3,4,5,6,7,8,9,10,11,12,13 and 14 months were used for the extraction of the features that were used to train and test the neural networks and neuro-fuzzy models. The forecasting function can be represented by (13).

A neural network toolbox developed by Nabney based on MATLAB was used to conduct the experiments. A neural-fuzzy network developed by Botempi based on MATLAB was used to conduct the experiments. For ARMA a Matlab Toolbox for System Identification was used. One hundred and fiftytraining examples were used for training and one hundred instances were used as out-of-sample data for testing the models.

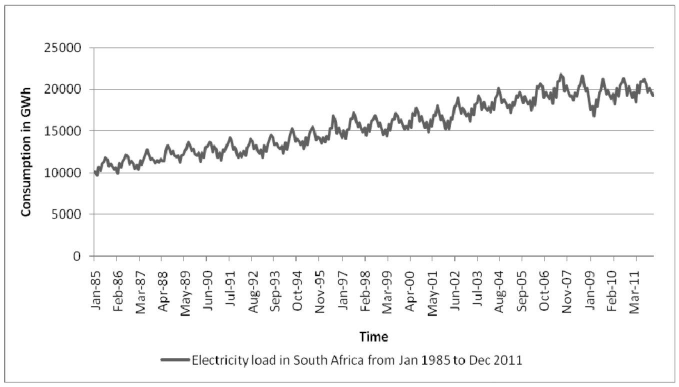

Figure 3: Electricity consumption data from 1985 to 2011

B. *Experimental results*

The experiment consists of two phases: training (supervised learning) and testing (out-of-sample data testing). After training the models different configurations for input-output mapping were found and they were all tested with out-of-sample data.Various accuracy measurement methods such as symmetric mean absolute percentage error (sMAPE), mean absolute percentage error (MAPE) and Root Mean Squared Error were considered for this experiment to compare the differences in their accuracy measurement.Table 1, 2and 3presents the forecasting errors as measured by the various error measurement methods.

1) *ARMA results:* The data was examined for stationarity by plotting autocorrelogram. The data was found to be non-stationary because the plot was decaying slowly and only after perfoming first order differencing was the data found to be stationary because the plot decayed rapidly and decays to zero after 4 lags. The partial autocorrelation function was used to determine the AR lag for the model and it was found to decay to zero at a lag of 5 which means that the model is given as ARMA(5,4). An autocorrelogram plot of the residuals was used to check for model suitability. The autocorrelations of the residuals were found to be insignificant which means the model is suitable. Table 1 presents the out-of-sample results of the ARMA model.

2) *MLP results:* Neural networks models were trained using back-propagation algorithm (3000 training epochs). The architectures of the neural networks models were as shown in table 2. The Model with 12 inputs for MLP produced the most accurate out-of-sample test results as shown in table 3.

3) *ANFIS results:* To train the neuro-fuzzy systems the Gaussian membership function and hard cluster means (HCM) were used for clustering and structure determination. To train the neuro-fuzzy models the error rates were propagated backwards and the parameters were updated by gradient descent method.A neuro-fuzzy system has two types of tuning, namely structural and parametric tuning that are required to build a model. Structural tuning works to find anappropriate number of rules and a proper partition of the input space. Once a satisfactory structure is attained, the parametric tuning searches for the optimal membership functions together with the optimal parameters of the consequent models [27]. In some cases there may be multiple structure and parameter combinations which make the fuzzy model perform in a satisfactory way.

The problem can be formulated as that of finding the structural complexity that is able to give the best generalisation performance [28]. The approach taken in this process chooses the number of rules as the measure of complexity to be properly tuned on the basis of available data. An incremental approach where different architectures having different complexity (i.e. number of rules) are first assessed by cross validation and then compared in order to select the best one was used.The model with 12 inputs for neuro-fuzzy had the most accurate out-of-sample test results as shown in table 4All the models were more accurate when using three rules.

Table 1: ARMA forecasting errors

No of inputs	sMAPE	MAPE	RMSE
5 [y(t-5)]	8.0%	8.2%	0.0098

Table 2: MLP architectures after training

Neural networks architectures
4-6-1
5-6-1
6-7-1
7-8-1
8-8-1
9-10-1
10-11-1
11-12-1
12-12-1
13-13-1
14-14-1

Table 3: MLP forecasting errors

No of Inputs	sMAPE	MAPE	RMSE
4 [y(t-4)]	14.5%	14.9%	0.0212
5 [y(t-5)]	8.1%	8.3%	0.0099
6 [y(t-6)]	13.4%	14.7%	0.0126
7 [y(t-7)]	15.4%	15.8%	0.0163
8 [y(t-8)]	14.8%	15.4%	0.0186
9 [y(t-9)]	13.9%	15.2%	0.0133
10 [y(t-10)]	13.9%	15.2%	0.0131
11[y(t-11)]	12.8%	13.9%	0.0116
12 [y(t-12)]	5.0%	5.7%	0.0075
13 [y(t-13)]	6.7%	7.5%	0.0082
14 [y(t-14)]	7.6%	8.1%	0.0094

Table 4: Neuro-fuzzy forecasting errors

No of Inputs	sMAPE	MAPE	RMSE
4 [y(t-4)]	13%	13.5%	0.0290
5 [y(t-5)]	9.4%	9.0%	0.0115
6 [y(t-6)]	9.1%	8.8%	0.0120
7 [y(t-7)]	14.4%	14.3%	0.0348
8 [y(t-8)]	8%	7.7%	0.0083
9 [y(t-9)]	8.3%	8.1%	0.0087
10 [y(t-10)]	6.1%	5.9%	0.0053
11[y(t-11)]	6.2%	6.0%	0.0056
12 [y(t-12)]	4.3%	4.4%	0.0031
13 [y(t-13)]	5.2%	5%	0.0037
14 [y(t-14)]	5.8%	5.6%	0.0040

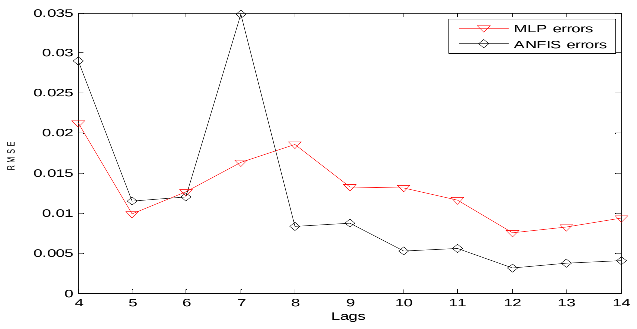

Figure 4: RMSE: MLP errors vs ANFIS errors

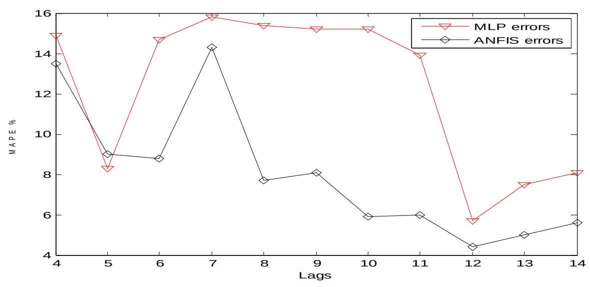

Figure 5: MAPE: MLP errors vs ANFIS errors

C. Discussion

For both techniques, neuro-fuzzy and neural networks models, the model with 12 inputs produces the most accurate results. This seems to indicate that 12 inputs are a better representation of the dynamics of the consumption in a year which makes monthly modeling much more efficient. The results in table 1 and 2 also show that the errors decline as the number of inputs increase until the input size reaches 12 and beyond that the errors begin to increase. It is further shown in the two tables that the neuro-fuzzy model outperforms the neural network model. The comparison, as illustrated in Fig 4 and Fig 5, shows that neuro-fuzzy systems are better able to model the forecasting of energy consumption, which is a non-linear system. The combination of rule based modeling and neural networks makes neuro-fuzzy systems better modeling technique than just using neural networks. The ARMA model was compared with the best performing models of the other two models. The neuro-fuzzy and neural networks demonstrate that they are better able to model non-linear systems than ARMA which assumes linearity of the system.Neuro-fuzzy outperforms neural networks and ARMA and neural networks outperforms ARMA. The results are illustrated in tables 1, 2 and 3.The inability of ARMA to comprehend the non-linear nature of the systems has led to poor performance as compared to the other two AI techniques.

Neuro-fuzzy systems combine human-like representation and the fast learning methods used in neural networks. ANFIS appears to be one of the best tradeoff between neural and fuzzy systems combining fuzzy clustering interpolation and adaptability due to neural network backpropagation training technique. In addition, ANFIS has an advantage oftransparency over neural networks. Whereas neural networksmodels are treated as black boxes because it is difficult to interpret how they are constructed, ANFIS models have rules that can be interpreted. By incorporating the fuzzy rules into the modeling process ANFIS is able to model the uncertainty and the imprecision of the data. For this reason ANFIS has been found to perform better when compared to neural networks [29][30]. The performance of ANFIS in this work was found to be better than that of neural networks which is consistent with the previous findings in other studies [22]. Both neural networks and ANFIS performed better than ARMA which clearly demonstrates the superiority of nonlinear modeling methods as compared to linear methods. ANFIS and ANN approaches are based on the concept of the transformation of a single-variable series in a multidimensionalphase-space to represent the underlying dynamics of the data. This means that they can easily outperform linear methods such as ARMA in a domain that exhibits nonlinear dynamics.

V. CONCLUSION

In this workARMA, Neural networks and neuro-fuzzy systems areused to modeland forecast non-linear systems. An ARMA model and multiple univariate neural networks and neuro-fuzzy systems models were created. The assumption in using the electricity demand data in a univariate model is that the past data of the electricity demand is useful in predicting future electricity demand. The results showed that neuro-fuzzy has a better ability to model the system than ARMA andneural networks and neural networks is in turn better than ARMA.

ACKNOWLEDGMENT

The authors would to thank the Quantec database administrators for providing access to the data

REFERENCES

[1] R. Inglesi and A. Pouris, “Forecasting electricity demand in South Africa: A critique of Eskom’s projections.” S Afr J Sci. 2010;106(1/2), Art. #16, 4 pages. DOI: 10.4102/sajs.v106i1/2.16

[2] E.A. Feinberg and D. Genethliou, “Load forecasting, Applied mathematics for power systems”, pp. 269-285, 2003

[3] K.Y. Lee and J.H. Park, “Short-term load forecasting using an artificial neural network”, Transactions on Power Systems, volume 7, no. 1, 1992.

[4] R.Inglesi-Lotz,"The Sensitivity of the South African Industrial Sector’s Electricity Consumption to Electricity Price Fluctuations,” University of Pretoria Working Paper: 2012-25, 2012.

[5] N.M.Odhiambo, "Electricity consumption and economic growth in South Africa: A trivariate causality test," Energy Economics 31, 635–640, 2009.

[6] H.Amusa, K. Amusa, andR. Mabugu, "Aggregate demand for electricity in South Africa:An analysis using the bounds testing approach to cointegration", Energy Policy, vol. 37, no.10, pp. 4167-4175, 2009.

[7] M.M.B.R.Vellasco, M.A.C. Pacheco, L.S.R.Neto and F.J. Souza, "Electric load forecasting: evaluating the novel hierarchical neuro-fuzzy BSP model," International journal of electrical power and energy systems, 26(2), pp. 131-142, 2004.

[8] G. Box and G. Jenkins, “Time series analysis: Forecasting and control,” 1970.

[9] H. D. Block, “The perceptron: a model for brain functioning”. *Reviews of Modern Physics* **34**(1), 123–135, 1962.

[10] T. Chen and H. Chen, “Universal approximations to nonlinear operators by neural networks with arbitrary activation functions and its application in dynamical systems,” IEEE trans. Neural Networks, vol. 6(4),pp. 911–917, 1995.

[11] C. M. Bishop, “Neural networks for pattern recognition”. Oxford: Oxford University Press. 1997.

[12] J. L. McClelland and D. E. Rumelhart, “Parallel distributed processing: Explorations in the microstructure of cognition,” Psychological and biological models, Cambridge, MA: MIT Press,vol. 2, 1986.

[13] J. Jang and C. Sun, “Functional equivalence between radial basis function networks and fuzzy inference systems.,” IEEE Transactions on Neural Networks, no. 4(1), p. 156-159, 1993.

[14] M. Sentes, R. Babuska, U. Kaymak, and H. van Nauta Lemke, “Similarity measures in fuzzy rule base simplification,” IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, vol. 28, no. 3, pp. 376–386,1998

[15] R. Babuska, Fuzzy modeling and Identification. PhD thesis, Technical University of Delft, Delft, Holland, 1991.

[16] J. W. Taylor, L.M. de Menezes and P. E. McSharr, A comparison of univariate methods for forecasting electricity demand up to a day ahead International Journal of Forecasting, 22 pp 1– 16, 2006.

[17] M.Y. Cho, J.C. Hwang and C.S. Chen,“Customer short term load forecasting by using ARIMA transfer function model”, Proceedings of EMPD '95.,vol 2, 21-23 Nov. 1995

[18] W. R. Christiaanse, “Short-term load forecasting using general exponential smoothing”. IEEE Transactions on Power Apparatus and Systems, PAS-90, 900– 902. 1971

[19] G. A. Darbellay and M. Slama, “Forecasting the short-term demand for electricity - Do neural networks stand a better chance?”, International Journal of Forecasting, 16, pp. 71– 83. 2000.

[20] M. Hanmandlu and B.K. Chauhan, “Load forecasting using Hybrid models”, Power systems, IEEE transactions, pp. 20-29, 26(1), 2011

[21] A. Piras, A. Germod, B. Buchenel, K. Imhof and Y. Jaccard, "Heterogenous artificial neural network for short term electrical load forecasting" , Power systems, IEEE transactions, pp. 397-402, 11(1), 1996

[22]YevgeniyBodyanskiy,Sergiy Popov, and TarasRybalchenko,"Multilayer Neuro-Fuzzy network for short load forecasting"Computer Science – Theory and Applications Lecture Notes in Computer Science Volume 5010, 2008, pp 339-348

[23]M.N.Maralloo, Tehran, A.R Koushki, C Lucas and A.Kalhor, "Long term electrical load forecasting via a neurofuzzy model" Computer Conference, 2009. CSICC 2009. 14th International CSI

[24] , F. Pasila "Multivariate inputs for electrical load forecasting on hybrid neuro-fuzzy and fuzzy C-Means forecaster", Fuzzy Systems, 2008. FUZZ-IEEE 2008.

[25] I. Kaastra and M. Boyd, “Designing a neural network for forecasting financial and economic time series,” Neurocomputing, vol. 10, pp. 215–236, 1996.
[26] T. Kolarik and G. Rudorfer, “Time series forecasting using neural networks, department of applied computer science,” Vienna University of Economics and Business Administration, no. 1090, pp. 2–6, 1997.

[27]G. Bontempi and M.Birattari, TOOLBOX FOR NEURO-FUZZY IDENTIFICATION AND DATA ANALYSIS: For use with Matlab, Iridia, 1999

[28] V. N. Vapnik,The Nature of Statistical Learning Theory, Springer, New York, NY, 1995

[29] Abraham A and Nath B, Designing Optimal Neuro-Fuzzy Systems for Intelligent Control, The SixthInternational Conference on Control, Automation, Robotics and Vision, (ICARCV 2000), Wang J L(Editor), (CD ROM Proceeding, Paper reference 429 - FP7.3 (I), December 2000

[30] Abraham A and Nath B, Connectionist Models for Intelligent Reactive Power Control, AustralasianMATLAB Users Conference 2000, Melbourne, (http://www.ceanet.com.au/mluserconf/papers.asp),
November 2000.