A Hybrid Approach for Time series Forecasting using Deep Learning and Nonlinear Autoregressive Neural Network

Sanam Narejo and Eros Pasero
Department of Electronics and Telecommunications
Politecnico Di Torino
Torino, Italy
Email:{sanam.narejo, eros.pasero}@polito.it

***Abstract*—During recent decades, several studies have been conducted in the field of weather forecasting providing various promising forecasting models. Nevertheless, the accuracy of the predictions still remains a challenge. In this paper a new forecasting approach is proposed: it implements a deep neural network based on a powerful feature extraction. The model is capable of deducing the irregular structure, non-linear trends and significant representations as features learnt from the data. It is a 6-layered deep architecture with 4 hidden units of Restricted Boltzmann Machine (RBM). The extracts from the last hidden layer are pre-processed, to support the accuracy achieved by the forecaster. The forecaster is a 2-layer ANN model with 35 hidden units for predicting the future intervals. It captures the correlations and regression patterns of the current sample related to the previous terms by using the learnt deep-hierarchal representations of data as an input to the forecaster.**

Keywords-Feature Extraction; Deep Belief Network; Time series; Temperature forecasting.

I. INTRODUCTION

Weather forecasting has a long history and over the centuries it has always been a major topic of interest. It still remains an open issue that has a big impact on daily life. In the past, forecasting was simply based on the observation of weather patterns. In recent years the development of time series models and the increase in computational power have completely changed the approach for forecasting, improving the accuracy of the predictions.

Time series forecasting is based on the use of a model to predict future values based on previously observed values. It is obvious that a massive computational power is required to describe and predict the weather because of the chaotic nature of the atmosphere.

Artificial neural networks (ANNs) are one of the most precise and extensively used forecasting models. They have created dynamic applications in economics, engineering, social, foreign exchange, stock problems, etc. The application of neural networks in time series forecasting is based on the ability of neural networks to predict non-stationary behaviors. Traditional mathematical or statistical models are not suitable for irregular patterns of data which cannot be written explicitly in the form of function, or deduced from a formula, whereas ANNs are able to work with chaotic components.

The present paper deals with a new method for multistep time series forecasting. It consists in combining the feature extraction of the input time series through deep learning approach and a nonlinear autoregressive model for multistep prediction for future intervals. The focus of deep architecture learning is to automatically discover significant abstractions, from simplest level features to higher level complex representations of inputs [1]. This ability of deep architectures, to automatically learn the powerful features without using any hand engineered human effort or statistical approach is becoming increasingly popular as the range of applications in machine learning discipline continues to propagate. Temperature is one of the most common parameters for an accurate weather forecast and it has therefore, been selected as a case study for the current work. However, the methodology must be considered as general and applicable to different and larger sets of meteorological parameters.

A literature review is presented in Section II. The theoretical background is described in Section III. Section IV deals with the explanation of the research methodology while Section V presents the obtained results and discussion. The paper ends with conclusions and suggestions for possible future research specified in Section VI.

II. LITERATURE REVIEW

The fundamental aim of time series modeling is to carefully gather the data and thoroughly anticipate the past perceptions of time series to design a suitable model which depicts the genetic construction of a series. In statistical inference, a related topic is regression analysis, which is used to know how much uncertainty is present in a curve that fits the data observed with random errors. It is apparent that effective time series forecasting relies upon proper model fitting. An appropriate consideration should be given to fitting a satisfactory model to the underlying time series.

The research in the literature shows that Autoregressive Moving Average (ARMA) models provide analysis of time series as a stationary stochastic process in terms of two polynomials one for Autoregression and second for moving average [2]. Autoregressive Integrated Moving Average (ARIMA) models and the Box-Jenkins methodology became highly popular in the 1970s among academics. The traditional approaches to time series prediction, such as the ARIMA or Box Jenkins [3]-[7] undertake the time series as generated from linear methods. However, they may be inappropriate if the underlying mechanism is nonlinear. In

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-518-0

fact, the real world systems are often nonlinear. A pretty successful extension of the ARIMA model is the Seasonal ARIMA (SARIMA) [8]. The Seasonality is considered to understand the structure of time series if there exist repeated patterns over known, fixed periods of time within the data set. The restriction of these models is the pre-assumption of the time series in linear practice which is not suitable in real-world scenarios.

A considerable amount of research work has already been accomplished on the application of the neural networks for time series modeling and forecasting. An analysis on the state-of-the-art related to neural networks for time series forecasting is conducted in [9]. ANN is already present in the form of various forecasting models available in the literature [10]-[14]. The most widely recognized and prominent among them are multi-layer perceptrons (MLPs) [4], [9]. Other widely used variations are the Time Lagged Neural Network (TLNN), Recurrent Neural Network (RNN) and its variants.

Recently, the area of Deep Learning has received high attention from the Machine learning researchers. Deep learning has given marvelous performance not only in computer vision, speech recognition, phonetic recognition, natural language processing, semantic classification, but also information and signal processing [15]-[23]. Deep architectures have also shown the state-of-art performance in various benchmark tests [22], [23].

III. THEORETICAL BACKGROUND

In our work, a novel approach is implemented for forecasting the future values of time series data. The work combines the nonlinear feature extraction of input time series through a deep learning approach and nonlinear autoregressive model for multistep prediction for future samples. Meaningful features are extracted from the recorded temperature data series by developing and training Deep Architecture NN, specifically DBN (Deep Belief Network). The extracted features from the hidden layers of DBN form an input set, which is useful for training another model which can foresee future observations and work as a multi step forecaster.

Deep learning belongs to the training of deep architectures, which are composed of multiple levels of non-linear operations, which learn several levels of representation of the input. It is difficult to find the optimal parameter space of deep architectures. Optimization with gradient descent from the random starting point near the origin is not the best way to find a good set of parameters, as random initializations get stuck near poor solutions or local optima [24]. However, the emergence of DBNs holds a great promise to help by addressing the problem of training deep networks with more hidden layers.

DBN deep neural networks are composed of multiple layers of stochastic, unsupervised models such as Restricted Boltzmann Machines (RBMs). These are used to initialize the network in the region of parameter space that finds good minima of the supervised objective. RBMs are well-known probabilistic graphical models. RBMs are constructed on two types of binary units: hidden and visible neurons. The visible units correspond to the components of an observation and constitute the first layer. The hidden units model the dependencies between the components of observations. The layers are constructively added while training one layer at a time, which essentially adds one layer of weights to the network. This retraining of layers follows unsupervised learning at each layer to preserve information from input.

Fine tuning of the whole network is performed specifically, with respect to the subject of interest. The entire procedure is known as greedy layer-wise unsupervised learning to train the network as depicted in Fig. 1. The low level layers extract low level features from raw sensory data, whereas the upper layers of DBN are expected to learn more abstract concepts that explain the input set. The learnt model constructed on the combination of these layers can be used to initialize a deep supervised predictor or a neural network classifier. On the other hand, the learnt representations from the top layers can also be characterized as features that can be used as input for a standard supervised machine learning model.

An RBM with n hidden units and m visible units is a Markov Random field (MRF). Therefore, the joint distribution between hidden variables hi and observed variables vj are given by the Gibbs distribution. Expectations are approximated from the distributions based on Markov chain Monte Carlo (MCMC) technique, i.e., Gibbs sampling. Then the binary states of the hidden units are all computed in parallel using (1). Once binary states are chosen for the hidden units, a "reconstruction" is achieved by setting each vj to 1 with a probability given in (2). Wij is the weight associated between the units vj and hi whereas bj and ci are the bias terms. The change in weight parameter is then given by (3).

$$P(h_{i=1}|v) = sigmoid(\sum^{m}_{j=1} w_{ij} v_j + c_i) \quad (1)$$

$$P(v_{j=1}|v) = sigmoid(\sum^{n}_{i=1} w_{ij} h_i + b_j) \quad (2)$$

$$\Delta wij = 1|v) = \epsilon(<vjhi>_{data} - <vjhi>_{recon}) \quad (3)$$

IV. RESEARCH METHODOLOGY

As stated in the Introduction, our approach depends on following two main aspects: The first step is to create a DBN model which understands the underlying patterns and relations present in the data. This model is capable of producing the abstract features of recorded time series data. Deep learning in our work is achieved through training DBN in a way similar to [22],[23]. The second aspect of our approach is creating a forecasting model which is trained on these highly non-linear hierarchical abstractions. The forecasting model is developed to capture the linear dependencies and the nonlinear Autoregression entity because the time series exist with the natural temporal order of observations. The estimation of future values depends on previous observations available in the record, also including some external effectors and some stochastic term. The work flow of our adapted methodology is illustrated in Fig. 2.

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-518-0

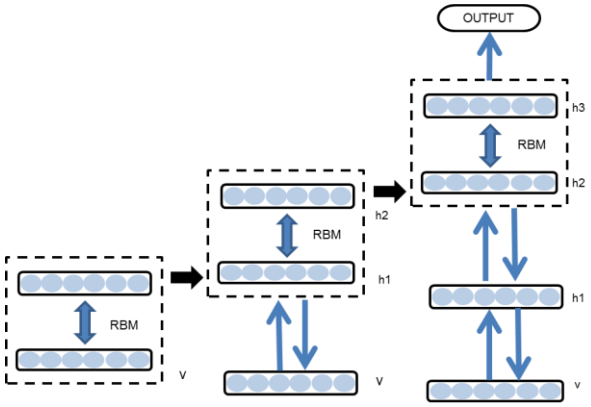

Figure 1. A stack of Restricted Boltzmann Machines (RBMs), greedily trained as Deep Belief Network (DBN).

A. *Data preprocessing*

The data were recorded from Meteo weather station of Neuronica Laboratory at Politecnico di Torino. The samples were taken from 4 October 2010 at 11:45 to 7 August 2015 at 09:15. The temperature was recorded with the frequency of 15 minutes. The data recorded through sensors may include noise, some of missing samples and unwanted frequency fluctuations or outliers. Data was inspected for any outliers prior to the training of the model, because the outliers make it difficult for the neural network to model the true underlying functional form. The missing time steps were replaced by applying a linear interpolation method. Afterwards, the data were filtered with a low pass second order Butterworth filter with a cutoff frequency of 0.11 mHz. Finally, the data was normalized in the range of (0,1). The input set given to the model for extracting the useful features was based on hour, month and a temperature at t-1 and at t-2. Around four years of data upto March 2014 was used to train the DNN. The rest of the data were kept aside to check the performance of the model on unseen samples.

B. *Experimental Setup*

In the first part of our work, a 6-layer DNN architecture was developed with 4 hidden layers, as illustrated in Fig. 3. The last top layer was the output layer which predicts the temperature data. The initial 120000 samples were used to train the DNN. The size of the input layer is 4 in correspondence with the input. The number of units in layers of DNN follows as 4-500-200-100-10-1. The size of each hidden layer was calculated through a Monte Carlo simulation. This section of layers was chosen because it is more efficient to blueprint the structure of DNN while selecting the size of layers either in increasing, decreasing order or keeping the layer size constant throughout the model. The nonlinear hidden units appearing layer-wise in our model are in decreasing order. Each layer was independently trained as RBM with sigmoid activation function. After training the first hidden layer of 500 units, the second hidden layer was added with 200 sigmoid neurons. The input data that used for training the second layer was the outcome from the first layer. The third layer comprising 100 units was trained with the output data from the second layer. Similarly, the last top layer took the abstractions of fourth layer bearing 10 neurons and attempted to predict the temperature as an output. The states of hidden nodes computed by the trained RBM were used as input to the next layer. After unsupervised training, the labels were provided at the output layer for linear mapping. The parameters used for pre-training of each layer are specified in Table I. The pretraining of each layer proceeded with only one epoch. The error and the mean approximations of each layer are presented in Table II.

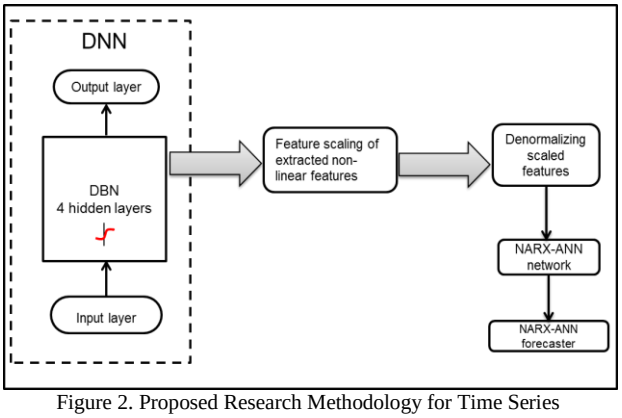

Figure 2. Proposed Research Methodology for Time Series Forecasting.

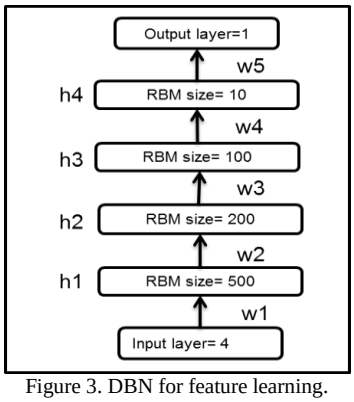

Figure 3. DBN for feature learning.

TABLE I. PARAMETER SETTINGS OF DBN

Parameters	Value
Max Iterations	1
Initial momentum	0.5
Final momentum	0.9
Learning rate	0.1
Batch size	5
Transfer function	Sigmoid

TABLE II. ERROR AND MEAN APPROXIMATION OF PRETRAINING

Layers	RMSE	Mean (hidden units)
Layer 1	0.2744	0.5001
Layer 2	0.0004	0.4986
Layer 3	0.0011	0.4975
Layer 4	0.0024	0.4950
Layer 5	0.0034	0.4992

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-518-0

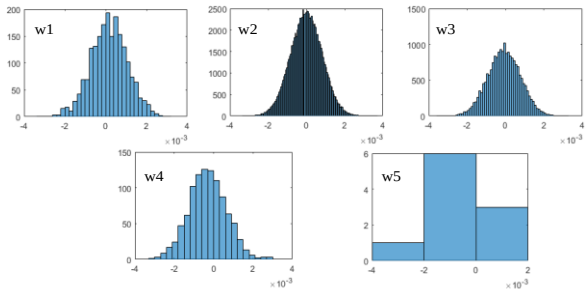

Figure 4. Histograms; Weight Distributions of DBN Layers after pretraining

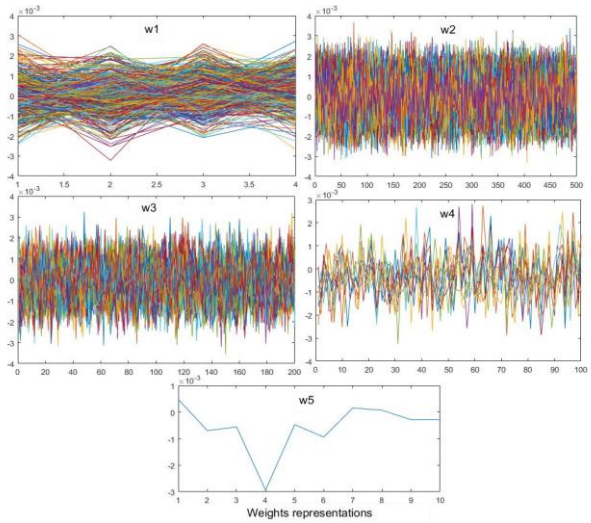

Figure 5. Illustration for weights of DBN after Pretraining

After completing the pretraining, the reformed outcomes as weight distributions associated with each layer are shown in Fig 4. It is clearly perceptible that pretraining initializes the weights and biases of the network to sensible values. The structure of weights connected with each layer as feature detectors of the model is visible in Fig. 5. This parameter initialization with unsupervised greedy layer-wise training expresses that weights linked with each layer are efficient but although not optimal ones. After unsupervised greedy layer-wise training, fine tuning was applied by backpropagation algorithm, training the model with stochastic gradient descent. The pre-training gave a good start to retrain the model by driving the loss function towards its minima.

A total 800 of iterations are used to train the pre initialized DBN model. The weights of model influenced by fine tuning are now modified into different depictions than the earlier ones. The patterns in weight matrices of DNN are illustrated in Fig. 6.

C. *Forecasting*

The task of forecasting temperature on the extracted features from the deep learning architecture was achieved by configuring the Non-linear AutoRegressive model with exogenous inputs i.e. NARX ANN. The literature shows that NARX networks are often much better at discovering long time dependencies than the conventional recurrent neural networks. The NARX feed forward network is created with one hidden layer of 35 neurons and an output layer consisting of one neuron. The size of the hidden layer was selected on the basis of optimal solution given by different models trained in the range of 10 to 40 neurons. The hidden layer activations were processed with a hyperbolic tangent sigmoid transfer function as shown in (4). The output layer was configured with a linear transfer function.

$$tansig(x) = 2/(1+exp(-2*x))-1 \quad (4)$$

The external input set to train the model contained hour, month samples and aggregated learned features from DBN. The preprocessed input was inserted in the network with tapped delay lines to store the previous values of lagged terms for network training. The forecasting model is shown in Fig. 7. Subsequently, once the model is trained to capture the data generation patterns, the model is extrapolated to forecast future values. For forecasting along with the external input another input set is considered, i.e., feedback connection from the network output. This indicates, for predicting multistep samples the predictions of y(t) will be used in place of actual future values of y(t). The architectural view of feedback model is represented in Fig. 8.

The number of previous terms to be used by forecaster can be called as delay terms. The associated delay terms were calculated with Autocorrelation. It describes the correlation between the values of the process at different times, as a function of the two times or of the time lag. By use of autocorrelation the sliding window size of lagged terms is calculated as 4. The data set was divided into training, validation and test set to a ratio of 70, 15 and 15. The network was further trained with Levenberg-Marquardt algorithm with 1000 iterations. The performance of the model was measured by Root Mean Square Error (RMSE) and Mean Square Error (MSE) on training, testing and validation datasets.

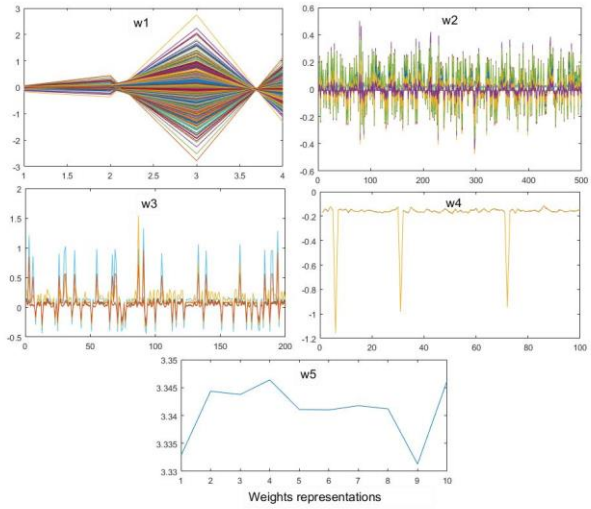

Figure 6. Illustration for weights of DBN-DNN after Fine tunning

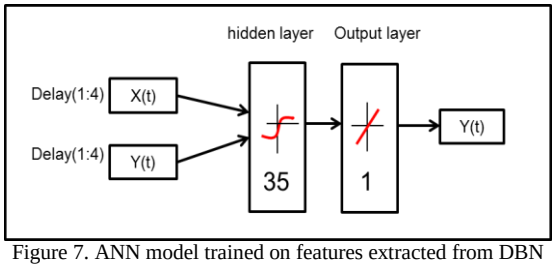

Figure 7. ANN model trained on features extracted from DBN

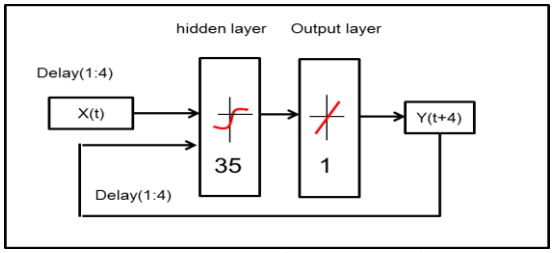

Figure 8. ANN model for Forecasting future

V. RESULTS AND DISCUSSION

A. *Nonlinear Hierarchal Feature Extraction*

The fully trained DNN achieves the error of 8.5e-04 on training data. Further, it results 7.99e-04 on the test set containing 5691 samples. This experiment took more than one day for fine tuning the pretrained model. The simulations were executed on 16 and 32 cores of HPC cluster of AMD Bulldozer CPU. The hierarchal features that are learned with our model are presented in Fig. 9. The graph at the top, shows the abstractions of the top hidden layer. The graph at the bottom, shows the nonlinear relations learned by the first low-level hidden layer. The first hidden layer abstractions are low level attributes that reside in time series of temperature. The top level hidden layer was able to learn the representations as close as possible to the target series which is clearly visible in the figure. Apart from this the top hidden layer extracts 10 valuable features. Along with the property of being significant, the features in the top layer also contained the element of redundancy. Moreover, the features extracted from the top most hidden layer are sent to forecaster for future predictions. These representations give evidence that they are salient for forecasting the future intervals. Several attempts were made to select the appropriate top hidden layer, with the different number of neurons in the range of 1-10. The finest representations learned were with 10 neurons. The prediction of temperature series through DNN is shown in Fig. 10. As seen in the figure, the predicted temperature has accurately replicated the actual temperature records.

B. *Temperature Forecasts*

The performance measurements of the forecasting model for predictions resulted as 0.0010 RMSE on training, 0.0011 on validation and 0.0011 on the test set. After training of the model, it is further extrapolated to forecast next four steps of time interval. As a forecaster for next hour prediction, it provides 0.0068 error performance on the training data while it gives good performance on the test set by achieving 0.0080 as MSE. The statistical analysis for one hour forecasting in the form of error histogram and regression plot is presented in the Fig. 11 and 12. Temperature samples from the test data for next one hour forecasting is shown in Fig. 13. Due to accurate predictions, the actual and forecasted values are almost same. In Fig. 14, only 50 samples are reported to better highlight the difference between actual and forecasted temperature values.

Furthermore, temperature samples from April 2014 to August 2015 to monitor the performance of both models. Feature extraction with DNN resulted in prediction error of 9.8240e-04, which is a good response as compared to the training result of DNN. A useful representations as features are those one, which are significant as an input to a supervised predictor. Henceforth, our model efficiently learned the expressive representations as a feature set for forecasting model. It has the capability to capture possible input configurations which were impossible to identify statistically or mathematically. These findings in our study are highly correlated with [25],[26].

For a comparative analysis of the proposed model with some conventional approaches, the summary of measurements is prescribed in Table III. It presents the relative analysis of our proposed approach with NAR, NARX and MLP. The NAR and NARX belong to the family of dynamic RNNs. The above mentioned models were created and trained by using one hidden layer comprising of 35 neurons, one input layer and one output layer a way similar to the training of Hybrid NARX approach. The rate of training error and test error for the performance of each model is measured in MSE. It is clearly noticeable that the proposed hybrid model outperforms the other models in the form of test error. MLP model is found to be with the least accurate model as compared to the other models.

TABLE III. ERROR RATE COMPARISIONS

Models	Training Error Rate	Test Error Rate
DBN-DNN-NARX	0.0068	0.0080
NAR	0.0056	0.012
NARX	0.069	0.019
MLP	0.045	0.045

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-518-0

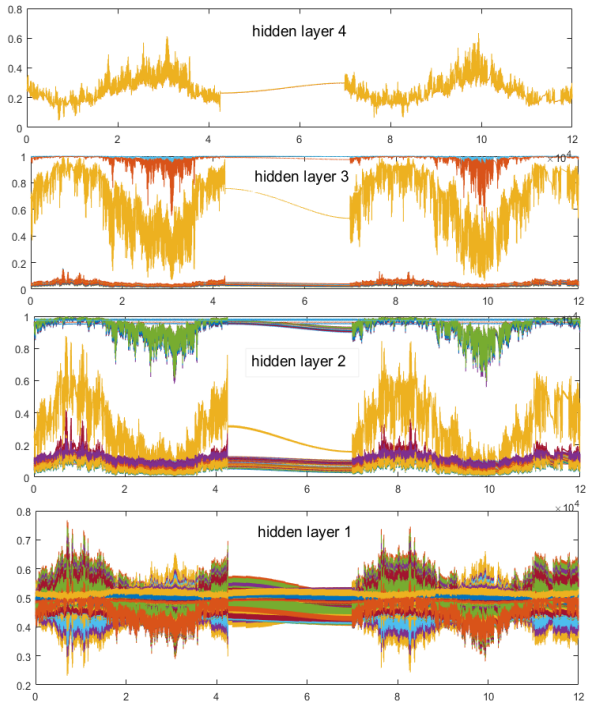

Figure 9. Learnt Nonlinear Representations of data from Hidden layers of DBN.

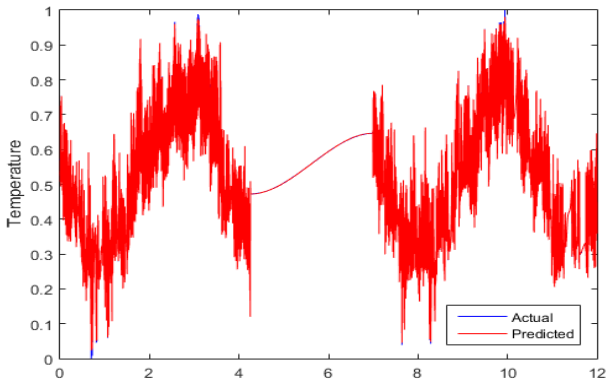

Figure 10. Temperature predictions by output layer of DBN-DNN

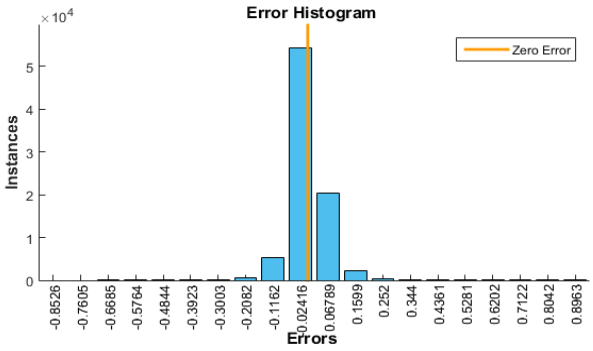

Figure 11. Error Histogram of 1 hour forecast.

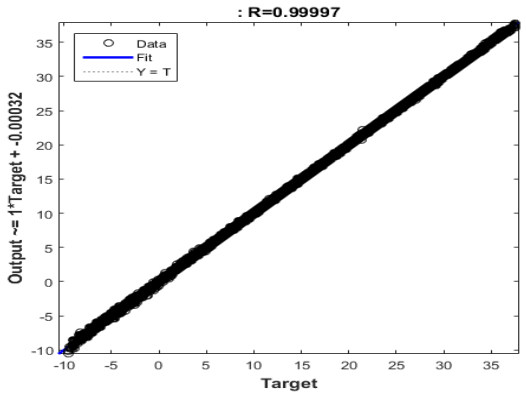

Figure 12. Regression plot for 1 hour forecast.

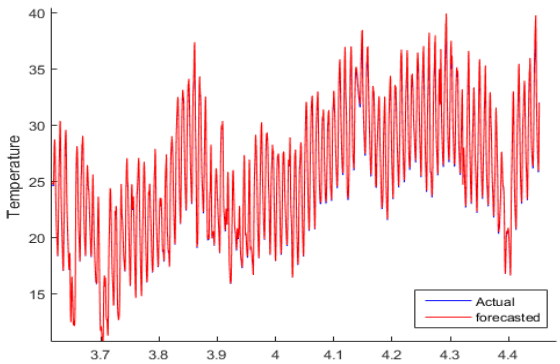

Figure 13. 1 hour Forecasting of Temperature

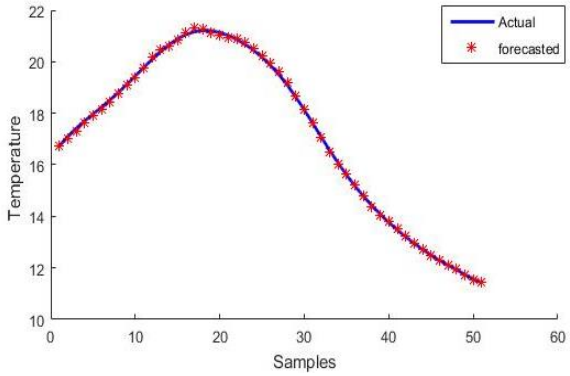

Figure 14. Fifty Samples of Forecasted Temperature.

VI. CONCLUSION

In the paper, a novel approach for temperature forecasting is presented with the aim of improving the prediction accuracy. The main innovation is that the extracted feature set, used for forecasting, is not constructed through statistical feature engineering methods, but it is extracted through the deep learning of a Deep Belief Network. Firstly, the nonlinear hierarchical representations are extracted through the hidden layers of the developed

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-518-0

DBN-DNN model. Subsequently, the raw input series are transformed into gradually higher level of representations as learnt features.These represent more abstract functions of input at the upper level of the layers by implementing a DBN-DNN architecture. The features extracted learnt the complex mapping between input and output, which is observable from the performance of DBN-DNN. The feature extracted are further, used as data to train the forecaster i.e NARX ANN model. The extracted abstractions reinforced the forecaster ANN model to more accurately predict the temperature, achieving outstanding performance against the mentioned approaches. The results obtained over 5 years of data collection demonstrated that the proposed approach is promising and can be further applied to the prediction of a different set of weather parameters.

ACKNOWLEDGMENT

Computational resources were partly provided by HPC@POLITO, (http://www.hpc.polito.it).

This project was partly funded by Italian MIUR OPLON project and supported by the Politecnico of Turin NEC laboratory.

A special thanks to Dr. Suela Ruffa for her precious suggestions.

REFERENCES

[1] Y. Bengio, "Learning deep architectures for AI. Foundations and trends® in Machine Learning, " vol. 2, no. 1, pp. 1-127, 2009.

[2] B. Choi, "ARMA model identification", Springer Science & Business Media, 2012.

[3] G.E. Box, G.M. Jenkins, G.C. Reinsel, and G.M. Ljung, " Time series analysis: forecasting and control. ", Holden-Day, 1976.

[4] G.P. Zhang, " Time series forecasting using a hybrid ARIMA and neural network model.", Neurocomputing, vol. 50, pp. 159-175, 2003.

[5] C. Brooks, "Univariatetime series modelling and forecasting.", Introductory Econometrics for Finance. 2nd Ed. Cambridge University Press. Cambridge, Massachusetts, 2008.

[6] J.H. Cochrane, Time series for macroeconomics and finance. Manuscript, University of Chicago, 2005.

[7] K.W. Hipel, and A.I. McLeod, Time series modelling of water resources and environmental systems, Elsevier. vol. 45, 1994.

[8] B.M. Williams, and L.A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results.", Journal of transportation engineering, vol. 129, no. 6, pp. 664-672, 2003.

[9] Z. Guoqiang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks:: The state of the art.", International journal of forecasting, vol. 14, no. 1, pp. 35-62, 1998.

[10] T. Kolarik, and G. Rudorfer, "Time series forecasting using neural networks. ", In ACM Sigapl Apl Quote Quad vol. 25, no. 1, pp. 86-94, ACM, 1994

[11] G.P, Zhang, "Neural networks for time-series forecasting. ", In Handbook of Natural Computing, Springer Berlin Heidelberg, pp. 461-477, 2012.

[12] J.J.M. Moreno, A.P. Pol, and P.M. Gracia, "Artificial neural networks applied to forecasting time series. ", Psicothema, vol. 23, no. 2, pp. 322-329. 2011.

[13] R.J. Frank, N. Davey, and S.P. Hunt, "Time series prediction and neural networks. ", Journal of intelligent and robotic systems, vol. 31 no. 1-3, pp. 91-103, 2001.

[14] E.G.A. Pasero, and L. Mesin, " Artificial neural networks to forecast air pollution.", pp. 221-240, 2010.

[15] G.E. Hinton, et al, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups". IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82-97, 2012.

[16] R. Salakhutdinov, A. Mnih, and G.E Hinton, " Restricted Boltzmann machines for collaborative filtering. " In Proceedings of the 24th international conference on Machine learning, pp. 791-798. ACM. 2007.

[17] M.R. Amer, B. Siddiquie, C. Richey, and A. Divakaran, "Emotion detection in speech using deep networks.", IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) , pp. 3724-3728, IEEE, 2014.

[18] I. Sutskever, J. Martens, and G.E. Hinton, "Generating text with recurrent neural networks. ", Proceedings of the 28th International Conference on Machine Learning, 2011.

[19] M.D. Zeiler, et al., "Facial expression transfer with input-output temporal restricted boltzmann machines. ",Advances in Neural Information Processing Systems, pp. 1629-1637, 2011.

[20] G.W. Taylor, and G.E. Hinton, "Factored conditional restricted Boltzmann machines for modeling motion style. ", In Proceedings of the 26th annual international conference on machine learning, pp. 1025-1032, ACM, 2009.

[21] G.E. Hinton, and R.R, Salakhutdinov, "Reducing the dimensionality of data with neural networks", Science, vol. 313, no. 5786, pp.504-507, 2006.

[22] G.E Hinton, S. Osindero, and Y.W Teh, " A fast learning algorithm for deep belief nets", Neural computation, vol. 18, no. 7, pp.1527-1554, 2006.

[23] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks. ", Advances in neural information processing systems, vol 19, p. 153, 2007.

[24] R. Salakhutdinov, and G. E. Hinton. "Deep boltzmann machines.", International Conference on Artificial Intelligence and Statistics, 2009.

[25] D. Erhan, et. Al., "Why does unsupervised pre-training help deep learning?. ", Journal of Machine Learning Research, vol. 11, no. Feb, pp. 625-660, 2010.

[26] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, "Exploring strategies for training deep neural networks. ", Journal of Machine Learning Research, vol. 10, no. Jan, pp. 1-40, 2009.

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-518-0