DeepEX: Bridging the Gap Between Knowledge and Data Driven Techniques for Time Series Forecasting *

Muhammad Ali Chattha[1,2,3][0000−0002−3336−5677]*, Shoaib Ahmed Siddiqui[2,3][0000−0003−4600−7331], Mohsin Munir[1,2][0000−0001−7818−0361], Imran Malik[3,4], Ludger van Elst[2], Andreas Dengel[1,2], and Sheraz Ahmed[2][0000−0002−4239−6520]*

[1] TU Kaiserslautern, Kaiserslautern, Germany
[2] German Research Center for Artificial Intelligence (DFKI) GmbH, Kaiserslautern
muhammad_ali.chattha@dfki.de, sheraz.ahmed@dfki.de
[3] School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad, Pakistan
[4] Deep Learning Laboratory, National Center of Artificial Intelligence, Islamabad, Pakistan

Abstract. Artificial Intelligence (AI) can roughly be categorized into two streams, knowledge driven and data driven both of which have their own advantages. Incorporating knowledge into Deep Neural Networks (DNN), that are purely data driven, can potentially improve the overall performance of the system. This paper presents such a fusion scheme, DeepEX, that combines these seemingly parallel streams of AI, for multi-step time-series forecasting problems. DeepEX achieves this in a way that merges best of both worlds along with a reduction in the amount of data required to train these modelsnow. This direction has been explored in the past for single step forecasting by opting for a residual learning scheme. We analyze the shortcomings of this simple residual learning scheme and enable DeepEX to not only avoid these shortcomings but also scale to multi-step prediction problems. DeepEX is tested on two commonly used time series forecasting datasets, CIF2016 and NN5, where it achieves competitive results even when trained on a reduced set of training examples. Incorporating external knowledge to reduce network's reliance on large amount of accurately labeled data will prove to be extremely effective in training of neural networks for real-world applications where the dataset sizes are small and labeling is expensive.

Keywords: Deep Neural Networks · Knowledge Incorporation · Time-Series · Residual Learning.

1 Introduction

Recent advances in computational hardware have made it possible to achieve state-of-the-art performance in various domains, by utilizing DNNs, ranging

* Code available at https://www.github.com/MAchattha4/DeepEX

from image classification [21], playing board games [15], natural language processing [6] to speech recognition [8]. As a result, there is heightened interest, both academically and industrially, in DNNs with deep learning being listed at the top of Gartner hype cycle for emerging technologies [5]. This increased interest coupled with advances in hardware has paved the way for the development of more sophisticated DNN algorithms, which may contain millions of parameters to train and optimize. Version of NASNet-A [21] model, for example, with highest accuracy on ImageNet dataset contains around 88.9M parameters. Optimizing such a huge number of parameters is a challenge itself and requires equivalently bigger training dataset that allows the model to extract enough features to train its parameters. As a result, these models perform exceptionally well in domains where ample data is available but in data scarce domains, these model suffer as they can easily overfit. This is even more so true for time-series domain, where scantiness of data is further compounded by the fact that time-series often do not have enough features for deep networks to work with. Leveraging information present in the form of knowledge can be particularly useful here. These techniques, especially the statistical ones, have shown considerable success in time-series domain which is evident from results of forecasting competitions, like M3 [11], M4 [12] and NN5 [16], which were dominated by statistical based techniques.

In contrast to DNNs, humans tend to rely on their knowledge while solving problems. This knowledge is acquired not only from problem specific examples but also from other sources, like education and experiences [10]. However, the very notion of "knowledge" is tricky to explain and equivalently difficult to collect and store in a form that is understandable or transferable to a computing program. Knowledge-Based Systems (KBS) aims to store such knowledge expressed in the form of logic rules or some other declarative language which can then be used to find solution to complex problems [19]. Similarly, there are statistical methods that are based on strong logical reasoning, like Auto-Regressive Integrated Moving Average (ARIMA), that do perform exceptionally well in their respective domains and are used by many experts to aid them in decision-making process. . .

In fact, complementing DNNs with expert knowledge or some form of extra knowledge has been actively researched upon [20, 3, 9]. Most of the work in the literature, although improves performance of the DNNs but adds extra dependency on the network on quality of expert information used [17, 9]. The focus of this work is to combine knowledge driven and data driven streams in a way that retains advantages of both while suppressing their individual disadvantages. Specifically, we aim to reduce the dependency of DNNs on the data by leveraging information contained in the knowledge stream. Finding state-of-the-art knowledge-based system or DNN model is not the focus here, but instead, the goal is to devise a knowledge incorporation scheme that bridges the gap between data and knowledge driven approaches and combines their strengths. Chattha et al. (2019) [4] recently introduced Knowledge Integrated Neural Network (KINN), a residual framework that combined information contained in the

knowledge stream with the data stream in order to reduce the dependence of DNN on large amount of accurately labeled data. However, KINN [4] failed to produce acceptable results on benchmark time-series datasets. KINN particularly suffered when dealing with time-series that encapsulated significant trend variation. This resulted in poor performance on more sophisticated time-series datasets. In this paper, we present DeepEX, that not only addresses the shortcomings of KINN [4], but also strengthens the network allowing information in the two streams to complement each other. We tested DeepEX on the CIF2016 and NN5 time-series forecasting benchmark datasets to signify its performance. In particular, following are the contributions of DeepEX:

- We introduce a novel approach to combine knowledge and data driven systems in an end-to-end learning framework
- We introduce new regularization on the activity of the network that helps the network in identifying strengths and weakness of both domains and decide optimal combination of both
- Introduction of a new network to capture the trend in order to decompose the problem into sub-problems which can be effectively solved
- Scale DeepEX to multi-step ahead prediction which is significantly difficult for the current generation of expert knowledge incorporation techniques

The rest of the paper is structured as follows. We will first briefly cover the previous literature in the direction of expert knowledge incorporation in Section 2. Then we will describe the proposed method in detail in Section 3. We will present the results from the various experiments, and discuss the findings in Section 3.5. Finally, we conclude the paper with concluding remarks in Section 3.6.

2 Related Work

Integrating domain knowledge or any sort of extra information to boost DNN's performance has been actively researched upon. Ghazvininejad et al. (2018) [7] presented a knowledge-grounded conversation model based on neural networks. In addition to utilizing data containing the conversation history, they also conditioned the output of their sequence-to-sequence (seq2seq) model on external details within the context of conversation. The resulting conversation model was able to produce more accurate responses that were labeled as more informative and appropriate by human judges. Although such schemes encouraged knowledge incorporation to improve the performance of the system, however, it made the system more dependent on both the external contextual information data.

Knowledge-based Artificial Neural Networks (KBANN) were proposed by Towell et al. (1994) [17]. They utilized propositional rules for knowledge representation which were structured in a hierarchical manner. The neural network was designed to have a one-to-one correspondence with the elements of the rule set. The rule set directly defined the number of neurons along with their corresponding weights. Additional neurons were also introduced to learn features not

specified in the rule set. Tran et al. (2018) [18] followed a similar approach where the network defined a logic rule set. Such techniques directly incorporates the information contained in the knowledge stream into the neural network. However, as a result of this direct incorporation, the network is confined to a structure that strictly complies to the hierarchical structure defined in the rule set. Additionally, this also abolishes the flexibility to be able to use different network architectures.

Venugopalan et al. (2016) [20] also proposed a neural network based video descriptor model that leveraged knowledge from both a neural language model as well as semantics obtained from a large text corpus in a LSTM based architecture. The results demonstrated significant improvements in grammar while also improving the overall descriptive quality. They introduced two fusion techniques, namely Late fusion and Deep fusion where they concatenated the hidden states from both video to text network and language LSTM network, fusing the information contained in both of the domains. The system is strongly dependent on the quality of the expert which in turn is dependent of large amount of data.

Buda et al. (2018) [3] used statistical forecasting models to aid neural network in producing forecasting results for an anomaly detection problem. They utilized multiple statistical forecasting models in conjunction with deep learning model. Predictions made by all of these individual models were combined into one framework. The predicted values from all models were compared with the ground-truth and value giving the lowest Root Mean Square Error (RMSE) score was selected as the final prediction. They refer this approach as single-step merge. Another voting based approach was also proposed where RMSE score is used to select a single model for all of the predictions. The system treats the predictions form the individual models separately, hence, it not able to leverage the advantages from both streams simultaneously. Munir et al. (2019) [13] also used statistical methods to enhance performance of the neural network in anomaly detection problem. Here auto-ARIMA was employed to forecast future values of the time series. These predictions were then integrated into neural network by using a residual scheme. The resulting model, FuseAD, achieves better performance compared to individual networks when used separately.

Hu et al. (2016) [9] again leveraged expert knowledge in the form of first order logic rules. Iterative knowledge distillation technique is used to transfer knowledge to network parameters. The expert network acts as a teacher network to the student network, the DNN. DNN tries to follow the teacher network by mimicking its predictions. Both the student and the teacher networks are updated at each iteration step. The goal is to find the best teacher network that fits the prediction of the rule set while also staying close to prediction made by the student network. KL-divergence between the probability distribution of the predictions made by the teacher network and the output distribution of predictions made by the student network is used to minimize the difference between the two distributions. The proposed framework achieved state-of-the-art performance for the evaluated classification tasks. However, the framework

strongly relied on the expert network and as the student network is trying to emulate predictions made by the teacher network.

Chattha et al. (2019) [4] proposed a residual learning scheme, called as KINN, where they incorporated expert knowledge in the form of prediction in the network by adding it to the network's output. Although the approach is highly promising, it couldn't be scaled for multi-step predictions. The first limitation is its inability to control the network's correction factor. The network makes useless corrections even in cases where it is not necessary. The second limitation is its inability to cope up with trend present in the sequence. This proves to be an impediment in the production of convincing results for complex time-series data. DeepEX addresses both these limitations.

3 DeepEX: The Proposed Method

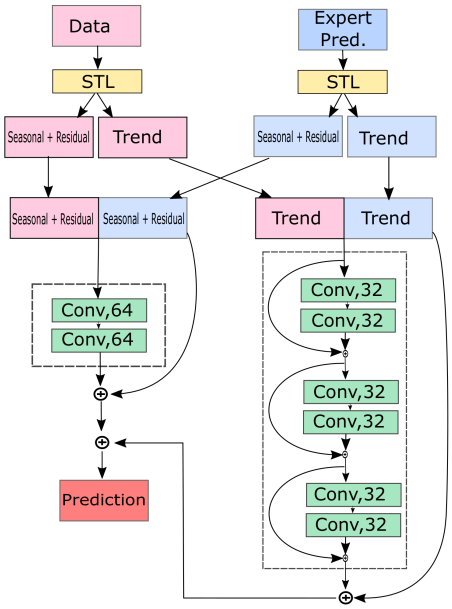


Fig. 1. Overview of DeepEX architecture

Fig. 1 shows the overall architecture of the DeepEX framework. There are several different components in DeepEX each targeted to cater for specific task. The first component is the expert module which contains information about expert opinion. The second component is the STL decomposition module which decomposes the input signal into its constituent parts. Finally, there are two CNN models, where one model is dedicated to handle the trend part while the other one handles the remaining signal.

The data and the expert model output is log normalized before feeding it to the STL decomposition layer. Log normalization has two major advantages: (i) re-scaling values and (ii) transformation of the multiplicative relation between the STL components to an additive one, which makes it easier to decouple the decomposed components. We will now describe each of these components in detail.

3.1 STL Decomposition

The Seasonal and Trend decomposition using Loess (STL) is a well-known time-series decomposition method which splits a time-series into three components namely (i) trend, (ii) seasonality, and the (iii) residual. In DeepEX, input data and output of the expert model are both fed into the STL decomposition module and trend from each of the signals is extracted from the rest of the signal. Residual and seasonal components are added together since only the trend is of relevance to us. Hence, output of STL decomposition contains two signals, trend and the rest of the signal which is a de-trended version of the input comprising of the seasonal and the residual components. These signals are then given to their respective CNN estimators as inputs. Although it is a common notion that neural networks are capable of modeling complex structures in data owing to their strong self adapting generalizing capabilities, more recent studies argue decomposing input signal or filtering out some component prior to modelling can produce better forecasting results [14, 16, 1]. Hence, we opt for a similar approach and decompose the original signal into two relatively less complex components.

3.2 Expert Model

Knowledge driven techniques offer their own advantages. Knowledge, however, can take many shapes and forms. It can be in the form of a human expert, logic rules or even some statistical method. One of the strengths of DeepEX is that it does not limit itself to any specific knowledge model as it is not dependent on architecture of the expert model but rather its predictions. Therefore, any knowledge model capable of producing predictions can be used. The expert model can be human feedback integrated into the system or a KBS or some other technique. For this particular paper, we used the 4Theta method[5] as our expert network. 4Theta is based on theta model that decomposes the original signal into theta lines, where theta lines are derived by modifying the local curvature of the time-series through the coefficient θ. This θ is then applied to the second differences of the data. 4Theta is an improvement to the Theta model, enabling it to handle complex time-series more efficiently which is evident from its performance M4 benchmark dataset [12].

[5] https://github.com/M4Competition/M4-methods/blob/master/005%20-%20vangspiliot/Method-Description-4Theta.pdf

3.3 CNN Models

DeepEX has two different CNN models aimed to estimate trend and de-trended signal which are obtained after STL decomposition. Although the focus of this work was to develop a knowledge incorporation technique and not the individual DNN or knowledge models but considerable effort has been invested in estimating hyperparameters of DNN. It was particularly observed that simple CNN model struggled the most in modeling the trend component of the signal, hence, model responsible for the estimation of trend is relatively complex compared to the seasonal and residual signal estimator. The trend estimation network comprises of three residual blocks, each containing two convolutional layers with 32 filters each. The other CNN model comprises of two convolution layers with each layer each having 64 filters. It is important to mention that although we have chosen convolutional neural network as our DNN, because CNNs are generally easier to optimize, DeepEX is flexible enough to work with any other DNN architecture.

3.4 Knowledge Incorporation Scheme

Expert predictions are incorporated in the form of a residual scheme, where expert predictions are added to the output of the DNN model and are also used for conditioning the DNN. This conditioning is achieved by sequentially stacking the expert predictions ($x_t^{tr'}$) to the input from the data. The proposed residual scheme changes the underlying mapping learned by the network and instead of learning the complete input to output projection, the network only learns the modification factor needed in the input to give the desired output. In a way it can be said that the DeepEX framework estimates efficacy of expert model and makes corrections to it by using information from the data. As we have used a statistical method (Section 3.2) as our expert model, some portion of the data (25% of the training set) is used to estimate parameters of 4Theta model. The resulting expert model is then used for making predictions on remaining portion of the dataset, by employing a rolling window approach where forecast for the next horizon is obtained by using all of the previous data. Hence, DeepEX is trained on 75% of the training data, which is the only portion of the dataset where expert predictions are available, since we do not have any expert predictions on 25% of the data on which 4Theta is trained. It should be noted that the test set was never used in either estimating parameters of the 4Theta model or in training DeepEX. Validation set was obtained by $\max(0.2 * |X|, H)$ where $|X|$ denotes the cardinality of the training set, while H denotes the horizon.

The input signal and expert predictions both are decomposed using STL (Section 3.1). Trend from both, the expert predictions and the data is fed into CNN responsible for trend estimation. This optimization problem for the trend estimating CNN ($\Phi : \mathbb{R}^h \mapsto \mathbb{R}^h$) can be represented as:

$$[\hat{x}_t^{tr}, \hat{x}_{t+1}^{tr}, .., \hat{x}_{t+h-1}^{tr}] = \Phi([x_{t-1}^{tr}, x_{t-2}^{tr}, ..., x_{t-w}^{tr}; x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}]; \mathcal{W}) + [x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}] \quad (1)$$

where $x_t^{tr'}$ represents the decomposed trend values predicted by the expert model, x_{t-1}^{tr} represents the decomposed trend values of the original input and $\hat{x}_t^{tr}$ output trend values by the network. As the network is just producing an offset, i.e the required change in the input signal to produce the desired output, instead of finding the complete input to output mapping it makes the optimization problem significantly easier to tackle. Expert predictions are incorporated in a similar fashion as in case of other CNN model. Finally, the overall output of the system is the sum of the output of the two CNNs, which can be represented as:

$$[\hat{x}_t, \hat{x}_{t+1}, .., \hat{x}_{t+h-1}] \quad = \quad [\hat{x}_t^{tr}, \hat{x}_{t+1}^{tr}, .., \hat{x}_{t+h-1}^{tr}] \; + \; [\hat{x}_t^{sr}, \hat{x}_{t+1}^{rr}, .., \hat{x}_{t+h-1}^{rr}] \quad (2)$$

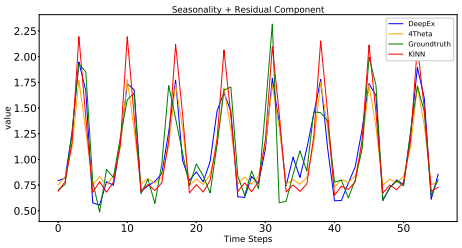
where $\hat{x}_t^{sr}$ represents the output from the trend network, $\hat{x}_t^{rr}$ represents the output of the seasonality and residual network, and $\hat{x}_t$ represents the output of the overall system. Both of the CNN models are optimized separately. A regularization term β is also added on top of the network activations in order to hinder the network from making unnecessary modifications. Therefore, The optimization problem for the CNN can be represented as:

$$\begin{aligned} \mathcal{W}^* = \arg\min_{\mathcal{W}} \|[x_t^{tr}, x_{t+1}^{tr}, .., x_{t+h-1}^{tr}] - \\ \big(\Phi([x_{t-1}^{tr}, x_{t-2}^{tr}, ..., x_{t-w}^{tr}; x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}]; \mathcal{W}) + [x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}]\big)\|_2 \\ + \beta \|\Phi([x_{t-1}^{tr}, x_{t-2}^{tr}, ..., x_{t-w}^{tr}; x_t^{tr'}, x_{t+1}^{tr'}, ..., x_{t+h-1}^{tr'}]; \mathcal{W})\|_2 \quad (3) \end{aligned}$$

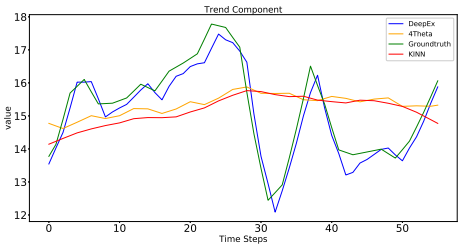
where β is a hyperparameter which controls the activity of the network. This formulation is used for both of the CNN models and the value of β is obtained via validation.

3.5 Results and Discussion

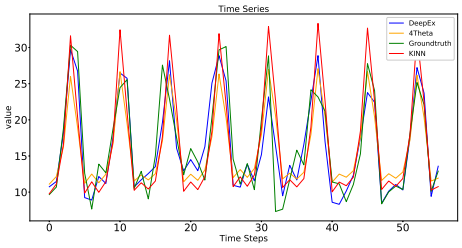
CIF2016 and NN5 forecasting benchmark datasets were chosen to evaluate the performance of DeepEX. Fig. 2 shows the performance of DeepEX on a randomly selected time-series from the NN5 dataset. It is evident from the Fig. 2b that DeepEX does a better job at following the trend of the time-series compared to the expert network, which struggled in correctly modeling the magnitude of the peaks. KINN [4] also closely followed the expert model. Similar pattern can be observed from Fig. 2a where DeepEX was able to capture minor variations in seasonal and residual components, especially in cases of minimas, as compared to the other models. DeepEX had a Symmetric Mean Absolute Percentage Error (SMAPE) score of 15.04 on this particular time-series whereas the SMAPE score of the expert model was 22.40, highlighting the efficacy of DeepEX in modeling the sequence. This dominance of DeepEX was found to be consistent on the entire dataset.



(a) Prediction of seasonal and residual components



(b) Prediction of trend component



(c) Overall prediction of the time-series

Fig. 2. Prediction of DeepEX and other networks on a randomly selected time-series from the NN5 dataset with a horizon 1

We performed a series of experiments in order to validate the effectiveness of DeepEX's knowledge incorporation scheme in helping DNN to reduce its dependence on data, along with its ability to scale to multi-step ahead prediction. As mentioned previously, for the first set of experiments, only 75% of the training data was utilized to train parameters of the DNNs. For NN5 dataset, the performance was evaluated for a horizon of 1, 3, 8 and 56. While for CIF2016, the performance was evaluated for a horizon of 1, 3 and 6/12. In the next set of experiments, training data was further reduced to 50% while the horizons were kept same. When the size of the dataset was reduced to half, many time-series in CIF2016 became so small that even having a horizon of one in the validation set was not possible. Hence, in this particular experimental setting, CIF2016 dataset was not evaluated for a horizon of 6/12.

Table **??** shows the results of the aforementioned experiments. In almost all of the experiments, DeepEX achieved lower SMAPE score compared to the other techniques. Even in data scarce scenarios, DeepEX showed an improvement of almost 46% in terms of SMAPE score when trained on only 50% of the data from the NN5 dataset with a horizon of 1. Similarly, for the same experimental setting, it showed an improvement of 38% in case of CIF2016 dataset. It was also observed that for bigger horizon, the percentage gain in terms of SMAPE was lower compared to experiments with smaller horizon since the complexity of the task was significantly enhanced. Nevertheless, even in these cases, DeepEX

Table 1. Results of CIF2016 dataset of different techniques ordered by mean SMAPE

Method	Mean SMAPE
LSTM.Cluster	10.53
LSTMs and ETS	10.83
ETS	11.87
MLP	12.13
REST	12.45
ES	12.73
DeepEX(trained on 75% data)	**12.80**
FRBE	12.90
HEM	13.04
Avg	13.05
BaggedETS	13.13
LSTM	13.33
Fuzzy c-regression	13.73
PB-GRNN	14.50
PB-RF	14.50
ARIMA	14.56
Theta	14.76

still outperformed other techniques. KINN [4] particularly struggled on these datasets as could not handle time series with trend component.

We compared DeepEX with the top performing techniques for both of these competitions i.e NN5 and CIF2016. Table 1 shows the comparison of results on CIF2016 dataset. DeepEX trained with 75% of the training data outperformed most of the other techniques, including BaggesETS [2], ARIMA and Theta methods, which were considered as benchmarks, in the competition and achieved comparative performance with that of the top performing models.

Table 2 shows the results obtained on the NN5 dataset including both DeepEX and other state-of-the-art models. Similar to the case of CIF2016, DeepEX trained on 75% of the data outperformed most of the techniques and is even slightly better then *LSTM.Cluster* [1] which was the best performing model for the CIF2016 dataset. This demonstrates the robustness of DeepEX and its ability to work on a different datasets.

3.6 Conclusion

We have presented a new knowledge incorporating residual framework that combines best of both knowledge as well as data driven approaches. In particular the aim of this work was to use information contained in the knowledge stream to reduce the dependence of DNNs on large amount of data without compromising the performance. Results obtained by DeepEX show that DeepEX not only alleviates data dependence but also significantly boosts the performance of the network. DeepEX trained on only 75% of the data ranked at 6th place overall in the NN5 competition and 7th in the CIF2016 competition. This is achieved by separating the forecasting model into two different components where the first one captures the trend while the second one captures the rest of the signal. Finally, these two outputs are added to the prediction made by the expert. Regularization term is also added that controls the activation of the DNN model

Table 2. Results of NN5 dataset of different techniques ordered by Mean SMAPE

Name	Mean SMAPE
Wildi	19.9
Andrawis	20.4
Vogel	20.5
D'yakonov	20.6
Noncheva	21.1
DeepEX(trained on 75% data)	**21.4**
LSTM.Cluster	21.6
Rauch	21.7
Luna	21.8
Lagoo	21.9
Wichard	22.1
Gao	22.3
LSTM.All	23.4
Puma-Villanueva	23.7
Autobox(Reilly)	24..1
Lewicke	24.5
Brentnall	24.8
Dang	25.3
Pasero	25.3
Adeodato	25.3
undisclosed	26.8
undisclosed	27.3
Tung	28.1
Naive Seasonal	28.8
DeepEx(trained on 50% data)	**32.8**
undisclosed	33.1

which inhibits the neural network from making unnecessary modifications. High rank achieved by DeepEX on different datasets belonging to different domains demonstrates that the model indeed generalizes well by leveraging a combination of the two streams.

This is only a step in the direction of merging knowledge and data driven techniques. There is still a large room for improvement, particularly in cases where forecasting horizon is large.

4 Acknowledgements

This work is partially supported by Higher Education Commission (Pakistan), "Continental Automotive GmbH" and BMBF project DeFuseNN (Grant 01IW17002).

References

1. Bandara, K., Bergmeir, C., Smyl, S.: Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. arXiv preprint arXiv:1710.03222 (2017)
2. Bergmeir, C., Hyndman, R.J., Benítez, J.M.: Bagging exponential smoothing methods using stl decomposition and box–cox transformation. International journal of forecasting **32**(2), 303–312 (2016)
3. Buda, T.S., Caglayan, B., Assem, H.: Deepad: A generic framework based on deep learning for time series anomaly detection. In: PAKDD. pp. 577–588. Springer (2018)

4. Chattha, M.A., Siddiqui, S.A., Malik, M.I., van Elst, L., Dengel, A., Ahmed, S.: Kinn. arXiv preprint arXiv:1902.05653 (2019)
5. Columbus, L.: Gartners hype cycle for emerging technologies, 2017 adds 5g and deep learning for first time. Forbes/Tech/# CuttingEdge (2017)
6. Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A.: Supervised learning of universal sentence representations from natural language inference data. arXiv preprint arXiv:1705.02364 (2017)
7. Ghazvininejad, M., Brockett, C., Chang, M.W., Dolan, B., Gao, J., Yih, W.t., Galley, M.: A knowledge-grounded neural conversation model. In: AAAI (2018)
8. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal processing magazine **29**(6), 82–97 (2012)
9. Hu, Z., Ma, X., Liu, Z., Hovy, E., Xing, E.: Harnessing deep neural networks with logic rules. arXiv preprint arXiv:1603.06318 (2016)
10. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction. Science **350**(6266), 1332–1338 (2015)
11. Makridakis, S., Hibon, M.: The m3-competition: results, conclusions and implications. International journal of forecasting **16**(4), 451–476 (2000)
12. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: The m4 competition: Results, findings, conclusion and way forward. International Journal of Forecasting (2018)
13. Munir, M., Siddiqui, S.A., Chattha, M.A., Dengel, A., Ahmed, S.: FuseAD: Unsupervised Anomaly Detection in Streaming Sensors Data by Fusing Statistical and Deep Learning Models. Sensors **19**(11) (2019)
14. Nelson, M., Hill, T., Remus, W., O'Connor, M.: Time series forecasting using neural networks: Should the data be deseasonalized first? Journal of forecasting **18**(5), 359–367 (1999)
15. Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al.: Mastering the game of go without human knowledge. Nature **550**(7676), 354 (2017)
16. Taieb, S.B., Bontempi, G., Atiya, A.F., Sorjamaa, A.: A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition. Expert systems with applications **39**(8), 7067–7083 (2012)
17. Towell, G.G., Shavlik, J.W.: Knowledge-based artificial neural networks. Artificial intelligence **70**(1-2), 119–165 (1994)
18. Tran, S.N., Garcez, A.S.d.: Deep logic networks: Inserting and extracting knowledge from deep belief networks. IEEE transactions on neural networks and learning systems **29**(2), 246–258 (2018)
19. Ullman, J.D.: Principles of database and knowledge-base systems, vol. 1. Computer Science Press, Incorporated (1988)
20. Venugopalan, S., Hendricks, L.A., Mooney, R., Saenko, K.: Improving lstm-based video description with linguistic knowledge mined from text. arXiv preprint arXiv:1604.01729 (2016)
21. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: (CVPR) (June 2018)