Dynamic cyclone wind-intensity prediction using co-evolutionary multi-task learning

Rohitash Chandra

Centre for Translational Data Science
The University of Sydney, NSW 2006, Australia.
rohitash.chandra@sydney.edu.au

Abstract. A new category called dynamic time series prediction is introduced to address robust "on the fly" prediction needed in events such as natural disasters. A co-evolutionary multi-task learning algorithm is presented which incorporates features from modular and multi-task learning. The algorithm is used for prediction of tropical cyclone wind-intensity. This addresses the need for a robust and dynamic prediction model during the occurrence of a cyclone. The results show that the method addresses dynamic time series effectively when compared to conventional methods.

Keywords: Backpropagation, modular network design, multi-task learning, modular pattern classification.

1 Introduction

In prediction for climate extremes [15,21,1,25], it is important to develop models that can make predictions dynamically, i.e robust "on the fly" prediction given minimal information about the event. In time series prediction, it is typical to reconstruct the time series into a state-space vector defined by fixed embedding dimension and time lag. The embedding dimension defines the minimal timespan or number of past data-points needed to make a prediction. This can be seen as a limitation since data from the past cyclones mostly considered readings every 6 hours [14]. Therefore, the model should have the feature to make timely prediction as soon as the cyclone has been detected. *Dynamic time series prediction* is defined as the ability of a model to make prediction for different number of input features or set of values of the embedding dimension. It has been highlighted in recent work [9] that recurrent neural networks trained with a predefined embedding dimension can only provide effective prediction for the same embedding dimension which makes dynamic time series prediction a challenging problem. A way to address such categories of problems is to develop robust models that can make prediction given different values of embedding dimension that define the number of features in the input. The overlapping information in different groups of features can harnessed through shared knowledge representation.

Multi-task learning employs shared representation knowledge for learning multiple instances from the same problem [3,10,24,23]. In the case of time series, multi-task learning can consider different a set of embedding dimensions as tasks that have shared knowledge representation. Neuro-evolution refers to the use of evolutionary algorithms for training neural networks [2]. Cooperative coevolution [19] has been a prominent neuro-evolution methodology where the neural network is decomposed into modules [18]. The modules are implemented as sub-populations and been used for time series prediction [4,7] Modular neural networks are motivated from repeating structures in nature [13,8]. Although neuro-evolution has been successfully applied for training neural networks, multi-task learning for enhancing neuro-evolution has not been fully explored. Evolutionary multi-task learning has been used for enforcing modularity in neural networks that can be beneficial when certain modules are perturbed or damaged [6]. There has not been any investigation that explores the embedding dimension of a time series as subtasks for multi-task learning which can be beneficial for dynamic time series prediction.

In this paper, a co-evolutionary multi-tasking learning algorithm is proposed that provides a synergy between multi-task learning and co-evolutionary algorithms. It enables neural networks to feature shared knowledge representation while retaining modularity for robust "on the fly" prediction considered to be dynamic time series. The original time series is reconstructed with a set of values for the embedding dimension that defines the subtasks for multi-task learning. Here, the tasks are referred as subtasks to avoid confusion with typical multi-task learning where a set of datasets define the tasks. The proposed method is used for tropical cyclone wind-intensity prediction and addresses the problem of dynamic prediction.

The rest of the paper is organised as follows. Section 2 gives details of the co-evolutionary multi-task learning method for dynamic time series prediction. Section 3 presents the results with a discussion. Section 4 presents the conclusions and directions for future research.

2 Co-evolutionary Multi-task Learning

2.1 Dynamic Time Series Prediction

In state-space reconstruction, the original time series is divided using overlapping windows at regular intervals that can be used for one-step-ahead prediction. Taken's theorem expresses that the state-space vector reproduces important characteristics of the original time series [22]. Hence, given an observed time series $x(t)$, an embedded phase space $Y(t) = [(x(t), x(t-T), ..., x(t-(D-1)T)]$ can be generated, where, T is the time delay, D is the embedding dimension (window), $t = (D-1)T, DT, ..., N-1$, and N is the length of the original time series. The optimal values for D and T must be chosen in order to efficiently apply Taken's theorem [11]. Taken's proved that if the original attractor is of dimension d, then $D = 2d + 1$ will be sufficient to reconstruct the attractor [22]. In the case of using feedforward neural networks, D is the a number of input neurons. As defined earlier, dynamic time series prediction refers to the ability of a model to make prediction for different number of input features or set of values of the embedding dimension. Therefore, one step ahead prediction $\hat{y}_{t+1}$ for subtask Ω_m that feature the respective embedding dimension can be given by

$$\begin{aligned} \Omega_m &= x_t, x_{t-1}, ..., x_{t-m} \\ \hat{y}_{t+1} &= f(\Omega_m) \end{aligned} \tag{1}$$

where $f(.)$ is a model such as a feedforward neural network, m is defines the length of the input features, where $m = 1, 2, ..., M$, and M is the number of subtasks.

2.2 Algorithm

The proposed algorithm is inspired by the strategies used in dynamic programming where a subset of the solution is used as the main building block for the optimisation problem. It enables neural networks to feature shared knowledge representation while retaining modularity for robust "on the fly" prediction considered to be dynamic time series. The original time series is reconstructed with a set of values for the embedding dimension that defines the respective subtasks. The problem is in sequentially learning the modules of a neural network that link with their subtasks. The base network module features lowest number of input features and hidden neurons. The base network module is part of larger cascaded neural network architecture that expands in size as more subtasks are considered. Additional network modules are required for the subtasks that have shared knowledge representation, hence, the method is called *co-evolutionary multi-task learning* (CMTL). The algorithm considers learning cascaded network module θ_m which is composed of network module Φ_m. The input time series is defined by embedding dimension that is referred as a subtask Ω_m. The cascaded network architecture can also be viewed as an ensemble of neural networks that feature distinct topologies as shown in Figure 1. The output layer employs one neurons in all the respective subtasks since the overall problem is one-step ahead prediction. The input-hidden layer ω_m weights and the hidden-output layer υ_m weights are combined for the respective network module Φ_m. The base module is given as $\Phi_1 = [\omega_1, \upsilon_1]$. Multi-task learning is used via coevolution to update the respective network module Φ_m given subtask Ω_m. Note that the cascaded network module θ_m of subtask m is constructed by combining with current Φ_m and previous network module Φ_{m-1} as follows.

$$\begin{aligned} \Phi_1 &= [\omega_1, \upsilon_1]; \;\; \theta_1 = (\Phi_1) \\ \Phi_2 &= [\omega_2, \upsilon_2]; \;\; \theta_2 = [\theta_1, \Phi_2] \\ &\vdots \\ \Phi_M &= [\omega_M, \upsilon_M]; \;\; \theta_M = [\theta_{M-1}, \Phi_M] \end{aligned} \tag{2}$$

The list of network modules considered for training or optimisation is therefore $\mathbf{\Phi} = (\Phi_1, \ldots, \Phi_M)$.

$$\begin{aligned} y_1 &= f(\theta_1, \Omega_1) \\ y_2 &= f(\theta_2, \Omega_2) \\ &\vdots \\ y_M &= f(\theta_M, \Omega_M) \end{aligned} \tag{3}$$

The loss L for subtask m can be calculated by root mean squared error.

$$L_m = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_{m_i} - y_i)^2} \tag{4}$$

where y is the observed time series and $\hat{y}_m$ is the prediction given by subtask m, and N is the number of samples.

Cooperative coevolution breaks down a problem into modules that are implemented as sub-population. Through the problem formulation via multi-task learning, it is natural to use sub-populations to optimise the given network modules Φ_m. The sub-populations are given by $S_1, S_2, ..S_M$, where M is number of subtasks. The individuals that make up the sub-populations that consist of respective network module m, $S_m = \Phi_m$, where $\Phi_m = [\omega_m, \upsilon_m]$. The sub-populations $S_{m_{ij}}$ consist of a pool of P individuals that are referenced by i. j is used to reference the elements in the individuals that consist of weights and biases in the network module. The individuals of the sub-population are also referred as genotype while the corresponding network module are referred as the phenotype.

Algorithm 1 gives further details which begins by initialising all the components which include the sub-populations S_m. The sub-populations S_m are initialised with real values in a range $[-\alpha, \alpha]$ drawn from uniform distribution where α defines the range. Once this has been done, the algorithm moves into the evolution phase where each subtask is evolved for a fixed number for generations defined by depth of search, β. The major issues of concern here is the way the phenotype is mapped into genotype where a group of weight matrices given by $\Phi_m = [\omega_m, \upsilon_m]$ that makes up subtask θ_m are converted into vector X_m. Next, the notion of utilizing knowledge from previous subtasks through multi-task learning is executed. In the case if the subtask is a base problem ($m == 1$), then the subtask solution X_m is utilised in a conventional matter where knowledge from other subtasks or modules are not needed. Given that the subtask is not a base problem, the current subtask individual X_m is appended with best individuals from the previous subtasks, therefore, $X_m = [B_1, ..., B_{m-1}, V_m]$ where B is the best individual from previous subtask and V is the current individual that needs to be evaluated. The algorithm then makes subtask prediction $\hat{y}_m = f(\theta_m, \Omega_m)$ and evaluates it though the loss function given in Equation 4. This procedure is executed for every individual in the sub-population and repeated for every sub-population until the termination condition is satisfied. The termination condition can be either the maximum number of function evaluations or a minimum fitness value (loss) from the training or validation dataset. Figure 1 shows an ensemble of the network modules, however, they are part of

Alg. 1 Co-evolutionary multi-task learning

```
Data: Reconstructed state-space vector for the respective subtasks $\Omega_m$.
Result: Prediction error for the respective subtasks $\Omega_m$.
initialisation
for each subtask $\Omega_m$ do
    1. Define different sub-populations $S_m$ using network module $\Phi_m$
    2. Initialise all individuals in sub-population $S_m$
end
while each phase until termination do
    for each sub-population $S_m$ do
        for each generation until depth $\beta$ do
            for each i individual in sub-population $S_{m_{ij}}$ do
                for each j in individual $S_{m_{ij}}$ do
                    Assign individual $V_m = S_{m_{ij}}$
                end
                if $m == 1$ then
                    1. $Z_m = [V_m]$
                    2. Fitness evaluation via Equation 4 by encoding $Z_m$ in the cascaded network
                    module $\theta_m$
                end
                else
                    1. Append to best Individual $B_{m-1}$ of previous sub-population:
                    $Z_m = [V_m, B_{m-1}]$
                    2. Fitness evaluation via Equation 4 by encoding $Z_m$ in the cascaded network
                    module $\theta_m$
                end
            end
            for each i individual in sub-population $S_{m_{ij}}$ do
                * Select and create new offspring via evolutionary operators:
                1. Selection, 2. Crossover, and 3. Mutation
            end
        end
    end
end
for each subtask $\Omega_m$ do
    1. Get best solution $B_m$ from sub-population $S_m$
    2. Load test data
    3. Report prediction performance given by loss $L_m$.
end
```

one cascaded network architecture. An implementation in Matlab has been given online [1].

3 Simulation and Results

This section presents an experimental study that compares the performance of CMTL with conventional evolutionary (single task learning) methods such as cooperative neuro-evolution (CNE) and an evolutionary algorithm (EA) for dynamic time series prediction. Tropical cyclones from South Pacific and South Indian Ocean are considered in order to address the minimal timespan problem as previously identified [9].

[1] https://github.com/rohitash-chandra/CMTL_dynamictimeseries

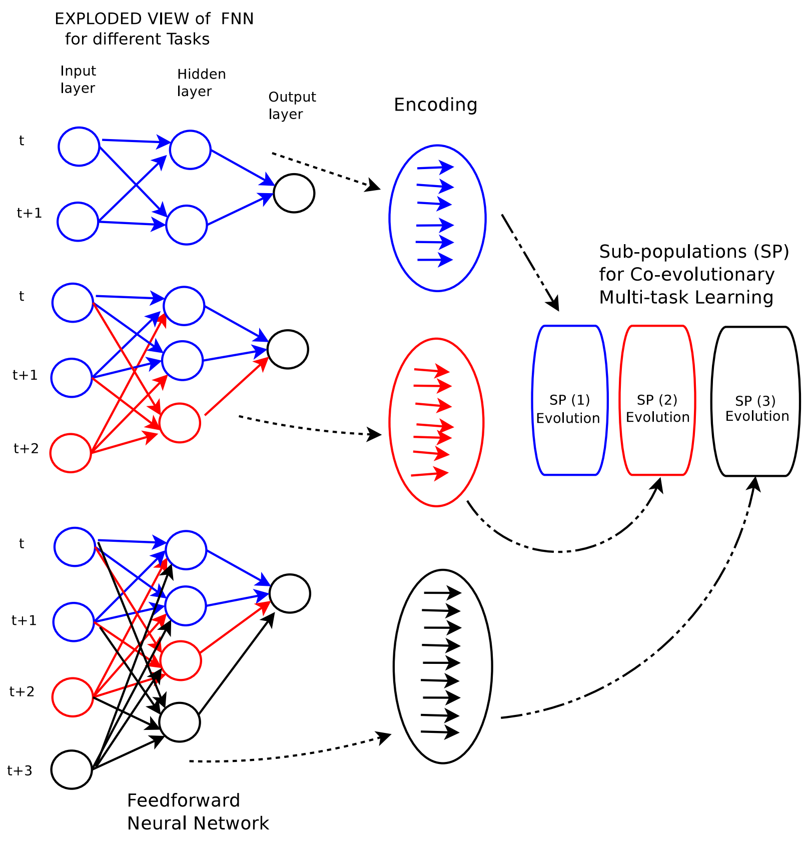
EXPLODED VIEW of FNN for different Tasks
Input layer
Hidden layer
Output layer
Encoding
t
t+1
t+2
t+3
Sub-populations (SP) for Co-evolutionary Multi-task Learning
SP (1) Evolution
SP (2) Evolution
SP (3) Evolution
Feedforward Neural Network

Fig. 1. Subtasks in co-evolutionary multi-task learning. Subtask 1 employs a network topology with 2 hidden neurons while the rest of the subtasks add extra input and hidden neurons.

3.1 Cyclone Time Series

The Southern Hemisphere tropical cyclone best-track data from Joint Typhoon Warning Center recorded every 6-hours is used as the main source of data [14]. The austral summer tropical cyclone season (November to April) from 1985 to 2013 is considered in the current study as data prior to the satellite era is not reliable due to inconsistencies and missing values. The original data of tropical cyclone wind intensity in the South Pacific was divided into training and testing set as follows:

– Training Set: Cyclones from 1985 - 2005 (219 Cyclones with 6837 data points)
– Testing Set: Cyclones from 2006 - 2013 (71 Cyclones with 2600 data points)

In the case for South Indian Ocean, the details are as follows:

– Training Set: Cyclones from 1985 - 2001 (285 Cyclones with 9365 data points)
– Testing Set: Cyclones from 2002 - 2013 (190 Cyclones with 8295 data points)

Note that although the cyclones are separate events, the cyclones are combined in a consecutive order as given by their data of occurrence. The time series is reconstructed with set of values for the embedding dimensions, $D_m = \{3, 5, 7\}$ that refers to the respective subtasks Ω_m. The time lag $T = 2$ is used for the respective subtasks.

3.2 Experimental Design

In the case of cooperative neuro-evolution, neuron level problem decomposition is applied for training feedforward networks [5] on the given problems. Covariance matrix adaptation evolution strategies (CMAES) [12] is used as the evolutionary algorithm in sub-populations of CMTL, CNE and the population of EA. The training and generalisation performances are reported for each case given by the different tasks in the respective time series problems. Note that only CMTL can be used to approach dynamic time series problem with the feature of multi-task learning, however, the results for single task learning methods (CNE and EA) are given in order to provide baseline performance.

The respective neural networks used sigmoid units in the hidden and output layer for all the different problems. The loss given by root mean squared error (RMSE) in Equation 4 is used as the main performance measure. Each neural network architecture was tested with different numbers of hidden neurons. The depth of search, $\beta = 5$ generations is used in the sub-populations of CMTL and CNE. Note that all the sub-populations evolve for the same depth of search. The population size of the CMAES in all the respective methods is given by $P = 4 + floor(3 * log(\gamma))$, where γ is the total number of weights and biases in the cascaded network architecture. The termination condition is fixed at 30 000 function evaluations for each task, hence, CMTL employs 120 000 function evaluations while single task learning methods use 30 000 for each of the respective subtasks for all the problems. Note that since there is a fixed training time, there was no validation set used to stop training.

3.3 Results

The results are presented for the performance of the given methods on the two selected cyclone problems, which focus on South Pacific and South Indian ocean as shown in Figure 3 and Figure 2, respectively. In the case of the South Pacific ocean, the results show that CMTL provides the best generalisation performance when compared to CNE and EA. This is also observed for the South Indian ocean.

3.4 Discussion

The goal of the experiments was to evaluate if the proposed algorithm can deliver similar results when compared to single task learning methods for the dynamic time series problems. Therefore, the comparison was to ensure that the approach does not lose quality in terms of generalisation performance when compared to single-task learning methods. The results have shown that CMTL not only addresses dynamic time series, but also improves the performance when compared

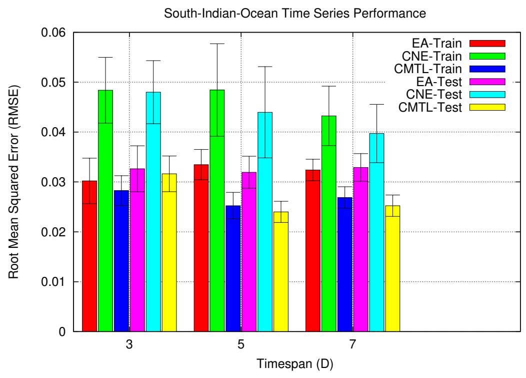

Fig. 2. Performance given by EA, CNE, CMTL for South Indian Ocean

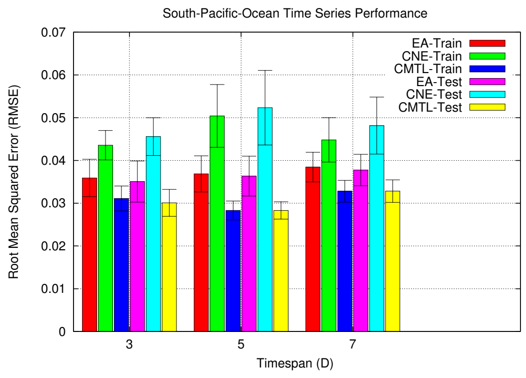

Fig. 3. Performance given by EA, CNE, CMTL for South Pacific Ocean

to single-task learning. This could be seen as a form of developmental learning [20,17] which enables modularity for dynamic time series prediction as demonstrated through the case of cyclones. The feature of modularity was implemented through the network modules that are combined as the nature or complexity of the problem increases. Modularity helps in making a prediction given knowledge from base network module in cases if the information in the other subtasks are not available. Modularity is important for the design of neural networks tailored for hardware implementation [16]. The disruptions in certain synapse(s) can result in problems which can be eliminated by persevering knowledge as modules [8].

Being evolutionary in nature, CMTL can be seen as a flexible method that can be used for multiple sets of data that have different features of which some are overlapping and distinctly contribute to the problem. The common features can be captured as a subtask. Through multi-task learning, the overlapping features can be used as building blocks to learn the nature of the problem through the model at hand. Although feedforward neural networks have been used in CMTL, other neural network architectures, and learning models can be used depending on the nature of the problem.

4 Conclusions and Future Work

A novel algorithm has been presented that provides a synergy between neuro-evolution and multi-task learning for dynamic time series problems. Co-evolutionary multi-task learning is needed for tropical cyclones where robust prediction needs to be given as soon as the event takes place. In the cyclones studied, each point in a given timespan represented six hours. Therefore, the proposed algorithm implemented a model that can work even when a single point is implemented as a subtask. As more points of data are given to the subtask, more predictions can be made which makes the model dynamic and robust. The results show that the proposed algorithm not only addressed dynamic time series, but also improved prediction performance when compared to conventional methods.

In future work, the proposed algorithm can be used for other time series problems that can be broken into multiple subtasks. It can also be extended to transfer learning problems that can include both heterogeneous and homogeneous domain adaptation cases. In case of tropical cyclones, which is a multivariate problem, the different subtasks can be redefined with information such as cyclone tracks, seas surface temperature, and humidity.

References

1. Ali, M.M., Jagadeesh, P.S.V., Lin, I.I., Hsu, J.Y.: A neural network approach to estimate tropical cyclone heat potential in the Indian ocean. IEEE Geoscience and Remote Sensing Letters 9(6), 1114–1117 (2012)
2. Angeline, P., Saunders, G., Pollack, J.: An evolutionary algorithm that constructs recurrent neural networks. Neural Networks, IEEE Transactions on 5(1), 54 –65 (1994)
3. Caruana, R.: Multitask learning. Machine Learning 28(1), 41–75 (Jul 1997)
4. Chandra, R.: Competition and collaboration in cooperative coevolution of Elman recurrent neural networks for time-series prediction. Neural Networks and Learning Systems, IEEE Transactions on 26, 3123–3136 (2015)
5. Chandra, R., Frean, M., Zhang, M.: On the issue of separability for problem decomposition in cooperative neuro-evolution. Neurocomputing 87, 33–40 (2012)
6. Chandra, R., Gupta, A., Ong, Y., Goh, C.K.: Evolutionary multi-task learning for modular training of feedforward neural networks. In: Neural Information Processing - 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16-21, 2016, Proceedings, Part II. pp. 37–46 (2016)
7. Chandra, R., Zhang, M.: Cooperative coevolution of Elman recurrent neural networks for chaotic time series prediction. Neurocomputing 186, 116 – 123 (2012)
8. Clune, J., Mouret, J.B., Lipson, H.: The evolutionary origins of modularity. Proceedings of the Royal Society of London B: Biological Sciences 280(1755) (2013)
9. Deo, R., Chandra, R.: Identification of minimal timespan problem for recurrent neural networks with application to cyclone wind-intensity prediction. In: International Joint Conference on Neural Networks (IJCNN). pp. 489–496 (July 2016)
10. Evgeniou, T., Micchelli, C.A., Pontil, M.: Learning multiple tasks with kernel methods. Journal of Machine Learning Research 6(Apr), 615–637 (2005)

11. Frazier, C., Kockelman, K.: Chaos theory and transportation systems: Instructive example. Transportation Research Record: Journal of the Transportation Research Board 20, 9–17 (2004)
12. Hansen, N., Müller, S.D., Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evolutionary Computation 11(1), 1–18 (2003)
13. Happel, B.L., Murre, J.M.: Design and evolution of modular neural network architectures. Neural Networks 7(6-7), 985 – 1004 (1994)
14. JTWC: Joint Typhoon Warning Center - tropical cyclone best track data site. `http://www.usno.navy.mil/NOOC/nmfc-ph/RSS/jtwc/` (2017), online; last accessed 16 August 2017
15. Knaff, J.A., DeMaria, M., Sampson, C.R., Gross, J.M.: Statistical, five-day tropical cyclone intensity forecasts derived from climatology and persistence. Weather Forecasting 18, 80–92 (2003)
16. Misra, J., Saha, I.: Artificial neural networks in hardware: A survey of two decades of progress. Neurocomputing 74(13), 239 – 255 (2010), artificial Brains
17. Morse, A.F., De Greeff, J., Belpeame, T., Cangelosi, A.: Epigenetic robotics architecture (era). IEEE Transactions on Autonomous Mental Development 2(4), 325–339 (2010)
18. Potter, M.A., De Jong, K.A.: Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evol. Comput. 8(1), 1–29 (2000)
19. Potter, M., De Jong, K.: A cooperative coevolutionary approach to function optimization. In: Davidor, Y., Schwefel, H.P., Mnner, R. (eds.) Parallel Problem Solving from Nature PPSN III, Lecture Notes in Computer Science, vol. 866, pp. 249–257. Springer Berlin Heidelberg (1994)
20. Prince, C., Helder, N., Hollich, G.: Ongoing emergence: A core concept in epigenetic robotics. In: Proceedings of the Fifth International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems. pp. 63–70. Lund University Cognitive Studies (2005)
21. Stiles, B.W., Danielson, R.E., Poulsen, W.L., Brennan, M.J., Hristova-Veleva, S., Shen, T.P., Fore, A.G.: Optimized tropical cyclone winds from quikscat: A neural network approach. IEEE Transactions on Geoscience and Remote Sensing 52(11), 7418–7434 (Nov 2014)
22. Takens, F.: Detecting strange attractors in turbulence. In: Dynamical Systems and Turbulence, Warwick 1980, pp. 366–381. Lecture Notes in Mathematics (1981)
23. Zeng, T., Ji, S.: Deep convolutional neural networks for multi-instance multi-task learning. In: Data Mining (ICDM), 2015 IEEE International Conference on. pp. 579–588 (Nov 2015)
24. Zheng, H., Geng, X., Tao, D., Jin, Z.: A multi-task model for simultaneous face identification and facial expression recognition. Neurocomputing 171, 515 – 523 (2016)
25. Zjavka, L.: Numerical weather prediction revisions using the locally trained differential polynomial network. Expert Systems with Applications 44, 265 – 274 (2016)