Bayesian Multi-task Learning for Dynamic Time Series Prediction

Rohitash Chandra *† and Sally Cripps *‡
* Centre for Translational Data Science, The University of Sydney, NSW 2006, Australia
† School of Geosciences, The University of Sydney, NSW 2006, Australia
‡ School of Mathematics and Statistics, The University of Sydney, NSW 2006, Australia

***Abstract*—Time series prediction typically consists of a data reconstruction phase where the time series is broken into overlapping windows. The size of the window could vary for different types of problems for optimal performance. Dynamic time series prediction refers to "on the fly" robust prediction given partial information where prediction can be made regardless of the window size. Multi-task learning features learning from related tasks through shared representation knowledge which has shown to be useful for dynamic time series prediction. This features uncertainty that can be addressed through synergy of Bayesian inference and multi-task learning. In this paper, we present a Bayesian approach to multi-task learning for dynamic time series prediction. The method provides uncertainty quantification given posterior distribution of weights and biases in a cascaded multi-task network architecture. The results show that the proposed method is able to provide competing prediction performance to the literature, featuring uncertainty quantification in prediction.**

***Index Terms*—Bayesian neural networks, multi-task learning, time series prediction, dynamic time series prediction**

I. Introduction

Dynamic time series prediction refers to the ability of a model to given a prediction given a set of input values that about past behaviour of the time series [1], [2]. The past behaviour of the time series is captured by overlapping windows which is also known as embedding dimension in the time series literature [3]. The size of the window is determined experimentally which can affect the prediction performance. Dynamic time series prediction essentially refers to a model that can give robust prediction given a set of unique values for the window, rather than one value. Recently, concepts from multi-task learning has been used to address dynamic time series prediction [2], [1]

Multi-task learning features learning from related tasks through shared representation knowledge for improved generalisation and efficient training performance [4], [5]. Multi-task learning has been applied pattern classification and computer vision applications [6], [7]. Neuroevolution considers the use of evolutionary algorithms for training neural networks [8], [9] for applications that range from control, prediction, and classification. Co-evolutionary multi-task learning (CMTL) is a neuroevolution method for dynamic time series prediction [1]. CMTL trains a cascaded multi-task network where a cascade is defined by different number of input features. A variation of CMTL with predictive recurrence network has also been used for multi-step time series prediction [10]. CMTL has also been used for pattern classification problems [11] where it had the feature of determining the contribution of different feature groups during learning. Although, the CMTL has given very promising performance for the different applications, there are challenges in timely convergence when considering large datasets due to the evolutionary operators. This can be addressed via gradient descent based learning algorithms. Another limitation of CMTL is that it provides one point estimates in prediction which lacks uncertainty quantification. This can be addressed through Bayesian methods which is the focus of this paper.

Bayesian learning methods can infer parameters or weights of a neural network by marginalizing them out of the posterior distribution [12], [13]. Bayesian neural networks use Markov Chain Monti-Carlo (MCMC) methods for inference of network parameters such as the weights and biases. Note that learning or optimisation typically refers to the strategy of finding the best set of weights in a neural network that best describes the training data. Inference refers to finding the probability distribution of weights via MCMC methods that properly describes the training data while featuring methodology for uncertainty quantification. Examples of MCMC methods include Hamiltonian Monte Carlo [14], expectation propagation [15], and variational inference [16]. These approaches have limitations when used for neural networks given scalability in both network architecture and data size. A number of attempts have been proposed combining MCMC techniques with the gradient optimization algorithms [17], [18], [19]. More recently, Chandra et. al presented a Langevin dynamics approach to speed up MCMC method via gradient descent for chaotic time series prediction [20]. The method was able to perform similar to standalone backpropagation in terms of prediction while featuring uncertainty quantification through the featured Bayesian methodology. Hence, uncertainty quantification in dynamic time series prediction can be addressed through Bayesian inference. The challenge would be in adapting canonical MCMC algorithm for featuring shared knowledge representation between the subtasks defined by the dynamic time series.

In this paper, we present a Bayesian neural learning approach for dynamic time series prediction. The method provides uncertainty quantification through posterior distribution of weights and biases in a cascaded multi-task network. This extends CMTL via MCMC that enables shared knowledge representation from the subtasks. The method is applied to

chaotic time series prediction problems from the literature and compared to results from CMTL.

The rest of the paper is organised as follows. Section 2 presents related work, while Section 3 presents the Bayesian multitask learning method for dynamic time series prediction. Section 4 presents the results with a discussion. Section 5 presents the conclusions and directions for future research.

II. Background and related work

Multi-task learning [4] has been applied for different types problems that include supervised and unsupervised learning [21], [22], [23], [24]. 'Negative transfer' is a concept from behavioural psychology that refers to the interference of the previous knowledge with new learning [25]. Negative transfer has been a challenge for multi-task learning that has been addressed through a number of approaches. One of them has been through the grouping of tasks where knowledge transfer is performed only within each group [26], [27]. Bakker et. al presented a Bayesian approach in which some of the model parameters were shared and others loosely connected through a joint prior distribution and the degree of similarity between tasks is inferred via the posterior distribution of these parameters [27]. Zhang and Yeung presented a convex formulation for multi-task metric learning by modelling the task relationships in the form of a covariance matrix [26]. Furthermore, Zhong *et al.* presented flexible multi-task learning framework to identify latent grouping structures in order to restrict negative knowledge transfer [28].

Multi-task learning has also inspired optimisation methods such as evolutionary algorithms where transfer optimisation exploits solutions from related problems [29]. This motivated the development of evolutionary multi-tasking for training feedforward neural networks for modular knowledge representation [30]. The different subtasks were implemented as different topologies given by the number of hidden neurons with the goal to store knowledge in modules so that perturbation of certain modules do not affect the decision making process as a whole.

In the case of time series, Xu *et al.* applied multi-task regression for forecasting problem [31] which provided a synergy of ensemble forecasting [32] with multi-task regression to estimate the optimal weights for combining the ensembles. Multi-task learning has been also used for multi-step-ahead prediction [10]. The method considers the prediction horizon as subtasks that have shared knowledge representation which has achieved very promising performance when compared to the literature.

Markov chain Monte Carlo (MCMC) methods are mainly used for calculating numerical approximations of multi-dimensional integrals for models used in Bayesian statistics. Recent developments show that via MCMC methods, it is possible to compute large hierarchical models that feature hundreds or even thousands of unknown parameters [33]. A number of attempts have been proposed combining MCMC techniques with the gradient optimization algorithms [17], the simulated annealing method [18], and evolutionary algorithms [19]. Considering time series prediction, Liang et. al presented an MCMC algorithm for neural networks for selected time series problems [34]. Bayesian techniques have also been used for controlling model complexity and selecting inputs in neural networks for the short term time series forecasting [35]. Moreover, recursive Bayesian recurrent neural networks [36] and evolutionary Baysiean neural networks have been used for time series forecasting [19].

III. Bayesian multi-task learning for dynamic time series

A. Preliminaries

Let y_t denote a univariate time series modelled by:

$$y_t = f(\mathbf{x}_t) + \epsilon_t, \text{ for } t = 1, 2, \ldots, T \tag{1}$$

where $f(\mathbf{x}_t) = E(y_t|\mathbf{x}_t)$, is an unknown function, $\mathbf{x}_t = (y_{t-1}, \ldots, y_{t-D})$ is a vector of lagged values of y_t, and ϵ_t is the noise with $\epsilon_t \sim \mathcal{N}(0, \tau^2)\ \forall t$.

In order to use neural networks for time series prediction, the original dataset is constructed into a state-space vector through Taken's theorem [3] which is governed by the embedding dimension (D) and time-lag (B).

We define

$$\mathcal{A}_{D,B} = \{t; t > D, \quad \text{mod } (t - (D+1), B) = 0\} \tag{2}$$

to be the set of indices for which predictions will be made, so that $\mathcal{A}_{D,B} = (a_1, \ldots, a_P)$, where $a_1 = D + 1$ and P is the total number of predictions made. The corresponding lagged values to be used as inputs are contained in $\mathbf{x}_t$, for $t \in \mathcal{A}_{D,B}$. We compute the prediction, $f(\mathbf{x}_t)$, by a feedforward neural network with one hidden layer defined by the function

$$f(\mathbf{x}_t) = g\left(\delta_o + \sum_{h=1}^{H} v_j g\left(\delta_h + \sum_{d=1}^{D} w_{dh} y_{t-d}\right)\right) \tag{3}$$

where H is the number of neurons in the hidden layer, and δ_o and δ_h are the biases for the output o and hidden h layer, respectively, for $h = 1, \ldots, H$. V_j is the weight which maps the hidden layer h to the output layer. w_{dh} is the weight which maps y_{t-d} to the hidden layer h, and g is the activation function, which we assume to be a sigmoid function for the hidden and output layer units.

B. Multi-task learning for dynamic time series

We wish to allow the embedding dimenison to vary, therefore we define

$$\Omega_M = [D_1, D_2, ...D_M] \tag{4}$$

to be a vector of integers with elements which represent a possible value of embedding dimension, D_m for $m = 1, \ldots, M$ and $D_1, < D_2, < \ldots, < D_M$. We now extend our

definitions of $\mathcal{A}$ and $\mathbf{x_t}$ to reflect the varying embedding dimension. Let

$$\mathcal{A}_{D,B} = \{t; t > D_M, \mod (t - (D_M + 1), B) = 0\} \quad (5)$$

so that the set of indices for which predictions will be made depends upon only the largest embedding dimension, D_M. Let

$$\mathbf{x}_{t,m} = (y_{t-D_m}, \ldots, y_{t-1})$$

be the set of input features used to predict $y_{t \in \mathcal{A}_{D_M,B}}$. Each value, D_m correspond to a subtasks. The design matrix X_m for each subtask is a P_M by D_m matrix, where P_M is the number of predictions made and is fixed for all $m = 1, \ldots, M$. Then we have

$$X_M = \begin{bmatrix} y_1 & \cdots & y_{D_M-1} & y_{D_M} \\ y_{1+B} & \cdots & y_{D_M+B-1} & y_{D_M+B} \\ \vdots & \ddots & \vdots & \vdots \\ y_{a_P-D_M} & \cdots & y_{a_P-2} & y_{a_P-1} \end{bmatrix}$$

$$X_{m<M} = \begin{bmatrix} y_{D_M-D_m+1} & \cdots & y_{D_M-1} & y_{D_M} \\ y_{D_M-D_m+1+B} & \cdots & y_{D_M+B-1} & y_{D_M+B} \\ \vdots & \ddots & \vdots & \vdots \\ y_{a_P-D_m} & \cdots & y_{a_P-D_m+1} & y_{a_P-1} \end{bmatrix}$$

where

$$a_P = \arg\max_{t \leq T} \mod (t - (D_M + 1), B) = 0.$$

For example, if $T = 100$, $B = 2$, $m = 1, 2$, $D_1 = 3$ and $D_2 = 4$, then $\mathcal{A}_{D_M,B} = (5, 7, 9, \ldots, 99)$

$$X_1 = \begin{bmatrix} y_2 & y_3 & y_4 \\ y_4 & y_5 & y_6 \\ \vdots & \vdots & \vdots \\ y_{96} & y_{97} & y_{98} \end{bmatrix}$$

while

$$X_2 = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 \\ y_3 & y_4 & y_5 & y_6 \\ \vdots & \vdots & \vdots & \vdots \\ y_{95} & y_{96} & y_{97} & y_{98} \end{bmatrix}$$

The task of prediction using different sets of input features is different from the conventional multi-task learning setting, where the output of the network defines the task. In our case, the task is defined by the input of the network, X_m, keeping the output of the network fixed.

For each set of input features X_m, let $\tilde{\mathbf{w}}_\mathbf{m}$ be the $D_m H \times 1$ vector of weights which map the D_m inputs to the H hidden neurons, with $\tilde{\mathbf{w}}_\mathbf{m} = (\mathbf{w_1}.', \ldots, \mathbf{w}'_{D_m.})'$, and $\mathbf{w_d}. = (w_{d1}, \ldots, w_{dH})'$, for $d = 1, \ldots, D_m$. Let $\boldsymbol{\upsilon}_m$ be the $H \times 1$ vector of weights which map the hidden neurons to the output, let $\boldsymbol{\delta}_{mh}$ and $\boldsymbol{\delta}_{mo}$, be the vector of bias terms in the hidden and output layers respectively. The parameters needed to evaluate the likelihood, given input features X_m, are denoted by $\boldsymbol{\theta}_m$, for $m = 1, \ldots, M$ are constructed as in our previous work [20]. Let $\Phi_1 = \boldsymbol{\theta}_1 = [\tilde{\mathbf{w}}_\mathbf{1}, \boldsymbol{\upsilon}_1, \eta = \log(\tau^2), \boldsymbol{\delta}_{1o}, \boldsymbol{\delta}_{1h}]$ be the set of parameters corresponding to $m = 1$, then parameters of the subsequent sub tasks are constructed as follows;

$$\begin{aligned} \Phi_1 &= [\tilde{\mathbf{w}}_\mathbf{1}, \boldsymbol{\upsilon}_1, \eta, \boldsymbol{\delta}_{1o}, \boldsymbol{\delta}_{1h}]; \;\; \boldsymbol{\theta}_1 = (\Phi_1) \\ \Phi_2 &= [\tilde{\mathbf{w}}_\mathbf{2}, \boldsymbol{\upsilon}_2, \eta, \boldsymbol{\delta}_{2o}, \boldsymbol{\delta}_{2h}]; \;\; \theta_2 = [\boldsymbol{\theta}_1, \Phi_2] \\ &\vdots \\ \Phi_M &= [\tilde{\mathbf{w}}_\mathbf{M}, \boldsymbol{\upsilon}_M, \eta, \boldsymbol{\delta}_{Mo}, \boldsymbol{\delta}_{Mh}]; \;\; \boldsymbol{\theta}_M = [\theta_{M-1}, \Phi_M] \end{aligned}$$

Note that the log of th variance of the error term η is the same for all subtasks. The subtasks considered for optimisation via modular back-propagation is therefore $\boldsymbol{\Phi} = (\Phi_1, \ldots, \Phi_M)$.

Let $\hat{y}_{mt}$ denote a predicted value of y at time t based on a set of input features given in X_m

$$\begin{aligned} \hat{y}_{1,t} &= f(\theta_1; X_1) \\ \hat{y}_{2,t} &= f(\theta_2; X_2) \\ &\vdots \\ \hat{y}_{M,t} &= f(\theta_M; X_M) \end{aligned} \quad (6)$$

The loss for P_M predictions made using input features in X_m, can be calculated by root mean squared error.

$$rmse_m = \sqrt{\frac{1}{P_M} \sum_{t \in \mathcal{A}_{D_M,L}} (\hat{y}_{m,t} - y_t)^2} \quad (7)$$

C. Bayesian multi-task learning

We take a Bayesian approach and make inference regarding the weights in the network, θ, via the posterior distribution, $p(\theta|\mathbf{y})$. We sample from this distribution using MCMC with a random walk proposal to move the chain around the parameter space and accept draws from this proposal distribution using the usual Metropolis Hastings acceptance probability.

1) Priors : We need to specify prior distributions for the elements of θ which we choose to be

$$\begin{aligned} \upsilon_h &\sim \mathcal{N}(0, \sigma_\upsilon^2) \text{ for } h = 1, \ldots, H. \\ \delta_o &\sim N(0, \sigma_\delta^2) \\ \delta_h &\sim N(0, \sigma_\delta^2) \\ w_{dh} &\sim \mathcal{N}(0, \sigma^2) \text{ for } h = 1, \ldots, H \text{ and } d = 1, \ldots, D_m, \\ \tau^2 &\sim \mathcal{IG}(\nu_1, \nu_2) \end{aligned} \quad (8)$$

The log likelihood for subtask m, $l_m(\boldsymbol{\theta_m})$, is

$$l_m(\boldsymbol{\theta_m}) = -\frac{P_M - 1}{2} \log(\tau^2) - \frac{1}{2\tau^2} \sum_{t \in \mathcal{A}_{\mathcal{D}_\mathcal{M},\mathcal{T}}} (y_t - \hat{y}_{m,t})^2 \quad (9)$$

We follow our previous work [20] and assume that the elements of $\theta_\mathbf{m}$ are independent *apriori*.

2) Algorithm: A number of issues need to be considered for implementation of our MCMC sampler. Algorithm 1 begins by the construction of the input features, X_m, corresponding to subtask m, for $m = 1, \ldots, M$. The goal of Algorithm 1 is to obtain a distribution of weights and biases via sampling through MCMC method that features transfer of network weights and biases given by Φm from one subtask to another. Given the subtask knowledge $\theta_1 = [\Phi_1]$ and $\theta_2 = [\theta_1, \Phi_2]$ that consists of weights and biases, the parameters in Φ_m, for $m = 1, \ldots, M$ are initialized. Afterwards, the sampling begins where for each subtask m, the $k + 1^{th}$ position in the chain, $\Phi_m^{[k+1]}$, is sampled by proposing new values of $\Phi[p]_m$ from normal distribution centred at the current values, $\Phi_m^{[k]}$ with a small standard deviation, denoted by σ_λ (also known as the step size) such that $\Phi_m^{[p]} = \Phi_m^{[k]} + \lambda$, where $\lambda \sim N(0, \sigma_\lambda)$. This is given by Step 2 in the Algorithm. The metropolis acceptance probability is computed if the proposed value is accepted $\Phi_m^{[k+1]} = \Phi_m^{[p]}$, otherwise $\Phi_m^{[k+1]} = \Phi_m^{[k]}$. The new value $\Phi_m^{[k+1]}$ is transferred to $[\theta_m^{[k+1]} = [\theta_{m-1}, \Phi_m^{[k+1]}]$. The procedure is repeated for the rest of the subtasks until the termination condition is satisfied. Note that there is one way transfer of knowledge, hence, there is no transfer of knowledge from the final subtask to the foundation subtask.

Data: Reconstructed univariate time series $\mathbf{y}$
Result: Posterior of weights and biases $p(\boldsymbol{\theta}|\mathbf{y})$
1: Reconstructed state-space vector for sub-task Ψ_s
2: Network for each sub-task $y_s = f(\theta_s, \Psi_s)$
4: Set parameters σ^2 ,ν_1, ν_2 for priors
Initialisation:
for *each sub-task* m **do**
  Propose initial weights and biases for the module: $\boldsymbol{\theta}_m^{[p]} = \mathcal{N}(0, \Sigma_\eta)$
end
Sampling:
for *each* k *until max-samples* **do**
  for *each sub-task* m **do**
    Step 1. Draw $\boldsymbol{\eta}$ from $\mathcal{N}(0, \Sigma_\eta)$
    Step 2. Propose set of weights and biases for the cascaded module given by the subtask: $\boldsymbol{\theta}_m^{[p]} = \Phi_m^{[k]} + \boldsymbol{\eta}$
    Step 3. Calculate posterior for subtask, Step 4. Draw from uniform distribution $u \sim \mathcal{U}[0, 1]$
    Step 5. Obtain acceptance probability α
    Step 6. Set $\Phi_m^{[k+1]} = \Phi_m^{[p]}$, if $u < \alpha$, otherwise $\Phi_m^{[k+1]} = \Phi_m^{[k]}$.
    Step 7. Transfer knowledge to subtask: $[\theta_m^{[k+1]} = [\theta_{m-1}, \Phi_m^{[k+1]}]$.
  end
end

Algorithm 1: Bayesian multi-task learning via MCMC sampler

The method used knowledge transfer between subtasks given algorithm presented in co-evolutionary multi-task learning [1]. The implementation of Algorithm 1 in Python is also given online [1].

[1]https://github.com/rohitash-chandra/mt-bnn-dts

IV. Experiments and Results

A. Benchmark Chaotic Time Series Problems

The Mackey-Glass, Lorenz, Henon and Rossler are the four simulated time series problems. The experiments use the chaotic time series with length of 1000 generated by the respective chaotic attractor.

The Sunspot and Lazer datasets are from real work problems that feature noise. The first 500 data points are reconstructed and used for training and the remaining 500 is used for testing. The ACI finance dataset is taken from NASDAQ stock exchange. All datasets are scaled in the range [0,1]. Further details of each of the time series problem is given in [1]. In all cases, the phase space of the original time series is reconstructed with the embedding dimensions for 3 datasets for the respective subtasks with embedding dimension $\Omega = s_1, s_2, s_3$ and time lag $T = 2$. We run two major experiments that are defined by the way the subtasks are created through different window size in the time series. In experiment 1, the subtasks are created by windows of size [2, 3, 4]. In experiment 2, the subtasks are created by windows of size [3, 5, 7].

B. Results

The results for the performance of the BMTL on the seven benchmark time series datasets are given in Table I. The performance in terms of mean and standard deviation of the root-mean-squared-error of the predictions based on posterior weights and the biases are also given. Note that 5 % of the samples in the initial phase were discarded as "burn-in" which is a standard procedure of MCMC methods.

We notice that the algorithm attempts to give predictions that are similar across the different subtasks. Given the test performance, for certain problems, the prediction performance deteriorated with increasing subtasks (Mackey, Lorenz, and Rossler). While in others, the prediction performance improves for increasing subtasks. We note the the problems where the performance improved were mostly the real-world problems that featured noise.

The prediction performance is better for the simulated problems (Lorenz, Mackey, Rossler) when compared to the real-world benchmark datasets (Sunspot, Lazer, ACI finance). The algorithm had difficulty in the Henon problem which is a simulated problem that contains less noise, however highly chaotic in nature.

The results in Figure 1 and Figure 2 show a visualisation of prediction performance for the test dataset given the respective subtasks for three selected problems. The confidence interval is highlighted that provides uncertainty quantification through the posterior distribution of weights and biases. With reference to the RMSE results given in Table I, we notice that there is a certain trend in the change of the standard deviation from Subtasks 1 - Subtask 3. We notice that the standard deviation for most of the problems is closer to zero for Subtask 3 in both training and test cases. This shows that the uncertainty is reduced due to the transfer of knowledge from previous subtasks.

TABLE I: Results for the respective methods

Problem		Subtask	Train (mean)	Train (std)	Test (mean)	Test(std)	% Accept
Lazer	BMTL	1	0.0686	0.0098	0.0477	0.0043	16.40
		2	0.0621	0.0080	0.0490	0.0040	
		3	0.0647	0.0071	0.0453	0.0088	
	CMTL[1]	1	0.0644	0.0059	0.0766	0.0060	
		2	0.0666	0.0060	0.0875	0.0061	
		3	0.0926	0.0087	0.1164	0.0107	
Sunspot	BMTL	1	0.0501	0.0006	0.0495	0.0013	8.77
		2	0.0527	0.0081	0.0502	0.0017	
		3	0.0405	0.0001	0.0394	0.0001	
	CMTL[1]	1	0.0475	0.0068	0.04121	0.0067	
		2	0.0428	0.0044	0.0366	0.0040	
		3	0.0485	0.0062	0.0420	0.0050	
Mackey	BMTL	1	0.0296	0.0015	0.0297	0.0014	6.04
		2	0.0267	0.0016	0.0267	0.0016	
		3	0.0417	0.0001	0.0418	0.0001	
	CMTL[1]	1	0.0385	0.0050	0.0389	0.0047	
		2	0.0467	0.0048	0.0476	0.0051	
		3	0.0465	0.0047	0.0549	0.0061	
Lorenz	BMTL	1	0.0169	0.0009	0.0155	0.0007	3.90
		2	0.0222	0.0001	0.0207	0.0001	
		3	0.0447	0.0000	0.0383	0.0001	
	CMTL[1]	1	0.0353	0.0057	0.0353	0.0057	
		2	0.0338	0.0048	0.0338	0.0048	
		3	0.0317	0.0039	0.0364	0.0040	
Rossler	BMTL	1	0.0304	0.0023	0.0287	0.0021	6.17
		2	0.0292	0.0030	0.0276	0.0027	
		3	0.0320	0.0001	0.0305	0.0001	
	CMTL[1]	1	0.0454	0.0052	0.0475	0.0054	
		2	0.0448	0.0043	0.0464	0.0049	
		3	0.0498	0.0052	0.0528	0.0058	
Henon	BMTL	1	0.1743	0.0014	0.1756	0.0032	49.55
		2	0.1639	0.0029	0.1624	0.0027	
		3	0.1648	0.0009	0.1644	0.0011	
	CMTL[1]	1	0.0912	0.0137	0.0911	0.0131	
		2	0.1248	0.0126	0.1276	0.0127	
		3	0.1558	0.0121	0.1612	0.0121	
ACI-finance	BMTL	1	0.0447	0.0006	0.0234	0.0009	11.57
		2	0.0387	0.0003	0.0194	0.0002	
		3	0.0332	0.0002	0.0137	0.0003	
	CMTL[1]	1	0.0440	0.0060	0.0416	0.0106	
		2	0.0430	0.0041	0.0398	0.0075	
		3	0.0520	0.0050	0.0597	0.0080	

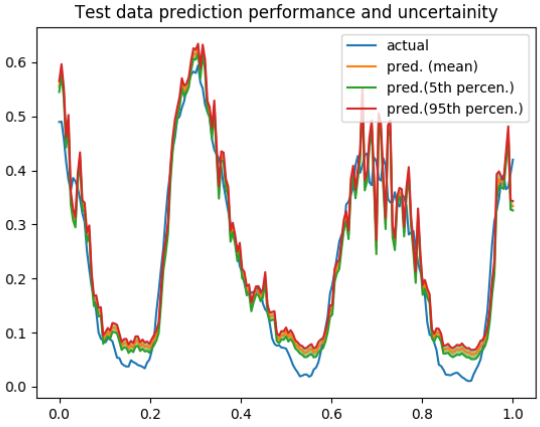
The results further compares the performance with co-evolutionary multi-task learning (CMTL) which was initially proposed for dynamic time series prediction [1]. In Table I, we observe that except for the Henon problem, the proposed Bayesian approach has been able to achieve similar or better results when compared to CMTL for the different subtasts. In the case of Henon problem, BMTL could not give competitive performance for Subtask 1 and 2 which could be due to local convergence through the random-walk sampler.

We note that in the CMTL paper [1], multiple experiments were conducted with different initial positions in weight space. In the case of MBTL, only results from 1 experimental run is shown for each problem. Due to feature of Bayesian inference, the methodology does not require multiple experimentation. A major limitation of the experiment results was when there was a local convergence in some of the subtasks for selected problems. The Mackey-Glass problem achieved local convergence in Subtask 3 as shown in Figure 3. The local convergence was due to poor initial position. The experiment was executed again for this case.

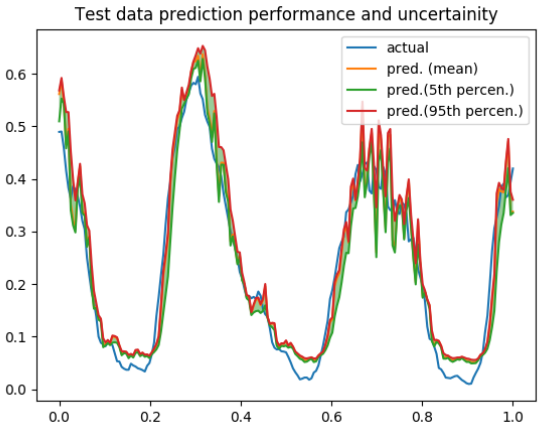
V. Discussion

In the literature on transfer learning and multi-task learning, there is not much work done that explores transfer of knowledge in a Bayesian context that employs MCMC methods. This paper addressed it through novel MCMC method that incorporates transfer of knowledge from certain subtasks.

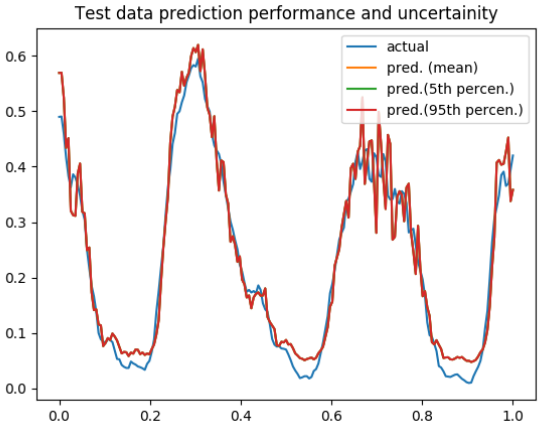
The proposed Bayesian method required only 1 experiment per problem as it features uncertainty quantification via distri-



(a) Subtask 1

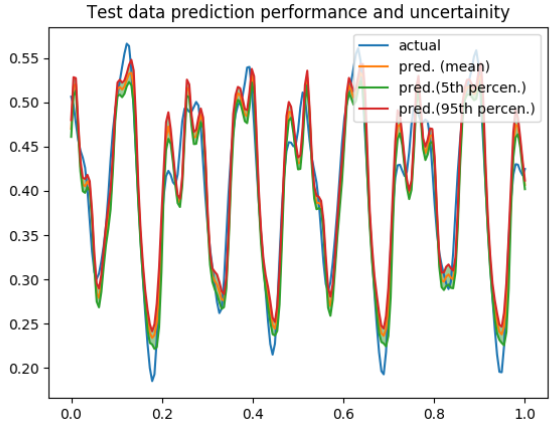


(b) Subtask 2

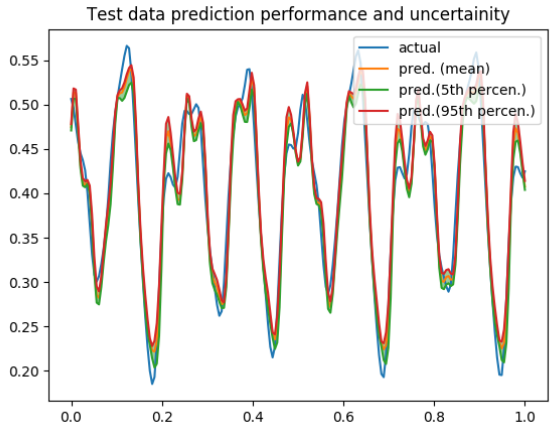


(c) Subtask 3

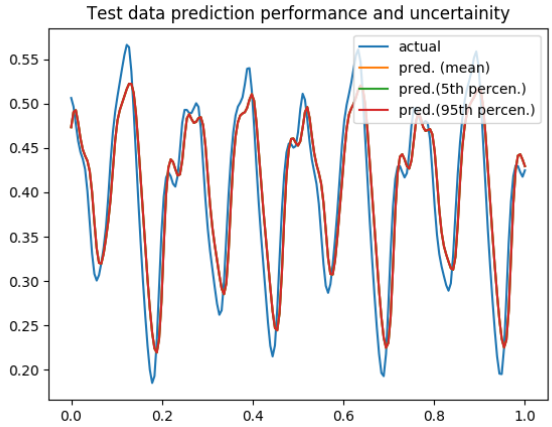
Fig. 1: Sunspot performance evaluation across different subtasks (1, 2, and 3)



(a) Subtask 1



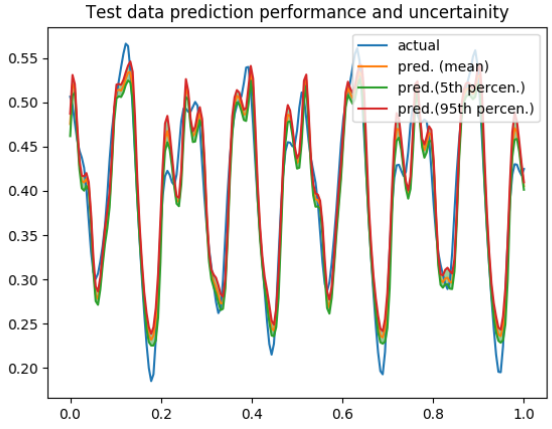
(b) Subtask 2



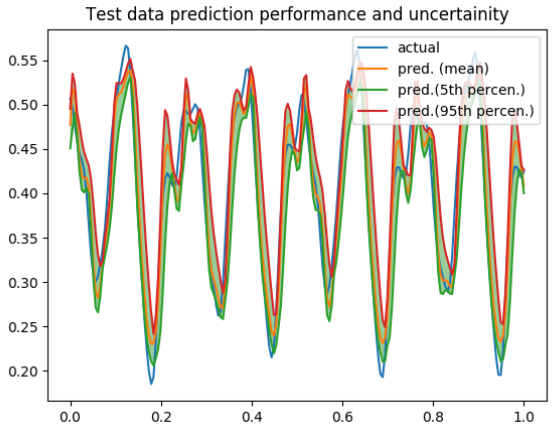
(c) Subtask 3

Fig. 2: Mackey-Glass performance evaluation across different subtasks (1, 2, and 3)

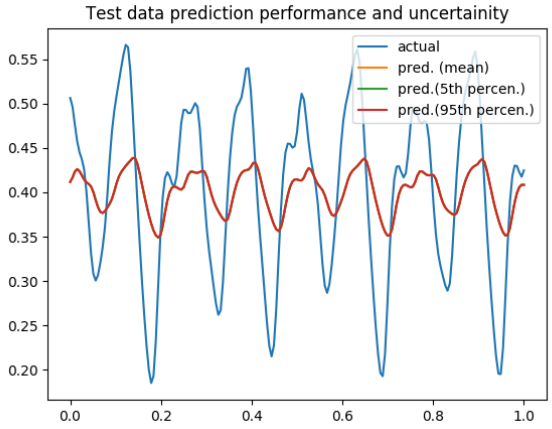
bution of weights and biases in the network that was attained through MCMC sampling. In case of CMTL which is an



(a) Subtask 1



(b) Subtask 2



(c) Subtask 3

Fig. 3: Local convergence in subtask 3 for Mackey-Glass problem

optimisation strategy, only one point estimate of weights is achieved. Therefore, for CMTL, multiple experiments were conducted to report mean and standard deviation which is considered a frequentist approach [37]. With the proposed Bayesian methodology, we achieve prediction performance that compares well with the frequentist approach while at the same time saves computation time of multiple experiments.

Another major observation is the effect of the prediction performance as the subtasks increases. Note that increasing subtasks essentially implied that the input features increase and knowledge from previous subtask are utilised through knowledge transfer. In the results shown in Table I, for most problems in the case of CMTL, the performance deteriorates as subtasks are increased. This shows that the algorithm has not been able to fully utilize the transfer of knowledge. The proposed method the other hand shows that, in some cases, the prediction performance is improved as the subtasks increases (Sunspot, Lazer, ACI-finance).

VI. Conclusions and Future Work

We presented Bayesian multi-task learning for dynamic time series prediction via cascaded neural networks. The approach decomposed a single task learning problem of time series prediction into multi-task learning through subtasks that have inter-dependencies defined by the size of the window used for embedding. This enables robust prediction given limited or missing information during execution of the model in certain real-world application problems. A major challenge was to adapt MCMC method for neural networks that featured subtasks that required transfer of knowledge.

The training algorithm employed dynamic optimisation strategy where the knowledge from the foundation subtask was utilized in the subsequent subtasks through transfer learning. The results show improvement for most of the problems when compared to related method from the literature. Although related, CMTL does not feature uncertainty quantification when compared to the proposed Bayesian approach.

In future work, the performance of the proposed method can be further improved through Langevin dynamics where gradient decent is used to improve performance of random-walk MCMC methods. The approach can be used for pattern classification tasks and also for deep learning methods.

References

[1] R. Chandra, Y. Ong, and C. Goh, "Co-evolutionary multi-task learning for dynamic time series prediction," *CoRR*, vol. abs/1703.01887, 2017. [Online]. Available: http://arxiv.org/abs/1703.01887

[2] R. Chandra, "Dynamic cyclone wind-intensity prediction using co-evolutionary multi-task learning," in *Neural Information Processing - 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part V*, 2017, pp. 618–627.

[3] F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Warwick 1980*, ser. Lecture Notes in Mathematics, 1981, pp. 366–381.

[4] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, Jul. 1997.

[5] T. Evgeniou, C. A. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," *Journal of Machine Learning Research*, vol. 6, no. Apr, pp. 615–637, 2005.

[6] H. Zheng, X. Geng, D. Tao, and Z. Jin, "A multi-task model for simultaneous face identification and facial expression recognition," *Neurocomputing*, vol. 171, pp. 515 – 523, 2016.

[7] T. Zeng and S. Ji, "Deep convolutional neural networks for multi-instance multi-task learning," in *Data Mining (ICDM), 2015 IEEE International Conference on*, Nov 2015, pp. 579–588.

[8] P. Angeline, G. Saunders, and J. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," *Neural Networks, IEEE Transactions on*, vol. 5, no. 1, pp. 54 –65, jan 1994.

[9] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, Sep 1999.

[10] R. Chandra, Y. Ong, and C. Goh, "Co-evolutionary multi-task learning with predictive recurrence for multi-step chaotic time series prediction," *Neurocomputing*, vol. 243, pp. 21–34, 2017.

[11] R. Chandra, "Co-evolutionary multi-task learning for modular pattern classification," in *Neural Information Processing - 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part VI*, 2017, pp. 692–701.

[12] D. J. MacKay, "A practical bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

[13] ——, "Hyperparameters: Optimize, or integrate out?" in *Maximum entropy and Bayesian methods*. Springer, 1996, pp. 43–59.

[14] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.

[15] P. Jylänki, A. Nummenmaa, and A. Vehtari, "Expectation propagation for neural networks with sparsity-promoting priors." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1849–1901, 2014.

[16] G. E. Hinton and D. Van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in *Proceedings of the sixth annual conference on Computational learning theory*. ACM, 1993, pp. 5–13.

[17] C. Li, C. Chen, D. Carlson, and L. Carin, "Preconditioned stochastic gradient langevin dynamics for deep neural networks," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[18] F. Liang, "Annealing stochastic approximation monte carlo algorithm for neural network training," *Machine Learning*, vol. 68, no. 3, pp. 201–233, 2007.

[19] O. Kocadağlı and B. Aşıkgil, "Nonlinear time series forecasting with bayesian neural networks," *Expert Systems with Applications*, vol. 41, no. 15, pp. 6596–6610, 2014.

[20] R. Chandra, L. Azizi, and S. Cripps, "Bayesian neural learning via langevin dynamics for chaotic time series prediction," in *Neural Information Processing - 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part V*, 2017, pp. 564–573.

[21] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *J. Mach. Learn. Res.*, vol. 6, pp. 1817–1853, Dec. 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1046920.1194905

[22] L. Jacob, J. philippe Vert, and F. R. Bach, "Clustered multi-task learning: A convex formulation," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Curran Associates, Inc., 2009, pp. 745–752. [Online]. Available: http://papers.nips.cc/paper/3499-clustered-multi-task-learning-a-convex-formulation.pdf

[23] J. Chen, L. Tang, J. Liu, and J. Ye, "A convex formulation for learning shared structures from multiple tasks," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 137–144. [Online]. Available: http://doi.acm.org/10.1145/1553374.1553392

[24] J. Chen, J. Liu, and J. Ye, "Learning incoherent sparse and low-rank patterns from multiple tasks," *ACM Trans. Knowl. Discov. Data*, vol. 5, no. 4, pp. 22:1–22:31, Feb. 2012. [Online]. Available: http://doi.acm.org.ezlibproxy1.ntu.edu.sg/10.1145/2086737.2086742

[25] O. Griffiths, A. M. Johnson, and C. J. Mitchell, "Negative transfer in human associative learning," *Psychological science*, vol. 22, no. 9, pp. 1198–1204, 2011.

[26] Y. Zhang and D.-Y. Yeung, "Transfer metric learning by learning task relationships," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '10. New York, NY, USA: ACM, 2010, pp. 1199–1208. [Online]. Available: http://doi.acm.org/10.1145/1835804.1835954

[27] B. Bakker and T. Heskes, "Task clustering and gating for bayesian multitask learning," *J. Mach. Learn. Res.*, vol. 4, pp. 83–99, Dec. 2003. [Online]. Available: http://dx.doi.org/10.1162/153244304322765658

[28] S. Zhong, J. Pu, Y.-G. Jiang, R. Feng, and X. Xue, "Flexible multi-task learning with latent task grouping," *Neurocomputing*, vol. 189, pp. 179 – 188, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231216000035

[29] A. Gupta, Y. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," *IEEE Trans. Evolutionary Computation*, vol. 20, no. 3, pp. 343–357, 2016.

[30] R. Chandra, A. Gupta, Y.-S. Ong, and C.-K. Goh, "Evolutionary multi-task learning for modular knowledge representation in neural networks," *Neural Processing Letters*, Oct 2017. [Online]. Available: https://doi.org/10.1007/s11063-017-9718-z

[31] J. Xu, P. N. Tan, and L. Luo, "Orion: Online regularized multi-task regression and its application to ensemble forecasting," in *2014 IEEE International Conference on Data Mining*, Dec 2014, pp. 1061–1066.

[32] M. Leutbecher and T. Palmer, "Ensemble forecasting," *Journal of Computational Physics*, vol. 227, no. 7, pp. 3515 – 3539, 2008, predicting weather, climate and extreme events. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999107000812

[33] S. Banerjee, B. P. Carlin, and A. E. Gelfand, *Hierarchical modeling and analysis for spatial data*. Crc Press, 2014.

[34] F. Liang, "Bayesian neural networks for nonlinear time series forecasting," *Statistics and computing*, vol. 15, no. 1, pp. 13–29, 2005.

[35] H. S. Hippert and J. W. Taylor, "An evaluation of bayesian techniques for controlling model complexity and selecting inputs in a neural network for short-term load forecasting," *Neural networks*, vol. 23, no. 3, pp. 386–395, 2010.

[36] D. T. Mirikitani and N. Nikolaev, "Recursive bayesian recurrent neural networks for time-series modeling," *IEEE Transactions on Neural Networks*, vol. 21, no. 2, pp. 262–274, 2010.

[37] J. M. Bland and D. G. Altman, "Bayesians and frequentists," *BMJ*, vol. 317, no. 7166, pp. 1151–1160, 1998.