Continuous Time-series Forecasting with Deep and Shallow Stochastic Processes

Dan Teng
Neuri Pte Ltd.
Singapore
dan@neuri.ai

Sakyasingha Dasgupta
Neuri Pte Ltd.
Singapore
sakya@neuri.ai

Abstract

Time series prediction has wide spread applications in fields like finance, health care, robotics, sensory data analytics etc. In the context of financial time series, forecasting needs the ability to deal with inherently non-stationary data, limited training examples, non-linearity and uncertainty. Prior work, have explored the use of Gaussian process (GP) in this context to adapt with test time data, making use of prior knowledge. However, their usage can be limited by a high computational time complexity, especially when applied in a continuous or online learning setting. In this paper, we juxtapose GPs with a recently proposed deep neural network model, conditional neural process (CNP) for online time-series prediction. Unlike GPs, CNPs have a linear time complexity and can be trained via stochastic gradient descent. We compare CNPs to standard GPs as well as deep Gaussian processes (DGPs) with hierarchical composition of GPs. Furthermore, in all cases we formulate an online or continuous learning scheme which allows the models to be trained with limited amount of data, continuously, adjusting to changes in data distributions occurring due to unexpected events. We evaluate all models first using the chaotic Mackey-Glass time series and finally on foreign exchange rate prediction task. We find that in an online setting, sparse single layer GPs outperform other models in terms of prediction accuracy metrics, however, require a larger training time complexity with pure trend following behavior learned for financial market data. CNPs can adapt faster and learn with limited new data. In all cases the deep GPs did not provide any significant performance gain. Using cross-correlation analysis we show that without extensive tuning on actual financial data, all models can suffer from trend following behavior, with the strongest effect identified in shallow GP models.

1 Introduction

Time series prediction is ubiquitous in numerous real life applications. Classical prediction methods such as auto-regression, moving average and exponential smoothing create linear functions on certain transforms of the observation to make the next step prediction [6; 20; 12; 11]. Recent work have applied deep neural networks (DNN) [4; 1; 2] with inherently less restrictions on the input and are aimed at exploring the underlying nonlinear relationship in the data to make better predictions. As DNNs learn to approximate a function that maps the observations to predictions, they are not typically designed to deal with non-stationarity and mismatch in data distribution. Furthermore, training DNNs require a large number of examples. Alternative to neural network based models, stochastic process based model like Gaussian process (GP) can make use of prior knowledge and learn a distribution over functions rather than a single function approximation [19; 5; 15]. This is particularly useful in the context of financial time series due to its inherent volatility, and the tendency of training data not being a true reflection of test time data.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

However, performance of GP models depend on the selection of a suitable prior with a kernel function. Expressiveness of the prior can be improved by hierarchical composition of GPs in the form of a deep GP model (DGP) [7; 21]. Nevertheless, the usage of GPs and DGPs may be limited by their high computational expense, even with sparse Gaussian process regression (SGPR) and variational Gaussian approximation (SVGP) or their deep variants [22; 17; 25]. Moreover, the application of sparse GP and DGP models for online time-series prediction has been under-explored. A recently proposed deep neural network model conditional neural process (CNP) was created based on flexibility of GPs while structured as a neural network and trained via stochastic gradient descent [9; 10]. This relaxes the requirement for a suitable prior selection and also significantly lowers the computational complexity. As CNPs are created with the goal of meta or continuous learning with limited data, here we explore their usage and performance in continuous time series prediction, juxtaposing with the performance obtained with shallow and deep GPs. In Section 2, we provide a brief overview of the prediction problem formulation with continuous learning (in supplementary Section 5 we provide overview of the shallow and deep DP models, along with details of formulation of CNP). In Section 3, we evaluate the performance of GP, DGP and CNP with different online settings for synthetic Mackey-Glass chaotic time series prediction, as well as for real foreign exchange rate prediction. The performance is discussed in the context of dealing with limited data, computational complexity and predictive accuracy. A final conclusion and discussion of future research directions are provided in Section 4.

2 Problem Formulation

A standard time series prediction problem can be formed as: Given a set of data points $\{\mathbf{x}_t, y_t\}$ for $t \in \{1, \cdots, n\}$ which represent the past n time steps' information, the goal is to learn the relationship between $\{\mathbf{x}_t\}$ and $\{y_t\}$ via a function approximation $y = f(\mathbf{x})$. This function can then be used to predict the next step y_{n+1} given $\mathbf{x}_{n+1}$. Where, $\mathbf{x}_t = \{x_t^1, x_t^2, ..., x_t^n\}$ is a multivariate signal. However, while working with non-stationary multivariate time-series, a single deterministic function approximation may not be suitable due to the limited predictive information and inability to account for change in time series data distribution. Alternately, in order to account for uncertainties in our predictions and a better generalization, we prefer predictions in the form of distributions of functions. See supplementary Section 5 for overview of deep and shallow neural processes learning such a distribution over functions.

2.1 Continuous or Online Learning

A common behavior in time series is that the pattern or relationship between $\mathbf{x}$ and y varies over time. In financial time series, this could be due to a number of external or internal events like, policy reforms, intrinsic turbulence in the market, news sentiments, natural disasters, etc. In order to cope with this, prediction models need to adapt continuously with limited new samples of data points. In this work, in order to formulate the stochastic process based models within an online framework, we update parameters of the model continuously over a moving window, making predictions more timely and robust. Varying the size of this moving window, it is possible to change the update frequency of the models, with a completely online model when updating at every interval of time. As depicted in Figure 1, the data is split into training and testing windows and the model learns with limited samples from within the training set. In a purely online update, the testing set consists of a single data point. The window can be shifted forward in a walk-forward step equal to the the number of data points in the testing set. Since the parameters are only updated during training, the window size and split between train and test, may act as hyper-parameters. This setting assumes that future values are affected more by the recent past. To make the models comprehensive, longer historical information can be added to the window by selecting points from the past.

3 Experiments

In this section, we evaluate the performance of GP, DGP and CNP models on time series prediction within a continuous learning setting as described in subsection 2.1. Here, we define a window size n_w and a step size n_s. The parameters of each model are trained on the data inside the window of size n_w and the earliest 10% of the data points in the window are replaced by a random selection of data points from earlier history. Then the trained parameters are used for the prediction of the following n_s data points and the model is retrained after each n_s step. This process is illustrated in Figure 1.

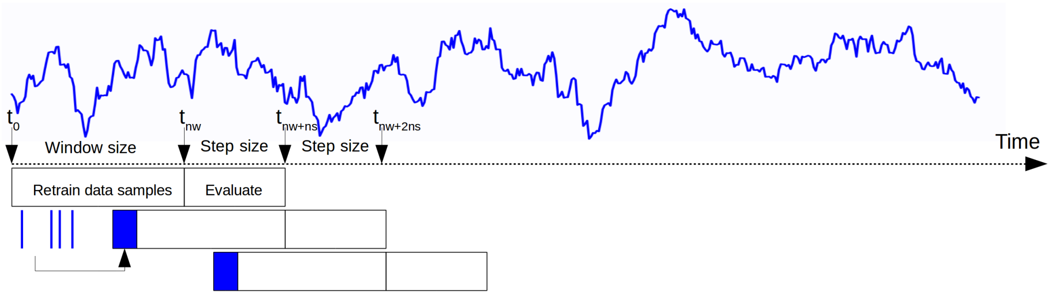


Figure 1: Illustration of the continuous time-series learning setup

In our first implementation, n_w and n_s are set as 1000 and 100 respectively, as such the model parameters are updated every 100 time steps, i.e., the prediction within 100 time steps are generated using the fixed parameters. Here the underlying assumption is that during the n_s consecutive data points the same underlying distribution as the earlier $n_w = 1000$ data points is maintained. We use this same configuration for comparing all the three models, GP, DGP and CNP. Additionally we examine a fully online CNP model (CNP1) with step size 1, as such the model parameters are updated at every time step. This model also uses a considerably smaller training window size of 300 data points, thus effectively updating with very limited examples. This is computationally more expensive compared to the 100 update step CNP model, but can be suitable for highly non-stationary time series. Here we did not consider the full online version of GP and DGP as unlike the iterative updating scheme in CNP, the parameters are retrained in the GP models *every batch*. Moreover the time complexity of fully online GP and DGP models would be too high to be practically considered.

The three models are implemented with the following hyper-parameters:

- GPs: SGPR, SVGP are implemented using built-in functions in GP-flow [15] library. The kernel function used is RBF, with number of inducing points and iterations both set to 100.
- DGPs: The implementation follows prior work from [21], two approaches are examined, one is using Adam optimizer for all layers (DDGP) and the other is with a natural gradient optimizer only for the last layer ($DDGP_{NG}$). The kernel function is RBF, number of inducing points, iterations and layers are 100, 100 and 3 respectively.
- CNPs: The context ratio is set to be 50% of the training data, the context and target sets are formed by selecting the first half and second half of data points after random shuffling of the data points. Training is performed by minimizing the negative log-likelihood incurred at all different context and target sets within a batch. 5000 iterations were used for each update.

In terms of synthetic dataset, we use the Mackey-Glass (MG) chaotic time series which exhibits delay induced chaotic behavior. The discrete time-delay series is generated as follows:

$$Y_{n+\tau} = Y_{n+\tau-1} - \gamma Y_{n+\tau-1} + \frac{\beta Y_n}{1 + Y_n^p}. \tag{1}$$

Where, τ is the delay factor indicating that the current value not only depends on the last value, but also on values τ time steps back; β and γ are weights for each component and p is the magnifying factor. As most time series with practical relevance are nonlinear and may exhibit chaotic signatures, the MG time series serves as a good representative. In our experiments, we look at two MG time series with increasing complexity, $\tau = 30$ and $\tau = 50$. β, γ and p were fixed at 0.2, 0.1 and 10, respectively.

In order to assess the performance of the models on actual financial time series, we use foreign exchange (FX) with time frequency defined as a day starting from January 30 2005. Here we train the models to predict the next day price of a single asset based on multivariate time-series as input. The input to the models, $\mathbf{x}$ was set as the past 20 days of CAD/USD price and the target y as the next day price. The same input and target settings were used in the MG experiments.

In both MG and FX experiments, the length of the time series is fixed at 4000 data points[1] (for the fully online CNP1 model only the first $n_w + 1000$ data points are used[2]). In all cases we stationarize the time series by taking the z-transform using a moving window of past 100 day price for each data point.

In Table 1, we summarize the prediction results across all models and experiments. We compare the performance of the models using the negative log-likelihood (NLL)[3] and the symmetric mean average percentage error $SMAPE = \frac{100\%}{n} \sum_{t=1}^{n} \frac{|\hat{y}_t - y_t|}{|\hat{y}_t| + |y_t|}$. Since CNP is a continuous learning model whose parameters are trained gradually, here besides the overall measures, we also include the same measures over the final 500 data points for comparison.

Table 1: Negative log likelihood (NLL) and symmetric mean average percentage error(SMAPE)

Model	MG ($\tau = 30$)				MG ($\tau = 50$)				FX (CAD/USD)			
	NLL		SMAPE (%)		NLL		SMAPE (%)		NLL		SMAPE (%)	
	Overall	Last 500	Overall	Last 500	Overall	Last 500	Overall	Last 500	Overall	Last 500	Overall	Last 500
SGPR	-5.36	-5.39	0.06	0.05	-5.18	-5.29	0.06	0.06	-3.78	-3.49	0.18	0.19
SVGP	-3.39	-3.50	0.38	0.32	-3.29	-3.34	0.42	0.42	-3.76	-3.49	0.19	0.19
DDGP	0.003	0.02	8.73	9.64	0.13	0.21	11.90	14.07	-2.42	-2.01	0.87	0.97
DDGP$_{NG}$	0.05	0.06	9.44	9.92	0.18	0.26	12.94	15.05	-2.40	-1.95	0.91	1.09
CNP100	-0.59	-2.01	4.17	1.83	0.48	-0.28	5.98	5.20	-0.77	-2.83	0.60	0.28
CNP1	-2.00	-2.69	0.89	0.12	-2.33	-2.56	0.64	0.17	-2.19	-1.27	0.45	0.64

From the results, we see that single layer sparse GPs (i.e., SGPR and SVGP) achieve the most accurate prediction in both MG and FX time series. However the performance of GPs relies heavily on the choice of kernel functions. Using the $Matern12$ kernel instead of RBF, leads to a large performance drop with clear sign of trend following. In addition, the performance of SGPR and SVGP also suffers with the reduction in window size. In terms of the NLL and SMAPE metrics, CNPs give the next best results. When comparing the overall and the last 500 data points' results with both metrics, the CNP models improve significantly (especially for CNP100) displaying ability to learn continuously, the performance gets better over time as more data points are observed. The prediction of the entire FX time series with CNP100 model is show in Figure 2(a) which displays a good fit to the data. It can also be observed that the fit to the data improves over time. In Figure 2(b) we show a zoomed in view of the same chart, displaying that though the overall fit is quite robust with continuous learning, there can be periods with some trend following behavior. We also observe that the performance of CNP remains relatively stable while testing with a smaller batch size. Contrary to results reported in recent literature, among all the models, the DDGP variants perform the worst on time series prediction task using the same number of training iterations.

Finally, we take into account the cross-correlation between the prediction and the actual time series, in order to check if the models learn a trend following behavior although the reported performance in terms of metrics may be high. In supplementary Table 2, we list down the lag where the peak of the correlation occurs. Lag $= \alpha$ indicates the prediction at time t has the highest correlation with the actual time series at time $t + \alpha$ for all t in the time series on average. For models SGPR, SVGP, CNP100 and CNP1 on MG time series, peak at lag $= 0$ is observed, showing clearly no sign of trend following. If lag is a negative value, it indicates that the prediction is more correlated with some value in the past, especially a peak at lag $= -1$ indicates one time step trend following behavior, see supplementary Figure 5. The shallow GP and CNP models all show a correlation peak at lag$= -1$. However, on closer inspection the GP models prediction seems to just copy the previous time step data, whereas the CNP model though on average has a peak at lag$= -1$, there are multiple periods in which it does not learn to merely copy the previous time step data (see Figure 2). We leave further analysis of such behavior for future work.

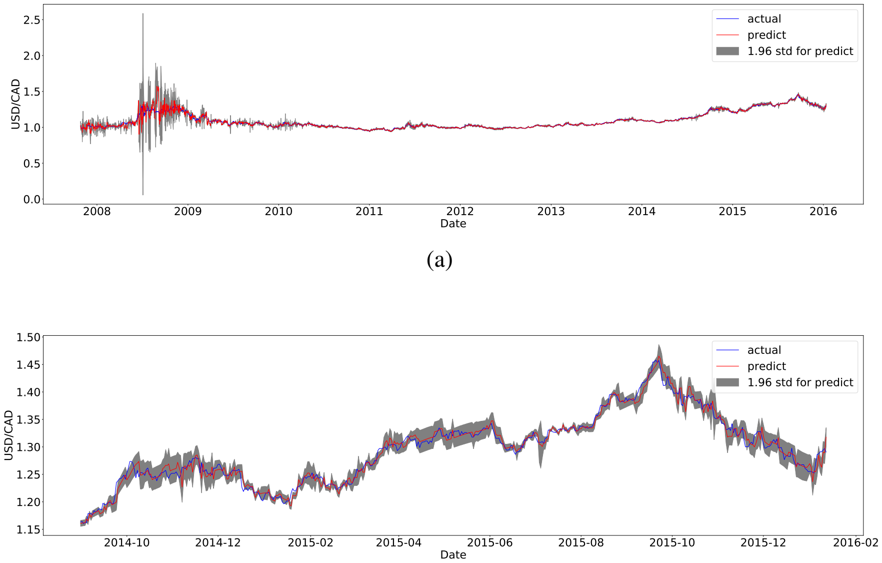
4 Conclusions and Future Directions

In this paper, we examined the performance of single layer sparse GP, deep GP and CNP models for continuous learning to predict Mackey-Glass time series and foreign exchange financial time series.

[1] In case of CAD/USD price data, this amounts to data from January 30, 2005 until January 12, 2016.

[2] For CNP1 model, n_w is set as 1000 for MG time series and 300 for FX time series.

[3] This is a relative measure, which can be positive or negative. When comparing models, the lower NLL indicates higher likelihood and better performance.



(a)

(b)

Figure 2: CNP100 model (Target VS Prediction) on daily CAD/USD foreign exchange prediction task. (a) Entire time series (b) Zoomed in view (see the corresponding performance of the CNP1 model in supplementary Figure 4).

Contrary to expectation, we observe that, sparse GPs outperform the other models in terms of NLL and SMAPE in both datasets. This shows that with suitable kernels, single layer GPs can achieve better results compared to the more generalized models DGPs and CNPs. However this comes at a high computational cost, which may make them less suitable in continuous learning settings. In addition for real world data, the GP models show a clear one-step trend following behavior on the entire dataset. As CNPs are designed as a clear continuous learning process, we show that it achieves better results after training on more data points. Furthermore with no requirements to choose a suitable kernel function, ability to learn with very few examples, and a linear computational time complexity, CNPs would be more flexible and adaptable to different real world time series with inherent non-stationary dynamics.

In all models, including CNPs, we observe an issue of some degree of trend following behavior when applied to financial data. We observe that similar behavior occurs with most neural network models when predicting a single time step into the future. As most reported results present performance based on metrics like root mean squared error, mean absolute error etc, such behavior is not identified directly. Moreover, based on further experiments, we observe that such behavior is less frequently present in multiple-step ahead prediction with sequence to sequence type models [23], recent stochastic models like the nonlinear dynamic Boltzmann machines [8], as well as with causal dilated convolution based models [16]. This remains outside the scope of this paper and will be investigated in future work to juxtapose with neural processes when applied in continuous learning settings. Another possible research direction is to ease the assumption of Gaussian distribution of the output obtained from the decoder in CNP, especially when applied in the context of financial time series.

References

[1] W. Bao, J. Yue and Y. Rao, *A deep learning framework for financial time series using stacked autoencoders and long-short term memory*, in Public Library of Science One, 12-7, 2017.

[2] A. Borovykh, S. Bohte and C.W. Oosterlee, *Conditional time series forecasting with convolutional neural networks*, arXiv: 1703.04691, 2017.

[3] S. Brahim-Belhouari and A. Bermak, *Gaussian process for nonstationary time series prediction*, in Computational Statistics & Data Analysis, pp. 705-712, 2004

[4] Z. Che, S. Purushotham, K. Cho, D. Sontag and Y. Liu, *Recurrent neural networks for multivariate time series with missing values*, in Scientific Reports, 8-1, 2018.

[5] L. Cheng, G. Darnell, C. Chivers, M.E. Draugelis, K. Li and B.E. Engelhardt, *Sparse multi-output Gaussian processes for medical time series prediction*, arXiv: 1703.09112, 2017.

[6] J. Contreras, R. Espinola, F.J. Nogales and A.J. Conejo, *ARIMA Models to predict next-day electricity prices*, in IEEE Transactions on Power Systems, 18-3, pp. 1014-1020, 2003.

[7] A. C. Damianou, and N. D. Lawrence, *Deep Gaussian processes*, in Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), 2013.

[8] S. Dasgupta and T. Osogami, *Nonlinear dynamic Boltzmann machines for time-series prediction*, in AAAI Conference on Artificial Intelligence, 2017.

[9] M. Garnelo, D. Rosenbaum, A. J. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. J. Rezende, and S. M. Eslami, *Conditional neural processes*, in Proceedings of the International Conference on Machine Learning (ICML), 2018

[10] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D.J. Rezende, S.M. Ali Eslami and Y. W. Teh, *Neural processes*, in Theoretical Foundations and Applications of Deep Generative Models Workshop, ICML, 2018.

[11] J.D.D. Gooijer and R. Hyndman, *25 years of time series forecasting*, in International Journal of Forecasting, 22, pp. 442-473, 2006.

[12] R.J. Hyndman, A.B. Koehler, R.D. Snyder and S. Grose, *A state space framework for automatic forecasting using exponential smoothing methods*, in International Journal of Forecasting, 18, pp. 4390454, 2002.

[13] J. Hensman, N. Fusi and N. D. Lawrence, *Gaussian processes for big data*, in the Conference on Uncertainty in Artificial Intelligence (UAI), 2013.

[14] J. Hensman, A.G.de G. Matthews and Z. Ghahramani, *Scalable variational Gaussian process classification*, in the International Conference on Artificial Intelligence and Statistics (AISTATS), 2015.

[15] A.G. de G. Matthews, M.v.d. Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani and J. Hensman, *GPflow: a Gaussian process library using Tensorflow*, in Journal of Machine Learning Research, 18, pp.1-6, 2017.

[16] A. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior and K. Kavukcuoglu, *Wavenet: a generative model for raw audio*, arXiv: 609.03499, 2016.

[17] M. Opper and C. Archambeau, *The variational Gaussian approximation revisited*, in Neural Computation, 21-3, pp. 786-792, 2009.

[18] C. E. Rasmussen and C. K.I. Williams, *Gaussian Processes for Machine Learning*, the MIT Press, 2006.

[19] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson and S. Aigrain, *Gaussian processes for time-series modelling*, in Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 371-1984, pp. 1-25, 2013.

[20] E.W. Saad, D.V. Prokhorov and D.C. Wunsch, *Comparative study of stock trend prediction using time delay recurrent and probabilistic neural networks*, in IEEE Transactions on Neural Networks, 9-6, pp. 1456-1470, 1998.

[21] H. Salimbeni and M. Deisenroth, *Doubly stochastic variational inference for deep gaussian processes*, in Advances in Neural Information Processing Systems (NIPS), pp. 4591–4602, 2017.

[22] E. Snelson and Z. Ghahramani, *Sparse Gaussian processes using pseudo-inputs*, in Advances in Neural Information Processing Systems (NIPS), 18, 2005.

[23] I. Sutskever, O. Vinyals and Q.V. Le, *Sequence to sequence learning with neural networks*, in Proceedings of Neural Information Processing Systems (NIPS), pp. 3104-3112, 2014.

[24] M. K. Titsias, *Variational learning of inducing variables in sparse Gaussian processes*, in the International Conference on Artificial Intelligence and Statistics (AISTATS), 2009.

[25] M. van der Wilk, C.E. Rasmussen and J. Hensman, *Convolutional Gaussian processes*, in Advances in Neural Information Processing Systems (NIPS), pp. 2845-2854, 2017.

5 Supplementary

5.1 Gaussian Processes and Deep Gaussian Processes

Bayesian models such as Gaussian processes [18] generate distributions of functions that describe the relationship between $\{\mathbf{x}_t\}$ and $\{y_t\}$. Let f be a function mapping some input space $\mathcal{X}$ to $\mathcal{R}$ and $\mathbf{f}$ be an n-dimensional vector of function values evaluated at n points $\mathbf{x}_i \in \mathcal{X}$, then $p(f)$ is a **Gaussian process** (GP) if for any finite subset $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathcal{X}$, the marginal distribution over that finite subset $p(\mathbf{f})$ has a multivariate Gaussian distribution. A GP is fully specified by its mean and covariance.

A typical idea for a GP to make prediction is to first set a Gaussian prior for the distribution of function f, with a given set of data points $\{\mathbf{x}_t, y_t\}$ which is assumed to contain Gaussian noise, the posterior is then obtained by marginalizing the product of likelihood $p(f^*|\mathbf{f})$ and prior $p(\mathbf{f})$ over $\mathbf{f}$. Despite the simple derivation, GP requires $O(n^3)$ computations to obtain the prediction mean and standard deviation which is expensive with large observation set. A few sparse versions of GP, such as SGPR, SVGP have been introduced in [14; 24; 13] and the main idea is to compute the approximation of $p(f^*, \mathbf{f})$ by assuming the independence of f^* and $\mathbf{f}$ given some inducing variables and computing the predictive probability through variational inference, the computational cost can be reduced to $O(nm^2)$ where m is the number of inducing points.

Single layer GP models are limited by the expressiveness of the kernel function in the selected prior, this could be addressed by introducing a hierarchical composition of GPs, known as **Deep Gaussian Process** (DGP). A DGP is a deep network in which each layer is modelled by a GP. It provides the possibility to model highly nonlinear functions for complex datasets. Due to the hierarchical structure, it often requires very few hyperparameters in each layer GP. Many DGP models [7] assume independence between layers to perform inference. A recently proposed method Doubly stochastic variational inference for deep Gaussian process (DDGP) [21] uses sparse variational inference to simplify the correlations within layers and maintain the correlations between layers.

5.2 Conditional Neural Processes

The original **Conditional neural process** (CNP) model was proposed in [9], as a combination of deep neural networks which are suitable for identifying nonlinear relationships among variables and Gaussian process. A CNP contains two components, an encoder which creates the relationship function r between inputs x and targets y, a decoder which takes input r and a new observation x to generate prediction mean and standard deviation. The model is illustrated in Figure 3.

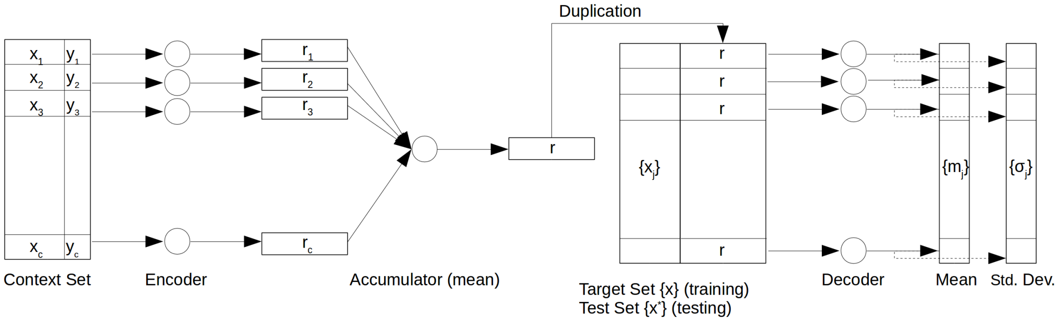


Figure 3: CNP model illustration

The standard procedure is to first split the observation dataset into context and target sets. At training stage, the encoder receives the context set $\{x_i, y_i\}$ and outputs r_i which are aggregated to r, together with the target set $\{x_j\}$, they are passed to the decoder to produce the mean $\{m_j\}$ and standard deviation $\{\sigma_j\}$. Then the negative likelihoods of the actual $\{y_j\}$ sampled from a Gaussian distribution with parameters $\{m_j\}$ and $\{\sigma_j\}$ are minimized through multiple iterations in order to update parameters in both encoder and decoder. Upon training the parameters, the test dataset $\{x^*\}$ is passed into the decoder to predict its mean and standard deviation. In contrast to other stochastic process models like GPs and DGPs, the time complexity for CNP is linear, $O(n)$.

It should be noted that the main idea in CNP is to produce the representation vector r and under this construction, the underlying correlations are assumed to be the same for the context, target and test sets. Even though the relationship is fully represented by exploiting the context set, the target set could tell us how to improve the decoder in order to align with its underlying relation. Additionally, as the output of the decoder is a distribution over the prediction, CNP is able to adapt to the changes in observation set. In comparison with GPs or DGPs, CNPs alleviate the restrictions posed by the mean and kernel function of the Gaussian prior. It is able to generate the underlying relationship with small observation set compared to GPs which typically require a large amount of data to train the parameters in the kernel function.

Table 2: Lag peak for cross-correlation between prediction and actual time series.

Model	MG ($\tau = 30$)	MG ($\tau = 50$)	FX (CAD/USD)
SGPR	0	0	-1
SVGP	0	0	-1
DDGP	-3	-5	-15
DDGP$_{NG}$	-3	-5	-22
CNP100	0	0	-1
CNP1	0	0	-1

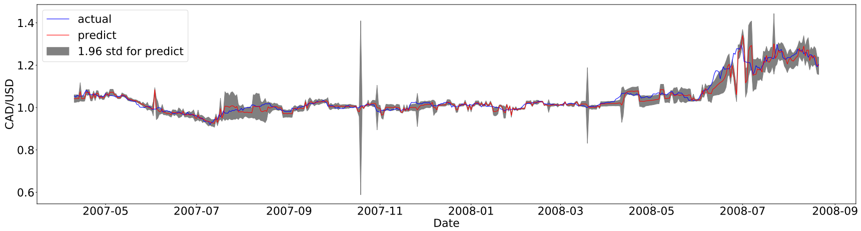
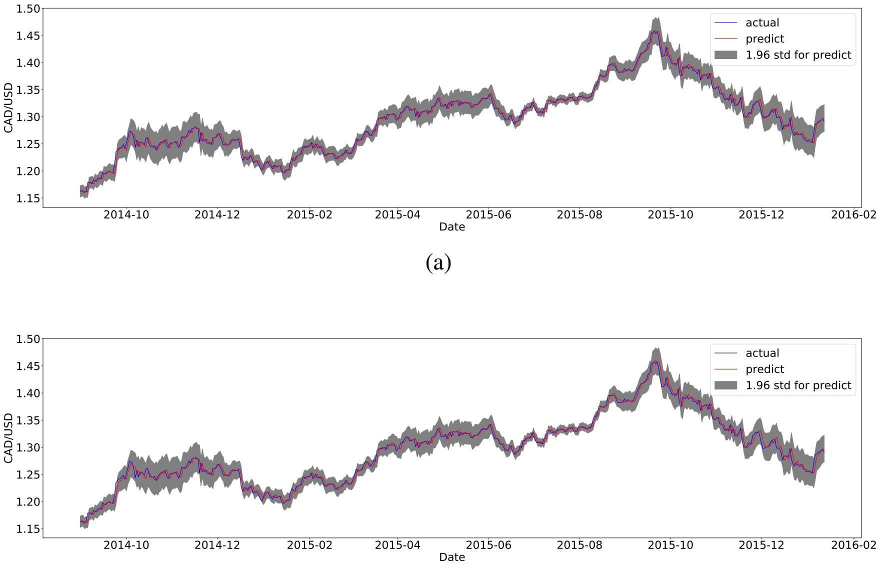


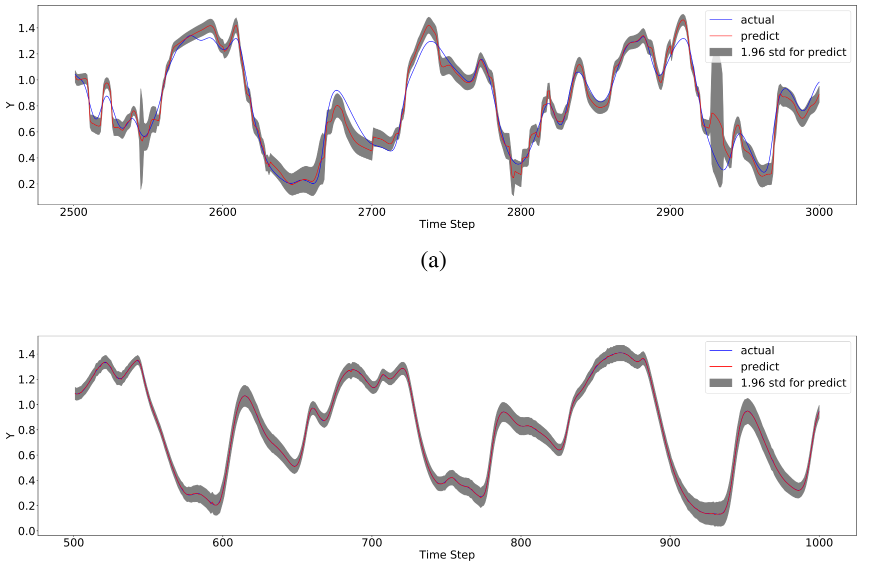
Figure 4: CNP1 models (Target vs Prediction) on daily CAD/USD foreign exchange task.



(a)

(b)

Figure 5: (a) SGPR (b) SVGP models (Target VS Prediction) on daily CAD/USD foreign exchange prediction task. A clear one-step trend following behavior is observed.



(a)

(b)

Figure 6: (a) CNP100 (b) Fully online CNP1 models (Target VS Prediction) on MG $\tau = 50$ prediction task (zoomed in to the last 500 data points).