A TRAINING METHOD USING DNN-GUIDED LAYERWISE PRETRAINING FOR DEEP GAUSSIAN PROCESSES

Tomoki Koriyama, Takao Kobayashi

School of Engineering, Tokyo Institute of Technology, Yokohama, Japan

{koriyama, takao.kobayashi}@ip.titech.ac.jp

ABSTRACT

This paper proposes a novel framework of training method of deep Gaussian processes (DGPs). DGPs are deep architecture models based on stacked multiple GPs, which can overcome the limitation of single-layer GPs. Although a stochastic variational inference (SVI)-based method has been proposed for DGP training with an arbitrary amount of training data, it does not always update model parameters appropriately due to repeated Monte Carlo sampling of multiple GPs. To resolve this problem, we propose a pretraining method to determine initial parameters of DGPs. In the proposed method, layer-wise training of single-layer GPs is performed using the hidden-layer values obtained by a deep neural network (DNN) that has an analogous structure to a target DGP model. The proposed method utilizes the characteristics of single-layer GPs and deep neural network (DNN) whose training is easier than DGP's. Experimental results using two speech synthesis databases with approximately 600 K and 1.4 M training data points, respectively, which were composed of hundreds of input and output features, gave the effectiveness of the proposed method.

***Index Terms*—** deep Gaussian process, neural network, pretraining, kernel method

1. INTRODUCTION

Machine learning using deep neural networks (DNNs) has become mainstream in diverse research fields, such as speech, language, and image processing. Various studies showed that DNN, which is generally the combination of linear transformation and non-linear activation functions, can express arbitrary mapping functions between input and output features. The performance of DNN is mainly attributed to weight matrices with a huge amount of parameters. However, a huge amount of DNN parameters often causes an overfitting problem, therefore, regularization techniques are essential for the generalization ability of DNN for unknown data [1, 2].

As another method of expressing functions, Damianou and Lawrence proposed [3] deep Gaussian processes (DGPs) that consists of multiple Gaussian process (GP) regressions. GP regression is based on kernel regression in which the pairs of input and output features are directly used for prediction instead of the weight matrices. An advantage of DGP over single-layer GP regression is expressiveness of kernel functions, which represent the relationships among input features. Since the performance of GP regression depends on its kernel function, the tuning of kernel function is a crucial problem and it often becomes a laborious task. In contrast, DGPs can construct complicated kernel functions automatically from training data.

The original DGP [3] had difficulty in computational complexity for a large amount of training data. Recently, Salimbeni and Deisenroth [4] proposed doubly stochastic variational inference (DSVI) technique to overcome the problem. Specifically, they combined the stochastic variational inference [5, 6], which optimizes evidence lower bound (ELBO) by stochastic optimization, and Monte Carlo sampling known as reparametrization trick [7]. However, the training of DSVI-based DGP becomes difficult when we use many layers. Since the DSVI-based DGP requires multiple Monte Carlo samplings, this may cause improper propagation of gradients.

To overcome the improper propagation problem, Salimbeni and Deisenroth [4] proposed a method using non-zero mean functions, while the mean functions of GPs are generally set to zero. In this method, they used simple linear transformation, such as primary component analysis (PCA)-based dimension reduction and identity matrix, as a mean function of GP of the respective layer. Similarly to the residual connection widely used in DNN architecture, it can avoid a gradient vanishing problem. It is true that this method is effective when we use simple feed-forward DGP models. However, to extend DGP to more complicated architecture, such as recurrent and convolution, the design of mean function becomes difficult. Therefore, we consider to use DGPs with zero mean functions and investigate training method of them based on the DSVI.

In this context, we propose a two-stage pretraining method that determines the initial values of the DVSI-based DGP. In the proposed pretraining method, first, we train a DNN that has similar architecture to a target DGP model, inspired from the analogy of neural network and Gaussian process reported in [8–10]; a neural network with infinite units is regarded as a sample from a Gaussian process. After that, we train the parameters of GP for each layer individually using hidden layer values obtained from the DNN as the guides of mapping functions. Then, we use the trained layer-wise parameters as the initial parameters of DGPs. This method reduces the difficulty of the training of DGPs by utilizing models that are easy to train, such as DNN and single-layer GP.

In this paper, we investigate the effectiveness of the proposed method using speech synthesis database. We apply the proposed training method to a traditional statistical parametric speech synthesis based on DNN [11]. We train models where a frame-level context vector is used as the input and corresponding acoustic features as the output. Context vectors, which represent linguistic and positional information, generally consist of complicated features with hundreds of dimensions. Hence, the performance with shallow GP is not sufficient to encode the complicated input features efficiently. We demonstrate that the initial parameter values obtained by the proposed method can give smaller distortion in earlier training epochs than others.

A part of this work was supported by KAKENHI Grant Number 17K12711.

2. BACKGROUND

2.1. DGP Based on Doubly Stochastic Variational Inference

DGP [3] is a stacked model of multiple GPs. Specifically, it is assumed with DGPs that the latent function of a single-layer GP is a composite function whose partial functions are sampled from individual GPs. This latent function f is represented as

$$f = f^L \circ f^{L-1} \circ \cdots \circ f^1 \tag{1}$$

$$f^\ell \sim \mathcal{GP}(m^\ell(\cdot), k^\ell(\cdot,\cdot;\theta^\ell)) \tag{2}$$

where L is the number of layers of the stacked model, and $m^\ell(\cdot)$ and $k^\ell(\cdot,\cdot;\theta^\ell)$ denote mean and kernel functions, respectively. Let the hidden layer variables of layer ℓ given input $\mathbf{x}_i$ be

$$\mathbf{h}_i^\ell = f^\ell(f^{\ell-1}(\ldots(f^1(\mathbf{x}_i)))) = f^\ell(\mathbf{h}_i^{\ell-1}). \tag{3}$$

The relationship between the output variable $\mathbf{y}_i$ and the output of the latent function, $f(\mathbf{x}_i) \triangleq \mathbf{f}_i$, is represented by likelihood function $p(\mathbf{y}_i|\mathbf{f}_i)$. For example of regression case, a Gaussian distribution is generally used for the likelihood function.

DSVI-based DGP [4] is an approximation technique that enables us to train DGP models by stochastic optimization. The DSVI-based DGP takes the following ELBO $\mathcal{L}$ as a cost function:

$$\mathcal{L} = \sum_{i=1}^{N} \Bigg\{ \mathbb{E}_{q(\mathbf{f}_i)}\left[\log p(\mathbf{y}_i|\mathbf{f}_i)\right] - \frac{1}{N}\sum_{\ell=1}^{L} \mathrm{KL}(q(\mathbf{U}^\ell)\|p(\mathbf{U}^\ell|\mathbf{Z}^\ell)) \Bigg\} \tag{4}$$

where N is the size of training data, and $\mathbf{Z}^\ell$ and $\mathbf{U}^\ell$ are the inducing inputs and outputs [12] of layer ℓ. $\mathrm{KL}(\cdot)$ is Kullback-Leibler divergence and $q(\mathbf{U}^\ell)$ is a variational distribution that has mean $\mathbf{M}^\ell$ and covariance $\mathbf{S}^\ell$ as parameters. The first and second terms of (4) represent data-fit and complexity penalty, respectively, and $q(\mathbf{f}_i)$ is a variational posterior defined as

$$q(\mathbf{f}_i) = \mathrm{SVGP}(\mathbf{f}_i; m^L(\cdot), k^L(\cdot,\cdot), \hat{\mathbf{h}}_i^{L-1}, \mathbf{Z}^L, q(\mathbf{U}^L)). \tag{5}$$

$\mathrm{SVGP}(\mathbf{f}_i; m(\cdot), k(\cdot,\cdot), \hat{\mathbf{x}}_i, \mathbf{Z}, q(\mathbf{U}))$ represents the variational posterior used in single-layer GP with stochastic variational inference (SVI) [6, 13], in which mean and variance are defined as

$$\boldsymbol{\mu}_i = m(\mathbf{x}_i) + \boldsymbol{\alpha}_i^\top(\mathbf{M} - m(\mathbf{Z})) \tag{6}$$

$$\Sigma_i = k(\mathbf{x}_i,\mathbf{x}_i) - \boldsymbol{\alpha}_i^\top(K(\mathbf{Z},\mathbf{Z}) - \mathbf{S})\boldsymbol{\alpha}_i \tag{7}$$

$$\boldsymbol{\alpha}_i = K(\mathbf{Z},\mathbf{Z})^{-1}K(\mathbf{Z},\mathbf{x}_i) \tag{8}$$

where $K(\mathbf{X},\mathbf{X})$ is a matrix whose (i,j) element is obtained by the kernel function $k(\mathbf{x}_i,\mathbf{x}_j)$. For $\ell = 1\ldots L-1$, the sampling is performed by

$$\hat{\mathbf{h}}_i^\ell = \boldsymbol{\mu}_i^\ell + \boldsymbol{\epsilon}_i^\ell \sqrt{\Sigma_i^\ell} \tag{9}$$

where $\boldsymbol{\epsilon}_i^\ell$ is a noise sampled from a standard normal distribution, and $\boldsymbol{\mu}_i^\ell$ and Σ_i^ℓ are the mean and variance of variational posterior $\mathrm{SVGP}(\mathbf{h}_i^\ell; m^\ell(\cdot), k^\ell(\cdot,\cdot), \hat{\mathbf{h}}_i^{\ell-1}, \mathbf{Z}^\ell, q(\mathbf{U}^\ell))$.

As a result, the posterior $q(\mathbf{f}_i)$ is obtained by repeating sampling as shown in Fig. 1. The model parameters are composed of inducing inputs $\{\mathbf{Z}^\ell\}_{\ell=1}^L$, the variational parameters of inducing outputs $\{(\mathbf{M}^\ell,\mathbf{S}^\ell)\}_{\ell=1}^L$, and the kernel function parameters $\{\theta^\ell\}_{\ell=1}^L$.

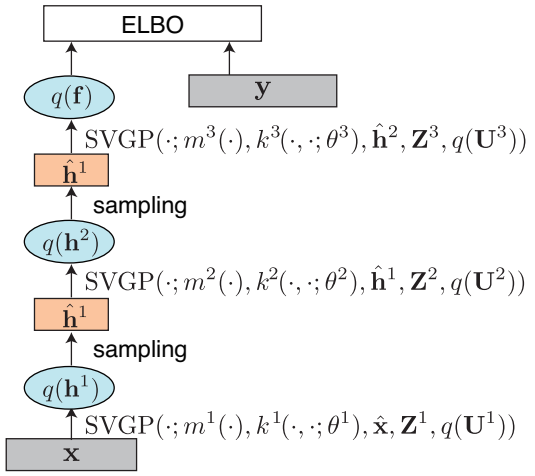


Fig. 1: ELBO calculation process of a 3-layer DGP model based on DSVI.

2.2. Non-zero Mean Function DGP

As described in Sect. 2.1, DSVI-based DGP uses repeated sampling. Since this makes gradient propagation difficult, the method using non-zero mean functions has been proposed in [4]. Specifically, a copy function $m(\mathbf{x}) = \mathbf{x}$ was used when the dimensions of two layers are equivalent, and when the output dimension is smaller than the input, linear transformation $m(\mathbf{x}) = \mathbf{W}\mathbf{x}$ was used in which $\mathbf{W}$ is a dimension reduction matrix obtained by PCA.

By fixing the mean functions during training the predictive mean of (6) can be represented as follows:

$$\boldsymbol{\mu}_i' = \boldsymbol{\mu}_i - m(\mathbf{x}_i) = \boldsymbol{\alpha}_i^\top(\mathbf{M} - m(\mathbf{Z})) = \boldsymbol{\alpha}_i^\top\mathbf{M}'. \tag{10}$$

This implies that the predictive mean of residual $\boldsymbol{\mu}_i'$ can be modeled by zero mean GP with variational parameter $\mathbf{M}'$. Therefore, the method using non-zero mean functions is regarded as a residual network.

3. PRETRAINING FOR DSVI-BASED DGP

In this paper, we propose pretraining that determines initial values of DGP parameters instead of using the residual network. We utilize the fact that the training of single-layer GP regression is much easier than the simultaneous optimization of multiple layers of DGP. This leads to pretraining of each layer individually in advance and using the pretrained parameters as the initial values. However, we do not know the hidden layer values that are necessary for single-layer GP training. Hence, we consider utilizing DNN so that we can determine hidden layer values roughly. The procedure of the proposed method is shown in Fig. 2[1].

First, we define a DGP model structure. Then we replace each partial function $f^\ell(\cdot)$ in (1) by the following perceptron block to obtain a DNN whose structure is similar to the DGP model:

$$\tilde{f}^\ell(\mathbf{h}_i^{\ell-1}) = \mathrm{BN}(\mathbf{V}^\ell\phi(\mathbf{W}^\ell\mathbf{h}_i^{\ell-1} + \mathbf{b}^\ell)) \tag{11}$$

where $\mathbf{V}^\ell$ and $\mathbf{W}^\ell$ are weight matrices and $\mathbf{b}^\ell$ is a bias vector. $\phi(\cdot)$ is an activation function and BN represents batch normalization. The purpose of batch normalization is to normalize output data to zero

[1] Sample codes: https://github.com/hyama5/deepgp_pretraining

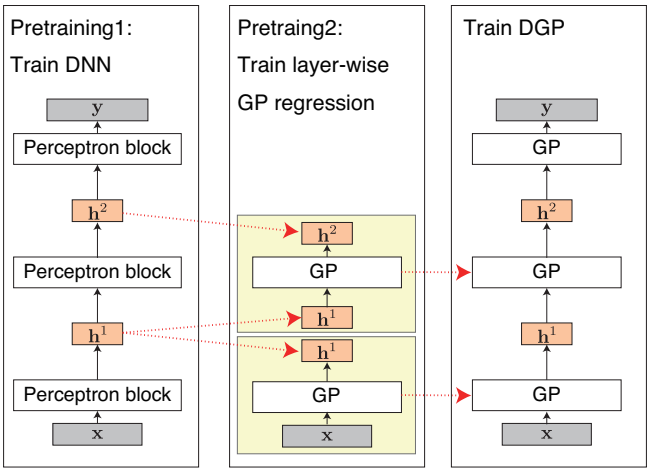


Fig. 2: Outline of the proposed method using two-stage pretraining.

mean in the same way as general GPs. Then, we perform training of the DNN.

Next, we train single-layer GPs based on SVI [6] for each layer excepting the top layer, that is ($\ell = 1 \dots L-1$), using hidden layer values obtained by the DNN. Specifically, the pairs of input and output values ($\mathbf{h}_i^{\ell-1}$, $\mathbf{h}_i^{\ell}$) are used for optimizing the GP parameters ($\mathbf{Z}^{\ell}, \mathbf{M}^{\ell}, \mathbf{S}^{\ell}, \theta^{\ell}$). After the training of the layer-wise GPs, we use these parameters as the initial values of DGP. By the proposed pretraining method, it is expected that the training of DGP becomes similar to the training of single-layer GP of the top L-th layer, which is easy to train. The gradients of parameters of other layers are expected to become small because they are already trained sufficiently.

4. EXPERIMENTS

4.1. Experimental conditions

We applied the proposed training method to statistical parametric speech synthesis (SPSS) task and evaluated training performances. We modeled the relationship between frame-level context vectors as input and frame-level acoustic feature vectors as output in accordance with a traditional DNN-based SPSS [14]. We used English and Japanese speech databases, which were CMU ARCTIC [15] and XIMERA [16], respectively. For English, 1000 sentences (596961 frames, approx. 49 min.) of a female speaker SLT were used as a training data set and development and test sets consisted of individual 66 sentences, respectively. For Japanese, 1533 sentences (1389352 frames, approx. 116 min.) of a female speaker F009 were used as a training data set and development and test sets consisted of individual 60 sentences, respectively.

The context vectors included linguistic features, such as phoneme, stress, and accent, and positional information of the target frame. The linguistic features consisted of not only one-hot encoding of linguistic information such as phoneme entity but also binary features obtained by questions about linguistic features such as *"Is the previous phoneme a vowel?"* The dimensions of input feature vectors of English and Japanese speech were 721 and 574, respectively.

The acoustic feature vector was constructed using STRAIGHT [17], a widely-used analysis-synthesis toolkit. We extracted fundamental frequency f_{o}, spectral envelope, and aperiodicity every 5 ms from the speech signal at a sampling rate of 16 kHz, and obtained 0-39th mel-cepstrum, log f_{o}, and 5-band aperiodicity. We used a 139-dimensional vector by cascading the obtained features and their Δ, Δ^2, and a voiced/unvoiced flag.

We used two kernel functions: not only widely-used RBF kernel but also ArcCos kernel [18]. RBF kernel is defined as

$$k_{\mathrm{RBF}}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{x}')^{\top}\mathbf{\Lambda}^{-1}(\mathbf{x}-\mathbf{x}')\right) \tag{12}$$

$$\mathbf{\Lambda} = \mathrm{diag}[l_1^2, \dots, l_D^2]. \tag{13}$$

The length-scale parameters $(l_1, \dots, l_D)$ were used for automatic relevance determination (ARD), which tunes the importance of input dimensions. ArcCos kernel is a kernel function derived from a neural network with an infinite number of hidden units and the ReLU activation function. One-layer ArcCos kernel with normalization is defined by the following equations.

$$k_0(\mathbf{x}, \mathbf{x}') = \sigma_{b0}^2 + \sigma_{w0}^2 \mathbf{x}^{\top}\mathbf{\Lambda}\mathbf{x}' \tag{14}$$

$$k_1(\mathbf{x}, \mathbf{x}') = \sigma_{b1}^2 + \sigma_{w1}^2 \sqrt{k_0(\mathbf{x}, \mathbf{x})}\sqrt{k_0(\mathbf{x}', \mathbf{x}')} \cdot (\sin\theta + (\pi - \theta)\cos\theta)) \tag{15}$$

$$\theta = \cos^{-1}\frac{k_0(\mathbf{x}, \mathbf{x}')}{\sqrt{k_0(\mathbf{x}, \mathbf{x})}\sqrt{k_0(\mathbf{x}', \mathbf{x}')}} \tag{16}$$

$$k_{\mathrm{ArcCos}}(\mathbf{x}, \mathbf{x}') = \frac{k_1(\mathbf{x}, \mathbf{x}')}{\sqrt{k_1(\mathbf{x}, \mathbf{x})}\sqrt{k_1(\mathbf{x}', \mathbf{x}')}}. \tag{17}$$

We imposed kernel normalization of (17) to restrict the range of kernel outputs for stable computation. The initial values of ARD coefficients $\{l_d\}_{d=1}^{D}$ of RBF kernel were 32, and those of ArcCos kernel were $\sqrt{1/D}$ where D is the dimensionality of input $\mathbf{x}$. The parameters of ARD kernel, σ_{b0}^2, σ_{w0}^2, σ_{b1}^2, and σ_{w1}^2, were all initialized to one.

The dimensionality of the hidden layers and the number of inducing points were 32 and 1024, respectively. The settings of pretraining are shown as follows:

- Training of DNN: The number of hidden units in the perceptron blocks was 1024 and ReLU activation function was used. We performed 20% dropout at the hidden layers within the perceptron blocks as regularization. The DNN parameters were initialized by He normal initializer. Adam optimization with learning rate 0.001 was employed and we adopted early stopping using the development set.
- Layer-wise GP: The inducing input parameters $\mathbf{Z}^{\ell}$ were initialized by K-means clustering using the hidden layer values. The parameters of variational distributions of inducing outputs, $(\mathbf{M}^{\ell}, \mathbf{S}^{\ell})$, were initialized as zero mean and unit variance. Adam optimization with learning rate 0.01 was employed. The number of training epochs was fixed at 1 or 10.

The minibatch size was set to 1024 frames through the pretraining and DGP training. Adam optimization with learning rate 0.01 was employed for the DGP training.

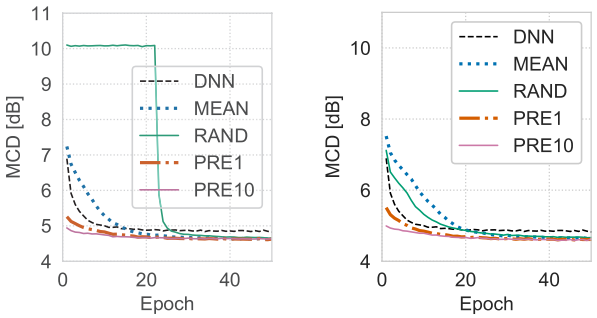
We compared five methods in the performance evaluation: DNN, MEAN, RAND, PRE1, and PRE10.

DNN represents the case that we predicted speech parameters from the DNN model used in the pretraining.

MEAN is the conventional method using DGP with non-zero mean functions.

Other methods, RAND, PRE1, and PRE10, were DGPs with zero mean functions.

RAND determines the initial values of inducing inputs and outputs randomly instead of pretraining.



(a) ARCTIC, ArcCos kernel (b) ARCTIC, RBF kernel

(c) XIMERA, ArcCos kernel (d) XIMERA, RBF kernel

Fig. 3: Spectral distortions as a function of training epochs using 6-layer DGPs.

PRE1 performed the proposed pretraining method with only one-epoch optimization in the layer-wise GP training.

PRE10 performed the proposed pretraining method with ten-epoch optimization in the layer-wise GP training.

4.2. Results

We evaluated the performance of SPSS using spectral distortions measured by mel-cepstral distance (MCD) [19] between original and generated speech samples. The DGP model was feed-forward type one composed of six layers. Figure 3 shows the MCDs as a function of training epochs. It can be seen that the proposed pretraining methods, PRE1 and PRE10, yielded smaller distortion than the conventional method MEAN in early epochs, and the distortions after 50 epochs became close. This implies that the residual structure of MEAN was not always necessary. In the case of RAND with ArcCos kernel, the distortions hardly changed during early epochs. Once the distortions degraded, they converged to similar values to the proposed methods. Moreover, the distortions of DNN were larger than the proposed methods.

To evaluate the effect of depth on the proposed method, we calculated the spectral distortions as a function of the number of layers using the ARCTIC database. Table 1 shows the result of MCDs of RAND and PRE10 with ArcCos kernel calculated after 50 epochs training. It is seen from the table that the performance of 1-layer GPs was insufficient for this dataset. The distortion became smallest at the 4-layer PRE10 model and RAND gave similar results to PRE10 when the number of layers was small. However, the methods of RAND with more than 6 layers yielded much larger MCDs than the others. These cases were because of bad local maximum of ELBO in which DGPs predicted zero mean distributions almost everywhere. In contrast, we see that the training of PRE10 was successful even if the DGP got deeper.

Table 1: MCDs [dB] as a function of the number of layers using ARCTIC database and ArcCos kernel.

# of layers	1	2	3	4	5
RAND	5.11	4.72	4.65	4.65	4.65
PRE10	5.08	4.70	4.63	4.59	4.62
# of layers	6	7	8	9	10
RAND	4.65	10.08	10.08	10.08	10.08
PRE10	4.63	4.60	4.63	4.65	4.62

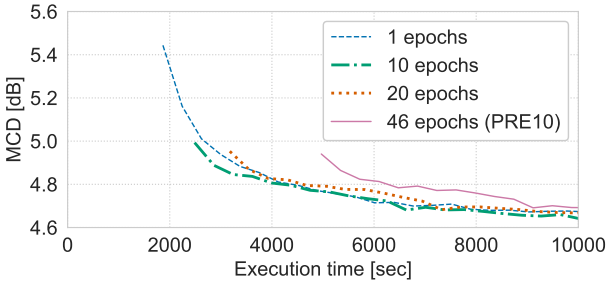


Fig. 4: MCDs as a function of computation time including pretraining using multiple numbers of DNN training epochs.

One problem of the proposed method is the computational cost of DNN training. To evaluate the cost of pretraining, we reduced DNN training epochs in PRE10 into 1, 10, and 20. In this experiment, we used ARCTIC database, ArcCos kernel, and the six-layer DGP model. Computation time was evaluated using NVIDIA TITAN X GPU, Intel Core i7-7700X CPU, and TensorFlow 1.7 implementation. Figure 4 shows the MCDs as a function of training time including pretraining. It is seen the changes of MCDs were similar among the cases of 1, 10, and 20 epochs. Therefore, it could be sufficient even if DNN training was done with only one epoch.

5. CONCLUSIONS

In this paper, we proposed a training method of DGPs based on DSVI using two-stage pretraining to determine initial parameters of DGPs. In the proposed method, we first train DNN, in which the mapping functions of DGP are replaced by perceptron blocks. Then we train layer-wise GP parameters using the hidden layer values of DNN, and apply the obtained parameters to the initial values of DGPs. The experimental results showed that the proposed method gave smaller distortions at early epochs of training than the conventional method based on non-zero mean function.

Future work will apply the proposed method to a more complicated architecture of DGPs instead of feed-forward type one. For example, since the proposed pretraining method and the residual connection are not exclusive, we should evaluate the performance by combining them. Moreover, we should consider incorporating the proposed method into DGP with convolutional network proposed in [20].

6. REFERENCES

[1] S. J Nowlan and G. E Hinton, "Simplifying neural networks by soft weight-sharing," *Neural computation*, vol. 4, no. 4, pp. 473–493, 1992.

[2] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[3] A. Damianou and N. Lawrence, "Deep Gaussian processes," in *Proc. AISTATS*, 2013, pp. 207–215.

[4] H. Salimbeni and M. Deisenroth, "Doubly stochastic variational inference for deep Gaussian processes," in *Proc. NIPS*, 2017, pp. 4591–4602.

[5] M. D Hoffman, D. M Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.

[6] J. Hensman, N. Fusi, and N. Lawrence, "Gaussian processes for big data," in *Proc. AUAI*, 2013, pp. 282–290.

[7] D. P Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. ICLR*, 2014.

[8] R. M Neal, "Priors for infinite networks," in *Bayesian Learning for Neural Networks*, pp. 29–53. Springer, 1996.

[9] J. Lee, Y. Bahri, R. Novak, S. S Schoenholz, J. Pennington, and J. Sohl-Dickstein, "Deep neural networks as Gaussian processes," in *Proc. ICLR 2018*, 2018.

[10] A. G. de G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani, "Gaussian process behaviour in wide deep neural networks," in *Proc. ICLR*, 2018.

[11] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. ICASSP*, 2013, pp. 7962–7966.

[12] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *The Journal of Machine Learning Research*, vol. 6, pp. 1939–1959, 2005.

[13] J. Hensman, A. G de G Matthews, and Z. Ghahramani, "Scalable variational Gaussian process classification," in *Proc. AISTATS*, 2015, pp. 1648–1656.

[14] H. Zen, K. Tokuda, and A.W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.

[15] J. Kominek and A. W Black, "The CMU ARCTIC speech databases," in *Proc. 5th ISCA Speech Synthesis Workshop*, 2004, pp. 223–224.

[16] H. Kawai, T. Toda, J. Ni, M. Tsuzaki, and K. Tokuda, "XIMERA: A new TTS from ATR based on corpus-based technologies," in *Fifth ISCA Workshop on Speech Synthesis*, 2004.

[17] H. Kawahara, I. Masuda-Katsuse, and A. de Cheveigne, "Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech Communication*, vol. 27, no. 3-4, pp. 187–207, 1999.

[18] Y. Cho and L. K Saul, "Kernel methods for deep learning," in *Proc. NIPS*, 2009, pp. 342–350.

[19] R.F. Kubichek, "Mel-cepstral distance measure for objective speech quality assessment," in *Communications, Computers and Signal Processing, 1993., IEEE Pacific Rim Conference on*, 1993, vol. 1, pp. 125–128.

[20] A. Garriga-Alonso, L. Aitchison, and C. E. Rasmussen, "Deep convolutional networks as shallow Gaussian processes," *arXiv preprint arXiv:1808.05587*, 2018.