Generation of F_0 Contour Using Deep Boltzmann Machine and Twin Gaussian Process Hybrid Model for Bengali Language

Sankar Mukherjee, Shyamal Kumar Das Mandal

Centre for Educational Technology
Indian Institute of Technology Kharagpur
sankar1535@gmail.com, sdasmandal@cet.iitkgp.ernet.in

10.21437/Interspeech.2014-237

Abstract

In Text to Speech synthesis system F_0 contour plays an important role in conveying prosodic information but the process of synthesizing F_0 contour from the underlying linguistic information using deep architecture has not been investigated in case of Bengali languages. This paper describes a method for synthesizing F0 contours of Bengali readout speech from the textual features of input text using Deep Boltzmann Machine (DBM) and Twin Gaussian Process (TGP) hybrid model. DBM will capture the high-level linguistic structure of input text and improve the prediction accuracy when plugged into the TGP model. Unlike Gaussian Process (GP) models which only focus on the prediction of a single output (F_0), TGP can generalize across multiple outputs (F_0, delta F_0, delta-delta F_0) by encoding relations between both inputs and outputs with GP priors. The performance of the proposed method is evaluated and compared with other available methods using objective and perceptual listening tests and the results are found to be satisfactory.

Index Terms: F_0 contour Prediction, Deep Boltzmann Machine, Twin Gaussian Process, Bengali Language, Speech synthesis.

1. Introduction

Prosody plays the most important role in generating natural and intelligent speech. Prosody is a collection of supra-segmental features (duration, intonation, co-articulation pattern) which contributes additional information to speech that do not found in text. In prosody, F_0 contour has a significant contribution that is crucial to human speech perception. Knowledge of these parameters are implicit in speech signal i.e. it's hard to capture the rules governing these knowledge. Previous work on F_0 prediction for Indian languages as well as Bengali refers to a syllable based neural network modeling [1] [23], Fujisaki model [24]. But all of them used traditional shallow architectures [1] [21] (i.e. statistical model with few levels of computation units). Use of deep architecture (i.e. statistical model with many layers of computational units) for Bengali language has not been done before.

Discovering high level representations of high dimensional sensory data lies at the core of solving many AI tasks. Theoretical and biological arguments suggest that building such system will require deep architecture with many layers of non-linear information processing. Theoretical results [2] suggest that shallow architectures like support vector machine, kernel regression etc. with only one layer of information processing failed to find good internal representation of input data. Training of such systems requires large amount of labeled data and it's hard to find such data. One way to bypass this problem is to learn a deep generative model for the input data distribution. [3] introduced a fast unsupervised greedy learning algorithm to train these deep generative models which recently had a great impact [4] on machine learning community. This is due to three following characteristics. First, greedy layer wise learning can find a good set of model parameters for many layers. Second, learning algorithm makes the efficient use of large unlabeled data, so model can be pre-trained in unsupervised fashion. Finally, given input there is an efficient way of approximating inference to compute the values of hidden variables in the deepest layer.

The disadvantage of this greedy algorithm is that it is based on a very approximate inference procedure, limited to a single bottom-up pass. It totally ignores the top-down influences on the inference process for which the model can fail to adequately account the uncertainty when interpreting ambiguous sensory inputs. Moreover, as it learns features one layer at time it never readjusts its lower level parameters. This is where comes the Deep Boltzmann machine (DBM) [5] which is a Markov random field with undirected between layer connections. It incorporates a top-down feedback in addition to the usual bottom-up pass. This makes DBM better generalized for uncertainty about ambiguous inputs. Moreover, all layers parameters can be jointly optimized by following the approximate gradient of a variational lower-bound on the likelihood function. DBM have been successfully applied to tasks such as handwritten digit recognition, object recognition [5], spoken query detection [6], phone recognition [7] and multi modal learning [8]. In this work, DBM has been applied to find the high-level linguistic structure from the input text features. After training the DBM with large amount of text, the weights of the DBM is used to initialize a Deep Neural Network (DNN). The output of DNN is the state-level F_0 and its 1^{st} order (delta) and 2^{nd} order (delta-delta) derivatives. DNN is fine-tuned by a set of labeled dataset. The last hidden layer of DNN is considered as the high-level linguistic representation of the input text features. The parameters of the last layer are used as the input to a non-parametric or exemplar-based model.

Non-parametric or exemplar-based model can reproduce some details of data that any parametric model might not be able to reproduce or requires more number of parameters in order to do so. Gaussian Process [10] is a non-parametric model which focuses on the prediction of a single output. For multiple outputs generalizations is derived by training independent models for each one. If the outputs are independent and identically distributed then GP is fine. But if the outputs are correlated (such as F_0, delta and delta-delta) then separate independent GP models fails to capture them [11]. This is where Twin Gaussian Process (TGP) comes in. It is an extension of GP. TGP overcomes this problem by specifying a covariance function over outputs which makes covariance

matrix sensitive to correlations between outputs. TGP predicts the continuous output by forcing input covariance function to be similar to output covariance function. Moreover, it is desired that (1) input-output training pairs whose inputs are far away from the test input should make small contributions to prediction and (2) training input-output pairs whose inputs are near the test input but the corresponding outputs are far away from the test output should again make small contributions to estimates. This objective reflects the assumption that similar inputs should produce similar outputs. TGP has been successfully applied to stock market [12], binary classification [13] and human pose estimation [14].

Recently the hybrid structure [15] of DNN and GP has been applied to model the F_0 contour. Deep Belief Network (DBN) is used as the features extractor from the input text and Gaussian Process (GP) regression model is trained with the learned DBN features which ultimately lead to prediction of F_0 values. Our work is inspired by this but different in the sense that –

1) DBM accounts for the top-down feedback which DBN completely ignores. Also, perhaps more importantly, parameters of all layers can be optimized jointly which leads to learning of a better generative model.
2) In the work of [15] three independent GP models have been used for F_0, delta and delta-delta modeling. Theoretical results [11] suggest that this would fail to account for the output correlation among F_0, delta and delta-delta. TGP captures this correlation by placing GP priors on input as well as output and minimizing Kullback-Leibler divergence between them.

Focus of this paper is to generate F_0 contour where exemplars are state observation of a phoneme with an aim of developing a more natural Bengali TTS system. The rest of the paper is organized as follows. Section 2 describes proposed modeling approach. Section 3 describes the experimental setup. Section 4 presents evaluation of our proposed method. Section 5 draws conclusion.

2. Proposed Modeling Approach

Proposed F_0 modeling approach consists of two stages. First, an initial model (DBM-DNN) non-linearly transforms the input features into some high level features and then these features are fed into an exemplar-based model (TGP) as input. The schematic diagram of the proposed model is shown in Figure 1. At first, DBM is trained with the input features and then after training the learned weights of DBM are used to initialize a DNN. This is called pre-training. Figure 1 (a) shows a pre-trained DNN of *n* hidden layer. DNN is fine-tuned by a labeled set of inputs $x_t \in X$ (i.e. textual features) and targets $y_t \in Y$ (state level F_0, delta and delta-delta). After the fine-tuning the n^{th} hidden layer of DNN (i.e. the last hidden layer) $z_t \in Z$ is act as the high level transformed feature of x_t. z_t is paired with respective outputs. The pairs $\{z_t, y_t\}$ act as the training samples for TGP. Training and prediction process of TGP is shown in Figure 1 (b). At TGP training time hyper-parameters φ are learned from an independent validation set. TGP then predicts the state-level targets corresponding to each transformed test input features with the learned hyper-parameters and training samples. Finally, F_0 contour is generated by including the delta and delta-delta sequences using the well-known parameter-generation algorithm in HMM synthesis [16]. Next we review in details the functionalities of DBM and TGP models.

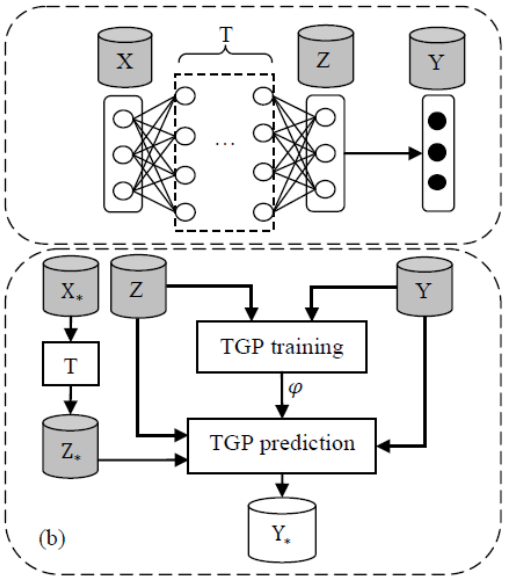

Figure 1: *(a) Deep Neural Network training which result feature transform T and transformed feature Z, (b) Twin Gaussian Process training and prediction.*

2.1. Deep Boltzmann Machine

A Deep Boltzmann Machine [5] is a network of symmetrically coupled stochastic binary units. It contains a set of visible units $\mathbf{v} \in \{0,1\}^D$, and a sequence of layers of hidden units $\mathbf{h}^1 \in \{0,1\}^{F1}, \mathbf{h}^2 \in \{0,1\}^{F2}, \ldots, \mathbf{h}^L \in \{0,1\}^{FL}$. There are connections only between hidden units in adjacent layers and between the visible units and the hidden units in the first hidden layer. The energy of the state $\{v, h\}$ for a three hidden layer (L = 3) Deep Boltzmann Machine is defined as:

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\mathbf{v}^T\mathbf{W}^1\mathbf{h}^1 - \mathbf{h}^{1T}\mathbf{W}^2\mathbf{h}^2 - \mathbf{h}^{2T}\mathbf{W}^3\mathbf{h}^3 \quad (1)$$

Where $\mathbf{h} = \{\mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3\}$ are the set of hidden units, and $\theta = \{\mathbf{W}^1, \mathbf{W}^2, \mathbf{W}^3\}$ are the model parameters which represent visible-hidden and hidden-hidden symmetric interaction terms. The probability that the model assigns to a visible vector $\mathbf{v}$ is:

$$P(\mathbf{v}; \theta) = \frac{1}{Z(\theta)} \sum_h \exp\left(-E(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3; \theta)\right) \quad (2)$$

where $Z(\theta)$ is the partition function. The derivative of the log-likelihood with respect to parameter vector $\mathbf{W}^1$ takes the following form:

$$\frac{\partial \log P(\mathbf{v}; \theta)}{\partial \mathbf{W}^1} = E_{P_{data}}\left[\mathbf{v}\mathbf{h}^{1T}\right] - E_{P_{model}}\left[\mathbf{v}\mathbf{h}^{1T}\right] \quad (3)$$

where $E_{P_{data}}[\cdot]$ denotes an expectation with respect to the completed data distribution $P_{data}(\mathbf{h}, \mathbf{v}; \theta) = P(\mathbf{h}|\mathbf{v}; \theta) P_{data}(\mathbf{v})$, with $P_{data}(\mathbf{v}) = \frac{1}{N}\sum_n \delta(\mathbf{v} - \mathbf{v}_n)$ representing the empirical distribution, and $E_{P_{model}}[\cdot]$ is an expectation with respect to the distribution defined by the model. The derivatives with respect to parameters $\mathbf{W}^2$ and $\mathbf{W}^3$ take similar forms but instead involve the cross products $\mathbf{h}^1\mathbf{h}^{2T}$ and $\mathbf{h}^2\mathbf{h}^{3T}$ respectively. Exact computation of the maximum likelihood learning for this model is intractable. In [5] DBM is learned via stacking of modified Restricted Boltzmann Machines (RBM) which eliminate the double-counting problem when

top-down and bottom-up influences are subsequently combined.

2.2. Twin Gaussian Process Model

2.2.1. Gaussian Process Regression

In a regression problem we want a function f(x) which can make some assumption of the input data. Generally we expect f(x) to be quadratic, cubic or even non-polynomial. In GP f(x) is represented obliquely rather than assuming its characteristics. It is done by letting the 'data' speak more clearly for themselves. Formally, a Gaussian process generates data located throughout some domain such that any finite subset of the range follows a multivariate Gaussian distribution [10]. GP can be completely specified by its input-dependent mean $m(x)$ and covariance function $k(x, x')$. Underlying samples of GP is corrupted by independent, identically distributed Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ and assumed a constant mean to arrive at the model.

$$y = f(x) + \epsilon \tag{4}$$

$$f(X) \sim \mathcal{N}(n, K) \tag{5}$$

$$K_{ij} = k(x_i, x_j) + \sigma_n^2 \delta_{ij} \tag{6}$$

where σ_n is the noise variance, δ the Kronecker-delta function, K is the covariance matrix and $k(x_i, x_j)$ is the covariance or kernel function. The final form of GP is

$$\begin{bmatrix} Y \\ Y_* \end{bmatrix} \sim \mathcal{N}\left(m, \begin{vmatrix} K(X,X) + \sigma^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{vmatrix}\right) \tag{7}$$

The output is calculated using the conditional distribution $p(Y_* \mid Y)$.

2.2.2. Twin Gaussian Process

Unlike GP regression, the Twin Gaussian Process (TGP) [9] uses GP priors on both input as well as output. The input and output are separately modeled by GPs with different covariance function. We adopted Redial Basis Function (RBF) as the covariance function which takes the form:

$$k(x_i, x_j) = \exp\left(-\gamma_r \|x_i - x_j\|^2\right) \tag{8}$$

where $\gamma_r \geq 0$ is the kernel or covariance function width parameter. If the input and output have the Gaussian distribution $\mathcal{N}_X$ and $\mathcal{N}_Y$ then TGP makes predictions by minimizing Kullback-Leibler (D_{KL}) divergence.

$$Y_* = \underset{y}{\operatorname{argmin}}[D_{KL}(\mathcal{N}_Y || \mathcal{N}_X)] \tag{9}$$

where,

$$D_{KL}(\mathcal{N}_Y || \mathcal{N}_X) = -\frac{N}{2} - \frac{1}{2}\log|K_{Y \cup Y_*}| + \frac{1}{2}T_r\{K_{Y \cup Y_*} K_{X \cup X_*}^{-1}\} + \frac{1}{2}\log|K_{X \cup X_*}| \tag{10}$$

N is the number of samples. The input and output covariance matrix is defined as

$$K_{X \cup X_*} = \begin{bmatrix} K(X,X) & K(X_*,X) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \tag{11}$$

$$K_{Y \cup Y_*} = \begin{bmatrix} K(Y,Y) & K(Y_*,Y) \\ K(Y_*,Y) & K(Y_*,Y_*) \end{bmatrix} \tag{12}$$

Equation 10 is solved using a second order, BFGS quasi-Newton optimizer with cubic polynomial line search for optimal step size selection [9]. By setting the input kernel width parameter (γ_x) large which in turn makes the input covariance matrix close to zero we can limit the impact of the training pairs that have inputs dissimilar to the test input. On the other hand if we set the output kernel width parameter (γ_y) large we can downgrade the impact of training data with similar inputs but dissimilar outputs. In this perspective, TGP goes beyond GP regression where similar inputs but dissimilar outputs negatively impact in the estimation. The hyper-parameters ($\varphi = [\gamma_x, \gamma_y, \sigma_n]$) of TGP are optimized via a cross validation set.

3. Experimental Setup

CRBLP speech corpus [17] is employed here for the whole experiment which has one male voice of age 27. Sampling frequency of the corpus is 44100 Hz. HTK toolkit [18] is used to phonetically align the corpus with 5-state HMMs. Each observation corresponds to roughly 1/5 of the phonemes i.e. for each phoneme we have 5 states. From the corpus phonetically balanced 6000 sentences are chosen for training and 2000 sentences for cross validation purpose. After HTK segmentation phonetic alignment of 1000 sentences are corrected manually. Out of these 1000 sentences, 500 are chosen for DNN fine-tuning, 300 for TGP and GP training and 200 for testing the proposed system. F_0 values are extracted using STRAIGHT [19] with 10-ms frame length and 5-ms shift. Delta and delta-delta sequences are computed using the equations $\delta_1[y(n)] = y(n+1) - y(n-1)$ and $\delta_2[y(n)] = 0.5y(n+1) - 2y(n) + 0.5y(n-1)$. Values of these three sequences are scaled [0.1 - 0.99] with zero mean. Each row of these three sequences is paired with the input features to produce the training data for neural network modeling. Input features are listed in Table 1 which is extracted from the input text and they are re-encoded using One-of-N codes which resulted of 346 dimensions (binary). The phoneme set consists of 30 consonants and 16 vowels. Max syllable length 6, max 10 syllable words and max 6 phonemes in a syllable is considered and which is sufficient for Bengali language.

Table 1 *Input features*

Feature Name	No. of Features
Phoneme identity [Previous/Current/Next]	46 * 3 = [138]
Place & Manner of articulation [Previous/Current/Next]	42 * 3 = [126]
No. of phonemes in syllable [Previous/Current/Next]	6 * 3 = [18]
No. of syllable in current word	[10]
Current phoneme position in syllable [Forward/Backward]	6 * 2 = [12]
Current syllable position in word [Forward/Backward]	10 * 2 = [20]
Vowel position in current syllable [Forward]	[6]
Vowel identity in the current syllable	[16]
Total	**346**

Each DBM layer is pre-trained for 100 epochs as a RBM (see [25] for details) with mini-batch of size 10. Average gradients are computed on the mini-batches and parameters are updated with a learning rate of 0.002 and a momentum of 0.95. With the DBM learned weights DNN is pre-trained. DNN output has 3 dimensions which consist of state-level F_0 and their

delta, delta-delta values. We used mean squared error as the objective function with sparsity target of 0.05 and 0.002 weight decay. Back-propagation training is evaluated using the cross validation set with mini-batches of 1000 states. Loss is measured for the 10 epochs and if loss increased then learning rate is decreased by a factor of two and weights are restored to their value at the beginning of the epoch. Sigmoid activation function is used in this work. Input kernel width parameter $\gamma_x = 2 \times 10^{-5}$ and output kernel width parameter $\gamma_y = 6.08$ with noise variance $\sigma_n = 0.0316$ is used for TGP training.

For comparison a DBN-GP [15] based F_0 prediction system for Bengali language is developed based on the input features as describe in Table 1. A Bengali HTS [22] system is constructed using HTS toolkit [20] which contains 9000 HTK segmented Bengali sentences with 34th order MGC coefficient, 10ms Blackman window, 5ms shift and 0.53 frequency wrapping factor with Global Variance. HTS uses clustering tree with multi-space probability distribution (MSD-HMM) to model the F_0 contour.

The experiment is carried out on DELL precision T3600 workstation which is a 6 core computer with a CPU clock speed of 3.2 GHz, 12MB of L3 cache and 64GB DDR3 RAM. The training also used an NVIDIA Quadro 4000 general purpose graphical processing unit (GPGPU) for matrix multiplication.

4. Evaluation

4.1. Objective Evaluation and Model Selection

Several network architectures which range from 100 to 500 hidden units with 3 hidden layers have been trained. Out of them four models are selected according to the Mean Square Error (MSE) computed on the test set during the DNN training. System configurations of them are listed in Table 2.

Table 2 *DBM-DNN configuration*

System Name	Configuration
A	200-200-200
B	350-350-450
C	350-300-450
D	150-150-150

Two types of objective metrics are used for evaluation i.e. MSE and cross-correlation (XCORR). Using these four DNN configurations four DBM-TGP and four DBN-GP system have been built. Objective metrics have been calculated on the test set between the phoneme state-level values. Results are shown in Figure 2. It can be observed from the Figure 2 that system *'B'* has the best performance. System *B* is further compared with MSD-HMM based on the objective metrics and the results are reported in Table 3.

4.2. Subjective Evaluation

DBM-DNN-TGP model is evaluated against the MSD-HMM and DBN-DNN-GP model based on AXB preference test. Two sets of AXB preference test is done one is DBM-DNN-TGP with MSD-HMM and other is DBM-DNN-TGP with DBN-DNN-GP. In this experiment, the 50 test sentences are synthesized using DBM-DNN-TGP, MSD-HMM and DBN-DNN-GP. MLSA[21] filter has been used to synthesized the sentences with spectrum and duration information taken

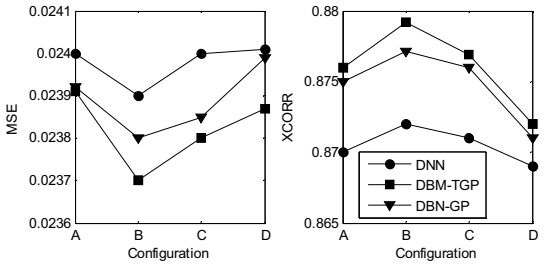

Figure 2: *XCORR and MSE of the predicted F_0 calculated on test set. Variance of the test set is 0.0238.*

Table 3 *Objective evaluation on the test set.*

	MSE	XCORR
MSD-HMM	0.597	0.7512
DBM-DNN	0.239	0.8710
DBN-DNN-GP	0.238	0.8771
DBM-DNN-TGP	0.237	0.8792

from Bengali-HTS system. Ten subjects (7 male, 3 female) are selected. All of them are native speakers of Bengali and none of them are speech experts. Participants are asked to choose their preferred one or neither and they are allowed to listen as many times as they needed before giving their answers. 500 responses are recorded for both tests. Table 4 and Table 5 represent the result of the preference test between DBM-DNN-TGP vs MSD-HMM and DBM-DNN-TGP vs DBN-DNN-GP where NULL indicates no preference.

Table 4 *Results of preference test between* DBM-DNN-TGP and MSD-HMM

	MSD-HMM	NULL	DBM-DNN-TGP
Counts	188	88	224
%	37.6	17.6	44.8

Table 5 *Results of preference test between* DBM-DNN-TGP and DBN-DNN-GP

	DBN-DNN-GP	NULL	DBM-DNN-TGP
Counts	198	84	218
%	39.6	16.8	43.6

The perceptual differences between the pair of systems are statistically evaluated by Wilcoxon signed-rank test with numerically re-encoding the answers as [-1; 0; +1]. In first case $p < 0.001$ indicates the significance of our proposed method. In second case the $p < 0.01$ level indicate the preference toward the proposed system.

5. Conclusions

The proposed method harnesses the combined power of both parametric and non-parametric techniques. For the parametric component Deep Boltzmann Machine (DBM) has been employed which act as a high level feature extractor from the input text features. These extracted features are used as an input to the non-parametric exemplar-based, Twin Gaussian Process (TGP) regression model. From the objective and subjective evaluation it is found that proposed model has more preference than common MSD-HMM based model.

6. References

[1] Sreenivasa Rao, K., and Bayya Yegnanarayana. "Intonation modeling for Indian languages." Computer Speech & Language 23.2 (2009): 240-256.

[2] Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards AI. Large-Scale Kernel Machines. MIT Press.

[3] Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. Neural Computation, 18, 1527– 1554.

[4] Yu, Dong, and Li Deng. "Deep learning and its applications to signal and information processing." Signal Processing Magazine, IEEE 28.1 (2011): 145-154.

[5] Salakhutdinov, R., Hinton, G.: "Deep Boltzmann machines". In: Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS 2009). (2009) 448–455

[6] Yaodong Zhang, Ruslan Salakhutdinov, Hung-An Chang, and James R. Glass, "Resource configurable spoken query detection using deep Boltzmann machines.," in ICASSP, 2012, pp. 5161–5164.

[7] You, Zhao, Xiaorui Wang, and Bo Xu. "Investigation of deep Boltzmann machines for phone recognition." Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013.

[8] Srivastava, N., Salakhutdinov, R.: Multimodal learning with deep boltzmann machines. In Bartlett, P., Pereira, F., Burges, C., Bottou, L., Weinberger, K., eds.: Advances in Neural Information Processing Systems 25. (2012) 2231–2239.

[9] Bo, L., Sminchisescu, C.: Twin gaussian processes for structured prediction. Int. J. Comput. Vision 87 (2010) 28–52

[10] C.E. Rasmussen and K. I. Williams. Gaussian processes for machine learning. 2006.

[11] Weston, J., Chapelle, O., Elisseeff, A., Scholkopf, B., & Vapnik, V. (2002). Kernel dependency estimation. In Advances in neural information processing systems, 2002.

[12] M. Mojadadi, M. Nabi, and S. Khadivi, Stock Market Prediction using Twin Gaussian Process Regression. International Journal for Advances in Computer Research (JACR) preprint 2011.

[13] He, Jianjun, Hong Gu, and Shaorui Jiang. "Twin Gaussian processes for binary classification." Data Mining (ICDM), 2011 IEEE 11th International Conference on. IEEE, 2011.

[14] Han, Hong, et al. "Discriminative Human Pose Estimation Based on the Bandelet2 Image Descriptor." Image and Graphics (ICIG), 2011 Sixth International Conference on. IEEE, 2011.

[15] R. Fernandez, A. Rendel, B. Ramabhadran, and R. Hoory, "F_0 contour prediction with a deep belief network-Gaussian process hybrid model," in Proc. ICASSP, 2013, pp. 6885–6889.

[16] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM based speech synthesis," in ICASSP, 2000, pp. 1315–1318.

[17] S. D. A. Alam Firoj, Habib S. M. Murtoza and K. Mumit, "Development of annotated bangla speech corpora, spoken language technologies for under-resourced language," in Proceedings of (SLTU10), vol. 1. Penang, Malasia, 2010, pp. 35 – 41.

[18] S. J. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland, The HTK Book Version 3.4. Cambridge University Press, 2006.

[19] H. Kawahara, I. Masuda-Katsuse, and A. de Cheveigné, "Restructuring speech representations using a pitch adaptive time-frequency-based F_0 extraction: Possible role of a repetitive structure in sounds," Speech Commun., vol. 27, no. 3–4, pp. 187–207, 1999.

[20] "HMM-based speech synthesis system (HTS)," [Online] Available: http://hts.sp.nitech.ac.jp/.

[21] Yamagishi, Junichi, et al. "Speaker-Independent HMM-based Speech Synthesis System: HTS-2007 System for the Blizzard Challenge 2007." (2007).

[22] S. Mukherjee and S. K. D. Mandal, "A Bengali HMM based speech synthesis system," in International Conference on Speech Database and Assessments (Oriental COCOSDA), 2012, pp. 225–259.

[23] Reddy, V. R., and K. S. Rao. "Intonation modeling using FFNN for syllable based Bengali text to speech synthesis." Computer and Communication Technology (ICCCT), 2011 2nd International Conference on. IEEE, 2011.

[24] Mandal, SK Das, et al. "Analysis and Synthesis of F0 Contours for Bangla Readout Speech." Proc. of Oriental COCOSDA 2010 .

[25] Hinton, Geoffrey. "A practical guide to training restricted Boltzmann machines."Momentum 9.1 (2010): 926.