F0 CONTOUR PREDICTION WITH A DEEP BELIEF NETWORK-GAUSSIAN PROCESS HYBRID MODEL

Raul Fernandez [1], *Asaf Rendel* [2], *Bhuvana Ramabhadran* [1], *Ron Hoory* [2]

[1] IBM TJ Watson Research Center, Yorktown Heights, NY 10598 – USA
[2] IBM Haifa Research Lab, Haifa – Israel

ABSTRACT

In this work we look at using non-parametric, exemplar-based regression for the prediction of prosodic contour targets from textual features in a speech synthesis system. We investigate the performance of Gaussian Process regression on this task when the covariance kernel operates on a variety of input feature spaces. In particular, we consider non-linear features extracted via Deep Belief Networks. We motivate the use of this hybrid model by considering the initial deep-layer model as a feature extractor that can summarize high-level structure from the raw inputs to improve the regression of an exemplar-based model in the second part of the approach. By looking at both objective metrics and perceptual listening tests, we evaluate these proposals against each other, and against the standard clustering-tree techniques implemented in parametric synthesis for the prediction of prosodic targets.

Index Terms— speech synthesis, intonation generation, neural networks, Gaussian processes

1. INTRODUCTION

Data-driven speech synthesis systems can be broadly contrasted in terms of the ways in which they make use of the data during the learning and run-time stages of the process to infer and predict prosodic properties of the acoustic waveform. In the case of unit-selection systems, typical architectures exploit prosodic models to generate desired target values to use as a component of the cost function driving the unit search. Retrieved units can then be post-processed to closely match these targets, or be minimally post-processed to retain most of the natural prosody inherent in the units, an approach that often lends unit-selection systems their prosodic naturalness. Toward the other end of a continuum, we have fully parametric, model-based systems which use the training data only during the learning stage to adapt the model parameters, and then use the models at run-time to generate prosodic parameters that can be used directly in the speech-generation stage. Since the data plays no further role after training, these systems incur a small footprint size, which is one of their desirable properties. In this work we are interested in exploring a particular intermediate point where, while still assuming an underlying model-based architecture, we consider non-parametric, exemplar-based models that can exploit the data, adaptively, at run time based on their relevance to the run-time cases. We focus on the task of F0 generation where exemplars are state-sized observations, a temporal granularity that could still allow keeping the footprint of the system in check for many applications. We review the components of this model in section 2 and related work in 3, and present objective and perceptual evaluations of the system in 4.

2. MODELING APPROACH

The approach consists of a two-stage model where inputs are first non-linearly transformed using a neural network and then used as the input to a Gaussian-process regression model (see Fig. 1). The motivation behind such a proposal aims to combine the relative strengths of both a parametric and a non-parametric model in a regression setting: an initial model is used to extract high-level structure in the form of a vector of parameters, and then an exemplar-based model can better exploit the structured representation over that afforded by the raw inputs in order to improve the quality of the regression.

On one hand, neural networks (NNs), particularly when structured with several deep layers, have been documented to be able to extract high-level structure from raw inputs, that can then be used directly in a high-level classification or regression task (such as digit recognition [1] or acoustic modeling for speech recognition [2]). Being parametric in nature, however, such models can be adversely affected by a large input dimensionality in the case of insufficient training data. On the other hand, non-parametric models (such as the Gaussian processes (GPs) with constant number of dimension-independent hyperparameters we consider here) can be more robust to a large dimensionality in the presence of limited data, and, being exemplar-based, can reproduce some of the detail in the data that a parametric model might not be able to reproduce or require a large number of parameters in order to do so.

The cascade and use of these individual models reflect these observations. An NN is first trained to reproduce the targets $\mathbf{y}_t$ from a given set of inputs $\mathbf{x}_t$ (Fig. 1-1). In addition to serving as a full predictive model $\mathcal{X} \rightarrow \mathcal{Y}$, such a structure can also be used as a (non-linear) feature extractor by tapping into the ouputs after the inputs have been transformed by the n^{th} layer. These n^{th}-level features $\mathbf{z}_t$ can then be paired with the respective outputs and treated as the exemplars by the GP. This approach further allows to automatically incorporate dimensionality reduction by simply constraining the number of nodes in the n^{th} layer (i.e., by imposing a bottleneck structure on the NN). To keep the number of parameters in check, the NN is trained in a "context-independent" manner; that is, the training tokens consist of the pairs $\{\mathbf{x}_t, \mathbf{y}_t\}$. Context, however, can be easily incorporated in the GP regression by considering the augmented exemplars $\{[\mathbf{z}_{t-M}^T, \cdots, \mathbf{z}_t^T, \cdots, \mathbf{z}_{t+M}^T]^T, \mathbf{y}_t\}$ when training the hyperparameters θ of the GP model (Fig. 1-2).

Since in this work we consider only single-output Gaussian processes, 3 independent GP models are learned in the training phase, one associated with each stream k of the target F0 vector generally used in parametric synthesis: log F0, as well as the delta and delta-delta sequences. These sequences represent state-level mean statistics of the respective frame-level curves, where the state segmentation has been previously generated by forced alignment, with 3-state hidden Markov models (HMM), between the acoustic wave-

forms and the phonetic transcripts (i.e., the observations correspond to roughly $1/3$ of a phone). At run time (Fig. 1-3), all that is needed to generate the predictions from the inputs are the transform associated with the n^{th} layer ($T = T_1 \circ T_2 \cdots \circ T_n$, where $T_j = g(W_j \mathbf{x}_j)$, W_j are the weights of the j^{th} layer, $\mathbf{x}_j$ the inputs arriving at that layer, and $g(\cdot)$ is the logistic function in our implementation), the GP model's hyperparameters $\theta^{(k)}$ for each stream k, and the exemplars in the training database. We next review in more detail the training of the NN, and provide a brief overview of GP regression and training.

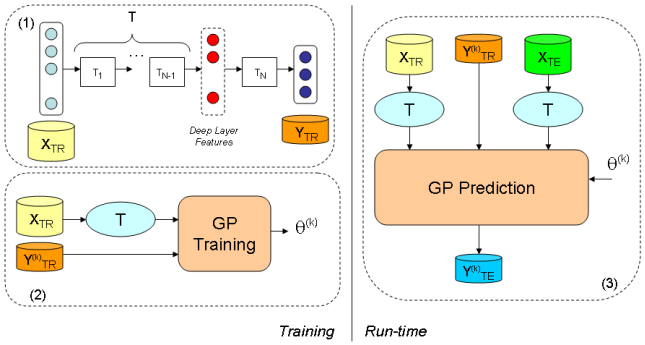

Fig. 1. Model describing (1) neural network training, resulting in feature transform T (2) Gaussian Process training (3) prediction of the hybrid model.

2.1. Neural Networks

This work makes use of recent techniques for neural-network (NN) training in which a two-pass approach combines an underlying generative model, trained in an unsupervised manner, with standard supervised techniques for training the discriminative model. In the first, or pretraining, phase of this procedure, all but the output layer of a deterministic, feed-forward NN are replaced with an undirected, probabilistic, generative model. Such a structure, known as a Deep Belief Network (DBN), is first trained in an unsupervised manner (i.e., ignoring the network targets). After the weights of this structure have been learned, they are used to initialize the feed-forward structure which, with the output targets now restored, is further trained using back-propagation to minimize the loss function on the output layer (in our case, mean squared error) between the targets and the predictions.

The training of the DBN is performed layer-wise by learning the weights between each pair of layers at a time. Each undirected bipartite graph structure resulting from this decomposition is known as a Restricted Boltzmann Machine and can be trained fairly efficiently using the gradient-descent with Contrastive Divergence algoritmh proposed in [1] and [3]. Given that all inputs to the model have been encoded as boolean indicators, the RBM only contains connections between Bernoulli-distributed variables.

For the reported experiments, both pretraining and back-propagation are trained using stochastic gradient descent, with mini-batches of 100 states. The back-propagation training is guided by the performance on an independent development set. After four full passes through the data, loss is measured on the development set. If the loss has increased, the learning rate is reduced by a factor of two and the weights are restored to their value at the beginning of the epoch. The learning stops after several reductions of the learning rate.

2.2. Gaussian Processes

A Gaussian Process is a collection of random variables, any finite collection of which have a jointly Gaussian distribution, and which can be completely specified by its input-dependent mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$ [4]. We allow for the underlying samples of the GP to be corrupted by independent, indentically distributed Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$, and assume a constant mean to arrive at the model:

$$\begin{aligned} y &= f(\mathbf{x}) + \epsilon && (1)\\ f(\mathbf{X}) &\sim \mathcal{N}(\mathbf{m}, K) && (2)\\ K_{ij} &= k(\mathbf{x}_i, \mathbf{x}_j) + \sigma_n^2 \delta_{ij} && (3) \end{aligned}$$

The jointly Gaussian definition above also implies a conditional Gaussian over a subset of such variables by the marginalization property of Gaussian distributions. Considering two sets of variables corresponding to the observations $\mathbf{y}_{TR}$ in a set of training cases and the underlying function values of a set of test cases $\mathbf{f}_{TE}$, then it still holds that:

$$\begin{bmatrix} \mathbf{y}_{TR} \\ \mathbf{f}_{TE} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{m}, \begin{bmatrix} K(\mathbf{X}_{TR}, \mathbf{X}_{TR}) + \sigma_n^2 I & K(\mathbf{X}_{TR}, \mathbf{X}_{TE}) \\ K(\mathbf{X}_{TE}, \mathbf{X}_{TR}) & K(\mathbf{X}_{TE}, \mathbf{X}_{TE}) \end{bmatrix}\right) \quad (4)$$

from which, after marginalization, the conditional distribution $p(\mathbf{f}_{TE}|\mathbf{y}_{TR})$ can be shown to follow

$$\begin{aligned} \mathbf{f}_{TE}|\mathbf{y}_{TR} &\sim \mathcal{N}(\bar{\mathbf{f}}_{TE}, \mathrm{cov}(\mathbf{f}_{TE})) && (5)\\ \bar{\mathbf{f}}_{TE} &= \mathbf{m} + \{K(\mathbf{X}_{TE}, \mathbf{X}_{TR}) \times \\ &\quad [K(\mathbf{X}_{TR}, \mathbf{X}_{TR}) + \sigma_n^2 I]^{-1}(\mathbf{y}_{TR} - \mathbf{m})\} && (6)\\ \mathrm{cov}(\mathbf{f}_{TE}) &= K(\mathbf{X}_{TE}, \mathbf{X}_{TE}) - \\ &\quad \{K(\mathbf{X}_{TE}, \mathbf{X}_{TR})[K(\mathbf{X}_{TR}, \mathbf{X}_{TR}) + \sigma_n^2 I]^{-1} \times \\ &\quad K(\mathbf{X}_{TR}, \mathbf{X}_{TE})\} && (7) \end{aligned}$$

Since the conditional is also a Gaussian, Eqs. 6 and 7 provide the MAP estimate and error bars for a given set of test cases $\mathbf{X}_{TE}$ based on the exemplar pairs $\{\mathbf{X}_{TR}, \mathbf{y}_{TR}\}$. The computation of these equations involves evaluating (and inverting) matrices whose entries are determined by the choice of correlation function $k(\mathbf{x}_i, \mathbf{x}_j)$. To ensure positive definiteness, $k(\cdot, \cdot)$ must be a valid kernel (i.e., it can be represented as an inner product of functions). Of the various commonly studied functions in the literature which satisfy this condition, we have adopted a simple squared exponential covariance function[1]:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-h \frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma_k^2}\right), \quad (8)$$

where h and σ_k, in addition to σ_n (the noise-model variance), are hyper-parameters of the GP ($\theta = [h, \sigma_k, \sigma_n]^T$). As variations in this hyper-parameter set can lead to very different output processes, it is important for accurate prediction to properly estimate them from the training set. This can be done by maximizing the marginal likelihood of the training observations (the evidence function) given by:

$$\begin{aligned} \log p(\mathbf{y}_{TR}|\mathbf{X}_{TR}, \theta) &= -\frac{1}{2}(\mathbf{y}_{TR} - \mathbf{m})^T K_{TR}^{-1}(\mathbf{y}_{TR} - \mathbf{m}) \\ &\quad -\frac{1}{2}\log|K_{TR}| - \frac{n}{2}\log(2\pi) && (9)\\ K_{TR} &\doteq K(\mathbf{X}_{TR}, \mathbf{X}_{TR}) + \sigma_n^2 I && (10) \end{aligned}$$

[1] In fact, earlier experiments with a variety of kernel types yielded very similar results.

It should be clear that, though we know the test inputs are arranged in a pre-defined sequence at run-time, the collection of random variables that define this model do not have a temporal arrangment among them (i.e., it is not a sequential model). There is no implicit notion of time beyond what is addressed by the augmented *input* context. The training exemplars contribute to the prediction based on their correlation to the test exemplars, as measured in input space (i.e., the text-based features, not time). The dynamic evolution is addressed by including the delta sequences and using these within the well-known parameter-generation algorithm in HMM synthesis [5].

In the experiments reported, gradient descent was used to maximize Eq. 9. Both during training and prediction, the FITC approximation was used to deal with the inversion of the Grammian matrix K_{TR} containing a large number of exemplars [6]. Furthermore, stochastic gradient descent was performed over smaller batches (containing approximate 15000 exemplars), and an independent development set was used to track the quality of the model after every batch iteration and to select the best-performing model.

3. RELATED WORK

Although both NNs and GPs are established techniques in the statistical modeling literature, we believe that their combined use in the hybrid approach we have proposed for predicting intonation from text in a speech synthesis system represents a novel contribution in this area. A variety of NN structures, including multi-layer perceptrons and recurrent networks, were previously investigated on the task of F0 modeling for Japanese TTS in the work of [7]. Besides being a fully parametric approach (i.e., non exemplar based), that work also differs from ours in the specific NN training methodology, and in that a Fujisaki model decomposition of the F0 curve is assumed prior to modeling whereas our observations are the direct state-level sequences. Though the use of DBNs to do unsupervised pre-training of NNs has been investigated in other speech applications, such as excitation modeling [8] and, most notably, speech recognition [2]), their use in prosody modeling represents a novel application.

GPs, perhaps due to their quadratic dependence on the size of the exemplar database, have not been as fully explored in speech applications. They have, however, received some recent attention for voice-conversion applications in the work of [9] where they are integrated into a mixture-of-experts framework, and for speech recognition (i.e., classification not regression) in the work of [10]. In the area of synthesis, the most relevant prior work is that of [11], who looked at using GPs as the building blocks of a dynamical system whose state transitions and state emissions can both be described in terms of GP models. There are several differences between that work and ours worth higlighting: (i) as mentioned, we do not address dynamics in this work during the state-level F0 generation (that is done, rather, during generation of the F0 frame-level curve by taking into account the predicted deltas); (ii) we model each stream (F0 and the delta sequences) independently whereas they use a GP with latent variables (GP-LVM), a model which allows coupling between the streams (we, instead, couple them during the NN training); (iii) being primarily a proof-of-concept, their work offers very limited evaluation (based on 2-sentence resynthesis task), whereas we have used a full database to build a full synthesis system and replaced the standard F0-generation component with the current proposal in the evaluation.

4. EVALUATION

4.1. Data Description

The approach was evaluated using a synthesis training database containing about 3 hours of profesionally recorded speech from a female speaker of North American English. Independent development and test sets of about 20 minutes each were also used for model-selection and final evaluation tasks. The corpus was phonetically aligned using 3-state HMMs [12], and the F0 contours were extracted using a 5-msec. frame rate. First- and second-order delta sequences were computed using the operators $\delta_1[y(n)] = y(n+1) - y(n-1)$ and $\delta_2[y(n)] = 0.5y(n+1) - 2y(n) + 0.5y(n-1)$, and state-level mean statistics for all 3 sequences aggregated using the state-level alignments. The front-end module of a TTS engine was used to extract a set of text-based features typically used for prosody prediction in speech synthesis (e.g., phonetic identity, syllable counts, etc.). Prior to neural-network modeling, all numerical features were quantized, and each resulting categorical feature was re-encoded using One-of-N codes, bringing the final dimensionality of the input (binary) vector to 240. All features values defined above the state level were propagated down to the constituent states (with 3 states per phone), and the resulting state-level predictors paired with the state-level output streams to produce the modeling datasets.

4.2. Objective Evaluation and Model Selection

We explored a variety of NN structures and used the development set to score the different approaches. A series of preliminary tests demonstrated that features extracted from the deepest layer of the network (one layer prior to the target) yielded consistently better results than features extracted from lower layers of processing. This is consistent with our expectations that deeper layers in a NN manage to extract more structure from the input, and all results reported here correspond to features extracted from the third layer of a $240(input) \times 256 \times 256 \times X \times 3(target)$ structure, where $X \in (64, 128, 256)$. The output target, driving the back-propagation, consists of a 3-d vector containing the state-level means of the log F0, as well as the means of the frame-level first and second derivatives (i.e., delta sequences). The inclusion of the delta sequences in the target vector has improved the log F0 prediction metrics of the NN on its own, as well as of the GP model based on the NN-transformed features.

Fig. 2 shows various objective metrics computed using a development set after the NN deep-layer features for these 3 different structures have been modeled with a GP. For comparison, the results of modeling the raw features directly (i.e., the inputs to the NN are directly modeled with the GP, bypassing the NN modeling) are also included. Each point marked on the curves corresponds to adding another left and right state of predictive context to the center state to form the input vector to the GP (recall that the NNs are trained without any context). The x-axis on the plots corresponds to the overall dimensionality after adding context $dim_{GP} = (2 \times L_{ctxt} + 1) \times dim_X$, where $L_{ctxt} = \{0, 1, 2, \cdots\}$ is the context size, and dim_X is the dimensionality of the context-free feature vector (64,128, 256 or, in the case of the raw features, 240). Mean squared error (MSE) and cross-correlation (XCORR) are computed between the state-level GP predictions and the corresponding dev-set targets for any given model. The variance (VAR) and the log likelihood are computed directly from the dev-set predictions (to put these values in context, the observed natural variance of the dev set is 0.0567).

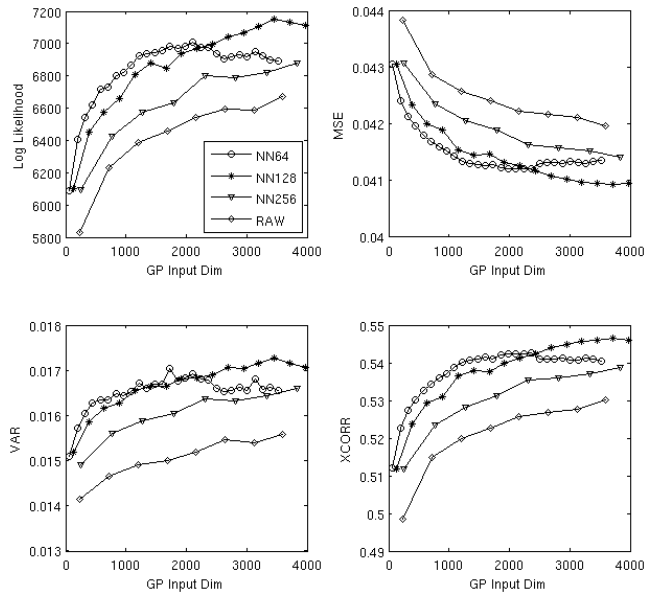

Fig. 2. Development-set metrics from the GP predictions as a function of GP input dimension (which is a function of context size). Models built on different NN structures, and on the raw features, are compared.

We can see that all the metrics improve, for any context size, when a combination of deep-layer features with GPs is used instead of directly modeling the input features with a GP, providing validation for the composite model. In terms of NN structures, we see that the effect of reducing the dimensionality by pinching the deepest layer too much, though marginally better for smaller context sizes, eventually hurts as more slices of context are added. Of the structures considered, the 128-D structure provides a better trade-off and reaches an optimal point around 13 slices of context (corresponding to a GP input dimension of 3456). This is the model that we select for further evaluation.

The metrics on the independent test (not used during training) set are finally reported in Table 1 for the selected model (context of 13), and compared to the model with no context to illustrate the effect of added context. Metrics are also reported for the predictions obtained with the NN structure directly at the output layer, without including the GP model, to illustrate the added benefit of the GP. Finally, all these figures are compared to the baseline decision-tree predictive model that is used in the synthesis engine to generate F0 predictions. This model is a clustering tree with multi-space Gaussian output distributions similar to the models described in [13] and included in the HTS synthesis engine toolkit [14].

	MSE	VAR	XCORR
Tree	.0535	.0156	.3278
NN	.0443	.0149	.4987
NN+GP0	.0443	.0156	.4976
NN+GP13	.0428	.0176	.5224

Table 1. Test set metrics. The full context-augmented model (NN+GP13) is compared to the full no-context model (NN+GP0) as well as to direct NN modeling and baseline (decision tree) model. The natural variance of the test set is 0.0574.

4.3. Subjective Evaluation

The NN(128) + GP(13) model selected in the previous section was evaluated against the baseline decision-tree model using a perceptual listening test and an AXB preference paradigm. Thirty different text inputs were used to generate 30 pairs of samples, each sample in the pair corresponding to a different F0-generation model, and presented to 17 different subjects (11 male and 6 female; 13 native speakers of North American English and 4 proficient in English) in one of 4 distinct randomized arrangements (where both the order of the text inputs and the order of presentation of the pairs were randomized) for a total of 510 responses. Subjects were allowed to listen to each sample in the pair (anonymized as *Sample 1* and *Sample 2*) as many times as needed before indicating whether they preferred sample 1, sample 2, or neither. The distribution of the answers is shown in Table 2. The perceptual differences between the systems were statistically evaluated by numerically re-encoding the answers as $[-1, 0, +1]$ and using Wilcoxon signed-rank test of the null hypothesis that the data comes from a zero-mean distribution. This hypothesis was rejected at the $p < 0.01$ level indicating that the preference toward the proposed system, though modest, is significant. As we can see, subjects indicate a preference toward one of the two systems 64% of the time, and when they do, the majority preference is toward the NN+GP system (58% vs. 42%). Put in a different way, subjects found the proposed system to be the same as or preferable to (i.e., no worse than) the baseline system 73% of the time.

	DTree	NULL	NN+GP
Counts	137	185	188
%	27	36	37

Table 2. Results of perceptual listening test showing preference toward the different F0-generating systems. NULL indicates no preference.

5. CONCLUSIONS

In this work we have presented and reported results on a new statistical approach to predict F0 targets in a text-to-speech system that combines parametric and non-parametric techniques to offer improvements, both in terms of objective metrics and perceptual preference, over the baseline decision-tree F0-prediction models currently used in many standard synthesis systems. The parametric component of the system exploits recent work in Deep Belief Networks when training Neural Networks to extract high-level structure from low-level inputs. These extracted features are then used as inputs to a non-parametric, exemplar-based, Gaussian Process regression model, which, as we have illustrated, shows good ability in handling extended input contexts and improving as a result of it, leading to a reduction in the prediction mean-squared error and an increased variance indicating better expressivity.

6. AWKNOWLEDGMENTS

The authors would like to thank Tara Sainath for useful discussions on DBN training, and Andy Aaron and Larry Sansone for their help running the perceptual tests.

7. REFERENCES

[1] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[2] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, and A. Mohamed, "Making Deep Belief Networks effective for large vocabulary continuous speech recognition," in *Proc. ASRU*, Hawaii, 2011.

[3] G. Hinton, "A practical guide to training Restricted Boltzmann Machines," University of Toronto, Tech. Rep. 2010-003, 1996.

[4] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[5] 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.

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

[7] A. Sakurai, K. Hirose, and N. Minematsu, "Generation of F0 contours using a model-constrained data-driven method," in *ICASSP*, 2001, pp. 817–820.

[8] S. Vishnubhotlan, R. Fernandez, and B. Ramabhadran, "An autoencoder neural-network based low-dimensionality approach to excitation modeling for HMM-based text-to-speech," *ICASSP*, vol. 2, pp. 4614–4617, March 2010.

[9] N. Pilkington, H. Zen, and M. Gales, "Gaussian process experts for voice conversion," in *Interspeech*, 2011, pp. 2761–2764.

[10] H. Park and C. Yoo, "Gaussian process dynamical models for phoneme classification," in *NIPS 2011: Workshop on Bayesian Nonparametrics*, 2011.

[11] G. Henter, M. Frean, and W. Kleijn, "Gaussian process dynamical models for nonparametric speech representation and synthesis," in *ICASSP*, 2012, pp. 4505–4508.

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

[13] Y. Yamagishi, H. Zen, T. Toda, and K. Tokuda, "Speaker-independent HMM-based speech synthesis system – HTS-2007 system for the Blizzard Challenge," in *Blizzard 2007*, 2007.

[14] "HMM-based Speech Synthesis System (HTS)," http://hts.sp.nitech.ac.jp/.