ISCA Archive
http://www.isca-speech.org/archive

7th ISCA Workshop on
Speech Synthesis (SSW-7)
Kyoto, Japan
September 22-24, 2010

Refined Statistical Model Tuning for Speech Synthesis

Xu Shao, Vincent Pollet and Andrew Breen

TTS R&D, Nuance Communications
{xu.shao, vincent.pollet and andrew.breen}@nuance.com

Abstract

This paper describes a number of approaches to refine and tune statistical models for speech synthesis. The first approach is to tune the sizes of the decision trees for central phonemes in a context. The second approach is a refinement technique for HMM models; a variable number of states for hidden semi-Markov models is emulated. A so-called "hard state-skip" training technique is introduced into the standard forward-backward training. The results show that both the tune and refinement techniques lead to increased flexibility for speech synthesis modeling.

Index Terms: TTS, HSMM, decision tree, hard skip-state

1. Introduction

Statistical parametric Text-To-Speech (TTS) systems gained in popularity in recent years due to their larger trainability, flexibility and compactness compared to the traditional concatenative TTS systems. The statistical parametric speech synthesis systems often use standard left-to-right hidden Markov models (HMMs) to model speech features. The topologies of the HMMs such as the number of states, the number of mixtures etc. are often determined based on a number of heuristics such as target model size, data-fitness, decoding speed and modeling quality.

HMMs used in automatic speech recognition (ASR) systems need to generalize the speech features over different noise conditions and over different speakers in order to achieve a good recognition performance. However, for speech synthesis, HMMs are trained on a single speaker, clean speech corpus. TTS systems aim to produce high-quality speech as close as possible to a human in terms of naturalness and intelligibility. HMMs for TTS need to capture the complexity of human speech production. Standard HMMs do a reasonably good job to model speech but are in many ways too crude to capture the fine details of human speech production. The HMMs for TTS are trained on precise and fine-grained spectral and prosodic features. A good and precise speech parameterization leads to better speech re-generation [1]. However, this precise parameterization is often over-smoothed [2] due to the statistical modeling and parametric re-generation processes. This paper addresses the over-smoothing and modeling aspects as it introduces a number of approaches to refine and to tune statistical models during the training.

The first approach is to train and tune HMMs for speech synthesis in a more controllable and flexible way. Current HMM modeling strategies [3] generate one decision tree per stream and per state for all context models (a so-called phoneme-independent tree or PI tree). This method offers generalization to a great extend; different central phoneme contexts can be clustered into groups. As a consequence, the resulting HMMs are very compact. The model size can be further controlled by altering the clustering threshold based on the occupation counts or by imposing a size-based stopping criterion. However, using a single threshold seems not to be enough to tune the models of a specified central phoneme in context without touching others. Inspired by the work on perceptual modeling for speech synthesis [4], a method to differentiate the clustering thresholds between perceptual salient and non-salient phonemes is explored. The model clustering phase of the HMM training process was modified to generate phoneme dependant (PD) trees.

The second technique in this paper is to refine the HMMs. Often a uniform number of states are employed for all context models. The motivation of this is inspired by the speed of the system, compactness, setup and training method. HMMs for TTS typically have a large number of states to model more details. However, not all phoneme observations carry sufficient acoustic details for the states that are imposed by the basic model topology. It is observed that this results to sub-optimal models in such circumstances. Sub-optimal state-alignment is performed during the forward-backward training. To improve the HMM training, a refinement is proposed using a non-uniform number of states for different phonemes.

The remainder of this paper is structured as follows; Section 2 describes the phoneme dependent tree technique. Section 3 presents a method which uses a variable number of states. In each of following sections, the solutions, results and conclusions are addressed. The summary and future work is discussed in the last section.

2. Phoneme dependent trees

This section compares the TTS system using the PD trees with that using the PI trees.

2.1. Motivation and solution

The decision tree clustering is an important step in statistical model training for speech synthesis as it determines the quality of the models to a great extend. Typically, one decision tree for each stream and each state for all context models is created. The size of the trees is controlled by a single clustering threshold θ_0, as shown in the left hand side of Figure 1.

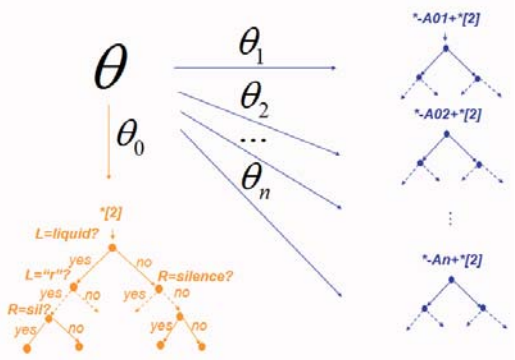

Figure 1 Phoneme Independent vs. Phoneme Dependent

A small threshold keeps the tree splitting, given sufficient observations, and consequently models finer details. Such a

small threshold is desired for perceptual salient phonemes such as stressed vowels (in context) to capture the fine structure of prosody, spectrum and excitation. On the other hand, the human ear is less sensitive to certain sounds such as fricatives. In this case, these phonemes can be modeled with a higher clustering threshold with the advantage of more generalization. In our TTS training corpora all center phonemes are deliberately covered by sufficient good observations for each voice. Therefore, the so-called phoneme dependent decision clustering tree is proposed. The context models with the same central phoneme are clustered into one tree per state per stream as shown on the right hand side of Figure 1. A number of thresholds, $\theta_1, \theta_2, ..., \theta_n$, are designated to control the tree clustering for each central phoneme identifier, for example, *A01, A02,…,An*, for each stream and each state. This modification allows controlling the tree size for each central phoneme identifier. The clustering threshold for specific center phonemes, such as fricatives is higher than for others such as vowels. Thus finer tuning ability for each central phoneme based models is gained.

2.2. Results and discussions

A number of experiments with different thresholds, θ, have been conducted, for example, setting θ to 1.5 for all fricatives and silence, setting θ to 0.8 for all vowels and setting others to 1.0. Informal listening test shows that only minor or even inaudible differences between voices produced by the PI tree and the PD tree clustering.

There are a couple of reasons for this. One threshold is assigned to each central phoneme class for each state and each stream individually which leads too many parameters that need to be tuned. This kind of tuning ability would require an expert both on phonetic and acoustic knowledge. For the experiments naive tuning was performed. Another reason may be explained by the vocoding quality of the system under test. The vocoding quality may mask the model improvements.

2.3. Conclusions

This approach provides an alternative way to train models with added tuning ability. A transparent smaller model can be obtained by increasing the clustering threshold on perceptual non-salient phoneme contexts. Another advantage is that the cpu-load to cluster PD trees is smaller than the cpu-load to cluster PI trees for the same threshold because only subsets of the data are consulted to build the trees. In general, the models of PD trees are less compact than the PI trees for identical clustering thresholds.

3. Non-Uniform HMM topology

This section describes an approach to model HMMs with variable number of states for speech synthesis.

3.1. Solutions

The motivation to adopt non-uniform number of states is inspired by over-definition of certain phoneme classes. A non-uniform number of states can be achieved in different ways.

The state-of-the-art statistical modeling techniques for TTS use so-called hidden Semi-Markov models (HSMMs)[5]. In HSMMs, the state durations are modeled by a single state with multiple streams. The value of each stream represents the state duration probability distributions of other acoustic features such as mel-cepstrum, fundamental frequency etc. This structure has to be reviewed to allow the non-uniform HMM states to be implemented. The basic data structures and tools have to be re-designed accordingly.

In this paper, an alternative solution is presented, hard state skipping, in which the unified number of states is still used, however, with additional information that explicitly describes which states are forced to be skipped. These hard skip-states have zero duration and don't have output emissions. When the skip-states are accessed, they are just skipped to the following state or the previous state without a time delay. This solution emulates an equivalent effect as using a non-unified number of states for different phonemes. To simplify the validation process, a number of assumptions are made; first, no adjacent states are skipped and secondly, the first and the last states are never skipped.

3.2. Algorithms and modifications

In HSMMs [6][7][8], the duration of each state is explicitly estimated. The transition probability is constrained in a way that only transitions to the next states are allowed. This heavily restricts the propagation paths of the forward-backward probability calculation. In our implementation, when a state is denoted as a skip-state, this would indicate that the output probability is approaching zero and the occupancy counts to remain in the hard skip-state are zero as well. To make the forward-backward training work with hard-skip states, a number of algorithmic changes were required which are addressed in following sections.

3.2.1. Forward-backward probability

Assuming that a uniform N-state continuous HSMM, λ, is outlined as follows,

$$\lambda = (A, Y, B, \Pi) \tag{1}$$

$A = \{a_{ij}\}_{i,j=1}^{N}$ is the transition probability from the i^{th} state to the j^{th} state, $Y = \{p_i(\cdot)\}_{i=1}^{N}$ and $B = \{b_i(\cdot)\}_{i=1}^{N}$ are the state duration probability mass function and the state output probability function respectively. Both terms are zero if the i^{th} state is a hard skip-state. $\Pi = \{\pi_i\}_{i=1}^{N}$ is the initial state probabilities. The initial state is start state and therefore $\pi_1 = 1, \pi_2 =, ..., \pi_N = 0$;

The forward-backward probability α and β are defined as (2) and (3) respectively,

$$\alpha_t(j) = P(o_1, ..., o_t, q_t = j \mid \lambda) \tag{2}$$

$$\beta_t(i) = P(o_{t+1}, ..., o_T \mid q_t = i, \lambda) \tag{3}$$

Where $o = \{o_1, ..., o_T\}$ is an observation vector sequence of length *T*.

A number of equations are established for the forward-backward probability estimation in various conditions such as t=1, t>1 and t=T etc. For most of the edge conditions, such as t=1 and t=T or i=1 and i=N, no change for hard skip state training is needed because of the assumptions defined in the previous section. However, in general cases such as $T > t > 1$, when the previous or following state is a hard-skip state, modifications are required shown as follows.

$$\alpha_j^{(q)}(t,1) = \left[\alpha_1^{(q)}(t,1) a_{1j}^{(q)} + \sum_{i=2}^{N_s - 1} \sum_{d=1}^{D^{(q)}} M1(i,d) a_{ij}^{(q)} \right] b_j^q(o_t) \tag{4}$$

where

$$M1(i,d) = \begin{cases} \alpha_{i-1}^{(q)}(t-1,d)\, p_{i-1}^{(q)}(d) & \mathrm{i} = \mathrm{n_s} \\ \alpha_i^{(q)}(t-1,d)\, p_i^{(q)}(d) & \text{otherwise} \end{cases} \tag{5}$$

The $\alpha_j^{(q)}(t,1)$ is the forward probability of the first frame staying at the j^{th} state in the q^{th} model at the time t where t is not the begin or the end frame. The D_i^q is the maximum duration for the i^{th} state in the q^{th} model. The n_s denotes the hard skip-state. The modification is made in case of the hard skip-state shown as in the first line of equation (5). This modification allows the forward probability to use the previous forward probability α and the duration probability function $p(\cdot)$ from previous (non-skip) state. For example, if the i^{th} state is the hard skip state, the α and the $p(\cdot)$ in the i^{th} state which are approaching zero are ignored and the values from the previous non-skip state, for example the $(i-1)^{th}$ state are employed.

Similar to the forward probability estimation, a number of equations are derived for the backward probability estimation for different conditions. Again, the general case is modified as follows,

$$\beta_i^{(q)}(t,d) = p_i^{(q)}(d)a_{iN_q}\beta_{N_q}^{(q)}(t,1) + p_i^{(q)}(d)\sum_{j=2}^{N_q-1} a_{ij}^{(q)} M2(j) + b_i^{(q)}(o_{t+1})\beta_i^{(q)}(t+1,d+1) \quad (6)$$

Where,

$$M2(j) = \begin{cases} b_{j+1}^{(q)}(o_{t+1})\beta_{j+1}^{(q)}(t+1,1) & \text{j} = \text{n}_\text{s} \\ b_j^{(q)}(o_{t+1})\beta_j^{(q)}(t+1,1) & \text{otherwise} \end{cases} \quad (7)$$

The $\beta_i^{(q)}(t,d)$ is the backward probability of staying at the i^{th} state for d frames in the q^{th} model at time t.

A similar modification as in the forward probability estimation is made at *M2(j)*. However, rather than using values from previous non-skipping state as in the forward probability estimation, the backward probability utilizes the values from the following (non-skip) state. For example, if the j^{th} state is the hard skip-state, the values of output probability and following backward probability from the $(j+1)^{th}$ state are employed.

The above equations show that the modifications are made around the transition probability, a_{ij} ,such that if i or j is a variable and it is a hard skip-state, the corresponding modification of *i=i-1* or *j=j+1* should be set to ignore the hard skip-state. Consequently, the forward-backward probability estimation using hard skipping state achieves the ability to emulate a non-unified number of states.

A data flow diagram of the modified forward-backward probability estimation is drawn in Figure 2. For clarity of the figure it is assumed that the duration of all states is one frame.

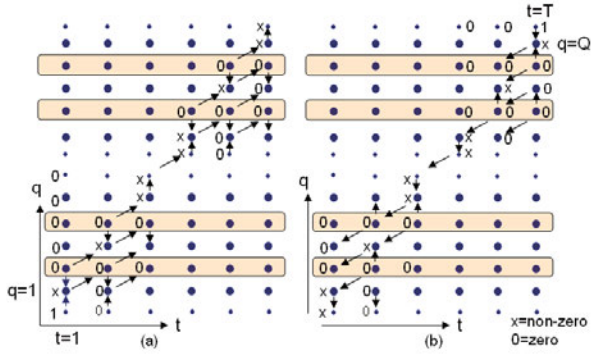

Figure 2: dataflow diagram of a) modified forward probability and b) modified backward probability

Figure 2 depicts the modified forward and backward probability estimation in two adjacent HSMM models respectively. Each model includes 5 states drawn as big circles and the small circles represent the entry and the exit dummy states for each model. In these models, it is assumed that the second and forth states of each model are forced to skip shown as the yellow regions. The values such as 1, 0 or x are the forward-backward probability estimations of the state at the time where the x means a non-zero value. The bottom-left in Figure 2-a is the initial condition (t=1, q=1) for the forward probability estimation while the top-right in Figure 2-b is the initial condition (t=T, q=Q) for the backward probability estimation. The diagonal arrows indicate that the transition probability, a_{ij}, is only valid when *j=i+1*. The downward and upward arrows in Figure 2 show the behavior of *i=i-1* and *j=j+1* in the modified forward-backward probability estimation respectively.

Figure 2 also demonstrates the solution that the modified forward-backward probability estimation goes through the hard skip-states. It can be proved that this dataflow diagram is equivalent to a dataflow diagram of two concatenated HSMMs where each of them has 3 states.

3.2.2. Parameter updating

Parameter updating is one of steps in the embedded training to estimate the probability of state occupation at each frame and to update the accumulators. A number of equations are developed to update the means, variances and weights for each model and each stream. Similar to the modification for the forward-backward probability estimation, a number of equations are reviewed to deal with hard skip-states shown as follows;

$$\chi_t^d(q,i) = \frac{1}{P(O|\lambda)}\alpha_i^{(q)}p_i^{(q)}(d)\cdot \left[\sum_{j=2}^{N_q-1} a_{ij}^{(q)} M3(j) + a_{iN_q}^{(q)}\beta_{N_q}^{(q)}(t,1)\right] \quad (8)$$

where

$$M3(j) = \begin{cases} b_{j+1}^{(q)}(o_{t+1})\beta_{j+1}^{(q)}(t+1,1) & \text{j} = \text{n}_\text{s} \\ b_j^{(q)}(o_{t+1})\beta_j^{(q)}(t+1,1) & \text{otherwise} \end{cases} \quad (9)$$

Where $\chi_t^d(q,i)$ is the probability of occupying the i^{th} state in the q^{th} model for d time frames at the time t.

$$\gamma_t(q,j,s,g) = \frac{1}{P(o|\lambda)}[\alpha_1^{(q)}(t,1)a_{1j}^{(q)}\beta_j^{(q)}(t+1) + \sum_{i=2}^{N_q-1}\sum_{d=1}^{D_i^{(q)}} M4(i,d)a_{ij}^{(q)}\beta_j^{(q)}(t,1) + \sum_{d=2}^{D_j^{(q)}}\alpha_j^{(q)}(t-1,d-1)\beta_j^{(q)}(t,d)]\cdot w_{jsg}^{(q)}N(V(o_{st});\mu_{jsg}^{(q)},\Sigma_{jsg}^{(q)})\prod_{k=1,k\neq s}^{S} b_{jk}^{(q)}(o_{kt}) \quad (10)$$

where

$$M4(i,d) = \begin{cases} \alpha_{i-1}^{(q)}(t-1,d)p_{i-1}^{(q)}(d) & \text{i} = \text{n}_\text{s} \\ \alpha_i^{(q)}(t-1,d)p_i^{(q)}(d) & \text{otherwise} \end{cases} \quad (11)$$

The $\gamma_t(q,j,s,g)$ is computed to update the mean vector, variance matrix and weights in a multi-space probability distribution (MSD-HSMMs) [9] for modeling the fundamental

frequency stream. The $w_{jsg}^{(q)}$ is the weight for the s^{th} stream at the j^{th} state in the q^{th} sub-word model with a space index g. The $V(o_{st})$ is a function to extract a set of space indexes from observation vector sequence. The $N(V(o_{st}); \mu_{jsg}^{(q)}, \Sigma_{jsg}^{(q)})$ is the Gaussian distribution of the observation vector $V(o_{st})$ for the given mean vector $\mu_{jsg}^{(q)}$ and covariance matrix $\Sigma_{jsg}^{(q)}$.

3.3. Evaluations

Two systems, a system using PD trees and a system using PI trees, were built using an US English data set. A set of tri-phone context models were trained in both systems. Each model contains 5 uniform states with a single Gaussian distribution in each state. During the training process, the second and the fourth states in the system using PD trees are denoted as hard skip states for all silence and fricatives models.

In order to determine which states were affected most by the new approach, a targeted comparison was made between the Gaussian models generated from baseline trees and the hard skip state trees using the same text messages. A dynamic time wrapping (DTM) [10] was adopted to align two model streams and to calculate the minimum distance between the two model streams. The Bhattacharyya distance [11] is employed as a cost function for two Gaussian distributions. The analysis was only applied on the spectrum stream. Partial results are provided in Table.1.

Table 1. *Results from model-compare tool*

Name	C1	C2	C3	C4	C5
L1	520	482	2802	4419.82	1.58
L2	310	290	1728	621.62	0.36
L3	290	270	1598	526.15	0.33
L4	80	68	862	331.50	0.38
L5	235	217	1362	392.68	0.29
L6	360	336	2056	576.14	0.28
L7	245	233	1351	555.33	0.41
L8	335	315	1890	517.23	0.27

Table 1. shows the results from the model-compare tool where C1 and C2 are the number of models in the baseline and the hard skip-state respectively; C3 is the total number of steps in frames in the DTW; C4 is the total cost across the whole utterance and C5 is the average cost in each DTW step.

As shown in Table 1 the values in the column C2 are always smaller than the values in the C1 because a number of states have been skipped. The column C5 indicates the measurement of similarity between the models obtained from baseline system and hard skipping state system. In this column, zero indicates two identical model streams and the smaller value means the more similarity of the two model streams.

The results indicate that the models produced by the state-skipping system are very similar to those produced by the baseline system. In addition informal listening tests with the largest cost in Table 1 there are fine differences between the utterances produced by baseline and by the hard skipping state system.

However, informal inspection of 203 synthesis examples highlighted five cases where the synthesis was deemed to be inferior to the baseline for certain sounds. Specifically, sounds generated by models which did not contain skip-states were observed to have different spectral properties.

The reason was the weight of a voicing mixture in a stream for the sounds differed to that of the baseline. This might indicate that the HSMMs are more compact when hard state-skipping technique is employed. As results the absolute default voicing threshold for the stream may need to be optimised accordingly. The parameter generation in TTS engine might also need to be reviewed in the case of hard skip-state, especially the way how the dynamics are determined.

3.4. Conclusions

This experiment demonstrated an approach to emulate a variable number of states using the hard skip state method.

The results show that the voice produced from hard skip state system is similar to those produced from the baseline system. This may be due to the fact that the states chosen for state-skipping are not optimized and imposed at suboptimal locations within the model. More tuning work might need to be investigated to optimize the whole system.

4. Summary and Future work

This paper presents a number of approaches to refine and to tune statistical models for speech synthesis. Phoneme dependent decision trees provide a very flexible way to tune voice. Tuning a voice via decision trees might need knowledge on phonetic properties and the relation to acoustic streams and hence perception. Trying to find an automatic tuning solution is a future research challenge. The hard state-skip refinement technique is a solution to emulate a non-uniform HSMM topology. More research on how to optimize the number of states and to assign the hard skips and the influence on the dynamics is needed.

5. References

[1] Zen, H., Toda, T. and Tokuda, K., "The Nitech-NAIST HMM-Based Speech Synthesis System for the BlizzardChallenge 2006," IEICE Trans Inf Syst, vol. E91-D, no. 6, pp.1764–1773, 2008

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

[3] http://hts.sp.nitech.ac.jp/?Home

[4] Pollet, V. and Breen, A., "Synthesis by generation and concatenation of multiform segments", Proc. Interspeech. pp. 1825–1828, 2008

[5] Murphy, K. P. "Hidden semi-Markov models (HSMMs)", http://www.ai.mit.edu/murphyk, Nov. 2002

[6] Zen, H. "Implementing an HSMM-based speech synthesis system using an efficient forward-backward algorithm", Nagoya Inst. of Technol., TR-SP-0001, Dec. 2007, Tech. Rep.

[7] Yu, S.-Z. and Kobayashi, H. "An efficient forward-backward algorithm for an explicit-duration hidden Markov model", IEEE Signal Processing Letters, 10(1):11–14, 2003

[8] Yu, S.-Z., and Kobayashi, H., "Practical implementation of an efficient forward-backward algorithm for an explicit-duration hidden Markov model", IEEE Trans. on Signal Processing, 54(5):1947–1951, 2006.

[9] Tokuda,K., Mausko,T., Miyazaki, and Kobayashi, N. T., "Multi-space probability distribution HMM", IEICE Trans. Inf. & Syst., vol.E85-D, no.3, pp.455-464, March 2002

[10] Rabiner, L. R. and Juang, B "Fundamentals of speech recognition", Prentice-Hall, Inc., 1993 (Chapter 4)

[11] Fukunaga, K, "Introduction to Statistical Pattern Recognition", Academic Press, Inc., 2nd edition, 1990