10th ISCA Speech Synthesis Workshop
20-22 September 2019, Vienna, Austria

Deep Mixture-of-Experts Models for Synthetic Prosodic-Contour Generation

Raul Fernandez

IBM Research AI
TJ Watson Research Lab
Yorktown Heights, NY, 10598 – USA

fernanra@us.ibm.com

Abstract

Deep recurrent neural networks have been shown to provide state-of-art performance when generating prosodic contours in a speech-synthesis system. These models benefit from the representational capacity obtained by increased compositionality across many layers. As larger amounts of data become available, larger and deeper architectures can be trained at the expense of obtaining models that are expensive both in terms of computation and latency. In this work we take an alternative approach and divide the learning among an ensemble of experts, each of which is a smaller and/or shallower learner whose predictions are then arbitrated by a switching module that assigns sequences of linguistic features to global, sequence-level posteriors, and uses this information to weigh the members of the ensemble. Compared with a single deep cascaded model, this approach is more parallelizable, and can be exploited to obtain a more efficient model in terms of computation (as measured by overall model-size reduction) and latency (as measured by reduction of parameters by branching). We present an architecture where the cluster assignment and prediction models can be trained simultaneously, and demonstrate such gains in efficiency without sacrificing the perceptual quality of the predictions in a subjective listening test.

Index Terms: speech synthesis, prosody modeling, neural network ensembles, recurrent neural networks

1. Introduction

Deep neural networks have been shown to provide state-of-art performance in many speech-synthesis tasks, including the generation of prosodic contours. These deep models benefit from the representational capacity obtained by increased compositionality across many layers, and, as larger amounts of data become available, larger and deeper architectures can be trained at the expense of obtaining models that can be expensive both in terms of computation and latency. Though there is increased availability of large corpora that are suitable for synthesis, resources are not always uniformly available across languages or styles, diminishing the potential of deep learning for these more data-impoverished conditions. In this work, we empirically investigate how a model that trades compositional depth for one that parallelizes that operation across simpler branches might perform compared to an optimized deeper baseline, by considering the benefits of fitting an ensemble of experts to solve a regression task. The experts are designed to be smaller and/or shallower learners, and their respective predictions are arbitrated by a module that assigns sequences of linguistic features to global, sequence-level, posteriors over the experts and uses this information to weigh the members of the ensemble. We present a simple architecture where both the gating module and all the experts can be jointly trained (Sec. 2), and conduct an evaluation against a solid baseline optimized over a set of configurations (Sec. 4) by carrying out a set of experiments that look into the trade-off between depth and module simplicity (Sec. 5). Then finally two models are evaluated in a subjective listening test against the baseline model (Sec. 6) before we draw some conclusions and outline some next steps to follow-up on this work (Sec. 7).

2. Model

Consider a pair of input and output sequences $x_{1:T} = \{x_1, \cdots, x_T\}$ and $y_{1:T} = \{y_1, \cdots, y_T\}$ where $x_t \in \mathbb{R}^M$ and $y_t \in \mathbb{R}^D$, and a Deep Mixture of Experts (D-MoE) consisting of an ensemble of K models, each of which performs an independent prediction:

$$\begin{aligned} z^0_{1:T} &= f_0(x_{1:T}; \Theta_0) \\ z^1_{1:T} &= f_1(x_{1:T}; \Theta_1) \\ &\cdots \\ z^{K-1}_{1:T} &= f_K(x_{1:T}; \Theta_{K-1}), \end{aligned} \tag{1}$$

a gating function $g(\cdot)$ mapping the full sequence to a K-dimensional vector:

$$\pi = g(x_{1:T}; \Gamma) \quad s.t. \quad \pi \in \mathbb{R}^K \quad \text{and} \quad \sum_k \pi_k = 1, \tag{2}$$

and a prediction $\tilde{y}_{1:T}$ obtained by the convex combination:

$$\hat{y}_{1:T} = \sum_k \pi_k z^k_{1:T} \tag{3}$$

Given a training set of N sequences, we can estimate the parameters of the model by:

$$\Theta^*, \Gamma^* = \arg\min_{\Theta,\Gamma} \sum_n \ell\big(\hat{y}^{(n)}_{1:T_n}, y^{(n)}_{1:T_n}\big) \tag{4}$$

where $\Theta = \{\Theta_0, \cdots, \Theta_{K-1}\}$ and $\ell(\cdot,\cdot)$ is a suitable loss function mapping two sequence arguments to a real-valued number. We consider an additive loss with 2 components: a *reconstruction* component that measures how well the targets are matched by the predictions in Eqn. 3, and an *expert-entropy* component that encourages all the experts to be active participants in the final prediction. For the reconstruction term, we will adopt a weighted average of L_1 norm loss (where the weighting scheme will be explained in more detail in Sec. 4). To promote equal participation among the experts, we introduce the loss term:

$$\ell_{ent} = \frac{1}{N} \sum\nolimits_n \left(1 - \frac{\sum_k \pi^n_k \log \pi^n_k}{\log K}\right)^2 \tag{5}$$

10.21437/SSW.2019-47

involving a normalized entropy term over the sequence-wise posteriors, averaged across all sequences. The denominator in (5) is the entropy of a uniform multinomial with K classes, the maximum entropy a multinomial distribution can attain, which bounds ℓ_{ent} to the interval $[0, 1]$ (lowest when the posterior is maximally distributed).

3. Previous and Related Work

The use of expert-gate architectures has had an established presence in statistical learning for a few decades since the introduction of the Mixture-of-Experts (MoE) model [1], and, with the advent of advances in deep learning, formulations of MoEs that involve deep experts have recentlly received attention [2, 3, 4, 5], with application to diverse tasks such as image classification (using convolutional-network experts) [6, 7, 8] and speech enhancement (using recurrent-network experts) [9, 10]. In spite of their applicability across areas, it is somewhat surprising that their use within speech-synthesis tasks, and prosody modeling in particular, remains rare (though the closely-related Products-of-Expert (PoE) model was investigated for statistical parametric speech synthesis [11]). In the current work, we carry out what we believe is a novel investigation that seeks to combine the advantages of ensemble- and deep-learning for prosody modeling.

4. Optimizing a Baseline Model

Model: To have a strong point of comparison for the ensemble of learners, we first selected and optimized an architecture for the given task by doing a grid search over a variety of configuration and optimization hyperparameters. The model type we selected as a base model is a multi-layer, bi-directional recurrent neural network with long short-term memory units (BiLSTM), an architecture we have previously deployed for prosody prediction and which provides a strong baseline [12, 13]. An LSTM layer is defined by the following set of recurrency equations for a forward (let-to-right) layer.

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (6)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (7)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (8)$$

$$g_t = \tanh(W_{xg}x_t + W_{hg}h_{t-1} + b_g) \quad (9)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (10)$$

$$h_t = o_t \odot \tanh(c_t). \quad (11)$$

Letting $h_{t+1} \leftarrow h_{t-1}$ and $h_{c+1} \leftarrow h_{c-1}$ in Eqns. (6-11) defines an analogous recurrency for a backward (right-to-left) layer. In pilot experiments we also considered some alternate architectures, like Residual [14], and Dense Networks [15]. However, as none of these significantly outperformed a simpler BiLSTM configuration, the latter is used throughout as both the baseline model, and as the configuration of the single experts, for simplicity. In the baseline model, a stack of M bidirectional layers (with each layer containing a forward and backward chain) is composed, where the input to the m^{th} layer consists of the forward and backward activations from the $(m-1)^{st}$ layer stacked, and whose final output is given by the linear transformation:

$$y_t = W_{\overrightarrow{h}y}\overrightarrow{h}_t^M + W_{\overleftarrow{h}y}\overleftarrow{h}_t^M + b_y. \quad (12)$$

Here, y_t represents a 4-dimensional vector of acoustic prosodic targets used in the back-end for speech generation (as explained in Section 6). The temporal granularity of the observations are not constant and approximately correspond to the initial, middle, and final one third of a phone (where the segmentations have been automatically derived from forced alignments against 3-state hidden Markov models). The 4 prosodic targets are the segment's duration, initial and final f_0, and energy. These targets are paired with an input vector x_t consisting of 342 linguistic features, comprising one-hot encodings of various categorical features (e.g., phoneme identity, part of speech, phrase type, etc.) and positional counts measuring a segment's distance to various preceding/following boundaries (syllable, word, phrase, etc.). In addition to the linguistic features, the input contains a speaker embedding to allow capturing speaker variability when training the model in a multi-speaker or multi-style mode. This embedding is a global, utterance-level vector that is broadcast over the entire length of x_t and appended to each observation, and which is learned simultaneously with the model parameters during training. For all the experiments reported here, we have used data from 8 professionally-recorded synthesis corpora, comprising 5 different speakers (one male; four female) and totaling 64 hours. The data for one of the female speakers contains various affective styles, each of which is treated as a separate subcorpus, and is allowed to learn its own embedding. Each of the subcorpora is split into a 90% training set and a 10% test set (each the speakers/styles is proportionally represented during training and testing). The optimization criterion is a weighted average of L_1 loss:

$$\ell_1 = \frac{\sum_{s=0}^{3}\alpha_s\sum_{n=0}^{N-1}\sum_{t=0}^{T_n-1}w_t^{n,s}\left|\tilde{y}_t^{n,s} - y_t^{n,s}\right|}{\sum_{s=0}^{3}\alpha_s\sum_{n=0}^{N-1}\sum_{t=0}^{T_n-1}w_t^{n,s}}, \quad (13)$$

where n, t, s index, respectively, sequence number, time, and each of the 4 prosodic streams described. The weights $w_t^{n,s}$ are observation- and stream-specific non-negative values reflecting the importance of the observation to the prosodic contour (and which are derived from information such as degree of voicing, silence status, etc.) The weights α_s are stream-specific global weights trading the fit among the different prosodic contours, and are set to 1 (for the energy component), 2 (for the duration component), and 4 (for the f_0 contours). Both w and α remain constant throughout all the experiments. Parameters are updated using the AMSGrad variant of the ADAM algorithm [16], and the PyTorch package is used in the all the experiments to train and evaluate the models [17].

We optimize the baseline model configuration and training hyperparameters by doing a grid search over a) the dimensionality of the speaker embedding, b) the number of recurrent layers, c) the dimensionality of the hidden units (this number is always assumed to be the same for the forward and backward chains within a given layer), and d) hyperparameters of the learning rate schedule. The learning rate exponentially decays from an initial value lr_0 (fixed at $lr_0 = 0.01$) to a final lr_{final} over T epochs, and remains constant after T (different combinations of T and lr_{final} are explored during the search). The grid search above generates a total of 192 configurations, with models ranging in size from 283K to 812K parameters.

The training recipe makes use of 1) a training set that is used both for updating the model parameters and diagnosing convergence, and 2) an independent test set used for selecting one model among the competing configurations. Unlike recipes where a third development set is used for early stopping [13], we instead opt for regularizing the model to prevent overfitting and diagnose convergence by monitoring the regularized loss instead. This is one attempt to deal with the empirical ob-

Config.	K	Shape	Hidden Layers	# Parms	LatMsr	WAE	ρ_{avg}	$\%\sigma^2_{avg}$
BSL	–	R	$\{75, 75, 75, 75\}$	666.6K	1.0	0.452	(0.72, 0.74)	(0.54, 0.78)
A.1	2	R	$\{45, 45, 45, 45\}$	664.9K	0.438	0.460	(0.72, 0.75)	(0.57, 0.75)
A.2		B	$\{54, 44, 35, 31\}$	663.5K	0.437	0.468	(0.71, 0.74)	(0.56, 0.78)
A.3	3	R	$\{34, 35, 35, 35\}$	666.4K	0.293	0.464	(0.71, 0.74)	(0.56, 0.77)
A.4		B	$\{40, 34, 28, 23\}$	665.1K	0.292	0.468	(0.71, 0.74)	(0.57, 0.74)
A.5	4	R	$\{28, 29, 29, 28\}$	666.4K	0.220	0.468	(0.71, 0.74)	(0.57, 0.77)
A.6		B	$\{34, 25, 22, 17\}$	666.2K	0.219	0.473	(0.71, 0.74)	(0.57, 0.75)
A.7	5	R	$\{24, 25, 24, 24\}$	664.0K	0.175	0.468	(0.71, 0.74)	(0.57, 0.75)
A.8		B	$\{28, 23, 19, 14\}$	665.2K	0.175	0.473	(0.71, 0.74)	(0.57, 0.76)
B.1	2	R	$\{51, 51, 51\}$	665.2K	0.438	0.459	(0.72, 0.74)	(0.55, 0.77)
B.2		B	$\{58, 46, 38\}$	664.9K	0.438	0.464	(0.71, 0.74)	(0.56, 0.75)
B.3	3	R	$\{39, 38, 39\}$	667.0K	0.293	0.463	(0.71, 0.74)	(0.57, 0.77)
B.4		B	$\{44, 34, 27\}$	666.2K	0.292	0.467	(0.71, 0.74)	(0.57, 0.78)
B.5	4	R	$\{32, 31, 30\}$	663.8K	0.219	0.467	(0.71, 0.74)	(0.58, 0.77)
B.6		B	$\{36, 27, 20\}$	666.9K	0.220	0.469	(0.71, 0.74)	(0.56, 0.76)
B.7	5	R	$\{27, 26, 27\}$	666.5K	0.176	0.469	(0.71, 0.74)	(0.58, 0.77)
B.8		B	$\{30, 23, 17\}$	665.0K	0.175	0.473	(0.71, 0.74)	(0.58, 0.78)

Table 1: *Summary of experiments investigating the performance of mixture models as a function of number of components (column 2) with respect to the baseline model (row 1). Columns 3-6 describe each expert's typology, number of parameters, and latency measure factor. Columns 7-9 list test-set objective metrics in terms of weighted absolute error, correlation, and variance ratio. All of the resulting D-MoE models are comparable to the baseline model in terms of number of parameters.*

servation that prosody models whose convergence is diagnosed via a small development set often stop too early (i.e., underfit) in the sense that they are perceptually less preferable to one that is allowed to learn beyond the "optimal" development-set point. This fact is a consequence of the one-to-many nature of prosodic assignment and the fact that, given a text string, the hypothesis generated by a model could be perceptually better while incurring higher objective loss with respect to a development set (which almost always includes only one prosodic realization out of many for a given text string, and one that need not be close to the one proposed in the hypothesis). (See Challenge #7 in [18] for a more detailed explanation of this phenomenon.) Regularization is introduced in the form of drop-out layers between any consecutive pair of recurrent layers (with drop-out probability $p = 0.2$), plus a drop-out layer after the inputs and embedding are merged (with drop-out probability $p = 0.05$). Optimization stops when the *regularized* loss stops decreasing after $T = 7$ iterations. The test set is then used to rank all the configurations and choose as the optimal baseline the model attaining the lowest loss on an independent set. In this experiment, that model is a 4-layer BiLSTM with 75 hidden units per layer, plus an initial embedding layer with an embedding dimensionality of 10. The number of parameters in the model is 666,684 (666,604 for the stack of recurrent layers, and 80 for the initial embedding matrix).

5. Deep Mixture of Experts

We next consider a mixture of individual deep learners modulated by a gate, which can be any model mapping a sequence of input observations onto a fixed K-dimensional vector (i.e., the gate emits one posterior over the mixture components conditioned on the entire sequence). As a gating mechanism, we also adopt a unidirectional (forward) LSTM architecture and use the last hidden state (which, after T updates, has absorbed information about the full sequence) as a sequence-level embedding:

$$h_{1:T} = \text{LSTM}(x_{1:T}) \tag{14}$$

$$\pi = \text{SoftMax}(Hh_T), \tag{15}$$

where LSTM$(\cdot)$ is the set of Eqns in (6-11), $H \in \mathbb{R}^{KxD}$, D is the dimensionality of the embedding (i.e., number of hidden units in the forward chain), and K is the number of mixtures.

For simplicity, we consider architectures where the K experts within one D-MoE share the same architecture type and size (but vary the size and topology across the different D-MoEs in the experiments described next). To focus on investigating the effects of the ensemble, and to limit the combinatorial space, we also fix the architecture of the gating network to be the same for all experiments: a single-layer LSTM with 50 hidden units (only H in (15) changes in size accordingly to accommodate the different number of mixtures explored). The size of the speaker embedding is fixed, and identical to the one optimized in the baseline configuration. The learning-rate hyperparameters are also taken from the optimal baseline configuration[1].

As mentioned in Sec. 2, when training the D-MoE we add another criterion (Eq. 5) to promote all the members of the ensemble, *on the average*, to be comparably active in making a prediction, and to avoid a single expert being majorly responsible (a behavior that was observed during initial pilot studies). The overall training criterion is a weighted sum of the two:

$$\ell = \ell_1 + C\ell_{ent}, \tag{16}$$

where C is a constant set to 500 in all the experiments in this section.

In the first set of experiments (Table 1) we consider D-MoE architectures that split the number of parameters of the baseline model across K mixtures plus the gate network (i.e., the total

[1]This is, again, done in the interest of limiting the number of comparisons to a manageable number. Clearly, further optimizations of the hyperparameters of the D-MoE architecture are possible.

Config.	K	Shape	Hidden Layers	# Parms	LatMsr	WAE	ρ_{avg}	$\%\sigma^2_{avg}$
BSL	–	R	$\{75, 75, 75, 75\}$	666.6K (1.0)	1.0	0.452	(0.72, 0.74)	(0.54, 0.78)
C.1	2	R	$\{35, 35, 35, 35\}$	479.1K (0.72)	0.299	0.463	(0.71, 0.74)	(0.54, 0.78)
C.2		B	$\{50, 35, 30, 25\}$	565.1K (0.85)	0.363	0.464	(0.71, 0.74)	(0.54, 0.75)
C.3	3	R	$\{30, 30, 30, 30\}$	557.0K (0.84)	0.238	0.468	(0.71, 0.74)	(0.56, 0.76)
C.4		B	$\{35, 30, 25, 20\}$	568.5K (0.85)	0.244	0.469	(0.71, 0.74)	(0.55, 0.78)
C.5	4	R	$\{25, 25, 25, 25\}$	569.9K (0.85)	0.183	0.469	(0.71, 0.74)	(0.56, 0.74)
C.6		B	$\{28, 25, 20, 10\}$	552.8K (0.83)	0.177	0.471	(0.70, 0.74)	(0.55, 0.74)
C.7	5	R	$\{21, 21, 21, 21\}$	560.8K (0.84)	0.144	0.474	(0.70, 0.74)	(0.55, 0.76)
C.8		B	$\{25, 20, 15, 10\}$	569.2K (0.85)	0.146	0.479	(0.70, 0.74)	(0.56, 0.77)
D.1	2	R	$\{45, 45, 45\}$	566.3K (0.85)	0.364	0.463	(0.71, 0.74)	(0.55, 0.75)
D.2		B	$\{50, 30, 25\}$	502.8K (0.75)	0.316	0.469	(0.71, 0.74)	(0.54, 0.73)
D.3	3	R	$\{33, 33, 33\}$	548.3K (0.82)	0.234	0.470	(0.71, 0.74)	(0.56, 0.79)
D.4		B	$\{40, 25, 20\}$	558.5K (0.84)	0.239	0.473	(0.71, 0.74)	(0.56, 0.78)
D.5	4	R	$\{28, 28, 28\}$	578.4K (0.87)	0.187	0.468	(0.71, 0.74)	(0.56, 0.80)
D.6		B	$\{30, 25, 20\}$	566.1K (0.85)	0.182	0.471	(0.71, 0.74)	(0.56, 0.74)
D.7	5	R	$\{23, 23, 23\}$	559.6K (0.84)	0.144	0.472	(0.71, 0.74)	(0.57, 0.75)
D.8		B	$\{25, 22, 20\}$	578.9K (0.87)	0.149	0.469	(0.71, 0.74)	(0.56, 0.76)

Table 2: *Summary of experiments investigating the performance of mixture models as a function of number of components (column 2) with respect to the baseline model (row 1) with overall model-size reduction. Columns 3-6 describe each expert's typology, number of parameters (including, in parenthesis, the fraction of the baseline model size), and latency measure factor. Columns 7-9 list test-set objective metrics in terms of weighted absolute error, correlation, and variance ratio.*

number of parameters in these configurations remains comparable to that of the baseline). For each of these we consider two types of sub-configurations: one that preserves the same depth in each of the experts as in the baseline (4 bidirectional layers; configurations A.1–A.8 in Table 1), and one that scales back on depth by considering 3-layer experts (configurations B.1-B.8 in Table 1). We also consider two types of topology on the hidden units: a *rectangular* topology that keeps the number of hidden units per layer roughly constant across the depth of the network, and a *bottleneck* topology that widens toward the inputs and tapers toward the outputs. These are labeled as R and T on the table respectively. This exploration over network typology and depth is an attempt to not rely on the optimal configuration of the baseline (optimized under different conditions) while maintaining overall model complexity (in the sense of number of parameters) when evaluating this approach. Since one of the goals of exploring the D-MoE architecture is to arrive at parallelizable, and shallower architectures, we introduce a simple measure of the reduction of complexity in the mixture model by looking at the reduction in the number of model parameters in each of the branches compared to the single deeper model. We call this a *latency measure* though it should be clear the number does not translate directly into real-time factors, or floating-point operations (and it is only meaningful because we have fixed the architecture types to be the same across all branches), but it is an easy-to-compute figure that correlates with the overall input-to-output performance in an implementation that exploits structure for parallelization. In addition to the expert's configuration, Table 1 lists a number of objective metrics that provide a summary the performance of the model. The weighted absolute error (WAE) is the ℓ_1 loss term of Eqn. 13 to enable direct comparison with the baseline model (whose performance is summarized on row 1, and which does not include an entropy-ratio loss term). Column 8 contains a tuple with the Pearson's correlation coefficients between predictions and targets (for the duration and f_0 streams respectively), averaged across all sequences in the test set:

$$\rho_{avg} = \frac{1}{N}\sum_n \rho^n_{y\tilde{y}} = \frac{1}{N}\sum_n \frac{Cov(y^n, \tilde{y}^n)}{\sigma_{y^n}\sigma_{\tilde{y}^n}} \tag{17}$$

The last column contains a similar tuple (for duration and f_0) with the ratio of the prediction-to-target variance, averaged across all sequences, a measure of what proportion of the dynamic range of the natural contours (a perceptually-relevant quantity) is approximated by the predictions:

$$\%\sigma_{avg} = \frac{1}{N}\sum_n \frac{\sigma^2_{\tilde{y}^n}}{\sigma^2_{y^n}} \tag{18}$$

In the second set of experiments, we investigate the performance of the ensemble models under the condition that the overall model is smaller in size: while retaining the same architecture for the gating network, we further reduce the number of hidden units in the various configurations to obtain models ranging from 72%-87% of the original number of parameters, as shown in Table 2. Notice that this naturally further reduces the original latency measure.

From inspection of Tables 1 and 2, we can make the following remarks: 1) We see a degradation in losses in the order of 1.5%–4.4% relative depending on the configuration. In most cases, the variants with bottleneck typology led to a worse error compared to the flatter structure within the same number of mixtures and overall parameter size, and structures with shallower experts also resulted in marginally lower losses within the same number of experts (i.e., A.x vs B.x; C.x vs B.x). 2) The correlation measure is stable throughout for both duration and f_0 streams. 3) The variance of the predicted duration contours never falls that of the baseline predictions, and in most cases there is a marginal improvement. For the f_0 contours, however, the tendency is for the mixture models to produce slightly lower variance compared to the baseline. This may have perceptual implications that are further investigated in a perceptual listening test (Sec. 6).

To get a sense of, and visualize, the behavior of the different experts, we have plotted the distribution of the sequence-level gate posteriors after convergence for all the utterances in the test set for a model with 3 experts. This is shown in Fig. 1 for data for 5 different speakers (corresponding to each row labeled $S1$-$S5$). As a result of training with the entropy-matching term on the loss, the weights are centered around $1/3$. However, we can see slightly different behaviors from the different components, particularly the second one. We can also observe some more subtle differences across different speakers within a single expert.

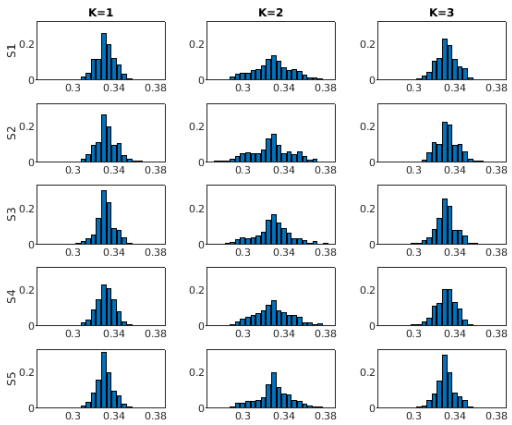

Figure 1: *Distributions of sequence-level gate weights for a 3-component model (K=1,2,3). Each row corresponds to a different speaker (S1-S5).*

6. Perceptual Evaluation

We carried out a listening test where two models from teh experiments described were compared against the baseline. Guided by the observations in the previous section, we opted for configuration B.3 which represents a good trade-off between test-set error and the other properties of the model (a relative 2.4% degradation of the baseline loss while maintaining the same number of parameters and achieving a *latency measure* of 0.293). From the reduced-model-size condition, and guided by the metrics on Table 2, we selected configuration D.1 (a model with 85% of the number of parameters in the baseline, a *latency measure* of 0.37 and a relative loss degradation of 2.4%).

The systems were evaluated using a crowd-sourced test, where subjects were asked to rate the overall quality and naturalness of the samples on a 5-point scale (Bad, Poor, Fair, Good, Excellent). A set of 53 test utterances not included in the training or test sets was used, and each stimulus (i.e. combination of utterance and system) was rated 30 different times. The predicted prosodic contours were evaluated using a fully-parametric, neural synthesizer described in more detail in [19, 20] and comprised, at a high level, of 3 main modules: (1) a prosody-generation module that takes linguistic and positional features derived from a text-analysis front-end to generate 4 prosodic targets (as described in Section 4), (2) a spectral-parameter generation module that takes phonetic information and predicted prosody to generate a 60-dimensional vector containing cepstral and pitch information that, after some enhancements, is passed to (3) a back-end neural LPCNet-based [21] vocoder to generate the final samples. The speaker embedding used for the evaluation sentences corresponded to the speaker who had been used to train modules (2) and (3) in a speaker-dependent fashion. The results are summarized on Table 3. There is only a minor, and statistically insignificant difference of 0.03 MOS points, between the deeper architecture, and the comparably-sized mixture of 3 experts. This difference widens to 0.05 MOS points (and reaches significance at the $p = 0.042$ level) when we consider a model that is 85% the original size, suggesting that distributing the parameters across experts while preserving overall model size is a better strategy.

Baseline	B.3	D.1
3.73 ($\sigma = 0.85$)	3.70 ($\sigma = 0.84$)	3.68 ($\sigma = 0.86$)

Table 3: *Mean opinion scores in a perceptual-evaluation task rating the overall quality and naturalness of the samples on a 5-point scale. The only statistically-significant difference is between the Baseline and D.1 configuration ($p = 0.042$)*

7. Conclusions and Further Work

We have explored the performance of a Deep Mixture-of-Models and shown that an ensemble of deep learners provides a viable architecture to streamline the performance of a deep model. By using a more parallelizable architecture that relies on running smaller and shallower experts, and then combining their outputs, we have demonstrated that the model can closely approximate the subjective perceptual quality of the prosodic contours of an optimized baseline that has as many parameters, but more depth and latency.

We have also investigated the perceptual implications for models that furthermore shrink the overall size of the baseline model, and see a very minor (though statistically significant) degradation of less than 0.1 MOS points, with a model that has about 85% of the parameters. This represents an initial exploration of the benefits of using more distributed architectures as an alternative to deeper and deeper models, a strategy that might be preferable when the amounts of data available, as might be the case for low-resource languages, may not fully deliver on the advantages of deep architectures. There are many potentially fruitful avenues to further explore after this initial work: 1) The current architecture, for instance, contains a fairly simple gating network whose training only exploited an entropy criterion to comparably favor all experts. A more interesting approach might combine this idea with clustering, and training criteria that enforce cluster discovery, to have the gating network find sequential modes and have each expert specialize within a mode. A straightforward extension would be to gate the observations at each time frame. 2) The entropy-matching criterion, though introduced to prevent mode collapse, may be causing too much uniformity across different inputs (as we showed in Fig. 1), but we could explore alternative training criteria to elicit more variance from the experts across different input sequences. 3) To facilitate comparisons, and to keep separate the effects of mixture modeling from those of mixing different architecture types, we purposely adopted and preserved a common architecture across the different experts and baseline. It would be worthwhile, however, to investigate the complementarity of different architectures, possibly ones with different temporal properties (e.g., clockwork RNNs with different rates [22]), or more sophisticated experts (e.g., autoregressive networks [23]).

8. Acknowledgements

We would like to thank Zvi Kons, Alex Sorin, and Slava Shechtman for their help on various aspects of this work.

9. References

[1] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, pp. 79–87, 1991.

[2] D. Eigen, M. Ranzato, and I. Sutskever, "Learning factored representations in a deep mixture of experts," in *Proc. ICLR Workshop*, Banff, Canada, 2014.

[3] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. V. Le, G. E. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *Proc. ICLR*, Toulon, France, 2017.

[4] B. Zhang, A. Azadeh Davoodi, and Y.-H. Hu, "A mixture of expert approach for low-cost customization of deep neural networks," *CoRR*, vol. abs/1811.00056, 2018. [Online]. Available: http://arxiv.org/abs/1811.00056

[5] K. Zhao, Y. Li, C. Zhang, C. Yang, and H. Xu, "Adaptive recurrent neural network based on mixture layer," *CoRR*, vol. abs/1801.08094, 2018. [Online]. Available: http://arxiv.org/abs/1801.08094

[6] J. Fan, T. Zhao, Z. Kuang, Z. Yu, and J. Yu, "Deep mixture of experts with diverse task spaces," in *Proc. ICMLA*, 2017, pp. 721–725.

[7] R. Cohen, "Mixture of convolutional neural networks for image classification," Master's thesis, The Hebrew University of Jerusalem, 2018.

[8] S. F. Dodge and L. J. Karam, "Quality robust mixtures of deep neural networks," *IEEE Trans. Image Proc.*, vol. 27, no. 11, pp. 5553–5562, 2018.

[9] S. E. Chazan, J. Goldberger, and S. Gannot, "Deep recurrent mixture of experts for speech enhancement," in *Proc. IEEE Workshop on Applications of Signal Proc. to Audio and Acoustics*, New Paltz, NY, 2017, pp. 359–363.

[10] P. Karjol and P. K. Ghosh, "Speech enhancement using deep mixture of experts based on hard expectation maximization," in *Proc. Interspeech*, Hyderabad, India, 2018, pp. 3254–3258.

[11] H. Zen, M. J. Gales, Y. Nankaku, and K. Tokuda, "Product of experts for statistical parametric speech synthesis," *IEEE Transactions Audio, Speech, and Language Processing*, vol. 20, no. 3, pp. 794–805, 2012.

[12] R. Fernandez, A. Rendel, B. Ramabhadran, and R. Hoory, "Prosody contour prediction with long short-term memory, bi-directional, deep recurrent neural networks," in *Proc. Interspeech*, Singapore, 2014, pp. 2268–2272.

[13] ——, "Using deep bidirectional recurrent neural networks for prosodic-target prediction in a unit-selection text-to-speech system," in *Proc. Interspeech*, Dresden, 2015, pp. 1606–1610.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, Las Vegas, US, June 2016.

[15] M. Strake, P. Behr, T. Lohrenz, and T. Fingscheidt, "DenseNet BLSTM for acoustic modeling in robust ASR," in *Proc. Spoken Language Technology Workshop*, Athens, Greece, 2018, pp. 6–12.

[16] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of ADAM and beyond," in *Proc. ICLR*, Vancouver, Canada, 2018.

[17] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS Autodiff Workshop*, 2017.

[18] A. Rosenberg, "Speech, prosody, and machines: Nine challenges for prosody research," in *Proc. Speech Prosody*, June 2018, pp. 784–793.

[19] Z. Kons, S. Shechtman, A. Sorin, R. Hoory, C. Rabinowitz, and E. d. S. Morais, "Neural TTS voice conversion," in *Proc. Spoken Language Technology Workshop*, Athens, Greece, 2018, pp. 290–296.

[20] Z. Kons, S. Shechtman, A. Sorin, C. Rabinowitz, and R. Hoory, "High quality, lightweight and adaptable TTS using LPCNet," *CoRR*, vol. abs/1905.00590, 2019. [Online]. Available: https://arxiv.org/abs/1905.00590

[21] J. M. Valin and J. Skoglund, "LPCNET: Improving neural speech synthesis through linear prediction," in *ICASSP*, Brighton, England, 2019, pp. 5891–5895.

[22] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork RNN," in *Proc. International Conference on Machine Learning*, vol. 32, no. 2, Bejing, China, 22–24 Jun 2014, pp. 1863–1871. [Online]. Available: http://proceedings.mlr.press/v32/koutnik14.html

[23] X. Wang and S. a. Takaki, "Autoregressive neural f0 model for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Spech, and Language Processing*, vol. 26, no. 8, pp. 1406–1419, 2018.