NEURAL HMMS ARE ALL YOU NEED (FOR HIGH-QUALITY ATTENTION-FREE TTS)

Shivam Mehta, Éva Székely, Jonas Beskow, Gustav Eje Henter

Division of Speech, Music and Hearing, KTH Royal Institute of Technology, Stockholm, Sweden

ABSTRACT

Neural sequence-to-sequence TTS has achieved significantly better output quality than statistical speech synthesis using HMMs. However, neural TTS is generally not probabilistic and the use of non-monotonic attention both increases training time and introduces "babbling" failure modes that are unacceptable in production. This paper demonstrates that the old and new paradigms can be combined to obtain the advantages of both worlds, by replacing the attention in Tacotron 2 with an autoregressive left-right no-skip hidden Markov model defined by a neural network. This leads to an HMM-based neural TTS model with monotonic alignment, trained to maximise the full sequence likelihood without approximations. We discuss how to combine innovations from both classical and contemporary TTS for best results. The final system is smaller and simpler than Tacotron 2, and learns to speak with fewer iterations and less data, whilst achieving the same naturalness prior to the post-net. Unlike Tacotron 2, our system also allows easy control over speaking rate.

Index Terms— seq2seq, attention, HMMs, duration modelling, acoustic modelling

1. INTRODUCTION

Text-to-speech (TTS) technology has advanced tremendously in the last decade, and output speech quality has seen a number of step changes as the field evolved. Statistical parametric speech synthesis (SPSS) based on hidden Markov models (HMMs) [1, 2], has now largely been supplanted by neural TTS [3]. Waveform-level deep learning [4] greatly improved segmental quality over signal-processing based vocoders, while sequence-to-sequence models with attention [5, 6] demonstrated greatly improved prosody. Combined, as in Tacotron 2 [7], these innovations produce synthetic speech whose naturalness sometimes rivals that of recorded speech.

However, not all aspects of TTS systems have improved along the way. The integration of deep learning with positional features into HMM-based TTS increased naturalness [8], but sacrificed the ability to learn to speak and align simultaneously, instead requiring an external forced aligner. Attention-based neural TTS systems [6] reintroduced the ability to learn to align, but are not grounded in probability and require more data and time to start speaking. Furthermore, their non-monotonic attention mechanisms do not enforce a consistent ordering of speech sounds. As a result, the synthesis is susceptible to skipping and stuttering artefacts, and may break down catastrophically, resulting in unintelligible gibberish.

In this article, we make the case that HMM-based and neural TTS approaches can be combined to gain the benefits of both worlds. Our main contribution is to describe a neural TTS architecture based on Tacotron 2, but with the attention mechanism replaced by a Markovian hidden state, to obtain a fully probabilistic, joint model of durations and acoustics. The model development leverages design principles from both HMM-based and sequence-to-sequence TTS. Experiments show that the model gives high-quality speech output on par with that of a comparable Tacotron 2 model, and produces intelligible speech already after 1k updates, a fifteen-fold improvement on Tacotron 2. Unlike Tacotron 2, it also allows control over speaking rate. For audio and code, please see our demo webpage.

2. BACKGROUND

The starting point of this work is [9], which identified four key differences between HMM-based SPSS and sequence-to-sequence attention-based TTS that had a notable impact on output quality:

1. Neural vocoder with mel-spectrogram inputs
2. Learned front-end (the encoder)
3. Acoustic feedback (autroregression)
4. Attention instead of HMM-based alignment

Among these, items 1–3 led to improved speech quality, whereas attention sometimes made the output significantly worse. This paper incorporates aspects 1–3 into a TTS system that leverages neural HMMs [10, 11] rather than attention for sequence-to-sequence modelling. Sec. 2.1, below, describes how to add aspects 1–3 to HMMs based on prior work, with attention (aspect 4) discussed in Sec. 2.2.

2.1. Neural TTS aspects in HMM-based TTS

For item 1, pre-trained neural vocoders like [12] driven by spectral output features afford high signal quality and also avoid the explicit averaging over pitch contours that leads to overly flat intonation in systems that parameterise speech using a separate $f0$ feature [9]. However, nothing prevents HMM-based TTS from using mel-spectrogram features and neural vocoders: this is just a straightforward change to the acoustic features, and the HMM-based approach proposed in this paper uses this setup.

Another factor in the improved prosody is item 2, the learned front-end (i.e., the encoder). Again, there is nothing that prevents using this idea in a system that leverages HMMs. The HMM-based systems we introduce all use the same encoder architecture as Tacotron 2 [7] with no additional linguistic features added.

The situation for item 3, autoregression (AR), is again similar, in that AR and HMMs are not mutually exclusive. Acoustic models in HMM-based TTS systems benefit from using positional and durational information [13, 8], that increases granularity by enabling the statistics of each generated frame to be different, together with dynamic features [14] to promote continuity across time. However, positional and durational features violate the Markov assumption (e.g., they depend on the time spent in the current state), preventing re-alignment during TTS training. In a model like Tacotron, positional information is instead mediated and continuity enforced by autoregression. Since this only involves dependencies on observed vari-

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

ables, it is possible to devise autoregressive models that do not violate the Markov assumption, and linear autoregressive HMMs (AR-HMMs) [15] have previously been explored in HMM-based SPSS [16, 17, 18]. In this paper, we describe HMMs that, like Tacotron, use stronger, nonlinear AR models defined by a neural network.

2.2. Attention in TTS

In a typical sequence-to-sequence based TTS system, the attention mechanism is responsible for duration modelling and for learning to align input symbols with output frames during training. Watts et al. [9] found that the use of neural attention did not necessarily benefit TTS, and more suitable TTS attention mechanisms have recently been a focus of intense research. Only some of the relevant work can be surveyed here; please see [3] for additional references. He et al. [19] emphasised that TTS alignments should be *local* (each output frame is associated with a single input symbol), *monotonic* (never move backwards), and *complete* (not skip any speech sounds). HMMs are local by design, while the two other concepts map directly onto the classes of *left-right* and *no-skip* HMMs. Most neural TTS attention mechanisms do not satisfy these requirements [19, 3].

Many systems that do satisfy all three criteria rely on external tools for input-output alignment to obtain duration data (see [3] for a list), and do not jointly learn to speak and align, unlike vanilla HMMs or Tacotron 1/2. However, some proposals do learn to speak and align without external tools, mostly (e.g., [20, 21, 22, 23, 24, 25]) by introducing duration models into neural TTS, which will be our focus here. Many of these models only optimise a lower bound on the sequence likelihood, either due to the use of variational methods (e.g., Non-Attentive Tacotron [23] and the VQ-VAEs in [24]) or by not marginalising over all possible alignments (Glow-TTS [22]). By using a mean squared error (MSE) duration loss, Glow-TTS also implicitly treats the positive, integer-valued durations (frame counts) as outcomes from a Gaussian distribution on the real line, which violates probabilistic assumptions. Our proposal avoids these issues.

AlignTTS [21] is more similar to an HMM and uses a variant of the HMM forward recursions [15], but requires a complex, four-stage training procedure that culminates in training a separate, non-probabilistic duration predictor that is used at synthesis time. AlignTTS is also parallel, while our proposal is autoregressive.

The constant-per-state transition probability of vanilla HMMs implicitly describes a geometric duration distribution, which is a poor fit for natural speech [26, 27]. A solution to this in SPSS was to introduce explicit duration modelling through *hidden semi-Markov models* (HSMMs) [26]. These sacrifice the Markovian property to describe more general duration distributions, by letting transition probabilities depend on the time spent in the current state. Independent, concurrent work [25] proposes to integrate HSMMs into neural TTS, obtaining better results than Tacotron 2 in a small-data experiment, but uses a variational approximation and again assumes a Gaussian distribution for the positive-integer frame durations. In contrast, [27] described how arbitrary discrete duration distributions can be parameterised implicitly via frame-dependent transition probabilities, and then predicted jointly with output frames in a single, joint model of durations and acoustics. This paper combines this idea with autoregression acting as an "acoustic memory" of the time spent in a state, to obtain a fully probabilistic model with general durations that can be trained efficiently on the exact log-likelihood.

The most similar work to ours is SSNT-TTS [20], which essentially describes a neural HMM for TTS, albeit under another name. We differ in applying an HMM perspective to the approach, integrating more SPSS ideas to improve our system, using a different duration-generation method, in demonstrating control over speaking rate, and in reporting better TTS quality, on par with Tacotron 2.

3. METHOD

We now (in Sec. 3.1 and Fig. 1) describe the key modifications used to put HMMs into neural TTS, specifically Tacotron 2. Sec. 3.2 then describes how ideas and implementation aspects from classic HMM-based TTS can be adapted to further improve neural HMM TTS.

3.1. Replacing attention with neural HMMs

The location-sensitive attention [28] used by Tacotron 2 is a function that uses information from previously-generated acoustic frames $\boldsymbol{x}_{1:t-1}$ to select which encoder output vector(s) $\boldsymbol{h}_l$ to pass on to the decoder in order to generate the next frame $\boldsymbol{x}_t$. (We use bold font for vector-valued quantities and index input sequence symbols by n and output frames by t.) The attention also has an internal state, in the form of previous attention weights $\alpha_{1:t-1,n}$. The procedure to generate one frame t of output using Tacotron 2 can be written as

$$\boldsymbol{a}_t = \mathrm{LSTM}(\mathrm{PreNet}(\boldsymbol{x}_{t-1}), \boldsymbol{g}_{t-1}, \boldsymbol{a}_{t-1}) \tag{1}$$

$$e_{t,n} = \boldsymbol{\omega}^\intercal \tanh\left(\boldsymbol{W}\boldsymbol{a}_t + \boldsymbol{V}\boldsymbol{h}_n + \boldsymbol{U}(\boldsymbol{F} * \textstyle\sum_{t'<t} \alpha_{t',n}) + \boldsymbol{b}\right) \tag{2}$$

$$\alpha_{t,n} = \exp(e_{t,n}) \,/\, \textstyle\sum_{n'} \exp(e_{t,n'}) \tag{3}$$

$$\boldsymbol{g}_t = \textstyle\sum_n \alpha_{t,n} \boldsymbol{h}_n \qquad (\boldsymbol{x}_t, \tau_t) = \mathrm{Decoder2}(\boldsymbol{g}_t, \boldsymbol{a}_t). \tag{4}$$

Here, $\boldsymbol{a}_{t-1}$ represents the hidden and cell state variables of the first decoder LSTM, $\mathrm{Decoder2}$ is the upper part of the decoder in Fig. 1a (which contains another LSTM state $\boldsymbol{a}'_t$), while $\tau_t \in [0,\,1]$ is the *stop token*. The latter is an estimate of the probability that the current frame is the last in the utterance, terminating synthesis if $\tau_t > 0.5$.

Our proposal is to remove the dependence on $\boldsymbol{g}_{t-1}$ from Eq. (1), and replace the remaining equations by a probabilistic upper decoder that estimates the distribution of the next frame $\boldsymbol{x}_t$ (by outputting the parameters $\boldsymbol{\theta}_t$ of an HMM emission distribution $\boldsymbol{o}(\boldsymbol{\theta})$) and turns the stop token into a *transition probability* $\tau_t \in [0,\,1]$ for the state $s_t \in \{1,\,\ldots,\,N\}$ (with $s_1 = 1$). Eqs. (2)–(4) are replaced by

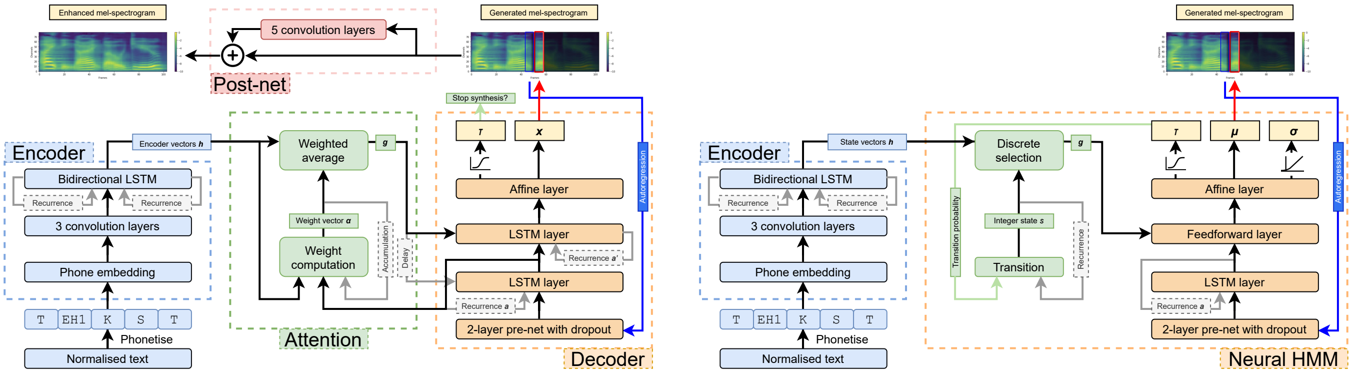
$$\boldsymbol{g}_t = \boldsymbol{h}_{s_t} \qquad (\boldsymbol{\theta}_t, \tau_t) = \mathrm{Decoder2}(\boldsymbol{g}_t, \boldsymbol{a}_t) \tag{5}$$

$$\boldsymbol{x}_t \sim \boldsymbol{o}(\boldsymbol{\theta}_t) \qquad s_{t+1} = s_t + \mathrm{Bernoulli}(\tau_t), \tag{6}$$

where $\mathrm{Bernoulli}(p)$ is a binary random variable on $\{0,\,1\}$ that equals 1 with probability p. The attention state variables $\alpha_{t,n}$ of Tacotron 2 have thus been replaced by a single, integer state variable s_t that evolves stochastically based on τ_t. This transition probability depends on the $\boldsymbol{h}$-vector of the current state s_t (through $\boldsymbol{g}_t$) and on the entire previous acoustics $\boldsymbol{x}_{1:t-1}$ (through $\boldsymbol{a}_t$), so it can be different for every frame t even for the same state. This can model arbitrary duration distributions [27]. $s_t > N$ terminates synthesis.

The end result is a left-right no-skip *neural HMM*, an AR-HMM parameterised by the decoder network in Fig. 1b. The encoder turns each input sequence into a unique HMM, where each vector $\boldsymbol{h}_n$ represents a state. Feeding this state vector and the AR input $\boldsymbol{x}_{1:t-1}$ into the decoder yields the HMM emission distribution $\boldsymbol{o}(\boldsymbol{\theta}_t)$ and next-state transition probability τ_t of state n at time t. Neural HMMs were first described concurrently by [10] and [11], the latter under the name *segment-to-segment neural transduction* (SSNT).

For the model to be a proper HMM satisfying the Markov property, $(\boldsymbol{\theta}_t,\,\tau_t)$ must not depend on anything other than the current state s_t (through the state vector $\boldsymbol{g}_t$) and the past observations $\boldsymbol{x}_{1:t-1}$. This necessitates an additional change to the Tacotron 2 architecture, namely removing the recurrence inside $\mathrm{Decoder2}$ by



(a) Nvidia Tacotron 2 implementation

(b) Neural HMM TTS (proposed)

Fig. 1: Synthesis-time architecture diagrams. Recurrences, delays, and the cumulative attention in Eq. (2) are drawn as grey arrows.

changing its LSTM layer to a feedforward layer, since an LSTM would propagate a dependence on past hidden states. This change also substantially reduces the number of parameters in the model.

Finally, the full Tacotron 2 architecture incorporates a non-causal convolutional *post-net* that enhances the initial AR-generated mel-spectrogram in a residual setup. This resembles post-filtering and global variance compensation [29] in classic SPSS. However, the non-invertibility of the Tacotron post-net makes it incompatible with likelihood-based modelling. A post-net can be added, but must either be trained separately like in [30], or be invertible like in [22]. We leave this as future work, and instead evaluate our proposal against Tacotron 2 output from both before and after the post-net.

3.2. Practical considerations

Numerical stability: When working with HMMs, it is crucial for numerical precision to perform all computations in the logarithmic domain using the “log-sum-exp trick”. Since zeroes in these computations map to $\ln 0 = -\infty$ in the log domain, care must be taken to avoid `NaN` gradients in deep-learning frameworks like PyTorch.

Like classic HMM-based TTS [31], we chose to use diagonal-covariance Gaussian emission distributions $\boldsymbol{o}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ in this work. We also used softplus (not exponential) nonlinearities for $\boldsymbol{\sigma}$, with a non-zero minimum value (“variance flooring”), here clamped at 0.001, since this has been important in other generative models.

Architecture enhancements: Tacotron 2 can represent intermediate states using soft attention, since the $\alpha_{t,n}$-values have many degrees of freedom. Major HMM-based synthesisers [31, 13] instead use 5 sub-states per input phone and run at 200 fps. Tacotron 2 runs at 80 fps, i.e., 40% the framerate, hence we use 2 states per phone to get the same time resolution as these HMMs. This is implemented by doubling the size of the decoder output layer and interpreting its output as two concatenated state vectors $\boldsymbol{h}$ for each phone.

Classic HMM-based TTS includes a model of dependencies between adjacent frames to promote temporally smooth output [31, 16, 13]. Although Tacotron 2 and the neural HMMs in this article only take the latest frame x_{t-1} as AR input, the LSTM in Eq. (1) means they can remember information arbitrarily far back which is beneficial for modelling utterance-level prosody. We also treat $\boldsymbol{x}_0$, the initial AR context (the “go token”) as a learnable parameter.

Initialisation: HMMs are often initialised using a *flat start*, in which all states have the same statistics [32]. By zeroing out all weights in the decoder output layer but initialising other layers as normal, all states will have the same output (zero), but different and nonzero gradients, thus enabling learning [33]. By choosing the last-layer biases, we can enforce $\boldsymbol{\mu} = \boldsymbol{0}$ and $\boldsymbol{\sigma} = \boldsymbol{1}$ at the start of training, which matches the global statistics of our normalised data.

Training: Neural HMM training [10] is a hybrid of old and new: We use the classic (scaled) forward algorithm [15] to compute the exact sequence log-likelihood, but then leverage backpropagation and automatic differentiation to optimise it (here using Adam [34]). These parts correspond to the E step and the M step of the (generalised) EM algorithm [35], respectively. Computations during training parallelise over the states but, like Tacotron 2, are sequential across time due to the temporal recurrences.

Maximum-likelihood estimation of linear AR models can lead to unstable models [17, 16]. A similar problem exists for nonlinear, autoregressive neural TTS [3]. Tacotron 2 works around this by adding dropout to the pre-net, and we retain that solution here.

Synthesis: We can iteratively use the equations in Sec. 3.1 and randomly sample new frames $\boldsymbol{x}_t \sim \boldsymbol{o}(\boldsymbol{\theta}_t)$. However, HMM-based TTS generally benefits from deterministically generating typical output rather than random sampling [36, 37]. For acoustics, this is done by generating the most probable output sequence [14], which is the same as the mean $\boldsymbol{\mu}_t$ when $\boldsymbol{o}(\boldsymbol{\theta}_t)$ is Gaussian. By iteratively taking $\boldsymbol{x}_t = \boldsymbol{\mu}_t$ (red arrow in Fig. 1b), we obtain a greedy approximation of [14]. This is closely related to Tacotron 2 output generation, since it is trained using the MSE, which is minimised by the mean $\mathbb{E}[\boldsymbol{X}_t]$.

SSNT-TTS found that randomly sampling transitions led to poor pause durations [20], and classic HMM-based systems typically base the time in each state at synthesis on the mean duration of the state [26]. That is not compatible with duration distributions implicitly defined through transition probabilities τ_t, as here. We instead use the simple algorithm from [27, 38] for deterministic duration generation based on duration quantiles (e.g., the median rather than the mean). Changing the quantile controls speaking rate. For the models evaluated in this paper, informal listening showed that deterministic generation of acoustics and durations both led to clear quality improvements; examples are provided on the webpage.

4. EXPERIMENTS

To validate our proposal to use neural HMMs for TTS, we performed a number of experiments (including a subjective listening test), comparing our proposal to a maximally similar Tacotron 2 [7] system. Synthetic speech examples from the different experiments can be found at https://shivammehta007.github.io/Neural-HMM/.

We based our systems on the PyTorch [39] open-source Nvidia

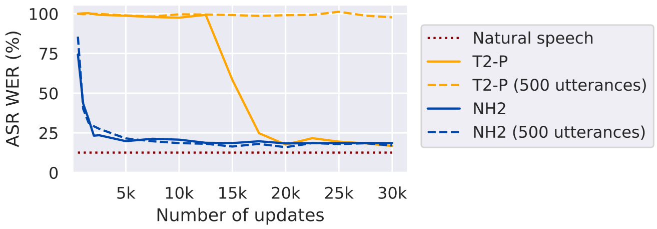


Fig. 2: Average utterance ASR WER of validation-set resynthesis.

implementation[1] of Tacotron 2, and trained them on the LJ Speech dataset [40], comprising utterances (normalised text and matching audio) adapted from free audiobooks read by a female speaker of US English. We used the default train/val/test split in the repository, which designates about 23 h of audio for training. We likewise used the default text-processing, including the pronouncing dictionary (CMUDICT), since this generally benefits neural TTS [41, 42]. Output features were normalised to zero mean and unit variance over the training data, and waveforms were generated using the default, pre-trained v5 "universal" WaveGlow [12] vocoder.[2]

We trained three systems: one Tacotron 2 baseline (T2) and two neural HMM systems, with either two (**NH2**) or one (**NH1**) state per phone. We expect NH2 to perform the best, with NH1 functioning as an ablation. All systems used the same architecture and hyperparameters (layer widths, learning rates, etc.) as the repository defaults, except that the size of the decoder output vectors was doubled to 1024 in the two-state system, since the decoder output now represents two concatenated state vectors. From the single Tacotron 2 baseline system, we synthesised two outputs: **T2+P**, using the full mel-spectrogram output after the post-net, and **T2−P**, using the initial mel-spectrogram prior to post-net enhancement, which is directly comparable to our neural HMMs. Model sizes for the different setups are listed in Table 1. We see that both neural HMMs are significantly smaller that Tacotron 2, even if the post-net is removed.

Each system was trained for 30k mixed-precision updates on 7 GPUs using a batch size of 6. It took approximately 14.5k updates for T2 to learn to speak coherently, whereas neural HMMs were intelligible after 1k updates. Fig. 2 graphs how the Google ASR word error rate (WER) of synthesising the 100 validation utterances evolves during training, including results from training on a small subset (500 utterances) of the data. Audio of synthesised speech during training is also provided on our demo webpage. We see that NH2 rapidly learns to speak intelligibly in both cases, much faster than Tacotron 2, which does not learn to speak at all on the smaller dataset. Even after the WER stabilised, we could consistently reproduce the effect where Tacotron 2 (including the best pre-trained system made available by Nvidia) degenerates into unintelligible babbling on long and short sentences, with examples provided on our webpage.

Both training and synthesis used pre-net dropout, as is the Tacotron 2 default [7], else attention breaks down. Our neural HMMs retained this dropout, since it improved subjective quality in informal listening. Audio examples without it are provided on our webpage.

Because natural-speech phone duration distributions are skewed to the right, and the median lies between the mode and the mean for skewed distributions, median-based output generation often generates speech that on average is faster than that in the training database; cf. [37]. Following the proposal in [38], the transition threshold of the deterministic duration-generation procedure was tuned to make the speaking rate of the NH systems match T2. The resulting threshold-quantile values were 0.57 for the two-state model and 0.45 for the single-state model. Our webpage provides examples of speech generated with different threshold quantiles, demonstrating control over speaking rate at synthesis time.

Type	Tacotron 2		Neural HMM	
Condition	T2+P	T2−P	NH2	NH1
Size	28.2M	23.8M	15.3M	12.7M
MOS	3.41±0.01	3.25±0.01	3.24±0.01	2.68±0.01

Table 1: Models from the experiments, with number of parameters and mean opinion scores (with 95% confidence intervals) for each.

We conducted a subjective listening test to evaluate the naturalness of speech generated by the four conditions in Table 1. In the test, participants were presented with four parallel stimuli at a time, one from each condition (unlabelled and in random order), all speaking the same sentence. Participants were asked to rate the naturalness of each stimulus on an integer scale from 1 (worst) to 5 (best), anchored using the classic MOS labels "Bad" through "Excellent" [43]. Stimuli were drawn from a pool of 9 sets of Harvard sentences [44], which are sets of 10 sentences each, designed so that each set is approximately phonetically balanced. All stimuli were loudness normalised to −20 dB LUFS following EBU R128 [45]. We manually verified that no T2 stimuli exhibited babbling due to failed attention.

We used Prolific to recruit 30 test participants ages 21 through 70, all self-reported native English speakers from UK, Ireland, USA, Canada, Australia, and New Zealand. Each participant rated 3 randomly selected sets of 10 Harvard sentences. All participants reported wearing headphones for the test. A completed test was rewarded with 3.50 GBP, with the average completion time being 17 minutes. This gave 3600 total ratings, 900 for each condition we evaluated.

The mean opinion scores (MOS) from the test are reported in Table 1, together with 95% confidence intervals based on a Gaussian approximation. Pairwise t-tests find all conditions to be significantly different (with $p<10^{-3}$) except NH2 and T2−P ($p>0.98$), whose respective mean opinion scores differ by less than 0.002 before rounding. We can conclude that the proposed neural HMM TTS (NH2), despite being simpler and lighter, achieved a naturalness on par with the most comparable Tacotron 2 condition (T2−P). This was not achieved by SSNT-TTS [20]. Neural HMMs were found to benefit from using two states per phone (NH2 vs. NH1), while Tacotron 2 improved from the use of a post-net (T2+P vs. T2−P).

5. CONCLUSION AND FUTURE WORK

We have described how classical and contemporary TTS can be combined to obtain a fully probabilistic, attention-free sequence-to-sequence model based on neural HMMs. The resulting system is smaller than Tacotron 2, yet achieves comparable naturalness, learns to speak and align faster, needs less data, and does not babble. To our knowledge, this is the first time an HMM-based system demonstrates a speech quality matching prior neural TTS. The neural HMMs also permit easy control over the speaking rate of the synthetic speech.

Future work includes stronger network architectures, e.g., based on transformers [46], and/or a separate post-net like in [30]. It also seems compelling to combine neural HMMs with powerful distribution families such as normalising flows, either replacing the Gaussian assumption (as done for non-neural HMMs in [47]) or as a probabilistic post-net like in [22]. This may allow the naturalness of sampled speech to surpass that of deterministic output generation.

[1] https://github.com/NVIDIA/tacotron2/
[2] https://github.com/NVIDIA/waveglow/

6. REFERENCES

[1] H. Zen, K. Tokuda, and A. W. Black, “Statistical parametric speech synthesis,” *Speech Commun.*, vol. 51, no. 11, pp. 1039–1064, 2009.

[2] Z.-H. Ling, S.-Y. Kang, H. Zen, A. Senior, M. Schuster, X.-J. Qian, et al., “Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends,” *IEEE Signal Proc. Mag.*, vol. 32, no. 3, pp. 35–52, 2015.

[3] X. Tan, T. Qin, F. Soong, and T.-Y. Liu, “A survey on neural speech synthesis,” *arXiv preprint arXiv:2106.15561*, 2021.

[4] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, et al., “WaveNet: A generative model for raw audio,” *arXiv preprint arXiv:1609.03499*, 2016.

[5] W. Wang, S. Xu, and B. Xu, “First step towards end-to-end parametric TTS synthesis: Generating spectral parameters with neural attention,” in *Proc. Interspeech*, 2016, pp. 2243–2247.

[6] Y. Wang, RJ Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, et al., “Tacotron: Towards end-to-end speech synthesis,” in *Proc. Interspeech*, 2017, pp. 4006–4010.

[7] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, et al., “Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions,” in *Proc. ICASSP*, 2018, pp. 4799–4783.

[8] O. Watts, G. E. Henter, T. Merritt, Z. Wu, and S. King, “From HMMs to DNNs: where do the improvements come from?,” in *Proc. ICASSP*, 2016, pp. 5505–5509.

[9] O. Watts, G. E. Henter, J. Fong, and C. Valentini-Botinhao, “Where do the improvements come from in sequence-to-sequence neural TTS?,” in *Proc. SSW*, 2019, pp. 217–222.

[10] K. M. Tran, Y. Bisk, A. Vaswani, D. Marcu, and K. Knight, “Unsupervised neural hidden Markov models,” in *Proc. Workshop on Structured Prediction for NLP*, 2016, pp. 63–71.

[11] L. Yu, J. Buys, and P. Blunsom, “Online segment to segment neural transduction,” in *Proc. EMNLP*, 2016, pp. 1307–1316.

[12] R. Prenger, R. Valle, and B. Catanzaro, “WaveGlow: A flow-based generative network for speech synthesis,” in *Proc. ICASSP*, 2019.

[13] Z. Wu, O. Watts, and S. King, “Merlin: An open source neural network speech synthesis system,” in *Proc. SSW*, 2016, pp. 218–223.

[14] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, “Speech parameter generation algorithms for hmm-based speech synthesis,” in *Proc. ICASSP*, 2000, pp. 1315–1318.

[15] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proc. IEEE*, vol. 77, no. 2, 1989.

[16] M. Shannon, H. Zen, and W. Byrne, “Autoregressive models for statistical parametric speech synthesis,” *IEEE T. Audio Speech*, vol. 21, no. 3, pp. 587–597, 2013.

[17] C. Quillen, “Autoregressive HMM speech synthesis,” in *Proc. ICASSP*, 2012, pp. 4021–4024.

[18] X. Wang, S. Takaki, and J. Yamagishi, “An autoregressive recurrent mixture density network for parametric speech synthesis,” in *Proc. ICASSP*, 2017, pp. 4895–4899.

[19] M. He, Y. Deng, and L. He, “Robust sequence-to-sequence acoustic modeling with stepwise monotonic attention for neural TTS,” *Proc. Interspeech*, pp. 1293–1297, 2019.

[20] Y. Yasuda, X. Wang, and J. Yamagishi, “Initial investigation of encoder-decoder end-to-end TTS using marginalization of monotonic hard alignments,” in *Proc. SSW*, 2019, pp. 211–216.

[21] Z. Zeng, J. Wang, N. Cheng, T. Xia, and J. Xiao, “AlignTTS: Efficient feed-forward text-to-speech system without explicit alignment,” in *Proc. ICASSP*, 2020, pp. 6714–6718.

[22] J. Kim, S. Kim, J. Kong, and S. Yoon, “Glow-TTS: A generative flow for text-to-speech via monotonic alignment search,” in *Proc. NeurIPS*, 2020, pp. 8067–8077.

[23] J. Shen, Y. Jia, M. Chrzanowski, Y. Zhang, I. Elias, H. Zen, et al., “Non-Attentive Tacotron: Robust and controllable neural TTS synthesis including unsupervised duration modeling,” *arXiv preprint arXiv:2010.04301*, 2020.

[24] Y. Yasuda, X. Wang, and J. Yamagishi, “End-to-end text-to-speech using latent duration based on VQ-VAE,” in *Proc. ICASSP*, 2021.

[25] Y. Nankaku, K. Sumiya, T. Yoshimura, S. Takaki, K. Hashimoto, K. Oura, et al., “Neural sequence-to-sequence speech synthesis using a hidden semi-Markov model based structured attention mechanism,” *arXiv preprint arXiv:2108.13985*, 2021.

[26] H. Zen, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, “Hidden semi-Markov model based speech synthesis,” in *Proc. SLP*, 2004.

[27] S. Ronanki, O. Watts, S. King, and G. E. Henter, “Median-based generation of synthetic speech durations using a non-parametric approach,” in *Proc. SLT*, 2016, pp. 686–692.

[28] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” in *Proc. NIPS*, 2015.

[29] T. Toda and K. Tokuda, “A speech parameter generation algorithm considering global variance for HMM-based speech synthesis,” *IEICE T. Inf. Syst.*, vol. 90, no. 5, pp. 816–824, 2007.

[30] H. Tachibana, K. Uenoyama, and S. Aihara, “Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention,” in *Proc. ICASSP*, 2018, pp. 4784–4788.

[31] H. Zen, T. Nose, T. Masuko, A. W. Black, and K. Tokuda, “The HMM-based speech synthesis system (HTS) version 2.0,” in *Proc. SSW*, 2007.

[32] S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, et al., *The HTK Book (for HTK Version 3.2)*, 2002.

[33] H. Zhang, Y. N. Dauphin, and T. Ma, “Fixup initialization: Residual learning without normalization,” in *Proc. ICLR*, 2019.

[34] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. ICLR*, 2015.

[35] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *J. Roy. Stat. Soc. B*, vol. 39, no. 1, pp. 1–38, 1977.

[36] G. E. Henter, T. Merritt, M. Shannon, C. Mayo, and S. King, “Measuring the perceptual effects of modelling assumptions in speech synthesis using stimuli constructed from repeated natural speech,” in *Proc. Interspeech*, 2014, pp. 1504–1508.

[37] G. E. Henter, S. Ronanki, O. Watts, M. Wester, Z. Wu, and S. King, “Robust TTS duration modelling using DNNs,” in *Proc. ICASSP*, 2016.

[38] G. E. Henter, S. Ronanki, O. Watts, and S. King, “Non-parametric duration modelling for speech synthesis with a joint model of acoustics and duration,” in *IEICE Tech. Rep.*, 2017, number 414, pp. 11–16.

[39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, et al., “PyTorch: An imperative style, high-performance deep learning library,” in *Proc. NeurIPS*, 2019, pp. 8024–8035.

[40] K. Ito and L. Johnson, “The LJ Speech dataset,” https://keithito.com/LJ-Speech-Dataset/, 2017.

[41] J. Fong, J. Taylor, K. Richmond, and S. King, “A comparison between letters and phones as input to sequence-to-sequence models for speech synthesis,” in *Proc. SSW*, 2019, pp. 223–227.

[42] É. Székely, G. E. Henter, J. Beskow, and J. Gustafson, “Spontaneous conversational speech synthesis from found data,” in *Proc. Interspeech*, 2019, pp. 4435–4439.

[43] ITU-T, “Methods for subjective determination of transmission quality,” ITU Recommendation ITU-T P.800, 1996.

[44] E. H. Rothauser et al., “IEEE recommended practice for speech quality measurements,” *IEEE T. Acoust. Speech*, vol. 17, no. 3, pp. 225–246, 1969.

[45] EBU, “Loudness normalisation and permitted maximum level of audio signals,” EBU Recommendation EBU R 128v4, 2020.

[46] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, “Neural speech synthesis with transformer network,” in *Proc. AAAI*, 2019, pp. 6706–6713.

[47] A. Ghosh, A. Honoré, D. Liu, G. E. Henter, and S. Chatterjee, “Normalizing flow based hidden Markov models for classification of speech phones with explainability,” *arXiv preprint arXiv:2107.00730*, 2021.