Incorporating Duration and Style Prediction to Deep Neural Network-based Sentence-level Controllable Speech Synthesis

Kurt Ahn

MSc in Speech and Language Processing
The University of Edinburgh
2018

Contents

Abstract

This paper presents a comprehensive text-to-speech system capable of enriching the speech output with a style specified by the user. Building upon an existing system, which only includes an acoustic model, we have incorporated a duration model and a sentence encoder, which is used to predict the style most appropriate for the text input, so as to provide a relevant point of reference to the user. The duration model, following the architecture of the acoustic model, is a typical feedforward deep neural network, augmented with an input projection layer that learns acoustic variations in recordings of training sentences not captured in the text. The sentence encoder, of the recurrent neural network architecture typically used in machine translation, condenses the token embeddings of a training sentence into a single vector, which is then projected onto the style space learned by the acoustic model.

Index terms— text-to-speech, controllable speech synthesis, semantics-to-style mapping, duration model, deep neural network

Chapter 1

Introduction

Text-to-speech (TTS) synthesis refers to the task of artificially generating speech given some text. Existing TTS systems are capable of producing intelligible speech, but human-like naturalness has been more difficult to achieve. One way to improve naturalness of a TTS system is to give it the ability to enrich speech output in qualities like speaking style and emotional content (Murray, Arnott, & Rohwer, 1996). The first step to achieving this goal is to create a system that allows the user to manually control such characteristics. The focus of this paper is to improve upon an existing system capable of controlling speaking style proposed by (Watts, Wu, & King, 2015). What we mean by "improve" will become clear in Chapter 2. In this chapter, we aim to provide a brief primer to the development in the field of TTS leading up to the aforementioned system, which will aid in understanding the discussion that ensues.

1.1 Statistical Parametric Speech Synthesis

A TTS system typically consists of a front end and a back end. The front end is responsible for extracting from the source text linguistic features, which are used by the back end to generate a waveform appropriate for the text. Such features may include the class of the current or a neighbouring phoneme, the position of the current syllable in the current word, the number of words in the text, and many more.

Broadly speaking, TTS techniques can be categorized into three groups: model-based, concatenative and statistical parametric. In statistical parametric speech synthesis (SPSS), which is the focus of this paper, linguistic features extracted from text are used to generate acoustic parameters of speech, such as spectral coefficients and F_0 values. These parameters are then fed to a vocoder, which can convert the parameters into waveform and vice versa. In SPSS, in order to map linguistic features to acoustic parameters, a probability distribution of acoustic parameters given linguistic features is required. This distribution is not known ahead of time but is rather learned from a dataset of pairs of linguistic features and acoustic parameters, extracted from natural speech and its transcription, respectively.

1.1.1 Hidden Markov Model Based Synthesis

Traditionally, SPSS was synonymous with hidden Markov model (HMM) based synthesis. HMM is a type of finite state automaton, whose states are not directly observable but instead “emit” observable values according to some probability distribution. In the context of speech synthesis, each state in an HMM corresponds to a particular linguistic feature configuration, typically referred to as phonetic context, and the emission distribution corresponds to the conditional distribution of acoustic parameters given the context. An alternative way to view HMM-based synthesis, which will be useful when discussing deep neural network based synthesis in the next section, is that there is a single piecewise emission distribution, which takes a feature vector as input. The first step in HMM-based synthesis is to convert the linguistic feature vector stream representing the source text into an HMM, by mapping each feature vector to a state and concatenating the states in a left-to-right fashion. Typically, a phoneme is divided into several, for instance, five parts, and each part is mapped to a single state. Then, starting from the left-most state, the HMM is traversed toward the right-most state, generating acoustic parameters at each time step according to the emission distribution of the currently occupied state (Zen, Tokuda, & Black, 2009). Note that, since each state is in a linguistic unit of speech rather than some temporal unit, the HMM may occupy a state more than once and emit multiple frames of acoustic parameters from it before moving onto the next state. How many frames are emitted by a state may be determined by the probability of transitioning back to the current state, but this provides a poor model for duration, and typically, a separate model is used for predicting the duration of each state.

Although it is theoretically possible to create a unique set of parameters to describe the distribution for every state, with a reasonably large binary feature vector of, say, 100 dimensions, a total of 5×2^{100} sets of parameters would be required. As it would be implausible to create and train such a large HMM, some way of allowing multiple similar states to share parameters is necessary. This can be done with the use of a decision tree. A decision tree shows how a group of objects can be identified by repeatedly asking yes/no questions. In the context of HMM-based synthesis, the root of such a tree would represent all possible states. The group of states represented by the root node can be split into two sub-groups by asking, for instance, the following: *Is the current phone a vowel?* We can place all states for which the answer is *yes* to one of the children nodes and the rest into the other child, and repeat the process for the two children nodes until some stopping criterion (e.g. minimum number of states associated with a node) is reached.

1.1.2 Deep Neural Network Based Synthesis

As (Zen, Senior, & Schuster, 2013) point out, speech quality of an HMM synthesizer is compromised by the usage of decision trees. This is because decision trees divide the training set into smaller sets of samples, and each sample set is used to train only the parameters of a single state cluster. As an alternative to HMM-based synthesis, the authors propose using deep neural networks (DNNs). A DNN, a neural network with three or more layers, is better at modelling complex functions than a decision tree and can use an entire dataset to train all its parameters. In DNN-based synthesis, acoustic parameters for each frame are produced by performing a forward pass through a DNN with the feature vector for that frame. As in the case of HMM-based synthesis, the front end produces feature vectors in some arbitrary sub-phonetic unit akin to an HMM state, and these need to be mapped into time scale before they can be fed to the DNN. This mapping can be performed with an additional DNN that determines how many frames should be converted from each state. Because of the similarity with HMM-based synthesis, DNN-based synthesis can be interpreted as a modified HMM-based synthesis with a non-piecewise emission distribution function.

1.2 Controllable Speech Synthesis

Controllable speech synthesis (CSS) refers to the task of generating speech while controlling aspects of speech beyond those defined by the text input such as speaking style, emotion and speaker characteristics. Whereas a non-controllable system, which only accepts text input, can generate exactly one waveform for a given text, a controllable system can produce multiple (potentially infinite) waveforms for a given text and requires an additional input to determine which of the numerous waveforms to generate. CSS is an important area of research, as it has the potential to not only make interactions with machines more pleasant but also improve intelligibility and efficiency of human-machine communication (Pitrelli et al., 2006).

There have been attempts to augment existing model-based, concatenative and statistical parametric synthesis techniques to incorporate controllability. In the context of HMM-based synthesis, (Yamagishi, Onishi, Masuko, & Kobayashi, 2003) proposed two simple methods of training a CSS system on a dataset of natural utterances, each labelled with the style it was spoken with (e.g. *rough*, *joyful*). The first method (style-dependent modelling) involves training a separate decision tree for each style, while in the second method (mixed style modelling), a single decision tree is constructed for all styles by treating style as a contextual parameter. While these methods can successfully generate speech with varied style, the range of style they can express is discrete and finite. (Nose, Yamagishi, Masuko, & Kobayashi, 2007) attempted to address this issue with a model that can interpolate between different styles. To create such a model, a separate model for each style is first trained, but the decision trees of the models are forced to share a common structure through the use of the shared tree clustering technique (Yamagishi et al., 2003). The decision trees are then merged into a single tree, whose each leaf node models a parametrised probability distribution that takes a *style control vector* as input. At synthesis, a user-specified style control vector, along with text, is fed to the model, and effectively queries the style space within the speech database.

(Watts et al., 2015) incorporated sentence-level style control to DNN-based synthesis with the usage of an input projection layer, whose job is to map each training utterance to the average style vector of the utterance. The output of the projection layer is concatenated with the linguistic feature vector of a particular frame of an utterance before being fed to the next layer of the network. During synthesis, the projection layer is detached from the network and the style vector is supplied directly by the user.

Although the systems proposed by (Nose et al., 2007) and (Watts et al., 2015) both require style control vector as input during synthesis, there is an important difference in the role the control vector plays in the two systems. Whereas, in (Yamagishi et al., 2003), the control vector is used to interpolate between several predetermined styles, in (Watts et al., 2015), it is used to plug a gap between the total variation in the training set acoustic space and the variation that can be explained with just the linguistic features alone. One main advantage of the approach by (Watts et al., 2015) over the previous approach is that it permits unsupervised learning of style and does not require assigning each training utterance with a style label, which can be both expensive and erroneous.

Chapter 2

System Description

In this chapter, we introduce the augmentations we have made to the system proposed by (Watts et al., 2015), henceforth referred to as the baseline. In Section 2.1, we discuss motivation for the present work by identifying some of the limitations of the baseline. Then, in Section 2.2, we provide a description of the architecture of the extended system as well as how to train it and use it to synthesize speech.

2.1 Limitations of the Baseline

2.1.1 Prediction of Duration

Perhaps the most glaring limitation of the baseline is that it lacks the ability to predict state durations from text and style input. Without a duration model, speaking rate, which is an important aspect of speaking style, cannot be controlled. Section 2.2.1 discusses how a duration model can be combined with the acoustic model.

2.1.2 Prediction of Style Inherent to Text

A rather subtle limitation of the baseline is that it cannot provide any point of reference for utterances not seen in training. The user of a CSS system may want to impose a particular style on only a few sentences out of many. At the same time, it may not be appropriate to apply the neutral, or average, style on all the rest of the sentences, as they would all have the exact same style. In Section 2.2.2, we explore the possibility of finding an appropriate style for a given sentence by composing a style vector from the constituent tokens.

2.1.3 Local Control

The baseline learns sentence-level style vectors from training utterances and is meant to only be used to synthesize speech with style input applied to all frames of an utterance. This means that the baseline provides no way of adding emphasis to a particular word in a sentence (so as to distinguish ***He** said that?* from *He said **that**?*) or change the style in the middle of a sentence, which can happen when quoting someone (e.g. *"Leave me alone," she screamed.*). In the present work, we do not attempt to resolve this issue, but instead, in Section 4.2, we provide some points to consider when tackling this issue in future research.

2.2 Augmentations

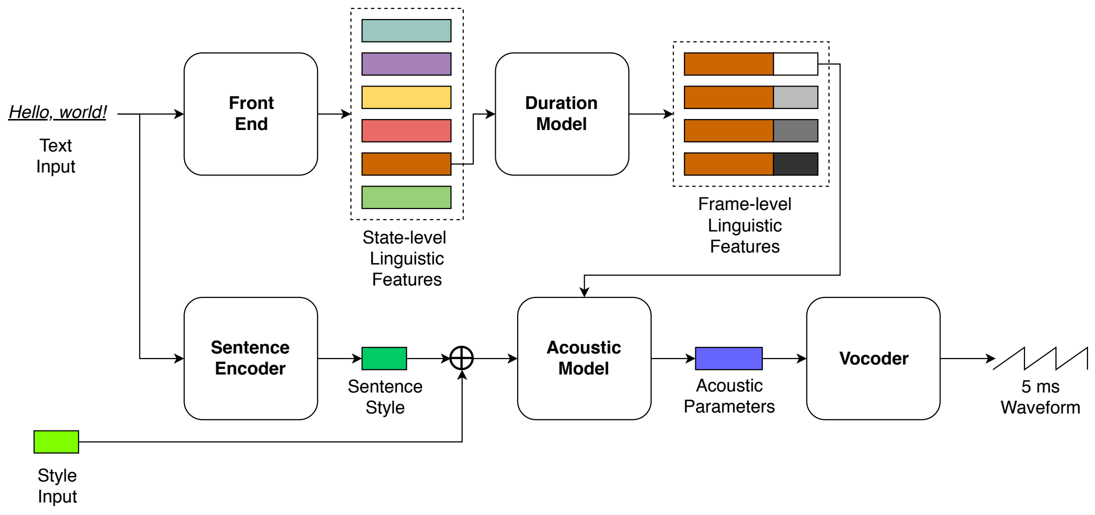

Figure 2.1: Generating a frame of speech with state durations predicted by the duration model and style predicted by the sentence encoder (Method C).

To deal with the limitations outlined in Sections 2.1.1 and 2.1.2, we have added a duration model and a sentence encoder to the baseline. Figure 2.1 illustrates how a frame of speech is generated using the full system. The system first finds an appropriate style vector for the text input using the sentence encoder and then applies the sum of the vector and the user-specified style offset to every frame of linguistic features fed to the acoustic model. The frame-level linguistic features are supplied by the duration model, which unpacks state-level features by predicting how many frames should be produced by a given state and then appending sub-state features, used to distinguish frames belonging to the same state from each other, to each frame.

2.2.1 Duration Model

Just like the acoustic model, the duration model can be implemented as a feedforward DNN with a projection layer that learns to map each training utterance to a style vector. Typically, in DNN-based synthesis, the duration model is trained separately from the acoustic model. If we were to follow that approach here, however, we would end up with two different projection layers for the two models. In other words, the models would map each training utterance to two different style vectors, effectively decoupling duration from other factors that affect speaking style. This could be problematic as it defeats the purpose of style control, which is not to control individual acoustic parameters separately, but to control them simultaneously and coherently so that predictable changes in style occur as the style input varies. Therefore, some way of allowing the two projection layers to share parameters is needed. In the present work, we present the following simple training method to enforce parameter sharing:

1. Train all layers of the acoustic model.
2. Initialize the projection layer of the duration model to that of the acoustic model.
3. Train all layers of the duration model except the projection layer.

There are many other ways to achieve the same goal, some of them perhaps more effective than the method proposed here, but exploration of approaches to parameter sharing is left for future research.

2.2.2 Sentence Encoder

Prediction of style from a sentence, described more precisely, is the mapping of the meaning of a sentence to a style appropriate for the sentence. The assumption that underlies our attempt at semantics-to-style mapping is that there is a link between semantics and style in speech. (Ben-David, Multani, Shakuf, Rudzicz, & van Lieshout, 2016) claim that prosody and semantics are not separable and

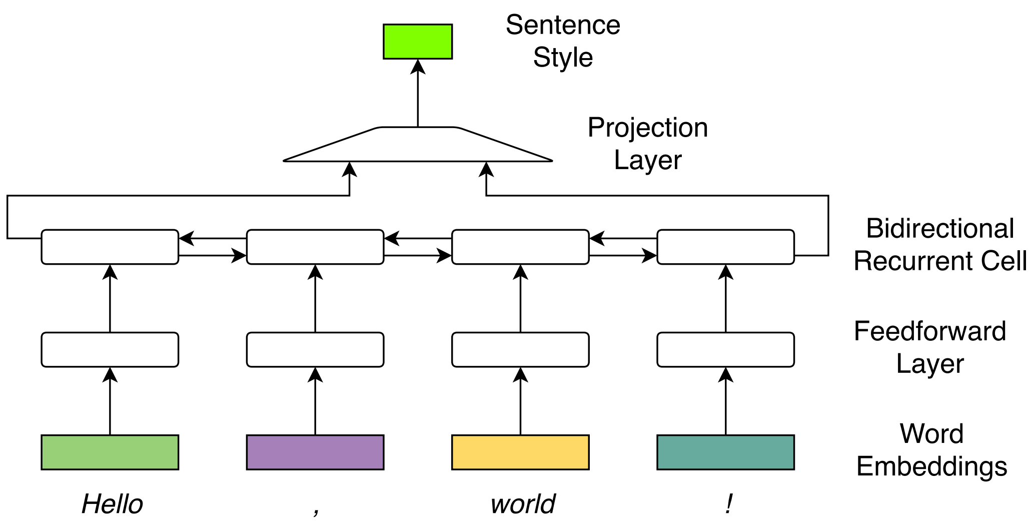

Figure 2.2: Computing the style for a sentence using a sentence encoder with one feed-forward layer and one bidirectional recurrent cell.

demonstrate that there is a preference for utterances with matching prosodic and semantic content to those with a prosody/semantics mismatch.

The meaning of a sentence can be constructed from the meaning of its constituent tokens. (Cho et al., 2014) introduced the recurrent neural network (RNN) encoder-decoder architecture to the field of machine translation. The encoder-decoder network consists of an encoder RNN, which condenses the source sentence token sequence to a vector representation, and a decoder RNN, which produces a token sequence in the target language from the encoded sentence representation. Taking inspiration from this work, for the purpose of semantics-to-style mapping, we propose the use of a bidirectional RNN encoder architecture like the one illustrated in Figure 2.2.

In order to compute the style of a sentence, the token sequence must be fed twice: once from left to right and once from right to left. Each time a token is fed to the network, the token is mapped to a vector that represents its meaning. Several models, such as word2vec (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) and GloVe (Pennington, Socher, & Manning, 2014), exist for obtaining such a mapping. The token vector can optionally be fed to one or more feedforward layers. The output of the final feedforward layer is then fed to one or more layers of bidirectional recurrent cells.

The encoder can be trained on a dataset that consists of token sequences of training sentences paired with the corresponding target style values. The target values can be sourced from the projection layer of a previously trained acoustic model.

Chapter 3

Experiments

In this chapter, we describe how we evaluated whether the augmentations made to the baseline have the intended effects. In Sections 3.1 and 3.2, we provide a full description of how the full system we tested was constructed. These sections are also meant to illustrate how the full system can be constructed from scratch and provides details that were intentionally left out in the previous chapter. Then, in Section 3.3, we describe the design of the two subjective evaluations we performed on the system.

3.1 Data

For the training and testing of the acoustic and duration models built for our experiments, the Blizzard Challenge 2017 dataset was used. The dataset consists of recordings from 56 children's books by a single speaker, as well as the transcripts for the recordings (King, Wilhborg, & Guo, 2017). After pruning utterances with transcripts that either were inaccurate or contained words not in the GloVe vocabulary (see below), 7,012 utterances, totalling 5.4 hours in duration, were left. Of these, 185 utterances were set aside for testing.

Using the front end of Merlin (Wu, Watts, & King, 2016), state-level alignment for each utterance was found. For duration modelling, 600 linguistic features were extracted from each state and were normalized to the interval [0.01, 0.99]. The normalized features were then paired with state durations, standardized over all utterances.

For acoustic modelling, state-level features were unpacked into frame-level features at a fixed rate of 5ms per frame. Each frame inherited the 600 features of the corresponding state and acquired nine sub-state features, which distinguish frames belonging to the same state from one another. As before, the linguistic features were normalized to [0.01, 0.99]. Then, MagPhase vocoder (Espic, Botinhao, & King, 2017) was used to extract acoustic parameters from the recordings. The vocoder was configured to produce 60 magnitude and 10 real and imaginary components per frame at a fixed frame rate of 5ms. For numerical stability, log F_0 values were exponentiated to the Hz scale. Then, unvoiced regions of the F_0 streams were linearly interpolated, and a binary flag, used to indicate voicing, was included in the list of 82 acoustic parameters. No delta or delta-delta parameters were used. After removing silent frames at the beginning and end of each recording, the standardized acoustic parameters were paired with the linguistic features of each frame.

For semantics-to-style mapping, a collection of 1.9 million 300-dimensional GloVe vectors pre-trained on the CommonCrawl dataset of 42 billion uncased tokens was used (Pennington et al., 2014). As mentioned before, any sentence with a token type not among the 1.9 million in the collection was pruned. The transcripts of the Blizzard Challenge recordings were tokenized using the tokenizer module of the Stanford Core NLP package (Manning et al., 2014). In order to ensure consistency between the GloVe vocabulary and the transcript tokens, the tokens were lowercased.

3.2 Training

All three modules of the proposed system were implemented from scratch using TensorFlow (Abadi et al., 2015), even though Merlin was used for preprocessing the dataset because Merlin is not flexible enough to readily accommodate less traditional architectures.

3.2.1 Acoustic Model

The optimal configuration for an acoustic model DNN with a control vector dimension of two was found to be six hidden layers with 256 nodes each. The configuration was determined through an exponential sweep of values between 16 and 1024 for the number of nodes per layer, and linear sweep of values between three and eight layers. Linear activation was used for the projection and output layers, while tanh activation was used for all other layers. Model parameters were initialized according to the method prescribed by (Glorot & Bengio, 2010).

95% of the available 3.5 million frames were allocated for training the model, and the rest were used for validation. Random allocation of data was performed in batches of fifty frames. The reason the frames were shuffled at the batch level and not the frame level was to minimize correlation between the training and validation sets by minimizing the number of instances where a training frame neighbours a validation frame in the original frame sequences. Our method of training is an improvement to the one employed by (Watts et al., 2015), in which training/validation split was performed at the sentence level, not at the frame level. This method of training is problematic as projection layer parameters associated with validation sentences are not updated during the training phase. This implies that validation loss is not an accurate reflection of the model's generalization performance.

Mini-batch training was performed with the mean absolute error (MAE) objective function and with L_2 regularization penalty of 10^{-5} applied to all non-bias weights except those in the projection layer. The decision to use the MAE objective over the mean squared error (MSE) objective, which was used by (Watts et al., 2015), was motivated by the fact that both the Blizzard Challenge dataset transcripts and linguistic feature set created with Merlin's front end were quite erroneous. It is well known that MSE is more sensitive to outliers than MAE is (Chai & Draxler, 2014), and an objective evaluation using distortion metrics showed an improvement in speech quality when the acoustic model was trained with the MAE objective.

All layers were trained with the Adam optimizer (Kingma & Ba, 2014), with an initial learning rate of 2×10^{-4}, which decayed exponentially at every iteration such that the learning rate at the end of an epoch was 90% of what it was at the beginning of the epoch. For the projection and output layers, the learning rate was multiplied by a factor of 10 and 2, respectively. The amplification of the projection layer learning rate, which was not performed by (Watts et al., 2015), was done in order to spread out the learned style vectors. The increased learning rate also helped reduce the validation error. Training was terminated at the 34th epoch when the validation error reached the minimum.

3.2.2 Duration Model

For duration modelling, a DNN of six hidden layers with 128 nodes each was created. This configuration was found with the same sweeping method used for the acoustic model. Just like the acoustic model, tanh activation was used for all layers except the projection and output layers, for which linear activation was applied. As per the plan described in Section 2.2.1, the projection layer parameters were initialized to those of the fully trained acoustic model and were fixed. Other parameters were initialized as per (Glorot & Bengio, 2010).

Similar to how the acoustic model was trained, mini-batch (batch size 20) optimization using Adam was performed on 95% of the 185,000 states with the MAE objective and 10^{-5} regularization penalty. An initial learning rate of 10^{-4} was exponentially decayed to 90% by the end of each epoch. The learning rate was equal for all trainable layers. Training was terminated after the 31st epoch when validation error stopped decreasing.

3.2.3 Sentence Encoder

The sentence encoder network was made of one feedforward layer of 300 nodes and one bidirectional gated recurrent unit (GRU)(Cho et al., 2014) cell of 300 nodes. Although smaller numbers of nodes were also tried, it appears that condensing word embedding vectors to low-dimensional vectors worsens encoder performance.

The embedding mapping, which converted each of the 4,787 token types appearing the dataset to an embedding vector, was initialized to the pre-trained GloVe values as mentioned in Section 3.1. Tanh activation was used for the feedforward layer, and the layer was initialized according to (Glorot & Bengio, 2010). The GRU cell, on the other hand, was activated with the rectified linear unit (ReLU) function and was initialized by the method proposed by (He, Zhang, Ren, & Sun, 2015).

The training set for the encoder was completed by pairing token sequences of training sentences to the corresponding style learned by the projection layer of the fully trained acoustic model. 95% of the 6,827 sentences were allocated for training and the rest for validation.

Similar to training the previous two models, Adam optimization on batches of 20 sentences was performed with the MAE objective. Since recurrent networks can be exposed to the exploding gradient problem (Pascanu, Mikolov, & Bengio, 2013), gradient was clipped such that the absolute value is at most 5. Regularization penalty of 10^{-5} was applied to the non-bias parameters of the feedforward layer. Initial learning rate of 5×10^{-5} with a per-epoch decay rate of 90% was applied equally to all layers, including the embedding layer.

To compensate for the scarcity of data and to account for the fact that a wide range of style is acceptable for a given text, each training example was fed ten times, each time with a small Gaussian noise with a standard deviation of 0.01 added to the target value. Training ended after four epochs when the validation error was minimum.

3.3 Experiment Design

Here we describe the two listening experiments that were designed and conducted in order to determine the effects the duration model and the sentence encoder had on controllability.

22 fluent English speakers with no known hearing defects participated in the experiments. Each subject was asked to complete a two-part online questionnaire in a sound-proof booth, listening to synthesized waveforms with headsets.

3.3.1 Effect of the Duration Model

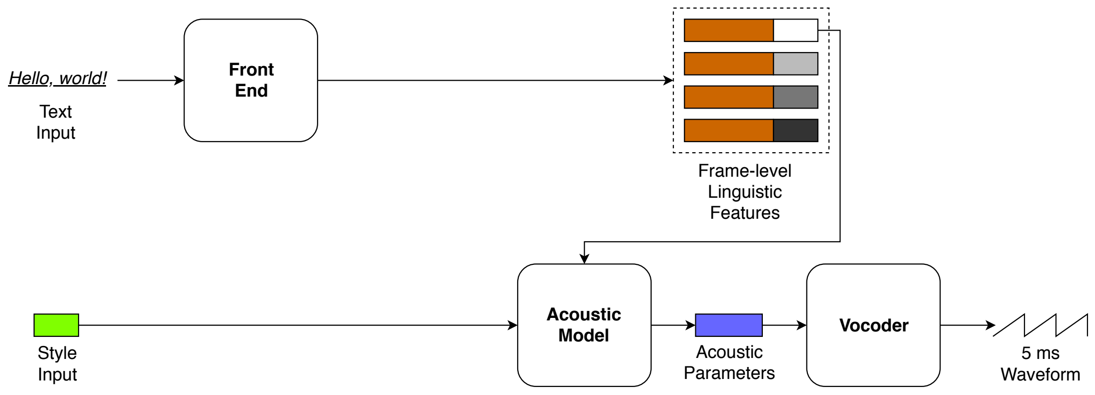

Figure 3.1: Generating a frame of speech with the acoustic model and natural state durations (Method A).

The addition of the duration model allows for the control of speaking rate, and therefore, we would expect there to be a greater perceptible range of style

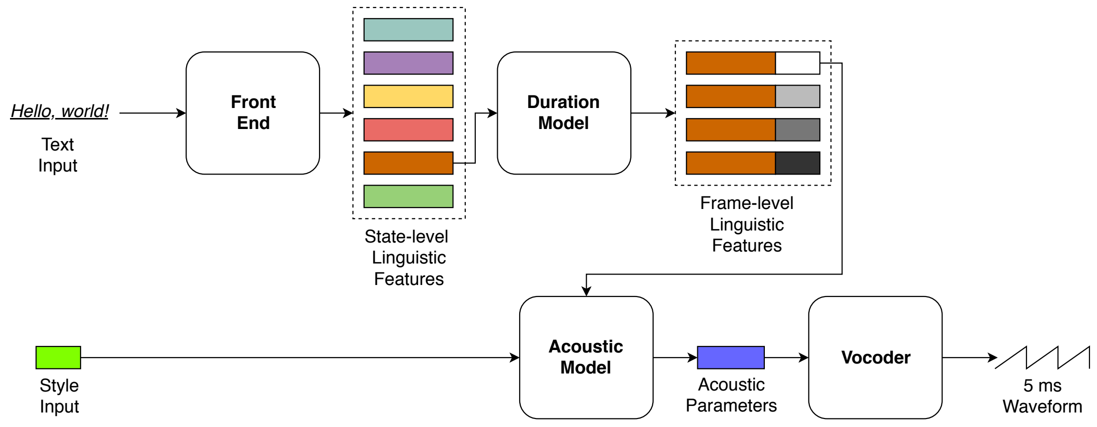

Figure 3.2: Generating a frame of speech with state durations predicted by the duration model (Method B).

expressed with the duration model than without.

To verify whether the duration model we implemented had the desired effect, we first created synthesis Methods A and B, illustrated in Figures 3.1 and 3.2, respectively. Method A, which corresponds to synthesis with the baseline system, uses natural state durations extracted from recorded speech. Method B, on the other hand, unpacks state-level features into the frame level with the help of the duration model.

The two methods were used to synthesize four sentences with three different sets of control vectors, shown in Table 3.1. The vectors were chosen to be not too far away from the neutral style. This was because style vectors, which are well beyond those of any of the training utterances, notably compromise speech quality. For each sentence/vector set combination, waveforms were synthesized with both vectors. Subjects were asked to listen to the waveform pairs produced with Methods B and C and indicate which pair they thought had more variation in expression.

Set	Vectors	
1	$(-0.5, -0.5)$	$(0.5, 0.5)$
2	$(-0.5, 0.5)$	$(0.5, -0.5)$
3	$(0.5, 0.0)$	$(0.0, 0.5)$

Table 3.1: Control vector sets used for the duration model experiment.

3.3.2 Effect of the Sentence Encoder

To determine whether the sentence encoder achieved its goal of predicting appropriate style from text, we synthesized 24 sentences in the test set using Method B and Method C, which corresponds to synthesis using the full system, illustrated in Figure 2.1. The selected sentences had predicted styles that were the furthest away from the training set mean style.

This decision was made due to the fact that, for many sentences, the predicted style was very close to the neutral style, and as a result, in a preliminary informal listening test, no perceptible difference was noted for many of the sentences in the test set. Although there may be concerns that our selection method could have introduced bias to the experiment, (Chevelu, Lolive, Maguer, & Guennec, 2015) argue that it is actually random selection of samples that creates a bias toward judgement that two systems being evaluated are not different. Not surprisingly, while only 14.6% of test set sentences had exclamation marks, 45.8% of the selected sentences did.

For each sentence, listeners were provided with the text and the waveforms produced with the two methods. Then, they were asked to listen to both waveforms and indicate which one they thought was most appropriate for the given text.

Chapter 4

Results and Discussion

This chapter includes the presentation and analysis of results for the listening experiments conducted and suggestions for further improving the proposed system for future research as well as some concluding remarks.

4.1 Results

4.1.1 Duration Model Experiment

Figure 4.1 shows whether listeners thought Method B (with the duration model) was superior to Method A at expressing a wider range of style for each sentence/vector set pair. On average, listeners indicated that Method B was superior to Method A for only four out of 12 sentences. Out of a total of 263 responses, 113 (43.0%) indicated superiority for Method B, showing that Method B was actually inferior to Method C, although the difference was not significant ($p = .943 > .05$). It appears that variation in duration, for the control vectors used, was trivial relative to variation in other aspects of speech style. Figure 4.2 shows how duration changes as the control vector varies for one of the sentences used for evaluation. For vectors $(-0.5, -0.5)$ and $(0.5, 0.5)$, the difference in duration is only 7.8%. Considering that the just-noticeable difference in speaking rate is around 10% (Quené, 2001), it is likely that the subjects noticed little to no variation in duration with most of the provided samples. From the result of the experiment, we conclude that the duration model does not enhance variability in style.

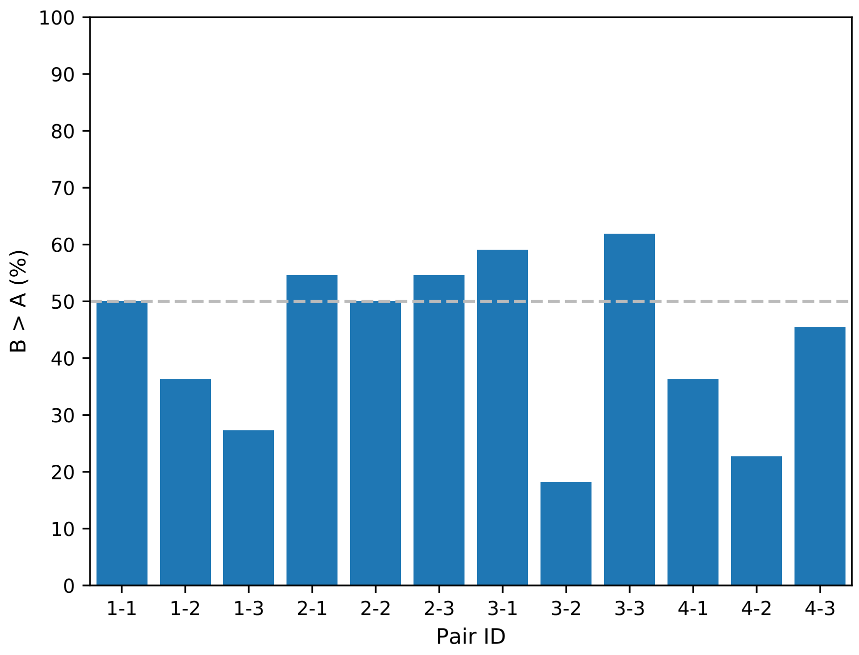

Figure 4.1: Percentage of participants preferring waveform synthesized with Method B to one synthesized with Method A for each sentence/vector set pair.

4.1.2 Sentence Encoder Experiment

Figure 4.3 shows a per-sentence breakdown of whether listeners thought waveforms produced with Method C (with the sentence encoder) sounded more appropriate than those produced with Method B. Of the 24 sentences, preference for Method C was greater than 50% for 16 sentences. In addition, out of the 527 responses, 303 (57.5%) indicated preference for Method C to Method B, and the observed superiority of Method C is likely not due to random chance ($p = 0.066 < 0.05$). Based on this result, we conclude that the sentence encoder we implemented is capable of predicting appropriate styles from text.

4.2 Concluding Remarks

In this study, we aimed to create a comprehensive sentence-level controllable TTS system by incorporating a duration model and a sentence encoder to the system proposed by (Watts et al., 2015). The duration model, while capable of varying duration in a predictable manner as the control vector changes, was unable to produce any audible variation in speech rate with a moderate variation in control

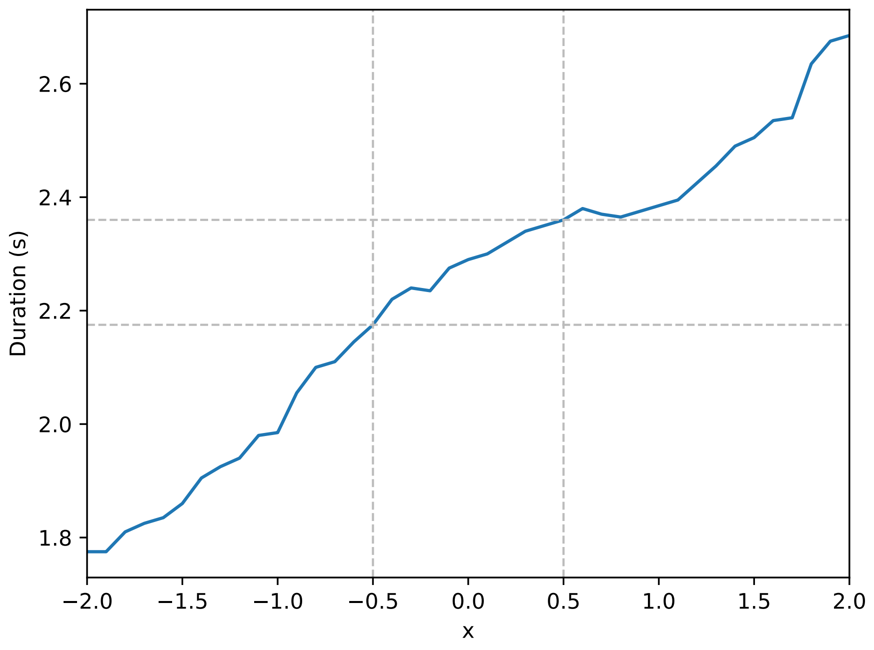

Figure 4.2: Duration of waveforms synthesized with Method B for the sentence *The prince looked so shocked he had to laugh.* with control vectors (x, x), where x is on $[-2.0, 2.0]$.

vector input, and as a result, did not enhance the perceptible range of style expressed. The sentence encoder, on the other hand, was demonstrably successful at predicting appropriate styles from text. We would like to close this piece by suggesting some ways to improve the proposed system in future research.

4.2.1 Duration Model

There are two possible explanations for why the duration model did not produce large enough variation in speaking rate. One possible explanation for this is that there is little variation in speaking rate in the training set, in which case, a dataset with a larger variation in speaking rate would need to be used for better duration control. Another possible explanation is that, since the projection layer of the duration model was not allowed to update, some of the duration variation in the training set was not captured. A future study can be conducted to determine whether this hypothesis is correct by reversing the training order, i.e. train the duration model first and then the acoustic model with a fixed projection layer, and seeing if duration control improves.

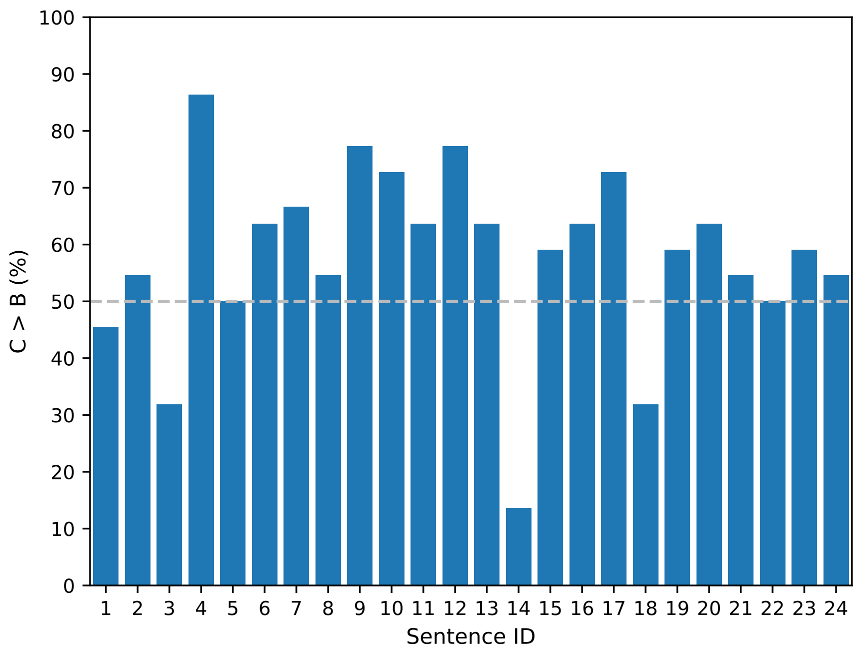

Figure 4.3: Percentage of participants preferring waveform synthesized with the Method C to one synthesized with Method B for each sentence.

4.2.2 Sentence Encoder

Although we did achieve good results with the sentence encoder, the average validation error of the fully trained encoder was rather high at 62% of the standard deviation of the target set. We believe that the high error was partially due to the fact that many of the "sentences" in the dataset were actually sentence fragments or sequences of sentences. Had the utterances been re-chunked to whole sentences, we might have obtained a lower error and a higher listener preference for Method C.

Another issue with the sentence encoder is that it essentially performs a natural language processing task on a speech dataset. The IMDB Dataset, which is a dataset of movie reviews useful for training sentiment analyzers, contains 100,000 movie reviews, each with multiple sentences (Maas et al., 2011). This number is not really feasible with a speech corpus recorded by one person. An interesting future project could involve training a sentiment analyzer on a text-only corpus and adapting it to the task of style prediction using a relatively small set of text/style pairs.

4.2.3 Sub-sentence Level Control

The issue of lack of local control discussed in Section 2.1.3 was not addressed in the present work. A naive way of implementing local control is to divide each sentence into small units, for instance, words, and train the acoustic and duration models as if each word in the training set was a sentence. In other words, the projection layer, instead of mapping a sentence to a style, would map a word to a style. Since only the frames belonging to the utterance of a word, not a whole sentence, would be used for training the style for the word, this way of implementing word-level control would lead to scarcity of training data for the projection layer, and similar to how states are clustered in HMM-based synthesis, some way of allowing similar word utterances to share parameters will be required.

Word-level control would also necessitate modification of the sentence encoder. Instead of outputting a single style vector for an entire sentence, the encoder would need to output a style vector for each *word* (as opposed to *token*). In fact, in a way, style prediction at the word level would be a type of translation task, meaning that the modified architecture should properly be referred to as an encoder-decoder.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... Zheng, X. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Retrieved from `https://www.tensorflow.org/` (Software available from tensorflow.org)

Ben-David, B. M., Multani, N., Shakuf, V., Rudzicz, F., & van Lieshout, P. H. H. M. (2016). Prosody and Semantics Are Separate but Not Separable Channels in the Perception of Emotional Speech: Test for Rating of Emotions in Speech. *Journal of Speech, Language, and Hearing Research*, *59*(1), 72-89. Retrieved from `http://dx.doi.org/10.1044/2015_JSLHR-H-14-0323` doi: 10.1044/2015_JSLHR-H-14-0323

Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature. *Geoscientific Model Development*, *7*(3), 1247–1250. Retrieved from `https://www.geosci-model-dev.net/7/1247/2014/` doi: 10.5194/gmd-7-1247-2014

Chevelu, J., Lolive, D., Maguer, S. L., & Guennec, D. (2015). How to Compare TTS Systems: A New Subjective Evaluation Methodology Focused on Differences. In *Proceedings of INTERSPEECH.* ISCA.

Cho, K., van Merriënboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014, October). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1724–1734). Doha, Qatar: Association for Computational Linguistics. Retrieved from `http://www.aclweb.org/anthology/D14-1179`

Espic, F., Botinhao, C. V., & King, S. (2017). Direct Modelling of Magnitude and Phase Spectra for Statistical Parametric Speech Synthesis. In *Proc.*

Interspeech 2017 (pp. 1383–1387). Retrieved from `http://dx.doi.org/10.21437/Interspeech.2017-1647` doi: 10.21437/Interspeech.2017-1647

Glorot, X., & Bengio, Y. (2010, 13–15 May). Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh & M. Titterington (Eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (Vol. 9, pp. 249–256). Chia Laguna Resort, Sardinia, Italy: PMLR. Retrieved from `http://proceedings.mlr.press/v9/glorot10a.html`

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *CoRR*, *abs/1502.01852*. Retrieved from `http://arxiv.org/abs/1502.01852`

King, S., Wilhborg, L., & Guo, W. (2017). *The Blizzard Challenge 2017* (Tech. Rep.). University of Edinburgh, United Kingdom.

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *CoRR*, *abs/1412.6980*. Retrieved from `http://arxiv.org/abs/1412.6980`

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011, June). Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies* (pp. 142–150). Portland, Oregon, USA: Association for Computational Linguistics. Retrieved from `http://www.aclweb.org/anthology/P11-1015`

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations* (pp. 55–60). Retrieved from `http://www.aclweb.org/anthology/P/P14/P14-5010`

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 26* (pp. 3111–3119). Curran Associates, Inc. Retrieved from `http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf`

Murray, I. R., Arnott, J. L., & Rohwer, E. A. (1996). Emotional stress in synthetic speech: Progress and future directions. *Speech Communication*, *20*(1), 85

- 91. Retrieved from http://www.sciencedirect.com/science/article/pii/S0167639396000465 (Speech under Stress) doi: https://doi.org/10.1016/S0167-6393(96)00046-5

Nose, T., Yamagishi, J., Masuko, T., & Kobayashi, T. (2007, 09). A Style Control Technique for HMM-Based Expressive Speech Synthesis. , *90-D*, 1406-1413.

Pascanu, R., Mikolov, T., & Bengio, Y. (2013, 17–19 Jun). On the difficulty of training recurrent neural networks. In S. Dasgupta & D. McAllester (Eds.), *Proceedings of the 30th International Conference on Machine Learning* (Vol. 28, pp. 1310–1318). Atlanta, Georgia, USA: PMLR. Retrieved from http://proceedings.mlr.press/v28/pascanu13.html

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532–1543). Retrieved from http://www.aclweb.org/anthology/D14-1162

Pitrelli, J. F., Bakis, R., Eide, E. M., Fernandez, R., Hamza, W., & Picheny, M. A. (2006, July). The IBM Expressive Text-to-speech Synthesis System for American English. *Trans. Audio, Speech and Lang. Proc.*, *14* (4), 1099–1108. Retrieved from https://doi.org/10.1109/TASL.2006.876123 doi: 10.1109/TASL.2006.876123

Quené, H. (2001, 01). On the just noticeable difference for tempo in speech. , *35*, 353-362.

Watts, O., Wu, Z., & King, S. (2015, 9). Sentence-level control vectors for deep neural network speech synthesis. In *INTERSPEECH 2015 16th Annual Conference of the International Speech Communication Association* (pp. 2217–2221). International Speech Communication Association.

Wu, Z., Watts, O., & King, S. (2016, 9). Merlin: An Open Source Neural Network Speech Synthesis System. In *9th ISCA Speech Synthesis Workshop (2016)* (pp. 202–207). doi: 10.21437/SSW.2016-33

Yamagishi, J., Onishi, K., Masuko, T., & Kobayashi, T. (2003, 01). Modeling of various speaking styles and emotions for HMM-based speech synthesis. In *Proc. EUROSPEECH.*

Zen, H., Senior, A., & Schuster, M. (2013). Statistical Parametric Speech Synthesis Using Deep Neural Networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 7962–7966).

Zen, H., Tokuda, K., & Black, A. W. (2009). Statistical parametric speech synthesis. *Speech Communication*, *51*(11), 1039 - 1064. Retrieved from `http://www.sciencedirect.com/science/article/pii/S0167639309000648` doi: https://doi.org/10.1016/j.specom.2009.04.004