EmoSpeech: Guiding FastSpeech2 Towards Emotional Text to Speech

Anonymous submission to SSW 2023

Abstract

State-of-the-art speech synthesis models try to get as close as possible to the human voice. Hence, the modeling of emotions is an essential part of Text-To-Speech (TTS) research. In our work, we selected FastSpeech2 as the starting point and proposed a series of modifications for synthesizing emotional speech. According to automatic and human evaluation, our model, EmoSpeech, surpasses existing models in terms of both MOS score and emotion recognition accuracy in generated speech. We provided a detailed ablation study for every extension to FastSpeech2 architecture that forms EmoSpeech. Crucial factor when generating speech with emotions, is to consider the uneven distribution of emotions in the text. Our model includes an additional mechanism that effectively handles this issue by allowing emotions to contribute to each phoneme with varying levels of intensity. The human assessment indicates that proposed modifications generate audio with higher MOS and emotional expressiveness.

Index Terms: text to speech, emotional text to speech, fast speech

1. Introduction

Over the past few years, the field of Text to Speech (TTS) achieved significant progress in terms of the quality of synthesized speech [1, 2, 3, 4]. At the same time, most models with state-of-the-art performance are based on Normalizing Flows [1, 2, 3, 4], and therefore suffer from slower generation speed than models based on the Transformer [5] architecture [6, 7]. Moreover, such models often require more time and data for training [2]. It can often become a limitation for using models in high-load product environments, e.g., when fine-tuning for every user is needed. One attempt to overcome inference speed limitations is FastSpeech2 [6]. Even though FastSpeech2 loses to the latest models that use Normalizing Flows [1, 2, 3, 4], according to quality metrics, i.e., Mean Opinion Score (MOS), it surpasses others in terms of inference speed. Our work mainly focuses on providing a solution for high-load environments, such as large social networks. We choose FastSpeech2 [6] architecture as a starting point.

While estimating the quality of generated speech, we often consider how realistic it sounds. To produce realistic speech, TTS models have to take into account many factors absent from simple text input, such as prosody, stress, rhythm, intonation, and emotion. Emotional speech is easier to perceive and enriches the meaning of the spoken text, making it an attractive option for a wide range of applications. Recently, research into speech synthesis models that make it possible to convey emotion began attracting more attention [8, 9, 10, 11, 12, 13]. These models are traditionally represented by Global Style Tokens (GST) [14], which are based on Tacotron2 [15]. Like many other similar approaches, GST makes it possible to learn an embedding vector with good expressiveness. When it comes to conditioning the encoded text sequence, the same embedding vector is usually added to each text token. This ignores that, given the relationship between emotion, intonation and stress, and the fact that the latter two have a different effect on each token in a textual sequence, an emotion embedding may also have a different impact on each token in a sequence. We raise the question of the uneven distribution of emotions across the text sequence and analyze it later in this paper.

The main contributions of this work are as follows:

- We propose an extension of the FastSpeech2 model architecture with several known and new modules that make it possible to synthesize speech with a desired emotion.
- The proposed model outperforms an existing implementation of FastSpeech2 extended for Emotional Speech Synthesis [1], both in terms of MOS and emotion recognition accuracy in generated speech, all without bringing inference speed latency.
- We propose a conditioning mechanism that makes it possible to account for the relationship between speech intonation and the strength of the emotion that falls on each token in the sequence, and show that EmoSpeech pays different attention to each part of the sentence depending on the given emotion.

This paper is organized as follows. In Section 2, we talk about prior research related to our work. Section 3 briefly describes the FastSpeech2 architecture, the architectural extensions that improve emotion transfer and form EmoSpeech, and the loss functions that are used for training and help prevent quality degradation. Then, we explain the experimental setup, report our results and show them in Section 4. Lastly, the conclusion and discussion are covered in Section 5.

2. Related Work

At a higher level, ETTS methods can be broadly classified into three distinct categories based on the nature of their conditioning data. These categories include models that use:

1. Categorical labels to represent one or more emotions [16, 17, 9, 12, 11].
2. Referenced speech with the desired emotional state [8, 18, 19].
3. Textual descriptions of the target emotional state as a form of conditioning data [20, 21].

When working with a labelled dataset, the first approach is tra-

[1] https://github.com/keonlee9420/Expressive-FastSpeech2

ditionally used, as it helps implement conditioning simply by introducing an embedding lookup table. The second approach is originally represented by Global Style Tokens (GST) [14]. GST aims to learn an embedding vector for each emotion from reference speech and text and introduce a global style token to obtain utterance-level embeddings. The advantage of GST is that style tokens can be learned in an unsupervised setup, and input text can be used for conditioning during inference. The third approach is the most flexible and could satisfy almost every need of emotional speech synthesis. However, training such a model requires larger datasets that contain rich text descriptions. This approach may be redundant when we need to synthesize speech with emotions from a limited set. Since our main focus in this work is to provide a model for a fast ETTS capable of synthesizing speech in a high-load product environment when given a small fixed set of speakers and emotions, we take the first approach.

The traditional approach to synthesizing emotional speech given a categorical label is presented in [17]. Originally, it was built on top of Tacotron2 [15] The authors of [17] suggested concatenating a categorical label with the pre-net output and then adding a layer to project the vector to match the size of the attention RNN input and the first layer of the RNN decoder of Tacotron2. Later, this approach was adopted for FastSpeech2 [6] in Expressive FastSpeech2 [16], where conditioning works similarly to multi-speaker conditioning through adding emotion embedding to the encoder output. Expressive FastSpeech2 implementation is the most relevant work for us, as it also uses FastSpeech2 and suits our inference speed constraints.

The choice of FastSpeech2 as a starting point for further model modification for expressive synthesis [16], zero-shot speech synthesis case [22], better quality in multi-speaker scenario [23] etc., is not novel. One of the most successful modifications is AdaSpeech [24] for custom voice synthesis, as well as GANSpeech [23] for multi-speaker synthesis.

The main concept introduced in AdaSpeech is Conditional Layer Norm (CLN). The idea of CLN is to extend well-known LayerNorm [25], but with a condition on speaker embedding for calculating scale and bias. Specifically, CLN takes the form of:

$$y = \mathrm{f(c)} \cdot \frac{\mathrm{x} - \mathrm{mean}}{\mathrm{var}} + \mathrm{f(c)}, \tag{1}$$

where f is a linear layer, x is a normalized hidden and c is a conditioning embedding. In AdaSpeech, all layer normalizations in the encoder were substituted with CLN. Later, in AdaSpeech4 [22], which was designed for zero-shot scenarios, it was suggested to integrate CLN in both the encoder and decoder for better performance.

Quality degradation is one problem that appears in FastSpeech2-like models after extensive conditioning. It is addressed in GANSpeech [23], where it occurs while training FastSpeech2 in a multi-speaker setup. In [23], the authors aimed to improve audio quality by using 2 phases of training. During the first phase, FastSpeech2 is trained using a classic reconstruction loss. In the second phase, the JCU discriminator without a hierarchically-nested structure from VocGAN [26] is applied for adversarial training. JCU discriminators consist of conditional and unconditional parts. The unconditional part of the discriminator processes the mel spectrogram x as it is, and the second processes x within the conditioning vector c. FastSpeech2 (G) and the JCU discriminator (D) were then optimized by using the least squares objective:

$$\begin{aligned} L_{adv}(\mathrm{D}) =& \frac{1}{2}\mathbb{E}_{\mathrm{c}}\left[D(\hat{\mathrm{x}})^2 + D(\hat{\mathrm{x}},\mathrm{c})^2\right] \\ &+ \frac{1}{2}\mathbb{E}_{(\mathrm{x},\mathrm{c})}\left[(D(\mathrm{x})-1)^2 + (D(\mathrm{x},\mathrm{c})-1)^2\right] \\ L_{adv}(\mathrm{G}) =& \frac{1}{2}\mathbb{E}_{\mathrm{c}}\left[(D(\hat{\mathrm{x}})-1)^2 + (D(\hat{\mathrm{x}},\mathrm{s})-1)^2\right]. \end{aligned}$$

In [23], the authors also apply feature matching loss to improve model quality and stability by computing the L1 loss between JCU discriminator feature maps of real and generated mel spectrograms:

$$L_{fm}(G,D) = \mathbb{E}_{\mathrm{x},\mathrm{c}}[\mathbb{E}_{\mathrm{l}}(D_l(\hat{\mathrm{x}},\mathrm{c}) - D_l(\mathrm{x},\mathrm{c}))],$$

where D_l is the output from the JCU discriminator layer l. GANSpeech is then trained with:

$$L_{total} = L_{rec} + L_{adv}(D) + L_{adv}(G) + \alpha_{fm} \cdot L_{fm} \tag{2}$$

3. Model Description

In this paper, we propose the EmoSpeech model, which is an extension of FastSpeech2 for emotional speech synthesis. In the following sections, we briefly describe FastSpeech2 and each component that forms EmoSpeech.

3.1. FastSpeech2

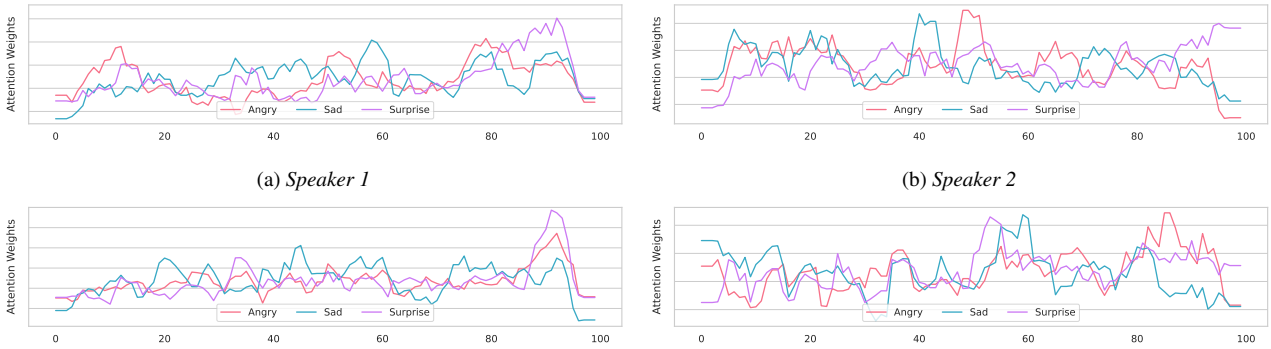
FastSpeech2 is a non-autoregressive acoustic model for fast and high-quality speech synthesis. The model takes a sequence of tokens as input and generates mel spectrograms, which are later upsampled to a waveform by a vocoder. The key components of FastSpeech2 are the encoder, variance adapter, and decoder [6]. The intuition behind each block is that the encoder extracts features from textual information about what would be said, the variance adaptor adds acoustic and duration information to the input sequence, and the decoder generates features from all of this for the mel spectrogram features. The encoder and decoder are a feed-forward transformer block, which is a stack of multi-head self-attention layers and 1D-convolution. An encoder converts a token embedding sequence into the token's hidden representation $\mathrm{h} \in \mathbb{R}^{n \times hid}$, where n, hid are the sequence length and hidden dimension, respectively. The variance adaptor consists of 3 predictors followed by a length regulator. Predictors take $\mathrm{h} \in \mathbb{R}^{n \times hid}$ and output the pitch, energy, and durations $(\mathrm{p}, \mathrm{e}, \mathrm{d})$ for each token. Then, the length regulators upsample $\mathrm{h} \in \mathbb{R}^{n \times hid}$, accumulated with $\mathrm{p}, \mathrm{e} \in \mathbb{R}^n$ according to $\mathrm{d} \in \mathbb{R}^n$. Token duration is measured by the number of mel spectrogram bars, which is why length regulator output is $\mathrm{h} \in \mathbb{R}^{\mathrm{m} \times hid}$, where $\mathrm{m} = \sum_{i=0}^{n} d_i$. After that, the upsampled hidden are passed through a decoder, and the hidden dimension is reflected in mel channels using a linear layer. The final output is a predicted mel spectrogram $\mathrm{y} \in \mathbb{R}^{\mathrm{m} \times \mathrm{c}}$. The model learns to generate a mel spectrogram from the input text sequence using reconstruction loss:

$$L_{rec} = ||\mathrm{y} - \hat{\mathrm{y}}|| + ||\mathrm{d} - \hat{\mathrm{d}}||^2 + ||\mathrm{e} - \hat{\mathrm{e}}||^2 + ||\mathrm{p} - \hat{\mathrm{p}}||^2,$$

where $\hat{\mathrm{y}}, \hat{\mathrm{d}}, \hat{\mathrm{e}}, \hat{\mathrm{p}}$ are the predicted mel spectrogram, duration, pitch, and energy.

3.2. Conditioning Embedding

We use embedding lookup tables to construct EmoSpeech from FastSpeech2 for naive speaker and emotion conditioning. Specifically, we form the conditioning vector c by stacking the speaker and emotion embeddings. As a starting point



(a) *Speaker 1* (b) *Speaker 2*

(c) *Speaker 3* (d) *Speaker 4*

Figure 1: *Averaged attention weights distribution over normalized sentence lengths. Note that the "surprise" emotion tends to have more weight at the end of the sentence, other emotions also follow an intuitive distribution. See section 4.3 for more details.*

of our modifications, we add a conditioning vector c to the encoder output before feeding it to the variance adaptor. Similarly to [16], we expand the embedding on a sequence length dimension.

3.3. Egemap Predictor

By design, the variance adaptor of FastSpeech2 could be extended by adding additional predictors [6]. For EmoSpeech, we add Egemap predictor (EMP) to the variance adaptor that predicts k features from the Extended Geneva Minimalistic Acoustic Parameter Set (eGeMAPS) [27]. In total, eGeMAPS consists of 88 features and is widely used for speech emotion recognition [28, 29]. However, we found that the extracted features are highly correlated and using only part of them simplifies the model without drop in quality. We describe the number of selected features in section 4. The idea behind EMP is to add to the utterance more information from the low-level speech descriptors that highly correlate with the target emotion. EMP has the same architecture as pitch and energy predictors – we follow the setup from [6], but unlike pitch and energy predictors, the egemap predictor operates on the utterance level rather than on the token level.

3.4. Conditional Layer Norm

Following the early success of AdaSpeech4 [22] in applying the Conditional Layer Norm (CLN) for zero-shot speech synthesis, we adopt CLN for emotional speech synthesis. We also found it beneficial to apply it instead of the traditional Layer Norm in the encoder and decoder blocks of EmoSpeech. CLN takes the same form as in Equation 1. For conditioning the embedding c,, we stack speaker and emotion embeddings that are taken from the embedding lookup tables.

3.5. Conditional Cross Attention

Expressive intonation is one of the characteristics of emotional speech. And if we listen to such speech, we can notice that sometimes the speaker puts a lot of emphasis on some parts of the sentence, which makes the emotion clearly distinguishable, while the rest of the sentence may sound very neutral. At the same time, the traditional approach in emotional speech synthesis is to add emotion embeddings to each text token with the same weight [9, 17]. In this work, we introduce a *Conditional Cross-Attention* (CCA) block to the encoder and decoder, which goal is to reweight tokens according to the given emotion. Specifically, we add CCA to the encoder and decoder so that each FFT layer (following the notation in [30]) consists of Self-Attention, Conditional Cross-Attention and position-wise feed forward.

We stack speaker and emotion embedding for the utterance and give this conditioning vector a notation $\mathrm{c} \in R^{hid}$ and notate Self-Attention output as $\mathrm{h} \in R^{n \times hid}$. CCA utilize W_q, W_k, W_v matrices from Self-Attention layer and forms: $Q = W_q \cdot \mathrm{h}, K = W_k \cdot \mathrm{c}, V = W_v \cdot \mathrm{c}$. Then we reweight the hiddens:

$$\mathrm{w} = \mathrm{softmax}(\frac{\mathrm{Q} \cdot \mathrm{K}^T}{\sqrt{d}}, \mathrm{dim} = 1) \tag{3}$$

$$\mathrm{cca} = \mathrm{w} \cdot V$$

Roughly, this operation can be seen as adding a special emotion token for every layer. Similarly to the multi-head self-attention mechanism, we add multi-head logic to conditional cross-attention in our implementation. Note that CCA is a substitution for a naive addition of conditioning embedding c expand on a sequence length dimension to the encoder output before feeding it to the variance adaptor. This is why we no longer make this addition in EmoSpeech after the modification. Section 4 shows the density distribution of attention weights w across the text utterance for different emotions.

3.6. Adversarial Training

Although the above methods help us improve the transmission of emotionality and natural intonation, numerous artifacts can still be heard in the generated speech. The quality degradation might occur while generalizing FastSpeech2 for a multi-speaker setup, as was mentioned in GANSpeech [23]. To boost the quality of generated speech, we utilize the same technique as in [23] and apply adversarial training for EmoSpeech, as the conditional architecture of JCU discriminator [26] suits our multi-speaker and multi-emotional setup. We used the same architectural setup and training objective as in [23]. However, we trained the discriminator along with the EmoSpeech during a single training phase. For the conditioning discriminator, we use the same conditioning embedding c, which is a stack of speaker and emotion embeddings.

Overall, EmoSpeech is trained with the same objective as in Equation 2, where $\alpha_{fm} = \frac{L^{sg}_{rec}}{L_{fm}}$ is added for training

stabilization [23], L_{rec}^{sg} notates stop gradient for L_{rec}. We also add an MSE for predicting eGeMAPS features to L_{rec}.

4. Experiments and Results

In this section, we describe the data and its preprocessing, training, and evaluation details, setup of conducted experiments, and analyze the results.

4.1. Datasets and Data Preprocessing

Dataset. In this paper, we focus on providing a lightweight solution for speech synthesis with a fixed set of emotions and multiple speakers. For our experiments, we use the English subset of the ESD [31] dataset. ESD is a well-known dataset for ETTS that contains audios of 10 speakers and 5 emotions each: Neutral, Angry, Happy, Sad, and Surprised. Each audio sample has a text annotation and corresponds to a single emotion label. We split the dataset into training and test according to the instructions in the original dataset.

Data preprocessing. Since EmoSpeech training requires not only audio along with the text transcript, but also phonemes, mel spectrograms, durations, pitch, and other acoustic features, we apply the following preprocessing:

1. Extract phonemes, punctuation, and silence tokens from text annotations using the grapheme-to-phoneme (GTP) model from the Montreal-forced-aligner (MFA) [32] toolkit.
2. Extract durations of extracted phonemes using MFA [32].
3. Extract pitch from ground truth mel spectrograms using `pyworld`[2] library.
4. Extract energies normalizing spectrograms by frequency dimension.
5. Extract eGeMAPS features using `opensmile` toolkit [33].

Pitch, energies, and eGeMAPS features are normalized and the first 2 are averaged frame-wise according to the phoneme durations. As for eGeMAPS features, we extract a single value for the audio sample.

eGeMAPS feature selection. Originally, eGeMAPS is a standard acoustic parameter set that consists of 88 low-level feature descriptors and is used for various areas of automatic voice analysis. To find what features are the most relevant for emotion transmission, we fitted CatBoost classifier [34] for emotion classification task and ranked all features in descending order of importance. Then we trained multiple models starting from no features at all and adding them one by one. We noticed that starting from using the third feature, the model begins degradation in terms of MOS. Therefore, in all experiments, we used only 2 first features.

4.2. Model Configuration and Training Details

Model configuration. The EmoSpeech architecture is based on FastSpeech2 [6] and is configured using the same hyperparameters, except phoneme embedding, encoder, and decoder hidden dimensions are set to 512; encoder and decoder Conv1D filter sizes are 512; encoder and decoder include 6 layers each. The hidden dimension of speaker and emotion embedding is 256. All layer normalizations in the encoder and decoder are replaced by CLN. The Egemap predictors follow the same architecture as the rest of the predictors in the Variance Adaptor block: 2 1D convolutions with kernel size 3, stride 1, followed by ReLU activation and dropout 0.5. For JCU Discriminator we follow the same architectural setup as in GANSpeech [23] including parameters for training configuration. We used iSTFTNet [35] vocoder trained on ESD dataset to synthesize audio from generated mel spectrograms [35]. For training configuration, we followed implementation[3] and parameters described in [35]. The Mel spectrogram was extracted from a waveform with a filter length of 48 ms, a hop length of 12 ms, and 80 mel channels.

Table 1: *The description of the used notation for sequential model modifications according to section 3.6 along with total model size.*

ID	Model	# parameters
#1	FastSpeech2 [6] + EMP	$47.1M$
#2	#1 + CLN	$53.4M$
#3	#2 + CCA	$53.4M$
#4	#3 + JCU (EmoSpeech)	$53.4M$

Training and inference details. We trained the model using 4 Nvidia A100 GPUs with a total batch size of 256 for 50000 steps, for the optimizer, scheduler, and related parameters we follow the setup in FastSpeech2 [6]. The JCU Discriminator was trained together with EmoSpeech using Adam optimizer [36] with a learning rate of 0.0001 with $(\beta_1, \beta_2) = (0.5, 0.9)$.

4.3. Experiments and Results

Baseline model. As an implementation of Expressive FastSpeech2 [17, 16] is based on FastSpeech2 and makes it possible to synthesize speech with a given emotion in very naive form by adding their embeddings to the encoder output before feeding it to the variance adaptor, we choose it as a baseline model.

Evaluation setup. We construct EmoSpeech starting from FastSpeech2 and modifying it by sequentially adding the components described in section 3.6. We assign special IDs for each combination of modifications to simplify references, table 1 provides information about them and total model sizes. Although the size of EmoSpeech increased with the added modifications by 15% compared with the same configured Expressive FastSpeech2, we did not notice any notable increase in the inference speed.

For all trained models we conduct both automatic, for proper selection of hyperparameters and quality control in the early stage, and human evaluations, to collect detailed feedback with the help of annotators.

Table 2: *The MOS and NISQA [37] scores on ESD dataset [31]. Original stands for ground truth audios and reconstructed are audios reconstructed from ground truth mel spectrograms.*

ID	Model	MOS (↑)	NISQA (↑)
-	Original	4.7 ± 0.49	4.17 ± 0.57
-	Reconstructed	4.54 ± 0.58	4.11 ± 0.58
-	Expressive FastSpeech2 [16]	3.74 ± 0.65	3.77 ± 0.74
#1	FastSpeech2 [6] + EMP	4.06 ± 0.65	3.71 ± 0.76
#2	#1 + CLN	4.25 ± 0.6	3.93 ± 0.66
#3	#2 + CCA	4.33 ± 0.6	3.95 ± 0.66
#4	#3 + JCU (EmoSpeech)	$\mathbf{4.37 \pm 0.62}$	$\mathbf{4.1 \pm 0.58}$

[2] `https://github.com/JeremyCCHsu/Python-Wrapper-for-World-Vocoder`

[3] `https://github.com/rishikksh20/iSTFTNet-pytorch`

Table 3: *Accuracy of emotions classified using human feedback. Overall, is an average of all emotions. Original stands for ground truth audios and reconstructed are audios reconstructed from ground truth mel spectrograms.*

ID	Model	Overall	Neutral	Angry	Happy	Sad	Surprise
	Original	0.89	0.69	0.92	0.97	0.96	0.91
	Reconstructed	0.86	0.59	1	0.92	0.94	0.87
	Expressive FastSpeech2	0.78	0.42	**0.96**	0.8	0.76	0.96
#1	FastSpeech2 + EMP	0.78	0.45	0.94	0.68	0.81	**1**
#2	#1 + CLN	0.82	0.48	0.92	0.88	0.82	**1**
#3	#2 + CCA	**0.85**	0.55	**0.96**	**0.91**	**0.83**	**1**
#4	#3 + JCU (EmoSpeech)	0.83	**0.56**	0.94	0.8	**0.83**	**1**

Automatic evaluation. For automatic evaluation, we used the `NISQA` library [4] [37]. In our work, we used the NISQA-TTS model and predicts naturalness score on a 5-point scale according to human MOS. The NISQA-TTS model was chosen as we focus on the realistic transmission of emotions that correlates with the naturalness of generated speech. The final results are represented in Table 2. The EmoSpeech model outperforms the baseline and all intermediate modifications and shows similar quality to reconstructed audio tracks.

Human evaluation. For human evaluation, we used 25 audios from 5 speakers in 5 emotions randomly selected from the test set. We asked 20 annotators to solve 3 tasks.

The first task was to measure whether emotion was transmitted correctly. We asked the annotators to select the emotion they hear in the audio: neutral or no emotion is recognized, anger, happiness, sadness, or surprise. We computed an accuracy score for each emotion and overall quality by averaging all values. The reported results are shown in Table 3.

All modifications successfully coped with the Surprise emotion, moreover, models managed to add emotion even where it was not heard in the original audio. Modification #3 has the highest accuracy across all emotions, with a slight quality increase from the EmoSpeech model. Accuracy drop after adversarial training is expected behavior as adversarial training was added to prevent synthesis quality degradation, its slightly smooth sharp intonation picks in the generated mel spectrograms. It can also be seen that the baseline model has the highest accuracy score for the anger emotion. For that model, the annotators mentioned that artifacts that appeared in the generated audio brought more anger to the sound, and therefore more samples were marked with the anger emotion.

Secondly, we evaluate audio quality using the Mean Opinion Score (MOS). We asked annotators to evaluate sound quality on a 5-point scale given additional instruction:

- 5 for excellent – no artifacts in the audio;
- 4 for good – some artifacts are heard, but they do not influence audio perception;
- 3 for fair – a lot of artifacts in the audio;
- 2 for bad – many artifacts in the audio, it is difficult to understand speech;
- 1 for poor – very noisy, almost impossible to listen to.

The results are shown in Table 2. Same as for the NISQA evaluation, the EmoSpeech model outperforms all intermediate modifications and baseline and shows synthesis quality close to the reconstructed audios.

Lastly, to measure naturalness, we conduct a pairwise audio comparison of the baseline model with each of our modifications. We asked the annotators whether they would rather choose the first or the second audio, given a target emotion. The results of the pairwise comparisons are shown in Figure 2. The EmoSpeech model is preferred by 10% more annotators than the baseline model.

[4] `https://github.com/gabrielmittag/NISQA`

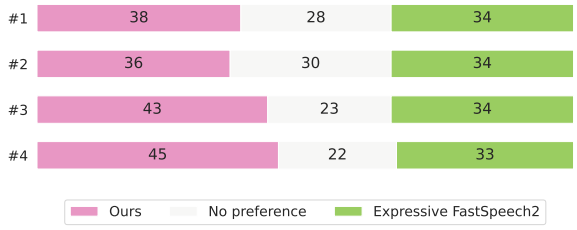


Figure 2: *Pairwise comparison of generated audio naturalness given a target emotion. The numbers indicate the percentage of responses in which annotators choose this model against other. Note that preference increases with adding modifications described in Table 3. See section 4.3 for more details.*

Analyzing attention weights in the CCA layer. The intuition behind Conditional Cross Attention (CCA) is that different tokens of the text sequence have different effects on emotional intensity. To show emotion distribution over sentences, we grouped all audios in a dataset by speakers, collected attention weights (Equation 3) from all layers, normalized sentence length, and plotted averaged attention weight distribution over the sentence. The result distribution for four speakers can be seen in (Figure 1). Higher attention weight means higher token importance for target emotion expressiveness. For the emotion ”surprise”, which often correlates with interrogative intonation, the importance of tokens increases at the end of a sentence; ”sad” tends to go down and has several spikes in the middle; ”angry” has different trends for every speaker.

5. Conclusion

In this paper, we developed EmoSpeech – an extension of the FastSpeech2 [6] model for Emotional Text-to-Speech Synthesis. We proposed multiple modifications for conditioning on a given emotion and stabilizing training while keeping the model as fast as the original FastSpeech2. Experiments showed that all our modifications enhance model quality and outperform the previous extensions of FastSpeech2 for ETTS, Extensive FastSpeech2 [16].

One of the key features of EmoSpeech is the conditional cross-attention mechanism, which takes into account the uneven distribution of emotion across a sentence. We demonstrated that

for different emotions, the model selected different phonemes to accent, and therefore, our model sounds more natural than the baseline. Our source code and generated samples could be accessed via the `link`[5].

6. References

[1] J. Kim, S. Kim, J. Kong, and S. Yoon, "Glow-tts: A generative flow for text-to-speech via monotonic alignment search," 2020.

[2] J. Kim, J. Kong, and J. Son, "Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech," 2021.

[3] V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, and M. Kudinov, "Grad-tts: A diffusion probabilistic model for text-to-speech," 2021.

[4] X. Tan, J. Chen, H. Liu, J. Cong, C. Zhang, Y. Liu, X. Wang, Y. Leng, Y. Yi, L. He, F. Soong, T. Qin, S. Zhao, and T.-Y. Liu, "Naturalspeech: End-to-end text to speech synthesis with human-level quality," 2022.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[6] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," 2022.

[7] N. Li, S. Liu, Y. Liu, S. Zhao, M. Liu, and M. Zhou, "Neural speech synthesis with transformer network," 2019.

[8] P. fei Wu, Z. hua Ling, L. juan Liu, Y. Jiang, H. chuan Wu, and L. rong Dai, "End-to-end emotional speech synthesis using style tokens and semi-supervised training," 2019.

[9] T. Li, S. Yang, L. Xue, and L. Xie, "Controllable emotion transfer for end-to-end speech synthesis," *CoRR*, vol. abs/2011.08679, 2020. [Online]. Available: https://arxiv.org/abs/2011.08679

[10] B. Schnell and P. N. Garner, "Improving Emotional TTS with an Emotion Intensity Input from Unsupervised Extraction," in *Proc. 11th ISCA Speech Synthesis Workshop (SSW 11)*, 2021, pp. 60–65.

[11] K. Zhou, B. Sisman, R. Rana, B. W. Schuller, and H. Li, "Speech synthesis with mixed emotions," 2022.

[12] Y. Guo, C. Du, X. Chen, and K. Yu, "Emodiff: Intensity controllable emotional text-to-speech with soft-label guidance," 2023.

[13] G. Zhang, Y. Qin, W. Zhang, J. Wu, M. Li, Y. Gai, F. Jiang, and T. Lee, "iemotts: Toward robust cross-speaker emotion transfer and control for speech synthesis based on disentanglement between prosody and timbre," 2023.

[14] Y. Wang, D. Stanton, Y. Zhang, R. J. Skerry-Ryan, E. Battenberg, J. Shor, Y. Xiao, F. Ren, Y. Jia, and R. A. Saurous, "Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis," *CoRR*, vol. abs/1803.09017, 2018. [Online]. Available: http://arxiv.org/abs/1803.09017

[15] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," 2018.

[16] K. Lee, "Expressive-fastspeech2," https://github.com/keonlee9420/Expressive-FastSpeech2, 2021.

[17] Y. Lee, A. Rabiee, and S.-Y. Lee, "Emotional end-to-end neural speech synthesizer," 2017.

[18] X. Cai, D. Dai, Z. Wu, X. Li, J. Li, and H. Meng. [Online]. Available: https://arxiv.org/abs/2010.13350

[19] Y. Wang, D. Stanton, Y. Zhang, R. Skerry-Ryan, E. Battenberg, J. Shor, Y. Xiao, F. Ren, Y. Jia, and R. A. Saurous, "Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis," 2018.

[20] D. Yang, S. Liu, R. Huang, G. Lei, C. Weng, H. Meng, and D. Yu, "Instructtts: Modelling expressive tts in discrete latent space with natural language style prompt," 2023.

[21] Z. Guo, Y. Leng, Y. Wu, S. Zhao, and X. Tan, "Prompttts: Controllable text-to-speech with text descriptions," 2022.

[22] Y. Wu, X. Tan, B. Li, L. He, S. Zhao, R. Song, T. Qin, and T.-Y. Liu, "Adaspeech 4: Adaptive text to speech in zero-shot scenarios," 2022.

[23] J. Yang, J.-S. Bae, T. Bak, Y. Kim, and H.-Y. Cho, "Ganspeech: Adversarial training for high-fidelity multi-speaker speech synthesis," 2021.

[24] M. Chen, X. Tan, B. Li, Y. Liu, T. Qin, S. Zhao, and T.-Y. Liu, "Adaspeech: Adaptive text to speech for custom voice," 2021.

[25] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016.

[26] J. Yang, J. Lee, Y. Kim, H. Cho, and I. Kim, "Vocgan: A high-fidelity real-time vocoder with a hierarchically-nested adversarial network," 2020.

[27] F. Eyben, K. R. Scherer, B. W. Schuller, J. Sundberg, E. André, C. Busso, L. Y. Devillers, J. Epps, P. Laukka, S. S. Narayanan, and K. P. Truong, "The geneva minimalistic acoustic parameter set (gemaps) for voice research and affective computing," *IEEE Transactions on Affective Computing*, vol. 7, no. 2, pp. 190–202, 2016.

[28] S. Latif, J. Qadir, and M. Bilal, "Unsupervised adversarial domain adaptation for cross-lingual speech emotion recognition," 2020.

[29] M. Schmitt, N. Cummins, and B. Schuller, "Continuous emotion recognition in speech - do we need recurrence?" in *Interspeech*, 2019.

[30] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech: Fast, robust and controllable text to speech," 2019.

[31] "Emotional voice conversion: Theory, databases and esd," *Speech Communication*, vol. 137, pp. 1–18, 2022.

[32] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, *Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi*, Aug 2017.

[33] F. Eyben, M. Wöllmer, and B. Schuller, "opensmile – the munich versatile and fast open-source audio feature extractor," 01 2010, pp. 1459–1462.

[34] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," 2019.

[35] T. Kaneko, K. Tanaka, H. Kameoka, and S. Seki, "istftnet: Fast and lightweight mel-spectrogram vocoder incorporating inverse short-time fourier transform," 2022.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[37] G. Mittag and S. Möller, "Deep learning based assessment of synthetic speech naturalness," in *Interspeech 2020*. ISCA, oct 2020. [Online]. Available: https://doi.org/10.21437%2Finterspeech.2020-2382

[5] `https://zenodo.org/record/7903317#.ZFarhOxBydY`