DISTRIBUTION AUGMENTATION FOR LOW-RESOURCE EXPRESSIVE TEXT-TO-SPEECH

Mateusz Lajszczak, Animesh Prasad, Arent van Korlaar, Bajibabu Bollepalli, Antonio Bonafonte, Arnaud Joly, Marco Nicolis, Alexis Moinet, Thomas Drugman, Trevor Wood, Elena Sokolova

Alexa AI, Amazon, Cambridge

ABSTRACT

This paper presents a novel data augmentation technique for text-to-speech (TTS), that allows to generate new (text, audio) training examples without requiring any additional data. Our goal is to increase diversity of text conditionings available during training. This helps to reduce overfitting, especially in low-resource settings. Our method relies on substituting text and audio fragments in a way that preserves syntactical correctness. We propose additional measures to ensure that synthesized speech does not contain artifacts caused by combining inconsistent audio samples. The perceptual evaluations show that our method improves speech quality over a number of datasets, speakers, and TTS architectures. We also demonstrate that it greatly improves robustness of attention-based TTS models.

Index Terms— Text-to-speech, data augmentation

1. INTRODUCTION

Data augmentation techniques play an important role in the training of neural networks, particularly in classification tasks [1, 2, 3]. They allow to improve performance of the data-hungry models and reduce overfitting. Despite a rapid progress of neural technologies in the field of text-to-speech (TTS) [4, 5, 6], very little work is devoted to applying data augmentation in this domain. At the same time, in many cases TTS data is scarce (e.g. for low-resource languages), thus data augmentation has the potential to be particularly impactful in such scenarios.

Data augmentation is underexplored in TTS partly because TTS is a generative problem. As argued in [7], transformations used for augmenting training data (e.g. in computer vision domain) often modify its distribution significantly. Thus, using them to train generative models without any countermeasures might lead to fitting the wrong distribution and, in result, to producing out-of-distribution samples during inference. Unsurprisingly then, data augmentation approaches for TTS focus mostly on generating more in-distribution data by applying techniques like voice conversion (VC) [8, 9]. Recent work on data augmentation for generative modelling attempts to overcome aforementioned limitations. In [7], a new method called distribution augmentation (DistAug) is proposed, where the generative model is conditioned on the augmentation itself. Thus, instead of a single distribution \( p(x) \), a conditional distribution is learned \( p(t(x) \mid t) \), where \( t \) represents a transformation used to augment data. Provided that the identity transformation, \( t = id_{x} \), is used during training, one can recover the model for unaugmented data by taking \( p(x \mid id_{x}) \). As explained in [7], DistAug method can be interpreted as a form of data-dependent regularization that reduces overfitting. Moreover, one can see it as a form multi-task learning that improves generalization by introducing beneficial inductive biases.

We take inspiration from this method and propose augmenting a TTS dataset with new samples that can potentially be out-of-distribution. We consider a family of augmentations which generate new training examples by assembling new texts and audios from existing ones. More precisely, similarly as in [10], we extract parse trees of text conditionings and create new texts by applying subtree substitution. We also apply a corresponding substitution to audio features taking advantage of available text-audio alignment. While our method does not improve acoustic diversity of the data, it increases diversity of text conditionings.

Our main contributions are as follows: 1) We introduce a novel data augmentation technique for TTS that can be used to considerably increase the diversity of text conditionings that are presented to the model. To our knowledge, this is the first application of distribution augmentation technique in neural TTS; 2) We demonstrate that our method reduces model’s overfitting to input text and greatly improves robustness of attention-based sequence-to-sequence TTS models; 3) We perform perceptual evaluations showing that our method improves quality of generated speech. We demonstrate this improvement on different data settings (varying speakers and dataset sizes) and two model architectures (attention based sequence-to-sequence model and a model using externally provided durations).

2. RELATED WORK

One of the most popular classes of methods for improving TTS model performance in low-data settings is transfer learning (TL) [11, 12]. In TL, the main idea is to transfer the knowledge learned in the source task to the target task.

NTTS systems based on TL methods first develop multi-speaker models as a source task. Then, the multi-speaker models are adapted to the target voice as a target task. TL methods improve the synthesis quality of low-resource target voices [11, 13]. However, they depend on the availability of multi-speaker datasets and complicate the training, as they often require fine-tuning with target speaker data.

To increase the amount of target speaker data, data augmentation approaches have been proposed [9, 8]. They are based on voice conversion (VC) models that allow to generate additional target speaker data provided that appropriate source speaker data is available.

The authors in [8, 9] applied the CopyCat VC model [14] to generate expressive synthetic speech in low-resource data settings. A drawback of this approach is that it requires additional speech datasets (including multi-speaker datasets) to train VC models and to provide source data for conversion. Moreover, performance of this approach depends on performance of the VC model.

In contrast to aforementioned methods, the data augmentation approach proposed in this work does not depend on collecting additional speech data or training additional models (like VC models). This can be especially beneficial for low-resource languages for which large multi-speaker datasets are not available. Furthermore, our method can in principle be applied on top of the existing approaches.
Expressive speech datasets are particularly challenging for TTS models. Many approaches were recently proposed to improve modeling of expressive speech [15, 16, 17, 18]. Our approach focuses more on alleviating issues with modeling expressive speech in low-resource setting. This includes issues like lack of robustness (in attention-based models) and lower signal quality that are particularly severe when amount of expressive data is limited.

3. PROPOSED METHOD

3.1. Data augmentation through word permutations

In this section we describe a general case of augmenting a TTS dataset by permuting words from this dataset. We point out potential issues with this kind of augmentations and propose techniques to mitigate them.

In TTS, the task is to model audio features $X$ (such as mel-spectrograms) for the given text representation $c$ (for example, a sequence of characters or phonemes). We can cast this as a probabilistic modeling problem, where the goal is to estimate the conditional distribution $p(X | c)$. Tacotron-like autoregressive models further factorize this distribution into $\prod p(x_i | x_{<i}, c)$. Given a dataset $D = \{(X_n, c_n)\}$ of independent data samples we can use standard methods like maximum likelihood estimation to fit $p(X | c)$.

Let us now consider the problem of augmenting $D$ with new samples that are derived from elements of $D$ by decomposing them into pieces (e.g. single words) and reassembling into new combinations. In this way we can generate practically unlimited number of new training examples $(X, c)$ in order to improve model performance and reduce overfitting. However, this poses a problem of generating out-of-distribution samples which would prevent us from fitting the right distribution. This can happen for the following reasons:

1. The marginal distribution of text conditionings $c$ is modified by augmentation (augmented samples can be syntactically or semantically unsound).
2. Assembled audio features have out-of-distribution local structure (audio joints, locations where audio pieces are joined, might sound unnatural).
3. Assembled audio features have out-of-distribution global structure (overall prosody of assembled audio is inconsistent).

In order to address (1) we only need to make sure that augmented dataset contains enough elements that follow the original distribution of $c$ and the model has enough capacity. As argued in [7], samples with out-of-distribution conditionings should not prevent the model from properly fitting in-distribution samples.

In order to address (1) and (3) we consider the following form of data augmentation. When sampling augmented data, we impose constraints on a distribution of augmented conditioning so that it does not diverge much from the original distribution of conditioning (i.e. often enough it follows rules of syntax and grammar). Additionally we ensure that augmented examples are assembled from long enough text/audio fragments so our models can learn consistent prosody. To implement augmentation following these constraints, we can use the substructure substitution method [10] and propose to extend it for TTS. We elaborate on it more in Section 3.2.

To counter (2) and (3), we introduce an additional conditioning that identifies the type of augmentation. This was originally proposed in [7] as a global conditioning on augmentation transformation identity. Since the global conditioning does not provide any information about the location of audio joints, we instead use a localized conditioning on audio joints, as depicted in Figure 3. It identifies the type of augmentation and at the same time addresses (2).

3.2. Constituency parse based tree substitutions

As mentioned in the previous section, we look for a constraint on the text generation process that would ensure that augmented text is often enough syntactically and grammatically correct. Below we describe the text generation technique of our choice that satisfies this requirement.

Tree Substitution Grammar [19] formalizes language generation as rewriting of non-terminal nodes of a tree (a parsed representation of a sentence), each with some arbitrary size subtrees. Given a text corpus with parses (e.g. constituency parse), these formulations allow estimating a generative grammar of the language [20]. Equivalently, without estimating the actual grammar, this formulation allows creation of new text samples by substituting a subtree with another subtree from the corpus (given that both the subtrees have the same type of non-terminal node). This is a popular data augmentation technique which not only works for low-resource text corpora but also shows gains for augmentation with massively pre-trained models [10]. An example of creating augmented text by substituting a constituency parse subtree can be found in Figure 1.

The following describes how we apply the subtree substitution technique for augmenting a TTS dataset. Assume we have a dataset $D$ consisting of triples $e_i = \{t_i, x_i, a_i\}$, where $t_i$ denotes text features, $x_i$ denotes audio features and $a_i$ denotes the alignment between text and audio. For each pair of training examples $e_i, e_j$ we take their respective texts $(t_i, t_j)$ and parse them using a constituency parser. Then, for each constituent $c^i_m$ in the parse tree of $t_i$ and each constituent $c^j_m$ in the parse tree of $t_j$, we create an augmented text by substituting $c^i_m$ by $c^i_n$, provided that $c^i_m$ and $c^j_n$ have the same constituency type. This procedure allows to create multiple new texts for each pair $e_i, e_j$. Then, we create augmented audio features taking advantage of available alignments $a_i$ and $a_j$. We simply concatenate appropriate fragments of audio features, following the alignments.

Note that this procedure only allows to create an augmented example from exactly two original examples. Also, for each augmented example there is exactly one subtree substitution. Thus, audio joints (as described in Section 3.1) are quite sparse: there is no more than two per utterance. Figure 2 presents distributions of lengths (in number of words) of parse tree constituents that were used to augment dataset $D_1$ (see Section 4.2). As can be seen, substituted constituents are usually short (1-3 words long). On the other hand, the remaining part of the utterance that surrounds substituted constituent is left intact and is usually much longer. This ensures that long enough audio pieces from original samples are presented to the model during training so it can learn consistent prosody spanning over longer fragments of text.

4. EXPERIMENTS

4.1. Model architectures

We apply our proposed method to two types of architectures (Figure 3): an attention-based architecture (A) and an architecture using externally provided durations (B). We choose them because they are the two main classes of models currently used in TTS for modeling mel-spectrograms. The attention-based one largely follows the architecture of Tacotron 2 [6]. The phoneme encoder consists
We run our experiments on two datasets:  

- A proprietary dataset containing highly expressive English speech recorded by a female voice talent ($D_1$).
- Two high-quality English voices selected from public HiFi TTS dataset [24], one male ($D_2^m$) and one female ($D_2^f$).

4.2. Data preparation

We use two model variants: one with attention between phoneme encodings and frames and another with attention replaced by upsampling. Input phonemes are concatenated with per-phoneme augmentation conditions.

We partition $D_1$ and $D_2$ into training, evaluation and test sets. To evaluate the proposed method in different data settings, we randomly select three subsets of sizes 10h, 5h and 2h from the $D_1$ train set. In the experiments with datasets $D_2^m$ and $D_2^f$ we always use 10h of data randomly selected from the training set. To facilitate trainings, we extract mel-spectrograms with 80 mel channels, using 12.5 ms frame shift and 50 ms frame length. We use proprietary TTS frontend to extract phonetic transcriptions of the audios and the Kaldi toolkit [25] to find alignments between text and audio.

In order to generate augmented dataset we apply the procedure described in Section 3.2. For constituency parsing we use Berkeley Neural Parser [26]. Note that it allows to produce millions of new samples from few thousand of original utterances. To simplify our experiments, we randomly select 500k augmented examples for each dataset. This roughly corresponds to 1k hours of speech. However, the augmented data does not offer more acoustic diversity than the original dataset. This is why we always report results for the number of hours of the original dataset.

4.3. Evaluation setup

We evaluate our models using the following protocol. For each tested configuration we randomly select 100 utterances from the testset of the respective dataset. We synthesize speech samples for these utterances using the baseline and the proposed model. Then we ask 60 native English speakers to rate each pair of samples in a preference test (we ask them to choose better sounding version).

For the attention-based variant we additionally run a robustness test measuring the word error (WER) rate and the phoneme error rate (PER) on the baseline and proposed models. This is because such models, trained on highly expressive data, are particularly prone to intelligibility issues such as mispronunciations, mumbling, skipping or repeating phonemes, or cut-offs. We run the robustness tests on a text corpus containing 2500 diverse utterances.

4.4. Results

Here we refer to the models developed without using any augmented data as baseline model (or simply baseline) and the models developed with augmented data as proposed models.
### Table 1: Word and phoneme error rates (bolded is better) of attention-based models for baseline (Base) and proposed models (Ours), relative to the WER and PER of 5h baseline model.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>WER</th>
<th>PER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base</td>
<td>Ours</td>
</tr>
<tr>
<td>$D_1^{10h}$</td>
<td>0.74</td>
<td><strong>0.26</strong></td>
</tr>
<tr>
<td>$D_1^{5h}$</td>
<td>1.00</td>
<td><strong>0.19</strong></td>
</tr>
</tbody>
</table>

### Table 2: Preference ratings (bolded is preferred) between baselines (Base) and proposed models (Ours) with architecture (A).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>% Preference</th>
<th>Test loss</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base</td>
<td>Ours</td>
</tr>
<tr>
<td>$D_1^{10h}$</td>
<td>38.7</td>
<td><strong>41.1</strong></td>
</tr>
<tr>
<td>$D_1^{5h}$</td>
<td>33.5</td>
<td><strong>46</strong></td>
</tr>
</tbody>
</table>

### Table 3: Preference ratings (bolded is preferred) between baselines (Base) and proposed models (Ours) with architecture (B).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>% Preference</th>
<th>Test loss</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base</td>
<td>Ours</td>
</tr>
<tr>
<td>$D_1^{10h}$</td>
<td>35.6</td>
<td><strong>40.3</strong></td>
</tr>
<tr>
<td>$D_1^{7h}$</td>
<td>37</td>
<td><strong>41.4</strong></td>
</tr>
<tr>
<td>$D_1^{5h}$</td>
<td>34.7</td>
<td><strong>43.2</strong></td>
</tr>
<tr>
<td>$D_1^{3h}$</td>
<td>34.5</td>
<td><strong>38.1</strong></td>
</tr>
<tr>
<td>$D_1^{1h}$</td>
<td>35.3</td>
<td><strong>38.3</strong></td>
</tr>
</tbody>
</table>

### Table 4: Preference ratings (bolded is preferred) for: 1) baseline (Base) vs proposed model without augmentation conditioning (W/o conditioning); 2) proposed model without augmentation conditioning (W/o conditioning) vs with conditioning (With conditioning).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>% Preference</th>
<th>With conditioning</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_1^{10h}$</td>
<td>39.8</td>
<td><strong>38.3</strong></td>
<td>-21.9</td>
</tr>
<tr>
<td>$D_1^{5h}$</td>
<td>33.7</td>
<td><strong>37.9</strong></td>
<td><strong>41.7</strong></td>
</tr>
</tbody>
</table>

4.4.1. **Attention-based models (A)**

For experiments with architecture (A) we use 5h and 10h variants of $D_1$. For each variant of $D_1$ we train one baseline model and one proposed model. We train the models for 200k steps using SGD with Adam optimizer. We also trained 2h variants, but the baseline failed to produce intelligible speech in this setting, and thus we do not evaluate it. At the same time the proposed model produces intelligible speech with only 2 hours of training data, which suggests that the impact of our method increases in lower data settings.

In order to demonstrate that the proposed method reduces overfitting, we calculate test set loss for both the baseline and proposed model. As can be seen in Table 2, the proposed models achieve lower test set losses than respective baselines. This shows that the proposed method indeed reduces overfitting.

Then, we evaluate the robustness of the trained models by running an ASR model on synthesized audio and comparing it to the text. The WER and PER are shown in Table 1. The proposed method brings a considerable improvement in terms of WER and PER when training on 5 or 10 hours of data. The higher error rates for the proposed system trained on 5h compared to 10h is a counterintuitive result. However, given the relatively small difference compared to the differences between the baseline and proposed models, we argue that this can be attributed to the stochasticity of training the models, selecting the data, and running the robustness analyses.

Finally, we run preference tests to compare the baseline and proposed models. Table 2 summarizes the results of the preference tests that render the proposed models being statistically significantly preferred over baseline (with p-value < 0.05). As the robustness test results suggest, the baselines suffer from robustness issues, so for the evaluation we filter out samples with nonzero WER, moving the results even more in favor of our model.

### 4.4.2. Models using externally provided duration (B)

In order to evaluate the proposed method on architecture (B) we use 5h and 10h variants of $D_1$. For each variant of $D_1$ we train one baseline model and one proposed model. We train the models for 200k steps using SGD with Adam optimizer. We also trained 2h variants, but the baseline failed to produce intelligible speech in this setting, and thus we do not evaluate it. At the same time the proposed model produces intelligible speech with only 2 hours of training data, which suggests that the impact of our method increases in lower data settings.

In this section we investigate the impact of the extra augmentation conditioning. To this end, we train a model with architecture (B) on the 10h variant of $D_1$ dataset augmented using our method, but without applying the additional conditioning. Without conditioning, the augmentation no longer brings improvement over the baseline in the tested setting. Table 4 presents the results of a preference test between the proposed model without conditioning and the baseline model. The baseline has higher preference ratings, although the result is not statistically significant (p-value > 0.05). We also confirm that the proposed model with conditioning is significantly preferred over the model without conditioning (see Table 4 for detailed results that are significant with p-value < 0.05). This suggests that using the augmentation conditioning is important for our method.

### 5. CONCLUSION

In this paper, we present a novel data augmentation technique for TTS that can considerably increase the diversity of text conditionings seen in training. We create new training samples by parse tree constituents substitution in both text and audio. We take inspiration from existing work on data augmentation for generative modelling to address the problem of augmented samples being out-of-distribution.

We show that our technique reduces overfitting in TTS models trained on expressive data. This overfitting reduction translates into improved speech quality according to perceptual tests. We showed effectiveness of our technique on a range of voices and architectures. Additionally, for attention-based architectures we demonstrated that our method greatly improves model’s robustness.
6. REFERENCES


