We propose a non-autoregressive Text-to-Speech model called VarianceFlow, which takes variance information such as pitch or energy as additional input during training. Unlike such models, VarianceFlow uses a Normalizing Flow (NF) module rather than directly, where the module performs modeling of the variance distribution. By performing the variance modeling based on NF, we improve the speech quality and variance controllability of VarianceFlow. In experiments, VarianceFlow outperforms the previous SOTA AR and non-AR TTS models in terms of speech quality. In addition, it provides a more accurate control over the variance information compared to the widely-used baseline non-AR TTS model, FastSpeech 2.

One-to-many problem in Text-to-Speech

When modeling TTS, one-to-many problem should be considered for better performance (i.e. there are many ways to pronounce a single sentence). For AR TTS models, however, the one-to-many problem is naturally resolved to some degree, because it normally learns to generate a mel-spectrogram frame given the previous frames as well as the text. However, AR TTS models have inevitable problems: (1) Slow inference speed; (2) Error vulnerability. Therefore, non-AR TTS models recently have been proposed.

Two types of solutions for Non-AR TTS models to solve the one-to-many problem

Type I: adopting more flexible generative frameworks such as Normalizing Flow or Score-based models (i.e. MSE-based training assumes the Gaussian distribution), e.g. Glow-TTS [1], Grad-TTS [2]

Type II: explicitly using variance information such as pitch or energy during training, which significantly eases the one-to-many problem. It also allows models to explicitly control the variance information, e.g. FastSpeech 2 [3], FastPitch [4]

We solve the problem remaining in FastSpeech 2 (Type II) by adopting the idea used in Type I models.

During training, FastSpeech 2 directly takes the variance information such as pitch or energy as a text input. Meanwhile, it has a module called variance predictor, which is jointly trained to predict the variance information from the text input based on MSE loss. At inference, FastSpeech 2 first predicts the variance information based on the input text using its variance predictor, and then it generates speech using the predicted variance values and text representations.

However, one-to-many problem also exists in predicting the variance information from the text input.