

OSM: One-Shot Multi-Speaker Text-to-Speech

...

Nikolay Kozyrskiy
Gleb Balitskiy

Problem Statement: Deep Learning Task

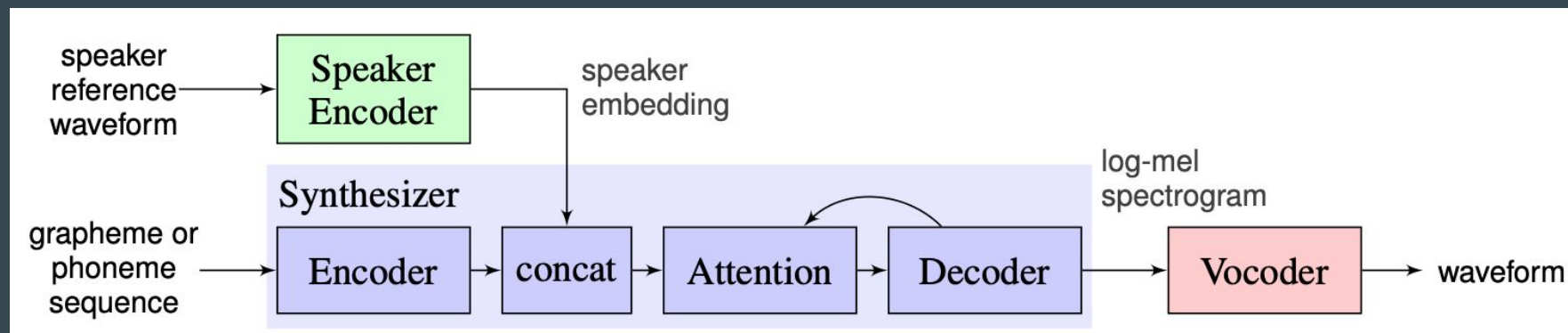
One-Shot Multi-Speaker Text-to-Speech (OS MS TTS) systems are aimed to transform text into speech with voice determined by small single sample. The main problem here is to reproduce the new unseen voice without retraining the network.



Problem Statement: Main approach

Three steps:

1. **Speaker Encoder:** produce voice embeddings to reveal the voice characteristics
2. **Synthesizer:** using text and embeddings produce mel spectrograms of target speech
3. **Vocoder:** transform mel spectrograms from Synthesizer to speech



Problem Statement: Project Goal

The goal: create a flexible framework to properly combine **Speaker Encoder, Synthesizer, Vocoder** and provide replaceable modules and methods in each part.

Motivation:

1. Lack of open source implementations
2. No possibility to replace one module (ex. Speaker Encoder) to verify new idea in existing solution.
3. No well documented solutions



Usage: Our framework will be useful for research groups working TTS tasks.

Main Challenges:


1. Properly designed API for all modules. (Interface for new Datasets and preprocessing functions)
2. Similarity of API for all modules
3. Proper connection of modules





Baseline Solution

 **CorentinJ / Real-Time-Voice-Cloning**  W

[Code](#) [Issues 20](#) [Pull requests](#) [Actions](#) [Wiki](#) [Security](#) [Insights](#)


 master ▾



 1 branch










 0 tags

Go to file

Add file ▾

 Code ▾

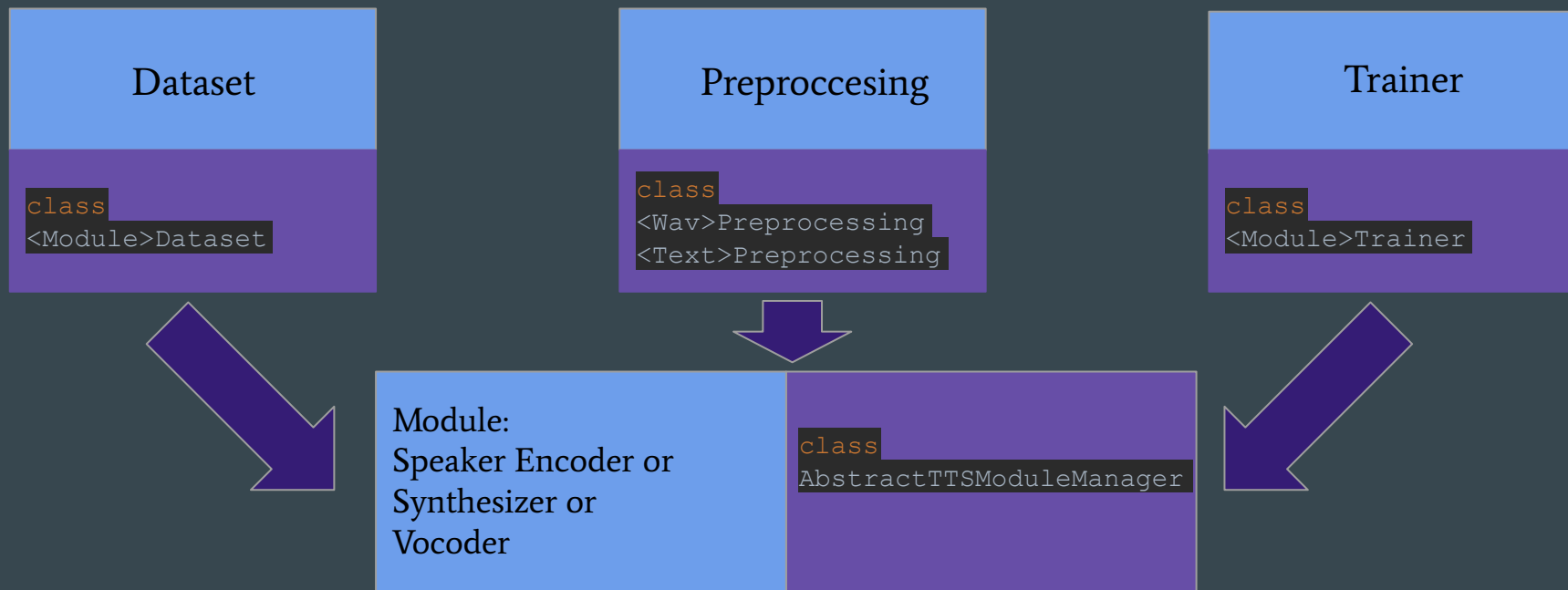
 **Jackson-Kang** Add epsilon to the norm of x to avoid dividing by zero (#678) 95adc69 on 23 Feb  282 commits

 encoder	Add epsilon to the norm of x to avoid dividing by zero (#678)	3 months ago
 samples	Added no_mp3_support argument and added a check for ffmpeg inst...	9 months ago
 synthesizer	Pytorch synthesizer (#472)	3 months ago
 toolbox	Pytorch synthesizer (#472)	3 months ago
 utils	Emphasize error message when model files not found (#668)	3 months ago
 vocoder	Librosa 0.8.0 compatibility (#572)	7 months ago
 .gitattributes	Create .gitattributes	2 years ago
 .gitignore	Fixed missing __init__ file in utils	2 years ago
 LICENSE.txt	Moved root of repo	2 years ago

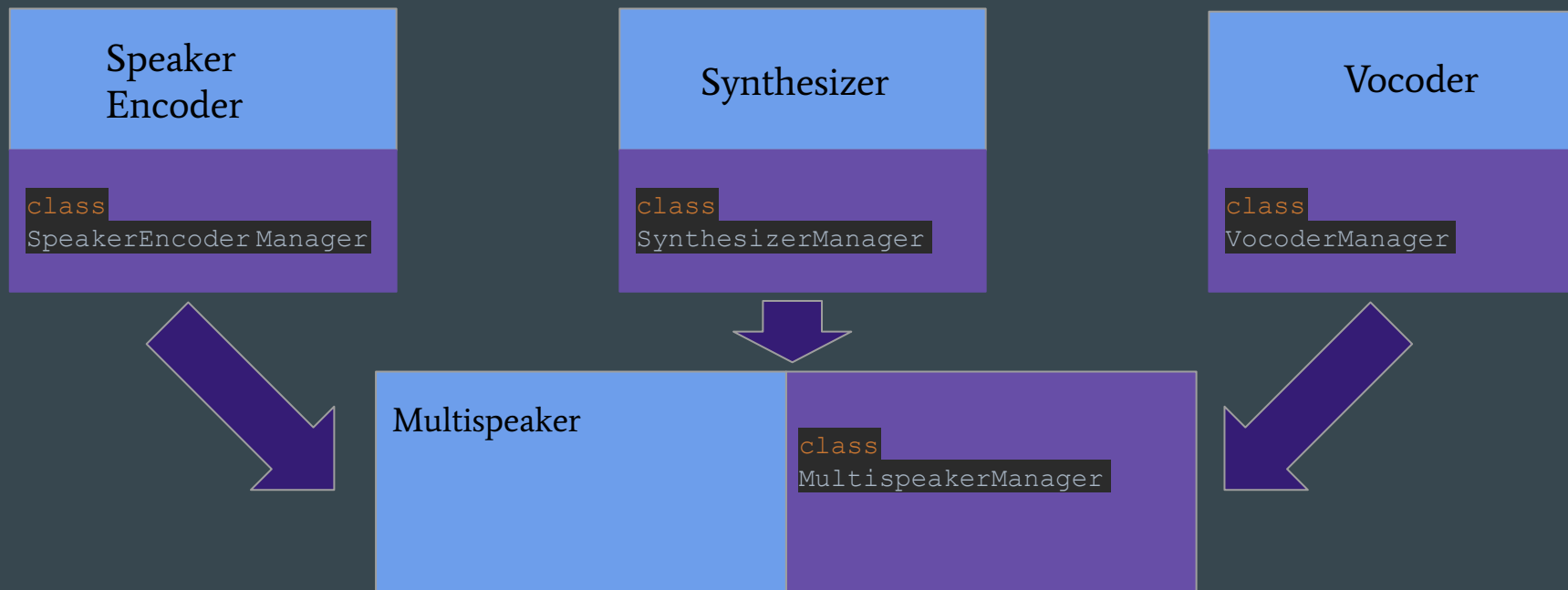
Our Improvement

1. Flexible modular structure
2. Proper API
3. Ability to simply change modules, datasets , and preprocessing procedures.

Project Structure: Main Module Structure



Project Structure: TTS structure



Project Structure: Module methods

1. `__init_trainer(self)` - init trainer (use only if train session is needed)
2. `train_session(self, ...)` - start training session
3. `_load_local_configs(self)` - load configs
4. `_init_baseline_model(self)` - init baseline model
5. `process_<module>(self, ...)` - compute module part during inference

Project Structure: Configuration Files

Updating configs:

1. Get default config via `get_default_<module>_config()`. Change attributes. Update via `update_config(..)`.
2. Create YAML config structured like in default config. Update via method `update_config(..)`.

Project Structure: Datasets and Data Preprocessing

1. Introduced interface to use custom Dataset
2. Introduced interface to use custom data Preprocessing
3. API differs from module to module , due to differences in dataset usage

Project Installation

Run “**pip install .**” from root directory

Summary

We implemented:

1. Working stack of models. Text and audio sample as input and speech as output.
2. Convenient API for external users.
3. Examples how to run and use it / documentation or Jupiter notebooks.
4. Library is installable and model weights are downloaded from a Dropbox upon the first run.
5. Functionality to train new models.