# How to Use This Code

## 1 Introduction

This is the accompanying code for our paper titled *Enhancing Piano Transcription by Dilated Convolution*, which has been accepted by ICMLA 2020 Special Session on Deep Learning. Here we go through the steps to prepare the environment for running this code. The relative paths mentioned hereafter are relative to your Python environment.

## 2 Environment Preparation

- Install Dependent Libraries

| | |
|---|---|
| tensorflow | 1.13.1 |
| librosa | 0.6.2 |
| soundfile | 0.10.2 |
| mido | 1.2.9 |
| magenta | 0.4.0 |
| madmon | 0.16.1 |
| mir_eval | 0.5 |

- There is a defect with the function *apply_sustain_control_changes* provided by magenta for translating the effect of sustain pedal into an extended note duration. This function is contained in script *sequences_lib.py*. We have fixed this defect and uploaded the updated script. Please update the corresponding script after installing magenta. The relative path of this script is *lib/python2.7/site-packages/magenta/music/*.

- Sparse convolution is not a standard operation coming with Tensorflow so we implemented it ourselves. The code for this operation is contained in script *harmonic_dense.py*. Add this operation to Tensorflow as per the following steps.

  - Append all the stuff in *harmonic_dense.py* to script *layers.py* whose relative path is *lib/python2.7/site-packages/tensorflow/contrib/layers/python/layers/*.
  - At the beginning of *layers.py*, there is a list variable named *__all__*. Append '*harmonic_dense*' to this variable.

  - There is a script *__init__.py* controlling the visibility of the operations defined in *layers.py*. We need to make the newly added operation visible. The relative path of this script is *lib/python2.7/site-packages/tensorflow/contrib/layers/python/*. This is done by adding *@@harmonic_dense* to the long list of similar terms at the beginning of this script.

- Add the function for note tracking to the *mir_eval* library. We will use a customized method to extract note onsets and offsets. This method is not available in mir_eval. This method has been implemented as a function stored in script *onset_frame_transcription_performance_fn.py*. Please append the content of this script to script *transcription.py* of mir_eval. The relative path of transcription.py is *lib/python2.7/site-packages/mir_eval/*.

## 3 Download Datasets

- MAPS
  Download the MAPS dataset as per the instruction given in the paper below.

  V. Emiya, R. Badeau, and B. David, "Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle," *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6), pp. 1643—1654, 2010.

  You may need to unzip the downloaded file. Then create an environment variable named *maps* pointing to the top folder of this dataset. Make sure the following 9 folders are under this folder.
  *ENSTDkCl_2/MUS*
  *ENSTDkAm_2/MUS*
  *AkPnBcht_2/MUS*
  *AkPnBsdf_2/MUS*
  *AkPnCGdD_2/MUS*
  *AkPnStgb_2/MUS*
  *SptkBGAm_2/MUS*
  *SptkBGCl_2/MUS*
  *StbgTGd2_2/MUS*

- MAESTRO
  Download the MAESTRO dataset from the link given in the paper below.

  C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C. A. Huang, S. Dieleman, E. Elsen, J. H. Engel, and D. Eck, "Enabling factorized piano music modeling and generation with the MAESTRO dataset," in *7th International Conference on Learning Representations*, ICLR 2019, New Orleans, LA, USA, May 6–9, 2019.

Note that this dataset has multiple versions. The version we used is v1.0.0. You may need to untar the downloaded file. Then create an environment variable named *maestro* pointing to the top folder of this dataset.

## 4 Generate Spectrograms

- Mel-Spectrograms
  The scripts that generate mel-spectrograms for MAPS and MAESTRO are *gen_mel_for_maps.py* and *gen_mel_for_maestro.py*, respectively. Before running them, please create two environment variables named *maps_mel* and *maestro_mel* pointing, respectively, to the folders where the spectrograms for maps and those for maestro will be stored. These folders will be created automatically if they do not exist beforehand.
- VQT Spectrograms
  We implemented functions for computing VQT in Matlab and will call them from Python to generate VQT spectrograms. Please follow the steps below to generate VQT spectrograms.
  1. Install Matlab.
  2. Follow the instruction at this link to install necessary libraries so that we can call Matlab functions from Python.
  3. The Matlab code for VQT is put in folder *matlab_code*. Please add this folder to Matlab's search path.
  4. The Python scripts that generate VQT spectrograms for MAPS and MAESTRO are *gen_vqt_for_maps.py* and *gen_vqt_for_maestro.py*, respectively. Before running them, please create two environment variables named *maps_vqt* and *maestro_vqt* pointing to the folders where the spectrograms will be saved.
- STFTs for MAPS
  The script that generate STFTs for MAPS is *gen_stft_for_maps.py*. Before running it, please create an environment variable named *maps_stft* pointing to the folder where the spectrograms will be stored. The folder will be created automatically if it does not exist beforehand.
- HCQTs for MAPS
  The script that generate HCQTs for MAPS is *gen_hcqt_for_maps.py*. Before running it, please create an environment variable named *maps_hcqt* pointing to the folder where the spectrograms will be stored. The folder will be created automatically if it does not exist beforehand.

## 5 Run Experiments

- Experiments for Comparing Acoustic Models

  - Script *comp_acoustic_models.py* implements three acoustic models listed in Table II, namely, the dense model, the harmonic model, and the dilated model. Before running this script, you need to configure some parameters, such as the acoustic model, the dataset for training, and the prefetch ration for the training split. Please refer to the script for details.
  - Scripts *h1.py* and *h6.py* implement the acoustic models HSg(1) + dense and HSg(6) + dense listed in Table II, respectively.
  - Script *hcqt.py* implements the fully convolutional acoustic model listed in Table II.

- Experiments for the Performance of Our AMT System
  The are three scripts under folder *note_tracking*, namely, *frame.py*, *onset.py* and *note.py*, which implements, respectively, the frame-wise pitch detector, the onset detector, and the extraction of note contours. You should run the first two scripts first to get the checkpoints for the two detectors so as to run the third script. Before running the scripts, you need to define some parameters, such as the dataset for training, and the prefetch ratio for the training split. Please refer to the scripts for details.

- You can view the results of the above experiments with tensorboard.