**How to make the CycleGAN VC2 work?**

**Run the file without changing anything:**

* Install the requirements using the pip install function. For pyWORLD, see the first command: <https://docs.anaconda.com/anacondaorg/user-guide/howto/#use-packages>
* Run the preprocess\_training.py file first
* Run the train.py file next. If this is the first time that you train the data, be sure to uncomment the “resume\_training\_at = None #Used not to resume training” line. Otherwise, comment it. (line 471)

**Some information about the structure:**

* The way the neural network is coded is done in three separate files. First, you have the “preprocess” file that convert the audio data into a vector
* We have the “preprocess\_training.py” that allows to create two datasets for the training A and B
* We then have the “model\_tf.py” file, where the CycleGAN code is. It codes the Generator and the Discriminator with classes.
* Finally, we have the “train.py” file that allows to train the CycleGAN by calling each function. It creates a “CycleGAN\_training” class to do so. **This code allows to generate either from A to B, or from B to A.** (line 57 – 60, we can see that there is 2 discriminators A and B, and two generators A2B and B2A). At the end of the code, each of the training set and target set are generated and “permuted”.

**Some information to tune the parameters:**

* Line 71 & 72 of the “train.py” file, we can find the learning rate + the learning rate decay used by the person that designed the model. **Originally the learning rate is = 0.0002 for the generator and 0.001 for the discriminator**
* We can also tune all the residual blocks parameters line 171 of “model\_tf.py”
* The size of a mini\_batch line 37 of “train.py” **Originally = 1**

Probably other to add!

**To generate the files:**

* The converted files are automatically generated at the end of the training and are placed in folders (output\_A\_dir\_default = “path of the saving”). To generate new files, we train the two datasets, and each of the data set will be converted to new audio files in the converted\_sound folder.
* The functions that handle this file saving are: “validation\_for\_A\_dir” and “validation\_for\_B\_dir” (to be confirmed but it seems to be the case). To generate those files, some arguments have to be fed, especially:

num\_mcep = 36

sampling\_rate = 16000

frame\_period = 5.0

n\_frames = 128

Those parameters seem important to convert the coded spectrogram to the audiofile back. **Those are certainly needed to be modified based on our sound data!**