Preprocessing

Use other tools to better study and choose the preprocessing method to build a spectrogram from an audio file, and then replicate the method in the python code

(e.g. Labview, Matlab, etc...)

Kaggle: TreeVibes

Link: https://www.kaggle.com/datasets/potamitis/treevibes

Dataset
More than 58K audio files of red palm weevils, organized as clean / infested samples (.wav format)

Python libraries
-Tensorflow
-Keras
-TFLite_Runtime
-Matplot
Etc...

Python code

-Choosing the ML model to use (e.g. Densenet121)
-Setting training hyper parameters (batch-size, epochs, etc…)
-Setting the size of training/validate sets to use to train the model
-Setting STFT parameters (SR, N_FFT, HOP_LEN, etc...)
-Preprocessing the audio files to obtain spectrograms
-Feeding spectrograms to the model
-Training the model

Training results:

-Accuracy
-Precision
-Recall
-F1-score
-Confusion matrix

model.tflite

(it's a binary file that contains the model structure, with also weights and biases of the trained model)

Arm NN Build Tool

Link:
https://github.com/ARM-software/armnn/tree/branches/armnn_24_08/build-tool

Download Arm NN binaries with:
git clone https://github.com/ARM-software/armnn.git armnn

Move to armnn/build-tool with:
cd armnn/build-tool

Build the docker for your armv8 architecture using 64 bit OS:
sudo docker build --no-cache --build-arg SETUP_ARGS="--target-arch=aarch64 --all" --build-arg BUILD_ARGS="--target-arch=aarch64 --tflite-classic-delegate --tflite-parser --neon-backend --ref-backend" --tag armnn:aarch64 --file docker/Dockerfile .

Check the docker with:
sudo docker ps -a

Run the docker with:
sudo docker run --name armnn_build_container -it armnn:aarch64 /bin/bash
After this command, you are located inside the docker at "home/arm-user"

Take note of your <containerID> (left side of the bash):
e.g if you see: arm-user@ccad50e1ad71:~$
then the <containerID> is: ccad50e1ad71

***Check the presence of the tarball generated containing the libraries and take note of its name (e.g armnn_aarch64_build.tar.gz) with**:*
arm-user@ccad50e1ad71:~$ls
The name of the tarball will be something like: armnn_aarch64_build.tar.gz

Exit from the docker with:
arm-user@ccad50e1ad71:~$exit

Create a folder while you are in armnn/build-tool with:
mkdir dst_folder

Copy the tarball in the destination folder with:
sudo docker cp <containerID>:/home/arm-user/armnn_aarch64_build.tar.gz dst_folder/

model.tflite

Move model.tflite and armnn_aarch64_build.tar.gz to the Raspberry-Pi with:

scp model.tflite armnn_aarch64_build.tar.gz pi@192.168.1.x:/home/pi

armnn_aarch64_build.tar.gz

Raspberry-Pi 3B with 64 bit OS

-Extract the tarball in a folder (e.g. create the folder aarch64_build_ref-neon) inside */home/pi/ (a.k.a.* the HOME directory).
So, inside /home/pi/aarch64_build_ref-neon you will see the libraries like libarmnn.so.33 and libarmnnTfLiteParser.so.24, and so on…
-Since these libraries are invisible to the OS when compiling, you have to export this path doing:
**pi@pi:~/aarch64_build_ref-neon $** export LD_LIBRARY_PATH=/home/pi/aarch64_build_ref-neon:$LD_LIBRARY_PATH
-Now make your cpp code for inference, then build it and run it. For example, after you made your code, build it with the command:
pi@pi:~/aarch64_build_ref-neon $ g++ inference_RPW.cpp -o inference_RPW_exe -g -I/home/pi/aarch64_build_ref-neon/include -I/usr/include/opencv4 -L/home/pi/aarch64_build_ref-neon/ -larmnn -larmnnTfLiteParser -lpthread -lopencv_core -lopencv_imgcodecs -lopencv_imgproc -lm -lsndfile -lfftw3
-And run the executable obtained by the build process with:
pi@pi:~/aarch64_build_ref-neon $ ./inference_RPW_exe model.tflite audio_file.wav CpuAcc

PS:
-The Raspberry-Pi 3B could work in 32 bit or 64 bit. If it's loaded a 32 bit OS in it, it will work as armv7l, while it will work as armv8 (aarch64) if 64 bit OS instead (check the architecture with the command: **uname -m**).
-The ArmNN library works only for armv8 and armv8-2, thus a 64 bit OS is mandatory (e.g. the default 64 bit Raspbian).