MINISTRY OF SCIENCE AND HIGHER EDUCATION OF THE RUSSIAN FEDERATION

Federal State Autonomous Educational Institution of Higher Education

“South Ural State University (National Research University)”

School of Electrical Engineering and Computer Science

Department of Electronic Computing Machines

“DEVELOPMENT OF A SPEECH RECOGNITION LIBRARY FOR ULTRA LOW POWER DEVICES”

for the master graduate qualification work of
A student of the group KE-228: F . K. Chemorion
Supervisor: D.V. Topolsky, PhD, Associate Professor

South Ural
State University

National Research
University

1943

Introduction

This project involves the creation of an ultra-low-power software complex that will enable development of real time speech recognition systems on small boards without need for internet

Relevance and Novelty

Many of the current speech recognition solutions are cloud based and require a lot of computing power and electricity to run locally. This project aims at developing a speech recognition device that can run offline on very small devices.

Tasks necessary to achieve the goal:

1. Analyzing the market for existing Libraries
2. Analyzing technological solutions to use to solve the problem
3. Design a model architecture for MBEDSpeech
4. Training the MBEDSpeech Model
5. Testing the model
6. Compiling a binary and a Library for Arduino
7. Testing the Arduino Library using Arduino Lint
8. Testing the template sketch for successful compilation

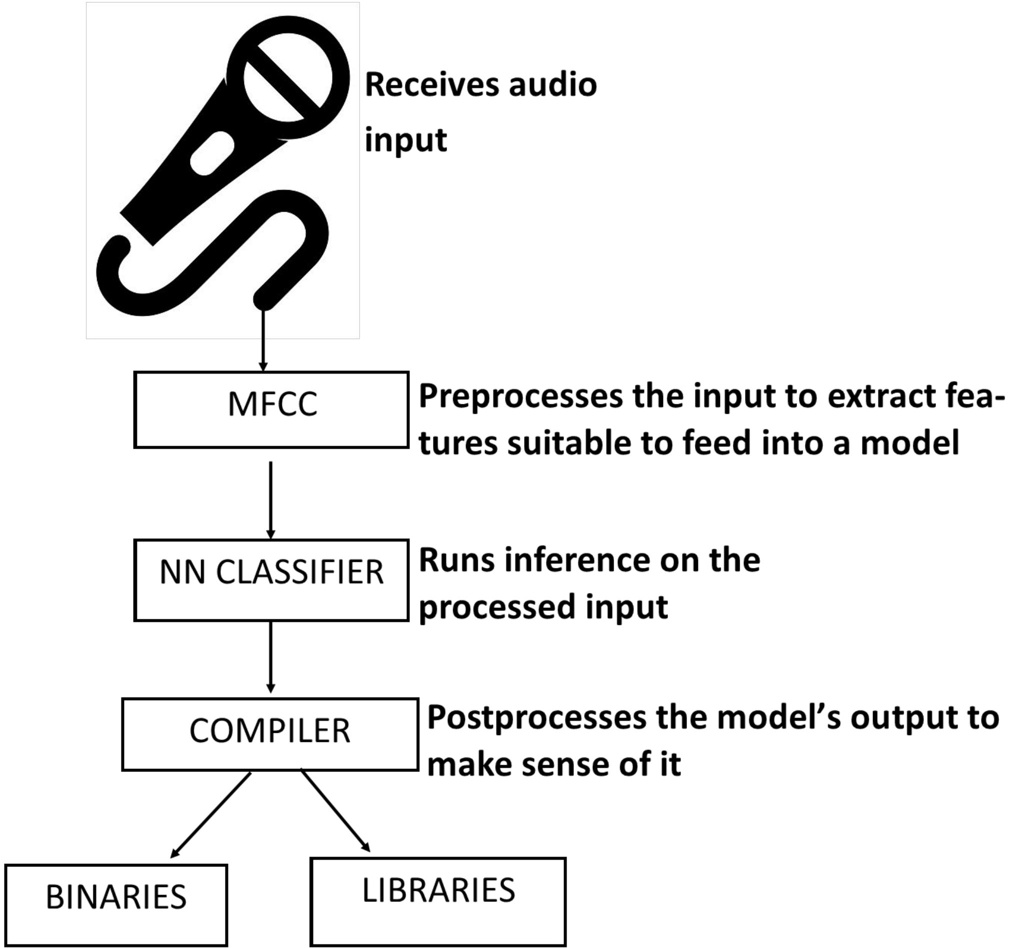
Task 1: Analyzing the market for Libraries

- Google Speech API
- Kaldi NL
- Speechmatics

Task 2 : Technological Solutions

1. Google Speech Dataset
2. Tensorflow
3. Keras
4. Github
5. Google Colab

Task 3: Architecture Design



Receives audio input

MFCC

Preprocesses the input to extract features suitable to feed into a model

NN CLASSIFIER

Runs inference on the processed input

COMPILER

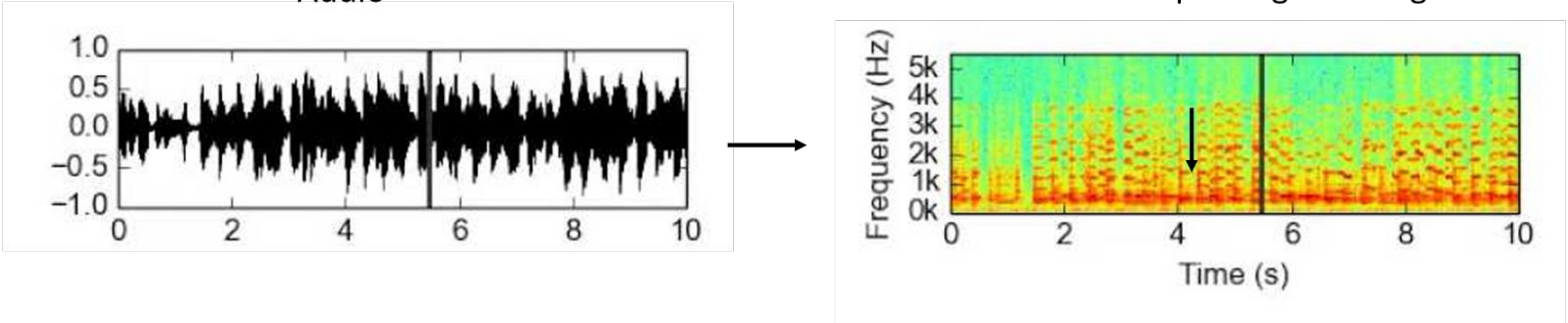
Postprocesses the model's output to make sense of it

BINARIES

LIBRARIES

Task 4: Training the MBEDSpeech model

a) MFCC

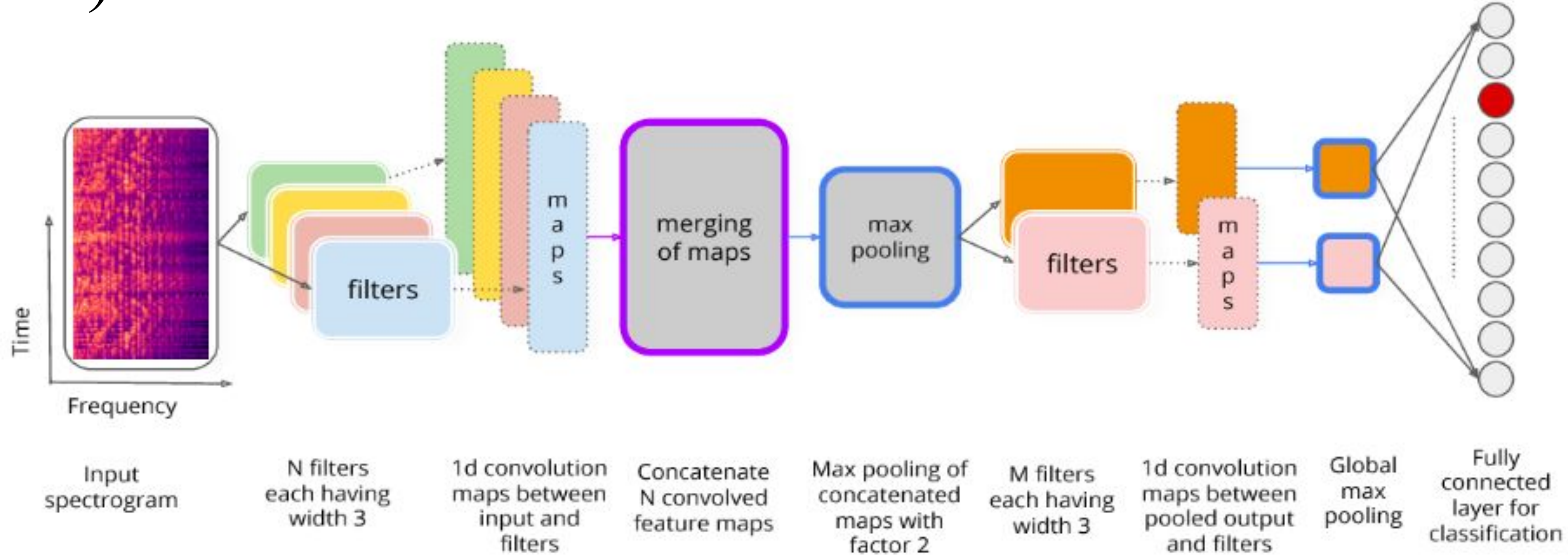


Audio

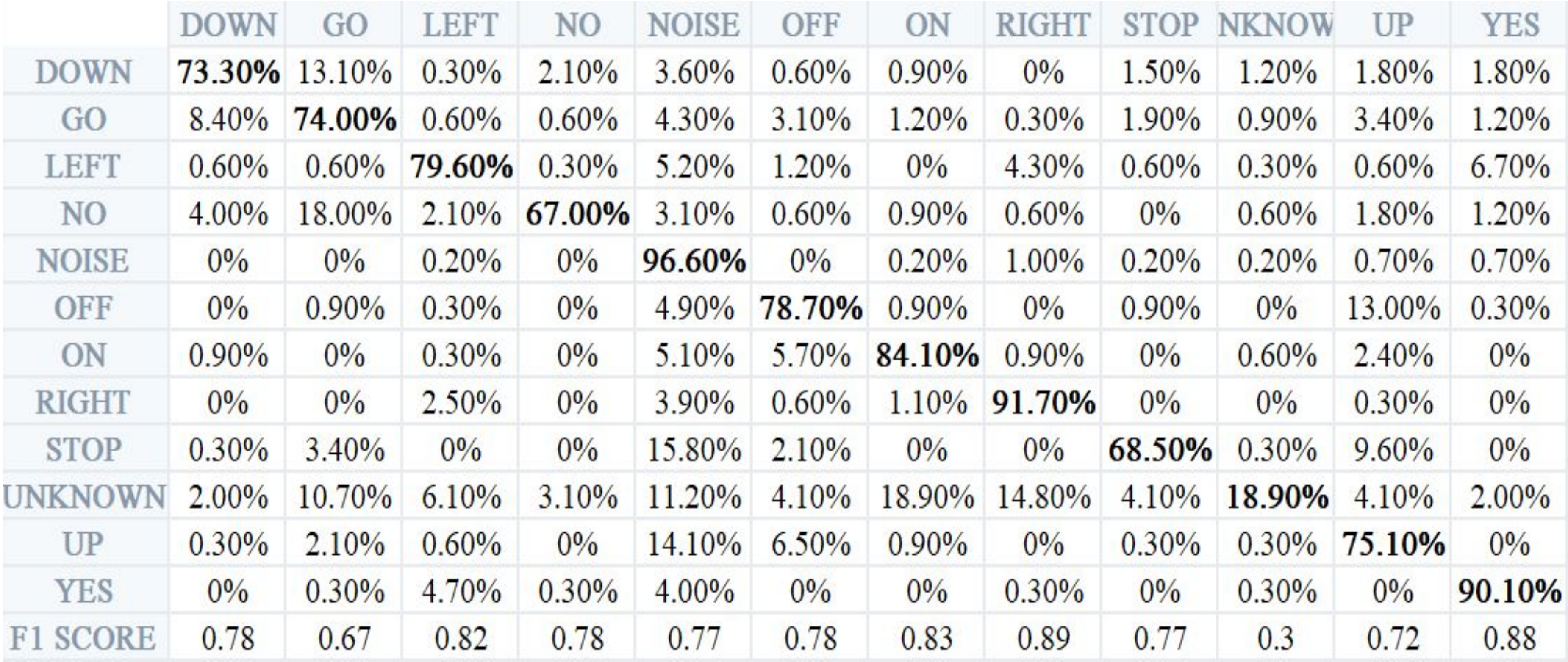
Extracted spectrogram image

Task 4: Training the MBEDSpeech model continued

b) The Neural Network Classifier

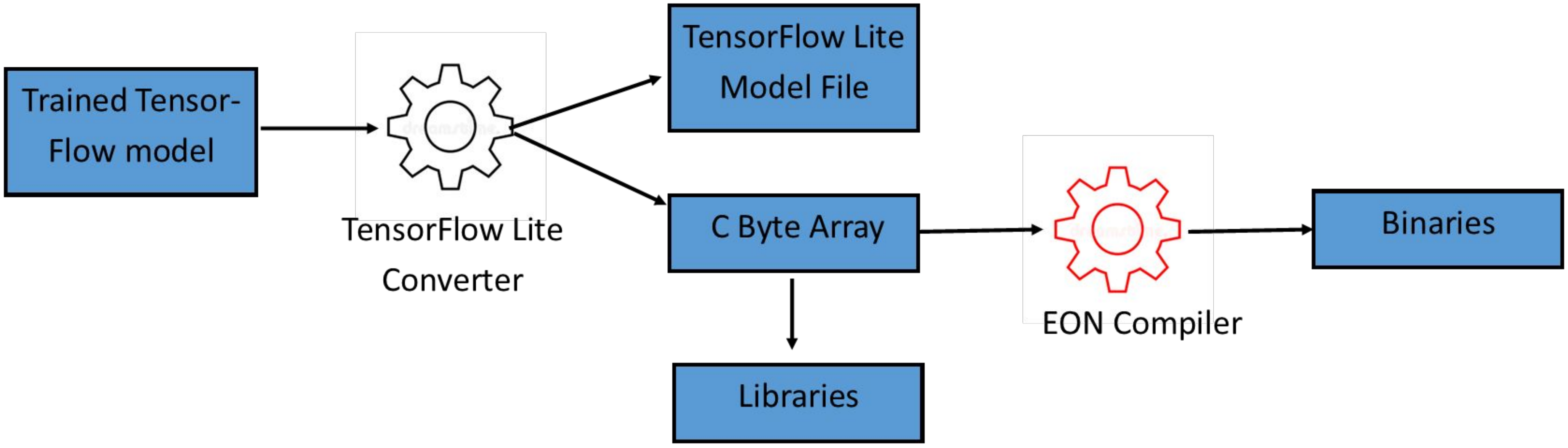

Time
Frequency
filters
maps
merging of maps
max pooling
filters
maps
Input spectrogram
N filters each having width 3
1d convolution maps between input and filters
Concatenate N convolved feature maps
Max pooling of concatenated maps with factor 2
M filters each having width 3
1d convolution maps between pooled output and filters
Global max pooling
Fully connected layer for classification

Task 5: Testing the model - confusion matrix

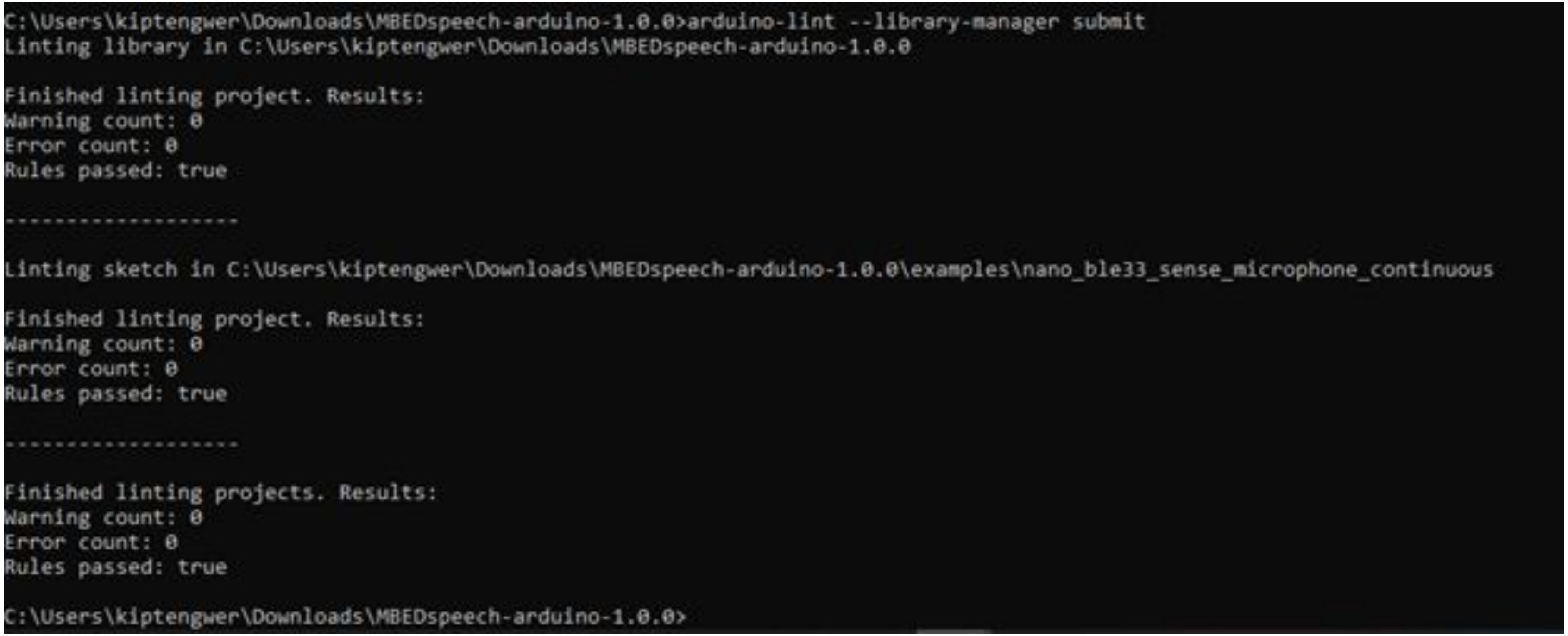


	DOWN	GO	LEFT	NO	NOISE	OFF	ON	RIGHT	STOP	NKNOW	UP	YES
DOWN	**73.30%**	13.10%	0.30%	2.10%	3.60%	0.60%	0.90%	0%	1.50%	1.20%	1.80%	1.80%
GO	8.40%	**74.00%**	0.60%	0.60%	4.30%	3.10%	1.20%	0.30%	1.90%	0.90%	3.40%	1.20%
LEFT	0.60%	0.60%	**79.60%**	0.30%	5.20%	1.20%	0%	4.30%	0.60%	0.30%	0.60%	6.70%
NO	4.00%	18.00%	2.10%	**67.00%**	3.10%	0.60%	0.90%	0.60%	0%	0.60%	1.80%	1.20%
NOISE	0%	0%	0.20%	0%	**96.60%**	0%	0.20%	1.00%	0.20%	0.20%	0.70%	0.70%
OFF	0%	0.90%	0.30%	0%	4.90%	**78.70%**	0.90%	0%	0.90%	0%	13.00%	0.30%
ON	0.90%	0%	0.30%	0%	5.10%	5.70%	**84.10%**	0.90%	0%	0.60%	2.40%	0%
RIGHT	0%	0%	2.50%	0%	3.90%	0.60%	1.10%	**91.70%**	0%	0%	0.30%	0%
STOP	0.30%	3.40%	0%	0%	15.80%	2.10%	0%	0%	**68.50%**	0.30%	9.60%	0%
UNKNOWN	2.00%	10.70%	6.10%	3.10%	11.20%	4.10%	18.90%	14.80%	4.10%	**18.90%**	4.10%	2.00%
UP	0.30%	2.10%	0.60%	0%	14.10%	6.50%	0.90%	0%	0.30%	0.30%	**75.10%**	0%
YES	0%	0.30%	4.70%	0.30%	4.00%	0%	0%	0.30%	0%	0.30%	0%	**90.10%**
F1 SCORE	0.78	0.67	0.82	0.78	0.77	0.78	0.83	0.89	0.77	0.3	0.72	0.88

Task 6: Compiling a binary and a Library for Arduino


Trained Tensor-Flow model
TensorFlow Lite Converter
TensorFlow Lite Model File
C Byte Array
Libraries
EON Compiler
Binaries

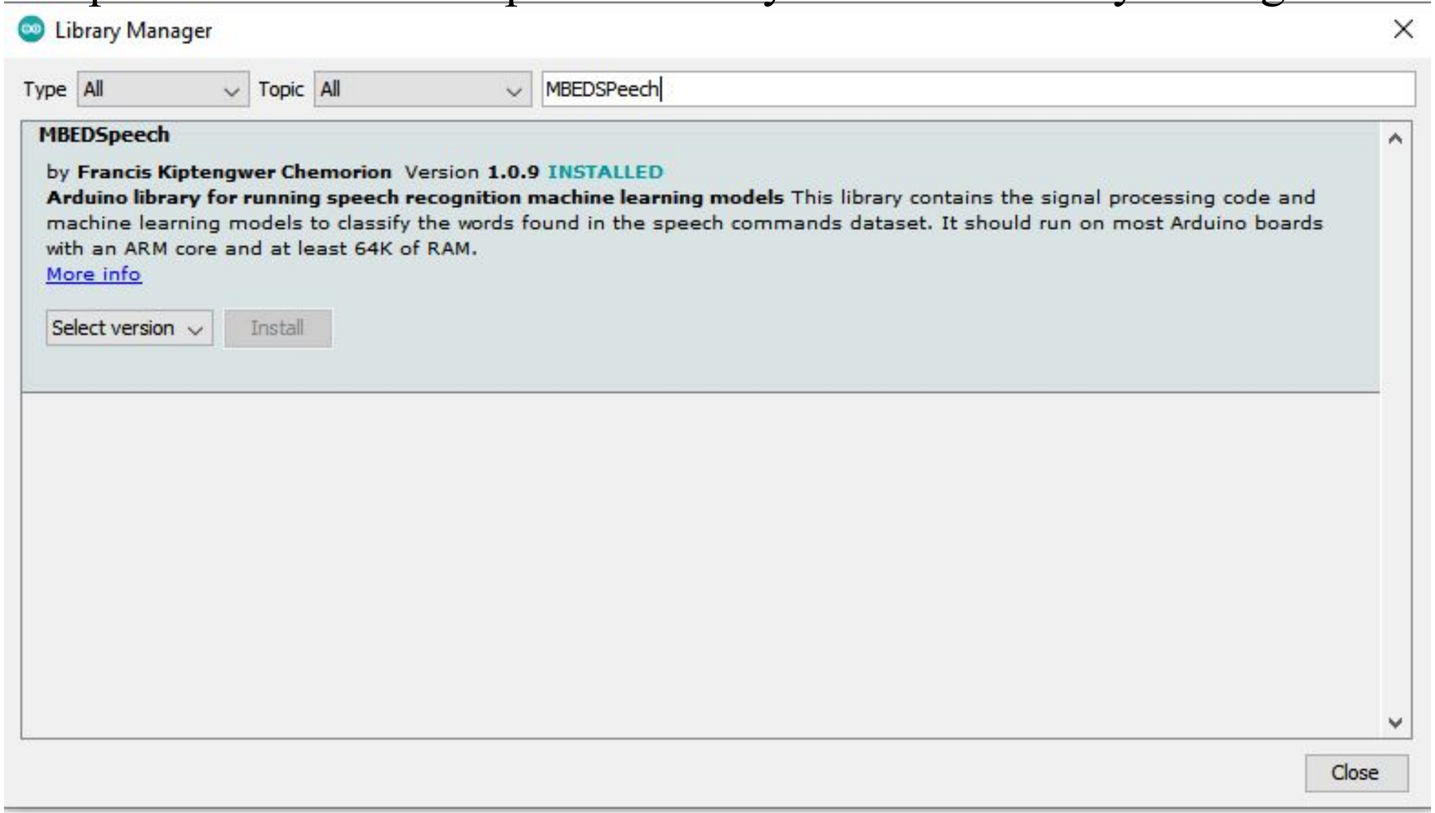
7. Testing the Arduino Library using Arduino Lint



```
C:\Users\kiptengwer\Downloads\MBEDspeech-arduino-1.0.0>arduino-lint --library-manager submit
Linting library in C:\Users\kiptengwer\Downloads\MBEDspeech-arduino-1.0.0

Finished linting project. Results:
Warning count: 0
Error count: 0
Rules passed: true

-------------------

Linting sketch in C:\Users\kiptengwer\Downloads\MBEDspeech-arduino-1.0.0\examples\nano_ble33_sense_microphone_continuous

Finished linting project. Results:
Warning count: 0
Error count: 0
Rules passed: true

-------------------

Finished linting projects. Results:
Warning count: 0
Error count: 0
Rules passed: true

C:\Users\kiptengwer\Downloads\MBEDspeech-arduino-1.0.0>
```

8. Testing the example sketch for successful compilation

Step 1 : Install MBEDSpeech Library from the Library manager



Library Manager

Type: All — Topic: All — MBEDSPeech

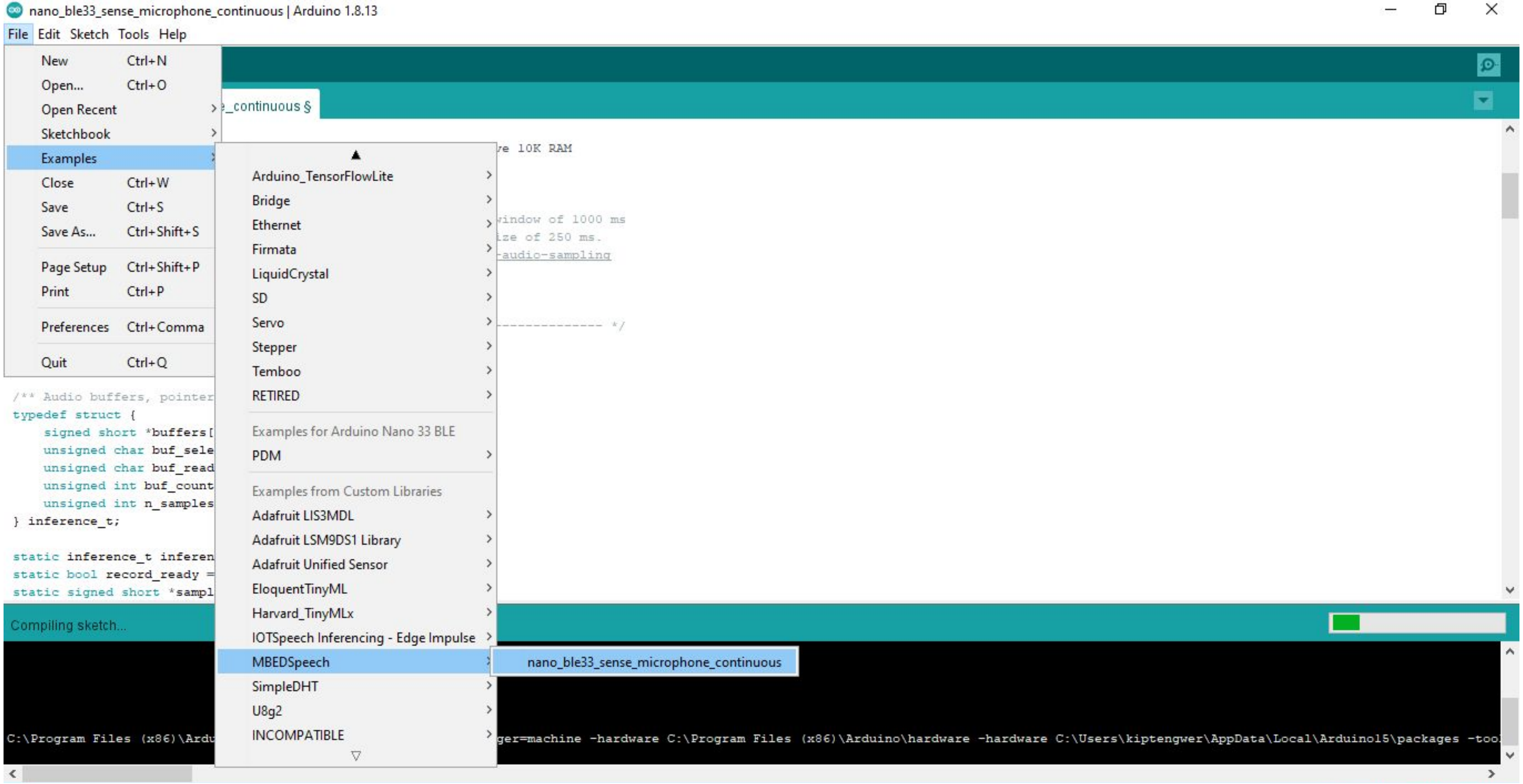
MBEDSpeech

by **Francis Kiptengwer Chemorion** Version **1.0.9** INSTALLED

Arduino library for running speech recognition machine learning models This library contains the signal processing code and machine learning models to classify the words found in the speech commands dataset. It should run on most Arduino boards with an ARM core and at least 64K of RAM.

More info

Step 2 : Look for MBEDSpeech in examples



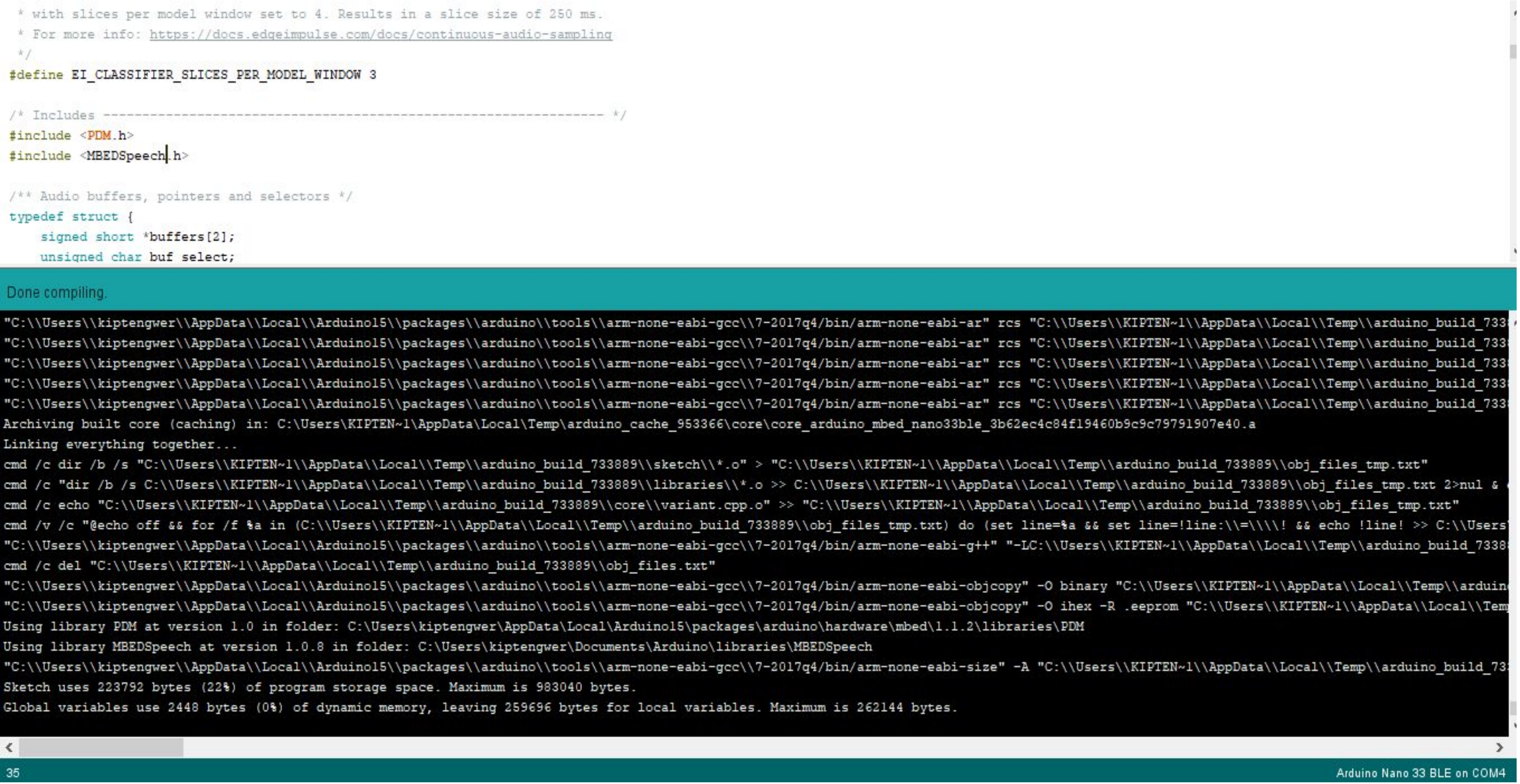
```
re 10K RAM

window of 1000 ms
ize of 250 ms.
-audio-sampling

-------------- */
```

```
/** Audio buffers, pointer
typedef struct {
    signed short *buffers[
    unsigned char buf_sele
    unsigned char buf_read
    unsigned int buf_count
    unsigned int n_samples
} inference_t;

static inference_t inferen
static bool record_ready =
static signed short *sampl
```

Compiling sketch...

```
C:\Program Files (x86)\Ardu ger=machine -hardware C:\Program Files (x86)\Arduino\hardware -hardware C:\Users\kiptengwer\AppData\Local\Arduino15\packages -too
```

Step 3 : Compile example sketch (compiles successfully)

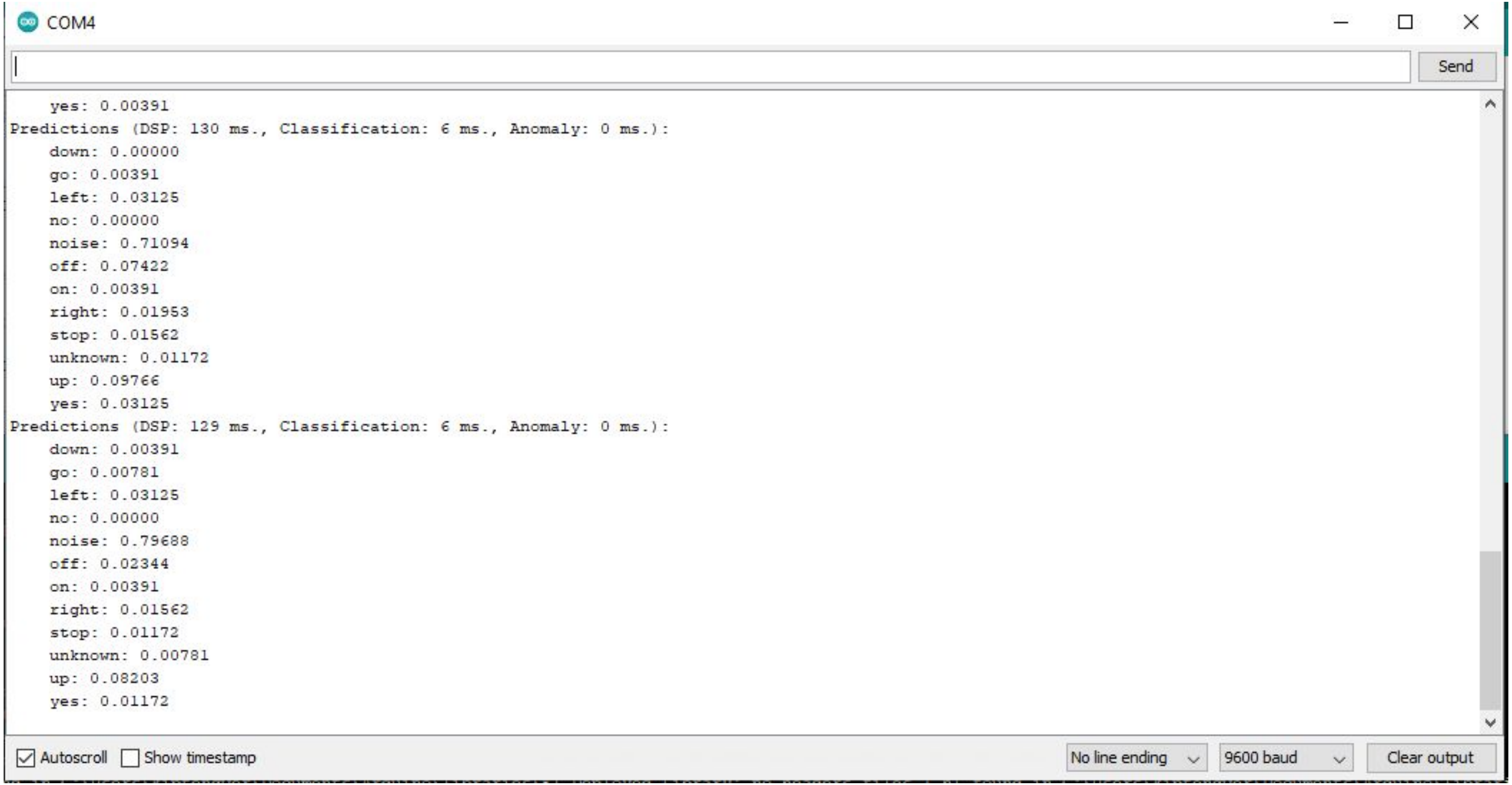


```
 * with slices per model window set to 4. Results in a slice size of 250 ms.
 * For more info: https://docs.edgeimpulse.com/docs/continuous-audio-sampling
 */
#define EI_CLASSIFIER_SLICES_PER_MODEL_WINDOW 3

/* Includes ---------------------------------------------------------------- */
#include <PDM.h>
#include <MBEDSpeech.h>

/** Audio buffers, pointers and selectors */
typedef struct {
    signed short *buffers[2];
    unsigned char buf_select;
```

Done compiling.

```
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-ar" rcs "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-ar" rcs "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-ar" rcs "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-ar" rcs "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-ar" rcs "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733
Archiving built core (caching) in: C:\Users\KIPTEN~1\AppData\Local\Temp\arduino_cache_953366\core\core_arduino_mbed_nano33ble_3b62ec4c84f19460b9c9c79791907e40.a
Linking everything together...
cmd /c dir /b /s "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733889\\sketch\\*.o" > "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733889\\obj_files_tmp.txt"
cmd /c "dir /b /s C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733889\\libraries\\*.o >> C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733889\\obj_files_tmp.txt 2>nul &
cmd /c echo "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733889\\core\\variant.cpp.o" >> "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733889\\obj_files_tmp.txt"
cmd /v /c "@echo off && for /f %a in (C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733889\\obj_files_tmp.txt) do (set line=%a && set line=!line:\\=\\\\! && echo !line! >> C:\\Users
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-g++" "-LC:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_7338
cmd /c del "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_733889\\obj_files.txt"
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-objcopy" -O binary "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduin
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-objcopy" -O ihex -R .eeprom "C:\\Users\\KIPTEN~1\\AppData\\Local\\Tem
Using library PDM at version 1.0 in folder: C:\Users\kiptengwer\AppData\Local\Arduino15\packages\arduino\hardware\mbed\1.1.2\libraries\PDM
Using library MBEDSpeech at version 1.0.8 in folder: C:\Users\kiptengwer\Documents\Arduino\libraries\MBEDSpeech
"C:\\Users\\kiptengwer\\AppData\\Local\\Arduino15\\packages\\arduino\\tools\\arm-none-eabi-gcc\\7-2017q4/bin/arm-none-eabi-size" -A "C:\\Users\\KIPTEN~1\\AppData\\Local\\Temp\\arduino_build_73
Sketch uses 223792 bytes (22%) of program storage space. Maximum is 983040 bytes.
Global variables use 2448 bytes (0%) of dynamic memory, leaving 259696 bytes for local variables. Maximum is 262144 bytes.
```

Step 4 : Flash code to MCU memory the check serial monitor for results



```
    yes: 0.00391
Predictions (DSP: 130 ms., Classification: 6 ms., Anomaly: 0 ms.):
    down: 0.00000
    go: 0.00391
    left: 0.03125
    no: 0.00000
    noise: 0.71094
    off: 0.07422
    on: 0.00391
    right: 0.01953
    stop: 0.01562
    unknown: 0.01172
    up: 0.09766
    yes: 0.03125
Predictions (DSP: 129 ms., Classification: 6 ms., Anomaly: 0 ms.):
    down: 0.00391
    go: 0.00781
    left: 0.03125
    no: 0.00000
    noise: 0.79688
    off: 0.02344
    on: 0.00391
    right: 0.01562
    stop: 0.01172
    unknown: 0.00781
    up: 0.08203
    yes: 0.01172
```

☑ Autoscroll □ Show timestamp

No line ending

9600 baud

Future considerations

1. Optimizing Latency
2. Optimizing Power Usage
3. Optimizing Model and Binary size

Conclusion

This project will enable us to be able to run speech recognition machine learning algorithms on very tiny devices.

https://github.com/kchemorion/MBEDSpeech.git

https://www.arduinolibraries.info/libraries/mbed-speech

https://colab.research.google.com/drive/1h4Eyq9ZKDQuO1SSIA3C_r1N5t2w7Lap2?usp=sharing

Thank you for your attention!