SOUND EVENT DETECTION WITH MACHINE LEARNING

Jon Nordby
Head of Data Science & Machine Learning
Soundsensing AS
jon@soundsensing.no

EuroPython 2021
INTRODUCTION
ABOUT SOUNDENSING

1. Sensor

2. Dashboard + API
SOUND EVENT DETECTION

Given input audio
return the timestamps (start, end)
for each event class
EVENTS AND NON-EVENTS

Events are sounds with a clearly-defined duration or onset.

<table>
<thead>
<tr>
<th>Event (time limited)</th>
<th>Class (continous)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car passing</td>
<td>Car traffic</td>
</tr>
<tr>
<td>Honk</td>
<td>Car traffic</td>
</tr>
<tr>
<td>Word</td>
<td>Speech</td>
</tr>
<tr>
<td>Gunshot</td>
<td>Shooting</td>
</tr>
</tbody>
</table>
APPLICATION

Fermentation tracking when making alcoholic beverages. Beer, Cider, Wine, etc.
ALCOHOL IS PRODUCED VIA FERMENTATION
AIRLOCK ACTIVITY
FERMENTATION TRACKING

Fermentation activity can be tracked as Bubbles Per Minute (BPM).
OUR GOAL

Make a system that can track fermentation activity, outputting Bubbles per Minute (BPM), by capturing airlock sound using a microphone, using Machine Learning to count each “plop”
MACHINE LEARNING NEEDS DATA!
SUPERVISED MACHINE LEARNING

Strongly-labeled

Weakly-labeled

Unlabeled

Test waveform

Sound Event Detector

Classes

Time

Alarm

Vacuum Cleaner

Cat

Cat
DATA REQUIREMENTS: QUANTITY

Need *enough* data.

<table>
<thead>
<tr>
<th>Instances per class</th>
<th>Suitability</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Minimal</td>
</tr>
<tr>
<td>1000</td>
<td>Good</td>
</tr>
<tr>
<td>10000+</td>
<td>Very good</td>
</tr>
</tbody>
</table>
DATA REQUIREMENTS: QUALITY

Need *realistic* data. Capturing natural variation in

- the event sound
- recording devices used
- recording environment
CHECK THE DATA
UNDERSTAND THE DATA

Note down characteristics of the sound

- Event length
- Distance between events
- Variation in the event sound
- Changes over time
- Differences between recordings
- Background noises
- Other events that could be easily confused
import pandas
labels = pandas.read_csv(path, sep='	', header=None,
                        names=['start', 'end', 'annotation'],
                        dtype=dict(start=float, end=float, annotation=str))
MACHINE LEARNING SYSTEM
AUDIO ML PIPELINE OVERVIEW

0.5 seconds Audio

Feature Extractor

32 frequency bands x 8 time frames Spectrogram

Classifier

Predictions for analysis windows

Event Tracker

Timeline of events

Class 1: Bubble
Class 2: ...

start end

Statistics estimator

Bubbles Per Minute (BPM)

Output
import librosa

audio, sr = librosa.load(path)
spec = librosa.feature.melspectrogram(y=audio, sr=sr)
spec_db = librosa.power_to_db(spec, ref=np.max)

lr.display.specshow(ps_db, x_axis='time', y_axis='mel')
from tensorflow import keras
from keras.layers import Convolution2D, MaxPooling2D

model = keras.Sequential([  
    Convolution2D(filters, kernel,  
                  input_shape=(bands, frames, channels)),
    MaxPooling2D(pool_size=pool),
    ....
])
Figure 3: FRR vs. FA per hour for the test set with various SNR values.
EVENT TRACKER

Converting to discrete list of events

- Threshold the probability from classifier
- Keep track of whether we are currently in an event or not

```python
if not inside_event and probability >= on_threshold:
    inside_event = True
    print('EVENT on', t, probability)
if inside_event and probability <= off_threshold:
    inside_event = False
    print('EVENT off', t, probability)
```
STATISTICS ESTIMATOR

To compute the Bubbles Per Minute

- Using the typical time-between-events
- Assumes regularity
- Median more robust against outliers
# API documentation: https://docs.brewfather.app/integrations/custom-stream

```python
import requests

url = 'http://log.brewfather.net/stream?id=9MmXXXXXXXXX'
data = dict(name='brewaed-0001', bpm=CALCULATED-BPM)
r = requests.post(url, json=data)
```
OUTRO
MORE RESOURCES

Github project: jonnor/brewing-audio-event-detection

General Audio ML: jonnor/machinehearing

- Sound Event Detection: A tutorial. Virtanen et al.
- Audio Classification with Machine Learning (EuroPython 2019)
- Environmental Noise Classification on Microcontrollers (TinyML 2021)

Slack: Sound of AI community
WHAT DO YOU WANT MAKE?

Now that you know the basics of Audio Event Detection with Machine Learning in Python.

- Popcorn popping
- Bird call
- Cough
- Umm/aaa speech patterns
- Drum hits
- Car passing
CONTINUOUS MONITORING USING AUDIO ML

Want to deploy Continious Monitoring with Audio? Consider using the Soundsensing sensors and data-platform.

1. Sensor

2. Dashboard + API

Get in Touch! contact@soundsensing.no
JOIN SOUNDENSING

Want to work on Audio Machine Learning in Python? We have many opportunities.

- Full-time positions
- Part-time / freelance work
- Engineering thesis
- Internships
- Research or industry partnerships

Get in Touch! contact@soundsensing.no
QUESTIONS?

Sound Event Detection with Machine Learning
EuroPython 2021

Jon Nordby
jon@soundsensing.no
Head of Data Science & Machine Learning
BONUS

Bonus slides after this point
SEMI-AUTOMATIC LABELLING

Using a Gaussian Mixture, Hidden Markov Model (GMM-HMM)

import hmmlearn.hmm, librosa, sklearn.preprocessing

features = librosa.feature.mfcc(audio, n_mfcc=13, ...)
model = hmmlearn.hmm.GMMHMM(n_components=2, ...)
X = sklearn.preprocessing.StandardScaler().fit_transform(data)
model.fit(X)
probabilities = model.score_samples(X)[1][::, 1]
SYNTHESIZE DATA

How to get more data without gathering “in the wild”? 

- Mix in different kinds of background noise. 
- Vary Signal to Noise ratio etc 
- Useful to estimate performance on tricky, not-yet-seen data 
- Can be used to compensate for small amount of training data 
- *scaper* Python library: [github.com/justinsalamon/scaper](github.com/justinsalamon/scaper)
import sounddevice, queue

# Setup audio stream from microphone
audio_queue = queue.Queue()

def audio_callback(indata, frames, time, status):
    audio_queue.put(indata.copy())

stream = sounddevice.InputStream(callback=audio_callback, ...)
...

# In classification loop
data = audio_queue.get()

# shift old audio over, add new data
audio_buffer = numpy.roll(audio_buffer, len(data), axis=0)
audio_buffer[len(audio_buffer)-len(data):len(audio_buffer)] = data
new_samples += len(data)

# check if we have received enough new data to do new prediction
if new_samples >= hop_length:
    p = model.predict(audio_buffer)
    if p < threshold:
        print(f'EVENT DETECTED time={datetime.datetime.now()}')
EVENT DETECTION WITH WEAKLY LABELED DATA

Can one learn Sound Event Detection without annotating the times for each event?

Yes!

- Referred to as *weekly labeled* Sound Event Detection
- Can be tackled with *Multiple Instance Learning*
- Inputs: Audio clips consisting of 0-N events
- Labels: True if any events in clip, else false
- Multiple analysis windows per 1 label
- Using temporal pooling in Neural Network
DATA COLLECTION VIA YOUTUBE

Criteria for inclusion:

- Preferably couple of minutes long, minimum 15 seconds
- No talking to the camera
- Mostly stationary camera
- No audio editing/effects
- One or more airlocks bubbling
- Bubbling can be heard by ear

Approx 1000 videos reviewed, 100 usable
CHARACTERISTICS OF AUDIO EVENTS

- Duration
- Tonal/atonal
- Temporal patterns
- Percussive
- Frequency content
- Temporal envelope
- Foreground vs background
- Signal to Noise Ratio
ANALYSIS WINDOWS

Window length bit longer than the event length.
Overlapping gives classifier multiple chances at seeing each event.
Reducing overlap increases resolution! Overlap for AES: 10%