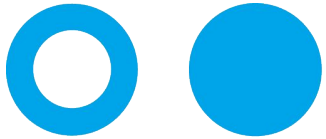


Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

ENVIRONMENTAL SOUND CLASSIFICATION USING DEEP LEARNING

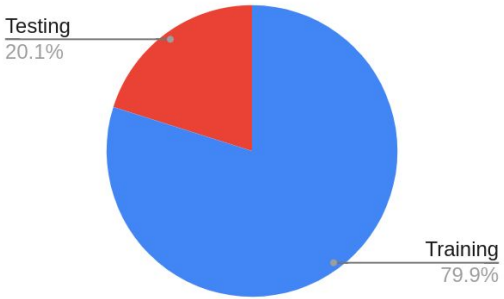
Problem formulation

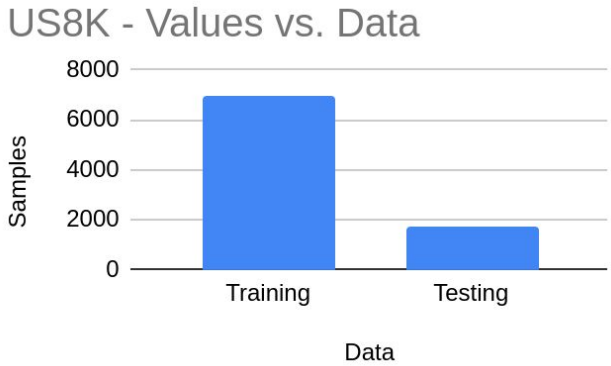
- Intelligent sound detection(ISR) is a technology for identifying sound events that exist in the real environment and embedding such ability in machines or robots.
- Environmental sound classification(ESC) serve as a fundamental steps of ISR.
- The main goal of ESC is to precisely classify the class of a detected sound.
- Automatic speech recognition (ASR) and music information recognition (MIR) will be inefficient when applying to non stationary ESC task because environmental sound is non stationary in nature.
- Therefore, it is essential to develop an efficient ISR system for environment sound(ES) recognition.
- The main obstacles of current algorithms for ESC task are
- 1) log-mel and MFCC are the most widely used acoustic features applied to ESC, but were originally designed for ASR and MIR. ES is non stationary signal and using single features lead to the failure of capturing the important information about Environmental sound.
- 2) In recent years neural network has gained popularity for environmental sound classification. However performance is still unsatisfactory.
- Hence, there is a need to develop appropriate auditory features and novel neural network models to achieve high categorization accuracy for ESC tasks.

Dataset

[1] Urbansound8k [3]:

- This dataset contains 8732 labeled sound of ~4 sec length in WAV format at 22.05 kHz.
- 10 classes were defined: air_conditioner, car_horn, children_playing, dog_bark, drilling, enginge_idling, gun_shot, jackhammer, siren, and street_music.
- Nearly 870 slices of data per class.



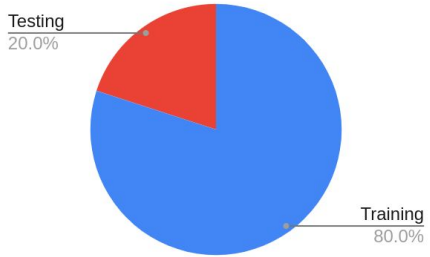


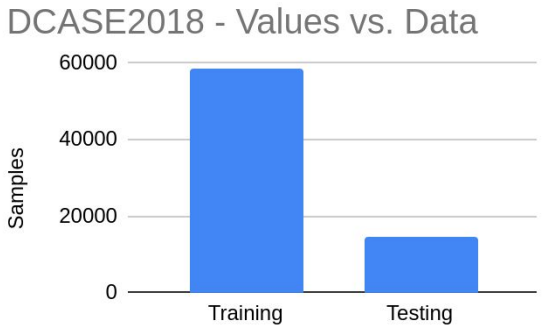
Dataset

[3] DCASE 2018, Monitoring of domestic activities based on multi-channel acoustics

- Total of 72984, 10 sec audio dataset.
- The continuous recordings in home environment were split into audio segments of 10s.
- 9 classes were considered,

Activity	# 10s segments
Absence (nobody present in the room)	18860
Cooking	5124
Dishwashing	1424
Eating	2308
Other (present but not doing any relevant activity)	2060
Social activity (visit, phone call)	4944
Vacuum cleaning	972
Watching TV	18648
Working (typing, mouse click, ...)	18644
Total	**72984**





Acoustic feature

Courtesy:Environment Sound Classification Using a Two-Stream CNN Based on Decision-Level Fusion
Yu Su 1,2,* , Ke Zhang 1, Jingyu Wang 1 and Kurosh Madani 2

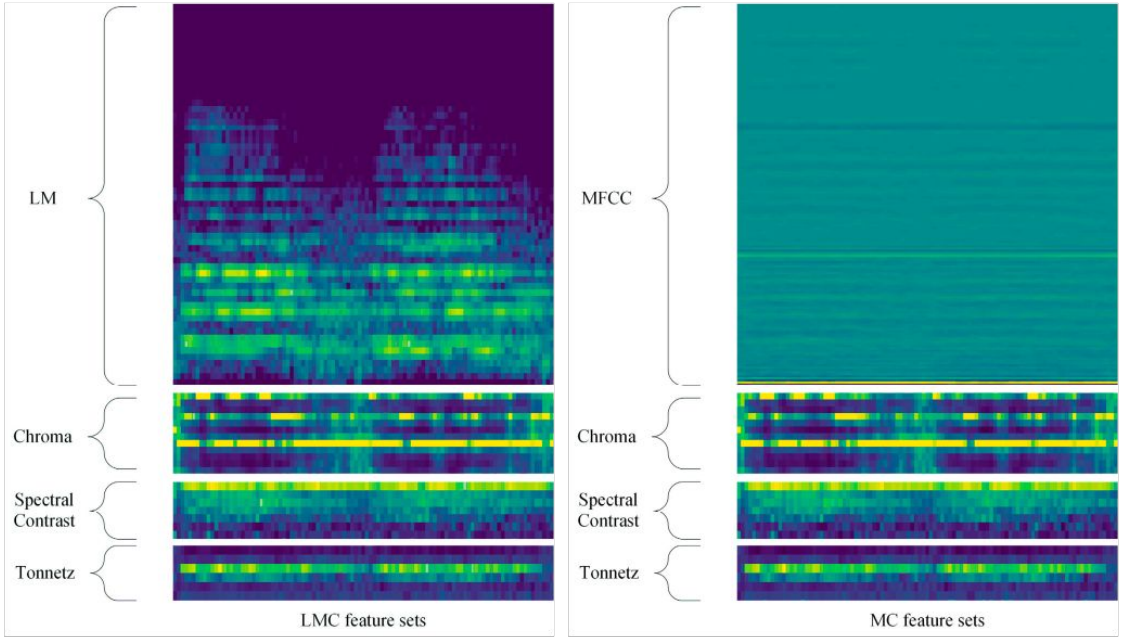


Figure 3: The spectrogram of LMC and MC feature sets

Courtesy:Environment Sound Classification Using a Two-Stream CNN Based on Decision-Level Fusion. Yu Su 1,2,* , Ke Zhang 1, Jingyu Wang 1 and Kurosh Madani 2

Table2. Comparison of classification accuracy with other models on UrbanSound8K datasets.

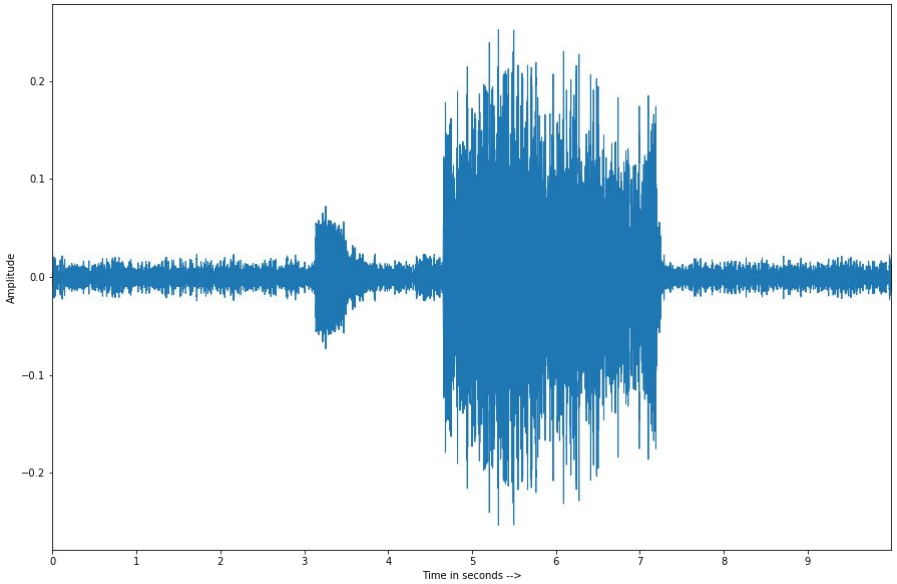
Model	Feature	Accuracy
Piczak [28]	LM	72.7%
Tokozume [35]	Raw Data	78.3%
Zhang X. [32]	Mel	81.9%
Zhang Z. [6]	LM-GS	83.7%
Li [20].	Raw Data-LM	92.2%
Boddapati [25]	Spec -MFCC-CRP	93%
LMCNet	LM-C	95.2%
MCNet	M-C	95.3%
TSCNN-DS	**MC and LMC**	**97.2%**

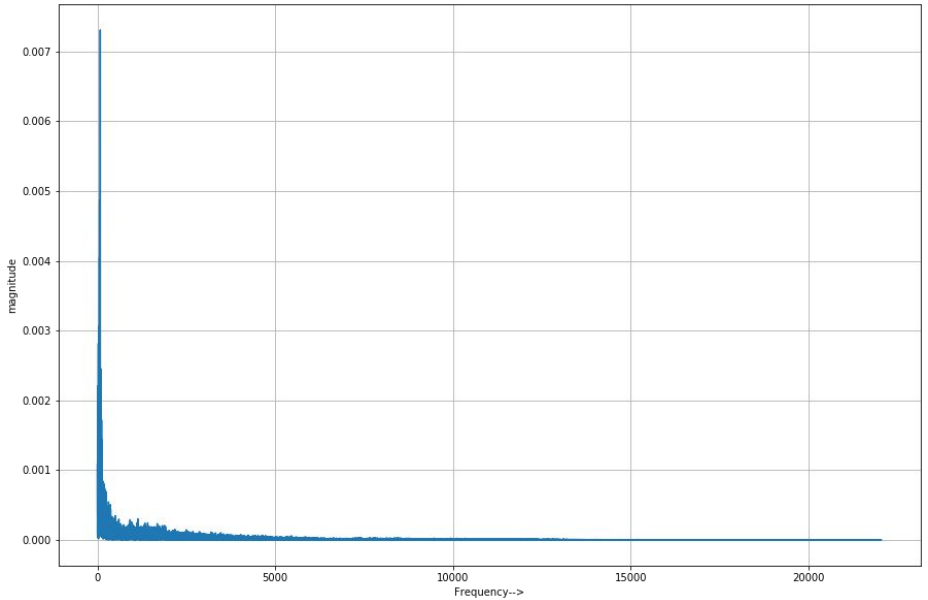
Courtesy:Environment Sound Classification Using a Two-Stream CNN Based on Decision-Level Fusion. Yu Su 1,2,* , Ke Zhang 1, Jingyu Wang 1 and Kurosh Madani 2

Table 1. Class-wise accuracy of four models with four-layer CNN evaluated on UrbanSound8K

Class	LMC (LMCNet)	MC (MCNet)	MLMC	TSCNN-DS
ac	98.6%	99.9%	99.2%	99.9%
ch	93.9%	91.4%	93.2%	94.2%
cp	97.3%	93.9%	96.1%	97.5%
db	92.6%	90.4%	94.2%	95.3%
dr	94.8%	95.0%	95.7%	97.2%
ei	98.9%	99.6%	98.5%	99.6%
gs	88.6%	91.1%	85.9%	95.4%
jh	93.2%	95.9%	91.1%	97.1%
si	98.6%	98.3%	98.5%	98.9%
sm	95.0%	97.4%	94.1%	96.9%
Avg.	95.2%	95.3%	94.6%	97.2%

Sound





Acoustic features

- MFCC
- Mel spectrogram

High-level

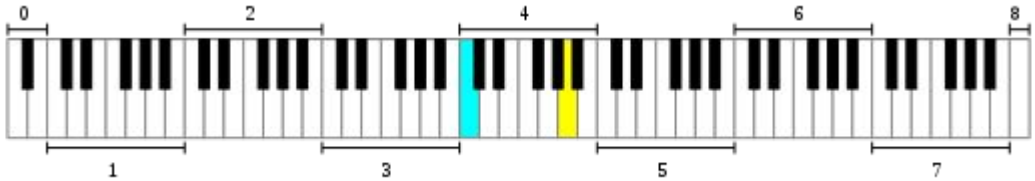
Examples: instrumentation, key, chords, melody, rhythm, tempo, lyrics, genre, mood

Mid-level

Examples: pitch- and beat-related descriptors, such as note onsets, fluctuation patterns, MFCCs

Low-level

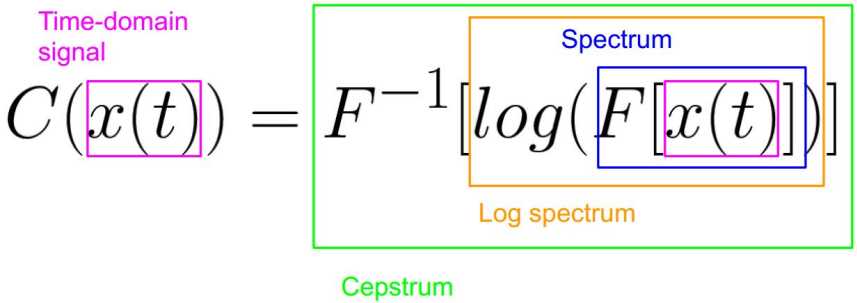
Examples: amplitude envelope, energy, spectral centroid, spectral flux, zero-crossing rate



Knees, P., & Schedl, M. (2016). *Music similarity and retrieval: an introduction to audio-and web-based strategies*

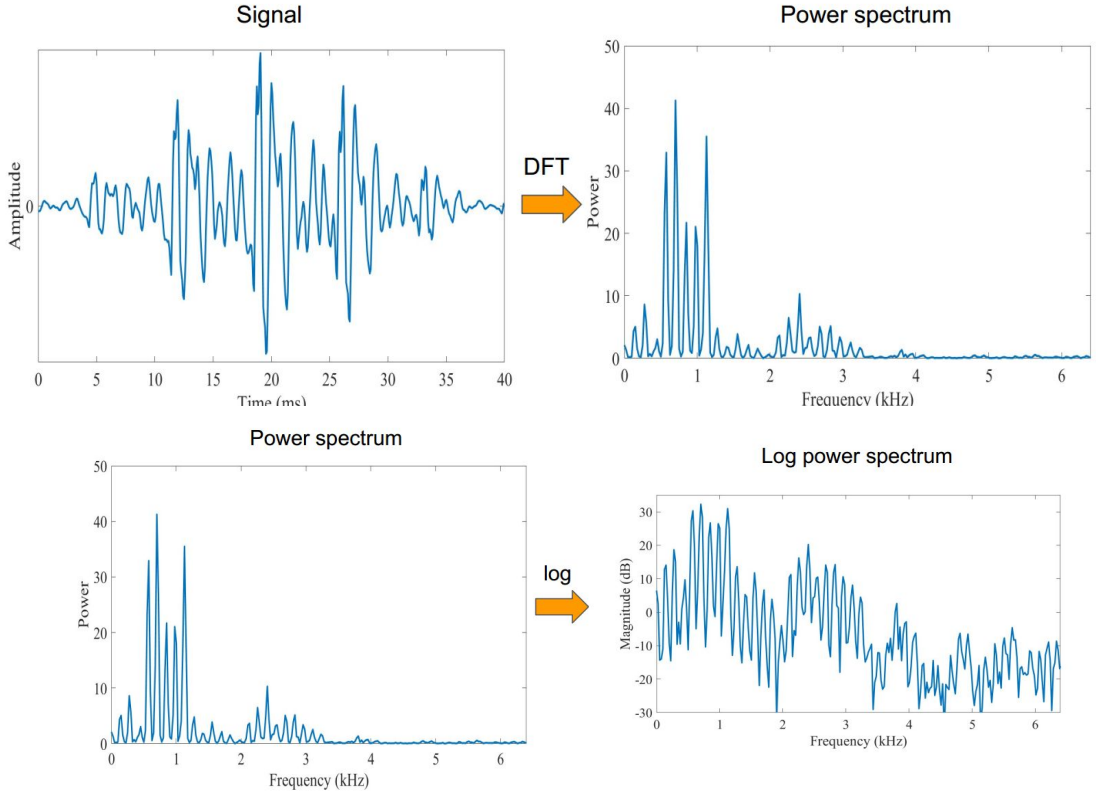
Acoustic features: MFCC

- Mel-Frequency Cepstral Coefficients. [5]



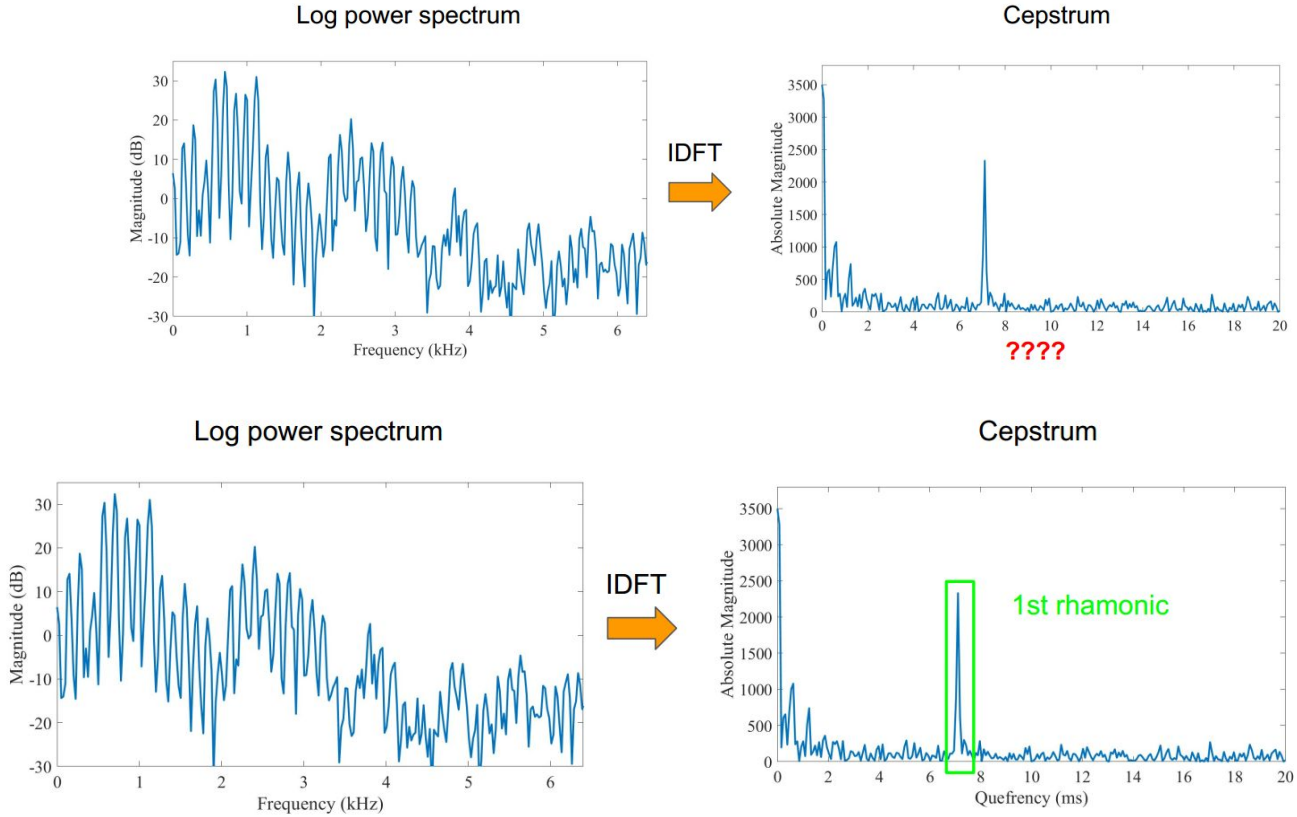
Acoustic features: MFCC

[5]



Acoustic features: MFCC

[5]



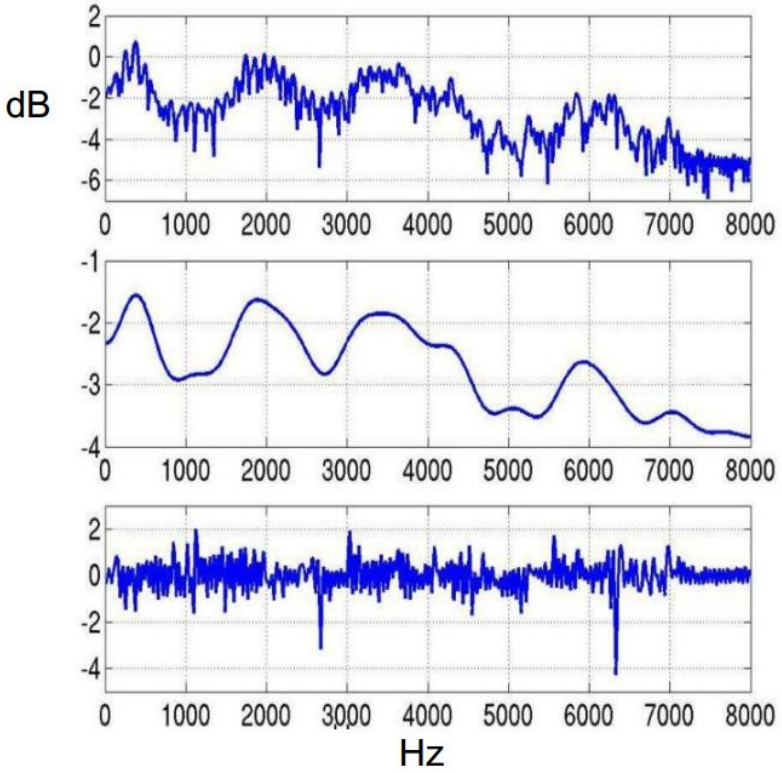
Acoustic features: MFCC

[5]

Log-spectrum

Spectral envelope

Spectral detail



Speech

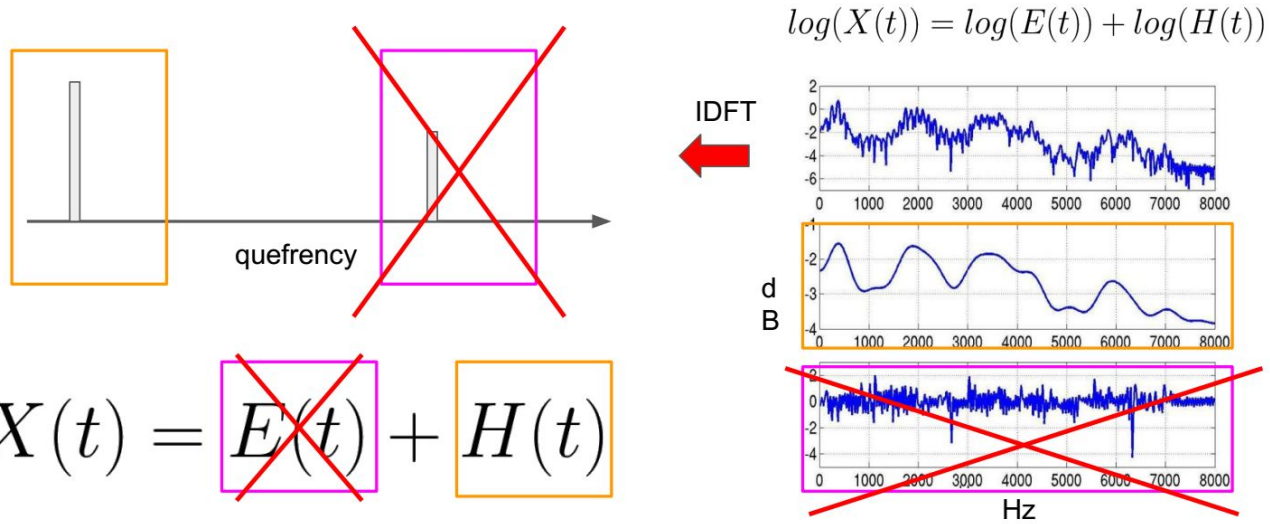
Vocal tract frequency response

Glottal pulse

Acoustic features: MFCC

[5]

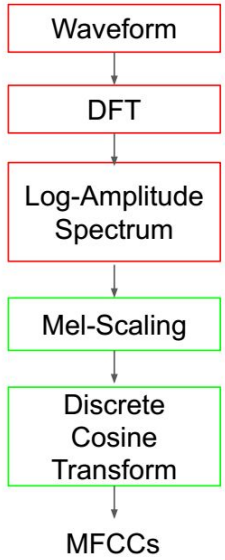
Separating components

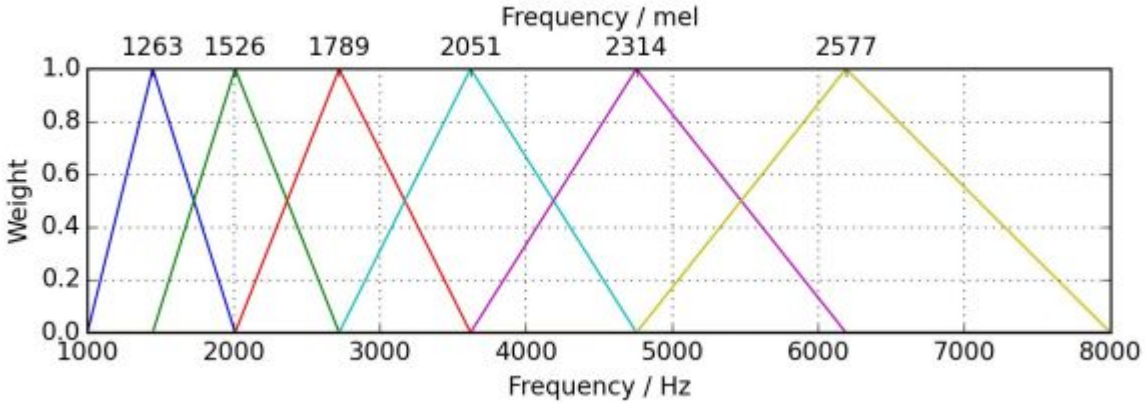


$$X(t) = E(t) + H(t)$$

Acoustic features: MFCC

[5]

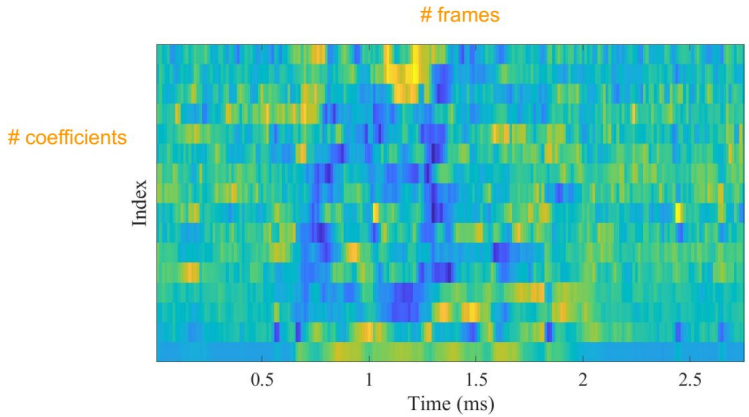




Acoustic features: MFCC

[5]

- Traditionally: first 12 - 13 coefficients
- First coefficients keep most information (e.g., formants, spectral envelope)
- Use Δ and ΔΔ MFCCs
- Total 39 coefficients per frame

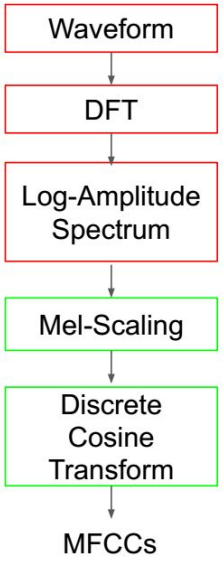


Approaches

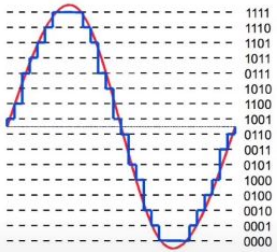
- Classical machine learning
  - KNN
  - SVM
  - Naive bayes
  - Random forest
  - Gradient boosting
  - XG boost
- Deep learning
  - Transfer learning
    - VGG model - Freezing each block and training
      - Random samples from data
      - Equal splitting of data

Data preparation for classical approach

- Compute the 20 mfcc features for each audio file.



Classical approach pipeline



frame 1: sample 1 … 128
frame 2: sample 64 … 192
frame 3: sample 128 … 256
frame 4: sample 192 … 320
...

DFT

Log-Amplitude Spectrum

Mel-Scaling

Discrete Cosine Transform

MFCCs

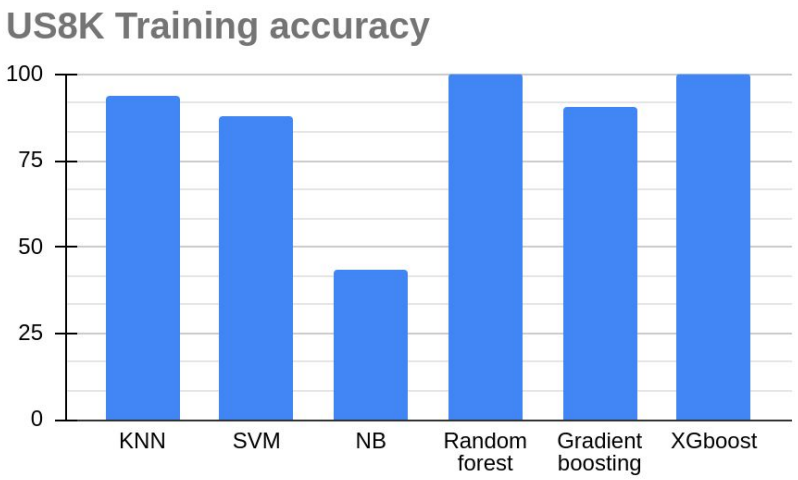
Classical model with default parameters

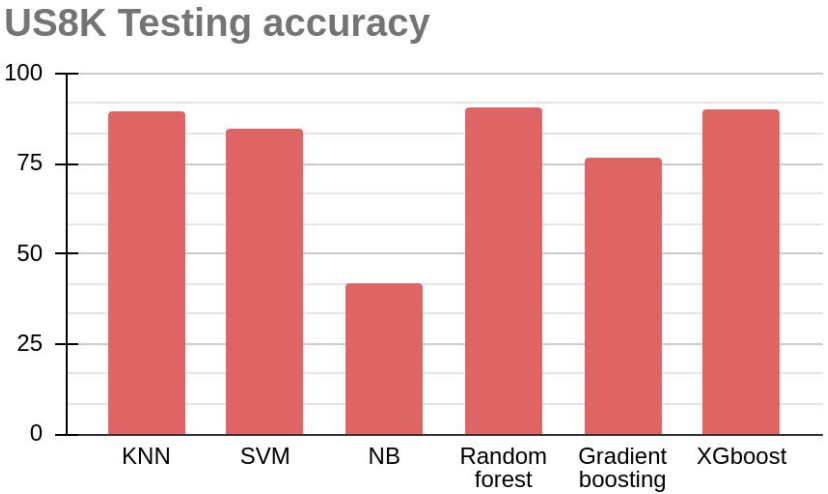
Classical model with hyperparameter tuning

Comparison of results

Classical approach - Urbansound8k(US8K)

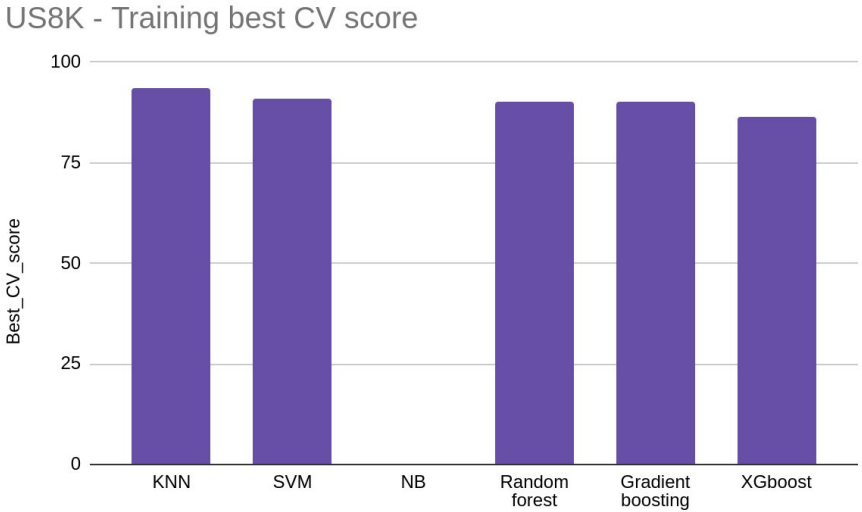
- Default parameter

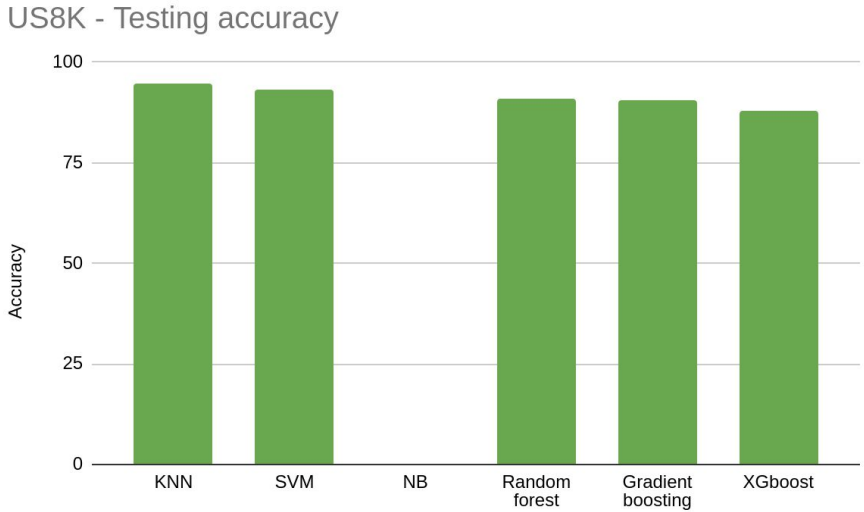




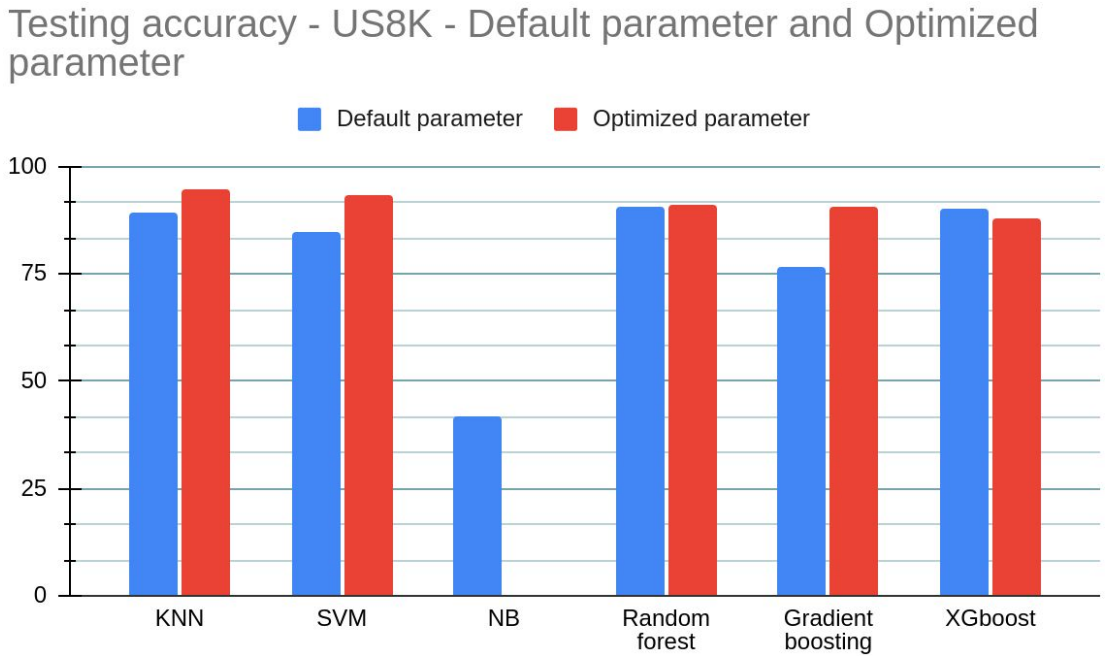
Classical approach - Urbansound8k(US8K)

- Optimized parameter



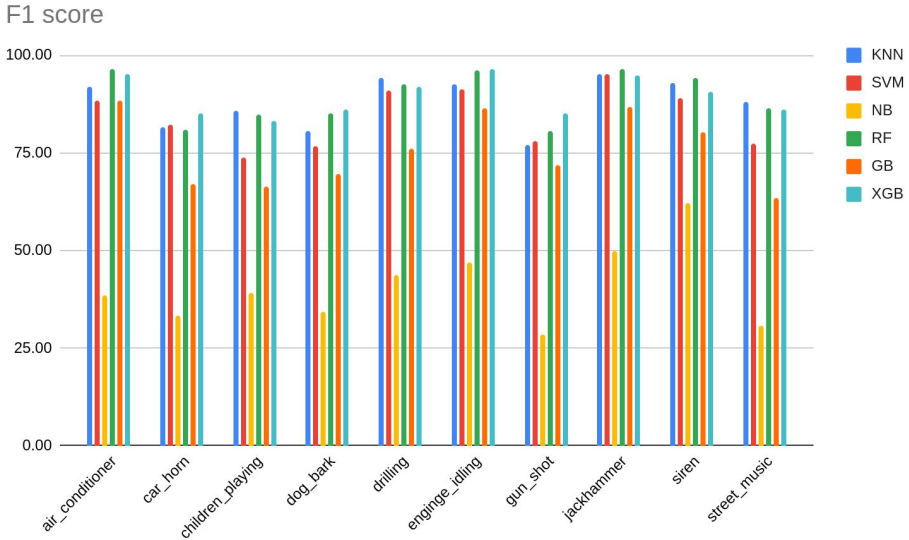


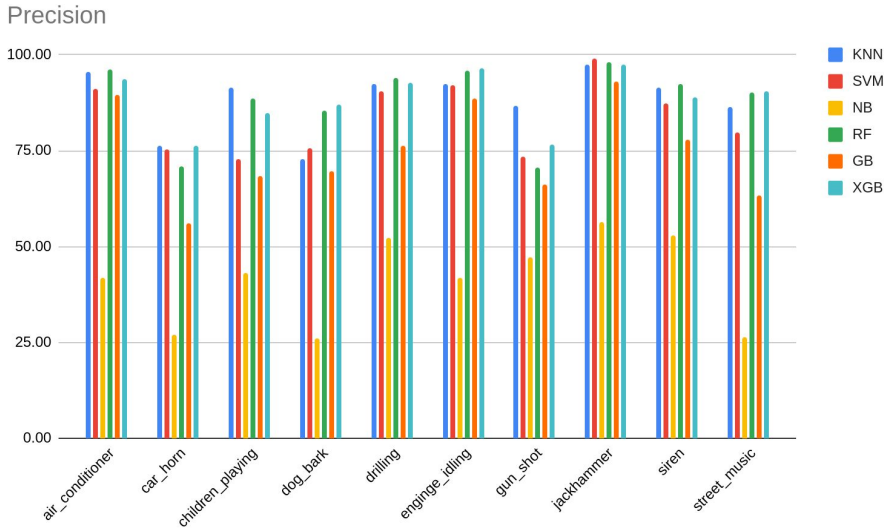
Classical approach - Urbansound8k(US8K)



Classical approach - Urbansound8K(US8K)

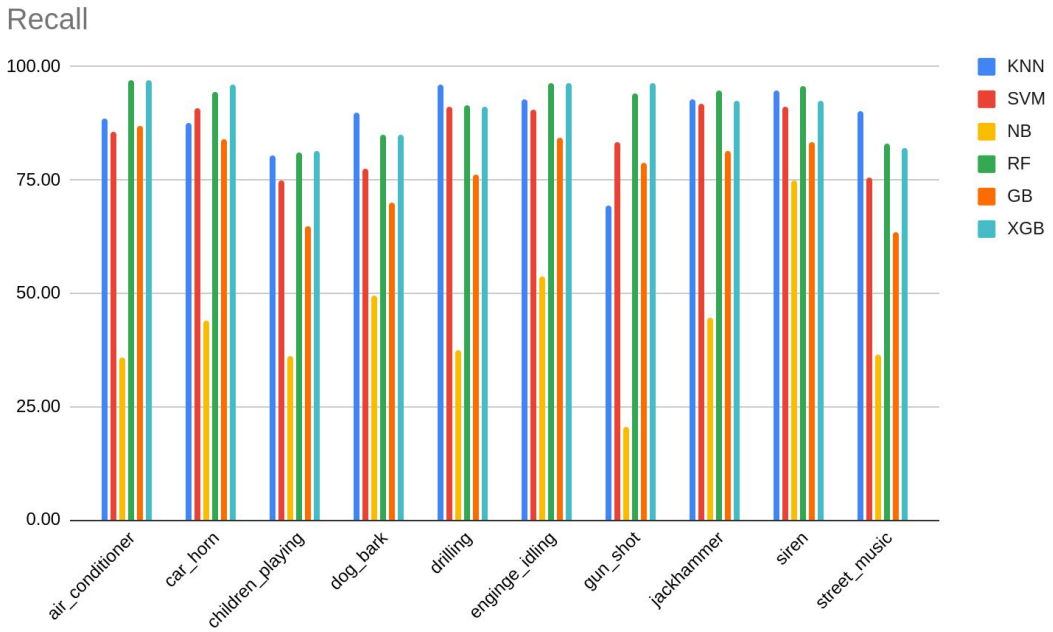
- Default parameter





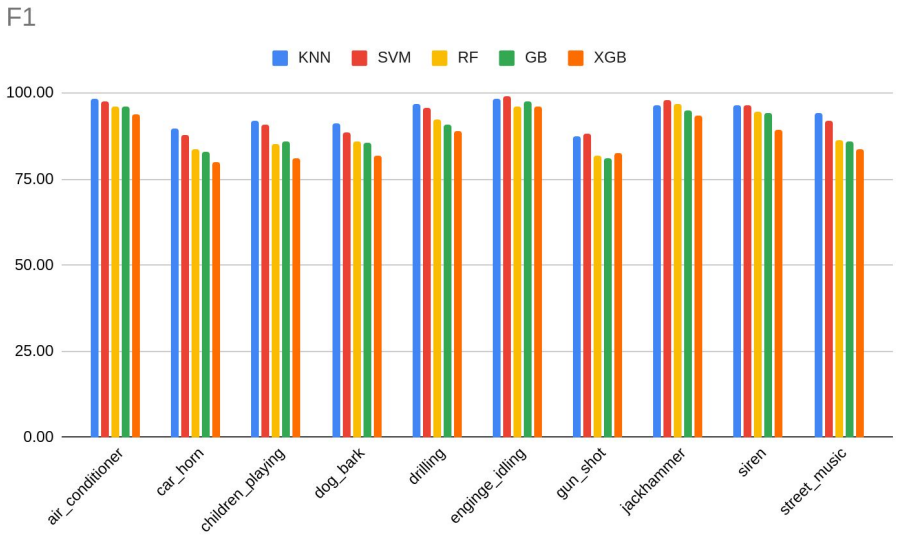
Classical approach - Urbansound8K(US8K)

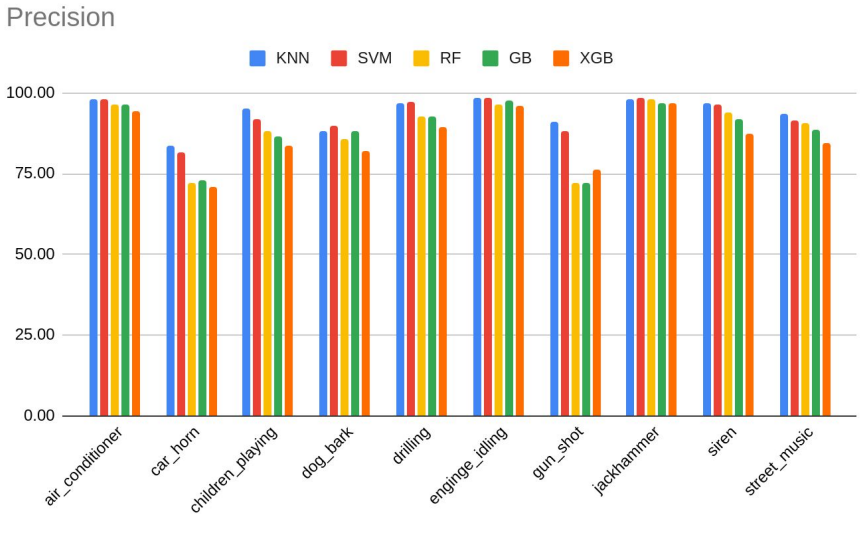
- Default parameter



Classical approach - Urbansound8K(US8K)

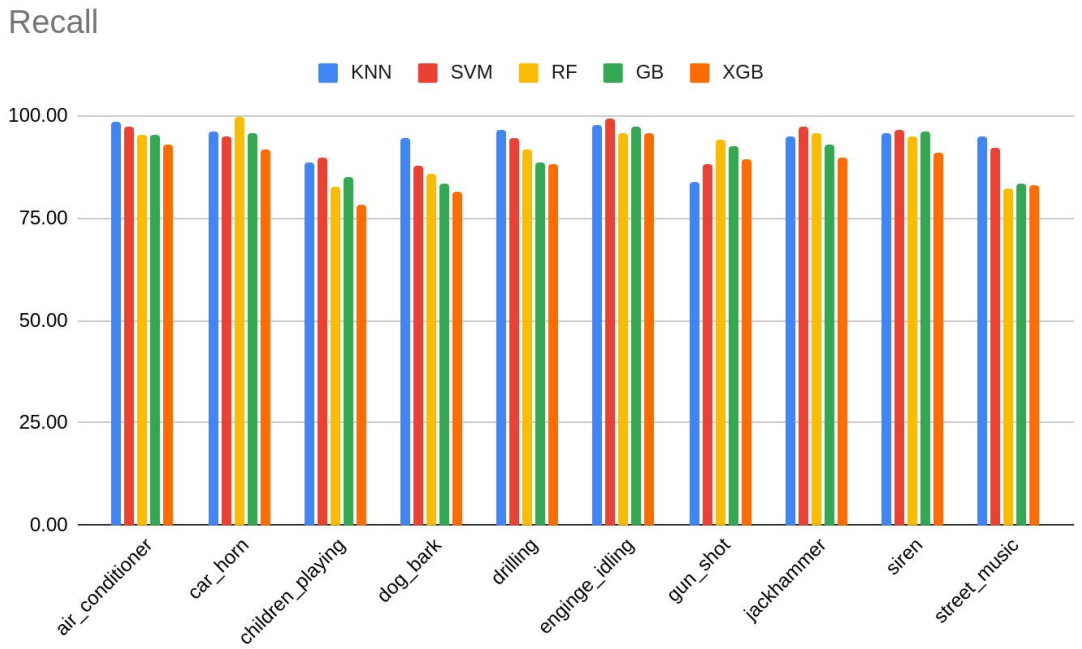
- Optimized parameter



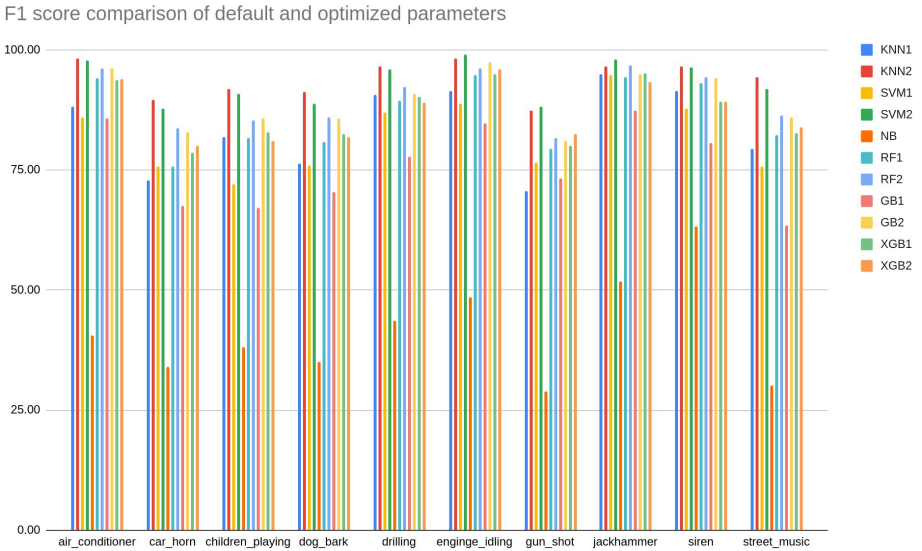


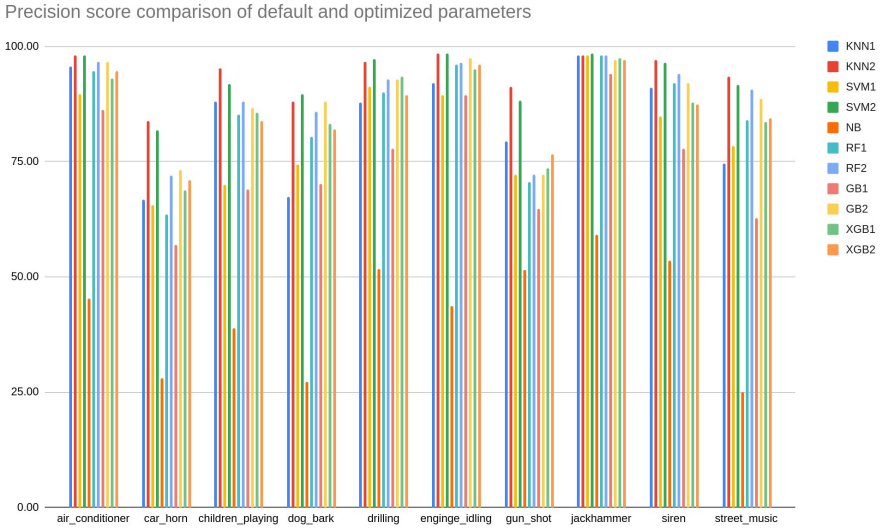
Classical approach - Urbansound8K(US8K)

- Optimized parameter

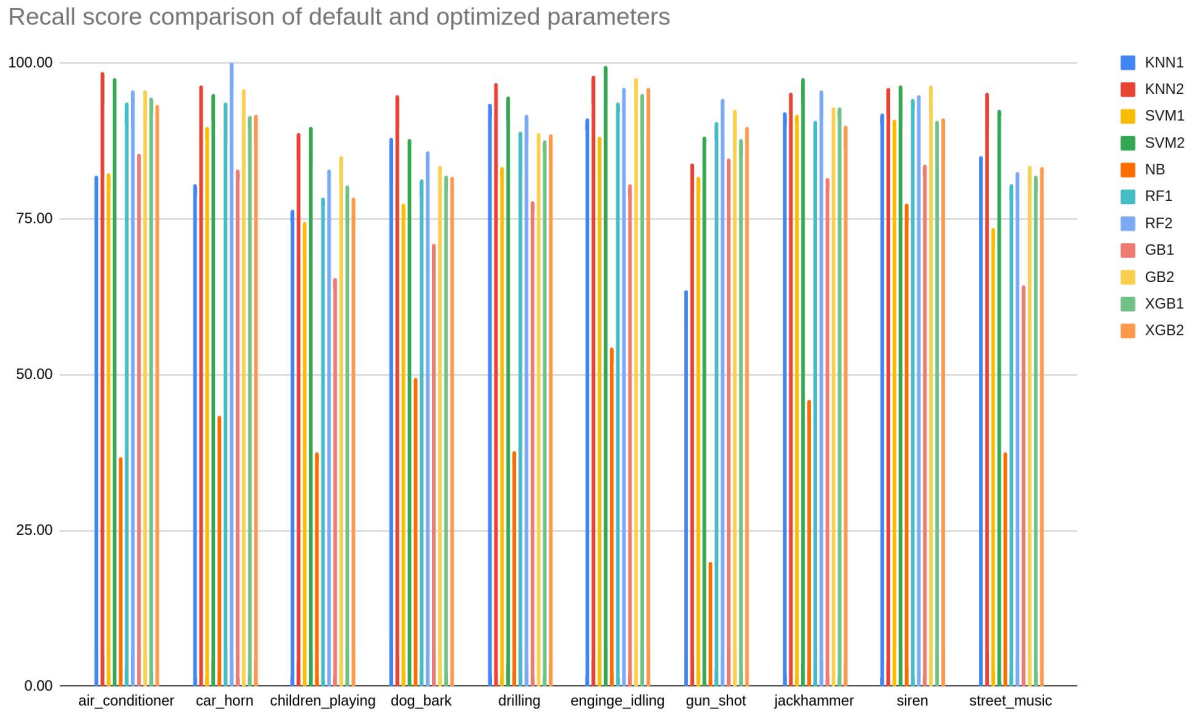


Classical approach - Urbansound8K(US8K)



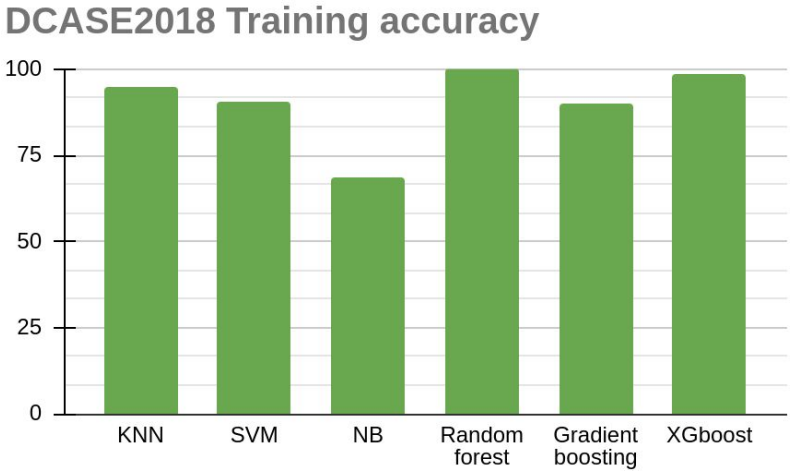


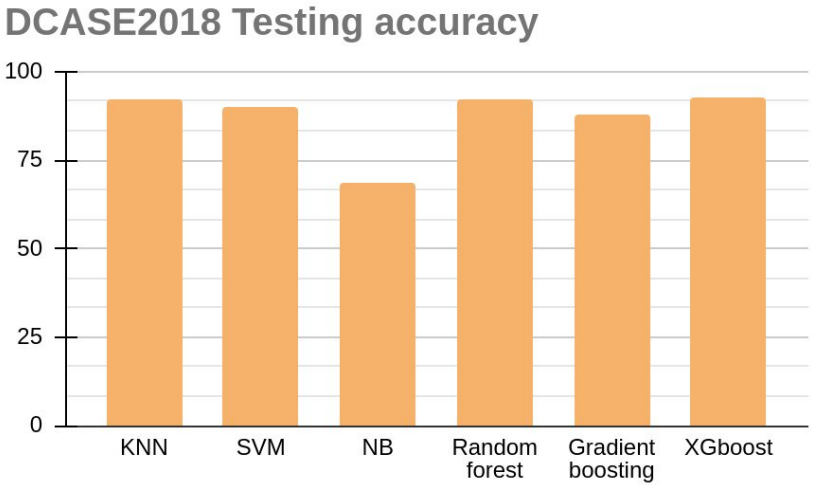
Classical approach - Urbansound8K(US8K)



Classical approach - DCASE2018

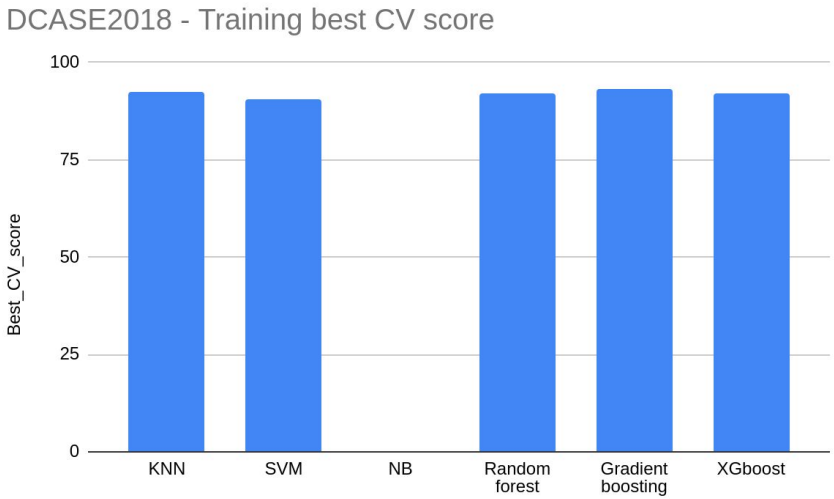
- Default parameter

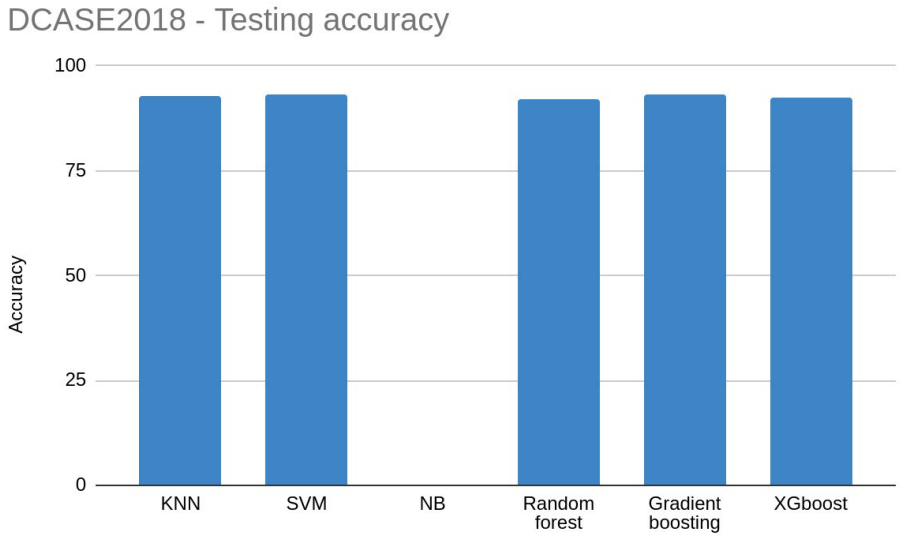




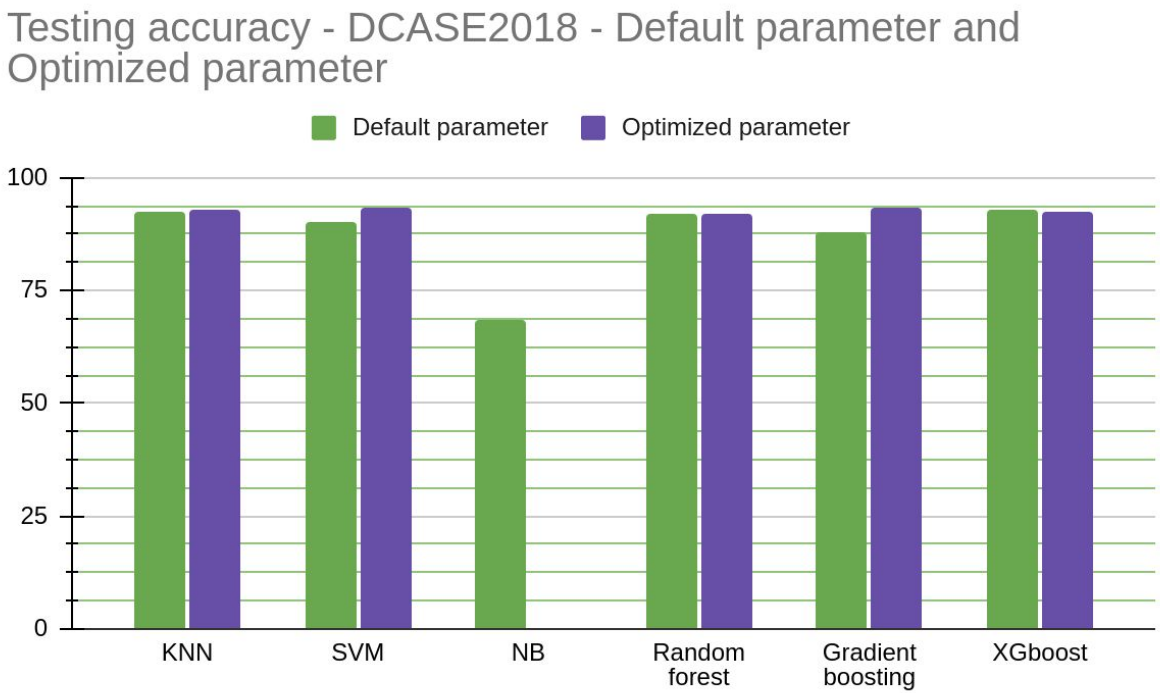
Classical approach - DCASE2018

- Optimized parameter



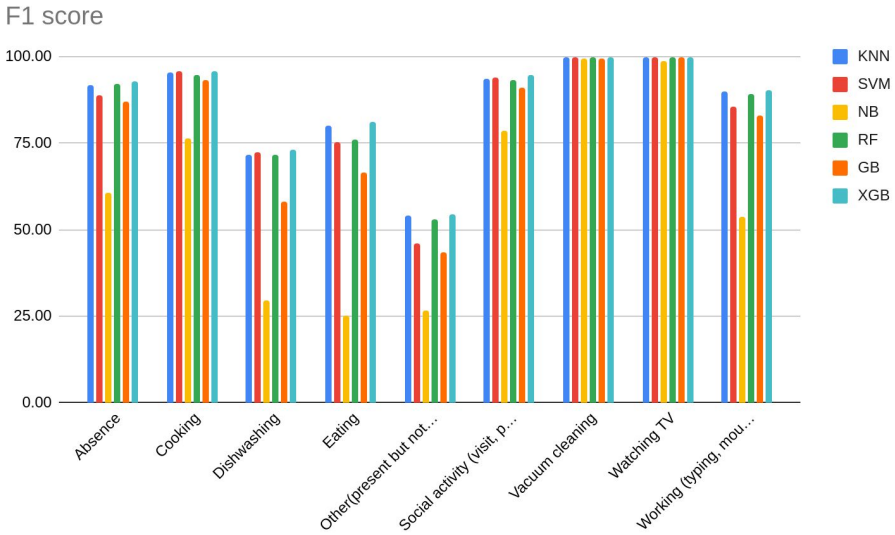


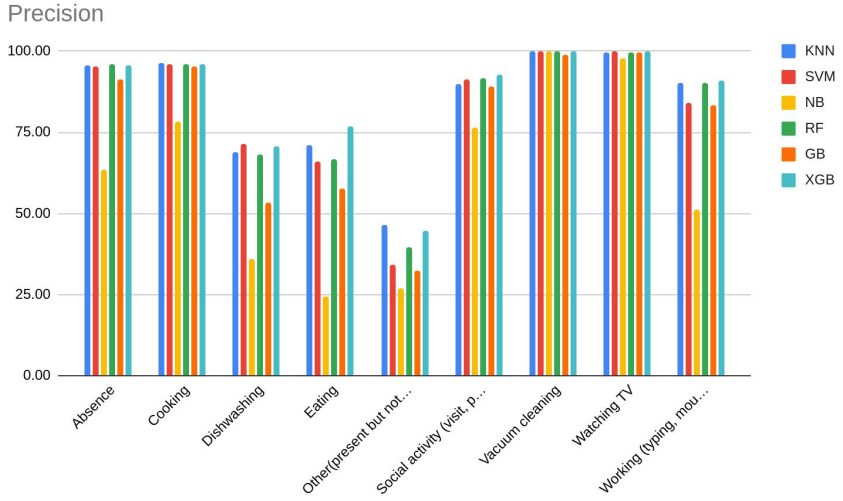
Classical approach - DCASE2018



Classical approach - DCASE2018

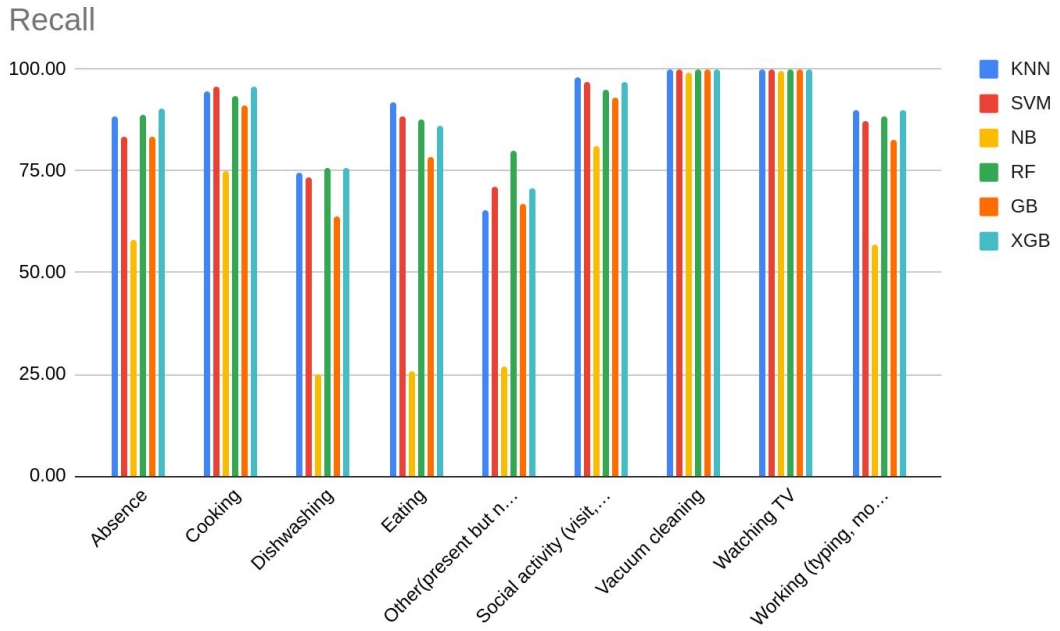
- Default parameter





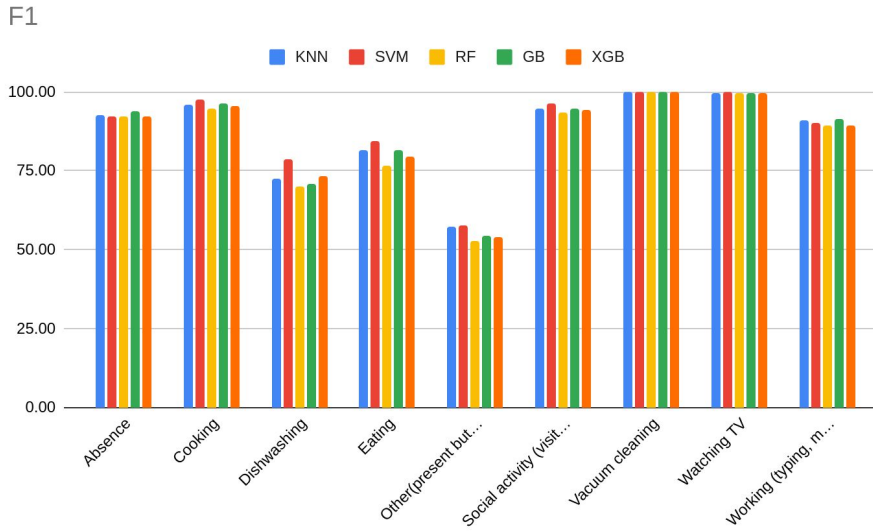
Classical approach - DCASE2018

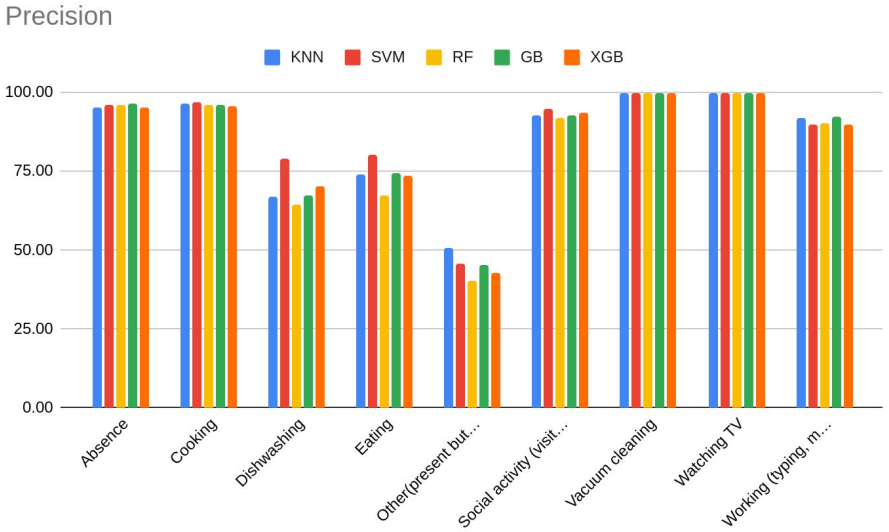
- Default parameter



Classical approach - DCASE2018

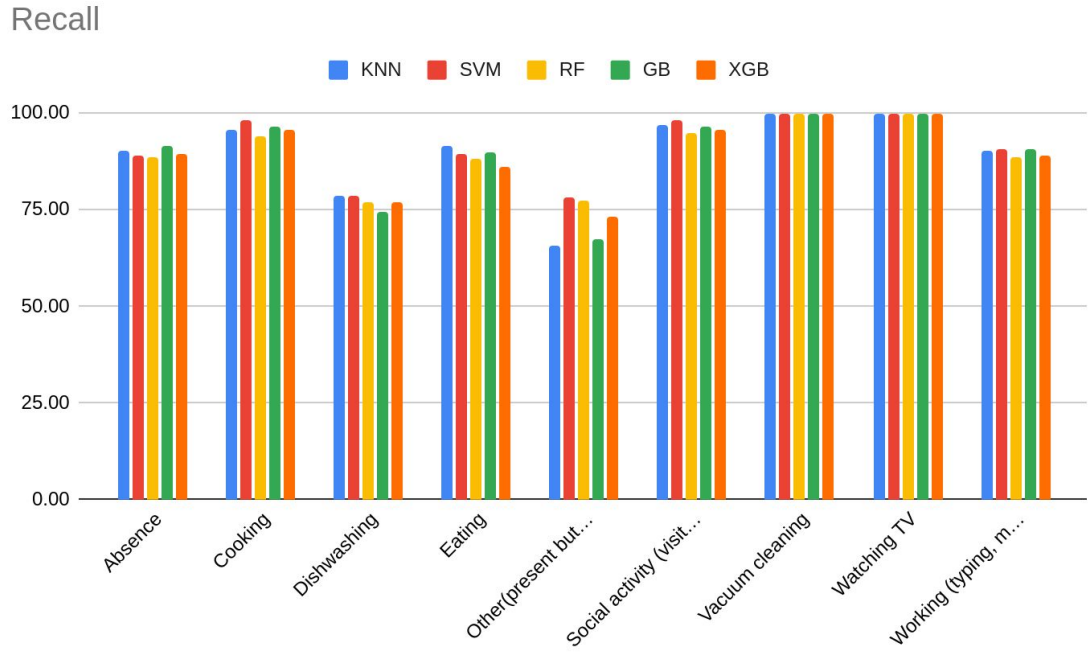
- Optimized parameter



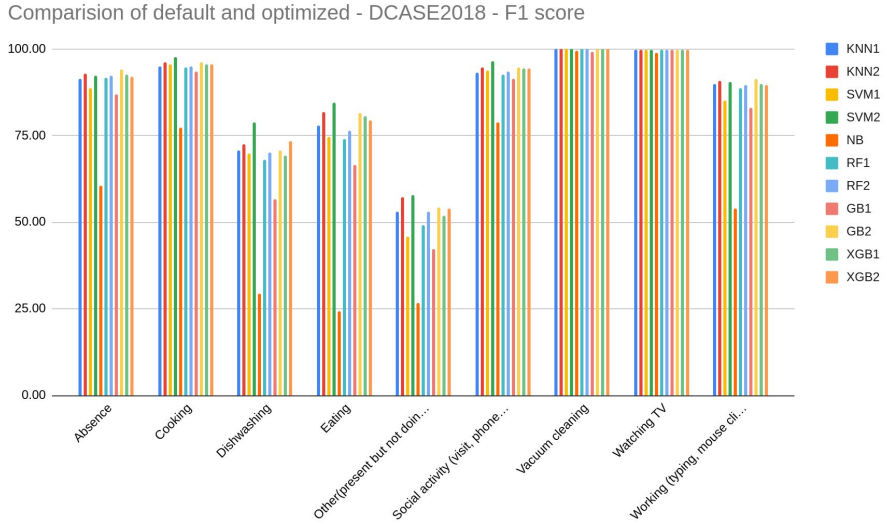


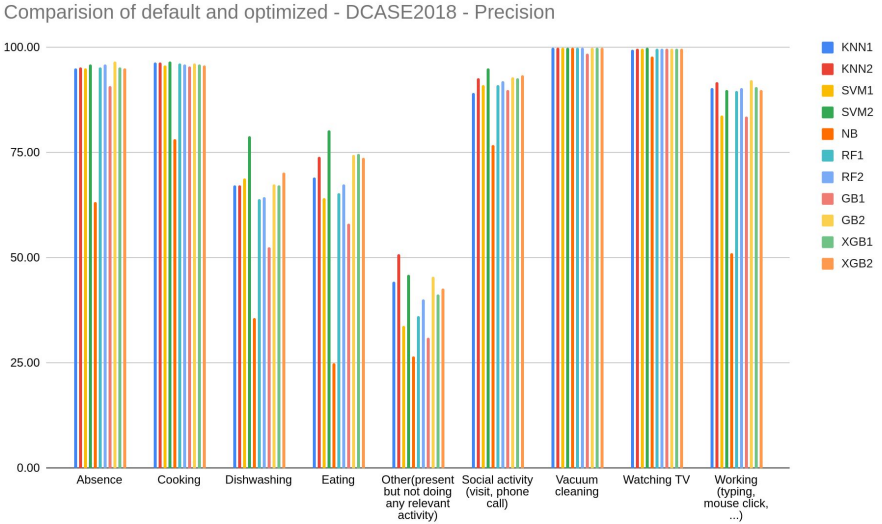
Classical approach - DCASE2018

- Optimized parameter

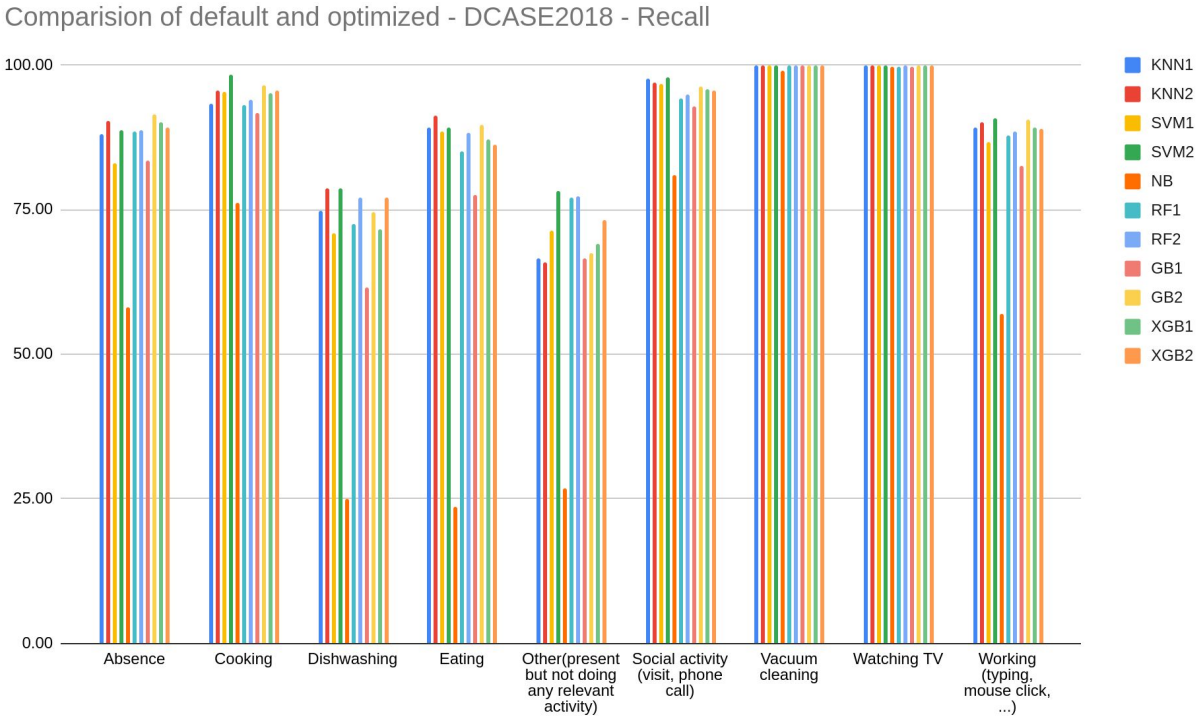


Classical approach - DCASE2018

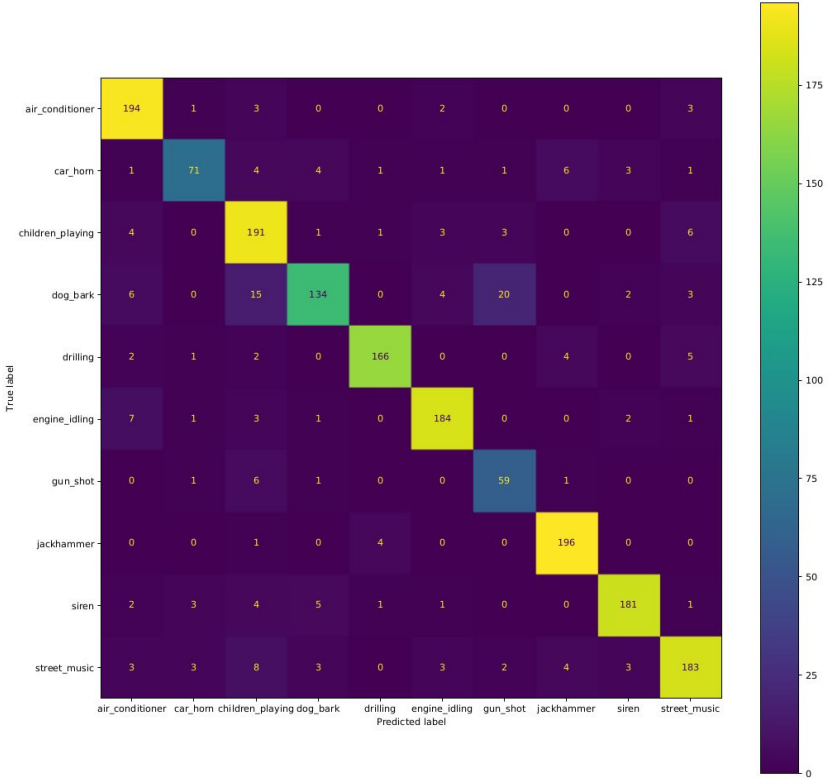


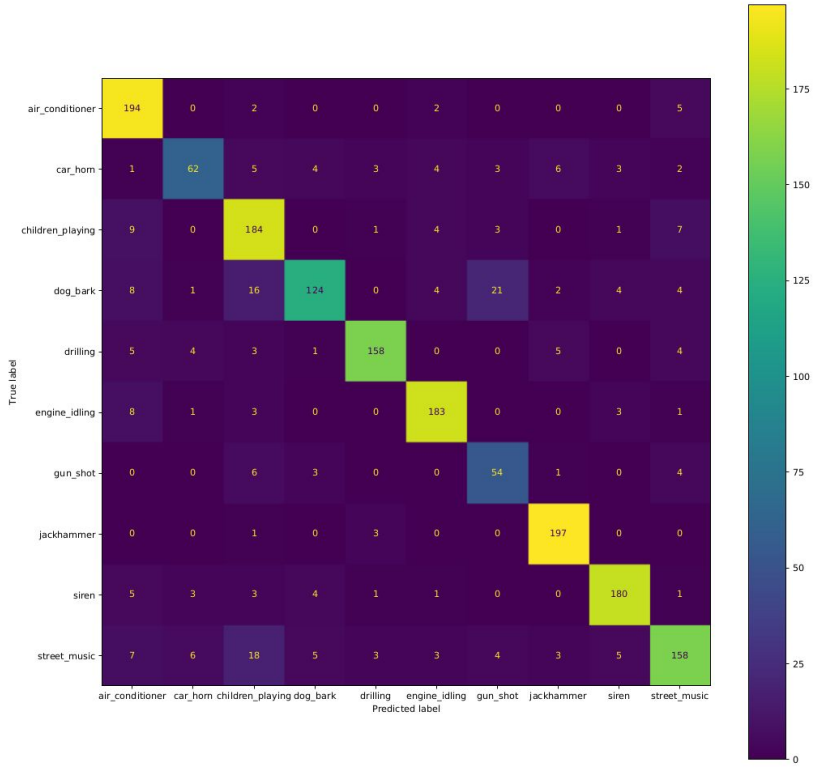


Classical approach - DCASE2018

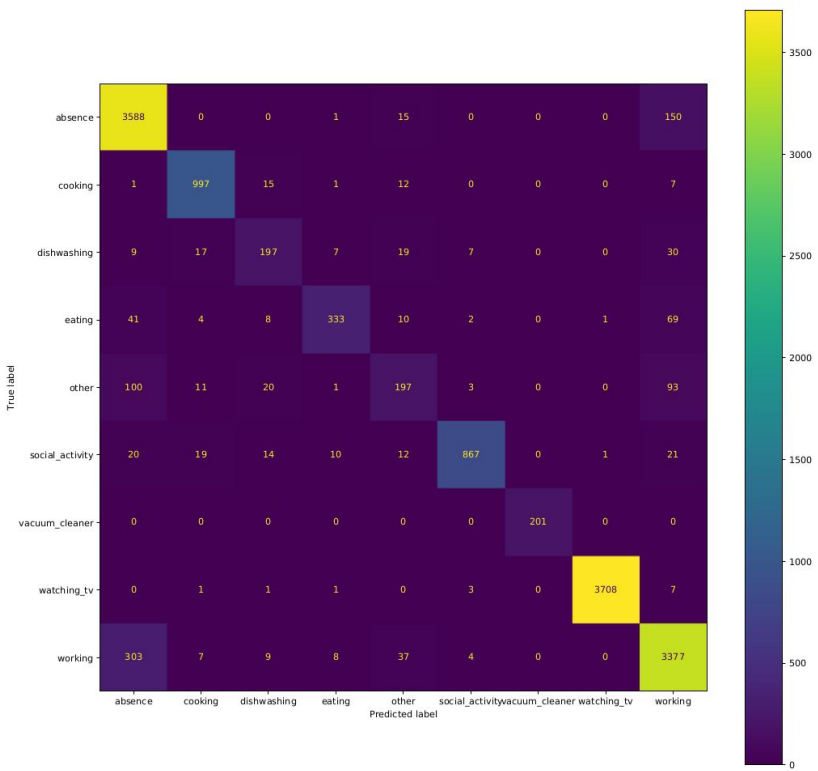


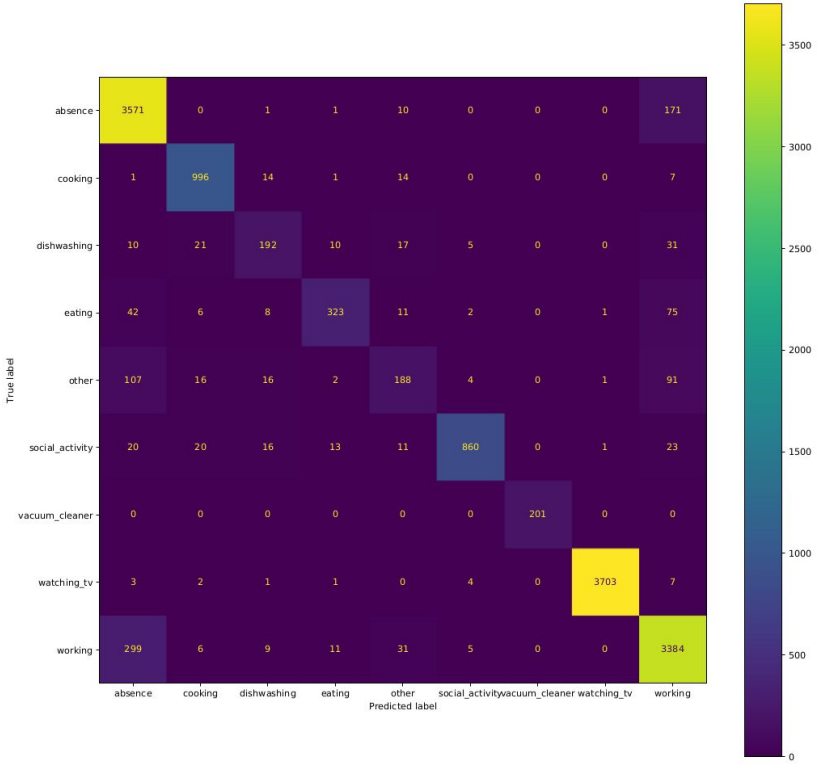
KNN - Urbansound8k



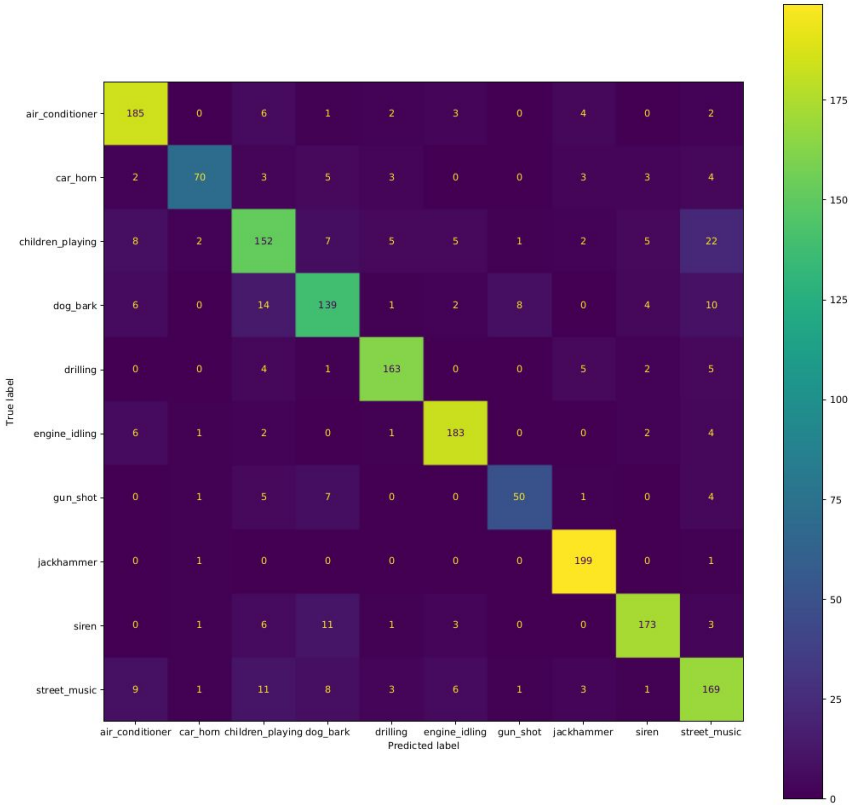


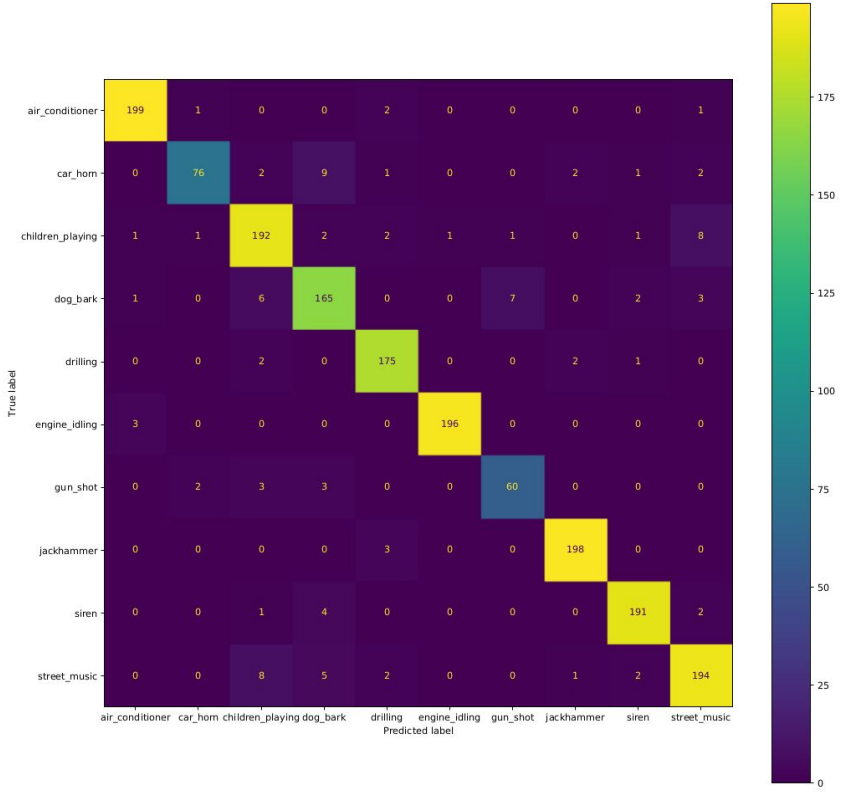
KNN - Dcase 2018



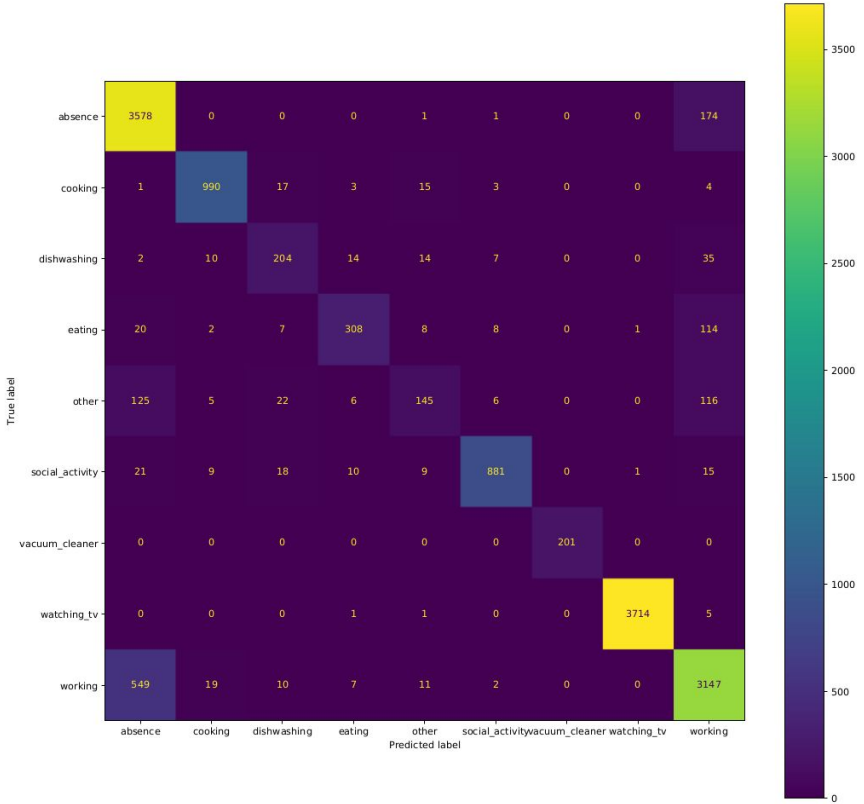


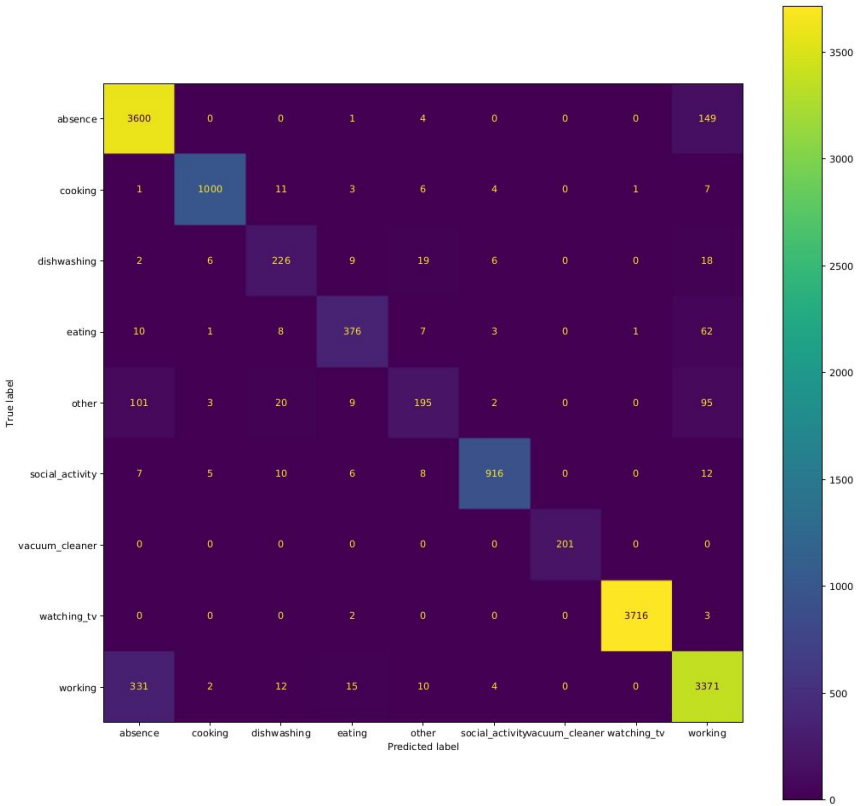
SVM - Urbansound8K





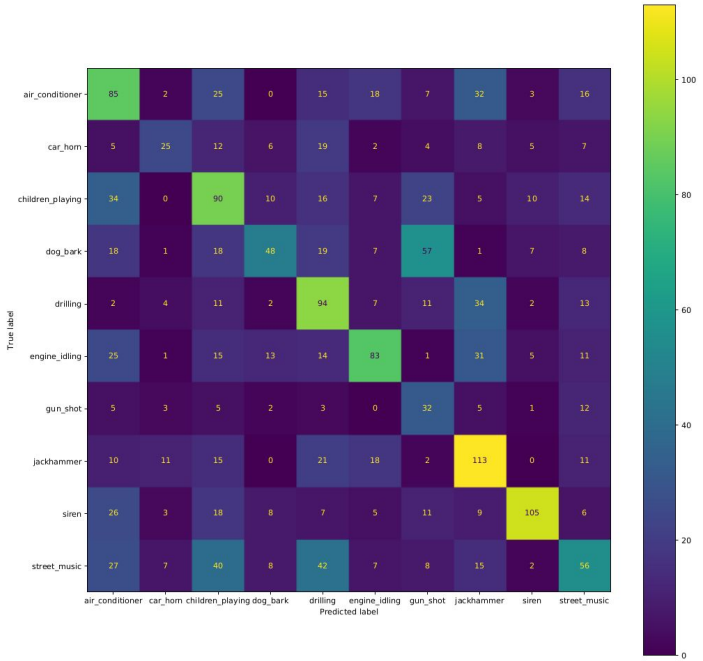
SVM - DCASE -2018

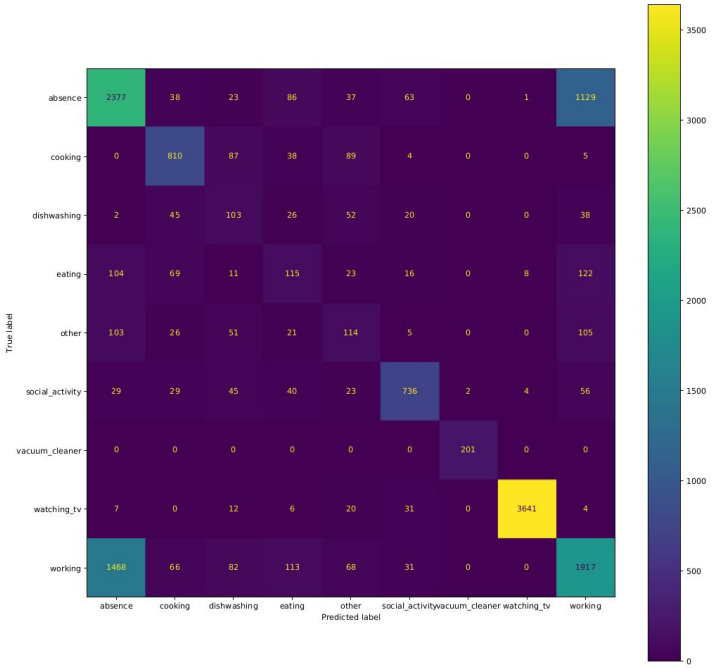




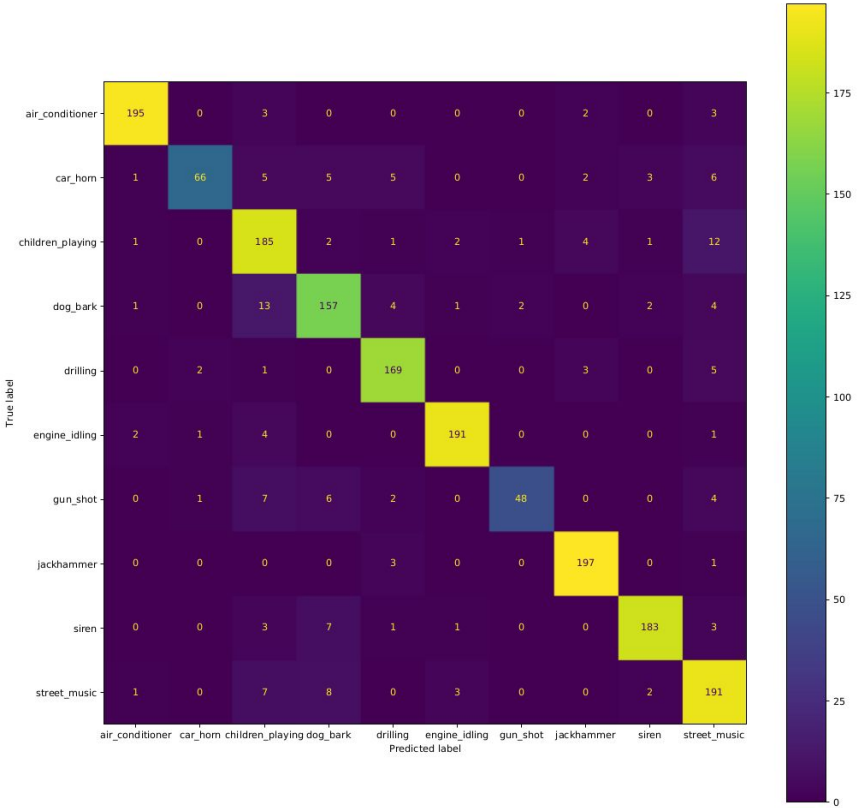
Naive bayes - Urabnsound8K and DCASE -2018

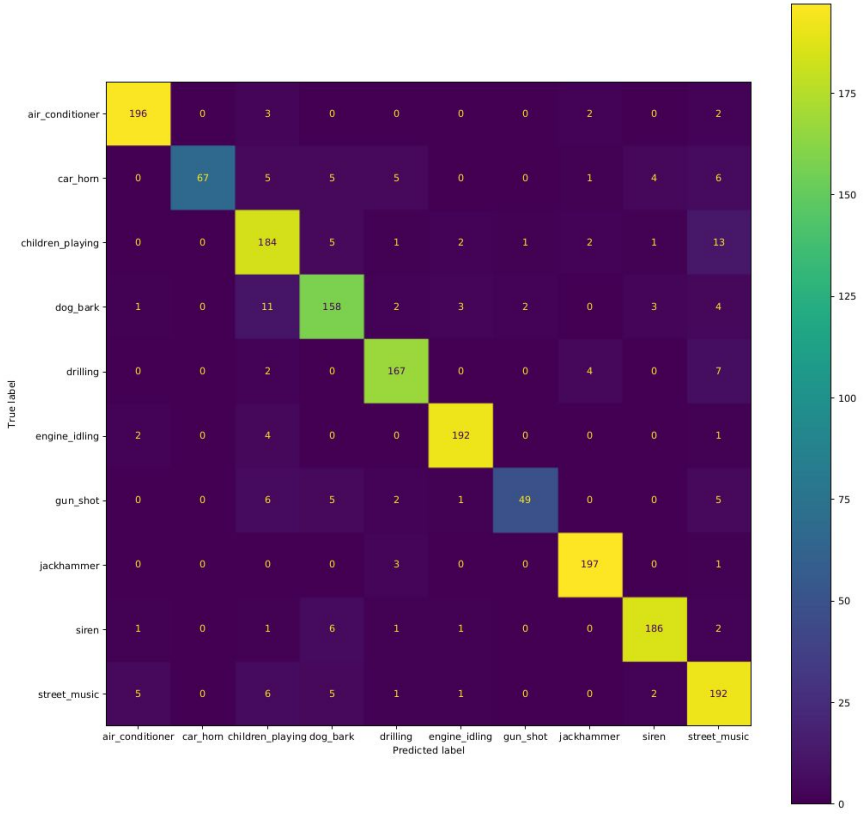
- MFCC as the extracted features of each audio using librosa library.
- Extracted mfcc features are fed into the naive bayes classifier.



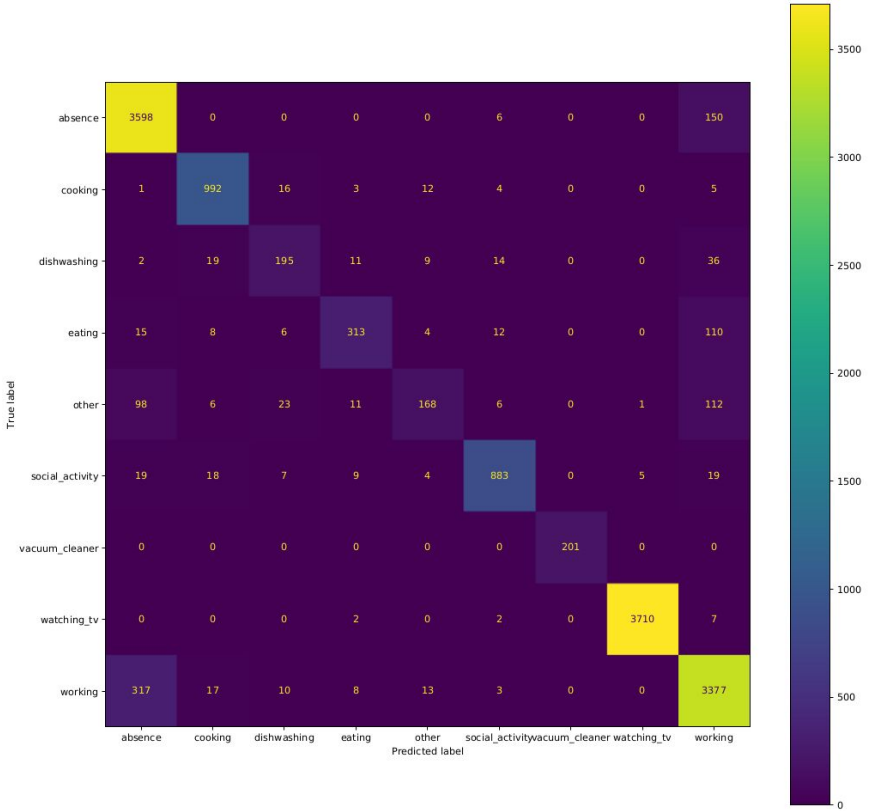


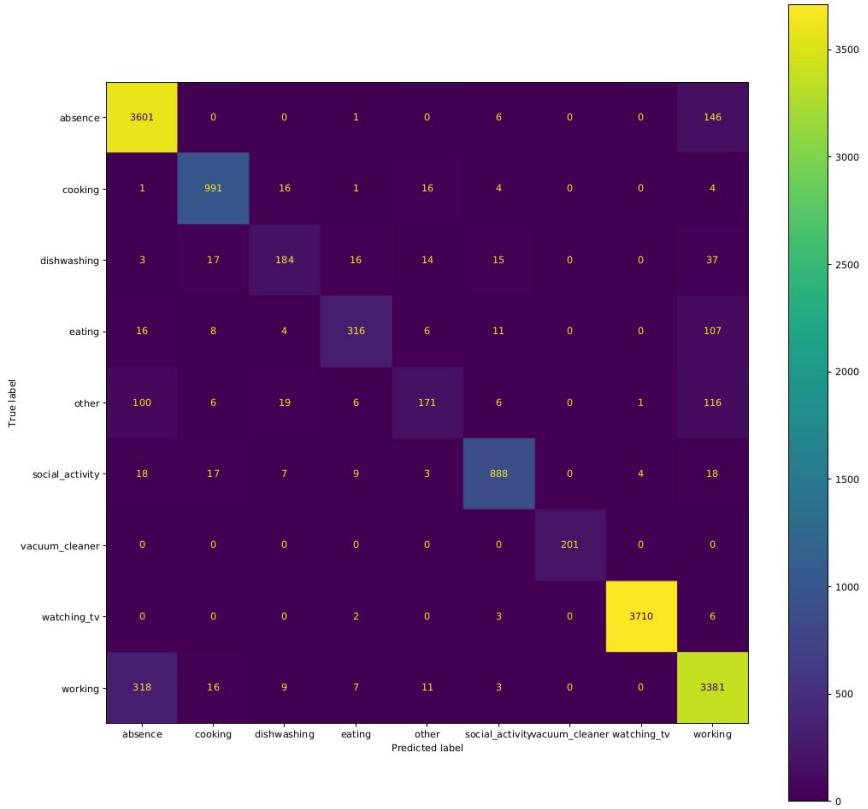
Random forest - Urbansound8K



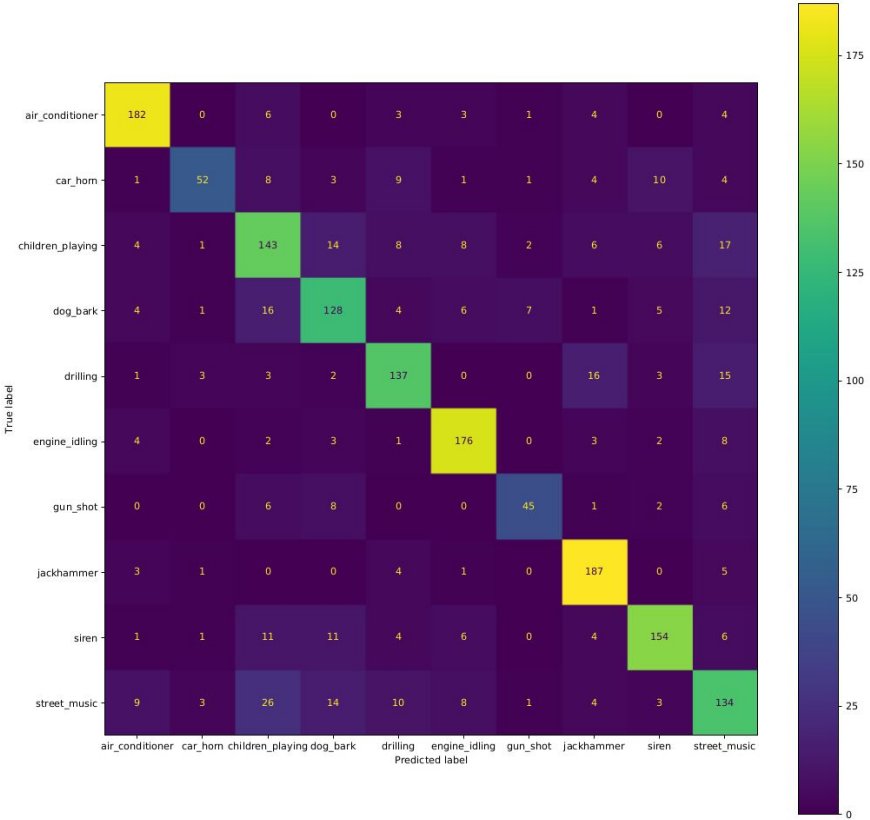


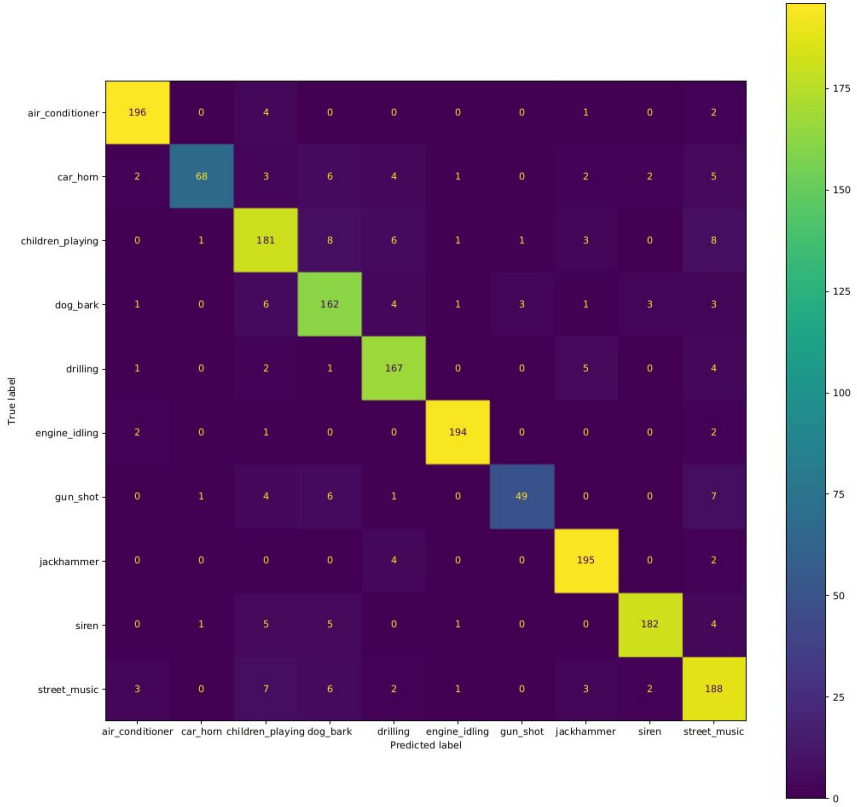
Random forest - DCASE2018



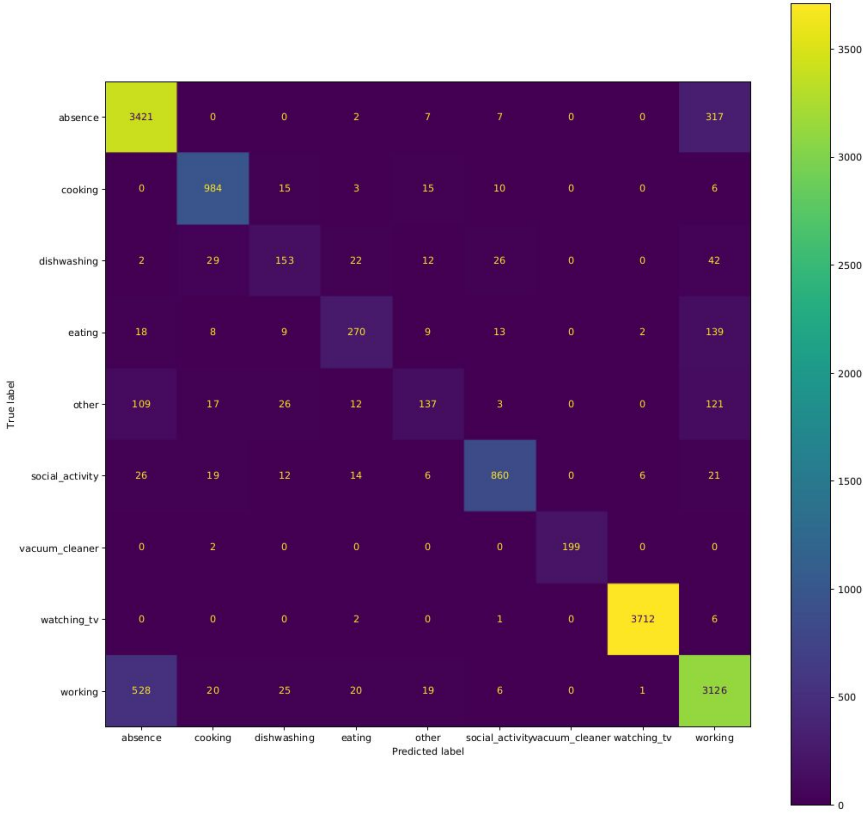


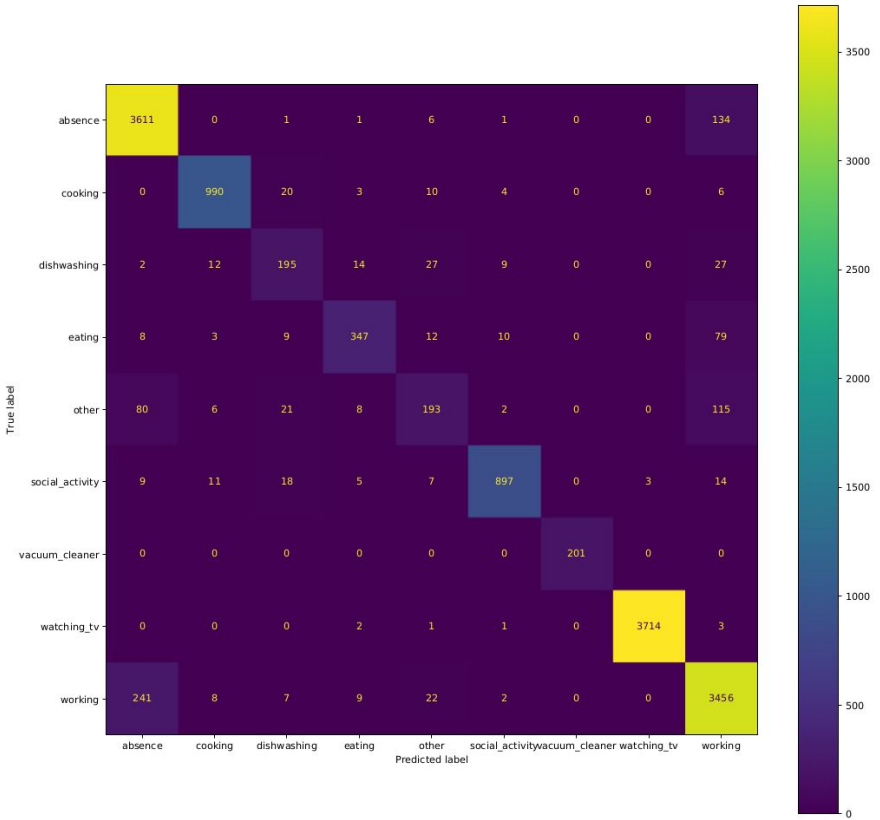
Gradient boosting - Urbansound8K



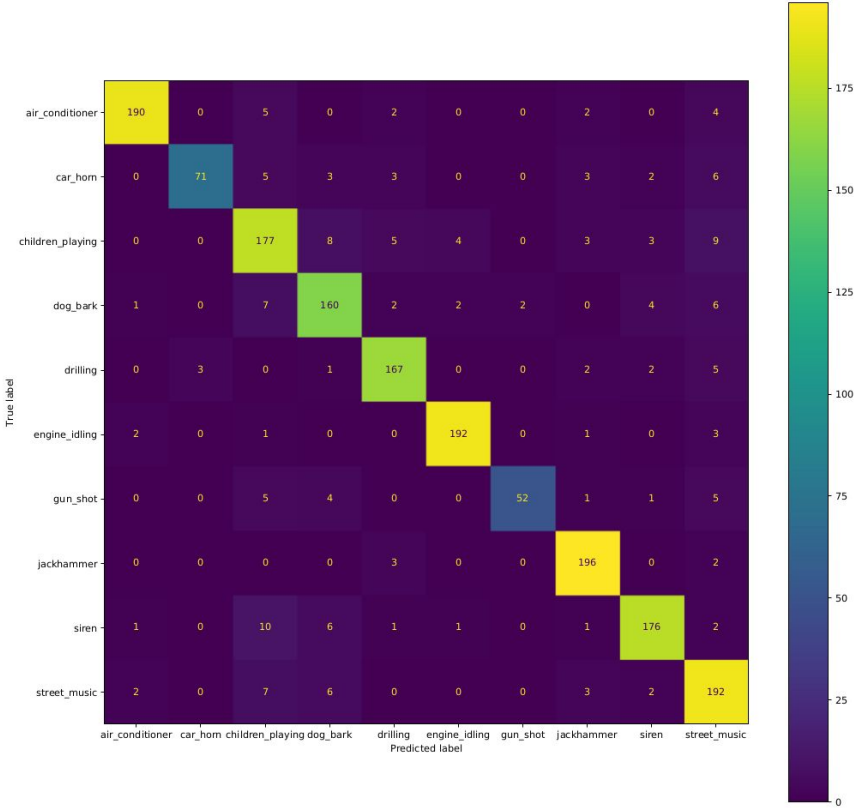


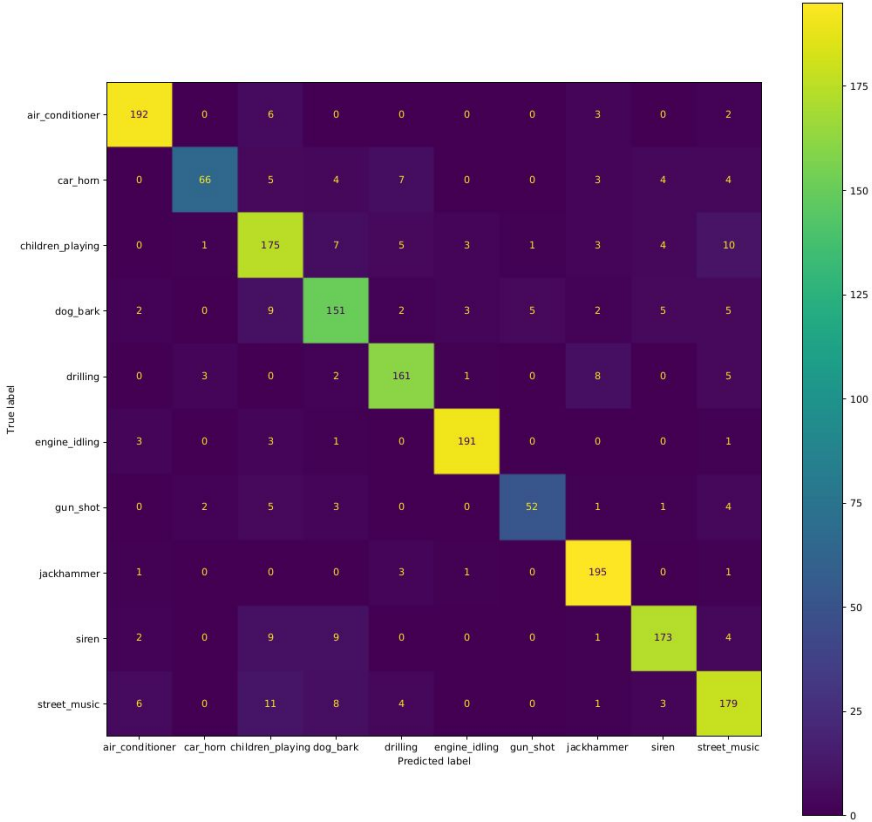
Gradient boosting - DCASE2018



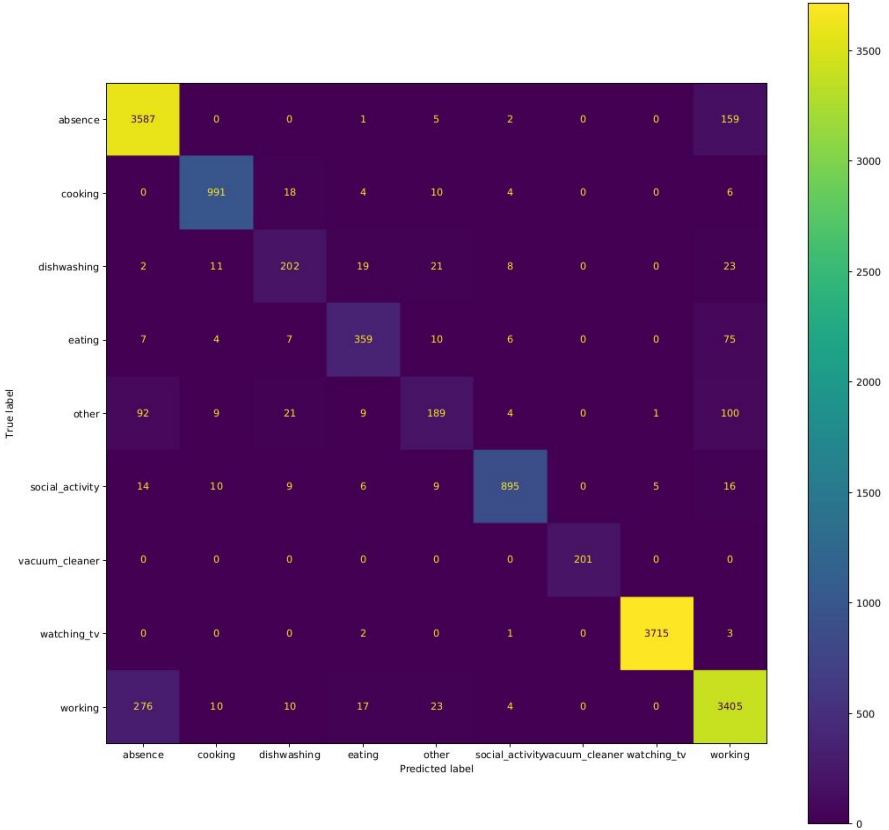


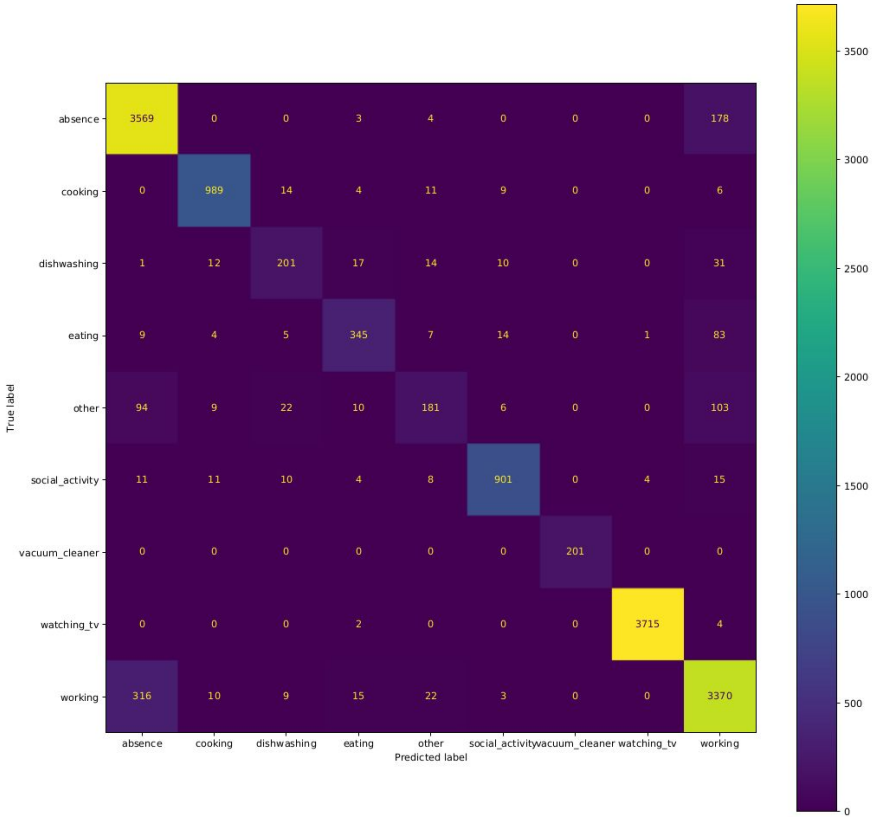
XG boosting - Urbansound8K



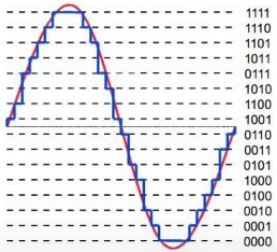


XG boosting - DCASE2018





Transfer learning Pipeline



frame 1: sample 1 … 128
frame 2: sample 64 … 192
frame 3: sample 128 … 256
frame 4: sample 192 … 320
...

Log mel spectrogram is generated using librosa framework

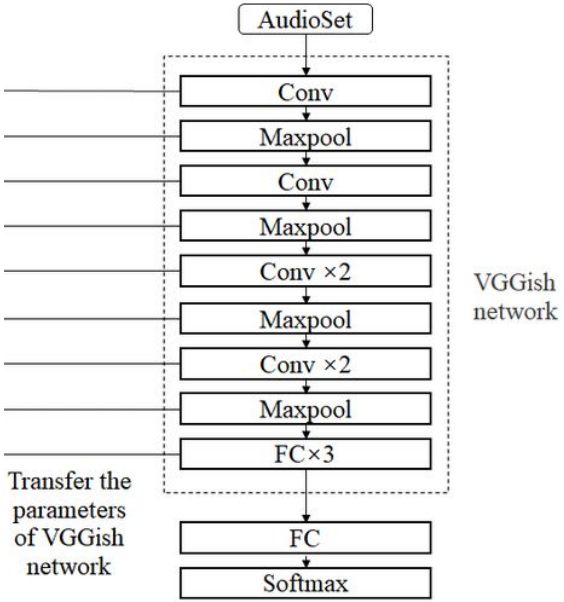
Generate samples from the log mel spectrogram

VGGish model [Pre-trained with audioset dataset]

Generate audio embeddings

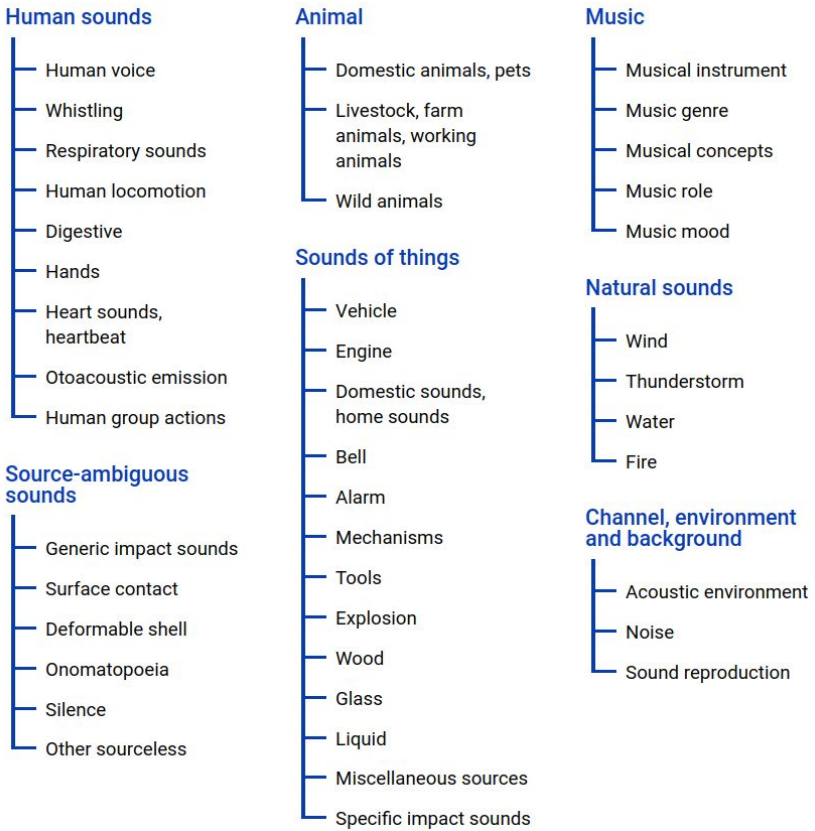
SVM classifier

VGGish model




```python
# Block 1
x = conv(64, name='conv1')(inputs)
x = maxpool(name='pool1')(x)

# Block 2
x = conv(128, name='conv2')(x)
x = maxpool(name='pool2')(x)

# Block 3
x = conv(256, name='conv3/conv3_1')(x)
x = conv(256, name='conv3/conv3_2')(x)
x = maxpool(name='pool3')(x)

# Block 4
x = conv(512, name='conv4/conv4_1')(x)
x = conv(512, name='conv4/conv4_2')(x)
x = maxpool(name='pool4')(x)

if include_top:
    dense = partial(tfkl.Dense, activation='relu')

    # FC block
    x = tfkl.Flatten(name='flatten_')(x)
    x = dense(4096, name='fc1/fc1_1')(x)
    x = dense(4096, name='fc1/fc1_2')(x)
    x = dense(params.EMBEDDING_SIZE, name='fc2')(x)

    if compress:
        x = Postprocess()(x)
else:
    globalpool = (
        tfkl.GlobalAveragePooling2D() if pooling == 'avg' else
        tfkl.GlobalMaxPooling2D() if pooling == 'max' else None)

    if globalpool:
        x = globalpool(x)
```

VGGish model trained with audioset



VGGish model trained with audioset

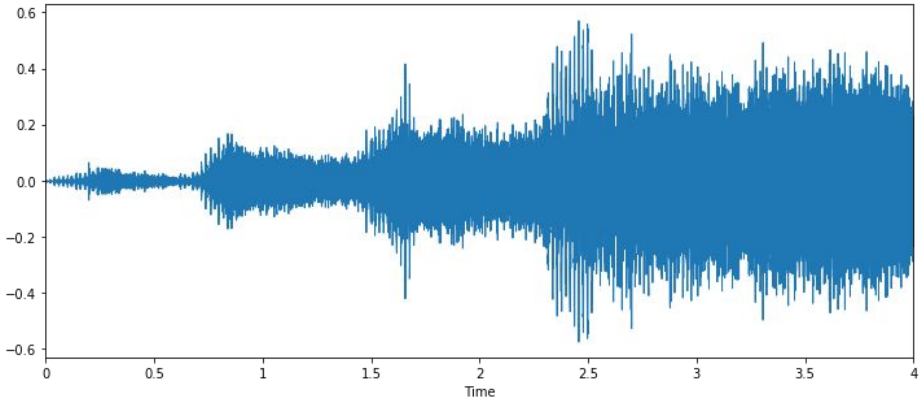
- AudioSet consists of an expanding ontology of 632 audio event classes
- A collection of 2,084,320 human-labeled 10-second sound clips drawn from YouTube videos.

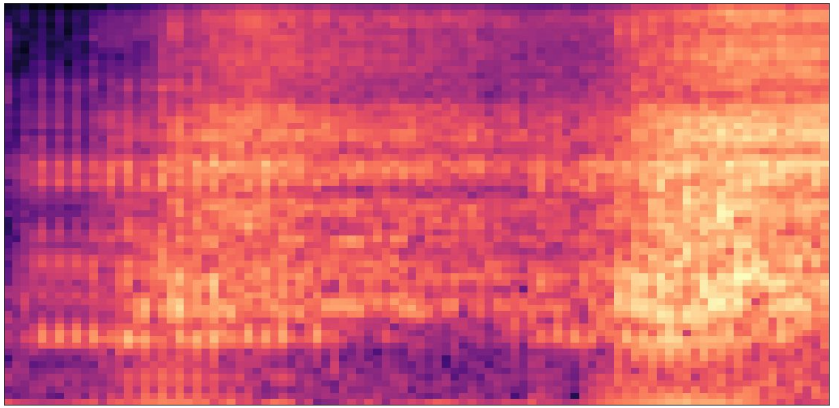
Label	Quality estimate?	Number of videos
Music	100%	1,011,305
Speech	100%	1,010,480
Vehicle	100%	128,051
Musical instrument	100%	117,343
Plucked string instrument	100%	44,565
Singing	100%	42,493
Car	100%	41,554
Animal	100%	40,758
Outside, rural or natural	100%	35,731
Violin, fiddle	100%	28,125
Bird	100%	26,894
Drum	100%	20,246
Engine	100%	16,245
Narration, monologue	100%	15,590
Drum kit	100%	15,169
Acoustic guitar	100%	14,568
Dog	100%	13,705
Child speech, kid speaking	100%	11,816
Bass drum	100%	9,292
Rail transport	100%	9,052
Motor vehicle (road)	100%	9,044
Water	100%	8,994
Female speech, woman speaking	100%	8,513
Siren	100%	8,498
Railroad car, train wagon	100%	8,361
Tools	100%	8,107
Silence	100%	7,662
Snare drum	100%	6,842
Wind	100%	6,805
Bird vocalization, bird call, bird song	100%	6,372
Fowl	100%	6,248
Wind instrument, woodwind instrument	100%	6,083
Emergency vehicle	100%	5,730
Laughter	100%	5,696
Chirp, tweet	100%	4,633
Rapping	100%	4,496
Cheering	100%	4,380
Gunshot, gunfire	100%	4,221
Radio	100%	4,214
Cat	100%	3,964
Hi-hat	100%	3,900
Helicopter	100%	3,698
Fireworks	100%	3,051
Stream	100%	2,847
Bark	100%	2,632
Baby cry, infant cry	100%	2,390
Snoring	100%	2,334
Train horn	100%	2,275
Double bass	100%	2,274
Explosion	100%	2,274
Crowing, cock-a-doodle-doo	100%	2,093
Bleat	100%	2,078
Computer keyboard	100%	2,042
Civil defense siren	100%	1,963
Bee, wasp, etc.	100%	1,868
Bell	100%	1,844
Chainsaw	100%	1,787
Oink	100%	1,784
Tick	100%	1,746
Tabla	100%	1,729
Liquid	100%	1,721
Traffic noise, roadway noise	100%	1,694
Beep, bleep	100%	1,600
Frying (food)	100%	1,594
Whack, thwack	100%	1,563
Sink (filling or washing)	100%	1,552
Burping, eructation	100%	1,302
Fart	100%	1,231
Sneeze	100%	1,200
Aircraft engine	100%	1,155
Arrow	100%	1,072
Giggle	100%	991
Hiccup	100%	931
Cough	100%	871
Cricket	100%	859
Sawing	100%	804
Tambourine	100%	745
Pump (liquid)	100%	603
Squeak	100%	192
Male speech, man speaking	90%	17,716
Keyboard (musical)	90%	10,473
Pigeon, dove	90%	8,523
Motorboat, speedboat	90%	8,078
Female singing	90%	7,949
Brass instrument	90%	7,513
Motorcycle	90%	7,261
Choir	90%	6,709
Race car, auto racing	90%	6,574
Chicken, rooster	90%	6,352

VGGish as feature extractor - Urbansound8k

- Randomly sampling 960ms clip from the input audio.
- Total samples = 4642*80+2851*80 = 228080 + 371360 = 599440 (Training: 4642*80 = 371360, Testing = 2851*80 = 228080). 98% of data is used for training and 1% data for validation and 1% data for testing.
- The 960 ms frames are decomposed with a short-time Fourier transform applying 25 ms windows every 10 ms.
- The resulting spectrogram is integrated into 64 mel-spaced frequency bins.
- This gives log-mel spectrogram patches of 96 × 64 bins that form the input to the VGGish model.
- The VGGish model is considered without the fully connected layer.
- VGGish converts audio input features into a semantically meaningful, high-level 512-D embedding that is fed as input to a downstream classification model.
- SVM as a classifier with C = 0.01
- Model score: 0.495

VGGish as feature extractor - Urbansound8k





VGGish as feature extractor - Urbansound8k

	precision	recall	f1-score
air_conditioner	0.38	0.32	0.35
car_horn	0.46	0.24	0.32
children_playing	0.48	0.52	0.50
dog_bark	0.51	0.64	0.57
drilling	0.59	0.64	0.61
engine_idling	0.27	0.38	0.31
gun_shot	0.28	0.86	0.43
jackhammer	0.61	0.54	0.57
siren	0.69	0.36	0.48
street_music	0.49	0.62	0.55
accuracy			0.50

VGGish as feature extractor - Urbansound8k

- Due to the poor performance of the pretrained VGGish model the weights of the model needs to be adjusted to the new dataset.

```
# Block 1
x = conv(64, name='conv1')(inputs)
x = maxpool(name='pool1')(x)

# Block 2
x = conv(128, name='conv2')(x)
x = maxpool(name='pool2')(x)

# Block 3
x = conv(256, name='conv3/conv3_1')(x)
x = conv(256, name='conv3/conv3_2')(x)
x = maxpool(name='pool3')(x)

# Block 4
x = conv(512, name='conv4/conv4_1')(x)
x = conv(512, name='conv4/conv4_2')(x)
x = maxpool(name='pool4')(x)

input_shape = (96, 64, 1)
model = Sequential()
model.add(vgg)
model.add(Dense(512,input_dim=input_shape))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(9, activation='softmax'))
model.compile(optimizers.RMSprop(lr=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
```

VGGish as feature extractor - Urbansound8k

Layer (type)	Output Shape	Param #
model (Functional)	(None, 512)	4499712
dense (Dense)	(None, 512)	262656
activation (Activation)	(None, 512)	0
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5130

Total params: 4,767,498
Trainable params: 3,807,754
Non-trainable params: 959,744

VGGish as feature extractor - Urbansound8k

- Common to all three training process:
  - Randomly sampling 960ms clip from the input audio.
  - Total samples: 599440 (Training: 4642*80 = 371360, Testing = 2851*80 = 228080)
  - The 960 ms frames are decomposed with a short-time Fourier transform applying 25 ms windows every 10 ms.
  - The resulting spectrogram is integrated into 64 mel-spaced frequency bins.
  - This gives log-mel spectrogram patches of 96 × 64 bins that form the input to the VGGish model.
- Training 1:
- Freeze block 1&2&3: https://wandb.ai/manojkl/vgg_training_03?workspace=user-manojkl
- Freeze block 1&2: https://wandb.ai/manojkl/vgg_training_04_1&2_frozen_layer?workspace=user-manojkl
- Freeze block 1: https://wandb.ai/manojkl/vgg_training_04_first_frozen_layer?workspace=user-manojkl

VGGish as feature extractor - DCASE2018

- Common to all three training process:
  - Randomly sampling 960ms clip from the input audio.
  - Total samples: 2,165,759
  - The 960 ms frames are decomposed with a short-time Fourier transform applying 25 ms windows every 10 ms.
  - The resulting spectrogram is integrated into 64 mel-spaced frequency bins.
  - This gives log-mel spectrogram patches of 96 × 64 bins that form the input to the VGGish model.
- Training 1:
- Freeze block 1&2&3: https://wandb.ai/manojkl/vgg_training_01_dcase_1&2&3_frozen_layer?workspace=user-manojkl
- Freeze block 1&2: https://wandb.ai/manojkl/vgg_training_01_dcase_1&2_frozen_layer?workspace=user-manojkl
- Freeze block 1: https://wandb.ai/manojkl/vgg_training_01_dcase_1_frozen_layer?workspace=user-manojkl

VGGish as feature extractor - Urbansound8k

- Common to all three training process:
  - Equally divide the entire audio frame into 960ms clip from the input audio.
  - Total samples: 21001
  - The 960 ms frames are decomposed with a short-time Fourier transform applying 25 ms windows every 10 ms.
  - The resulting spectrogram is integrated into 64 mel-spaced frequency bins.
  - This gives log-mel spectrogram patches of 96 × 64 bins that form the input to the VGGish model.
- Training 1:
- Freeze block 1&2&3: https://wandb.ai/manojkl/vgg_training_all_data_01?workspace=user-manojkl
- Freeze block 1&2: https://wandb.ai/manojkl/vgg_training_1&2_frozen_layer_all_data_03?workspace=user-manojkl
- Freeze block 1: https://wandb.ai/manojkl/vgg_training_first_frozen_layer_all_data_02?workspace=user-manojkl

VGGish as feature extractor - DCASE2018

- Common to all three training process:
  - Equally divide the entire audio frame 960ms clip from the input audio.
  - Total samples: 722,205
  - The 960 ms frames are decomposed with a short-time Fourier transform applying 25 ms windows every 10 ms.
  - The resulting spectrogram is integrated into 64 mel-spaced frequency bins.
  - This gives log-mel spectrogram patches of 96 × 64 bins that form the input to the VGGish model.
- Training 1:
- Freeze block 1&2&3: https://wandb.ai/manojkl/vgg_training_dcase_1&2&3_frozen_layer_all_data_01?workspace=user-manojkl
- Freeze block 1&2: https://wandb.ai/manojkl/vgg_training_dcase_1&2_frozen_layer_all_data_02?workspace=user-manojkl
- Freeze block 1: https://wandb.ai/manojkl/vgg_training_dcase_1st_frozen_layer_all_data_03?workspace=user-manojkl

Options

- Train a classification model with the original dataset. Generate audio embeddings with the VGGish model and give it as a input to the trained classifier for prediction of classes.
-

LSTM

- Random clip
- Entire image

LSTM Pipeline

- Random clip
- Entire image

Hybrid model CNN+LSTM Pipeline

- Random clip
- Entire image

Hybrid model CNN+LSTM

- Random clip
- Entire image

KNN - Urbansound8k

-

KNN - Dcase 2018

-

SVM - Urbansound8K

- MFCC as the extracted features of each audio using librosa library.
-

SVM - DCASE -2018

-

Random forest - Urbansound8K

- Optimized parameter

Random forest - DCASE2018

- Optimized parameter

Gradient boosting - Urbansound8K

- Parameter

Gradient boosting - DCASE2018

- Parameter

XG boosting - Urbansound8K

- Parameters

XG boosting - DCASE2018

- Parameters

Reference

1. Su, Y., Zhang, K., Wang, J., & Madani, K. (2019). Environment sound classification using a two-stream CNN based on decision-level fusion. Sensors, 19(7), 1733.
2. Lim, Hyungui, Jeongsoo Park, and Yoonchang Han. "Rare sound event detection using 1D convolutional recurrent neural networks." Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop. 2017.
3. https://urbansounddataset.weebly.com/urbansound8k.html
4. http://dcase.community/challenge2018/task-monitoring-domestic-activities
5. https://github.com/musikalkemist/AudioSignalProcessingForML