Audio Classification using Machine Learning

Jon Nordby jonnord@nmbu.no

November 15, 2018

Introduction

Goal

a machine learning practitioner
without prior knowledge about sound processing
can solve basic Audio Classification problems

Assumed knowledge

Machine learning basics

- Supervised vs unsupervised learning
- Common methods

Basic signal processing

- Sampling
- Frequency vs time-domain
- Fourier Transform
- Filter kernels, Convolutions

Study material

Computational Analysis of Sound Scenes and Events.
Virtanen,Plumbley,Ellis (2018)

Human and Machine Hearing - Extracting Meaning from Sound,
Second Edition. Richard F. Lyon (2018)

DCASE2018 Bird Audio Detection challenge

50+ papers on *Acoustic Event Detection* etc.

Machine Hearing

Examples

Various usecases and tasks that Machine Hearing can be applied to.

Speech Recognition

What is this person saying?

! failed to load audio sample

Musical key classification

What key is this music in?

! failed to load audio sample

Audio Scene

What kind of place is this from?

! failed to load audio sample

Medical diagnostics

Is this a healthy heart?

! failed to load audio sample

Industrial monitoring

Is this machine operating normally?

! failed to load audio sample

Ecoacoustics

What kind of animal is this?

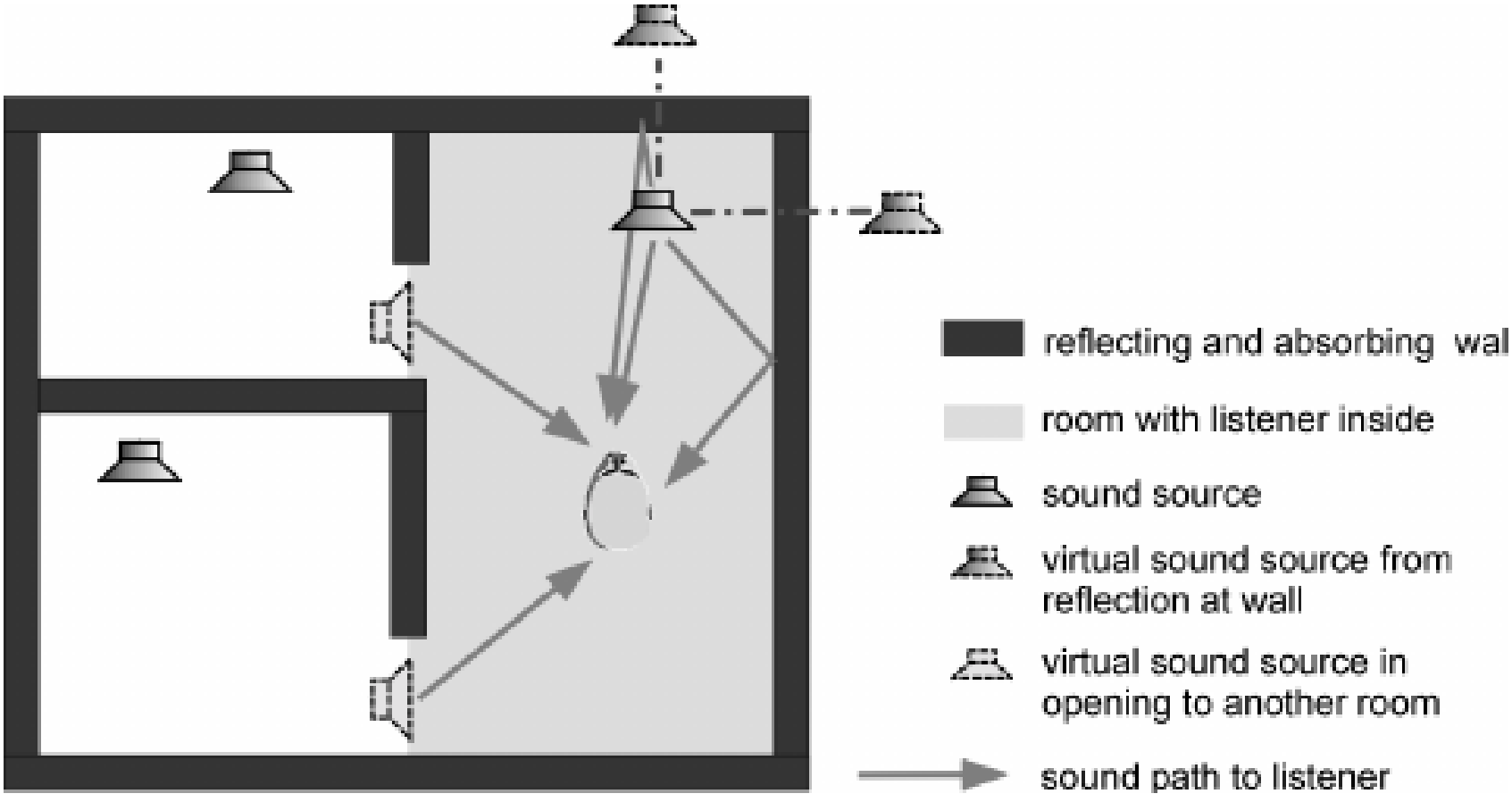
! failed to load audio sample

Established subfields

- Speech Recognition
- Music Information Retrieval
- **Sound Scenes and Events**

Brief primer on sound

Audio Mixtures



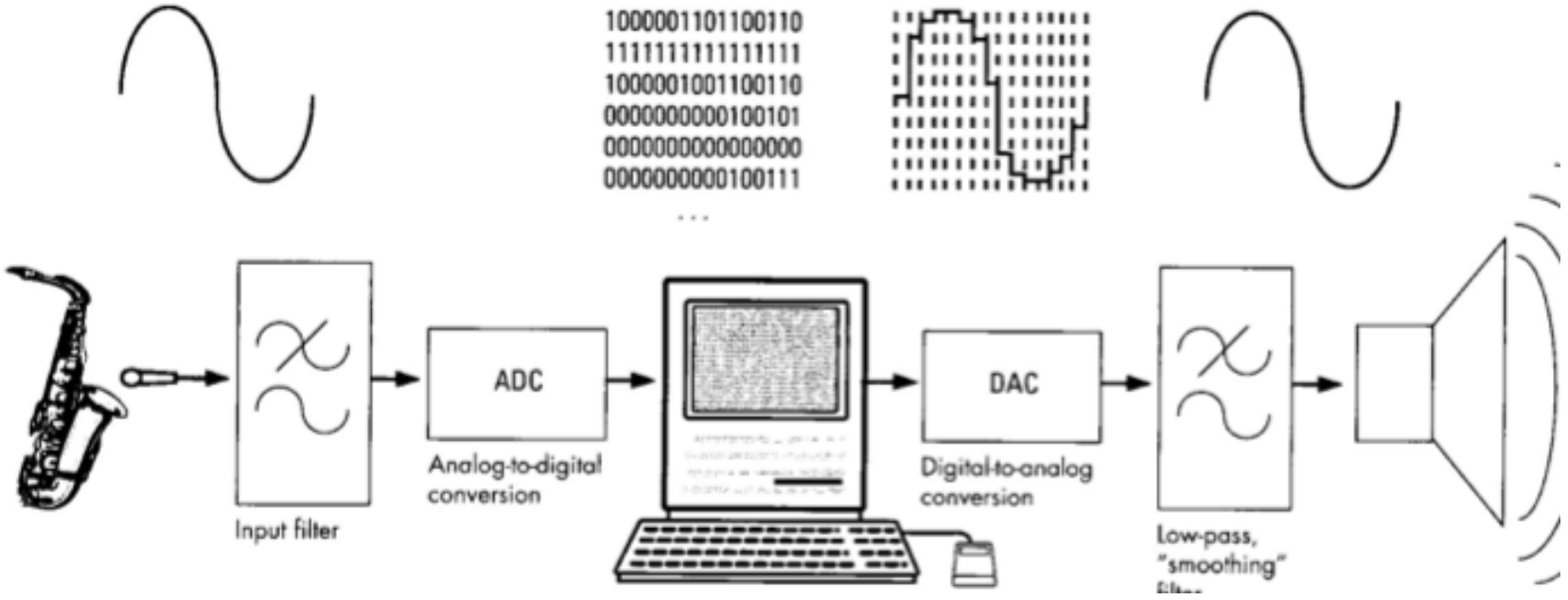
Human hearing

Two ears (Binaural). Frequencies approx 20Hz - 20kHz.

A non-linear system

- Loudness is not linear with sound pressure
- Loudness is frequency dependent
- Compression. Sensitivity lowered when loud
- Masking. Close sounds can hide eachother

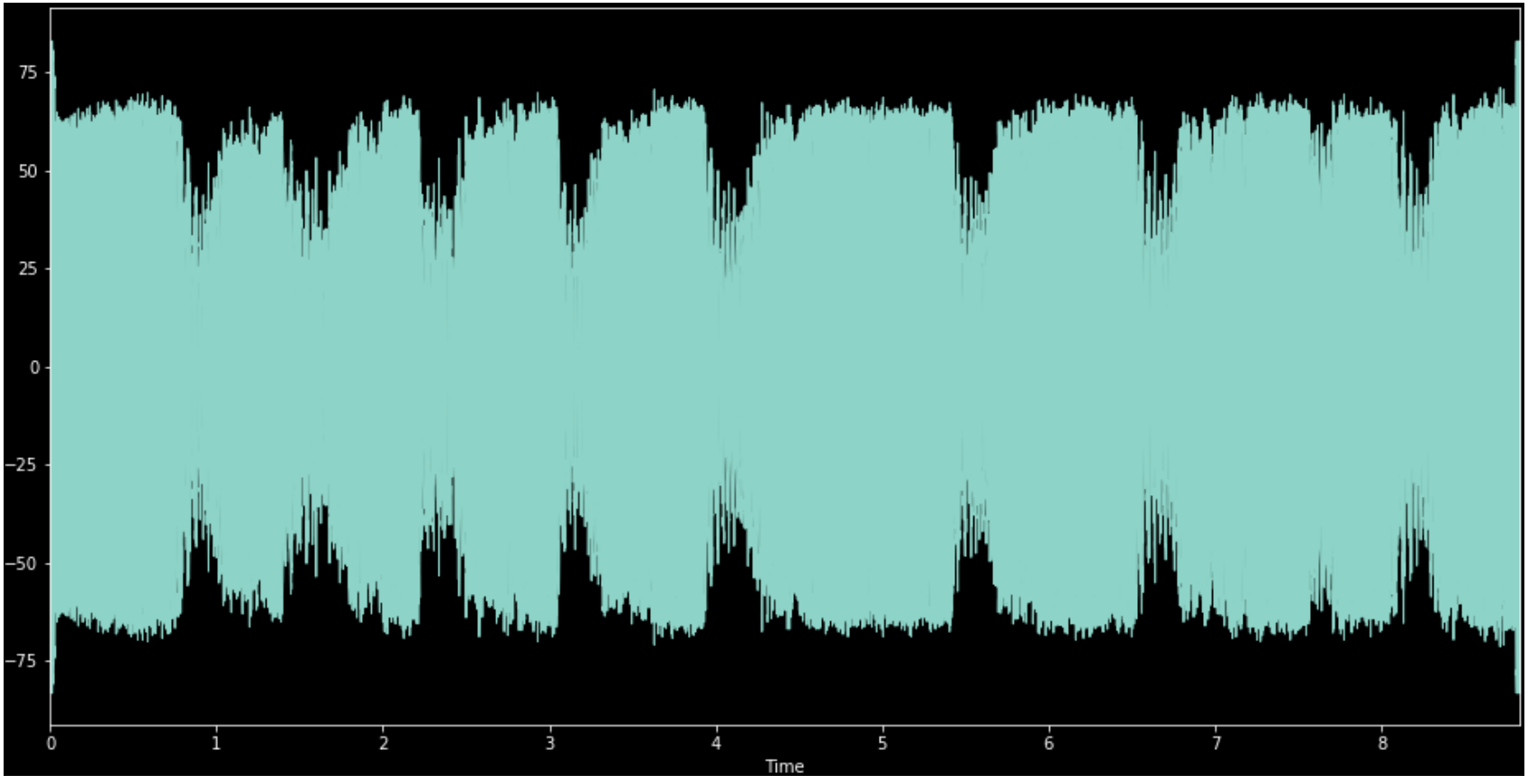
Digital sound pipeline



Digital sound representation

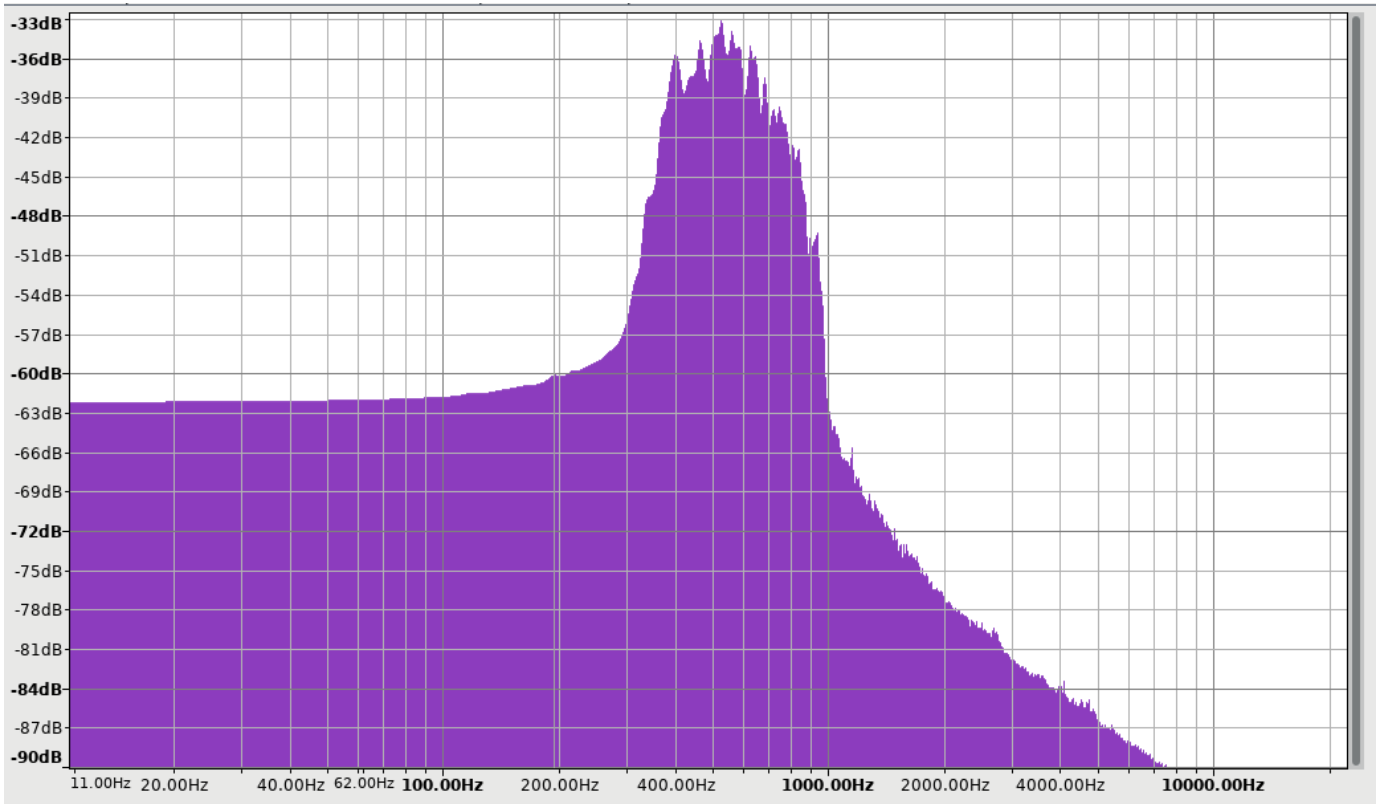
- Quantized in time (ex: 44100 Hz)
- Quantizied in amplitude (ex: 16 bit)
- N channels. **Mono**/Stereo
- Uncompressed formats: PCM .WAV
- Lossless compression: .FLAC
- Lossy compression: .MP3

Time domain



Normally logarithmic scale

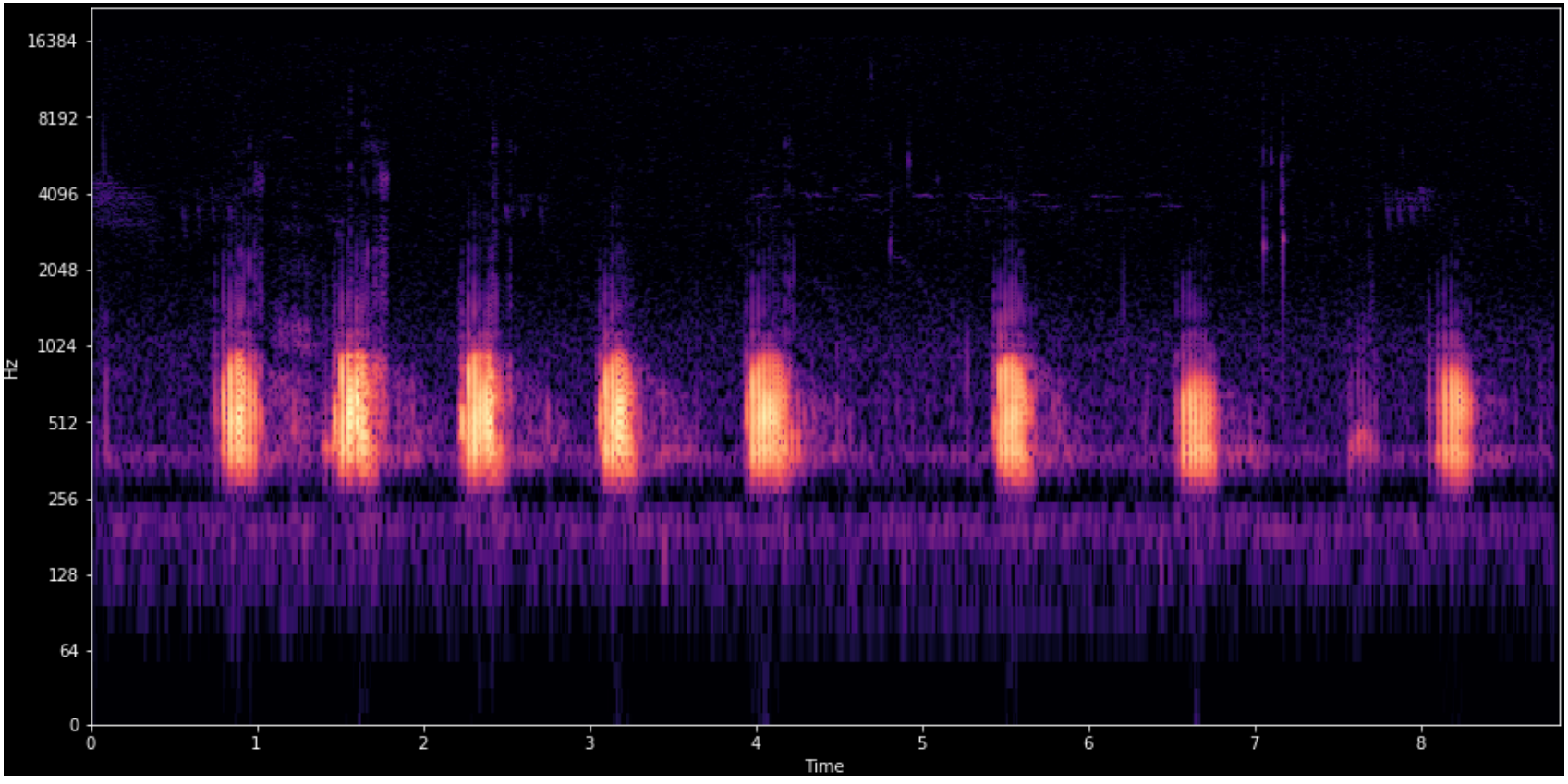
Frequency domain



Fourier Transform.

Time-frequency domain

Short-Time-Fourier-Transform (STFT)



Spectrogram

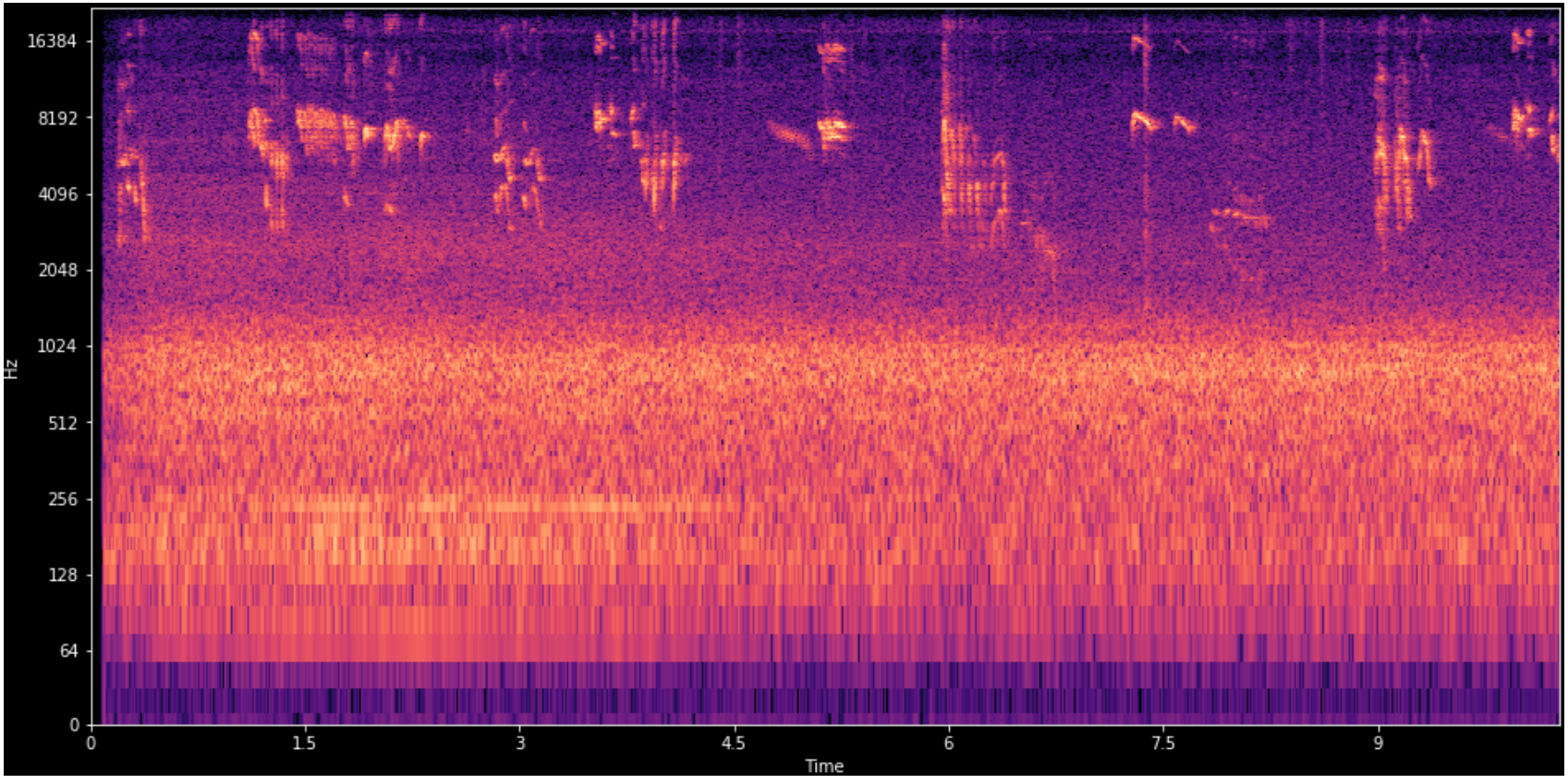
A practical example: Bird Detection

DCASE2018 challenge

- 10 second audio clips
- Has bird? yes/no => **binary classification**
- One label for entire clip => weakly annotated
- 3 training sets, 3 test sets. 45'000 samples

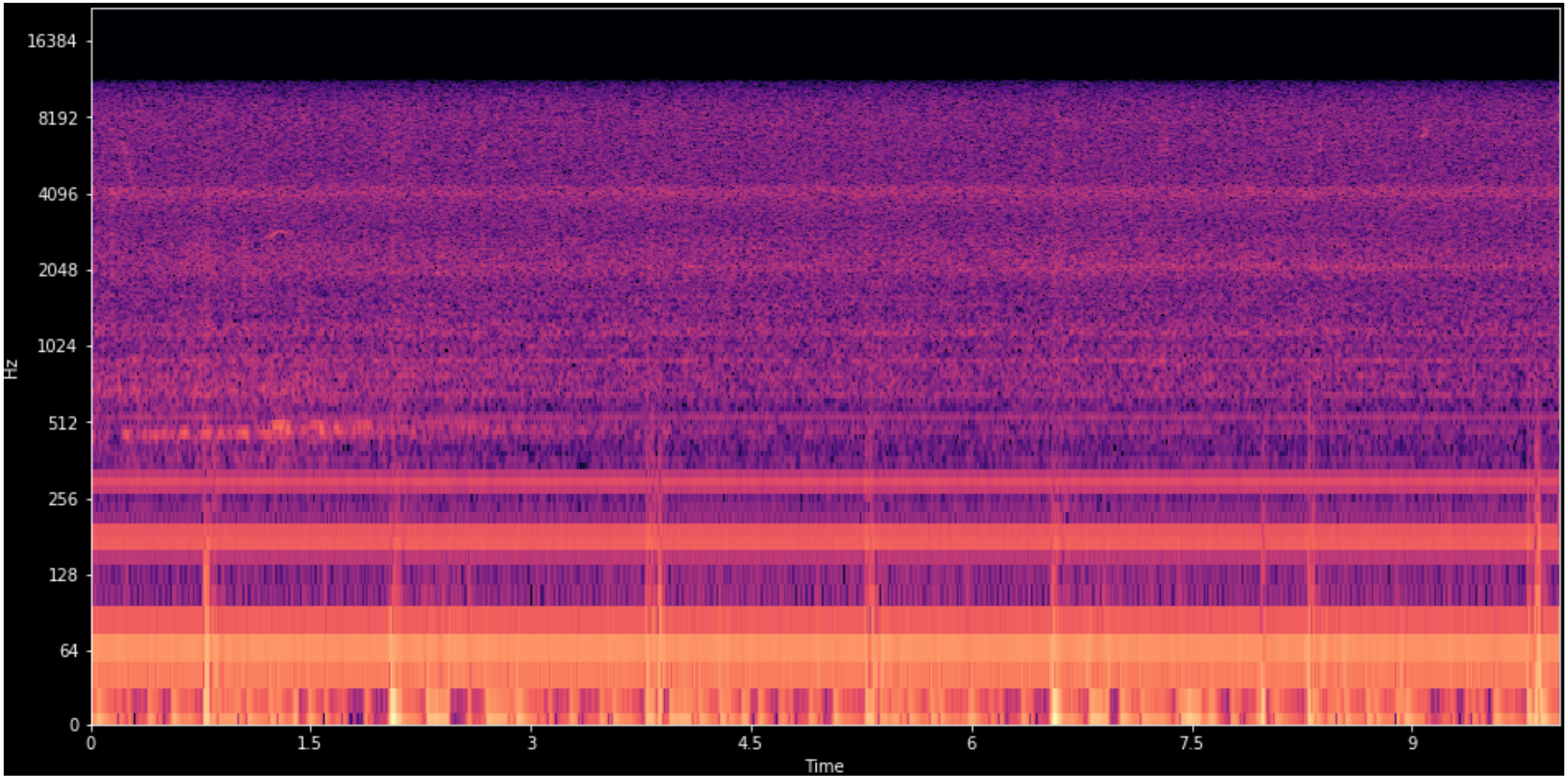
Mismatched conditions: 2 testsets with no training samples.

Bird sounds



! failed to load audio sample

More realistic



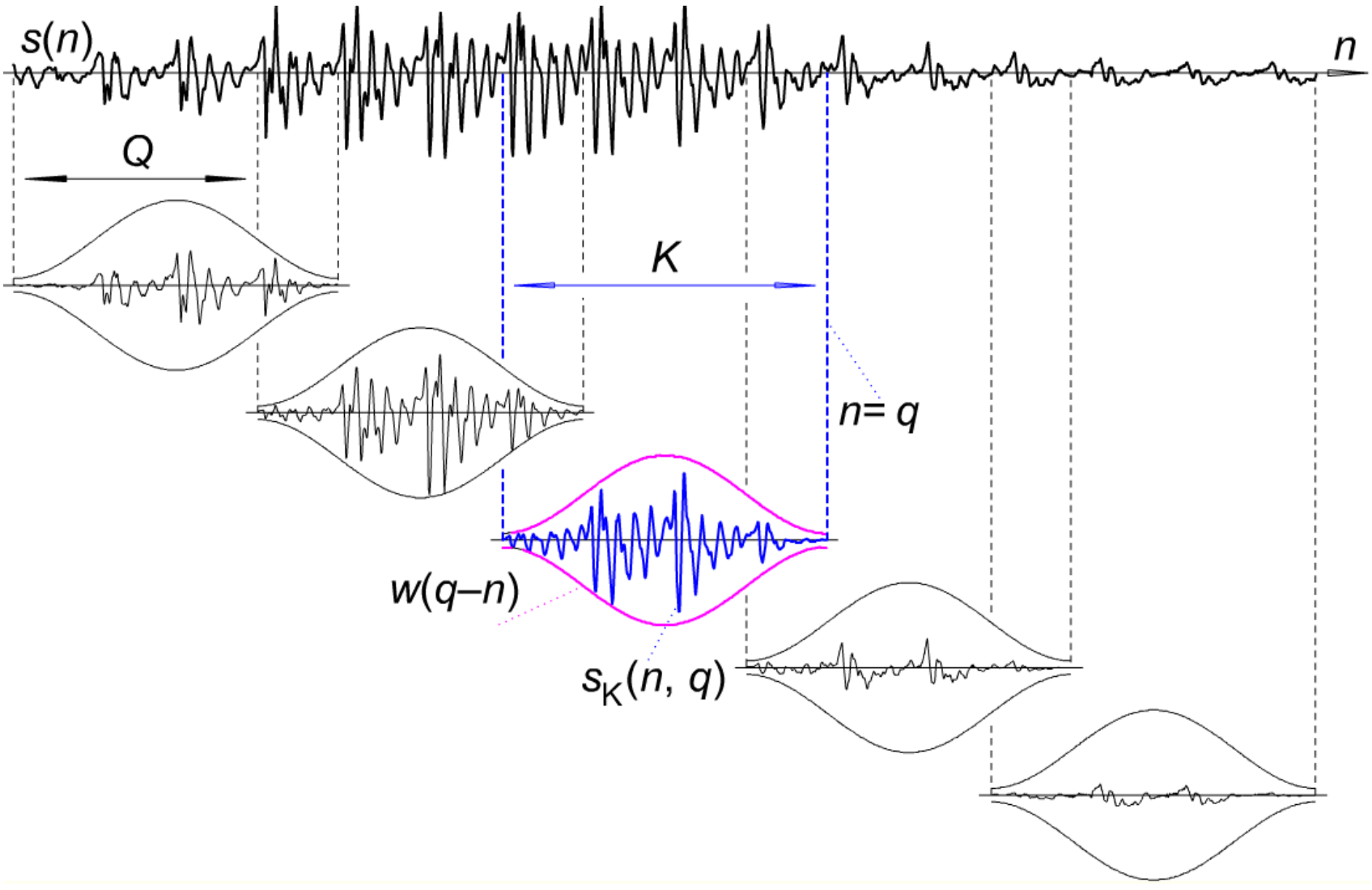
! failed to load audio sample

Feature extraction

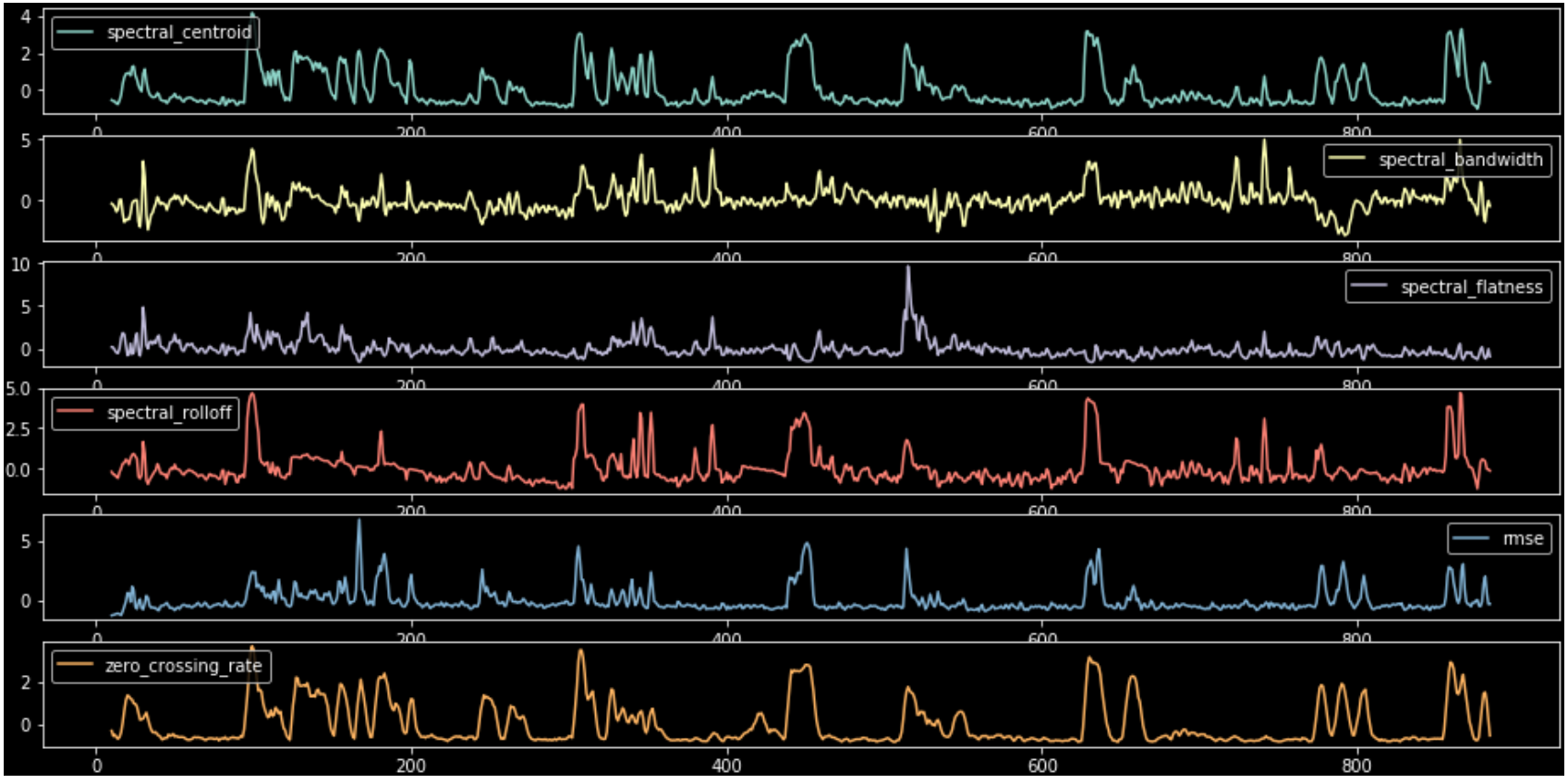
Audio classification pipeline

Frames

Cut audio into short overlapping segments

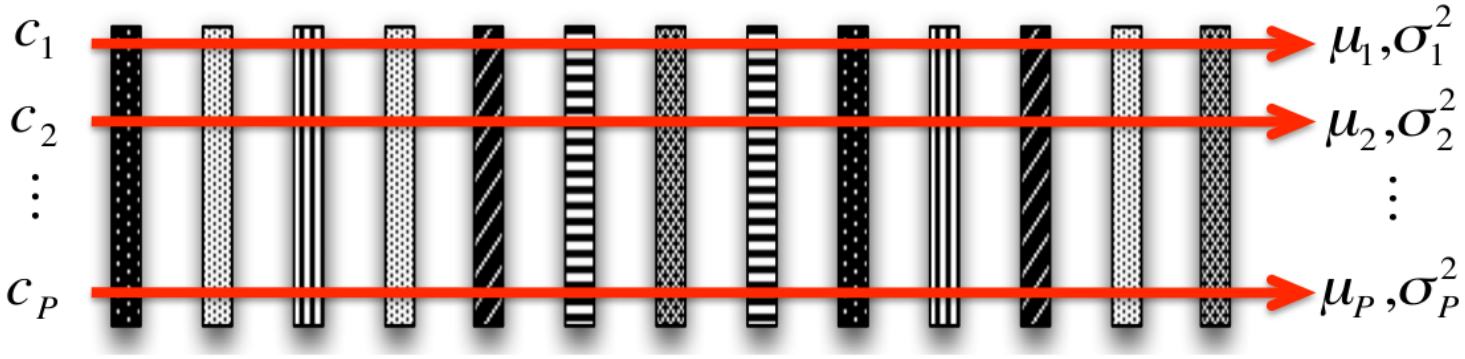


Low-level features



Basic statistics on spectrogram

Clip summarization



min,max,skew,Kurtosis,. . .

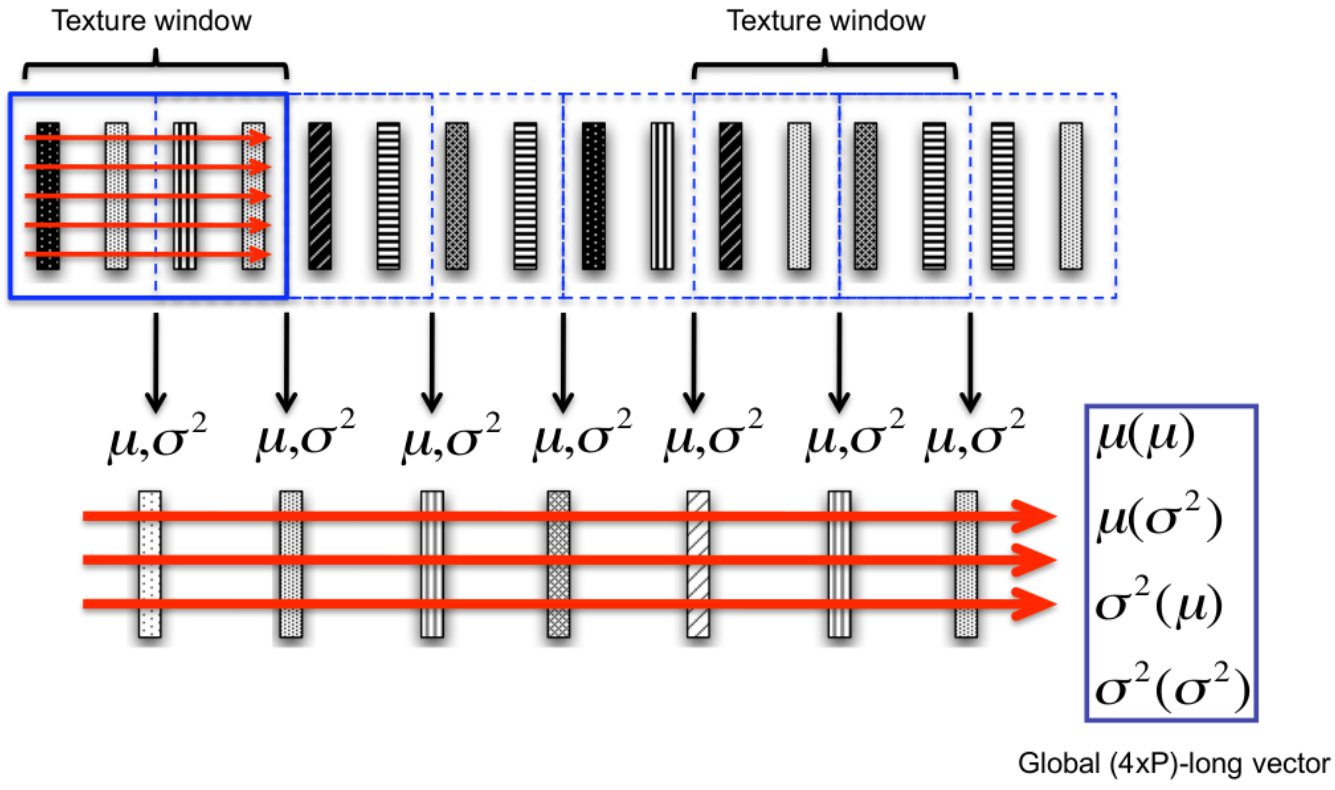
Delta frames

Delta frames: Difference between successive frames

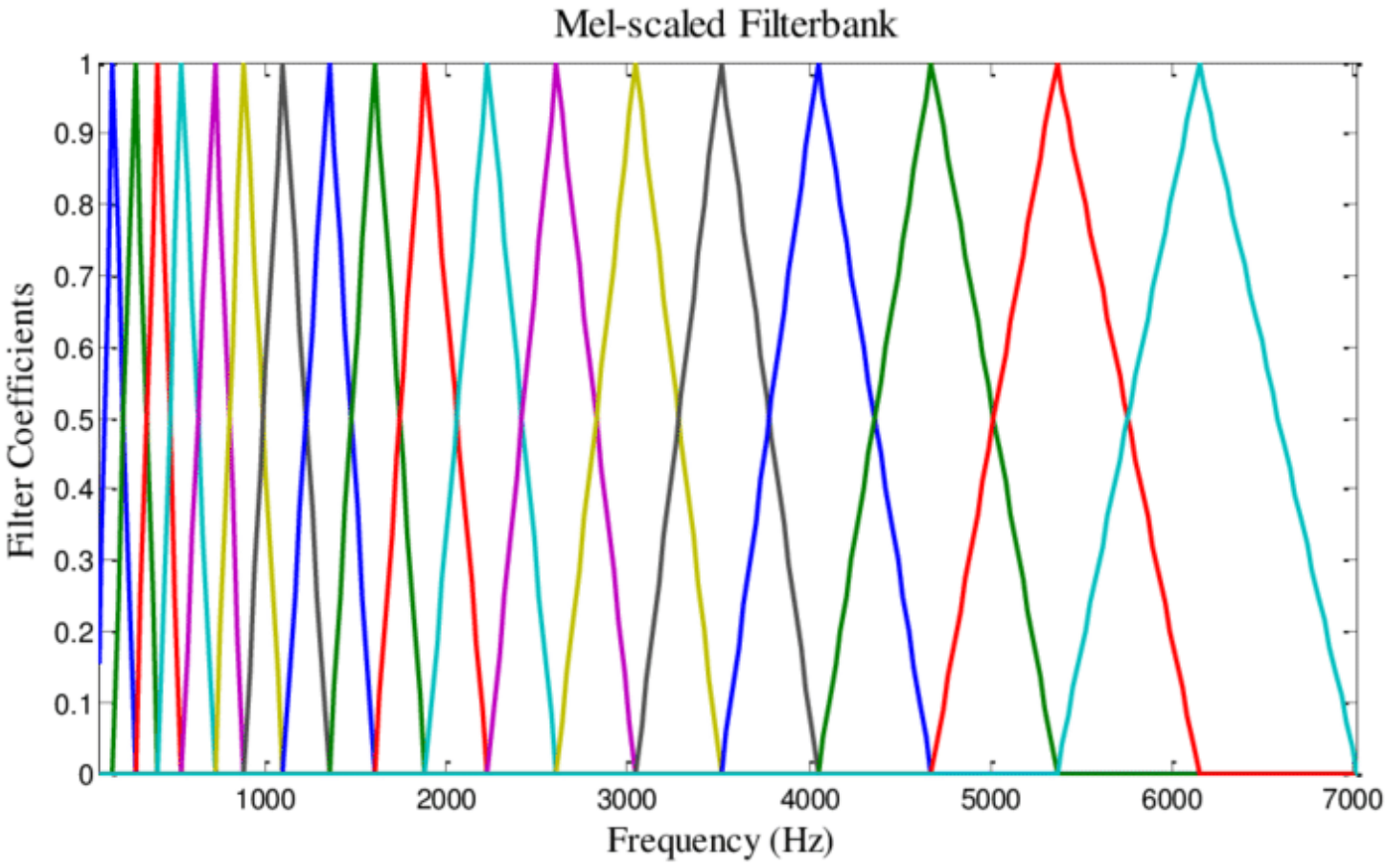
Delta-delta frames: Difference between delta frames

Summarized independently.

Texture windows

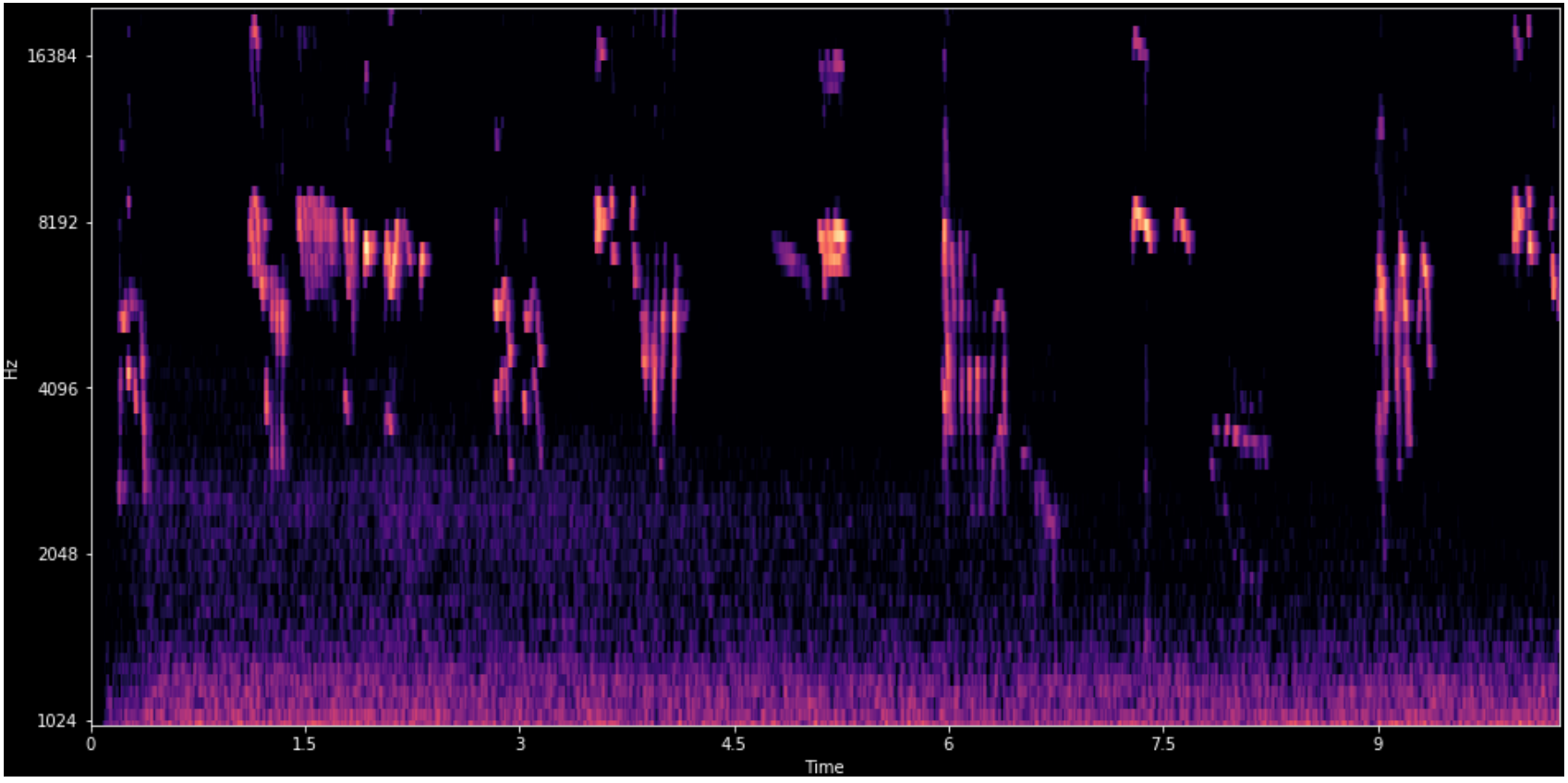


mel-scale filters



Reduces number of bands in spectrogram. Perceptually motivated.

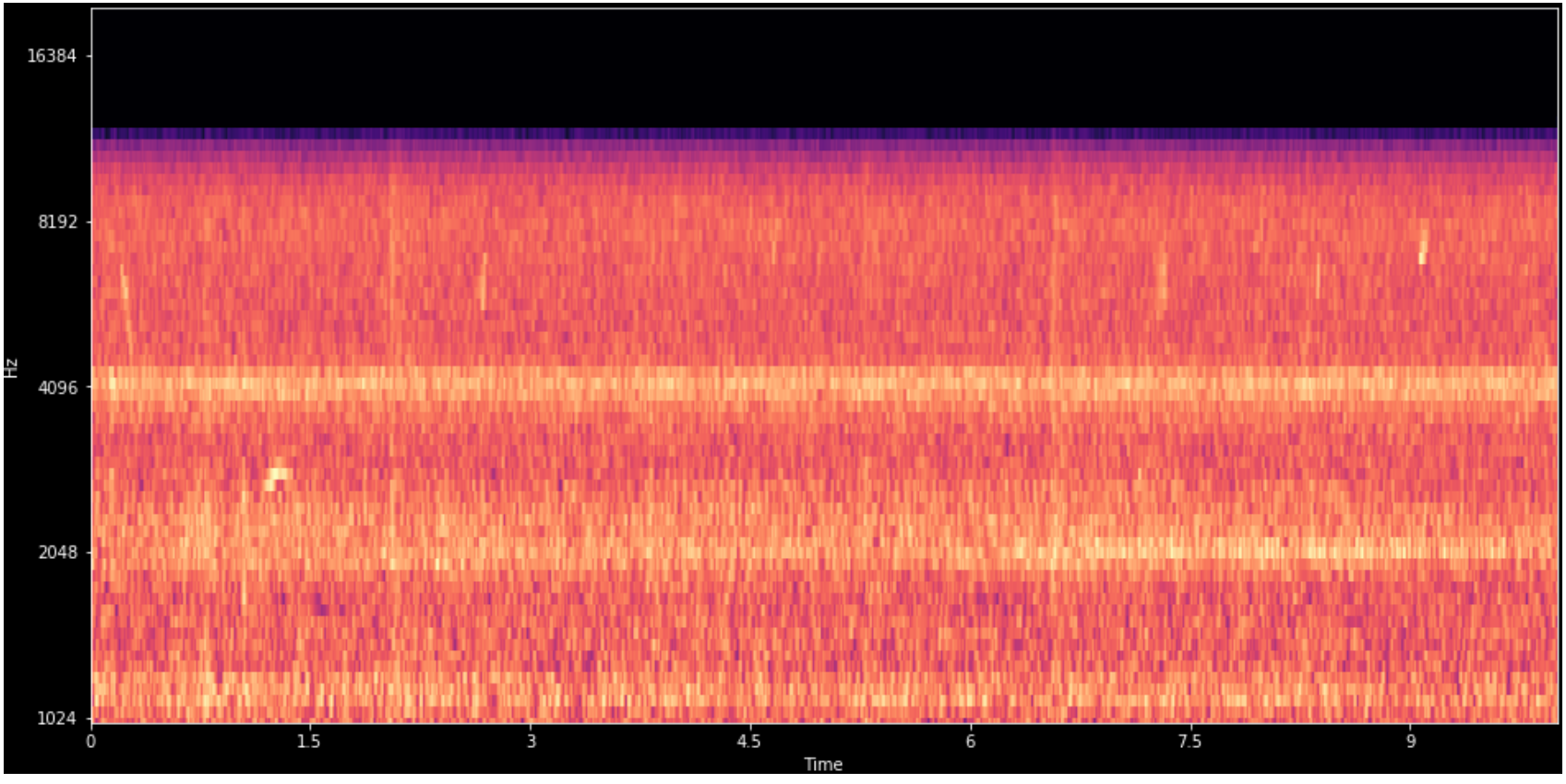
mel-spectrogram



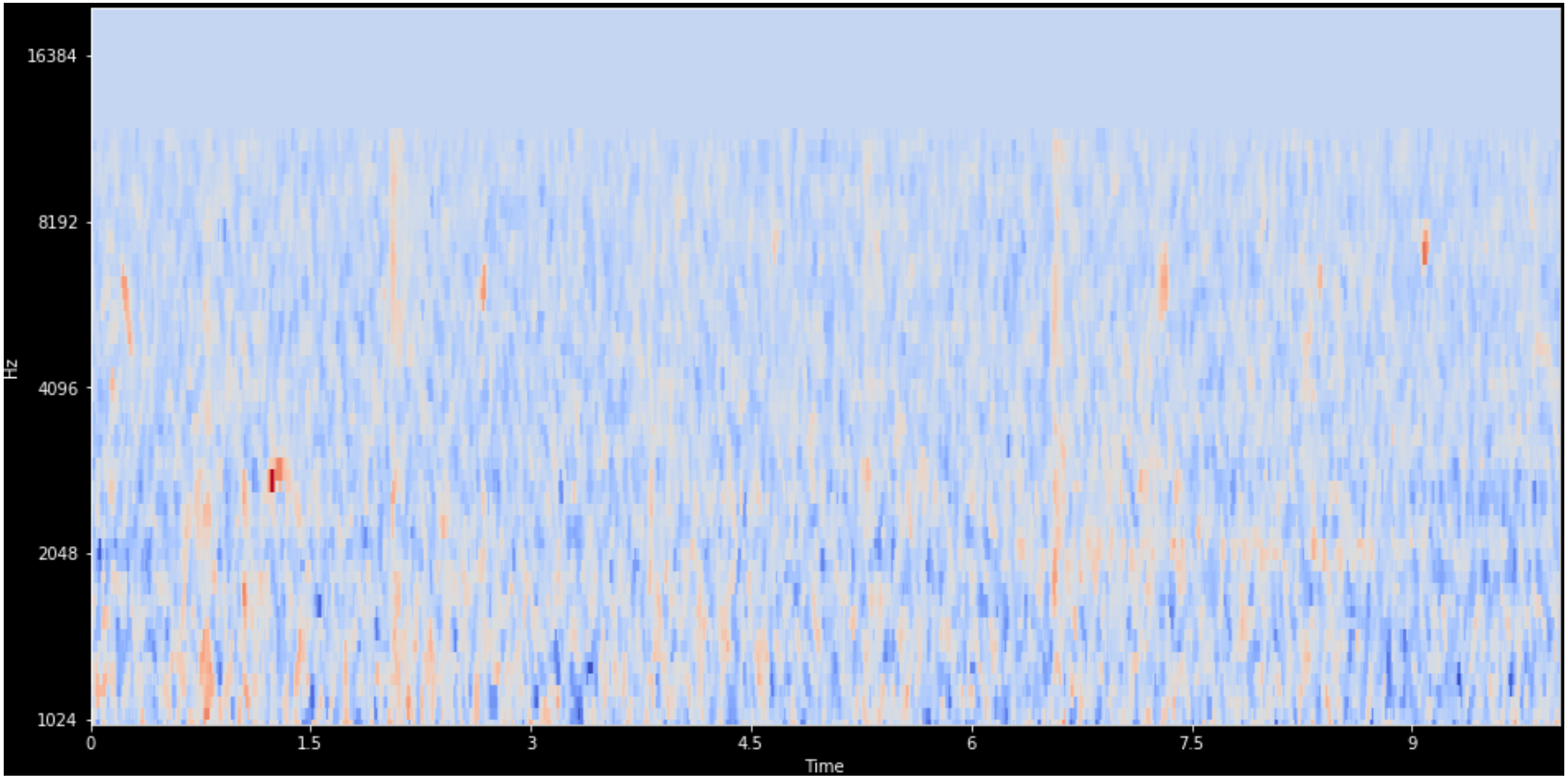
Spectrogram filtered by mel-scale triangular filters

What about noise?

There are birds in here!

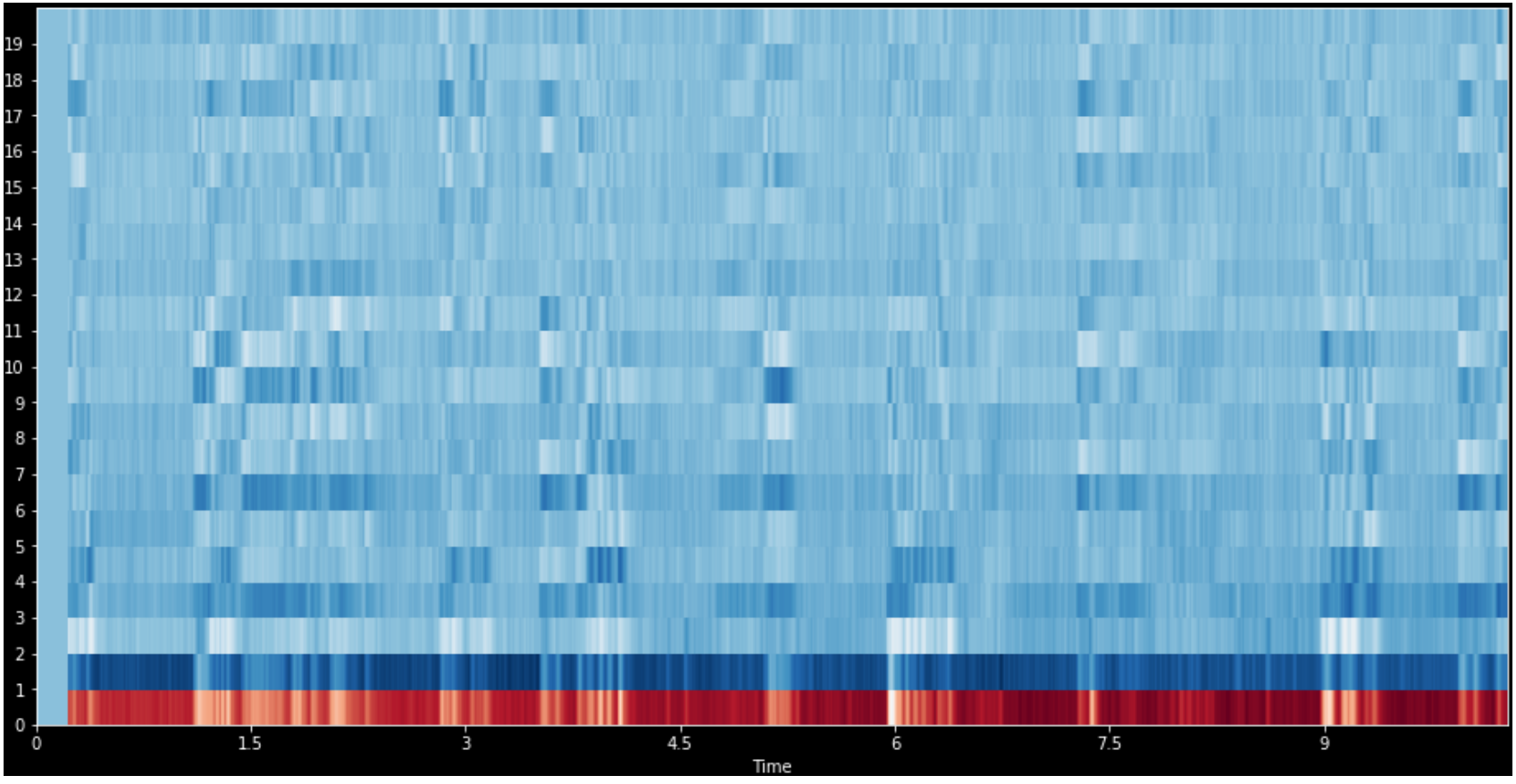


Filtered mel-spectrogram



Subtracted filterbank means, added Median filter (3x3)

Mel-filter Cepstrum Coefficients (MFCC)



Discrete Cosine Transform (DCT-2) of mel-spectrogram

Convolution

Local feature detector

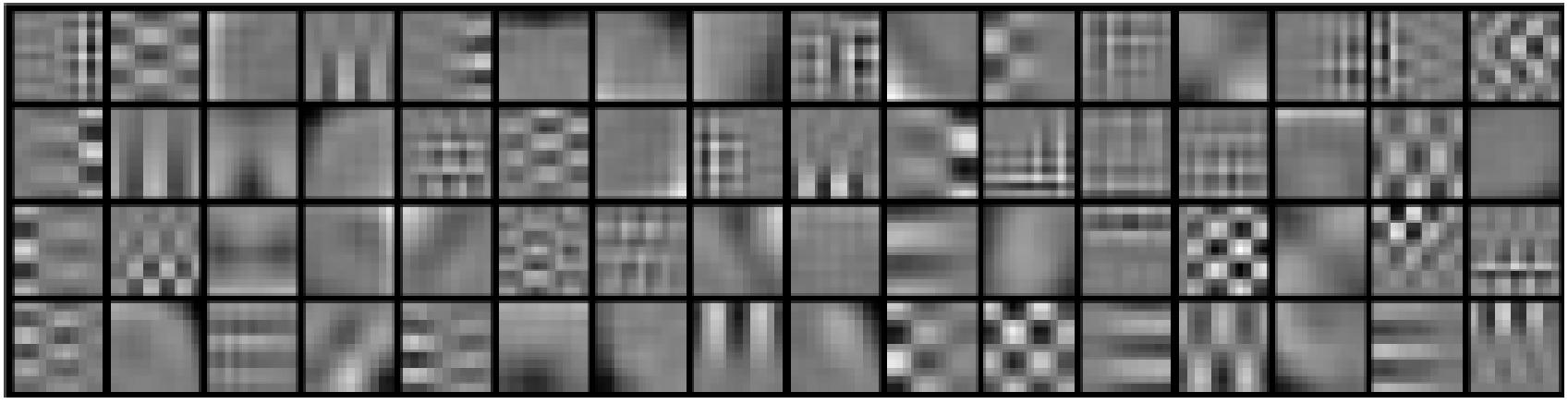
Input image

Convolution Kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Feature map

Dictionary of kernels



Feature learning

Unsupervised, from random spectrogram patches

- Clustering. Spherical k-means
- Matrix Factorization. Sparse Non-negative MF

Transfer: Copy from existing models

Advanced feature representations

Examples

- Wavelet filterbanks
- Scattering Transform
- CARFAC. Perceptual cochlear model

Not so much used

Classifiers

Classic models

The usual suspects

- Logistic Regression
- Support Vector Machine
- Random Forests

Also popular in Audio Classification

- Gaussian Mixture Models (GMM)
- Hidden Markov Model (HMM)

Deep learning

- Convolutional Neural Network (CNN) + dense layers
- Fully Convolutional Neural Network (FCNN)
- Recurrent Convolutional Neural Networks (RCNN)

Comparison of different approaches

Workbook

https://github.com/jonnor/birddetect

Important files:

- Model.ipynb
- dcase2018bad.py
- features.py

Classifier

```
rf = make_pipeline(
    RandomForestClassifier(n_estimators=100, min_samples_leaf=2, random_state=1),
)

X_train, X_test, Y_train, Y_test = \
  model_selection.train_test_split(train_X, train_Y, test_size=0.3)

start = time.time()
print('Starting train', X_train.shape, numpy.mean(Y_train))
rf.fit(X_train, Y_train)
end = time.time()
print('Train time', end-start)

print('train', model_selection.cross_val_score(rf, X_train, Y_train, scoring='roc_auc', cv=3))
print('test', model_selection.cross_val_score(rf, X_test, Y_test, scoring='roc_auc', cv=3))

```

```
Starting train (24983, 64) 0.5048232798302846
Train time 70.53949069976807
train [0.85059628 0.8534503  0.85257123]
test [0.84328148 0.83735134 0.83947204]
```

Feature extraction

```
def melspec_maxp(data, sr):
    params = dict(n_mels=64, fmin=500, n_fft=2048, fmax=150
    mel = librosa.feature.melspectrogram(y=data, sr=sr, **p

    mel = meansubtract(mel)
    mel = minmaxscale(mel)
    # mel = medianfilter(mel, (3,3))

    features = numpy.concatenate([
        numpy.max(mel, axis=1),
    ])
    return features
```

Dask parallel processing

```
    chunk_shape = (chunk_size, feature_length)
    def extract_chunk(urls):
        r = numpy.zeros(shape=chunk_shape)
        for i, url in enumerate(urls):
            r[i,:] = feature_extractor(url)
        return r

    extract = dask.delayed(extract_chunk)
    def setup_extraction(urls):
        values = extract(urls)
        arr = dask.array.from_delayed(values,
                                      dtype=numpy.float,
                                      shape=chunk_shape)
        return arr

    arrays = [ setup_extraction(c) for c in chunk_sequence
    features = dask.array.concatenate(arrays, axis=0)
    return features
```

Feature processing time

41'000 audio files... 0.2 seconds each

Laptop: **2 hours**

5 dual-core workers: **10 minutes**

Cost for 10 hours compute: <50 NOK

https://docs.dask.org/en/latest/setup/kubernetes.html

Results

Name	Features	Classifier	AUC ROC
Lasseck	melspectrogram	CNN	89%
.....	melspectrogram	CNN	78%-84%
skfl	melspec-conv-skmeans	RandomForest	73.4 %
jonnor	melspec-max	RandomForest	70%[1]
smacpy	MFCC-meanstd	GMM	51.7 %

http://dcase.community/challenge2018/task-bird-audio-detection-results

1. Public leaderboard score, not submitted for challenge

Best models also

Data Augmentation

- Random pitch shifting
- Time-shifting
- Time reversal
- Noise additions

Tricks

- Ensemble. Model averaging
- Self-adaptation. Pseudo-labelling

Summary

Feature representation

Try first **mel-spectrogram** (log or linear).

MFCC only as fallback

Machine Learning method

Try Convolutional Neural Networks (or RCNN) first.

Alternative: Learned convolutional kernels + RandomForest

Probably avoid: MFCC + GMM/HMM

Tricks

Subtract mel-spectrogram mean.

Consider median filtering.

Use data augmentation.

Try Transfer Learning. Can be from image model!

Questions?

Bonus

End2End learning

Using the raw audio input as features with Deep Neural Networks.

Need to learn also the time-frequency decomposition, normally performed by the spectrogram.

Actively researched using advanced models and large datasets.

Remaining work

- Implement kernel learning (spherical k-means)
- Implement a Convolutional Neural Network
- Compare the different models, summarize
- Finish writing report

DCASE2018 workshop

Reports from challenge tasks:

1) Acoustic scene classification
2) General-purpose audio taggging
3) **Bird Audio Detection**
4) semi-supervised: Domestic sound event detection
5) multi-channel acoustics: Domestic activities

I am going! November 19-21, London.

Continious Monitoring

Today: Audio collected and classified periodically

Future: Contious process using Internet-of-Things sensors

Writing a report in TIP360:

Designing a Wireless Acoustic Sensor Network for machine learning

Problem formulations

Classification

Return: class of this audio sample

- Bird? yes/no (binary)
- Which species is this? (multi-class)

Event detection

Return: time something occurred.

- "Bird singing started", "Bird singing stopped"
- Classification-as-detection. Classifier on short time-frames
- Monophonic: Returns most prominent event

Polyphonic events

Return: times of all events happening

Examples

- Bird singing, Human talking, Music playing
- Bird A, Bird B singing.

Approaches

- separate classifiers per ‘track’
- joint model: multi-label classifier

Audio segmentation

Return: sections of audio containing desired class

- Ex: based on Event Detection time-stamps
- Pre-processing to specialized classifiers

Source separation

Return: audio with only the desired source

- Masking in time-frequency domain
- Binary masks or continious
- Blind-source or Model-based

Other problem formulations

- Tagging
- Audio fingerprinting.
- Searching: Audio Information Retrieval

Misc

Desirable traits

What is needed for good audio classification?

- Volume independent
- Robust to mixtures of other sounds
- Handles intra-class variations. Different birdsong
- Can exploit frequency patterns
- Can exploit temporal patterns