Acoustic Scene and Room Classification for Real-Time Applications

Silvio Emmenegger

Final Presentation, Master Thesis FS20

July 1, 2020

Acoustic environments in daily life

▶ Play

Livingroom/Music
Lucerne, Home

▶ Play

Street/Quiet
Zurich, Bahnhofstr.

▶ Play

Cafeteria/Music+NoisySpeech
Lucerne, Spettacolo

Sources: https://bit.ly/31p4k32, https://bit.ly/3g6284E, https://bit.ly/2AfDwr6

Intention

- Creation of acoustic classifier system for scenes and soundscapes
- Classification at the edge in real-time → hearing aids
- Use of latest AI techniques → Convolutional Neural Networks (CNN)

Intention

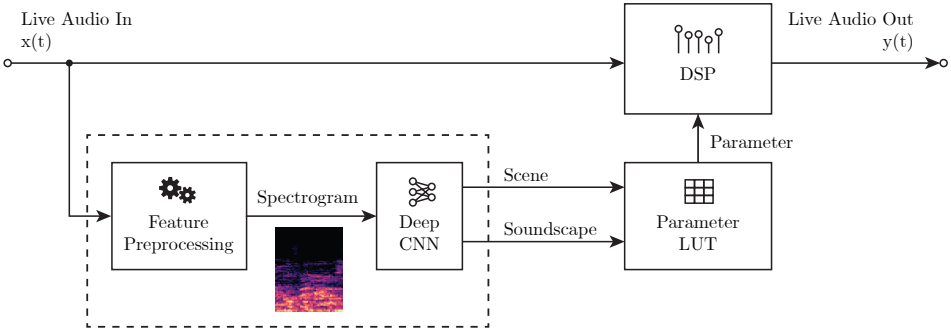
- Creation of acoustic classifier system for scenes and soundscapes
- Classification at the edge in real-time → hearing aids
- Use of latest AI techniques → Convolutional Neural Networks (CNN)

Intention

- Creation of acoustic classifier system for scenes and soundscapes
- Classification at the edge in real-time $\rightarrow$ hearing aids
- Use of latest AI techniques $\rightarrow$ Convolutional Neural Networks (CNN)

Intention

- Creation of acoustic classifier system for scenes and soundscapes
- Classification at the edge in real-time → hearing aids
- Use of latest AI techniques → Convolutional Neural Networks (CNN)



Contents

- Introduction
- Concept & Realization
- Results
- Demonstration
- Questions & Discussion

Real-Time Specifications for Hearing Aids

Parameter		Value	Unit
Inference Interval	Ambitious	1	sec
	Sufficient	5-10	sec
Audio	Sampling Freq.	22.05	kHz
	Quantization	16	bit
CPU	Clock Speed	5	MHz
Memory	Available	unknown	Mbit
	Total	32-48	Mbit
Battery	Capacity	45	mAh
	Voltage	4	V
	Lifetime	3-7	days

Source: https://bit.ly/388zFsc

Tasks

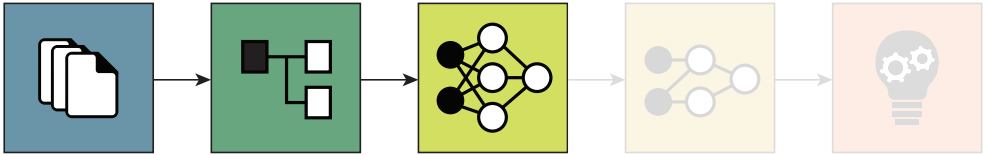
- Record new audio dataset and labels with high-quality equipment
- Preprocessing of audio into image-like features
- Train CNN model on created feature dataset
- Optimize CNN model with simple grid search method
- Propose real-time implementation concept for dedicated hardware

Tasks

- Record new audio dataset and labels with high-quality equipment
- Preprocessing of audio into image-like features
- Train CNN model on created feature dataset
- Optimize CNN model with simple grid search method
- Propose real-time implementation concept for dedicated hardware

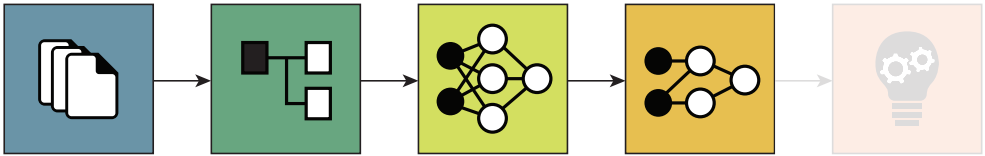
Tasks

- Record new audio dataset and labels with high-quality equipment
- Preprocessing of audio into image-like features
- Train CNN model on created feature dataset
- Optimize CNN model with simple grid search method
- Propose real-time implementation concept for dedicated hardware



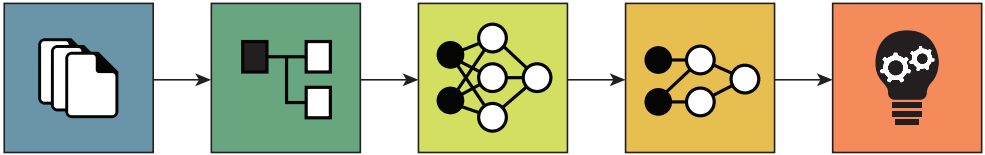
Tasks

- Record new audio dataset and labels with high-quality equipment
- Preprocessing of audio into image-like features
- Train CNN model on created feature dataset
- Optimize CNN model with simple grid search method
- Propose real-time implementation concept for dedicated hardware



Tasks

- Record new audio dataset and labels with high-quality equipment
- Preprocessing of audio into image-like features
- Train CNN model on created feature dataset
- Optimize CNN model with simple grid search method
- Propose real-time implementation concept for dedicated hardware



Dataset Creation

- Two labels with five classes each („2D“ label combinations)
- Multi-class multi-output classification problem, supervised
- Binaural recordings with $f_s = 48$ kHz and 16-bit quantization learning
- At least 60 min for each label combination $\rightarrow$ 25h in total

Scenes	Soundscapes
Cafeteria/Canteen	Music
Car (inside)	Noise
Livingroom	Speech
Nature	Noisy Speech
Street	Quiet

Dataset Creation

- Two labels with five classes each („2D“ label combinations)
- Multi-class multi-output classification problem, supervised
- Binaural recordings with $f_s = 48$ kHz and 16-bit quantization learning
- At least 60 min for each label combination $\rightarrow$ 25h in total

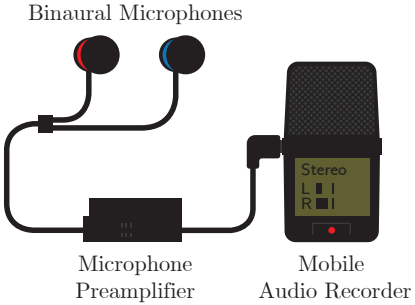
Scenes	Soundscapes
Cafeteria/Canteen	Music
Car (inside)	Noise
Livingroom	Speech
Nature	Noisy Speech
Street	Quiet

Dataset Creation

- Two labels with five classes each („2D“ label combinations)
- Multi-class multi-output classification problem, supervised
- Binaural recordings with $f_s = 48$ kHz and 16-bit quantization learning
- At least 60 min for each label combination $\rightarrow$ 25h in total

Scenes
Cafeteria/Canteen
Car (inside)
Livingroom
Nature
Street

Soundscapes
Music
Noise
Speech
Noisy Speech
Quiet

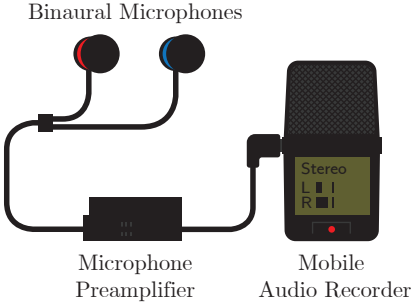


Dataset Creation

- Two labels with five classes each („2D“ label combinations)
- Multi-class multi-output classification problem, supervised
- Binaural recordings with $f_s = 48$ kHz and 16-bit quantization learning
- At least 60 min for each label combination $\rightarrow$ 25h in total

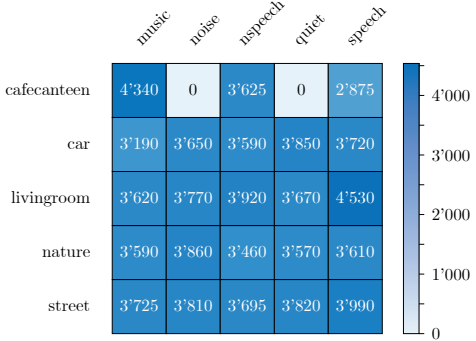
Scenes
Cafeteria/Canteen
Car (inside)
Livingroom
Nature
Street

Soundscapes
Music
Noise
Speech
Noisy Speech
Quiet



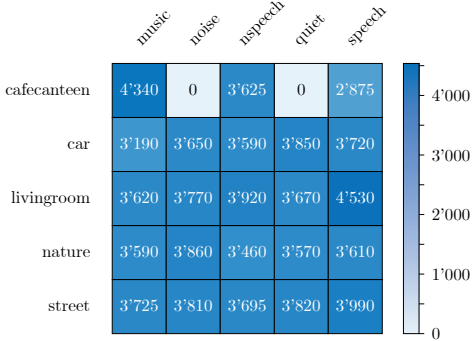
Resulting Dataset

- Total length of 23.8 hours → chunking to 1 sec
- No `Quiet` and `Noise` soundscapes recorded in `Cafeteria/Canteen`
- Soundscapes: live sources (50%), speaker replay (50%)
- Training: 70%, Testing: 30%, 5-fold cross-validation



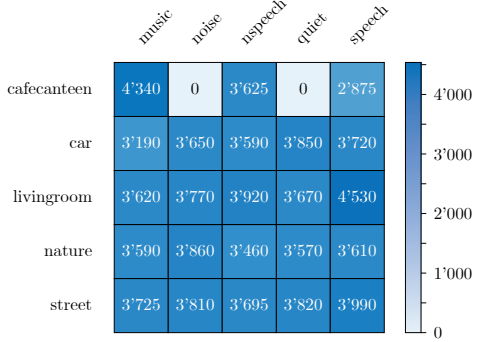
Resulting Dataset

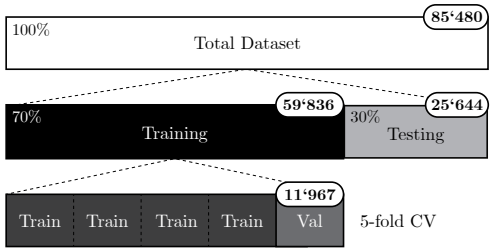
- Total length of 23.8 hours → chunking to 1 sec
- No `Quiet` and `Noise` soundscapes recorded in `Cafeteria/Canteen`
- Soundscapes: live sources (50%), speaker replay (50%)
- Training: 70%, Testing: 30%, 5-fold cross-validation



Resulting Dataset

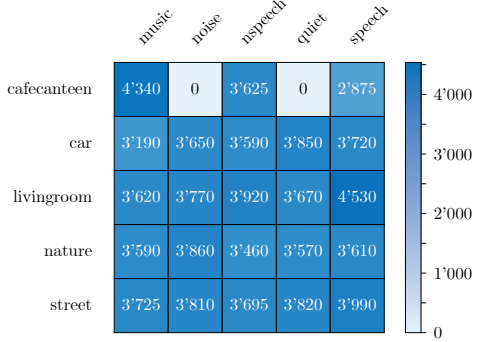
- Total length of 23.8 hours → chunking to 1 sec
- No `Quiet` and `Noise` soundscapes recorded in `Cafeteria/Canteen`
- Soundscapes: live sources (50%), speaker replay (50%)
- Training: 70%, Testing: 30%, 5-fold cross-validation

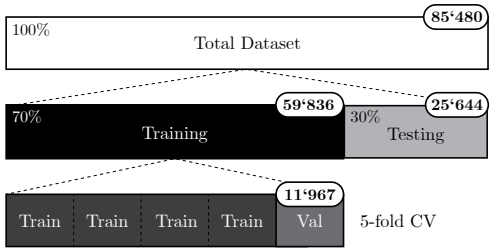




Resulting Dataset

- Total length of 23.8 hours → chunking to 1 sec
- No `Quiet` and `Noise` soundscapes recorded in `Cafeteria/Canteen`
- Soundscapes: live sources (50%), speaker replay (50%)
- Training: 70%, Testing: 30%, 5-fold cross-validation



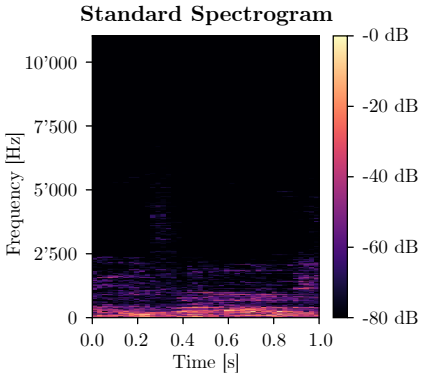


Feature Preprocessing

- Convert audio chunks into frequency domain
- Short-Time Fourier Transform (STFT) → standard spectrogram
- Logarithmic compression of frequency axis → Mel spectrogram

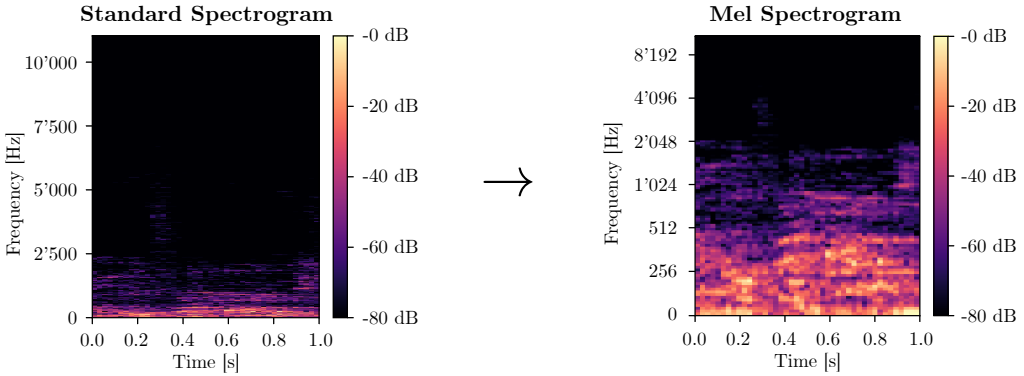
Feature Preprocessing

- Convert audio chunks into frequency domain
- Short-Time Fourier Transform (STFT) → standard spectrogram
- Logarithmic compression of frequency axis → Mel spectrogram



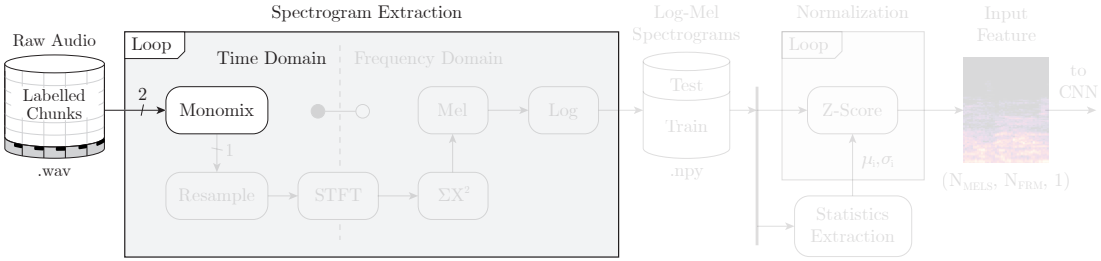
Feature Preprocessing

- Convert audio chunks into frequency domain
- Short-Time Fourier Transform (STFT) → standard spectrogram
- Logarithmic compression of frequency axis → Mel spectrogram



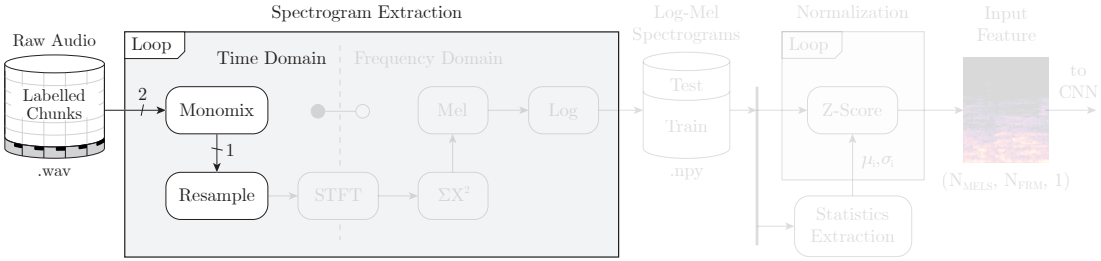
Total Preprocessing Chain

- Convert binaural signal to mono (Monomix)
- Resampling to target frequency
- Z-Score normalization for each frequency band
- Normalized Log-Mel power spectrograms as CNN input



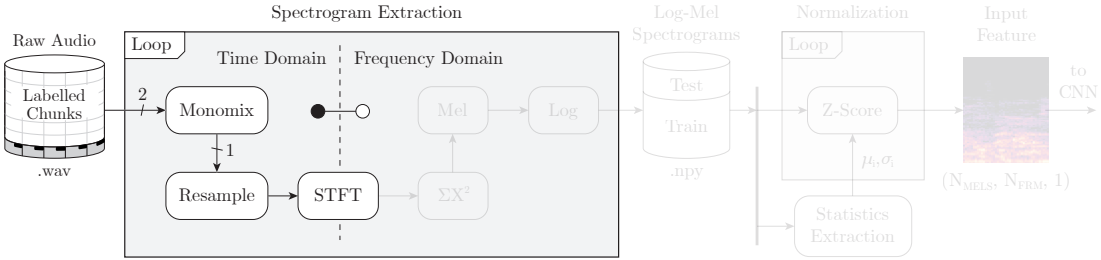
Total Preprocessing Chain

- Convert binaural signal to mono (Monomix)
- Resampling to target frequency
- Z-Score normalization for each frequency band
- Normalized Log-Mel power spectrograms as CNN input



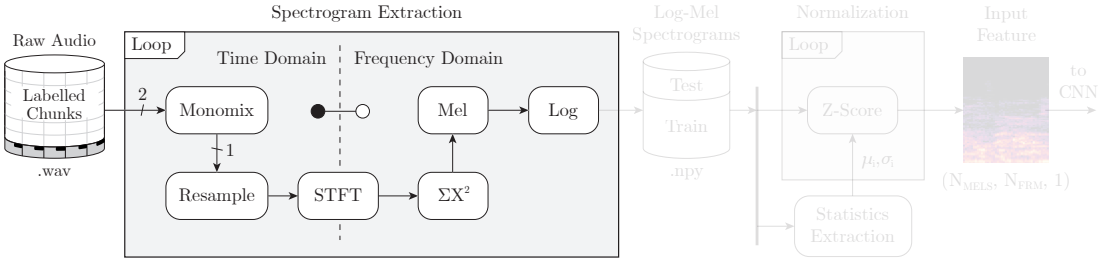
Total Preprocessing Chain

- Convert binaural signal to mono (Monomix)
- Resampling to target frequency
- Z-Score normalization for each frequency band
- Normalized Log-Mel power spectrograms as CNN input



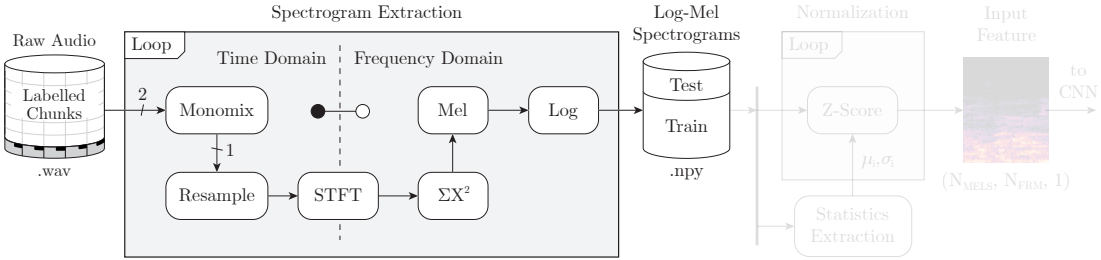
Total Preprocessing Chain

- Convert binaural signal to mono (Monomix)
- Resampling to target frequency
- Z-Score normalization for each frequency band
- Normalized Log-Mel power spectrograms as CNN input



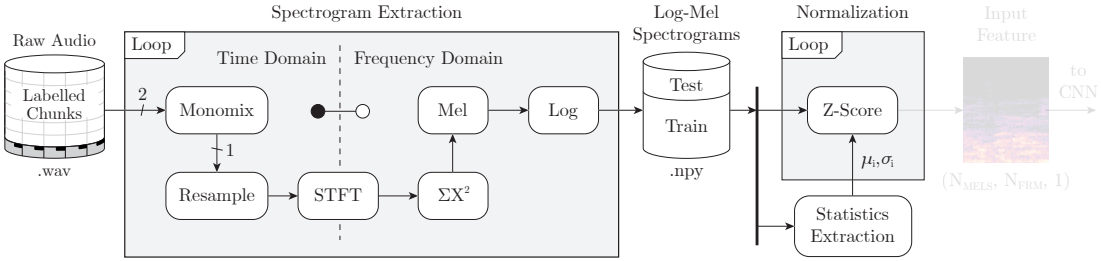
Total Preprocessing Chain

- Convert binaural signal to mono (Monomix)
- Resampling to target frequency
- Z-Score normalization for each frequency band
- Normalized Log-Mel power spectrograms as CNN input



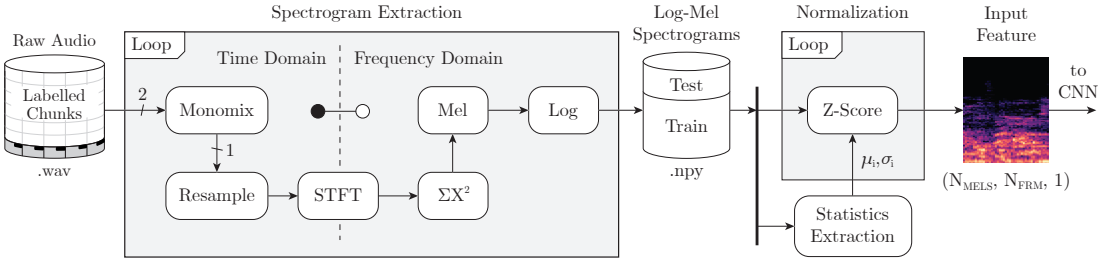
Total Preprocessing Chain

- Convert binaural signal to mono (Monomix)
- Resampling to target frequency
- Z-Score normalization for each frequency band
- Normalized Log-Mel power spectrograms as CNN input



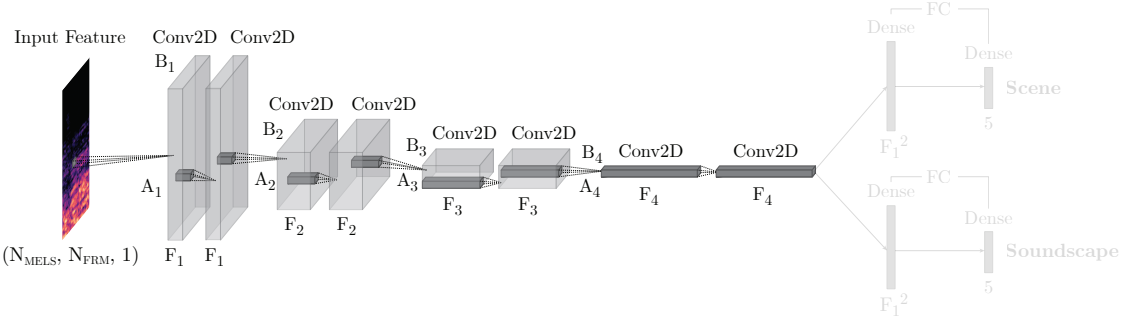
Total Preprocessing Chain

- Convert binaural signal to mono (Monomix)
- Resampling to target frequency
- Z-Score normalization for each frequency band
- Normalized Log-Mel power spectrograms as CNN input



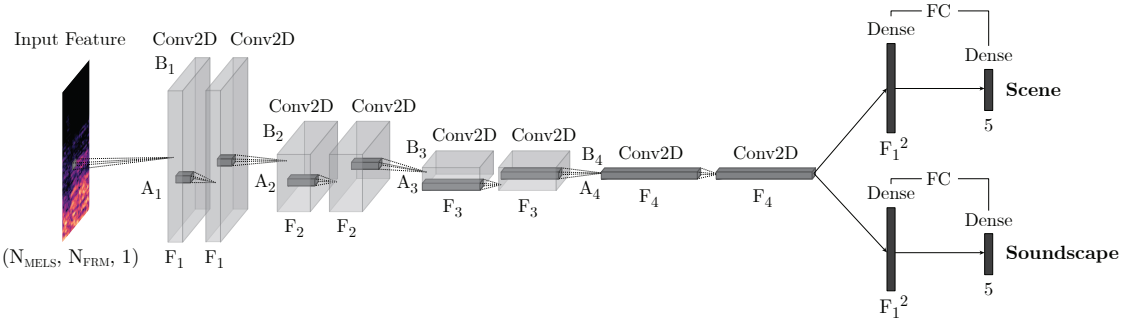
CNN Architecture

- Inspired by VGGNet-16 (image classification)
- Modification 1: Two fully-connected (FC) outputs share feature maps
- Modification 2: Sigmoid output activation for multi-label classification
- Modification 3: Dynamic shape of model/activation maps (A_i, B_i, F_i)



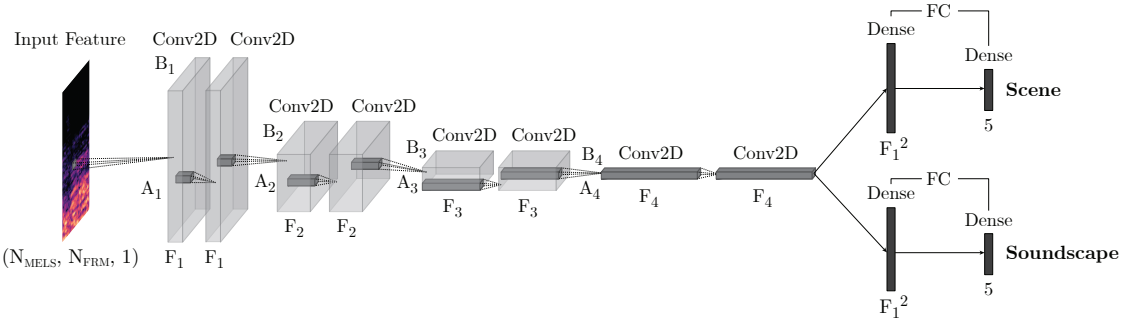
CNN Architecture

- Inspired by VGGNet-16 (image classification)
- Modification 1: Two fully-connected (FC) outputs share feature maps
- Modification 2: Sigmoid output activation for multi-label classification
- Modification 3: Dynamic shape of model/activation maps (A_i, B_i, F_i)



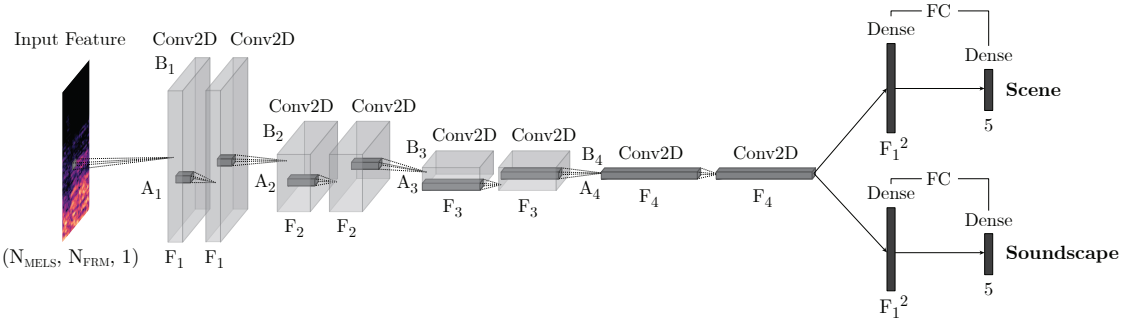
CNN Architecture

- Inspired by VGGNet-16 (image classification)
- Modification 1: Two fully-connected (FC) outputs share feature maps
- Modification 2: Sigmoid output activation for multi-label classification
- Modification 3: Dynamic shape of model/activation maps (A_i, B_i, F_i)



CNN Architecture

- Inspired by VGGNet-16 (image classification)
- Modification 1: Two fully-connected (FC) outputs share feature maps
- Modification 2: Sigmoid output activation for multi-label classification
- Modification 3: Dynamic shape of model/activation maps (A_i, B_i, F_i)

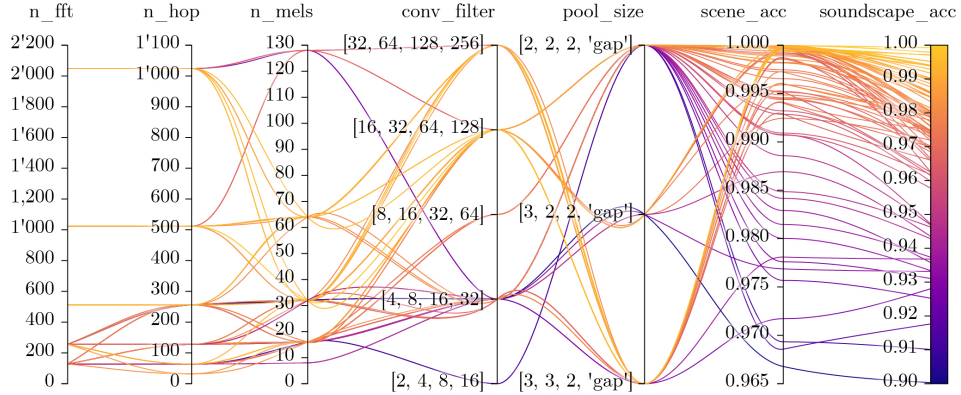


Training metrics of model candidates

- Iterative training of ≈ 100 models
- Parallel coordinates plot helped to evaluate model candidates
- Conclusion: soundscape always below scene training accuracy

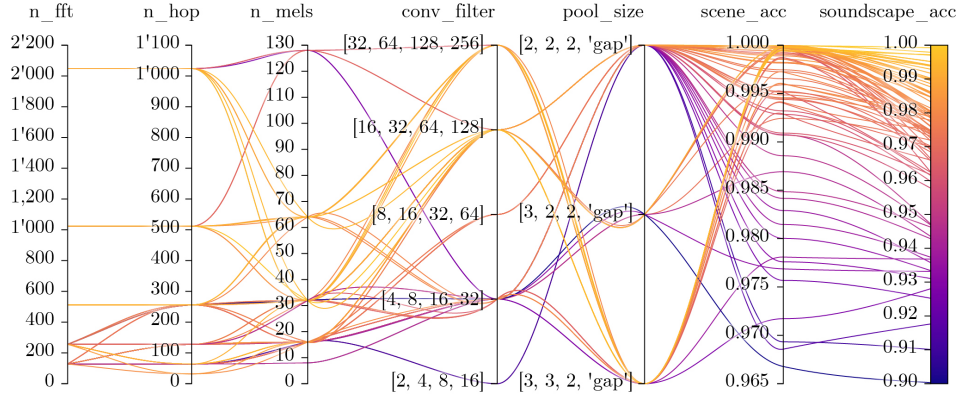
Training metrics of model candidates

- Iterative training of $\approx$ 100 models
- Parallel coordinates plot helped to evaluate model candidates
- Conclusion: soundscape always below scene training accuracy



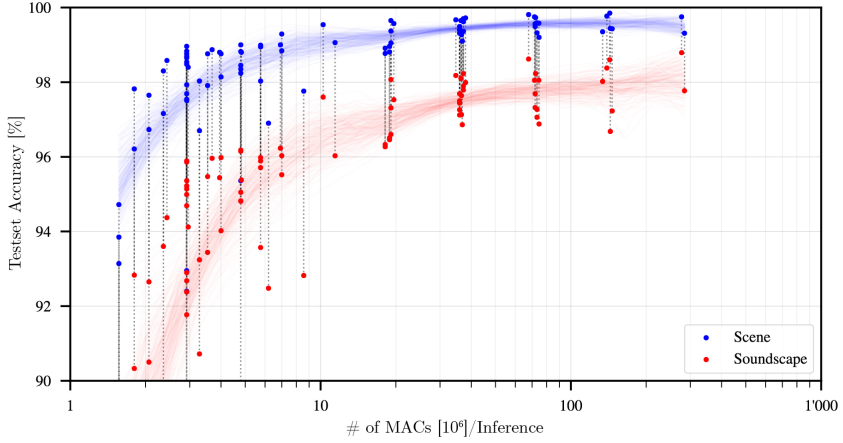
Training metrics of model candidates

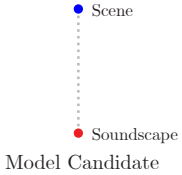
- Iterative training of $\approx$ 100 models
- Parallel coordinates plot helped to evaluate model candidates
- Conclusion: soundscape always below scene training accuracy



Inference Complexity

- Increasing trend of testset accuracies for models with more MACs[1]
- Model studies: selection of the three best models of each decade

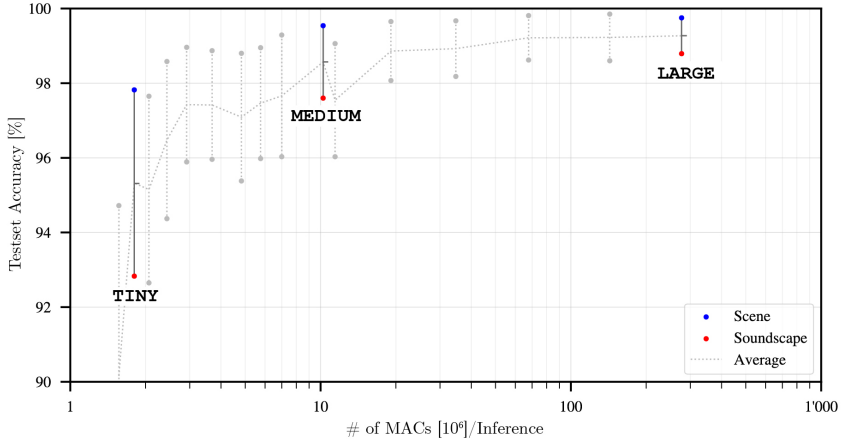


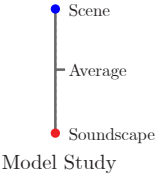


[1] Multiply-accumulate operations

Inference Complexity

- Increasing trend of testset accuracies for models with more MACs[1]
- Model studies: selection of the three best models of each decade

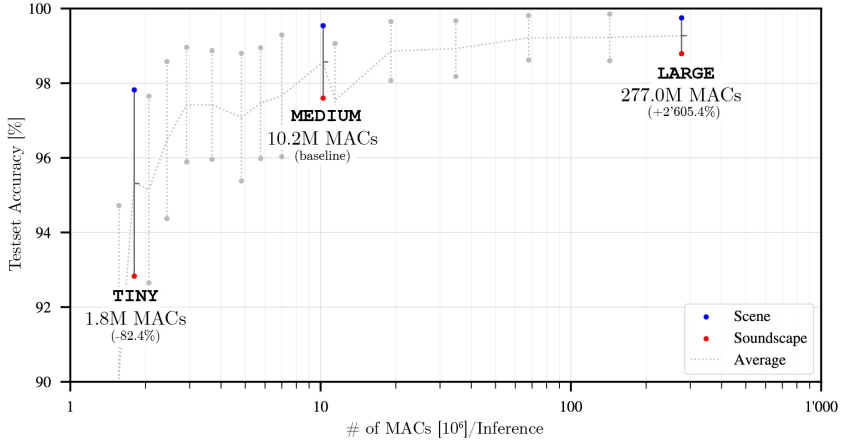


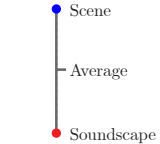


[1] Multiply-accumulate operations

Inference Complexity

- Increasing trend of testset accuracies for models with more MACs[1]
- Model studies: selection of the three best models of each decade



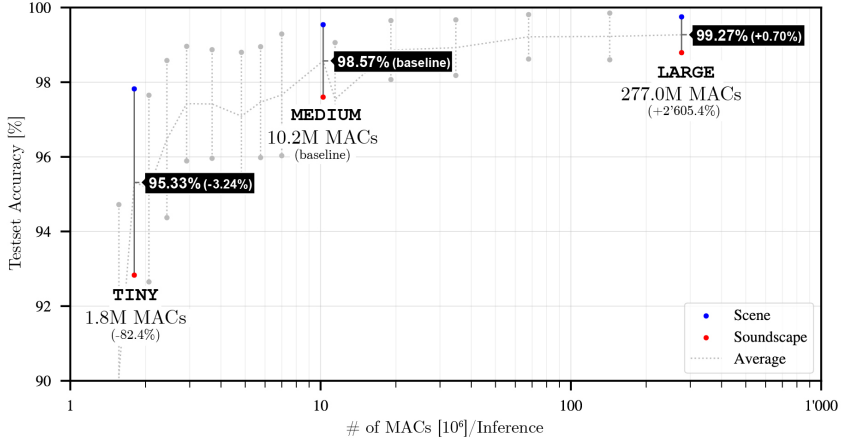


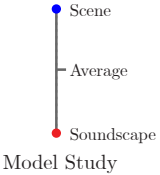
Model Study

[1] Multiply-accumulate operations

Inference Complexity

- Increasing trend of testset accuracies for models with more MACs[1]
- Model studies: selection of the three best models of each decade

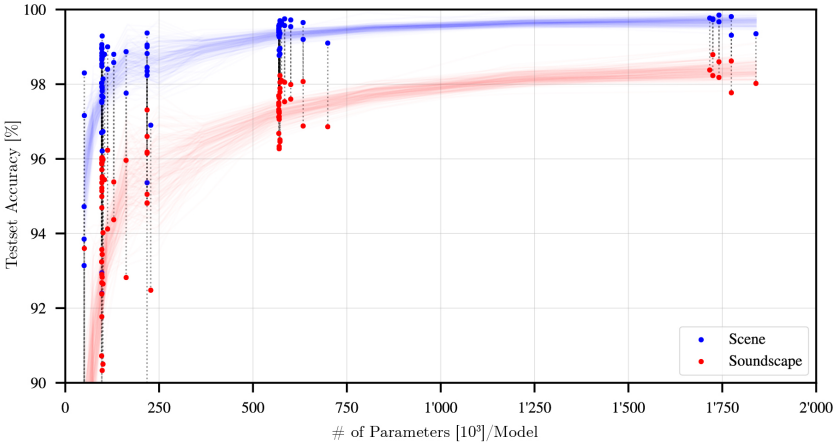


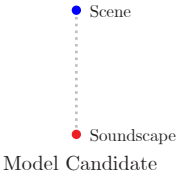


[1] Multiply-accumulate operations

Memory Complexity

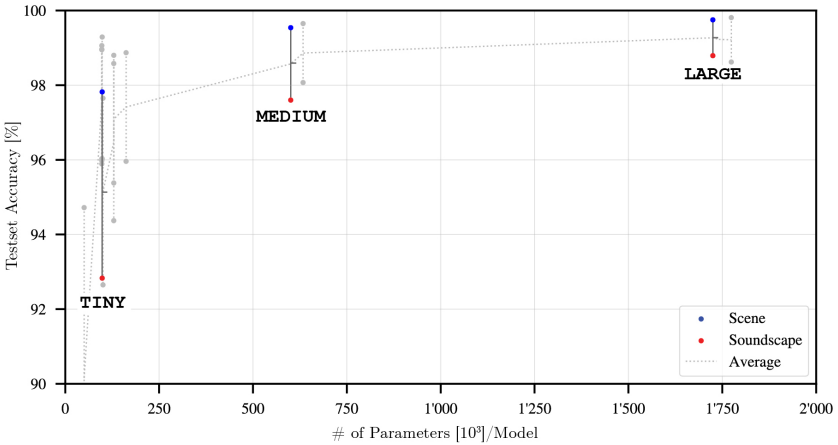
- Same trend: more parameters refer to higher testset accuracies
- Model studies: number of parameters scale on a linear basis

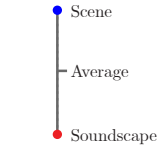




Memory Complexity

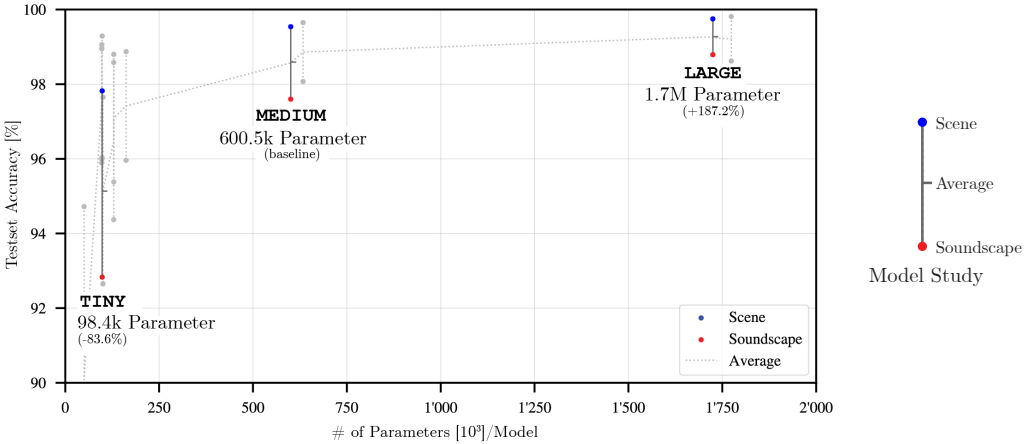
- Same trend: more parameters refer to higher testset accuracies
- Model studies: number of parameters scale on a linear basis





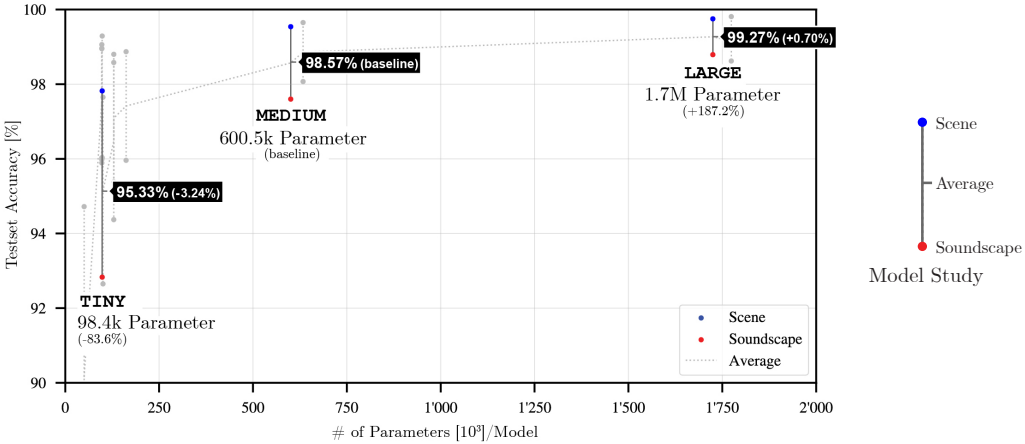
Memory Complexity

- Same trend: more parameters refer to higher testset accuracies
- Model studies: number of parameters scale on a linear basis



Memory Complexity

- Same trend: more parameters refer to higher testset accuracies
- Model studies: number of parameters scale on a linear basis



Implementation Concept

- Post-quantization of `float32` model into `int8` model
- Library `TFLite` provides functions for porting model to *C* language
- Approximation of `float32` values by rescaling and shifting:

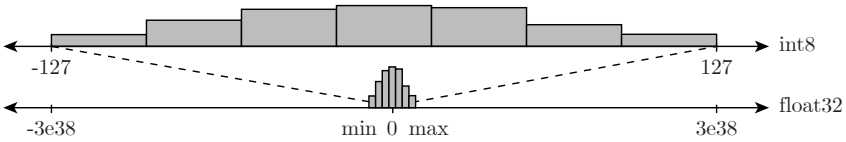
Implementation Concept

- Post-quantization of `float32` model into `int8` model
- Library `TFLite` provides functions for porting model to *C* language
- Approximation of `float32` values by rescaling and shifting:

Implementation Concept

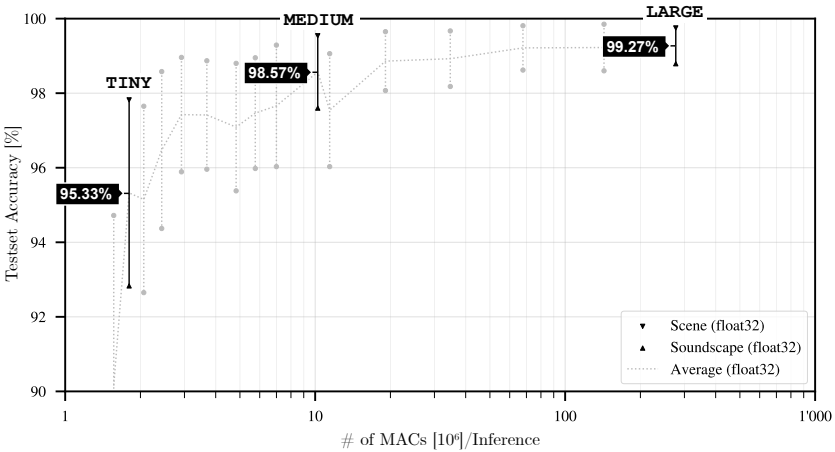
- Post-quantization of `float32` model into `int8` model
- Library `TFLite` provides functions for porting model to *C* language
- Approximation of `float32` values by rescaling and shifting:

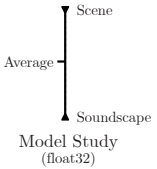
$$real\_value = (int8\_value - zero\_point) \times scale$$



Post-quantization

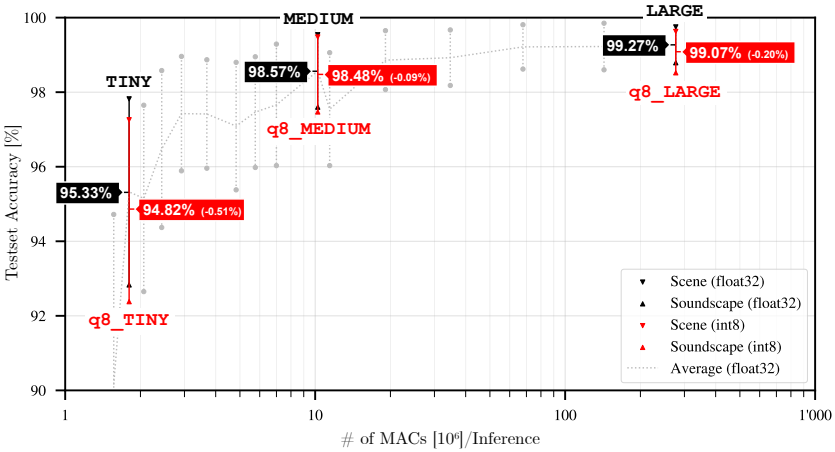
- 8-bit quantization applied to all three model studies
- Accuracy loss between [-0.56%, -0.06%] in worst/best case (label-avg)

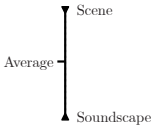


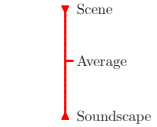


Post-quantization

- 8-bit quantization applied to all three model studies
- Accuracy loss between [-0.56%, -0.06%] in worst/best case (label-avg)

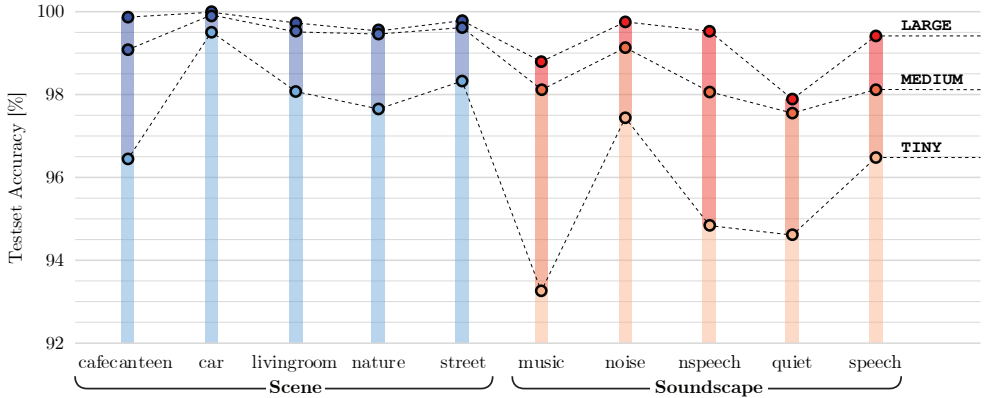






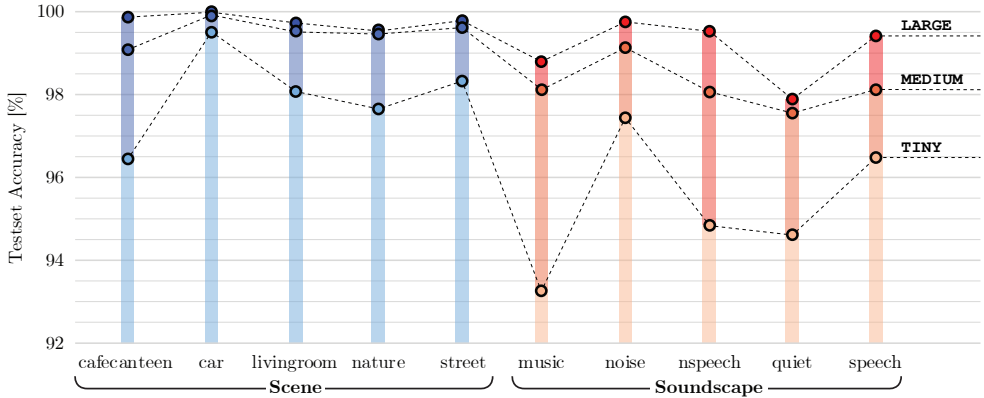
Class-wise Results

- Larger models always outperform their smaller neighbour model
- car and noise are not difficult to detect
- music is rather difficult to detect for TINY



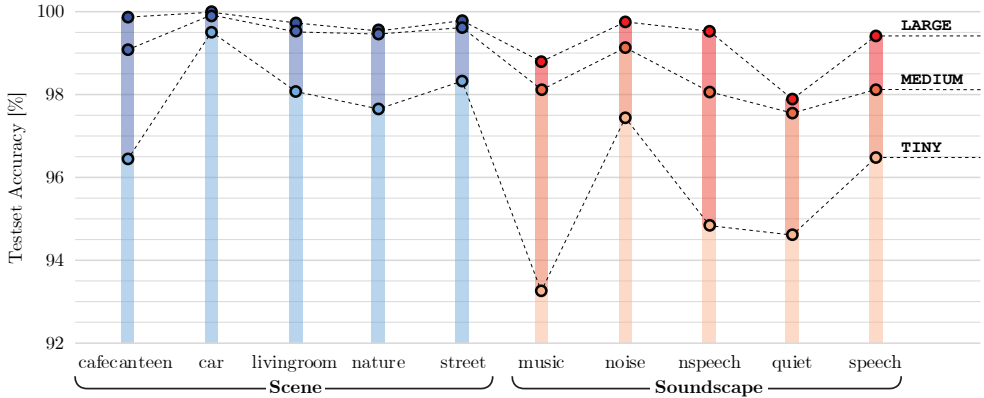
Class-wise Results

- Larger models always outperform their smaller neighbour model
- `car` and `noise` are not difficult to detect
- `music` is rather difficult to detect for `TINY`



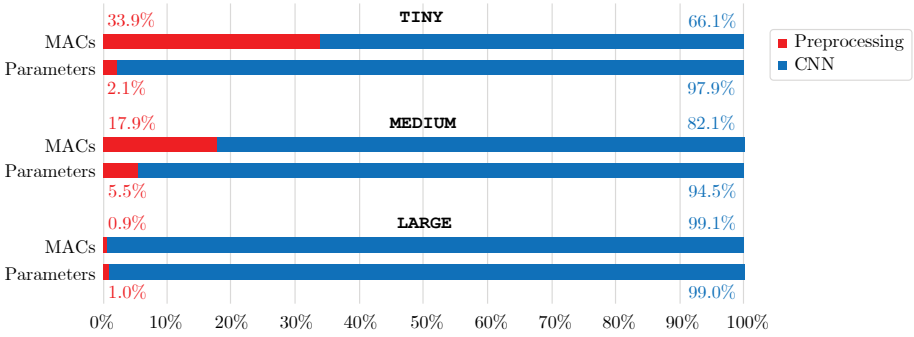
Class-wise Results

- Larger models always outperform their smaller neighbour model
- `car` and `noise` are not difficult to detect
- `music` is rather difficult to detect for `TINY`



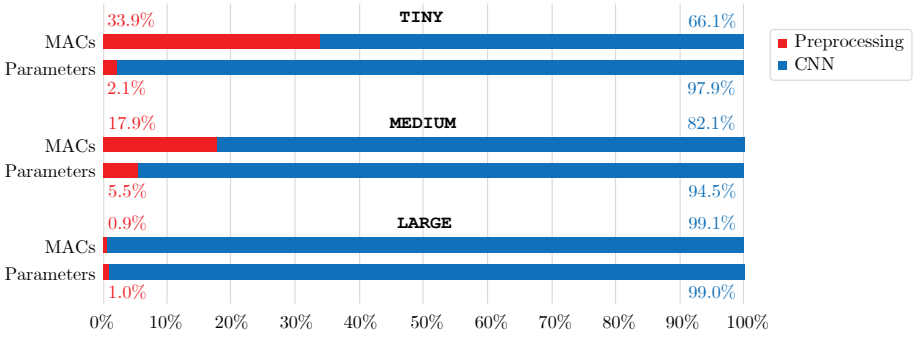
Partial Complexity Analysis

- CNN model require the most computational effort
- Largest relative part of preprocessing parameters is needed by `MEDIUM`
- Preprocessing does not grow linearly to CNN size



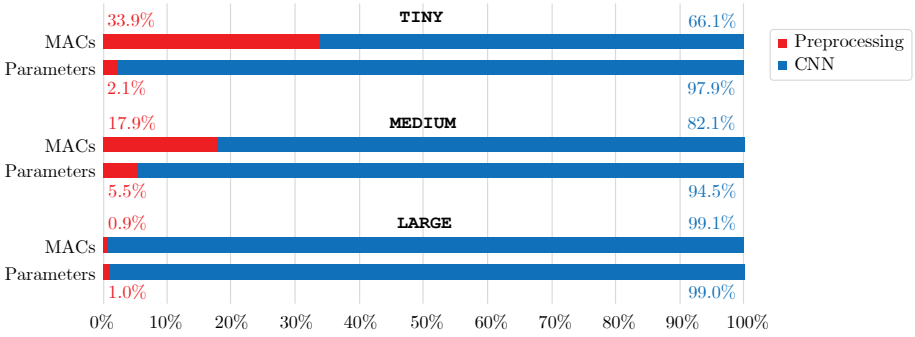
Partial Complexity Analysis

- CNN model require the most computational effort
- Largest relative part of preprocessing parameters is needed by `MEDIUM`
- Preprocessing does not grow linearly to CNN size



Partial Complexity Analysis

- CNN model require the most computational effort
- Largest relative part of preprocessing parameters is needed by `MEDIUM`
- Preprocessing does not grow linearly to CNN size



Inference Performance

Study	Required Memory	Clock Speed (min)
`TINY`	0.8 MBit	7.2 MHz
`MEDIUM`	4.8 MBit	41.0 MHz
`LARGE`	13.8 MBit	1.1 GHz

- Security margin of factor 2x for clock speed calculations
- Final conclusions:
  - `MEDIUM`: realizable compromise between accuracy and throughput
  - `TINY`: saving of $\approx$ 80% computational effort result in $\approx$ 3% loss of accuracy
  - `LARGE`: investment of $\approx$ 25x more computational effort result in $\approx$ 1% gain of accuracy
  - Free memory on hearing aids unknown (32-48 MBit total), but model sizes are feasible
  - Increase of clock speed on hearing aids at least by factor 2x: 5 MHz $\rightarrow$ 10 MHz (`TINY`)

Inference Performance

Study	Required Memory	Clock Speed (min)
`TINY`	0.8 MBit	7.2 MHz
`MEDIUM`	4.8 MBit	41.0 MHz
`LARGE`	13.8 MBit	1.1 GHz

- Security margin of factor 2x for clock speed calculations
- Final conclusions:
  - `MEDIUM`: realizable compromise between accuracy and throughput
  - `TINY`: saving of $\approx$ 80% computational effort result in $\approx$ 3% loss of accuracy
  - `LARGE`: investment of $\approx$ 25x more computational effort result in $\approx$ 1% gain of accuracy
  - Free memory on hearing aids unknown (32-48 MBit total), but model sizes are feasible
  - Increase of clock speed on hearing aids at least by factor 2x: 5 MHz $\rightarrow$ 10 MHz (`TINY`)

Inference Performance

Study	Required Memory	Clock Speed (min)
`TINY`	0.8 MBit	7.2 MHz
`MEDIUM`	4.8 MBit	41.0 MHz
`LARGE`	13.8 MBit	1.1 GHz

- Security margin of factor 2x for clock speed calculations
- Final conclusions:
  - `MEDIUM`: realizable compromise between accuracy and throughput
  - `TINY`: saving of $\approx$ 80% computational effort result in $\approx$ 3% loss of accuracy
  - `LARGE`: investment of $\approx$ 25x more computational effort result in $\approx$ 1% gain of accuracy
  - Free memory on hearing aids unknown (32-48 MBit total), but model sizes are feasible
  - Increase of clock speed on hearing aids at least by factor 2x: 5 MHz $\rightarrow$ 10 MHz (`TINY`)

Inference Performance

Study	Required Memory	Clock Speed (min)
`TINY`	0.8 MBit	7.2 MHz
`MEDIUM`	4.8 MBit	41.0 MHz
`LARGE`	13.8 MBit	1.1 GHz

- Security margin of factor 2x for clock speed calculations
- Final conclusions:
  - `MEDIUM`: realizable compromise between accuracy and throughput
  - `TINY`: saving of $\approx$ 80% computational effort result in $\approx$ 3% loss of accuracy
  - `LARGE`: investment of $\approx$ 25x more computational effort result in $\approx$ 1% gain of accuracy
  - Free memory on hearing aids unknown (32-48 MBit total), but model sizes are feasible
  - Increase of clock speed on hearing aids at least by factor 2x: 5 MHz $\rightarrow$ 10 MHz (`TINY`)

Inference Performance

Study	Required Memory	Clock Speed (min)
`TINY`	0.8 MBit	7.2 MHz
`MEDIUM`	4.8 MBit	41.0 MHz
`LARGE`	13.8 MBit	1.1 GHz

- Security margin of factor 2x for clock speed calculations
- Final conclusions:
  - `MEDIUM`: realizable compromise between accuracy and throughput
  - `TINY`: saving of $\approx$ 80% computational effort result in $\approx$ 3% loss of accuracy
  - `LARGE`: investment of $\approx$ 25x more computational effort result in $\approx$ 1% gain of accuracy
  - Free memory on hearing aids unknown (32-48 MBit total), but model sizes are feasible
  - Increase of clock speed on hearing aids at least by factor 2x: 5 MHz $\rightarrow$ 10 MHz (`TINY`)

Inference Performance

Study	Required Memory	Clock Speed (min)
`TINY`	0.8 MBit	7.2 MHz
`MEDIUM`	4.8 MBit	41.0 MHz
`LARGE`	13.8 MBit	1.1 GHz

- Security margin of factor 2x for clock speed calculations
- Final conclusions:
  - `MEDIUM`: realizable compromise between accuracy and throughput
  - `TINY`: saving of $\approx$ 80% computational effort result in $\approx$ 3% loss of accuracy
  - `LARGE`: investment of $\approx$ 25x more computational effort result in $\approx$ 1% gain of accuracy
  - Free memory on hearing aids unknown (32-48 MBit total), but model sizes are feasible
  - Increase of clock speed on hearing aids at least by factor 2x: 5 MHz $\rightarrow$ 10 MHz (`TINY`)

Inference Performance

Study	Required Memory	Clock Speed (min)
`TINY`	0.8 MBit	7.2 MHz
`MEDIUM`	4.8 MBit	41.0 MHz
`LARGE`	13.8 MBit	1.1 GHz

- Security margin of factor 2x for clock speed calculations
- Final conclusions:
  - `MEDIUM`: realizable compromise between accuracy and throughput
  - `TINY`: saving of $\approx$ 80% computational effort result in $\approx$ 3% loss of accuracy
  - `LARGE`: investment of $\approx$ 25x more computational effort result in $\approx$ 1% gain of accuracy
  - Free memory on hearing aids unknown (32-48 MBit total), but model sizes are feasible
  - Increase of clock speed on hearing aids at least by factor 2x: 5 MHz $\rightarrow$ 10 MHz (`TINY`)

Outlook

- Optimization: evolutionary algorithm, quantization-aware training, automated sweeps
- Dataset: more labels/classes, new recordings with hearing aid (microphone characteristic)
- Implementation: portation of model to hearing aid, energy estimations for single inference

Outlook

- Optimization: evolutionary algorithm, quantization-aware training, automated sweeps
- Dataset: more labels/classes, new recordings with hearing aid (microphone characteristic)
- Implementation: portation of model to hearing aid, energy estimations for single inference

Outlook

- Optimization: evolutionary algorithm, quantization-aware training, automated sweeps
- Dataset: more labels/classes, new recordings with hearing aid (microphone characteristic)
- Implementation: portation of model to hearing aid, energy estimations for single inference

Demonstrator

Questions & Discussion

Bibliography

[1] Fischer, M. (2020). BinArray: A Flexible Hardware Architecture for CNNs with Binary Encoded Weights. *Master of Science in Engineering, Master Thesis, HS19*. University of Applied Sciences and Arts, Feburary 2020.

[2] Johner, F. (2019). Efficient Evolutionary Architecture Search for CNN Optimization on GTSRB. *University of Applied Sciences and Arts*. https://ieeexplore.ieee.org/abstract/document/8999305.

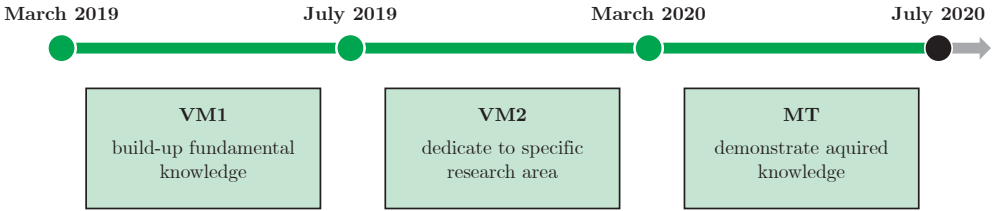
[3] Emmenegger, S. (2019). Acoustic Scene Classification with Neural Networks. *Master of Science in Engineering, VM1, FS19*. University of Applied Sciences and Arts, June 2019.

[4] Emmenegger, S. (2020). Classification of Acoustic Room Properties from Speech Samples. *Master of Science in Engineering, VM2, HS19*. University of Applied Sciences and Arts, Jan 2020.

Previous Work

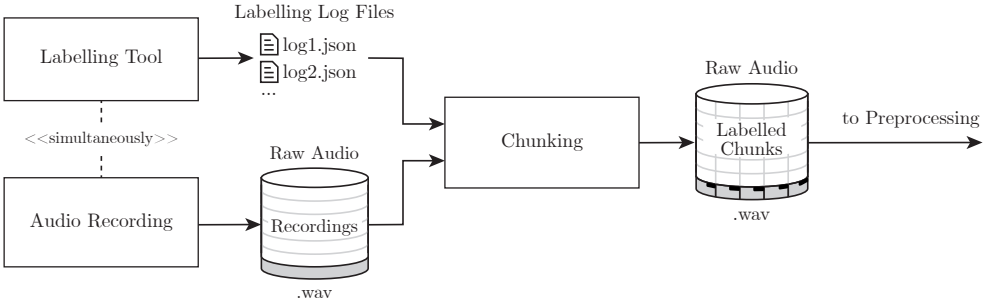
- VM1: Classification of ten different acoustic scenes (accuracy: 76.9%) [3]
- VM2: Classification of six artificial rooms with speech samples (accuracy: 99.7%) [4]

Project Scope



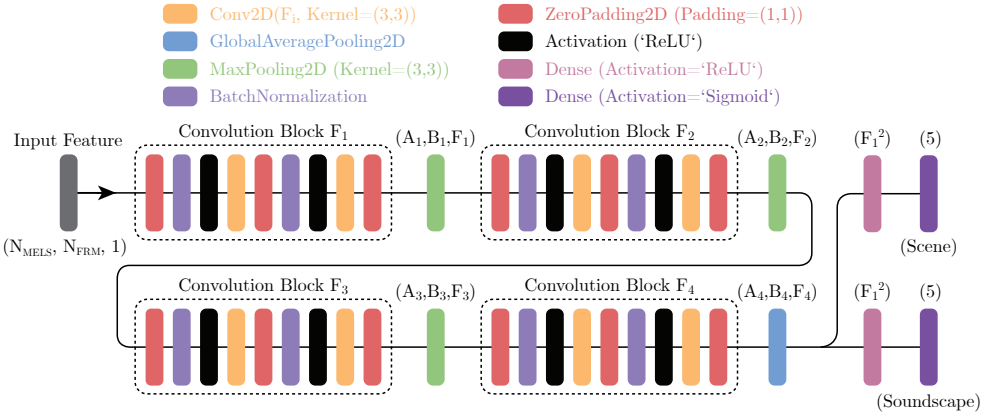
Labelling and Chunking

- Labelling tool as web application → .json files
- Start of recording and labelling simultaneously (timestamp synchronization)
- Chunking to 1 sec length



Detailed CNN Architecture

- Implemented in `Python` using `TensorFlow+Keras`
- Different sizes used for pooling P and convolutional filters F_i across model candidates



CNN Optimization

- Model candidates: grid search method using a subset of optimization parameters
- Optimization of total system: Preprocessing + CNN
- Objective metrics:
  - Memory complexity: number of parameters (PAR)
  - Inference complexity: number of multiply-accumulate operations (MAC)
- Model studies: selection of three optimized classifiers with different complexities

Name	PAR	MAC	Testset Accuracy
`TINY`	-50%	-90%	-5%
`MEDIUM`	(baseline)	(baseline)	(baseline)
`LARGE`	+100%	+1000%	+5%

CNN Optimization (grid search)

- Excel tool created which generates configuration files automatically
- Optimization parameters chosen by personal experience:

Key	Value Set
`n_fft`	{128, 256, 512, 1'024}
`n_hop`	{32, 64, 128, 256, 512, 1'024}
`n_mels`	{8, 16, 32, 64, 128}
`conv_filt`	{2, 4, 8, 16, 32}
`pool_size`	{{{2,2},{2,2},{2,2}}, {{3,3},{2,2},{2,2}}, {{3,3},{3,3},{2,2}}}

Available Implementation Concepts

- Hardware accelerated approach:
  - Binary Approximated CNN (BACNN), Fischer et al. [1]
  - FPGA-based architecture
  - High throughput: 5.7M input pixels/sec (820 inferences/sec)
- Software-only approach: fixed-point implementation with quantized model

Chosen Implementation Concept

→ Software-only implementation

- Reasons:
  - Audio requires only low throughput: 22'050 pixels/sec in worst case (1 inference/sec)
  - Integration in existing hearing aid software is easier
- Post-quantization of CNN model inside this work

Class-wise Results Detailed

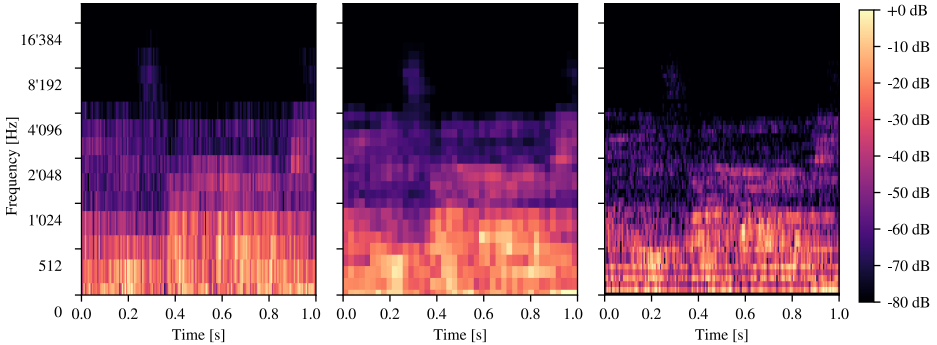
Scene	TINY Cherry-08	MEDIUM Lemon-34	LARGE Lemon-20
cafecanteen	96.42%	99.09%	99.88%
car	99.52%	99.94%	99.98%
livingroom	98.06%	99.54%	99.73%
nature	97.65%	99.47%	99.54%
street	98.33%	99.63%	99.80%
Average	97.82%	99.54%	99.75%

Soundscape	TINY Cherry-08	MEDIUM Lemon-34	LARGE Lemon-20
music	93.23%	98.14%	98.77%
noise	97.42%	99.14%	99.78%
nspeech	94.80%	98.06%	99.52%
quiet	94.62%	97.56%	97.87%
speech	96.48%	98.09%	99.40%
Average	92.83%	97.60%	98.79%

Study	Model	Testset Accuracy (avg)	
TINY	Cherry-08	95.33%	(-3.24%)
MEDIUM	Lemon-34	98.57%	(baseline)
LARGE	Lemon-20	99.27%	(+0.70%)

Log-Mel Input Features

- Compromise between spectral and temporal resolution
- Time-pooling in `TINY` model (no window overlap)



`TINY` model	`MEDIUM` model	`LARGE` model
$16 \times 87 \times 1$	$32 \times 22 \times 1$	$64 \times 87 \times 1$
(1'392 pixels)	(704 pixels)	(5'568 pixels)

Detailed Result Metrics

Study	PAR		MAC		Testset Accuracy (avg)	
TINY	98.4k	(-83.6%)	1.8M	(-82.4%)	95.33%	(-3.24%)
MEDIUM	600.5k	(baseline)	10.2M	(baseline)	98.57%	(baseline)
LARGE	1'724.8k	(+187.2%)	277.0M	(+2'605.4%)	99.27%	(+0.70%)