Training General-Purpose Audio Tagging Networks with Noisy Labels and Iterative Self-Verification

DCASE-2018 Challange Task 2

Matthias Dorfer and Gerhard Widmer
Institute of Computational Perception

JOHANNES KEPLER UNIVERSITY LINZ

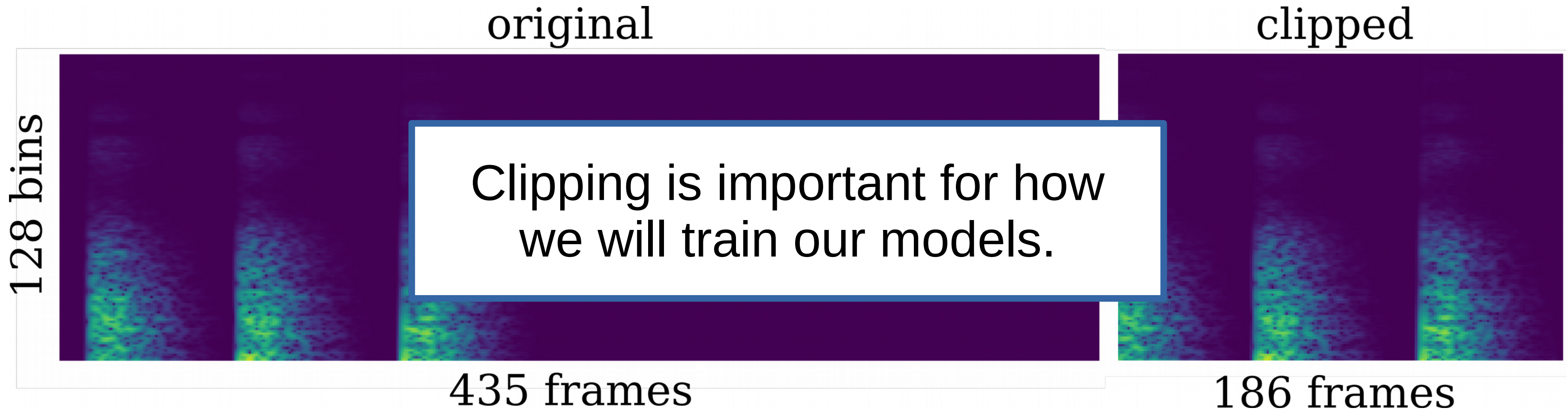
Institute of Computational Perception

Audio Signal Pre-Processing

- Normalize to a dB-level of -0.1
- Clip silence in beginning and end of signal
- re-sampled to 32 kHz

Audio Signal Pre-Processing

- Normalize to a dB-level of -0.1
- Clip silence in beginning and end of signal
- re-sampled to 32 kHz



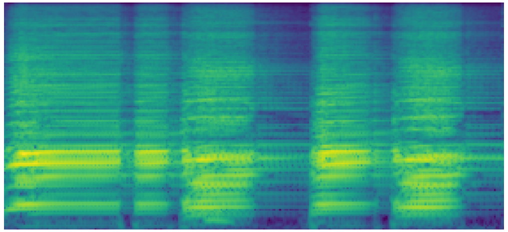
Clipping is important for how we will train our models.

Spectrogram Parameters

Two spectrogram types to capture different aspects of the audio

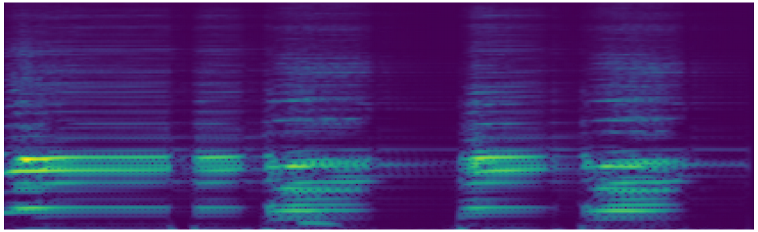
Version – 1

- STFT hop-size: 192
- 1024-sample hann windows
- Perceptual weighting
- Mel-scaled filterbank (128 bins)

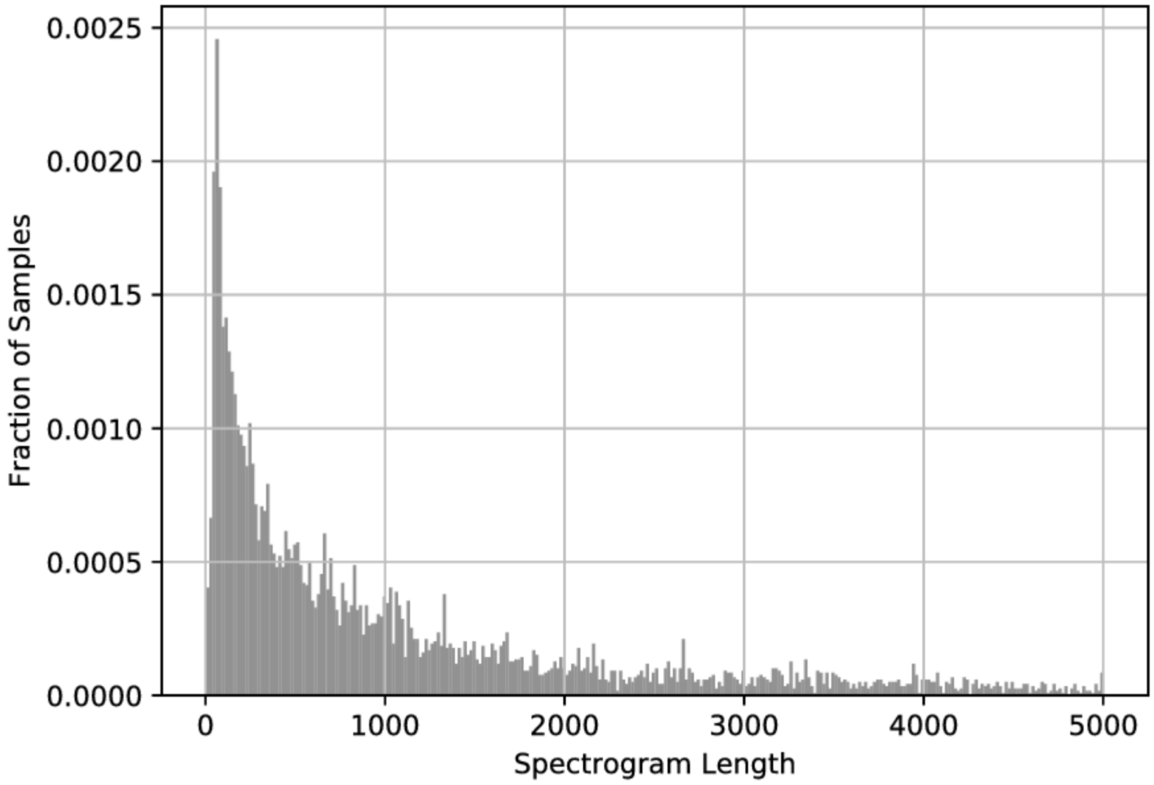


Version – 2

- STFT hop-size: 128
- 1024-sample hann windows
- Logarithm of the power spectrogram
- Log-scaled filterbank (128 bins)

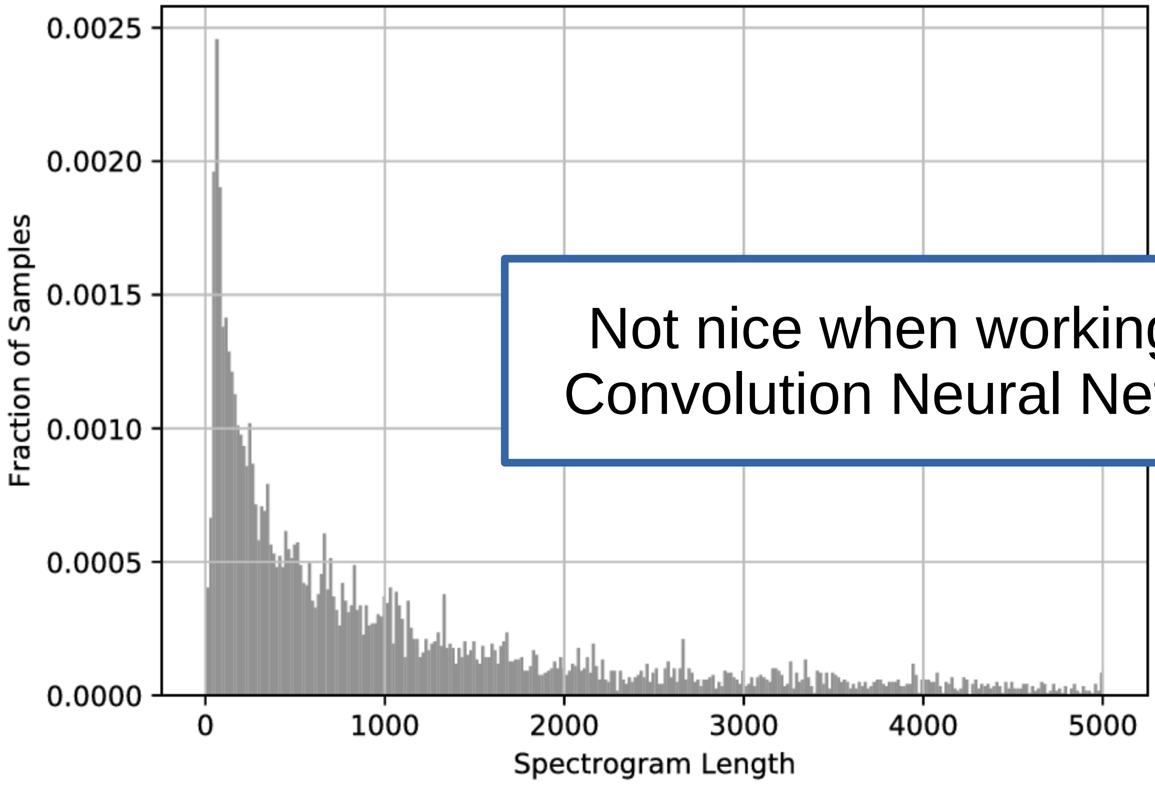


Spectrogram Length Distribution



- Unequal length distribution
- Few long examples
- Many short examples

Spectrogram Length Distribution



- Unequal length distribution
- long examples
- Many short examples

Not nice when working with Convolution Neural Networks

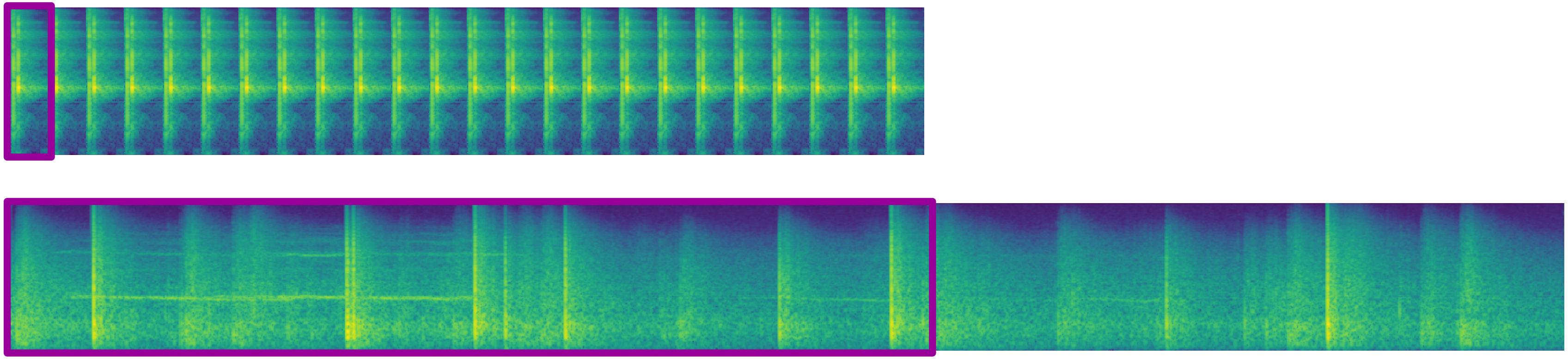
Dealing with Spectrogram Lengths

Fix length to 3000 frames

- Repeat a given excerpt in case it is too short
- Clip at 3000 frames in case it is too long

Repeat

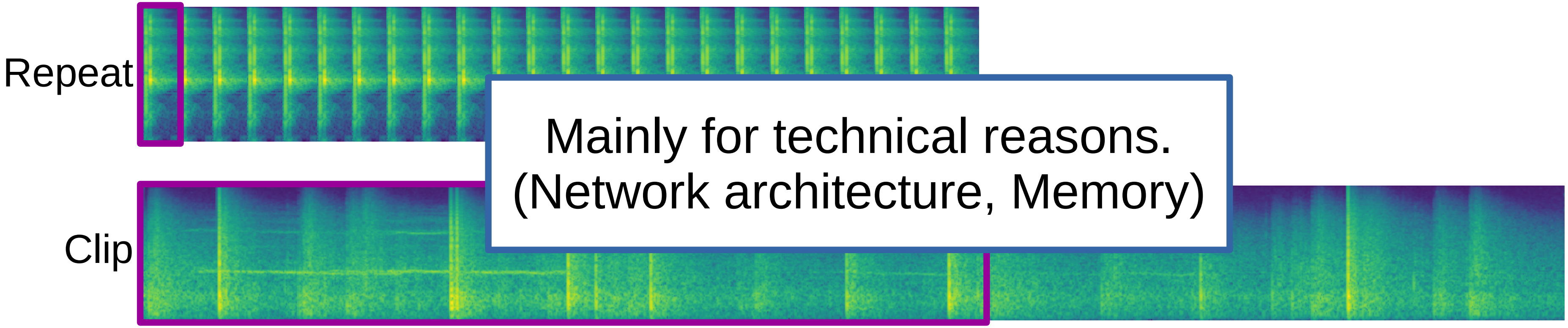
Clip



Dealing with Spectrogram Lengths

Fix length to 3000 frames

- Repeat a given excerpt in case it is too short
- Clip at 3000 frames in case it is too long



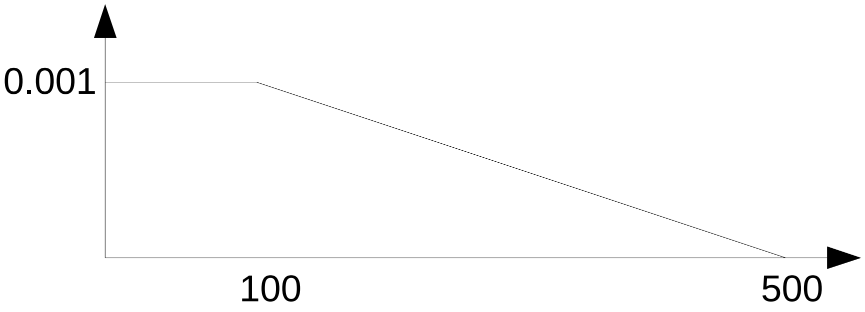
Mainly for technical reasons.
(Network architecture, Memory)

Network Architecture

- Fully Convolutional Neural Network
  - VGG-Style (3 x 3 convolutions & 2 x 2 max-pooling)
  - Global Average Pooling over 41 feature maps
- *Why?*
  - Less parameters in classification layer
  - Deals with varying spectrogram length
    (Nice to have for application time)

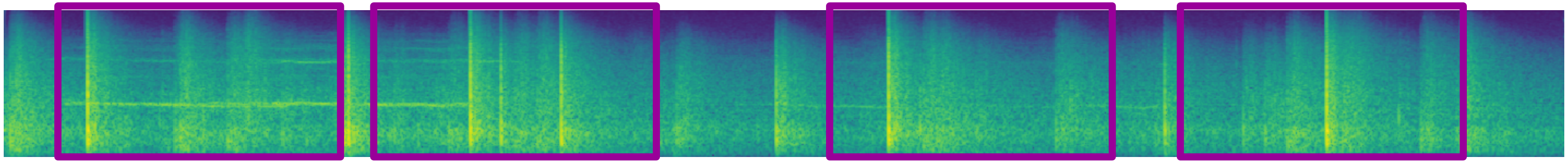
Training Procedure

- ADAM: 500 epochs with initial lerning rate 0.001
- Linear learning rate decay starting from epoch 100



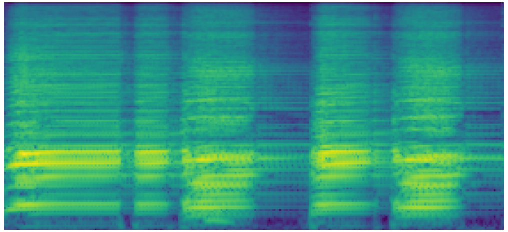
Training Procedure

- ADAM: 500 epochs with initial lerning rate 0.001
- Linear learning rate decay starting from epoch 100
- Spectrogram Excerpt Sub-Sampling (384 frame excerpts)

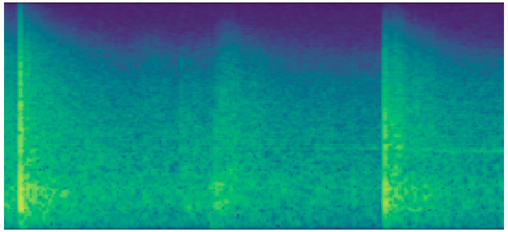


Training Procedure

- ADAM: 500 epochs with initial lerning rate 0.001
- Linear learning rate decay starting from epoch 100
- Spectrogram Excerpt Sub-Sampling (384 frame excerpts)
- Mixup Data Augmentation (α=0.3)



$+$

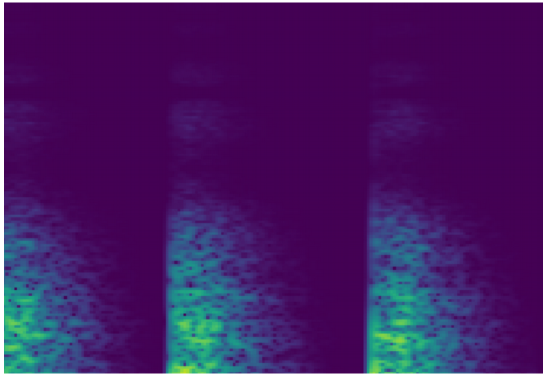


$\alpha \, \mathbf{X}_1$
$\alpha \, \mathbf{y}_1$

$(1 - \alpha) \, \mathbf{X}_2$
$(1 - \alpha) \, \mathbf{y}_2$

4-Fold Iterative Self-Verification

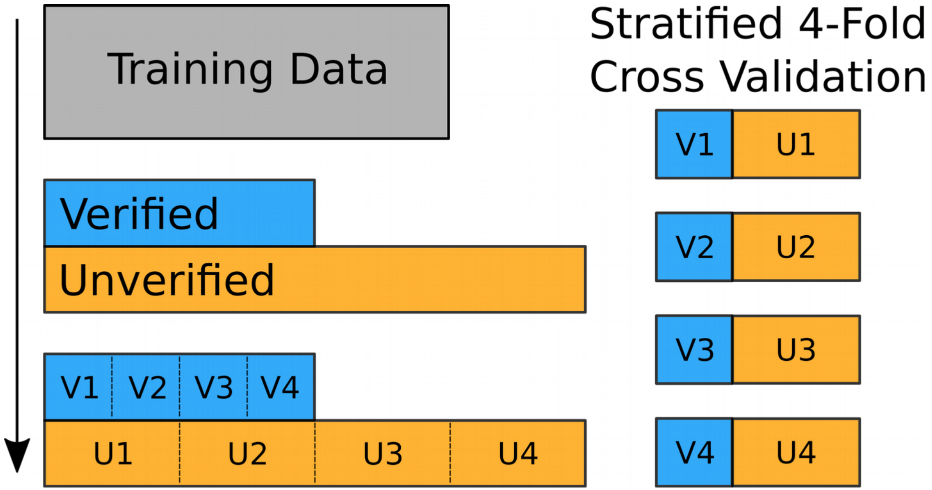
- Address the noisy labels in the development dataset.
- **Central Idea**: Gradually shift unverified labels into the verified, trusted training set for fine-tuning the models



Is this really class *Knock*?

4-Fold Cross-Validation Setup

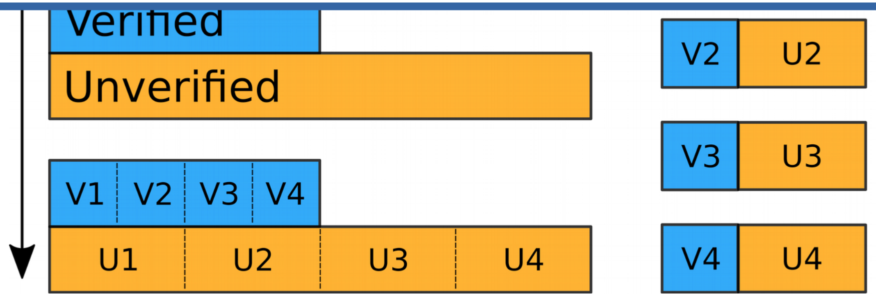
- Crucial component for self-verification
- Parts of the data (to be verified) must not be presented to the verification network for training
- Prediction would be worthless
- Stratified sub-folds!
  (keep label distribution)



4-Fold Cross-Validation Setup

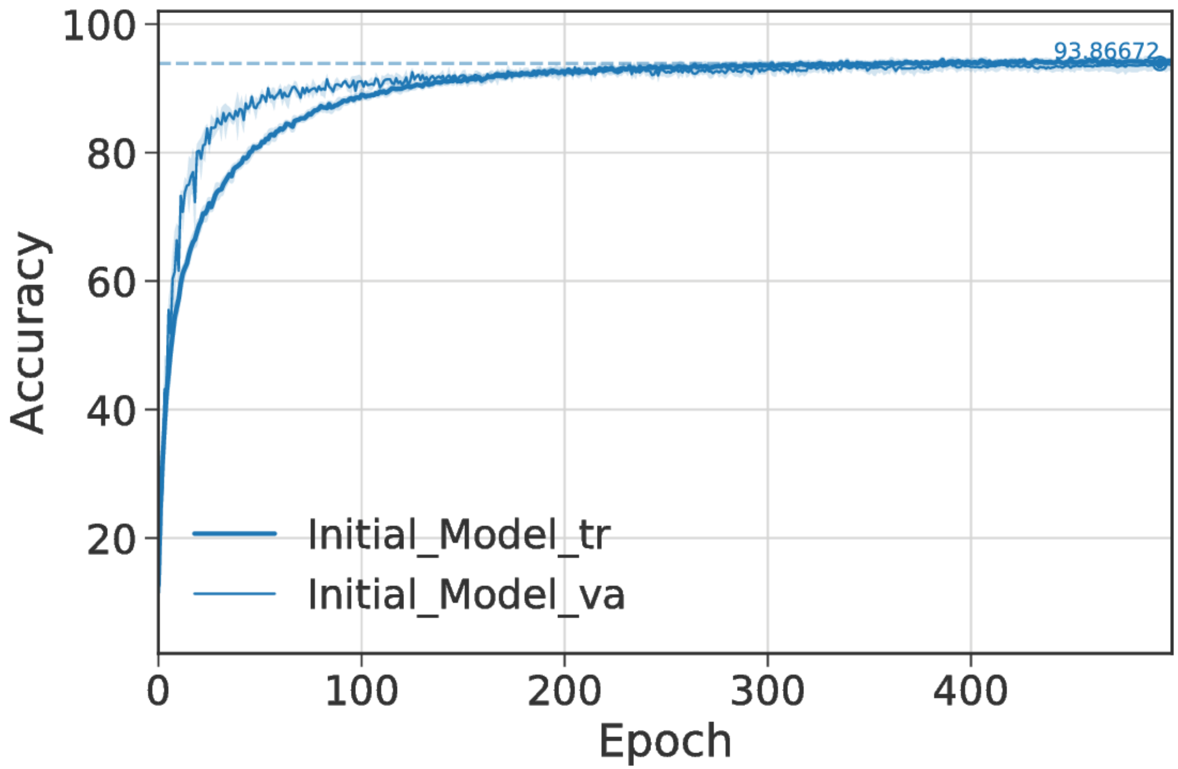
- Crucial component for self-verification
- Parts of the data (to be verified) must not be presented to the verification network for training

> *“The test set is composed of ~1.6k samples with manually-verified annotations and with a similar category distribution than that of the train set.”*



Iterative Self-Verification Loop

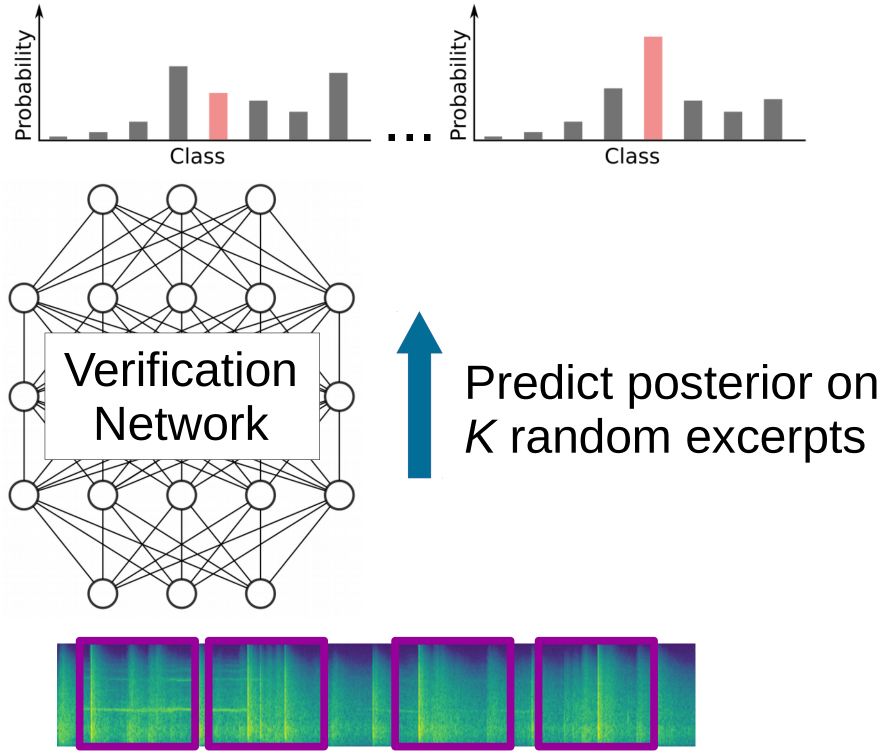
Train model on verified and unverified training data



Iterative Self-Verification Loop

Train model on verified and unverified training data

Predict on “unseen“, unverified validation data



Verification Example

Iterative Self-Verification Loop

Train model on verified and unverified training data

↓

Predict on “unseen“, unverified validation data

↓

Self-verify labels

<u>Verification Conditions</u>

1) Automatic annotation and avg. prediction agree ($y = y_p$)
2) Average target class posteriors exceeds 0.95
3) Count of 40 self-verified examples per class is not reached

Manually Verified	Automatically Verified	Unverified

Iterative Self-Verification Loop

Train model on verified and unverified training data

↓

Predict on “unseen“, unverified validation data

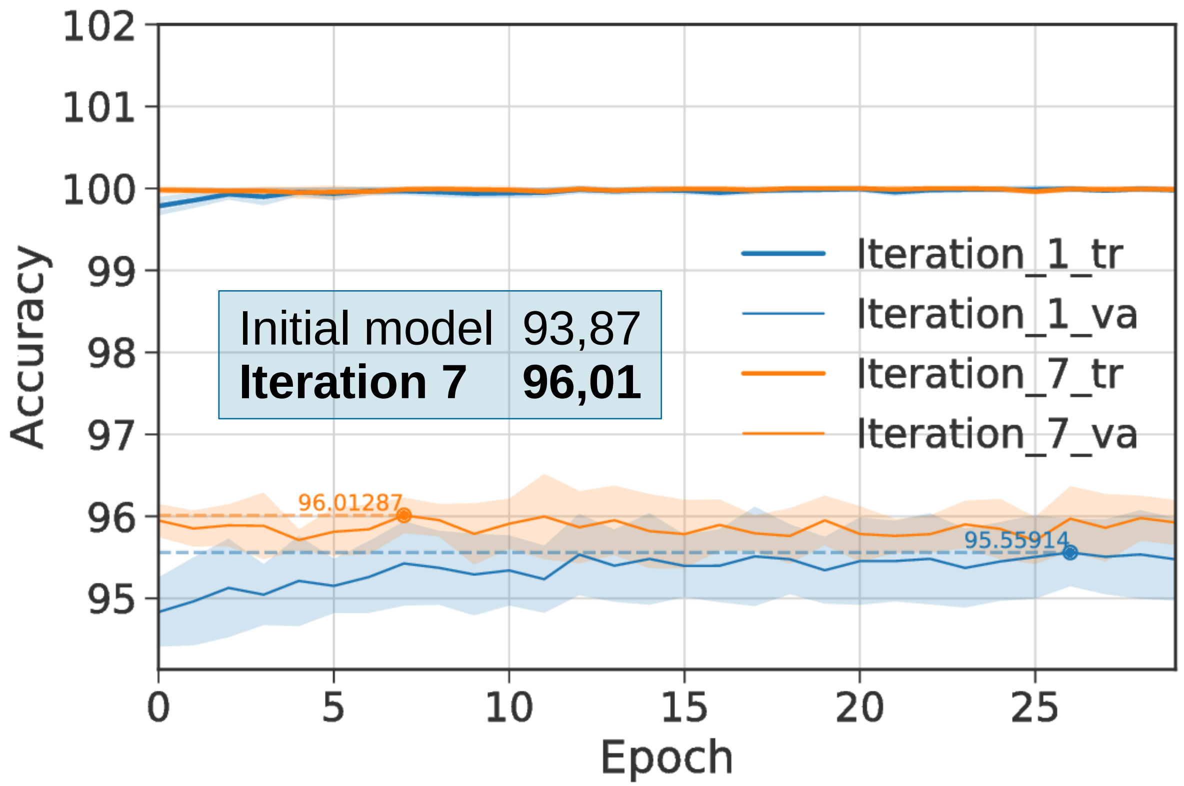
↓

Self-verify labels

↓

Finetune model with verified and self-verified labels

↺ (back to "Predict on “unseen“, unverified validation data")

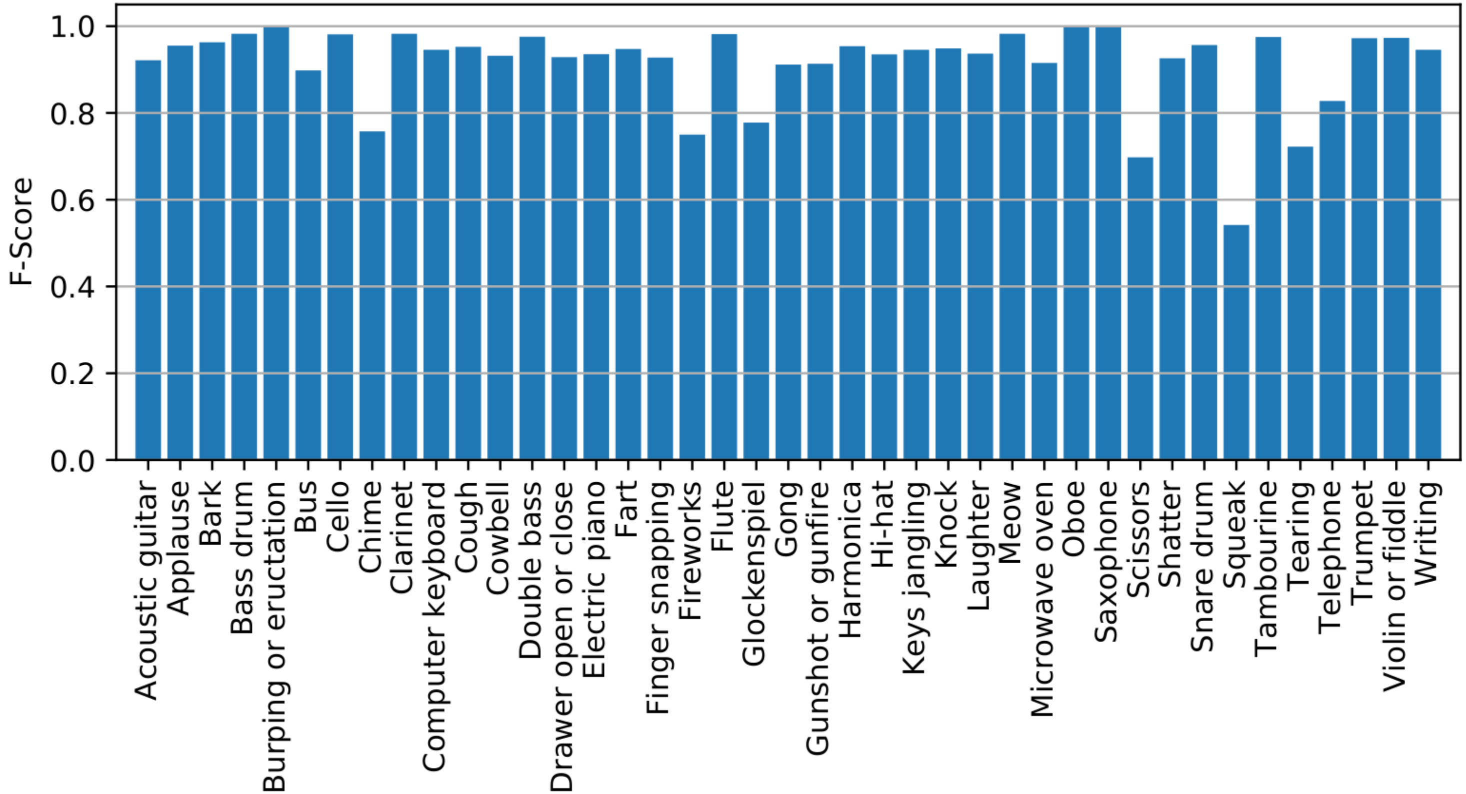


Experimental Setup

- We evaluate the model of the iteration with highest verified validation set score.
- Test set comprises 1600 unseen audio clips
- Evaluation Measures
  - Mean Average Precison (MAP@3)
  - F-Score for Individual Classes

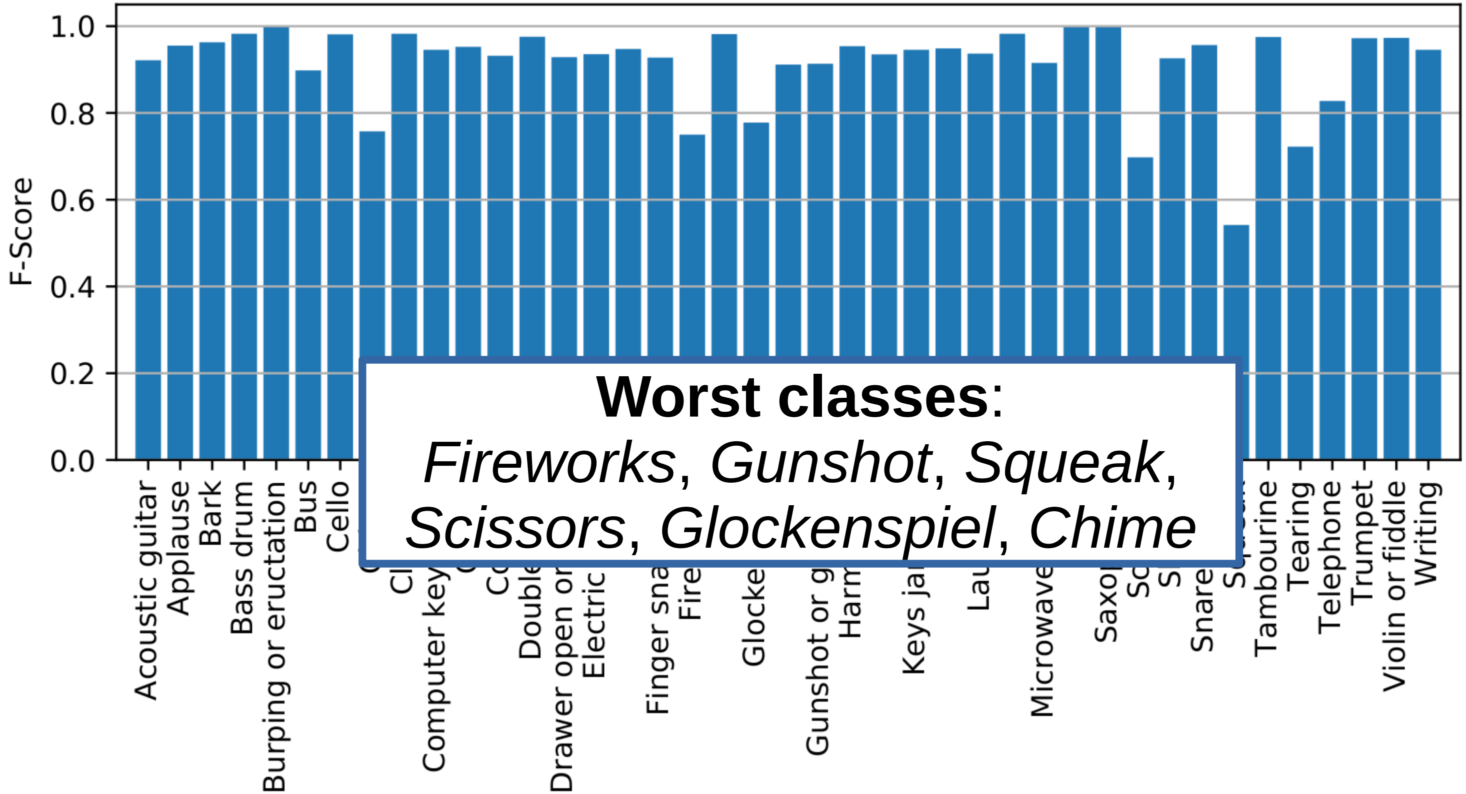
Experimental Results

	MAP@3
Public	**0.9563**
Private	**0.9518**



Experimental Results

	MAP@3
Public	**0.9563**
Private	**0.9518**

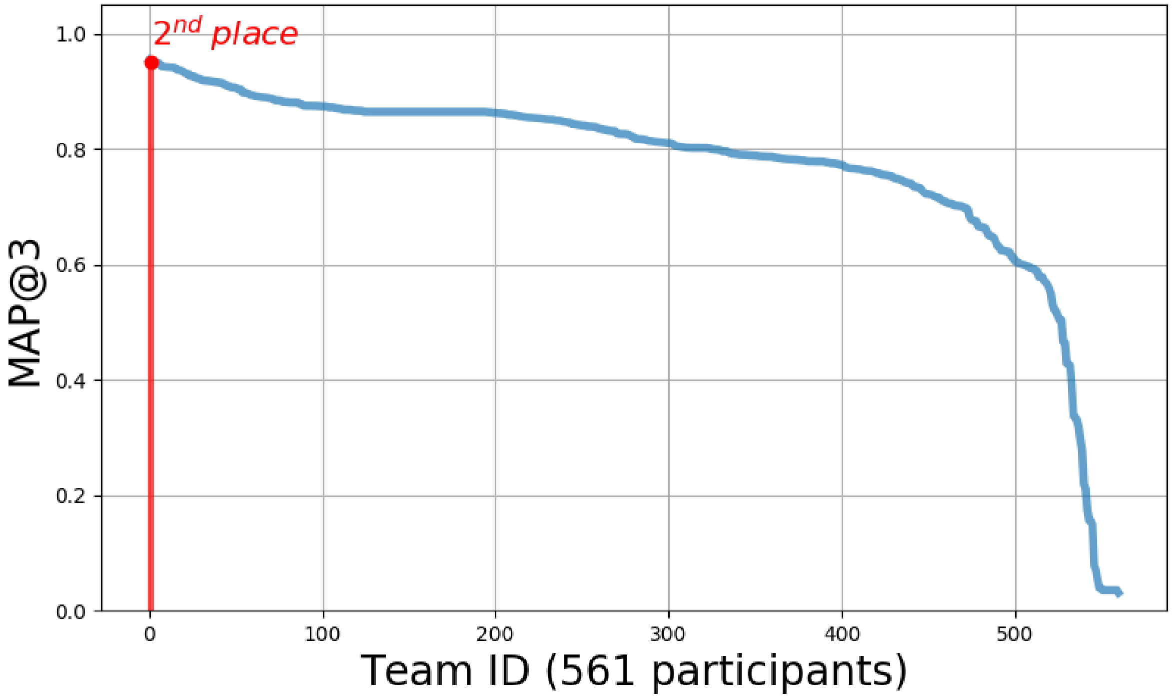


Worst classes:
Fireworks, *Gunshot*, *Squeak*, *Scissors*, *Glockenspiel*, *Chime*

Experimental Results

	MAP@3
Public	**0.9563**
Private	**0.9518**

Private Kaggle Leaderboard



Live Machine Listening Demo
on unseen sounds ...

Live Demo

Summary and Conclusions

- **Proposed Approach**:
  - Iterative Self-verification Loop
  - Fully Convolutional Neural Network
    (VGG, Global Average Pooling, **2nd place Task 1A**)
- Improvement from **93.87% to 96.01%**
  (nice but we can't expect miracles)
- Reminder for how important **the right ML setup** is
- Audio (Signal) **Pre-Processing is still key**
- *https://cpjku.github.io/dcase_task2/*