

# YOHO model for Audio Segmentation and Sound Event Detection

Davide Capone [SM3500601]    Enrico Stefanel [SM3500554]  
`{davide.capone, enrico.stefanel}@studenti.units.it`

Data Science and Scientific Computing Master's Course  
Department of Mathematics and Geosciences  
University of Trieste

A.Y. 2023–2024

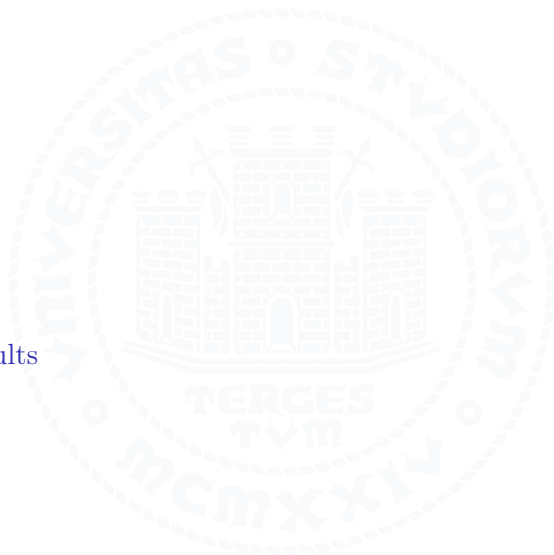
# Contents

Introduction

YOHO model

Work done and results

Conclusions



# Audio Segmentation and Sound Event Detection

The goal of automatic sound event detection (SED) methods is to recognize what is happening in an audio signal and when it is happening<sup>1</sup>. In practice, the goal is to recognize at what temporal instances different sounds are active within an audio signal. An example of sound event detection is presented below.

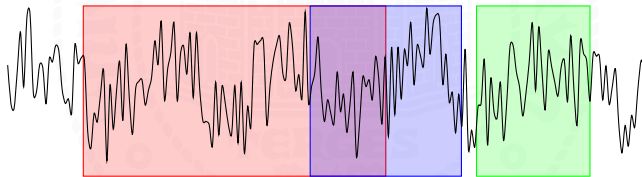


Figure 1: Event Detection in an audio track.

---

<sup>1</sup>Annamaria Mesaros et al. “Sound Event Detection: A tutorial”. In: *IEEE Signal Processing Magazine* 38 (2021), pp. 67–83. URL: <https://api.semanticscholar.org/CorpusID:235795366>.

## Datasets

Common datasets for Audio Segmentation and Sound Event Detection problems are:

- **TUT Sound Event Detection:** primarily consists of street recordings with traffic and other activity, with audio examples of 2.56 s and a total size of approximately 1.5 h. It has six unique audio classes – Brakes Squeaking, Car, Children, Large Vehicle, People Speaking, and People Walking;
- **Urban-SED:** purely synthetic dataset, with audio example of 10 s and a total size of about 30 h. It has ten unique audio classes – Air Conditioner, Car Horn, Children Playing, Dog Bark, Drilling, Engine Idling, Gun Shot, Jackhammer, Siren, and Street Music.

The first dataset is too small to train a Neural Network model and requires use of augmentation techniques (we used **SpecAugment**<sup>2</sup>).

<sup>2</sup>Daniel S. Park et al. “SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition”. In: *Interspeech 2019*. 2019, pp. 2613–2617. DOI: [10.21437/Interspeech.2019-2680](https://doi.org/10.21437/Interspeech.2019-2680).

## Metrics

A popular toolbox for Sound Event Detection models evaluation is **SED Eval**<sup>3</sup>.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F}_1\text{-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where TP, FP and FN are respectively (for each audio segment):

- the ground truth and system output both indicate an event to be active;
- the ground truth indicates an event to be inactive, but the system as active;
- the ground truth indicates an event to be active, but the system as inactive.

---

<sup>3</sup>Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. “Metrics for Polyphonic Sound Event Detection”. In: *Applied Sciences* 6.6 (2016). ISSN: 2076-3417. DOI: [10.3390/app6060162](https://doi.org/10.3390/app6060162). URL: <https://www.mdpi.com/2076-3417/6/6/162>.

# YOHO model

Presented in 2021<sup>4</sup>...

---

<sup>4</sup>Satvik Venkatesh, David Moffat, and Eduardo Reck Miranda. “You Only Hear Once: A YOLO-like Algorithm for Audio Segmentation and Sound Event Detection”. In: *Applied Sciences* 12.7 (Mar. 2022), p. 3293. ISSN: 2076-3417. DOI: 10.3390/app12073293. URL: <http://dx.doi.org/10.3390/app12073293>.

# Input shape

...



# Network Architecture

...





# Output shape

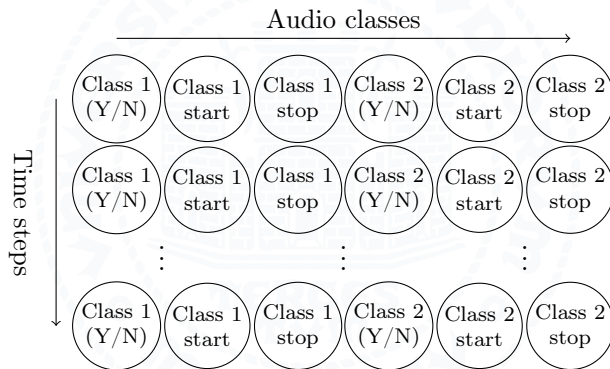


Figure 2: The YOHO output shape.

## Loss Function

$$\mathcal{L}_c(\hat{y}, y) = \begin{cases} (\hat{y}_1 - y_1)^2 + \\ (\hat{y}_2 - y_2)^2 + (\hat{y}_3 - y_3)^2 & \text{if } y_1 = 1 \\ (\hat{y}_1 - y_1)^2, & \text{if } y_1 = 0 \end{cases}$$

where  $y$  and  $\hat{y}$  are the ground-truth and predictions respectively.  $y_1 = 1$  if the acoustic class is present and  $y_1 = 0$  if the class is absent.  $y_2$  and  $y_3$ , which are the start and endpoints for each acoustic class are considered only if  $y = 1$ . In other words,  $(\hat{y}_1 - y_1)^2$  corresponds to **the classification loss** and  $(\hat{y}_2 - y_2)^2 + (\hat{y}_3 - y_3)^2$  corresponds to **the regression loss**.

# Other Details

...



## Implementation challenges

Starting from the original paper, we implemented the system using PyTorch<sup>5</sup>, writing the code keeping in mind that it had to be **clear** and permit **reproducible tests**.

```
$ python3 -m yoho.train --help
usage: train.py [-h] [--name NAME] [--epochs EPOCHS] [--batch-size BATCH_SIZE] [--cosine-annealing]
               [--autocast] [--spec-augment]

options:
  -h, --help            show this help message and exit
  --name NAME            The name of the model
  --epochs EPOCHS       The number of epochs to train the model
  --batch-size BATCH_SIZE
                        The batch size for training the model
  --cosine-annealing     Use cosine annealing learning rate scheduler
  --autocast             Use autocast to reduce memory usage
  --spec-augment        Augment the training data using SpecAugment
```

Listing 1: Training script parameters

We used ORFEO<sup>6</sup> computational resources for the trainings of the models.

<sup>5</sup>All the code is available at <https://github.com/enstit/YOH024>.

<sup>6</sup><https://www.areasciencepark.it/piattaforme-tecnologiche/data-center-orfeo/>

# Training results



Figure 3: Training and validation loss for YOHO model on UrbanSED dataset.

# Conclusions

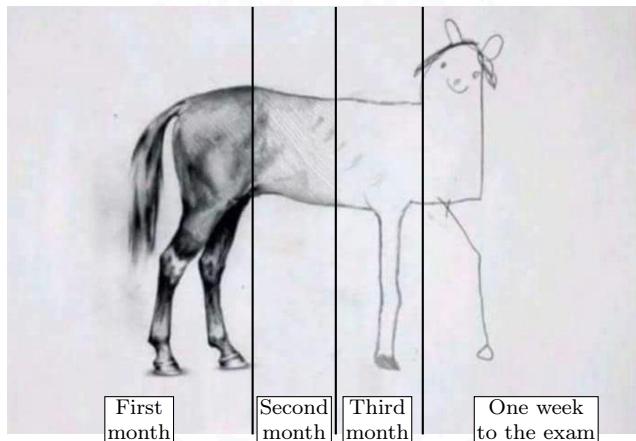


Figure 4: The roadmap of our journey.

# Conclusions

But, after all...

*It's all about the journey, not the destination.*

Thank you for your attention.