REAL-TIME SOUND EVENT DETECTION

USING WEAKLY LABELED DATASETS

HARSHIT MAHAPATRA, 201803187
PATRICK LEWANDOWSKI, 201802946

PROJECT WORK
February 2020
Advisor: Ira Assent

AARHUS
UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

REAL-TIME SOUND EVENT DETECTION

HARSHIT MAHAPATRA
PATRICK LEWANDOWSKI

Using Weakly Labeled Datasets

Project Work
Department of Computer Science
Science & Technology
Aarhus University

February 2020

ABSTRACT

Multimedia streaming has seen tremendous growth in the last few years giving rise to new opportunities as well as challenges. One of these challenges is to improve the user experience when it comes to audio. An interesting avenue for doing so is to analyze audio by extracting the contained events. The extracted events can then be used for a multitude of tasks ranging from real-time enhancements to better audio classification for recommender systems. The goal of our project is thus to make an event extraction system using deep learning methods, with the aim of applying it to streaming audio.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Multimedia streaming has seen tremendous growth in the last few years, with new streaming services popping up every few months. This has resulted in a highly competitive market with ever increasing consumer expectation. Audio is an integral part of streaming and can make or break the user experience. Improving streamed audio quality through analysis is thus an area of interest for streaming companies.

An interesting audio analysis method is to extract event information from a given audio clip. The extracted events can be then used for applications such as event based audio enhancement or classification and can lead to a significant increase in the streamed audio quality.

Recent advances in the field of deep learning have led to efficient information extraction methods capable of extracting complex information from given data. Furthermore there has been a lot of progress in specialised deep learning architectures designed specifically for handling time series data, making them an ideal fit for audio datasets.

Our project thus consists of making an event extraction system using deep learning methods, with the aim of applying it to streaming audio. Formally, the goal of this project is thus to test the following hypothesis:

"Is it possible to robustly extract audio events from a given audio clip using deep learning."

Here, by robust we mean that the system should be able to extract a variety of audio events, occurring simultaneously from a given audio source.

2

RELATED WORK

This section briefly covers the related work. We first present a brief summary of different kinds of audio event extraction, which is then followed by an overview of the traditional as well as new methods used to perform the said extraction. Finally we present the types of dataset used to extract audio events using deep learning.

Audio event extraction can be categorized into the following methods based on the kind of information extracted:

1. Audio Tagging: It is the task of predicting whether an audio event is present or absent in an audio clip.

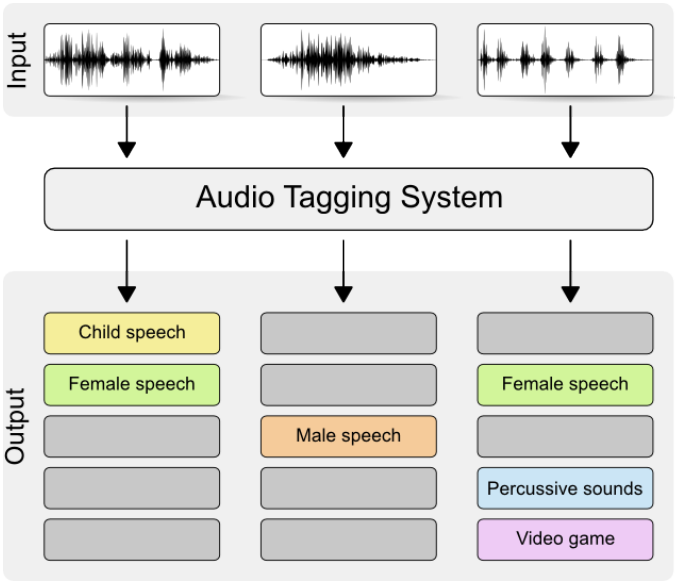

Figure 1: Overview of audio tagging system [4]

2. Audio Classification: It is the task of assigning only one label to an audio clip.

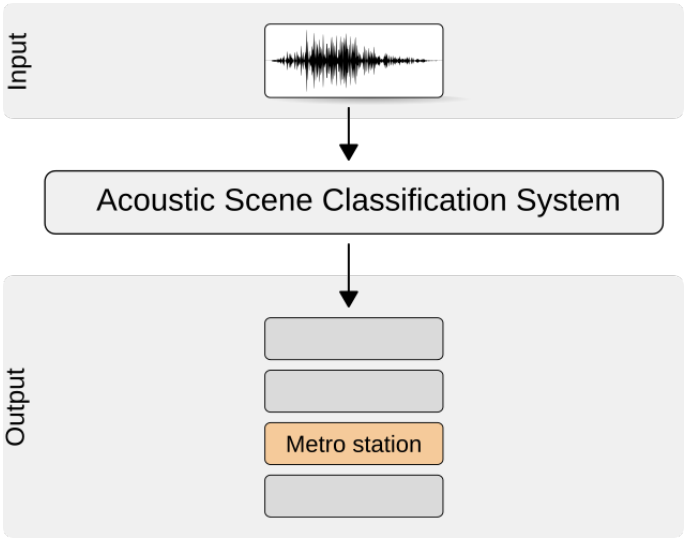

Figure 2: Overview of audio classification system [1]

3. Sound Event Localization: It is the task of predicting the location of an event in an audio clip.
4. Sound Event Detection: It is the task of predicting the location of an event as well as classifying the said event in a given an audio clip.

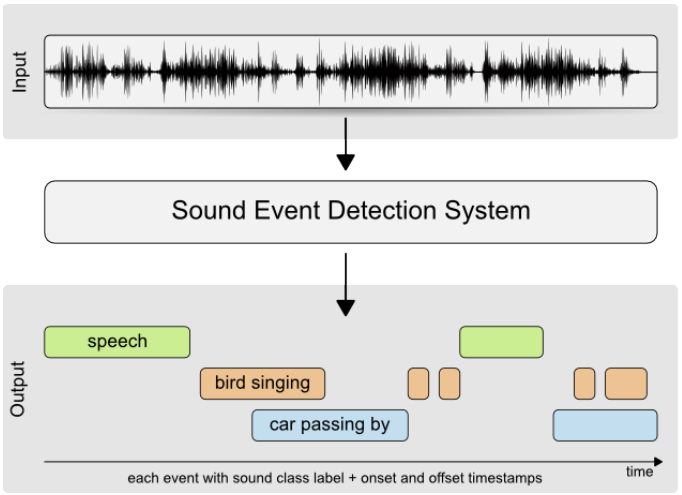

Figure 3: Overview of sound event detection system [11].

Traditional methods to extract events from audio clips include using signal properties such as mel-frequency cepstral coefficients (MFCCs), recurrence quantification analysis (RQA), stabilized auditory image (SAI),etc as features. Machine learning methods such as Gaussian mixture models (GMMs), Hidden Markov Models (HMM), Support Vector Machines (SVM) have also been used for audio classification and audio tagging with reasonable accuracy.

Deep learning methods such as LSTM, CNN and their variants have been used successfully for the tasks such as audio tagging and audio scene classification with high degree of accuracy [7], [12], [8].

The state of the art audio tagging methods use attention neural networks [8]. These networks were originally designed for the purpose of natural language processing, but have proved to perform well on sequential data in general. The general idea behind attention neural network is that the network learns to pay "attention" to a specific part of the input, in our case the network would pay attention to specific seconds of an input audio clip.

The first notable attempt at using attention neural network for audio tagging was presented by Kong, et al [7].

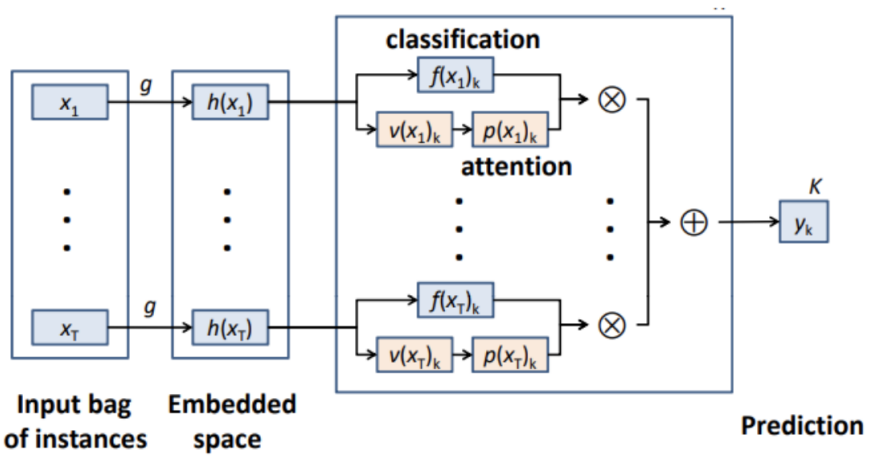

Figure 4: Decision Level Single Attention Network.

The basic idea of the proposed model is that the network learns to pay attention to specific parts i.e. seconds of the input. The network takes VGGish bottleneck features of an audio clip as input and outputs the probability of an event being contained in the said audioclip.

The VGGish bottleneck features are just features extracted from an audioclip using VGGish feature extractor network [5]. The VGGish network takes as input an audio clip and outputs a matrix with a 128 dimensional vector for each second of input.

Yu et al [12] improved on Kong's model by taking into account the output intermediate layers along with the last layer while calculating the distribution of attention.

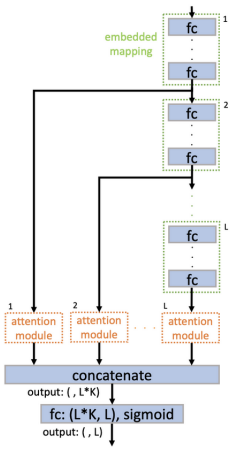

Figure 5: Multi-level Attention Model.

Recently Kong proposed *Feature Level Attention Networks* [8] which improved upon Yu's model by using features extracted from attended output of decision level single attention.

The performance of the attention neural networks on audio tagging is summarized in the table below:

Models	mAP	AUC	d-prime
Google's baseline	0.314	0.959	2.452
average pooling	0.300	0.964	2.536
max pooling	0.292	0.960	2.471
single_attention [1]	0.337	0.968	2.612
multi_attention [2]	**0.357**	**0.968**	**2.621**
feature_level_attention [3]	**0.361**	**0.969**	**2.641**

Figure 6: Literature Benchmarks.

Based on their labeling, audio datasets can be classified into two categories:

1. Weakly Labeled Datasets (WLD): These datasets contain labels which only specify whether a certain event has occurred in a given audio clip or not.

2. Strongly Labeled Datasets (SLD): These datasets contain labels specifying events contained in a given audio clip as well as their corresponding start and end times.

WLDs are generally larger and more prolific as compared to SLDs as the former is far easier to create than the later. SLDs though rich

in information are generally very specific and small in size, making them not ideal for data intensive algorithms like deep learning.

One of the most popular WLD is Audioset [6] released by Google. It has 2,042,985 samples and 527 different labeled events. Audioset was released with the aim to provide an audio dataset analogous to the famous ImageNet dataset. The dataset was originally released as a collection of links to YouTube videos, but later a VGGish embedding version of it was released, where each sample consisted of VGGish bottleneck features extracted from the corresponding audio clip. Furthermore, a balanced variant of the dataset containing 59 samples per class, having a total of 22,176 samples.

3

ANALYSIS

This section covers our analysis of the related work and the decisions we took based on it.

Based on our study, we found that the type of audio event extraction depends on the granularity of event information one wishes to extract. For example, if we want to predict whether a certain event happened in a given audio clip as a whole, we would be performing audio tagging and using methods designed for the same. On the other hand, if we want to extract events on a per second basis or less, we would have to precisely know the beginning and end of each event. This would require us to use Sound Event Detection Methods instead.

We also found that deep learning methods, specially the ones designed to handle sequential data have the potential to perform very well on audio event extraction tasks.

Finally we found that Strongly Labeled Datasets (SLDs) although rich in information, are not suitable for deep learning methods because of their small size.

Based on our analysis, we formulated the following objectives for the project:

1. Audio Tagging: To train an audio tagger on a WLD to predict whether an audio event is present in a given audio clip.
2. Sound Event Detection: To make the prediction more granular to predict the presence or absence of an event per second i.e. performing SED using WLD (called Weakly Supervised SED).

We also formulated the following stretch goals for the project:

1. Real-Time Sound Event Detection: To make the prediction algorithm online (receives a stream input, instead of the whole audio clip).
2. Proof of Concept: Have a demo website which takes as input the URL of audio/video clip and passes it through our trained model.

4

DESIGN OF ARCHITECTURE AND EVALUATION

Based on our analysis and objectives we designed the following overall architecture to as a proof of concept to test our hypothesis and achieve the project's objectives:

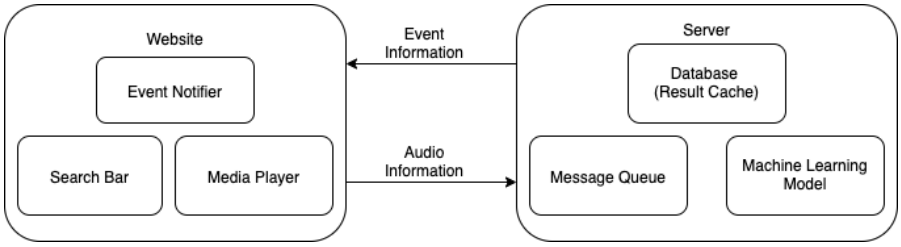

Figure 7: Initial Design

4.1 COMPONENTS AND THEIR INTERACTION

4.1.1 *Server*

The server is responsible for most of the processing. On a high level, the server's responsibilities are:

1. Maintaining a message queue to send and message to and from the website.
2. Retrieving the audio clip requested by the website either in chunks or as a whole depending on the method of operation (real-time/passive).
3. Performing required audio event extraction on the audio clip.
4. Storing the results in a database for caching
5. Sending the extracted event information to the message queue for the website to consume.

4.1.2 *Website*

The website acts as the user interface and it's responsibilities include:

1. Sending the URL of requested audio clip to server.
2. Playing the audio requested by the user.
3. Sending current playback time to server.

4. Retrieving audio event information from the server and presenting them to the user in the form of notification.

4.2 METHODOLOGY

This subsection briefly summarizes the different methods used in the project.

4.2.1 *Audio Tagging*

We decided to use attention neural networks [7] and [8] for the purpose of audio tagging due to their state of the art performance. For training the chosen models, we decided to use the audioset dataset due to it's size and diversity.

4.2.2 *Sound Event Detection*

For sound event detection, we had to figure out a way to perform granular prediction using models trained on WLDs. For this, we formulated the following approaches:

1. Sliding Window Prediction: A sliding window is passed through the input to predict the probability of an event at each point.
2. Weighted Sliding Window: Here also a sliding window is passed through the input to predict the probability of an event at each point, but the prediction of windows are weighed according to the distance of the center of window from each point being predicted on.
3. Attention Based Localization: The output of attention module is used for event localization.

   All the above methods output the probability of the occurrence of a given event at each point of the input audio clip. To be able to concretely predict the presence or absence of an event in a second, we decided to threshold the output from the methods described above.

4.2.3 *Real-Time Sound Event Detection*

[7] and [8] are very fast and predict on input almost instantaneously. The bottleneck with regards to time is the extraction of VGGish bottleneck features. To solve this, we feed chunks of input audio clip (as opposed to the whole audio clip) into VGGish and then pass the output to the chosen attention neural network.

4.3 EXPERIMENTS

4.3.1 *Audio Tagging*

To test the performance of audio tagging, we split audioset dataset into training and test set and plotted the results.

4.3.2 *Sound Event Detection*

To be able to predict on a given audio clip not in dataset using our models, we first needed to get the VGGish bottleneck features of the clip and then feed the said features to the attention neural network.

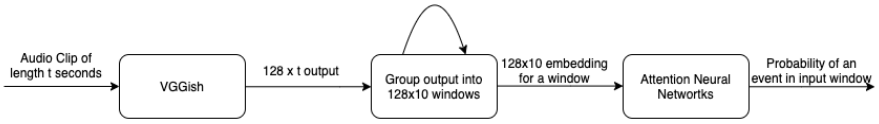

Figure 8: Prediction Pipeline for Audio Clip

To test the performance of SED quantitatively, we required labels with information about the start and end time of individual events. To be able to do so we searched for Strongly Labeled Datasets (SLDs) whose labels are similar to our training dataset.

We also devised an alternate method to test our SEDs using manual labels.

4.3.3 *Real-Time Sound Event Detection*

The performance of real time variant of our SED methods can be measured using the same experiments as non real time variants.

We also plan to measure the latency of the algorithm with respect to chunk size and the corresponding accuracy.

5 IMPLEMENTATION

The final implementation of our system is as follows:

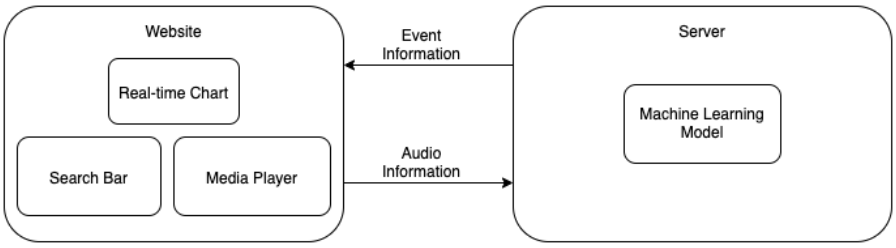

Figure 9: Implemented Architecture

It differs from the ideal implementation in the following ways:

1. The message queue and the caching database were not implemented as they are essential to test our hypothesis.
2. Instead of event notification, we implemented a real time chart showing the likelihood of most likely event each second.

The implementation details of the various methods used in the project are summarized as follows:

5.1 AUDIO TAGGING

We used google's implementation of VGGish [5] for the preprocessing part. The attention neural network themselves were implemented using tensorflow and were trained for 200000 epochs each. Furthermore the training was done on both balanced and unbalanced version of the chosen dataset.

5.2 SOUND EVENT DETECTION

5.2.1 *Sliding Window:*

We decided to slide a window of 10 second width through the audio clip's VGGish embedding and predict the probability of events (classes) being contained in the said window. The window is incremented by 1 second in each iteration and hence consecutive windows have 90% overlap. The predicted values of a window is then assigned to the first second of the bucket. This approach was implemented for both [7] and [8].

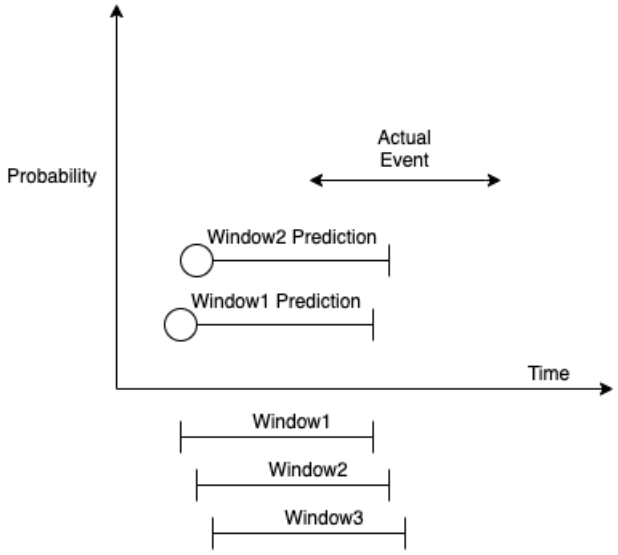

Figure 10: Sliding Window SED

Figure 10 shows the sliding window SED approach for a single event (class).

5.2.2 *Weighted Sliding Window:*

The first step of this approach is similar to sliding window SED in the sense that here also we slide a window through the audio clip's VGGish embedding and get the event probability for each window. However, the final probability of an event being present at a given time is calculated differently. The main idea here is the the probability of an event being present in a second is the sum of normalized weighted probabilities obtained from the windows passing through that second. The weight of a window with respect to a second is inversely proportional to how far away the center of the said window is from the second. The algorithm used to calculate the probability of an event e at a given time t is shown in table 1.

Weighted Sliding Window SED algorithm

Input:

n : window size

t : time which we need to find the probability of event e

e : event whose probability we need to find out

W : $w_0, ..., w_n$ set of windows containing event e

P : $p_0^e, ..., w_n^e$ probability of i'th window containing event e

Output:

s_t^e: normalized probability of time t having event t

Process:

1. weights=[], center=[], weightedProbs=[],s_t^e=0
2. For each window w_i in W:
   //Find the center
3. $c_i <- -(w_i^{end} - w_i^{start})/2$
   //find the weight of window w.r.t. time t
   //The weight is inversely proportional to the distance of c_i from t
   // The 0.1 is a smoothing constant to ensure non–zero weights
4. $weight_i <- -(\frac{n}{2} - |t - c_i|) + 0.1$
   //Normalize the weights i.e. make it a distribution
5. For each weight $weight_i$ in weights:
   $weight_i <- -\frac{weight_i}{\sum_{j=0}^{n} weights_j}$
   //Find weighted window probability
6. For each prediction p_i in P
7. $weightedProbs_i <- -p_i * weight_i$
   //Finally find the mean of weighted probability
8. $s_t^e <- -s_t^e + weightedProbs_i$

Table 1: Weighted Sliding Window SED Algorithm

This approach was implemented for both [7] and [8].

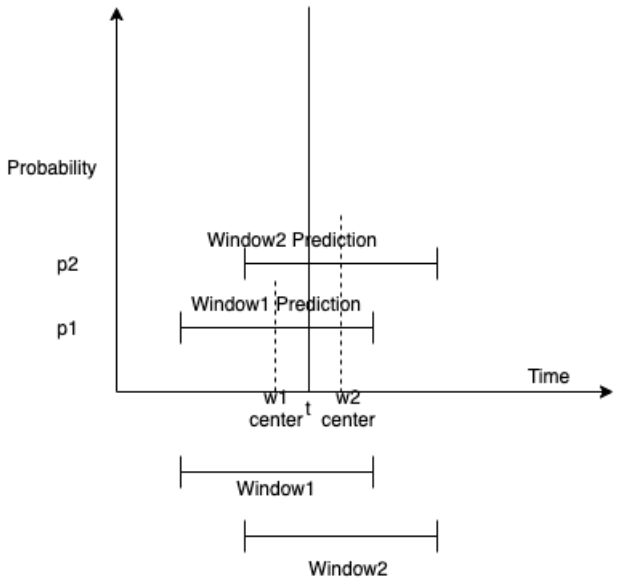

Figure 11: Weighted Sliding Window SED

Figure 11 shows how algorithm 1 would look like in practice. The probability at time t would depend on the mean of weighted probability of windows *Window1* and *Window2*. The weighted probability is calculated using the centers *w1 center* and *w2 center*, the time in question it and the probabilities of the windows *p1* and *p2*.

5.2.3 *Attention Based Localization:*

In this approach we first predict the probability of each class in a window of 10 seconds. We then sort the events by probability to get the most probable events for that window. Finally, we extract the output of attention module of the neural network for corresponding high probability events. The output of attention module corresponds to a probability distribution related to the actual location of events. This thus helps us locate the start and end times of the said events. This approach was implemented for [8].

5.2.4 *Thresholding*

The output of SEDs was first smoothed with Savitzky-Golay filter [10] and then a threshold was applied to get a binary output signifying the presence or absence of an event in a given second.

5.2.5 *Real-time Sound Event Detection*

For real-time SED, audio clips were fed in chunks of 30 seconds to the VGGish which was in turn connected to Attention Neural Networks. This approach was implemented for [8].

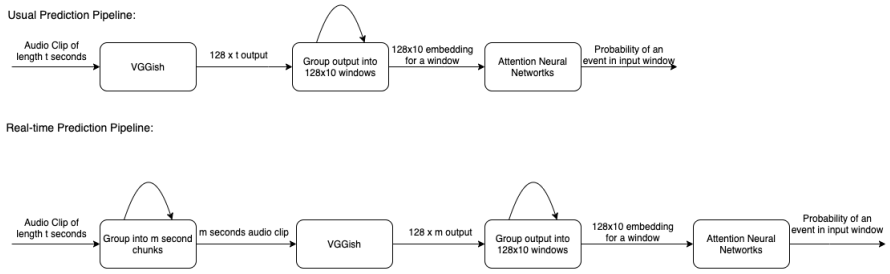

Figure 12: Real-time VGGish-ANN Prediction Pipeline

Figure 12 shows the difference between normal and real-time prediction pipelines. In our test cases the value of m is 30.

6 EVALUATION

This section provides in-depth description of experiments proposed in the Design and Methodology chapter and their results.

6.1 AUDIO TAGGING

To test the performance of audio tagging, we split audioset datasets into training and test set plotted the following results. We measured the performance in terms of Area Under Curve (AUC) and mean Average Precision (mAP).

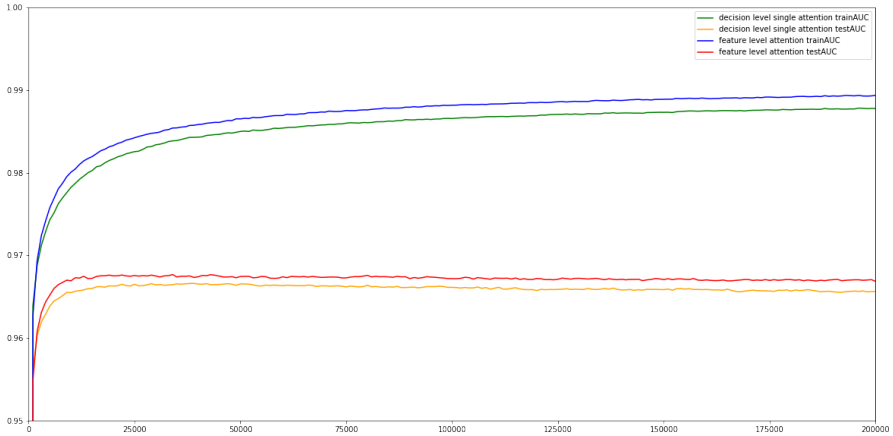

Figure 13: Area under curve (AUC) of training and testing for Decision Level Single Attention and Feature Level Attention

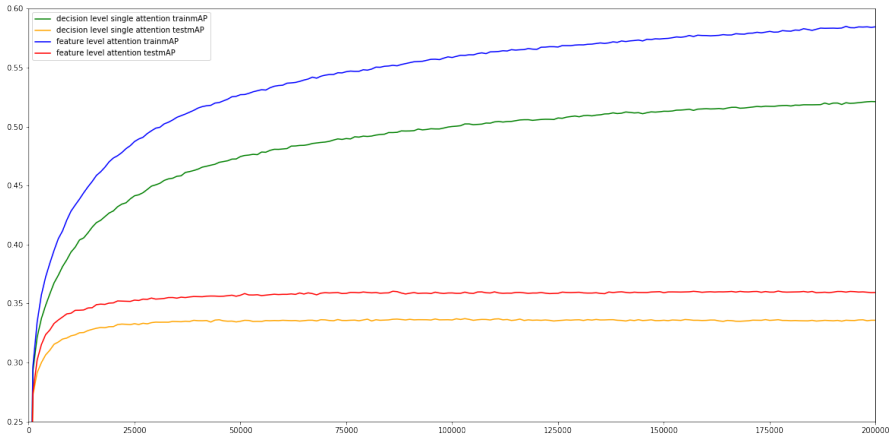

Figure 14: Mean Average Precision (mAP) of training and testing for Decision Level Single Attention and Feature Level Attention

We were able to obtain audio tagging performance at par with the original papers.

6.2 SOUND EVENT DETECTION

Although we were able to obtain some Strongly Labeled Datasets from the DCASE competitions [2], their corresponding labels did not correlate with audioset labels, making it hard to use the said datasets as test samples. We thus decided to use manual labeling to test our SED models. To do so, we selected a clip from YouTube, manually labeled it and then compared the prediction of our SED models with the manual labels.

In the following charts, the lines labeled *Speech SLD* and *Dog SLD* refer to the ground truth (Strongly Labeled Data) obtained via manual labeling of the clip.

6.2.0.1 *Sliding Window*

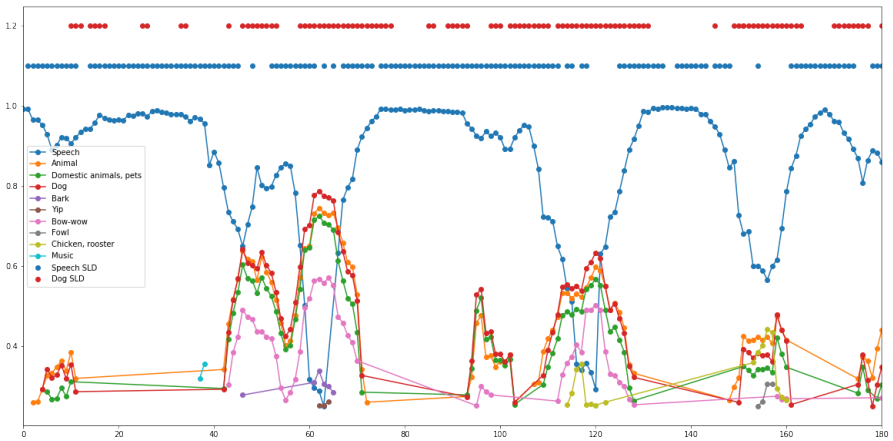

Figure 15: Single Level Attention with Sliding Window SED for [9].

6.2.0.2 *Weighted Sliding Window*

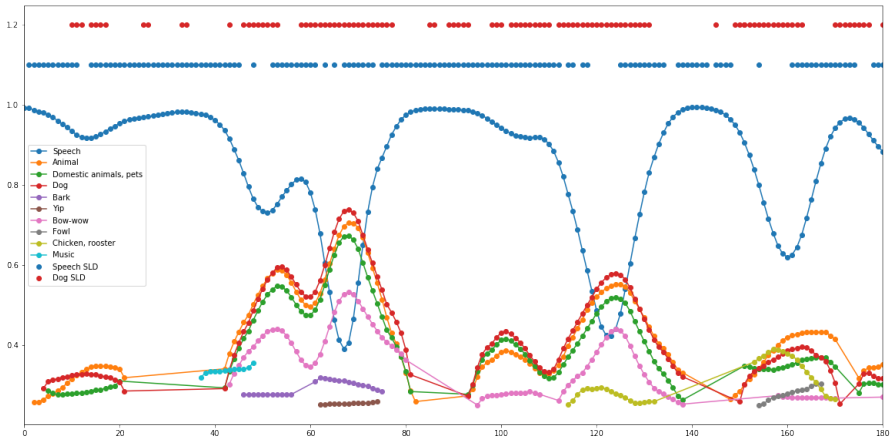

Figure 16: Single Level Attention with Weighted Sliding Window SED for [9].

6.3 ATTENTION BASED LOCALIZATION

In the following charts, the lines labeled *Speech* and *Dog* are the normalized attention distribution extracted from the chosen attention neural network. The labels *SDog* and *SSpeech* refer to the Savitzky-Golay filtered version of attention distribution. Finally, the labels *SGradDog* and *SGradSpeech* refer to thresholded version of the filtered attention distribution.

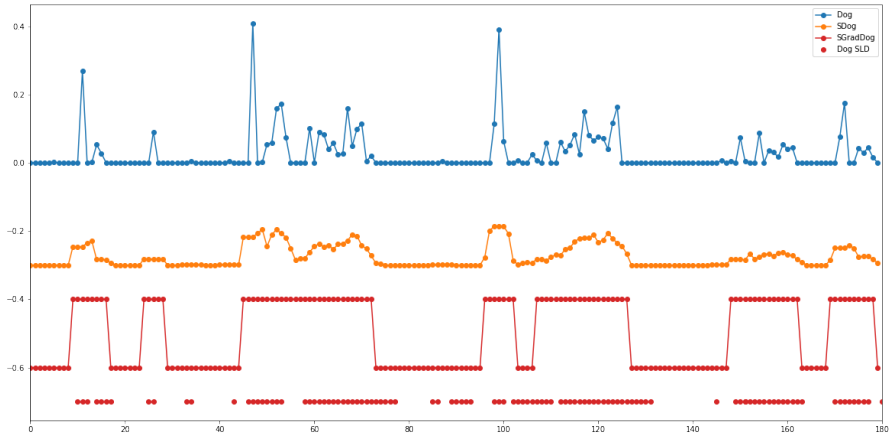

Figure 17: Attention Based Localisation for dog class in [9].

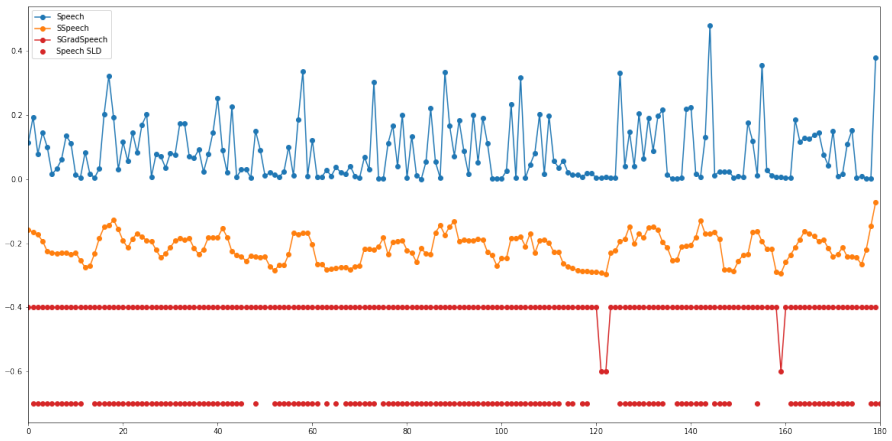

Figure 18: Attention Based Localisation for speech class in [9].

6.4 REAL-TIME SOUND EVENT DETECTION

For real time testing, selected audio clips were fed in chunks of 30 seconds to the VGGish Attention Neural Network pipeline and the predictions were plotted.

6.4.0.1 *Sliding Window*

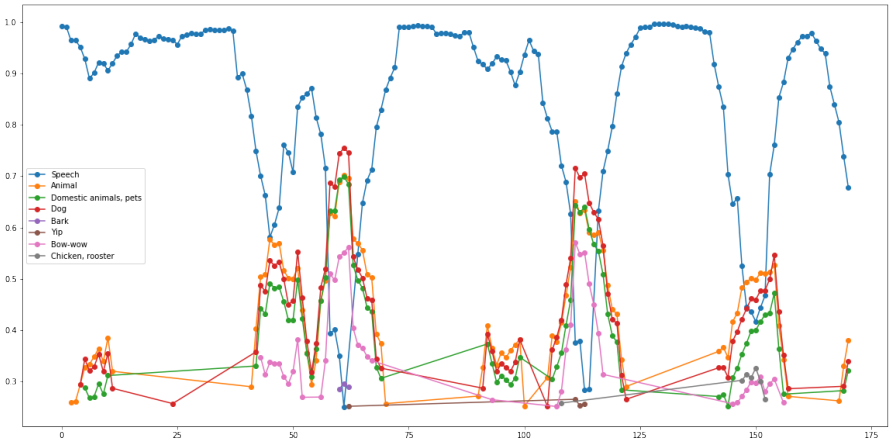

Figure 19: Single Level Attention with Sliding Window SED with 30 sec chunks of [9] fed to VGGish..

6.4.0.2 *Weighted Sliding Window*

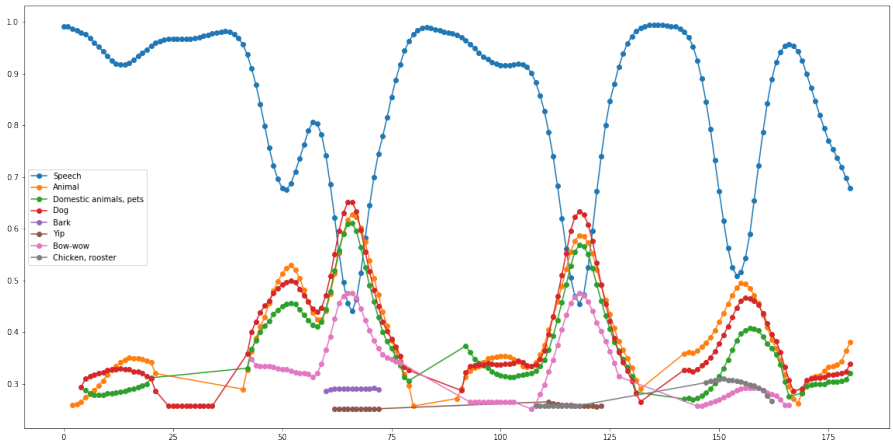

Figure 20: Single Level Attention with Weighted Sliding Window SED with 30 sec chunks of [9] fed to VGGish .

6.4.1 *Attention Based Localization*

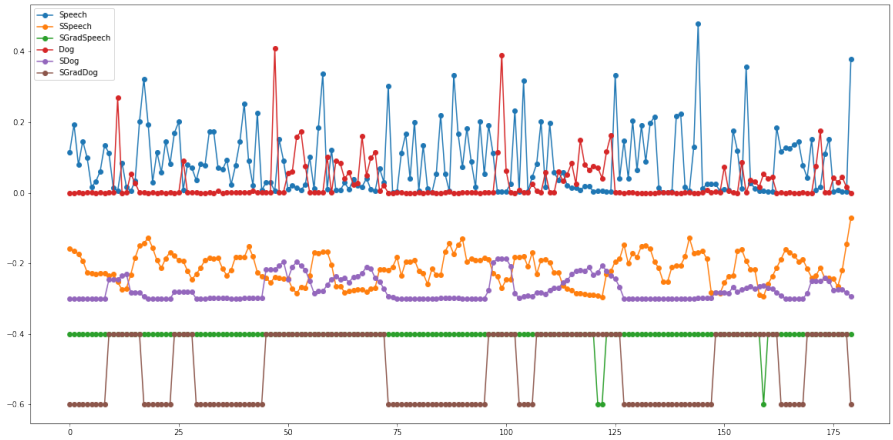

Figure 21: Single Level Attention with Attention Based Localization SED with 30 sec chunks of [9] fed to VGGish.

We were able to achieve prediction close to ground truth in both passive and real time implementation of our SEDs.

We partly implemented the website as a proof of concept for our proposed SEDs, however it was not polished enough to be included in this report.

7

CONCLUSION AND FUTURE WORK

In this project, we tested the following hypothesis:
"Is it possible to robustly extract audio events from a given audio clip using deep learning."

In order to test our hypothesis we attempted to achieve the following objectives:

1. Audio Tagging: Train an audio tagger on a WLD to predict whether an audio event is present in a given audio clip.
2. Sound Event Detection: Make the prediction more granular to predict the presence or absence of an event per second i.e. performing SED using WLD (called Weakly Supervised SED).
3. Real-Time Sound Event Detection: Make the prediction algorithm online (receives a stream input, instead of the whole audio clip).
4. Concept Demo: Have a demo website which takes as input the URL of audio/video clip and passes it through our trained model.

We approached the said objectives by exploring the literature for state of the art solutions, in addition to designing our own. We finalized on using [7] and [8] for audio tagging, whereas for SED we designed our own approaches namely Sliding Window, Weighted Sliding Window and Attention Based Localization. Finally for real-time variants of our proposed methods, we simply split our input into chunks and fed the same to the VGGish Attention Neural Network Pipeline.

We were able to validate the performance of our algorithms by predicting on a youtube clip and comparing the said predictions with ground truth obtained via manual labeling. Thus we can conclude that our hypothesis was validated.

Although we are very happy with what we achieved in this project, there are a few areas we would have liked to explore more, these include:

1. We would have liked to quantitatively measure the performance of our proposed methods by using a Strongly Labeled Dataset with similar labels as Audioset.

2. During our testing, we noticed that the quality of predictions varied slightly depending on whether an audio clip was obtained from youtube or not. We suspect this is possibly because of some compression algorithm used by youtube.

3. We would have liked to explore avenues to further speed up our predictions possibly by implementing our own variant of VGGish.

The above mentioned points form excellent starting point for future research.

It was the best of times, it was the worst of times,
it was the age of wisdom, it was the age of foolishness,
it was the epoch of belief, it was the epoch of incredulity,
it was the season of Light, it was the season of Darkness,
it was the spring of hope, it was the winter of despair

— Charles Dickens - A Tale of Two Cities [3]

ACKNOWLEDGMENTS

The above opening passage from 'A Tale of Two Cities ' by Charles Dickens would describe excellently the highs and lows in this project. In the following we want to acknowledge people who mattered during this time.

First and for most, we would like to thank Professor Ira Assent for supervising and supporting our project. Furthermore we would like to thank Jan Neerbek for giving us the opportunity to work for Roku on an interesting project.

The door to Professor Ira's office was always open whenever we had problems or questions regarding our research or writing.

BIBLIOGRAPHY

[1] *Acoustic scene classification*. http://dcase.community/challenge2019/task-acoustic-scene-classification. Accessed: 2020-01-13.

[2] Dcase. *Detection and Classification of Acoustic Scenes and Events*. URL: http://dcase.community/.

[3] Charles Dickens, Hablot K. Browne, and Frederick Barnard. "A Tale of Two Cities." In: *Print* (1942).

[4] *Domestic audio tagging*. http://dcase.community/challenge2016/task-audio-tagging. Accessed: 2020-01-13.

[5] Dan Ellis, Shawn Hershey, Aren Jansen, and Manoj Plakal. "Vggish." In: *URL https://github. com/tensorflow/models/tree/master/research/audioset/vggish* ().

[6] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. "Audio Set: An ontology and human-labeled dataset for audio events." In: *Proc. IEEE ICASSP 2017*. New Orleans, LA, 2017.

[7] Qiuqiang Kong, Yong Xu, Wenwu Wang, and Mark D. Plumbley. *Audio Set classification with attention model: A probabilistic perspective*. 2017. arXiv: 1711.00927 [cs.SD].

[8] Qiuqiang Kong, Changsong Yu, Turab Iqbal, Yong Xu, Wenwu Wang, and Mark D. Plumbley. *Weakly Labelled AudioSet Tagging with Attention Neural Networks*. 2019. arXiv: 1903.00765 [cs.SD].

[9] *Mujer Atacada por Dogo Argentino*. https://www.youtube.com/watch?v=3xCWI_22Z9A. Accessed: 2020-01-13.

[10] Ronald W Schafer et al. "What is a Savitzky-Golay filter." In: *IEEE Signal processing magazine* 28.4 (2011), pp. 111–117.

[11] *Sound event detection in domestic environment*. http://dcase.community/challenge2019/task-sound-event-detection-in-domestic-environments. Accessed: 2020-01-13.

[12] Changsong Yu, Karim Said Barsim, Qiuqiang Kong, and Bin Yang. *Multi-level Attention Model for Weakly Supervised Audio Classification*. 2018. arXiv: 1803.02353 [eess.AS].

DECLARATION

I declare that the report has been composed by myself and that the work has not be submitted for any other degree or professional qualification. I confirm that the submitted work is my own, that my contribution and those of the other authors to this work have been explicitly indicated. Additionally the experimental work is entirely the work of the other authors and me. I confirm that appropriate credit has been given within this report where reference has been made to the work of others.

Aarhus, February 2020

Harshit Mahapatra

Patrick Lewandowski

COLOPHON

This document is a Project Report for project titled *Real-time Sound Event Detection* supervised by Prof. Ira Assent from Aarhus University Department of Computer Science, Aarhus, in Denmark.

Title	Real-time Sound Event Detection
Version:	*Final Version* as of February 23, 2020 (`classicthesis`).
Date:	February 2020, Aarhus
Authors:	Harshit Mahapatra, 201803187 Patrick Lewandowski, 201802946
University:	Aarhus University Åbogade 34 8200 Aarhus Science & Technology Department of Computer Science Master of Science
First Supervisor:	Ira Assent

Final Version as of February 23, 2020 (`classicthesis`).