Automated Detection of Seizures in Rodents

Wuichak Wong

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2018

Abstract

In order to improve the welfare of rodents, the company Actual Analytics Ltd developed the Home Cage Analysis (HCA) System for research usage, which collects only side-view infrared videos and RFID records. At the same time, detection of seizures in rodents is required for many research needs, the traditional way to detect is to detect seizures manually by specialists, that is a rather labor-intensive annotation work; and the normal automated ways for seizure detection uses EEG and ECG as their indicators, other than videos or RFID records. Thus, in this thesis, I'm aiming to use video-based methods to achieve the automated detection of seizures in rodents, based on the data collected by HCA system.

According to my research, there is no such project has the same conditions using side-view infrared video for rodents seizure detection. So, based on the methods for video-based seizure detection (on human) and the ones for rodent behavior recognition, I proposed an Detector System as the solution for automated detection for seizures in rodent. It is based on the studies of computer vision and machine learning tasks, which includes 4 parts: Preprocessing, Feature Extraction, Model Training, Detection & Evaluation.

The design of the Detector System is partly implemented and tested, that the auto-tracking is still required for the future work. As the final result, the use of video-based methods for seizure detection is preliminary tested to be practical and effective for the 'motor' seizures, but poor at 'Absent seizures'. Also some other experiments of the combinations of model settings are done to find the best detection model for the current task. Some promising improvement directions are pointed out at the end of the thesis.

Acknowledgements

This project is sponsored by Actual Analytics Ltd, who has created a home-cage analysis system and provides videos and annotations for the project.

I'd like to thank to my supervisor Prof. Douglas Armstrong and researcher Rowland Sillito, for their great help offering the technical supports and advice on the project. It's a great honor to communicate with them. And also thanks to SZ.R and ZY.C, my friends in the lab, giving me comments on the project, which greatly inspires me.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Wuichak Wong*)

Table of Contents

Chapter 1

Introduction

In this chapter, I am going to introduce the general motivation of the project, explain the project background and objective. I will stress the original contributions of the project and outline the structure of the thesis. The introduction is partly taken from my Informatics Project Proposal(IPP)

1.1 General Motivation

Epilepsy is a common neurological disorder that causes seizures in people and animals of all ages.

> ”Animal models of epilepsy and seizures, mostly involving mice and rats, are used to understand the pathophysiology of the different forms of epilepsy and their comorbidities, to identify biomarkers, and to discover new antiepileptic drugs and treatments for comorbidities.”Lidster et al. (2016)

Thus the detection of seizures is one of the most important indicators in laboratory research. And this project intends to automatically detect seizures in rodents, which can be applied to different laboratory research to reduce or replace the labor-intensive seizures annotation work.

1.2 Project Background and Objective

This part I will introduce the general project background and objective. Detailed Home-cage information will be explained in next chapter, section2.3.

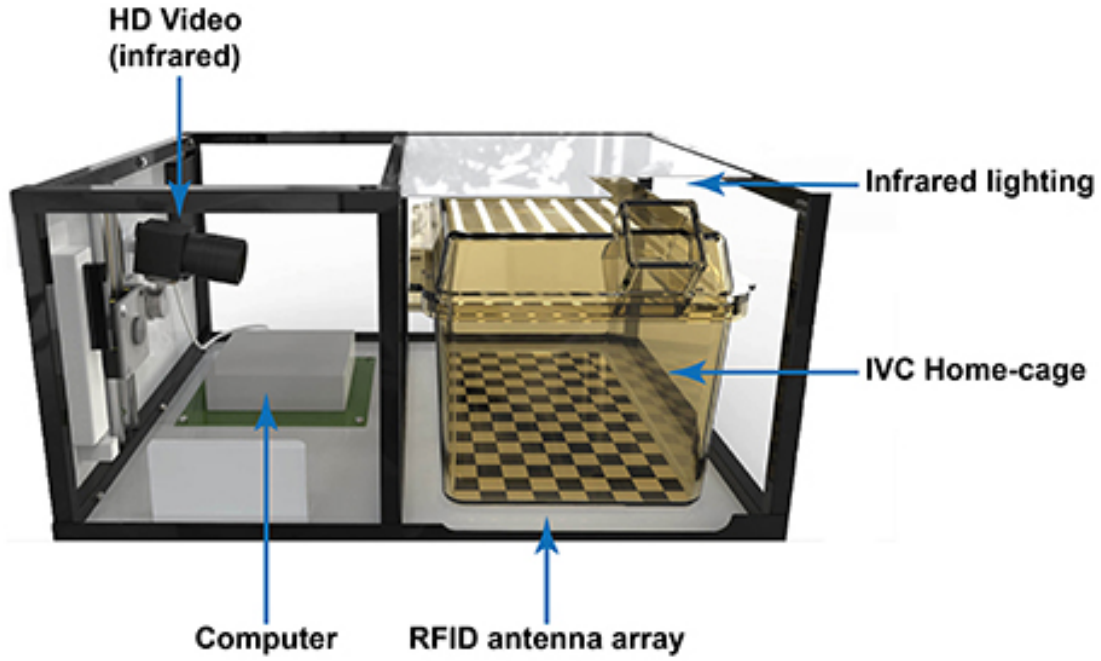

Figure 1.1: This figure shows the illustration of Home Cage Analysis (HCA) System, proposed in the research Bains et al. (2016).

1.2.1 Project Background

To be specific about the project, 'Automated Detection of Seizures in Rodents', it is raised from the development of the home-cage analysis system (Bains et al., 2016; Redfern et al., 2017), which is created by a UofE spin-out company Actual Analytics Ltd, sponsored by the NC3Rs. It is developed for analyzing individual mouse activity in group housed animals of different inbred strains, and could be used for other applications including "safety pharmacology, toxicology, circadian biology, disease models and drug discovery"Redfern et al. (2017). In this home-cage system, the animals remain in their social group, rodents are housed in cages where they eat, sleep, drink, groom, and interact socially. And the system collects RFID-based data and records IR videos, shown in figure 1.1.

According to the paper Bains et al. (2016),

> "central nervous system disorders such as autism as well as the range of neurodegenerative diseases such as Huntington's disease are commonly investigated using genetically altered mouse models."

And

> "the current system for characterizing these mice usually involves removing the animals from their home-cage environment and placing them into

> novel environments where they undergo a battery of tests measuring a range of behavioral and physical phenotypes. These tests are often only conducted for short periods of times in social isolation."

However human manifestations of such disorders are more likely to presented over long periods of time, which leads to significant social impacts.

The labor-intensive seizures annotation(detection) work is one of the problem remain in the analysis system. In order to track the actual behaviors of rodents beyond the experimenting time, the home-cage system records all the daily activities of rodents in a long period without human interaction. And the long-lasting videos and other sensor data are extracted from the record and waiting to analysis. One of the annotation work is seizures detection. Mostly the seizures can only be detected through the 24-7 videos within this home-cage system, since it doesn't record EEG or ECG of rodents, because of the welfare of animals and some other reason. So it is a time-consuming and low-efficient work for human annotators, that seizures can only happen few times in several hours of videos and it can be annotated differently by different people.

To solve the problem, we need to develop a method to automatically detect and annotate seizures for the recorded videos from home-cage. So that, the annotated video can be easily checked by human annotator, to speed up the data cleaning(annotating) process; or even better accomplish the data annotating process completely automatically.

1.2.2 Objective

As is mentioned in before 1.2.1, this project is based on the HCA system. The data we use in this project comes from the research group, collected by the HCA system, which will be detailed introduced in section 2.3. The data can be separated into three parts:

(i) Side-view whole-cage infrared (grayscale) video data;

(ii) RFID chips (height & position) data;

(iii) Annotation of videos for seizure behaviors.

However, the RFID chips (height & position) data is not available, due to some current circumstances. The project will only be able to use the video data and the annotation.

1.2.2.1 System for Automated detection of Seizures in rodents

In this project seizures are classified based on Racine stages, will be explained in section2.1. So there will be different stages of seizures. And this project should at lease detect the obvious seizures (high levels), and try to detect some lower level seizures.

The final goal is implement a practical **seizure detection system (algorithm) for rodents**, to accurately identify every specified type of seizure episode from 24hrs of video, which means all the obvious seizures should be captured by the seizure detection system. For example, after training, given a 30 minutes Home Cage monitoring videos, the system will annotate (predict) all the possible seizures. The prediction will be compared to the ground truth annotation as an evaluation.

The detection doesn't require a classification result, which means the detection result can be a sequence of binary 'true/false' tags for the chunked time-bins of videos. And the final result is represented as the list of start/end time of each episode of detected seizures during the given 24hrs videos. The result can be different, when choosing different size of time-bin, but the smallest element should larger than a 'frame-time per bin'.

1.2.2.2 True positives and False positives limitation

The whole dataset is unbalanced: there are greater number of examples for the 'Non-seizure data' than the 'Seizure data'. Thus, we can't simply use 'accuracy' as our evaluation indicator.

Since we want to detect all the specified seizures, the true positives should be high enough and the false negatives are low enough, that nearly all the seizures can't be missed.

However, we still need to reduce the false positive down. To explain more about the idea, we can make an extreme assumption: As we need a highly sensitive seizures detector(seizure detect system), we make our detector output all the examples as 'seizures', so the False-Negatives will be 'zero', and we won't miss any 'ground-truth seizures' in the dataset. However, the detector will be useless, as it won't filter-out any 'Non-seizures data'.

Thus, we need to keep the detector sensitive (high True Positive Rate) and specific (low False Positive Rate).

The way to test the true/false positive is to compare the automatic seizures detection result to the human annotation on the same dataset. Detailed evaluation method will

be explained in section 3.6.

1.3 Contributions

This project work contains original contributions, including 'designs,implementation and investigations', as followed:

(I) Design the **Detector**, the auto-detection system for 'Side-view' 'Video-based' seizure detection, including the 'Preprocessing', 'Feature extraction', 'Training Model' and 'Detection and Evaluation'.

(II) **Investigate** the performance of some existing 'video tracking' and 'object detection' methods on the actual dataset, for preprocessing.

(III) **Design** and **Implement** the **Manual Video Preprocessing Toolkit**.

(IV) **Clean up** and **Preprocess** the video dataset manually, which generate a trainable dataset in a self-proposed format.

(V) **Implementation of 'Feature extraction'**, based on the existing open-source code and company-supported toolkit.

(VI) **Implementation of 'Training Models'**, based on the LIBLINEARFan et al. (2008) open-source toolkit.

(VII) **Experiments on different combinations** of the extracted Features and Models, and **Evaluate** their **performances**.

1.4 Thesis outline

The rest of the thesis are structured as follows:

- Chapter 2 introduces the background knowledge about rodent seizures detection, and the Home Cage Analysis (HCA) system that we use in the project.
- Chapter 3 talks about the related works, which use video-based methods for seizure detection and rodents behavior analysis.
- Chapter 4 is about the general designs of solutions for the project problem, and the structure of the proposed detector system.
- Chapter 5 introduces the preprocessing implementation, which gives the cores of preprocessing methods.

- Chapter 6 is about the detector system, that includes the features extraction, detection models, evaluation methods and prediction enhancement.
- Chapter 7 is the experiments I have done for different combinations of the implemented system, to search for the best detector settings for seizure detection.
- Chapter 8 restates the work and contributions of the project, analyzes the performance of the detector system and points out the future working direction for the project.

1.5 Summary

This chapter explained the general motivations and background for the project, that using video-based methods will be a great substitute to improve animal welfare for seizure detection; and I have also introduced the contributions of the work which includes the implementation of the solution to the task and the experiments for the better solution.

Chapter 2

Background

In this chapter, we will talk about the definitions for rodent seizures, some seizure detection methods and the details about Home Cage Analysis (HCA) System and my project data information. Introductions in this chapter is partly taken from my Informatics Project Proposal(IPP).

2.1 Seizures and Racine stages

In the paper Fisher et al. (2005), seizures is defined as:

> **An epileptic seizure is a transient occurrence of signs and/or symptoms due to abnormal excessive or synchronous neuronal activity in the brain.**

For epileptic seizures, "this statement concerning the electrical activity of the brain during the episode appears to be necessary in the definition, but difficult to apply in clinical practice Fisher et al. (2005)". According to International League Against Epilepsy (ILAE) Fisher et al. (2017), we can simply classify seizures as 'Motor' and 'Non-motor'. 'Motor' seizures are the kinds that abnormal neuronal activities involve the motor system, which make the subject behave abnormally and can be captured from appearance. Vice versa, 'Non-motor' seizures do not affect motor system, and is hard to detect without specific sensors. So, in this project, we intend to detect 'Motor' seizures .

In the book *Models of Sizures and Epilepsy* Pitkänen et al. (2017), seizures can present as short loss of awareness (staring and disruption of ongoing normal activity), automatisms (undressing, chewing, repetitive movement, etc.), sensations (visual, sensory, auditory, olfactory, or gustatory), or motor phenomena. Motor events can be very

brief muscle jerks, uncontrolled rhythmical movements of limbs, body stiffening, or sudden loss of muscle tone.

For seizures in rodents, 'Racine stages' are a most accepted categorization of epileptic seizures proposed by Ronald J. Racine in 1972 Racine (1972). In our project, we will detect seizures of rodents based on 'Racine stages', five observable actions stages:

(i) Mouth and facial movement;

(ii) Head nodding;

(iii) Forelimb clonus;

(iv) Rearing with forelimb clonus;

(v) Rearing and falling with forelimb clonus.

According to Ronald's work Racine (1972), when the level of stimulus for brain neurons increases, the result of involuntary movement goes down the level of stages; levels further down the Racine stages also contain symptoms of the previous stages. For example, the rat which is demonstrating the actions of a stage four seizure may also demonstrate head nodding (indicative of a level two seizure). So the higher Racine stage the seizure is, the easier it should be detected.

2.2 Seizure Detection

As said in the previous chapter (Chap 1), seizure detection is commonly used in different laboratory research areas, including safety pharmacology, toxicology, circadian biology, disease models and drug discovery. Therefor, lots of automatic seizure detection methods have already existed.

In the paper Ramgopal et al. (2014), it is concluded that

> "All seizure detection algorithms involve two main steps".

First is the **'Feature computation'**, which collects appropriate quantitative values or features. Values and features can be Electroencephalography(EEG) features, movements, or other biomarkers that can be computed from data, based on research theories or models. In this step, we can do several things:

(i)processing or filtering;

(ii)computation;

(iii)feature reduction or extraction.

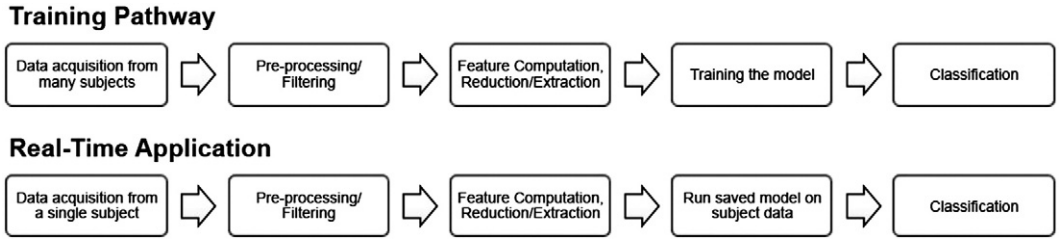

Figure 2.1: This is the general idea of detection methods process summarized from the paper Ramgopal et al. (2014). Steps involved in feature extraction and classification for seizure detection and prediction. Seizures are detected using a sequential process of steps. During the training phase, features are computed from a large number of subjects to optimize the model. During the implementation phase, real-time data are acquired from an individual subject, and features are computed. Data from the saved model are applied for classification.

The second is **'Classification'**. Classification needs a classifier, which is a model with threshold for the mathematical combination of the values and features. To make the detection result better, we try different features, thresholds and models; or we can train them using the idea of machine learning. Figure 2.1 shows the normal process of detection and the one for real-time application.

Methods for seizure detection can be vary. If we classify the methods for seizure detection by their feature source, they can be sorted into three main class(List from the mostly used ones):

(i) **EEG**-based - Using features extracted from Electroencephalography;

(ii) **ECG**-based - Using features extracted from Electrocardiography;

(iii) **Others** - Features extracted from other source of data, including but not limited to different movements information, videos, Sound recording, In-planted Sensor Chips or other biomarkers.

2.2.1 Electroencephalography - EEG

Electroencephalography (EEG) is mostly used for seizures detection, for the abnormal neuronal activities, basis definition of seizures, are more directly to analysis. It is a kind of representation of brain electrical activity measurements, research in the paper Iasemidis (2011) for example. EEG contains one of the most valuable sources of information for epilepsy and seizures research, but rich resource may still be underutilized Ramgopal et al. (2014). However, EEG requires equipment and sensors to

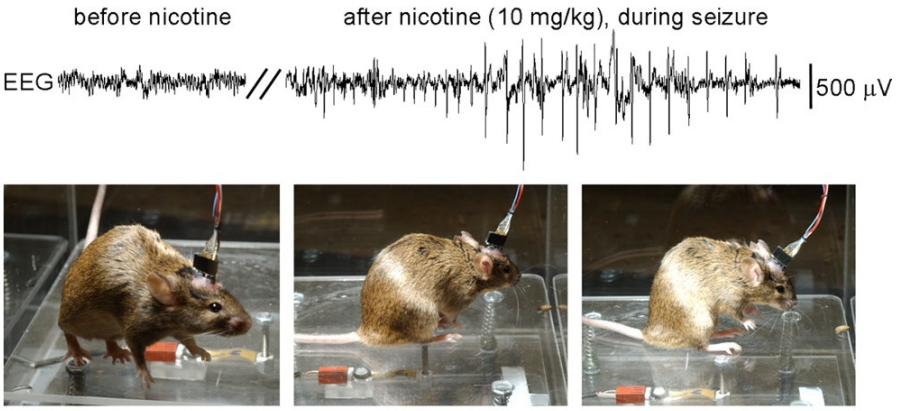

Figure 2.2: This is an example of seizure detection method, using Electroencephalography(EEG), taken from the paper Fonck et al. (2005), with an slight edition.

extract brain electrical signals, which is not easy to apply and influences the welfare of experimented subjects. An example of applying EEG on rodents seizure detection is in the figure 2.2.

2.2.2 Electrocardiography - ECG

Electrocardiography (ECG) is the second most used approach, which analysis the changes in heart rate and conduction of subjects, for example Jansen and Lagae (2010). Since epileptic seizures can cause short-term and long-term heart rate disturbances, it can be used for seizure detection. ECG has the same limitation as EEG with the equipment problem, that the equipment and sensors applied on the rodents might greatly influence their welfare and limit their daily movements. An example of applying EEG on rodents seizure detection is in the figure 2.3.

2.2.3 Other Methods

Besides EEG and ECG, Accelerometry [Nijsen et al. (2005)] and Video detection systems [Cuppens et al. (2010), Karayiannis et al. (2001) and Cuppens et al. (2010)] can be used in seizure detection.

In figure 2.4 is an example for using in-planted chips to collect accelerometry data for seizure detection. And also figure 2.5 for seizure detection based on video data.

To be more specific, video data can be range data or normal image pixel sequences. These two methods aim to analysis the motion elements of actual behavioural pheno-

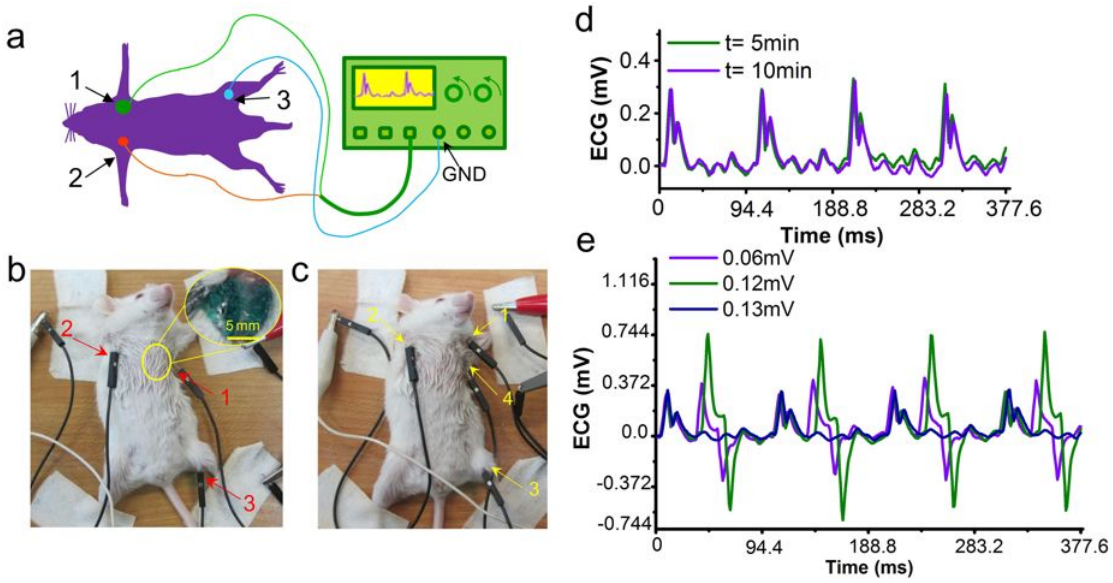

Figure 2.3: This is an example of seizure detection method, using Electrocardiography(ECG). The figure is taken from the paper Jin et al. (2013).

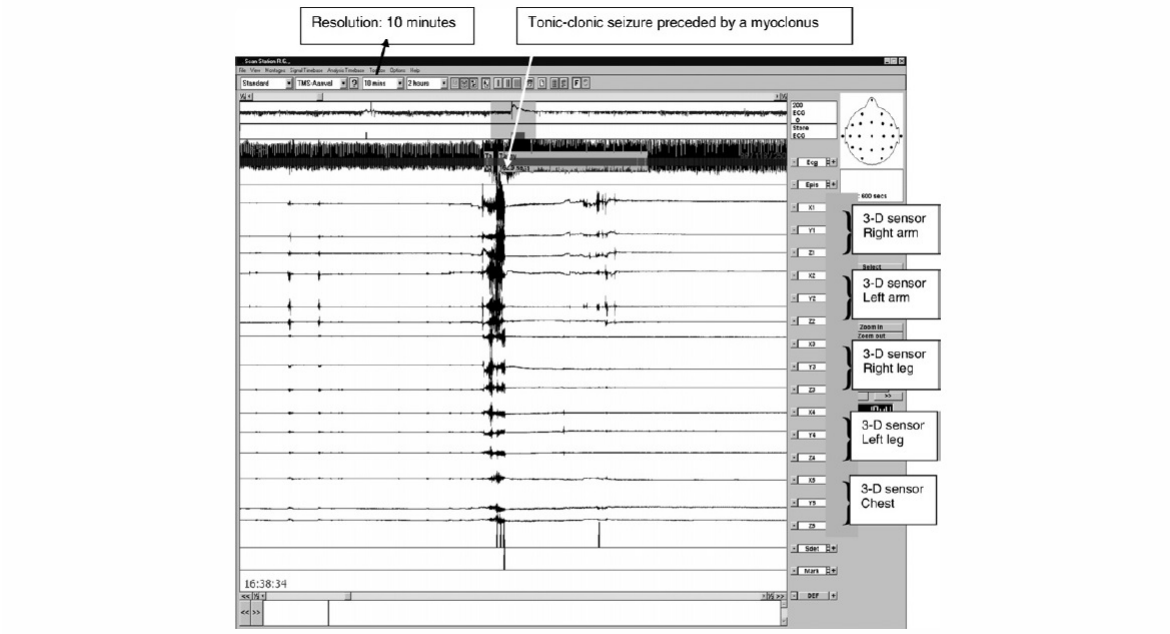

Fig. 1. Examples of specific seizure patterns that can be observed in ACM data. Shown here is a myoclonic seizure that evolved into tonic–clonic contractions.

Figure 2.4: This is an example of seizure detection method, using in-planted chips extracted Accelerometry, taken from the paper Nijsen et al. (2005).

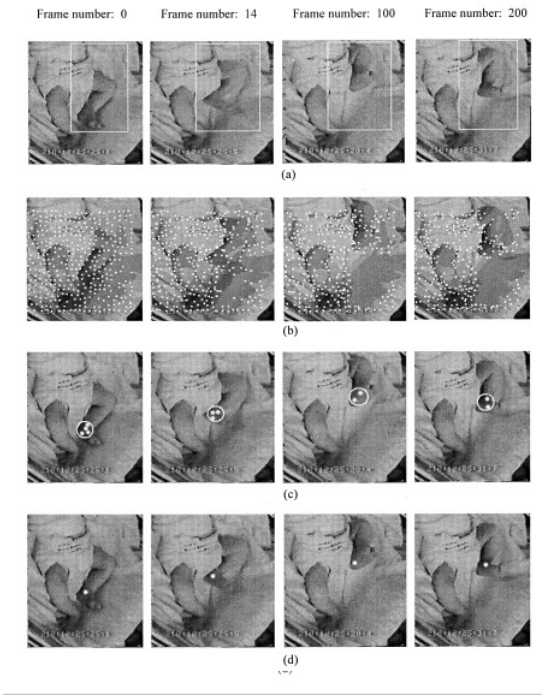

Figure 2.5: This is an example of seizure detection method, using video data, taken from the paper Karayiannis et al. (2001).

typing, such as velocity, area, angular speed, and duration.

As for Video detection systems, it only requires videos of the experimented subject. With the development of computer vision, video analysis methods, such as object tracking and activity classification, are powerful and largely used for the detection. Videos can be captured from different angle and motion elements can be captured, based on the set model.

However, motion data can be various and similar between activities. The existed approaches can only analysis a limited set of seizure types so far (Pediaditis et al., 2012), the detect results are worse than EEG/ECG, and each of them requires a well-defined motor activity model to approach the actual seizure detection results using the captured data.

However, they are still promising methods, that the data they required are much easier to measure than EEG/ECG.

According to the paper Pediaditis et al. (2012), video detection systems for seizures have been developed by groups of work, but most of them are marker-based, that some parts of the subject (hands, joints) in the video have already marked, so that the system can detect the specific interested points in the video easily using simple techniques

in computer vision area. Also, in my research, most of the video detecting systems for rodents are simply for their daily and social behaviors, not specified on seizures; others non-mark-based detecting systems are not for rodents, and the camera views are more likely the 'top-down' ones. They are useful but not directly comparable to our project condition. Next section we will introduce our project system for data collecting - Home Cage Analysis (HCA) System.

2.3 Home Cage Analysis (HCA) System & Project Data

This part we will generally explain the HCA system(Actual Analytics Ltd, UK) we will get involve in the project.

The HCA system is proposed in the paper Bains et al. (2016) and figure 1.1 shows the illustration of the HCA system, as is explained in the project background (section1).

The HCA system is designed to "fit into two rack spaces of a standard IVC rack". One half contains the computer, infrared camera and the power supplies, for monitoring cage of mice. The other half "comprises an RFID reader baseplate with antennae located on predetermined locations. This provides the base for placing the individually ventilated mouse home cage." And there is also a infrared light source above the mouse home cage, "allowing for continuous video capture without compromising the quality of the image".

The HCA system collects 2 main kinds of data:

(i) Infrared Video Data; (ii) RFID Data.

I am going to explain them more in the next sections.

2.3.1 Infrared Video

The HCA system can provide side-view whole-cage infrared video. With the help of infrared LEDs providing 'lights', the system is able to continuous collect the '24-7' video. A USB 3.0 camera with matched 4.5mm lenses and daylight filters (700 nm cut-off) is used to capture the infrared (grayscale) video at 25 fps at HD (720p) resolution.

Figure2.6 shows a single frame of the raw infrared video collected by HCA system.

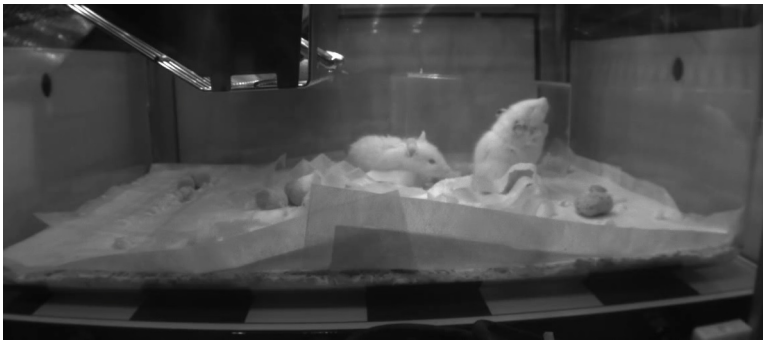

Figure 2.6: This figure shows a single frame in the infrared video collected by Home Cage Analysis (HCA) System, proposed in the research Bains et al. (2016). The HCA system provides a side-view whole-cage infrared video. With the help of infrared LEDs providing 'lights', the system is able to continuous collect the '24-7' video. A USB 3.0 camera with matched 4.5mm lenses and daylight filters (700 nm cut-off) is used to capture the infrared (grayscale) video at 25 fps at HD (720p) resolution.

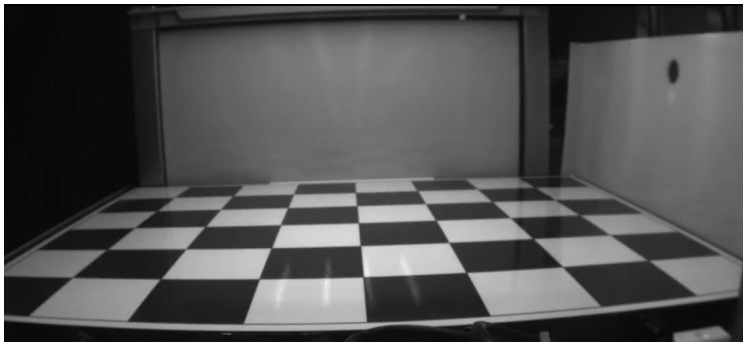

Figure 2.7: This figure shows the real RFID platform in the gray scale infrared video in Home Cage Analysis (HCA) System, proposed in the research Bains et al. (2016).

2.3.2 RFID Data

There is "a low profile base-plate that contains a 2D array of 18 RFID antennae (in a 3×6 array) directly beneath the home-cage", shown as the RFID antenna array in figure1.1. The RFID reader baseplate will create a electromagnetic field. At the same time, RFID chips will be injected into the groin of the subject mice, so that the RFID reader baseplate can capture the **position** and **height** of chips, which could be used for analyze the behaviors of mice.

Details can be found in figure2.8, which shows the schematic overview of RFID transponder activation and detection by baseplate.

2.3.3 Introduction to Project Data

As said before, this project only have two kinds of data:

(i) Infrared Video Data;
(ii) Seizure Annotation.

Video data in this project, is 5 chunks of side-view infrared '.flv' video collected by Home Cage Analysis(HCA) system Bains et al. (2016). Each video lasts for an hour, sum up to 5-hours video, which contains a continuous monitoring data for the same 2 mice, as shown in figure 2.6. The 5-hours video is not preprocessed and not selected, it can generally represent the normal living activities of the two mice. The video doesn't have sound, so that it only a sequence of frames:

$$5(\text{hour})\times 60(\text{min/hour})\times 60(\text{sec/min})\times 25(\text{frame/sec}) = 450{,}000(\text{frame}).$$

Annotations for seizures are '.csv' files for each video chunks. And there are 16 types of status recorded:

> 'Clonic seizures'; 'Straub tail'; 'Grooming'; 'Loss of righting reflex'; 'Absent seizures'; 'Myoclonic jerk'; 'Hunched'; 'Excessive chewing'; 'Lordotic posture'; 'Generalised full motor seizures'; 'Death'; 'Rearing (seizures)'; 'Rearing and falling'; 'Forelimb clonus'; 'Head nodding'; 'Tonic seizures'.

The annotations are recorded every 40ms ($= 1000(ms)/25$), which means each annotations correspond to each frames.

Also the annotation is recorded as a list of binary '0/1' for each record for the 16 status, as shown in the figure 2.9.

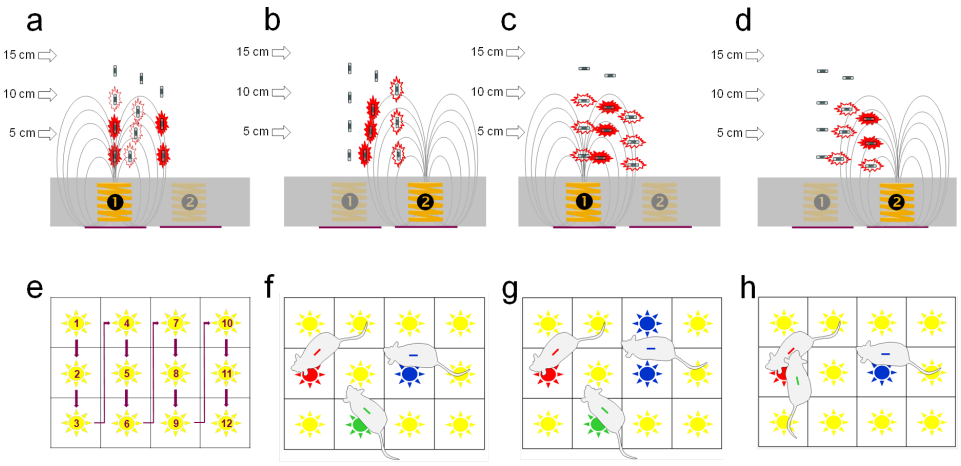

Figure 2.8: **This is the Schematic overview of RFID transponder activation and detection by baseplate**Redfern et al. (2017).

Upper panels: Side view of baseplate showing 2 adjacent antennae ('1' and '2') with figurative magnetic field lines generated by active antenna (indicated by brighter color). The thickness of the jagged red outline around the RFID transponder indicates signal strength, generated by magnetic induction; this is maximal when aligned with the magnetic field direction. Each antenna is activated for 75 ms. (a; b) RFID transponder in horizontal orientation in various positions above the baseplate; signal strength is generally highest in between antennae. (c; d) RFID transponder in vertical orientation in various positions above the baseplate; signal strength is highest directly above an active antenna or where magnetic field lines are returning towards the baseplate in a near-vertical plane between adjacent antennae. In this way, there are no 'dead spots' across the baseplate. Lower panels: Birds-eye view of the baseplate. (e) Read sequence of the 12 antennae; total scan cycle takes 1080 ms. (f) Rats are detected by the nearest antenna, reading the ID and temperature from the RFID transponder. (g) At intermediate positions, an animal (blue) may be detected by two (or more) adjacent antenna. This can cause 'flickering' of the motion detection between the two antennae when the animal is virtually stationary, minimized by applying a basic filtering algorithm. (h) Two animals (red and green) close to the same antenna, resulting in detection of the stronger signal (red) and temporary drop-out of the RFID signal from the other individual (green). https://doi.org/10.1371/journal.pone.0181068.g002

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	t	Clonic seizures	Straub tail	Grooming	Loss of righting reflex	Absent seizures	Myoclonic jerk	Hunched	Excessive chewing	Lordotic posture	Generalised full motor seizures	Death	Rearing (seizures)
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	40	0	0	0	0	0	0	0	0	0	0	0	0
4	80	0	0	0	0	0	0	0	0	0	0	0	0
5	120	0	0	0	0	0	0	0	0	0	0	0	0
6	160	0	0	0	0	0	0	0	0	0	0	0	0
7	200	0	0	0	0	0	0	0	0	0	0	0	0
8	240	0	0	0	0	0	0	0	0	0	0	0	0
9	280	0	0	0	0	0	0	0	0	0	0	0	0

Figure 2.9: This figure shows the annotations format in the '.csv', used in this project. The first column keeps the millisecond of annotiation, and the rest columns use binary '0/1' to record 16 different types of status: 'Clonic seizures'; 'Straub tail'; 'Grooming'; 'Loss of righting reflex'; 'Absent seizures'; 'Myoclonic jerk'; 'Hunched'; 'Excessive chewing'; 'Lordotic posture'; 'Generalised full motor seizures'; 'Death'; 'Rearing (seizures)'; 'Rearing and falling'; 'Forelimb clonus'; 'Head nodding'; 'Tonic seizures'.

It needs to be stressed that the annotations we here regard as 'Golden Standards', are marked by human specialists, without EEG/ECG support, only by watching the monitoring infrared videos. The start/end time for the seizures would contains some variances by human annotator and differences between specialists.

And also, the annotations are not specified for the specific subject(mouse), that it only marked as 'True' whenever there's a referred types of seizures happens in the video, without telling the subject ID. This might, or not, influence the further training accuracy.

2.4 Summary

In this Chapter, I have introduced the project background, the basic seizures definition and detection methods; also the details of the project resource, HCA system, and the data format of the project.

Chapter 3

Related Work

In this chapter I am going to introduce the relative work which are developed and used in the project. Part of the introduction is taken from my Informatics Project Proposal(IPP).

3.1 Detection System

As is introduced in 2.2, concluded by paper Ramgopal et al. (2014),

> "All seizure detection algorithms involve two main steps".

First is the **'Feature computation'**, the second is **'Classification'**. In this project we follow the idea of the basic process structure of seizure detection system, seen in figure 2.1.

Feature Computation collects appropriate quantitative values or features:

(i)processing or filtering;
(ii)Feature computation;
(iii)feature reduction or extraction.

Classification, it needs a classifier, which is a model with threshold for the mathematical combination of the values and features. This can be the simple linear regression functions, Support vector machine (SVM), Neural Network (NN), Hidden Markov Model(HMM) or other machine learning classifier. We train the features that extracted before into the classifier, and use the classifier for the further prediction.

In my research, even though, I didn't find a paper using video data for seizure detection in rodents, there are studies for video-based human-seizure detection and rodents behavior recognition (Ramgopal et al., 2014; Nijsen et al., 2005; Cuppens et al., 2010;

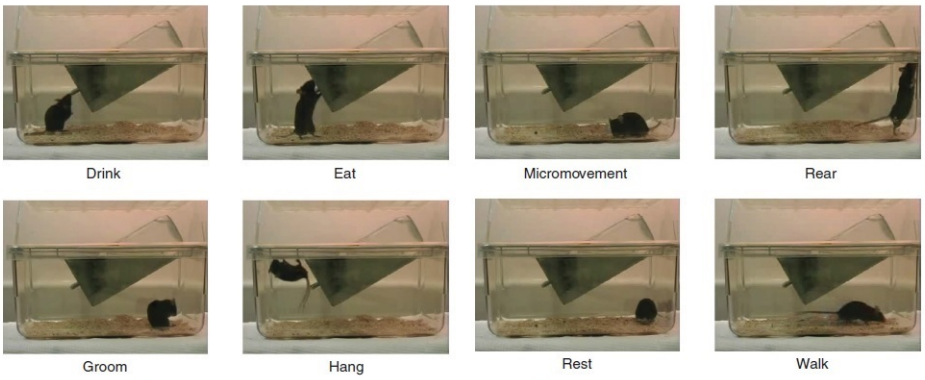

Figure 3.1: This is an example of mice home-cage behaviors, studied by the group (Jhuang et al., 2010).

Karayiannis et al., 2001; Cuppens et al., 2010). According to the review (Pediaditis et al., 2012), almost all the video-based seizure detection project treat the 'seizure problem' as a normal behavior recognition problem. They use different methods to collect their interested features from video and build up models for detection.

In the following sections, I'll introduce the relative methods used for seizure detection and behavior recognition, which can be applied in this project.

3.2 Behavior Recognition

Before go into the details, I am going to generally introduce the Behavior Recognition, to be specific, **Short Term Action Recognition**. This introduction is based on the course slides of Advanced Vision, in the University of Edinburgh, presented by professor Bob Fisher (Fisher, 2018).

Short Term Action Recognition problem focus on the behaviors that we can recognize in a short term of time, within seconds. For example, shaking, running, swimming and etc.. Or even in a long term activities, we can break down into different behavior: when playing soccer, behaviors can be shooting, heading or running. In rodents home cage activities, this can be drinking, eating, grooming, hanging, rearing, resting or walking, studied by the group (Jhuang et al., 2010), seen in figure 3.1.

The recognition problem can be divide into three types (Fisher, 2018):
(An example for detecting activities in football match, seen in figure 3.2)

(i) Near field (bodies 300 pixels high), we need to Use geometric model;

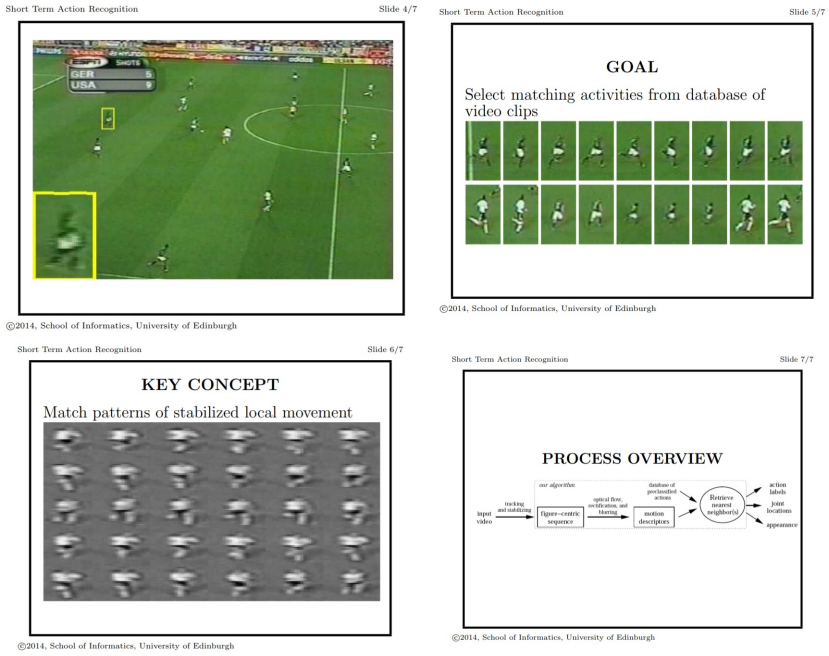

Figure 3.2: This is the slides from Advanced Vision course (Fisher, 2018), which gives general idea of short-term action recognition and the process overview.

(ii) Far field (bodies 3 pixels high), we can only use tracking;

(iii) medium field (bodies 30 pixels high), use Motion Descriptors.

In this project, our object (mouse) size is the medium field size, range from 80 to 300 pixels high. So I will later introduce only behavior recognition methods for medium field problem.

As shown in the figure 3.2, the example process is comparable to the previous seizure detection process, figure 2.1:

(i) preprocessing: tracking and stabilizing ;

(ii) feature extraction: optical flow, rectification and blurring (motion descriptors);

(iii) classification: retrieve nearest neighbor(s).

This is just a practical process for this kind of behavior recognition problem. In the following sections, I will introduce some other methods for preprocessing, feature extraction and classification.

3.3 Preprocessing

This part I am focusing on preprocess methods for video data, based on the behavior recognition process, which is the main problem in the project.

The purpose of preprocessing, is to reduce the dataset size, and generalize the centralized stable image sequence of the detecting subject, which includes methods such as sampling frames, reducing resolution, background substraction, object detection and tracking methods.

3.3.1 Methods for Reducing Dataset Size

As video data is a set of frame by frame images, we can reduce the data within or through time: Sampling Frames, Reducing Resolution.

Sampling frames means we can sample the frames through timeline, because we don't need to use all of the frame by frame data, which can reduce computation complexity. For example, we can just analyze the data every 5 or 10 frames, and get rid of the rest frames.

Reducing Resolution is the way to find the interest points within the frames. In the project, we have video resolution of 1200×500, but the single mouse in the frame only occupies range from 80 to 300 pixels high (bounding box size); so, if we can detect the mouse's position (pixels of images in frame), we can reduce a lot useless data. Also the idea of reducing resolution can be the concept video compressing, that we don't need too much detailed information for recognizing. Figure 3.3 shows different resolution of image of mouse in the project.

3.3.2 Methods for stabilizing Image Sequence

Besides reducing dataset size, stabilizing image sequence is more important for the behavior detection. This step will greatly influence the qualities of motion descriptors (features) in the next part of process.

Because the video in the project is a side-view whole-scene type, collected by the fixed camera, we need to try make the center of object fixed at the center of 'processed' video, seen as the examples in figure 3.3. In order to do so, we have to create the Bounding Box (BBox) for the mice (subject), which involves background substraction, object detection and tracking methods can be applied.

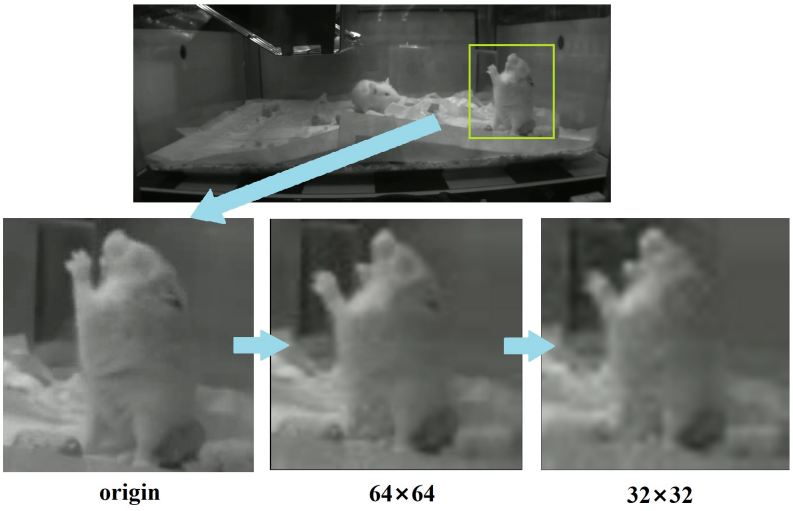

Figure 3.3: This figure is an example of different resolution of image of mouse in the project, extracted from the original video. And the size of data is compressed from '500×1200' to '32×32'.

The general process for generating the stabilized video can be explained as followed:

(i) **Background Subtraction**. Use video frames to do substraction with a background, that the background can be a fixed or dynamic one. Subtraction methods can vary in detailed implementation.

(ii) **Object Detection**.
After the substraction, we can also convert it into binary image using threshold, . Then we find the large connected region as our detected object.
Other methods such as **Edge Detection** or simply **Color Threshing**, can also be used without background substraction, to detect the object in a single frame image.
In this stage, technically we can already 'draw' the Bounding Box (BBox) of the object (mouse).

(iii) **Tracking**.
Based on the frame-by-frame detected object BBox, we have a sequence of object information in each frame, but they are not connected. If there are more than an object detected, we need to know which object in the previous frame refers to the object in the next frame. Tracking use color histogram, motion models, combines the probability models, **Kalman filter** for example.
Tracking can also be done without object detection, that we can extract feature points in the frames, such as Scale Invariant Feature Transform

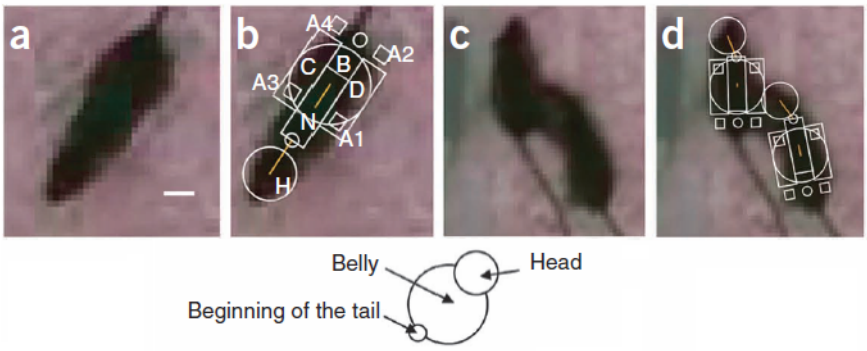

Figure 3.4: This figure shows a mouse model for feature extraction (De Chaumont et al., 2012)

(SIFT), and match the corresponding feature points.

These are only some but not all of Tracking methods that can be used as preprocessing in this project.

3.4 Feature extraction

After we obtain the stabilized video, the next step is to extract features.

When we use video data for seizures detection, features can be sorted as tracking features and motion features (van Dam et al., 2013).

Tracking features are features generated from the preprocessed tracking position result for each subject, such as velocity, area, angular speed, and movement duration. It is widely used in different studies van Dam et al. (2013); Jhuang et al. (2010); Ohayon et al. (2013). Once the parts of mice can be recognized and tracked, such as head, limb and tails, we can also set up models to detect the angle and distance between each mouse's body parts (De Chaumont et al., 2012), to figure out more useful featuers, seen in figure 3.4.

Motion features are the features extract from space-time interest points: the pattern of apparent motion of objects, surfaces, and edges in a visual scene. There are lots of widely used features in CV, such as histogram of oriented gradients (HOG) (Dalal and Triggs, 2005), histogram of optical flow (HOF) (Horn and Schunck, 1981) and scale-invariant feature transform (SIFT) (Lowe, 2004). HOG, HOF are similar, these both use the idea of **Optical Flow**, and then generate their own interested 'features', in figure 3.5 shows the idea of optical flow. And here, in video processing,

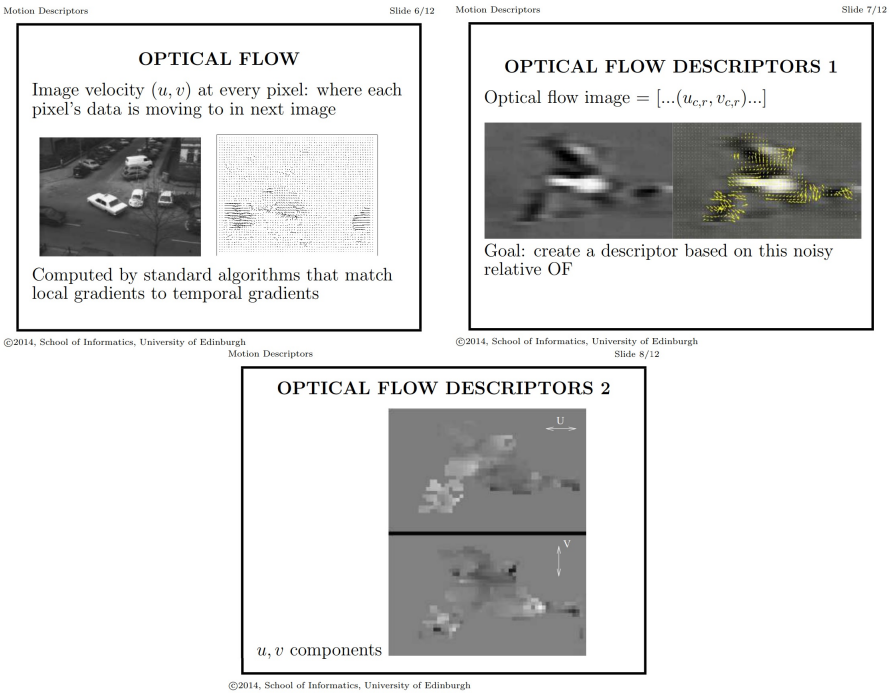

Figure 3.5: This figure is an introduction to Optical Flow, given in Advanced Vision course (Fisher, 2018). Optical flow records the **Image velocity** (u, v) at every pixel: where each pixels data is moving to in next image. 'Slide 7/12' shows the visualized optical flow descriptors, and 'Slide 8/12' shows the cleaned two direction of optical flow descriptors.

I am going to introduce two more improved methods for extracting motion features: *Spatial-temporally binned Histogram of Orientation Gradients (HOG)* (Uijlings et al., 2015) and *Local Trinary Patterns* (LTP) (Yeffet and Wolf, 2009).

***Spatial-temporally binned Histogram of Orientation Gradients* (HOG)** (Uijlings et al., 2015), shown in figure 3.6, is a useful method to extract HOG (motion feature). Since HOG is kind of widely used Bag-of-Words for video and counts occurrences of gradient orientation in localized portions of an image. This speeded up method of HOG can be promising for this project.

***Local Trinary Patterns* (LTP)** (Yeffet and Wolf, 2009) is the extension of Local Binary Patterns (LBP). It is an efficient method that can be computed online, suitable for long term video analysis and is said to meet state of the art performance. It extract the feature as trinary patterns : {-1, 0, 1}. And it doesn't require optical-flow computation. And only needs the pixel-patch histogram substraction within the local time bin. And the method won't be effect by the poor optical-flow computation result.

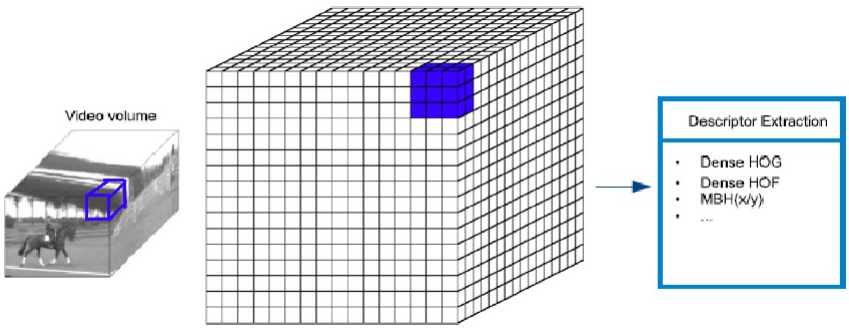

Figure 3.6: This figure is collected and edited from paper Uijlings et al. (2015). The figure shows the way to extract Spatial-temporally binned features. It treat the video as a 3-D block. Based on the pre-set block size, the pixels matrix get chunk into blocks, then the blocks in a video volume can be reused for descriptor extraction. In the paper, their descriptors consist of 3×3 blocks in space and 2 blocks in time, shown in blue (color figure online).

3.5 Classification and Detection

After we have extracted the features, we can use the features for seizure classification and detection.

Use the features, we can build up different detector (classification) :

(i) we can simply divide the bags-of-word features into time bins as input, and the target is the manual annotation; so that we can use machine learning methods to solve the classification problem, including Neural Network(NN) based methods such as convolutional neural network(CNN) (LeCun et al., 1998), or support vector machine(SVM) (Suykens and Vandewalle, 1999).

In most of the behavior recognition and video-based seizure detection methods, SVM is one of the most used method. And note that, LIBLINEAR (Fan et al., 2008) is a fast SVM-based classification toolkit used by many research (Uijlings et al., 2015; Redfern et al., 2017).

(ii) Or maybe the problem can be treated as motion tracking problem, that we set up daily activity models for mice. Since other research have tracked the behaviors of mice well (Burgos-Artizzu et al., 2012; Jhuang et al., 2010), if well applied in this dataset, we can try to build up the activity-transition probabilistic model and use the idea of Hidden Markov Model(HMM) for motion detection (Yamato et al., 1992; Jhuang et al., 2010).

3.6 Evaluation

Two methods will be the main evaluation method of the project: Cross Validation; Sensitivity and Specificity measurements.

(i) Cross validation is used for separating the given dataset into training/validation. If nothing else, then we will use K-fold cross-validation. Since the trainable (manually annotated) data is limited, we use cross validation to overcome the lack-of-data issue (to hypothetically enlarge dataset).

(ii) Sensitivity and Specificity measurements, or in another word Receiver operating characteristic (ROC), will be used as the main evaluating indicator. We assume the human annotation reflects the ground-truth for the rodents seizure episodes. As we have the detection results from the implemented system, we can calculate the correlation between auto-detecting result and human annotation.

Figure 3.7 shows how to calculate the sensitivity and specificity. And in this project, we want the sensitivity approach to '1' and specificity $(1 - FalsePositive)$ higher than the limit.

3.7 Summary

This chapter, I have introduced the relative work that can applied into the project:
(i) the general idea of detection system; (ii) introdution to the relevant problem in Computer Vision (CV) area, behavior recognition; (iii)preprocessing methods; (iv) feature extraction methods; (v) classification methods in machine learning area; (vi) and the evaluation methods.

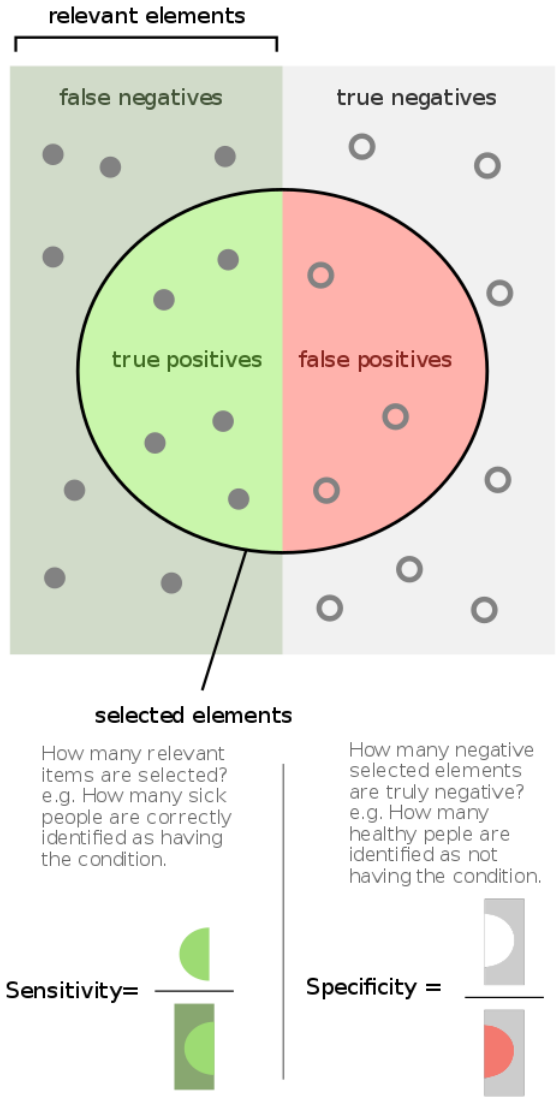

Figure 3.7: This figure shows the idea of sensitivity and specificity measurements.

Chapter 4

Detector System Designs and Structure

This chapter I am going to introduce the designs for the automated detection system for seizures in rodent, based on the current situation of project. I'll compare the project to other research and give the design structure of the project. Actual implementation will be explained in the later chapters.

4.1 Comparison and Analysis

In the previous chapters, the three figures 1.1, 2.6 & 2.9 have shown the present situation of the project.

There are only 5-hour raw infrared home-cage monitoring 'flv' videos, collected by HCA System, for the two mice (in the figure); and the corresponding annotations for seizures saved as 'csv' files.

The video is of resolution-500$\times$1200, gray-scale and side-view. And the mice size range from 80 to 300 pixels high, as it move from the back of video-view to the front, seen in figure 4.1.

The data is gray-scale video, which increase the difficulty for tracking. And as we can see, the mice in the project are **white** ones, which is **similar to the background color**. Also we found strong reflection on the sides of home cage, recorded in video. These will cause problem for tracking.

In my research results, most of the seizure detection methods use video source only for supporting EEG/ECG-based detection, not serve as the key 'feature'; and most of the video data are 'top-down view', that the shape and size of the mice(rodents) have

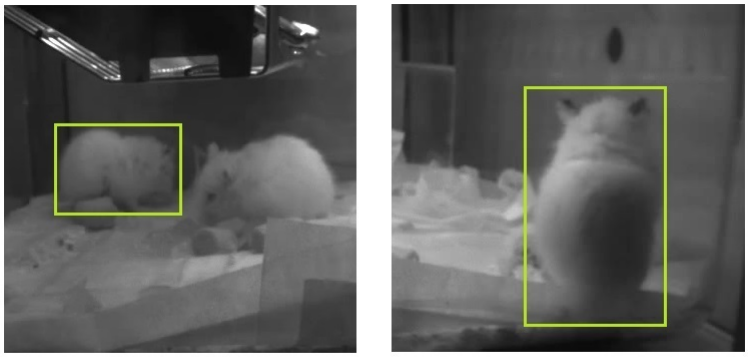

Figure 4.1: This figure shows different size of mouse in the video, changing with the position. Range from 80 to 300 pixels high.

little variation; other 'side-view' video methods are 'color' video for rodents social behavior recognition, and I have only found few human-seizure-detection researches using 'side-view' 'gray-scale' video. **So there is no directly comparable research, that I can develop from.**

Besides, in the annotations, there is no mark to distinguish the two mice(subjects). They only record whether there is a seizure happening in the video. This requires us to do some cleaning job to make the seizure annotations match the tracked video.

All the video-based system are similar and based on behavior recognition, using features train in classification models.

Thus, in this project, I use the same structure for seizure detection:

(i) Preprocessing; (ii) Feature Extraction; (iii) Model Training & Evaluation; (iv) Detection.

4.2 System Structure

Figure 4.2 shows the design for detection system for the project. Though RFID data for the corresponding videos is not available at the moment, I still leave the interface for it.

Preprocessing aims to align the different source of raw data, and make it easier to extract features in the next step. As for video data, we need fixed-size tracking videos for the rodents, and it needs to be stabilized and centralized. This involves the tracking problems in CV areas.

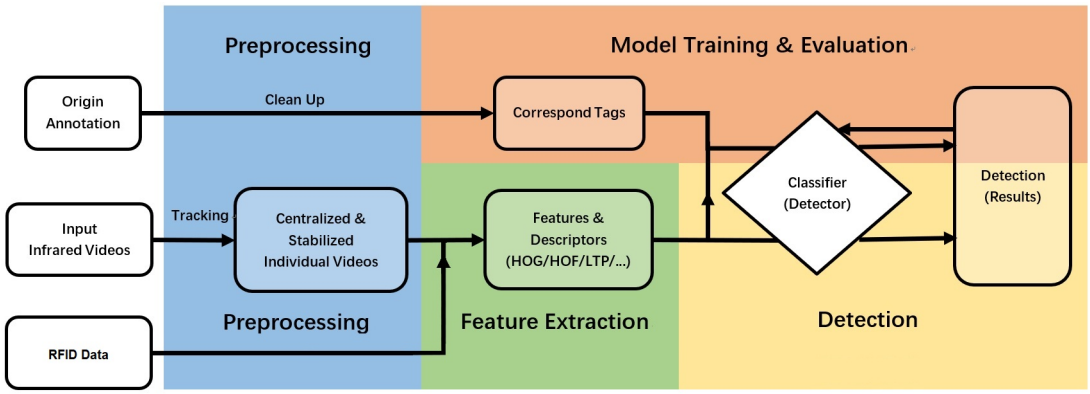

Figure 4.2: This figure shows the design for seizure detector system, in this project situation. Assume we can treat the processed RFID Data as a kind of feature, methods different from video data.

Feature Extraction aims to sort the preprocessed data into a structured format, and train the features in classifiers, widely done in Machine Learning area. Motion features such as HOG/HOF features are most used in behavior recognition problem; tracking features can also be extracted from video and RFID data, combining these features might train the detector with a better result.

Model Training uses the ideas from machine learning, that we train the features and annotations together. And the simplest way is to use SVM. Neural Network can also be applied. However, more annotations for daily social behaviors might required, if we want to train Hidden Markov Model (HMM) better.

Evaluation methods gives us ideas about how good the Detectors are, so that we can choose the best detector as our final model for auto-detection.

Detection is the final product, that given new Video/RFID data; generate the features using the same preprocess and feature extraction methods; and use the chosen Detector to make prediction.

Since the main purpose of the project is to solve the problem from the labor-intensive seizures annotating work, all the methods mentioned above should be automatic.

However, in the actual implementation, due to the limitation of time, the final version for **preprocessing** is not automatic. I will explain more in the next chapters.

4.3 Environment Setup

Since the project is totally new, it doesn't set any conditions for setting up the environment. As the company mostly uses python as the coding language and it is widely used in Machine Learning fields, I try to make python as my first choice, but I still consider other programming environments to support the project.
And finally, my environments for the project are as below:

(i) Operating System: Windows 10 (Microsoft);

(ii) Preprocessing: Python2 & OpenCV / Python3 & OpenCV, all coding in IDE-Spyder, built on Anaconda;

(iii) Feature Extraction: Matlab 2015a / Python2 & OpenCV;

(iv) Model Training & Evaluation: Python3, jupyter notebook.

4.4 Summary

This chapter, I have restated the conditions of the project, and pointed out some of the problem might exist in the preprocessing part. And then, I have given the general and practical designs for the project. In the following chapters, I will talk about the implementations of the project.

Chapter 5

Preprocessing Implementation

In this chapter, I am going to introduce the implementation for preprocessing methods. As is explained before, preprocess includes 2 parts: one for the annotation, the other for the videos. And its aim is to reduce the dataset size and make it easier to extract features. To point out that, the automatically tracking method for video data isn't successfully achieved in the project, so I have slightly change the process to a manual annotation tool, and leave the auto-preprocess as a future work.

5.1 Implementation plans for Preprocessing

Before I started on the preprocessing part, I have looked up the 5 hours video, and here are some findings:

> The 'seizure' data is much less than the 'non-seizure' data. And it is a pretty unbalanced dataset;
>
> The two mice doesn't have seizures together at the same time, which means when we do the training;
>
> The two mice move from side to side rapidly, and the pixel-sizes for the mice are always changing.

After that, I designed a plan for the preprocessing:

> (i) Cut up the videos into chunks, and in each chunk, no more than one mouse will having seizures. Finally set as 5 minutes per chunk.
>
> (ii) Keep the ones with seizures, and leave the rest 'no-seizure' chunks. (Reduce the data set size and make the dataset more balance). Also divide up the annotations to match the video chunks.
>
> (iii) Track the mice individually, and record the Bounding Box (BBox) of each mouse in each frame; then extract fixed resolution videos based on

the tracking BBoxes. Among the video chunks, pick the ones with seizures as the training data. As there will be only one mouse with seizures within the video chunks, the number of 'sub-video' chunks will the same as the divided annotations.

5.2 Annotation Preprocessing

The indexes of annotations are the timing in the videos, each in a grid of 40 ms per annotation, range from 0 ms to the last millisecond of the video. Because it is a '25 frames per second(fps)' video, the interval time between frames is

$$1000(ms)/25(frames) = 40(ms/frame) \tag{5.1}$$

So the annotations are just matching the frames, when we take the annotations one by one, it will be exactly the annotations for the video frames one by one. And we don't need to keep the timing for the annotations. And then we have a list of 'Total-Frame-Numbers'$\times$16 for each video, that in each row the binary 'True(1)/False(0)' correspond to the type of seizures for the current frame.

Next I make slices, that pick the columns for seizures in the table, and make a sum of each row.

```
ifSeizure = np.sum(List)
pos = np.array(ifSeizure)
pos = (pos>0)*1
neg = (pos≤0)*-1
new_target = pos+neg
```

After process, we combine all the seizures together without distinguish the types, and the annotations are cleaned into 1-dimension list for the '1 as seizures' and '-1 as non-seizure'. However we could still use the original annotations for the classification in the future, and I am only doing this to simplify the project problem from a multi-class classification to a Binary-Classification problem.

As we divide the videos into 5-mins chunks, the 5-hours videos are divided into 60 video chunks. In the final cleaned video chunks, there are only 31 video chunks contain 'seizure' annotations.

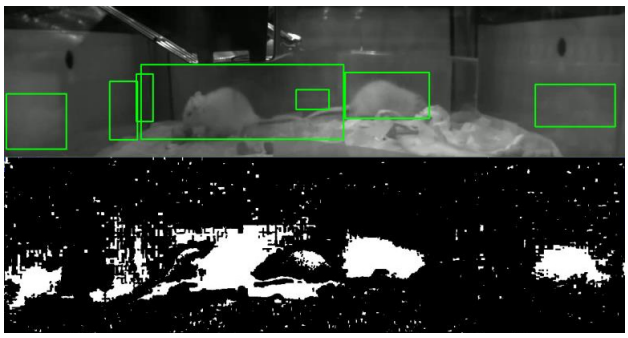

Figure 5.1: This figure shows an example of tracking results using **ViBe**(Barnich and Van Droogenbroeck, 2011), with the code implemented by Researcher R.Sillito.

5.3 Automatically Video Tracking Attempts

Based on the previous cleaning methods, we have only 31 video chunks left to operate.

In the origin plan of preprocessing, we need to implement the auto-tracking program to track the mice individually and generate their Bounding Box. Based on the existing methods and open-source packages, I have investigated the following approaches for auto-tracking: Background Subtraction, Color Threshing and Optical flow and Tracking.

5.3.1 Background Subtraction

First, I tried with an fixed background.

The **first frame** with Gaussian Smooth as background, and also another attempt with **Averaging frames** of all the video;
A simple substraction between frames and background;
Threshold for the pixel-size of connected fields.

However, there are reflections of mice, and the fillings of the Home Cage floor are changing. The results seem to be a mess and unstable.

And then, I tried the code for **ViBe** (Barnich and Van Droogenbroeck, 2011), implemented by the researcher R.Sillito from Actualanalytics ltd, UK. It is an algorithm using Adaptive and Probabilistic Background Model. However, the results are still too poor to track the mouse, due to the same problem above. The parameter settings still need tuning. Figure 5.1 shows an example of tracking results of ViBe.

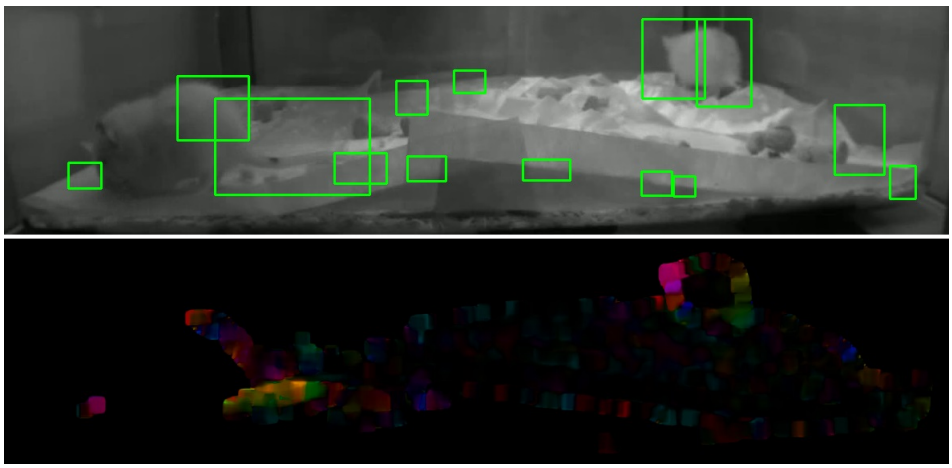

Figure 5.2: This figure shows an example of tracking results using **Optical Flow**, the function *calcOpticalFlowFarneback*, an open-source function in OpenCV-3.

5.3.2 Optical flow

I tried optical flow using OpenCV with python3. Sometimes we have good detection of the mice, but we still can't have a stable tracking result when the mice are not moving, or the fillings of Home Cage ground greatly changed, or when the reflection appears. Figure 5.2 is an example of the tracking results using optical flow.

5.3.3 Other Tracking Methods

I also tried other OpenCV packages for tracking: 'BOOSTING', 'MIL','KCF', 'TLD', 'MEDIANFLOW' and 'GOTURN'. Detailed implementation can be found from the tech-blog: `www.learnopencv.com/object-tracking-using-opencv-cpp-python`.

Figure 5.3 shows the tracking result of using 'MIL traker'. The other trackers' results are similar to the MIL tracker, that they all successfully track the object in the first few seconds (around 3 or 4 sec), and then they start to loose track or have wrong bounding box. Need to point out that, all these trackers require manual operation for object bounding box in the first frame.

As a result, all the trackers didn't meet the requirement. Thus, I implement the manual toolkit instead.

Figure 5.3: This figure shows an example of tracking results using MIL Tracker, the function implemented in the open-source package, OpenCV3.

5.4 Manually Sub-Video Generating Tool

This is a simple tool for generating sub-videos, easy and efficient, the interface is shown in figure 5.4. As the manual work won't be the final product for the ultimate goal of project, the toolkit developing process is based on the Principal of minimum development, but still keep the efficiency and flexibility. The logic of the tool can be divided into 3 parts:

(i) Bounding Box Drawing;
(ii) Mid-Frame Padding;
(iii) Sub-Video Generating.

5.4.1 Bounding Box Drawing

The basic function of this part, is allowing annotator (user) draw the bounding box of the object. Shown in figure 5.4, this is an interactive tool, that users use cursor to draw the boundingbox by dragging the cursor from the left-top to right-bottom within the frame.

However, if we draw the bounding box for every frames, it would be a huge amount of work. So I set an adjustable 'FrameStep' that the user can draw the bounding box every *FrameStep* frames. As this tool is actually used in my project, I set the *FrameStep* to '10'. Because I notice that in these video examples, the mice are not moving such fast that we need to draw the BBox frame by frame; but when we set the *FrameStep* too large, the generated mid-frames won't be accurate enough for the motion features extraction.

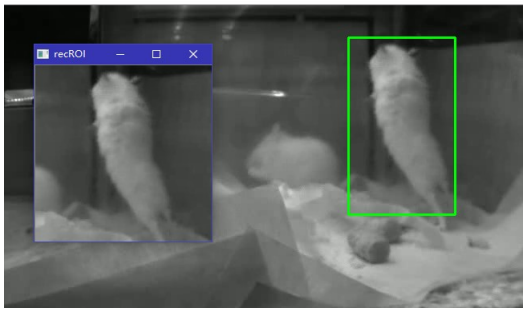

Figure 5.4: This figure shows the interface of the sub-video generating tool. The toolkit allows users to manually draw BoundingBox for the mice each FrameStep (Adjustable, not frame by frame); automatically generate the BoundingBox for the mid-frames; and create the cleaned video for future usage. As the BoundingBox is not fixed-size, the toolkit also resizes the sub-video into fixed-size & fixed-resolution video.

Also in the practical usage of the project, I have also added some more supported methods:

(i) Reset & Confirm. The bounding box is not easy to be accurately drawn every time, so I add the *Reset('R')* Button for re-drawing the BBox. Once you have done the bounding box drawing, press 'C' to confirm and continue, the image of bounding box result will be cut from the video frame, and display on the side.

(ii) Skip (Forward). The activities in the home-cage video include grooming, sleeping and other types that the rodent stays still in a long term of time and its bounding box doesn't change at all. To reduce the cost of repeat drawing work, I add the *Skip (forward, 'F')* button that the user doesn't have to draw the bounding box of the frame. The idea is similar to adding a key-frame in a animation, that we only have to create the key-frame information, and the program will generate the rest automatically.

(iii) Saving, Auto-Saving & Resume from Break-Point. The videos are always last longer than minutes, and the BBox Drawing work is always time-consuming. In order to save the work from unknown cause of system crashing or program flashing back, I implement the *Save('S')* bottom to initiatively save the bounding box results, and auto-save every 5 **draws**. The next time when you start the work, the program will search the saved result for the same video, and the user can resume from the last saved work. In my practical usage, these do save a lot time from repeating work.

(iv) Roll Back. Sometimes, the work that has been 'confirmed' doesn't seem to work well, or the mouse moves fast during the skipped frames that we might lose track, and need to roll back to previous frames. So I implement the *Roll Back('B')* bottom for user to correct the mistakes.

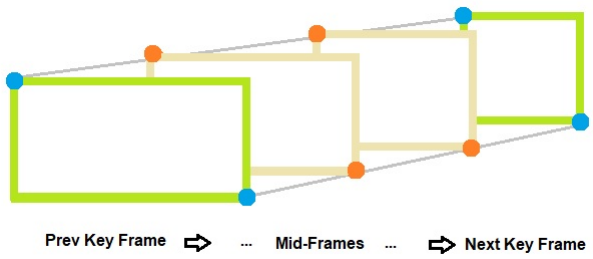

Figure 5.5: This figure shows the idea of Mid-Frame Padding, using *interpolation*. It is similar to the idea of generating the mid-frames when creating flash animation, based on the key points in key frames.

5.4.2 Mid-Frame Padding and Sub-Video Generating

Mid-Frame Padding assumes the movement of the object is uniform. In the previous part, we have already drawn the BBoxes of the object for the key frames, the result contains a 'left-top' point and a 'right-bottom' point. Using **Interpolation**, we can easily generate the BBoxes for the mid-frames, shown in figure 5.5. After process, we have all the bounding box results for each frame.

Sub-Video Generating generating the fixed resolution video, using the BBoxes results from the previous parts *Mid-Frame Padding*, that for each video frame, we have the bounding box for the interested object. First we extend the bounding box to the best fit of out-put video shape; Next, do the padding with 'zeros(black)' to the out of frame area; Finally, reshape the patch into the pre-set resolution.

We set the output video resolution slightly smaller than the minimum rodent shape. In this way, no matter which position the rodent is, as long as the rodent is having the same posture, the final output video will be the same, that we don't need to take care of the influence of size changing.

After **half-a-month** manual preprocessing, using the toolkit I implemented, a total $31 \times 5 = 151$-minutes videos have been marked with BBoxes. All the annotated seizures appear in the sub-videos. And as said before, the BBoxes of mice range from 80 to 300 pixels high, so I generated 2 sets of sub-videos with resolution of '32×32' and the other with '64×64'. The two resolutions are widely used in the machine learning area for computer vision problem, that we can easily apply the existed ML models for machine learning, and the dataset sizes are small enough to handle for multi-tests.

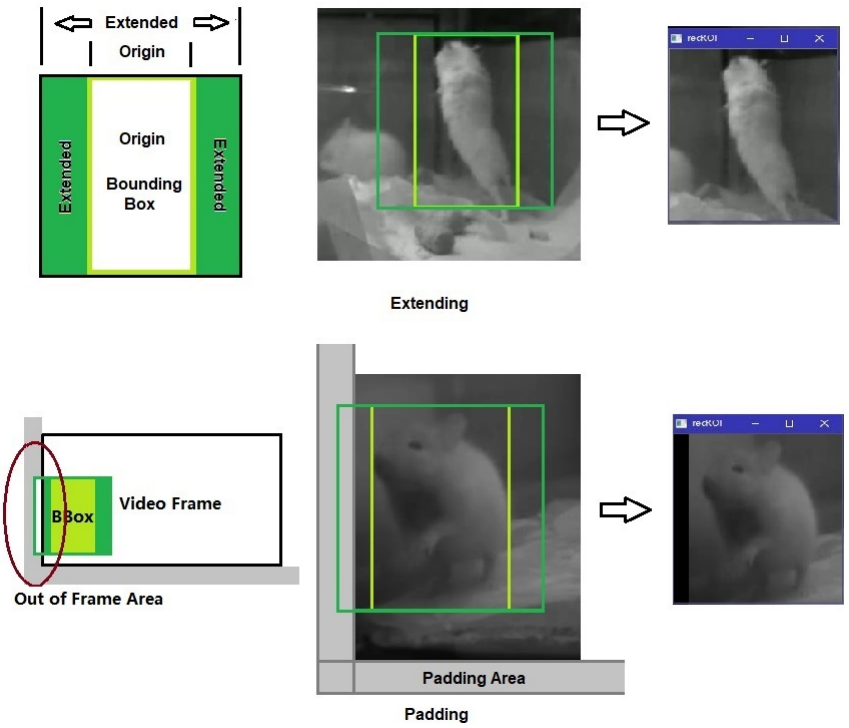

Figure 5.6: This figure shows the methods for generating fixed size sub-video from frames based on the bounding boxes. When given the bounding box, and resolution size, we adjust the bounding box to the best fit into the resolution window, which will extend the origin bounding box to the extended one. When the extended bounding box is out of the frame area, we use 'zeros(black)' for padding, so that the generated video is still the pre-set resolution size.

5.5 Summary

In this chapter, I have introduced the implementation of preprocessing. The general contribution for preprocessing, algorithm logic for annotation preprocessing, the attempts for the auto-tracking and the manual toolkit implementation. Next chapter, I am going to explain the implementations for detector system: Feature Extraction and Classification Models.

Chapter 6

Detector System Implementation

This chapter I am going to introduce the implementation of detector system, which includes the feature extraction, classifier models and evaluation.

6.1 Feature Extraction

The purpose of Feature Extraction is explained before, that we extract the useful features to train the classifier more efficiently.

In this project, we extract 3 kinds of motion features and self-defined tracking features. To be specific, in the project I have extracted the *Spatial-temporally binned Histogram of Orientation Gradients (HOG) and Histogram of Optical Flow (HOF)* (Uijlings et al., 2015), using the online code from the paper's research group, with Matlab 2015a; *Local Trinary Patterns* (LTP) (Yeffet and Wolf, 2009), using the code implemented by R.Sillito, researcher from Actualanalytics ltd, UK; and Tracking features with BBox info, speed and acceleration, extracted from the preprocessed bounding box data.

The changing parameter settings of HOG and HOF are the same, which are two $<1\times3>$ vectors defining the *blockSize*-(row,column,frame), *NumOfBlock*-(row,column,frame). The rest parameters are same as the default settings, such as the *frameSampleRate* to '1', that we use all the frames from the video; the Quantization of orientation to '8', that the Histogram contains 8 orientation information; and it use 'Horn-Schunck' for optical flow. As shown before, figure 3.6 shows the Blocks of Spatial-temporally binned video.

The settings in LTP are all the default ones, with $delta = 3$, $threshold = 1000$ and $grid = [4,4]$, which defines the *windowsize* of feature, the threshold for trinarizing

HOG & HOF FEATURES		
VIDEO RESOLUTION	BLOCKSIZE	NUMOFBLOCK
32X32	$4 \times 4 \times 5$	$3 \times 3 \times 2$
32X32	$4 \times 4 \times 10$	$3 \times 3 \times 2$
32X32	$8 \times 8 \times 5$	$3 \times 3 \times 2$
32X32	$8 \times 8 \times 10$	$3 \times 3 \times 2$
64X64	$8 \times 8 \times 5$	$3 \times 3 \times 2$
64X64	$8 \times 8 \times 10$	$3 \times 3 \times 2$

LTP FEATURES	MOTION FEATURES
VIDEO RESOLUTION	WINDOWSIZE
32X32	5
64X64	10
ORIGIN(500X1200)	25

Table 6.1: This table shows the kinds of features that we have extracted from the pre-processed data.

data, and the grid size (row,column). Th e combination of *grid* and *delta* is similar to the *blockSize* for HOG/HOF.

Tracking features, here I call it the *Motion*, which are extracted motion elements from the BBox data, that I have manually drawn. Features for each frame is a $< 1 \times 11 >$ vector: <Central point x, Central point y, Width, Height, Speed x, Speed y, Acceleration x, Acceleration y, Width/Height, Acceleration(Scalar), Speed(Scalar)>. And here, the unit for speed is 'pixels per frame'; the unit for acceleration is changing speed per frame, 'pixels per frame square'. The final features for training are adjusted by the *windowsize*, which sorts each continuous *windowsize* frames into one vector.

Table 6.1 shows the types of features, I have extracted for the project. As you can see, I didn't extract HOG/HOF Features from the original video. The reason is that it requires a lot RAM, which will cause the memory exceed error, while extracting 5-mins original resolution video chunk, using 8G-RAM DICE machine.

And the Features file-sizes range from less than 200MB to greater than 20GB. And the file-sizes influence the training and detecting speed.

2. Large Linear Classification (Binary and Multi-class)

LIBLINEAR supports two popular binary linear classifiers: LR and linear SVM. Given a set of instance-label pairs $(\boldsymbol{x}_i, y_i), i = 1, \ldots, l,\ \boldsymbol{x}_i \in R^n,\ y_i \in \{-1, +1\}$, both methods solve the following unconstrained optimization problem with different loss functions $\xi(\boldsymbol{w}; \boldsymbol{x}_i, y_i)$:

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{l} \xi(\boldsymbol{w}; \boldsymbol{x}_i, y_i), \tag{1}$$

where $C > 0$ is a penalty parameter. For SVM, the two common loss functions are $\max(1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i, 0)$ and $\max(1-y_i\boldsymbol{w}^T\boldsymbol{x}_i, 0)^2$. The former is referred to as L1-SVM, while the latter is L2-SVM. For LR, the loss function is $\log(1+e^{-y_i\boldsymbol{w}^T\boldsymbol{x}_i})$, which is derived from a probabilistic model. In some cases, the discriminant function of the classifier includes a bias term, b. LIBLINEAR handles this term by augmenting the vector $\boldsymbol{w}$ and each instance $\boldsymbol{x}_i$ with an additional dimension: $\boldsymbol{w}^T \leftarrow [\boldsymbol{w}^T, b], \boldsymbol{x}_i^T \leftarrow [\boldsymbol{x}_i^T, B]$, where B is a constant specified by the user. The approach for L1-SVM and L2-SVM is a coordinate descent method (Hsieh et al., 2008). For LR and also L2-SVM, LIBLINEAR implements a trust region Newton method (Lin et al., 2008). The Appendix of our SVM guide[2] discusses when to use which method. In the testing phase, we predict a data point $\boldsymbol{x}$ as positive if $\boldsymbol{w}^T\boldsymbol{x} > 0$, and negative otherwise. For multi-class problems, we implement the one-vs-the-rest strategy and a method by Crammer and Singer. Details are in Keerthi et al. (2008).

Figure 6.1: This is the explanation of LIBLINEAR from its paper (Fan et al., 2008), that the package implements several state-of-art linear classifiers.

6.2 Detection Models

The detection(classification) models here we choose Support vector machine(SVM), which is widely used for behavior recognition problems. And it is also one of the simplest models for classification tasks in machine learning, without too complicated model structure, that it would be more likely to show the the effectiveness of the detection system .

As for the implementation, I choose the LIBLINEAR(Fan et al., 2008) package, the open-source SVM classification models for large amount of data. The package LIBLINEAR implements different SVM models, as a set of easy-to-use APIs for binary and multi-class classification problem. Figure 6.1 is the explanation of the implementation of LIBLINEAR, from its published paper (Fan et al., 2008), that different combinations of loss functions and problem solving approaches are included in the package.

To match the training and predicting(detection) APIs in LIBLINEAR, I implement a toolkit for loading the extracted features (data files) as the package-required format. The main 'class' is called 'dataManager', along with other important functions are packed in 'toolkit.py'; and the other operating scripts are written with *jupyter notebook*.

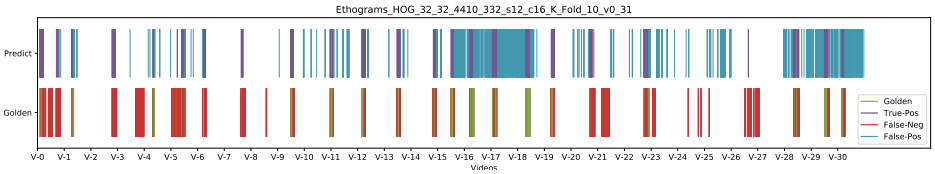

Figure 6.2: This is an example of Ethogram with all the prediction results are from the test-set results, when doing the cross-validation.

6.3 Evaluation and Prediction Enhancement

As the dataset is unbalanced, we can't simply use the accuracy to evaluate the detection model, so we choose Sensitivity and Specificity, seen in figure 3.7. I first calculate the True Positives(TP), False Negatives(FN), False Positives(FP) and True Negatives(TN), and then calculate the true positive rate (TPR) as sensitivity and True Negative rate (TNR or SPC) for specificity:

$$TPR = TP/P = TP/(TP+FN) \tag{6.1}$$

$$SPC = TN/N = TN/(FP+TN) \tag{6.2}$$

To give an intuition to the Detection results, we use Ethogram. And also we want to make best use of the data, so I use the K-Fold cross-validation. When I plot the ethogram, I first run the cross-validation first, so we have results for each fold as test set; next I splice all the test-set results together as one detection result; finally, I plot the ethogram with the 'all test results'. Figure 6.2 is an example for the Ethogram.

However the detection from the models doesn't look so well. Thus we propose a way to *Enhance* the prediction, which includes two approaches:

(i) Threshold adjusting; (ii) Prediction extending.

Threshold adjusting. We are using regression SVM to generate the prediction, and we use a threshold to decide whether the sample is a seizure. Same as we've done for the ROC-curve, when we change the threshold, the predictions for 'seizures' will increase or decrease. **Prediction extending** is the method to extend predictions from detector, based on the pre-set windowsize, seen in figure 6.3.

6.4 Summary

In this chapter, I have introduced the implementation for feature extraction, classifier(detector) model, and evaluation methods. Based on the existing code supported by

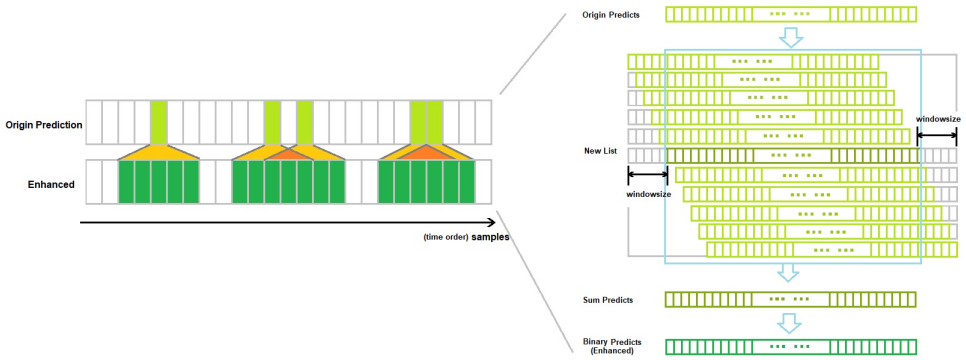

Figure 6.3: This figure shows the idea of enhancing the prediction. We want to mark all the samples as 'seizure', which have a 'seizure neighbor' within *windowsize*. To achieve so, I extend the origin predicts into a new list, sum up the list, and then transform the results into binary predicts.

the company and found from online open-resource, I implement a toolkit package and a few scripts for model training, predicting and evaluating. Next chapter, I am going to discuss the experiment results.

Chapter 7

Experiments Results

In this chapter, I am going to introduce the experiments that I have done to test different kinds of combination of detecting approaches. The experiments will be divided as (i) *Raw Video*, (ii) *Tracking Features*, (iii) *Preprocessed Tracking Video*, (iv) *Prediction Enhancement* and (v) *Overtraining*.

In the experiments, I use the three kinds of SVM regression models, implemented in the *LIBLINEAR* package (Fan et al., 2008):

(i) s11 – L2-regularized L2-loss support vector regression (primal)
(ii) s12 – L2-regularized L2-loss support vector regression (dual)
(iii) s13 – L2-regularized L1-loss support vector regression (dual)

In the following sections, I will simply call the three models as 's11', 's12' and 's13'. And if without specification, the video data I use, are the 31 tracking chunks with seizures, the seizure types are shown in figure 7.1. The details of LIBLINEAR can be found in `https://www.csie.ntu.edu.tw/~cjlin/liblinear/`. And the reasons for choosing LIBLINEAR is explained in section 6.2.

Also, when I use the expression *Motion* other than 'motion feature', I am talking about the tracking features that extracted from the 'Bounding Box' results. You can find the definitions in last chapter (Section 6.1).

7.1 Raw Video - Baseline

This section is about the experiment I have done on the Raw Videos. I choose this as my baseline model, that it is the most direct method for seizure detection, without too much work. The experiment settings are shown in table 7.1. As you can see, the

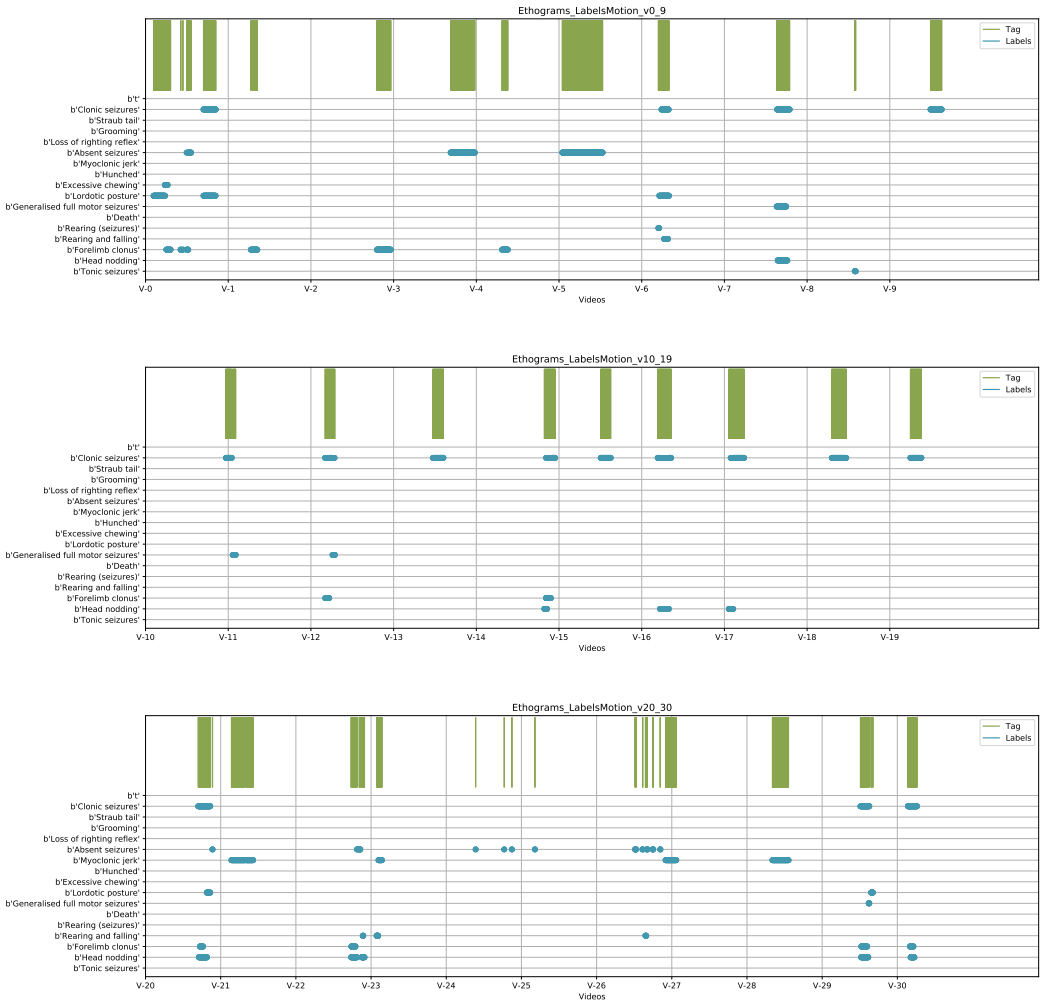

Figure 7.1: This figure shows the types of seizures for the 31 seizure video chunks. The video chunks are sorted by chronological order. The X-axis is the frames in the videos, against the Y-axis as the labels of seizure types. The green bars at the top are the frames marked as seizures.

ID	MATERIAL	FEATURE TYPES	MODELS
LTP-origin	*origin* (500x1200)	*LTP-default*	*SVM*

Table 7.1: This table shows the experiment settings, I have done for Raw Videos.

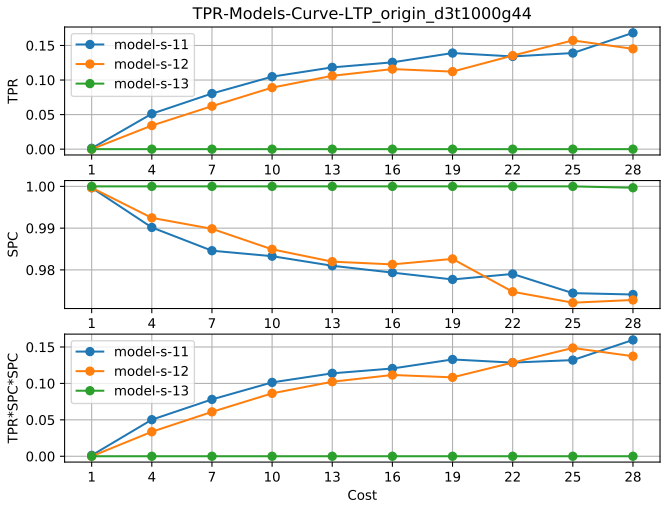

Figure 7.2: This figure shows different results from the models, training with the raw videos. The y-axis is the True Positive Rate (TPR), the True Negative Rate (Specificity, SPC), and an *indicator* $= TPR \times SPC \times SPC$, to evaluate the model.

experiment only extracts LTP features, because the package for 'HOG/HOF' requires larger memory space, which cause the 'Memory Exceed' warning.

For the details, I use the 31 chunked-up original-videos (all with seizures) as the material to extract LTP features. And then, I train the SVM models with these features.

First I separate the LTP features into 10-folds without shuffling the data, that all the data is in the time order, and then I choose the first fold as the test-set. And for the 3 regression SVM models, I vary the 'cost' from 1 to 28 with a grid step of 3, to find the best parameter settings for the features.

The reason why I choose the first fold is that, as you can see in figure 7.1, the first 3 videos have almost most of the seizure types, it would be better to test on them than the other folds. Figure 7.2 is the experiment results for different model settings. I use 'TPR$\times$SPC$\times$SPC' as the indicator to find the best model, which proves to work pretty well. According to the result, I choose the 's-11 c-28 SVM' model for the further experiments, whose $TPR = 16.84\%$, $SPC = 97.41\%$, and the indicator $TPR \times SPC \times$

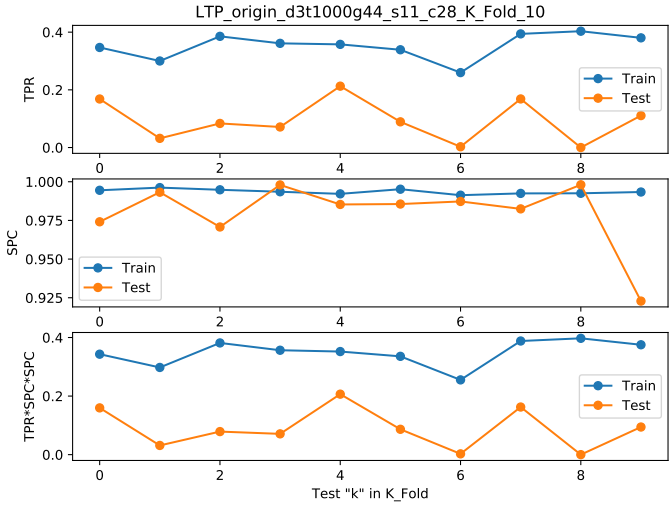

Figure 7.3: This figure shows different results from the models, training with the raw videos. The y-axis is the True Positive Rate (TPR), the True Negative Rate (Specificity, SPC), and an *indicator* $= TPR \times SPC \times SPC$, to evaluate the model.

$SPC = 15.97\%$.

Next, I do the cross-validation on the model 's-11 c-28 SVM', and plot the Ethogram, Cross-validation Results and the ROC curve, seen in figure 7.5, 7.3 and 7.4. In the ethogram, we can see that, the purples show the model can detect the seizures, but still not accurate and missed a lot seizures.

In all, the detector using the raw video to extract LTP features, training with SVM models, does work in some way for seizure detection, but still misses a lot seizure samples.

7.2 Preprocessed Tracking Video

Similar to the experiments for Raw Video, I extract different features from the different resolution preprocessed tracking-video with different SVM models. Table 7.2 shows the experiments I have done. And I do the same experiments for each 'Material-Feature-Model' combinations with different settings: (Details can be found in the appendix A:

(i) Multi-model Test, to find the best SVM model setting;

(ii) Cross-validation for the best SVM model found in the previous step;

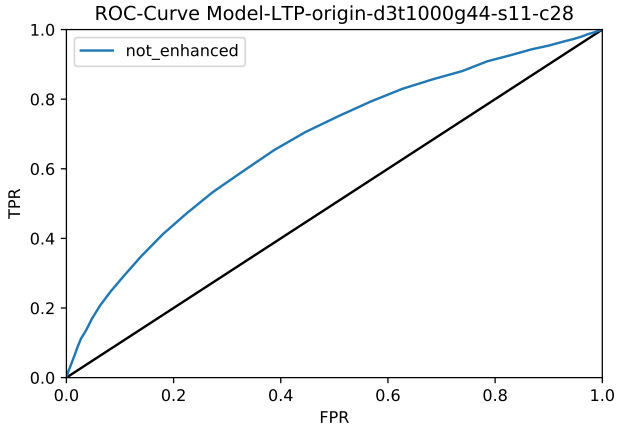

Figure 7.4: This figure shows ROC curve for LTP features extracted from origin video, using 's-11 c-28 SVM' regression model.

(iii) Plot the Ethogram & ROC-Curves.

.

Figure 7.6 shows the best experiment results for different model combinations. And in figure 7.7, I save the average and variation of the crossvalidation results for the 5 models: 'LTP-origin', 'LTP-M2', 'HOG-M2', 'HOG-M6' and 'HOF-M4' ('LTP-origin' is the best model for Raw Video, as the baseline). And we can see that, after preprocessing, the results are much better than the baseline. The ethograms for the 4 models, are shown in figure 7.8; in this figure, all the models both works for some video chunks, and missed some seizures at the same time. I have also plot out the other models combines the 'res-64×64','LTP/HOG/HOF' and their best-SVMs, and the ethograms are rather similar to each other.

7.3 Tracking Features

In this section, I am going to introduce the experiments on tracking features. Tracking features are the vector of combinations about the mouse position, shape of the BBox, speed and acceleration which are extracted from the tracked BBox data as the result of preprocessing (tracking). The BBox details are explained in chapter 5, and the tracking features are explained in chapter 6.

Basically we only track acceleration and velocity as the tracking features. In the paper of acceleration-based seizure detection (Nijsen et al., 2005), the accelerations

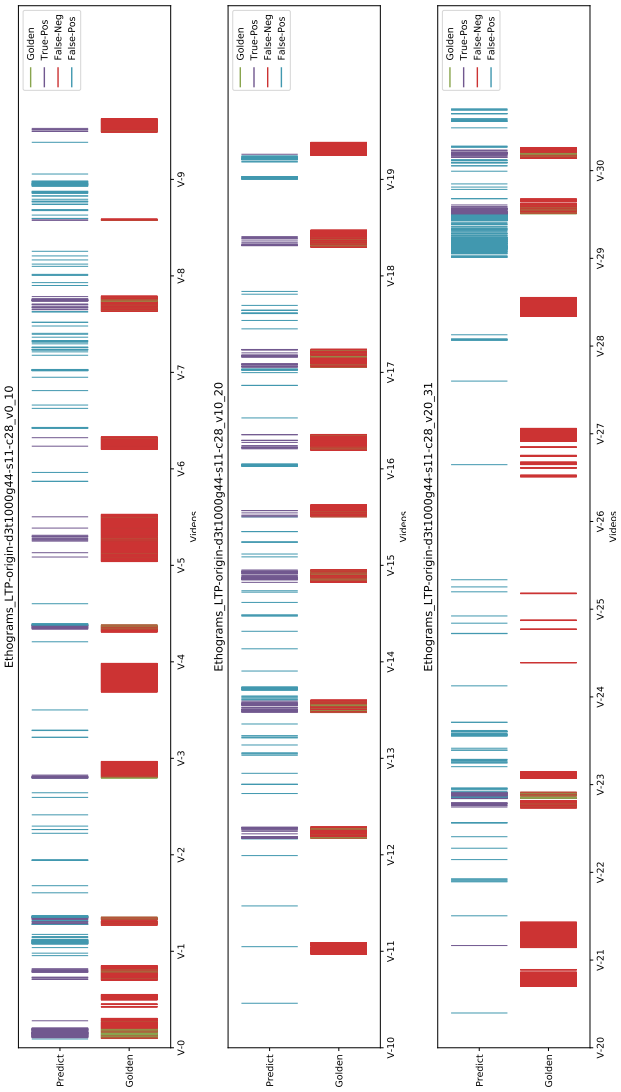

Figure 7.5: These are the Ethograms for the experiments with LTP features extracted from the original video, with 's-11 c-28 SVM' regression model. In the figure, it shows the prediction results for each video chunks, against the 'golden standard' (manual annotations as ground truth). The green ones and the purple are the True Positive (*TP*), the blue are the False Positives (*FP*), and the red ones are False Negatives (*FN*). **NOTE:** In this resolution plotting, the TPs are not alwayse matched between the green and the purple ones. The explaination is that, the plotting results are over compressed, that within a pixel length, it represents more than one sample, so the display color is the majority color. As the result, when we see the purple against red, it means prediction is *TP*, but still too sparse then the golden standards.

ID	MATERIAL(RES)	FEATURE TYPES	SETTINGS	MODELS
LTP-M1	32×32	*LTP*	DEFAULT	*SVM*
LTP-M2	64×64	*LTP*	DEFAULT	*SVM*
HOG-M1	32×32	*HOG*	(4,4,5),(3,3,2)	*SVM*
HOG-M2	32×32	*HOG*	(4,4,10),(3,3,2)	*SVM*
HOG-M3	32×32	*HOG*	(8,8,5),(3,3,2)	*SVM*
HOG-M4	32×32	*HOG*	(8,8,10),(3,3,2)	*SVM*
HOG-M5	64×64	*HOG*	(8,8,5),(3,3,2)	*SVM*
HOG-M6	64×64	*HOG*	(8,8,10),(3,3,2)	*SVM*
HOF-M1	32×32	*HOF*	(8,8,5),(3,3,2)	*SVM*
HOF-M2	32×32	*HOF*	(8,8,10),(3,3,2)	*SVM*
HOF-M3	64×64	*HOF*	(8,8,5),(3,3,2)	*SVM*
HOF-M4	64×64	*HOF*	(8,8,10),(3,3,2)	*SVM*

Table 7.2: This table shows the experiment settings, I have done for Raw Videos.

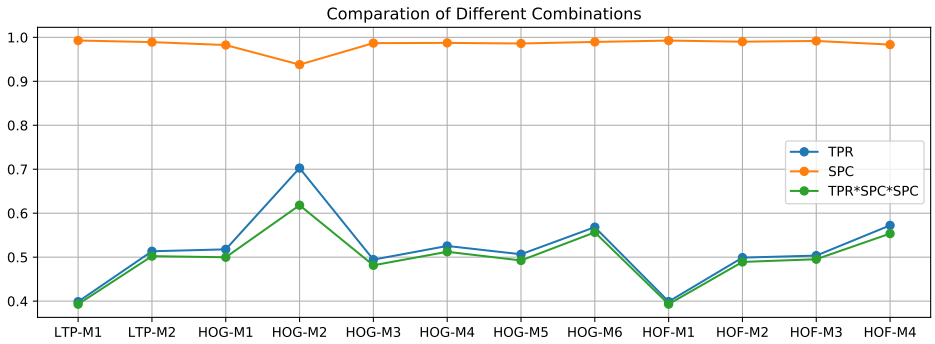

Figure 7.6: This figure shows the best indicator results for the experimented combination, along with their *TPR*s and *SPC*s. The best indicator results are using the same plot as 7.2, that I vary the kinds of SVMs and the value of *Cost*. The model settings are given in table 7.2. And we can see the '64×64' resolution videos are better than the '32×32' ones; also the settings of '(8,8,10)' is better than '(8,8,5)'.

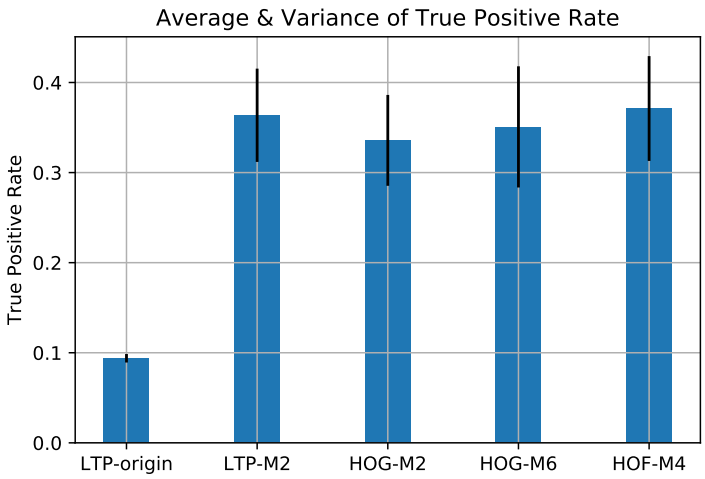

Figure 7.7: This figure shows the Average and Variance of the 10-folds crossvalidation results, for the 5 models: 'LTP-origin', 'LTP-M2', 'HOG-M2', 'HOG-M6' and 'HOF-M4'. The settings can be found in 7.1 and 7.2.

for seizure detection are collected from different parts of the subject, that the subject's behavior can be divide into the accelerations of different body parts. However, in my current tracking result, the subject (mouse) is considered as a whole, that the acceleration is only for the whole bounding box. When we look into the videos of seizures, such as the 'Lordotic posture', the bounding box for the whole mouse is not moving too much, but the forelimbs are shaking rapidly, that we are loosing too much acceleration information. Thus, the results from the tracking features are much poorer than the other tasks.

In order to make up the shortage of poor tracking results, we need to generate more information from the tracked results. So I added the ratio of width and height, the *W/H*, as part of the tracking feature, to indicate the behaviors of the mouse:

(i)Resting. The mouse is more likely to huddling as a 'ball', that the W/H would be around 1.0.

(ii) Rearing. The mouse is standing on its feet, that the height is much larger than the width, $W/H < 1$;

(iii) Walking or Running. When the mouse is running, its body is stretching, the height is much smaller than its width, so $W/H > 1$.

The use of ratio will ignore the size of bounding box, no matter the mouse is near or far from the camera.

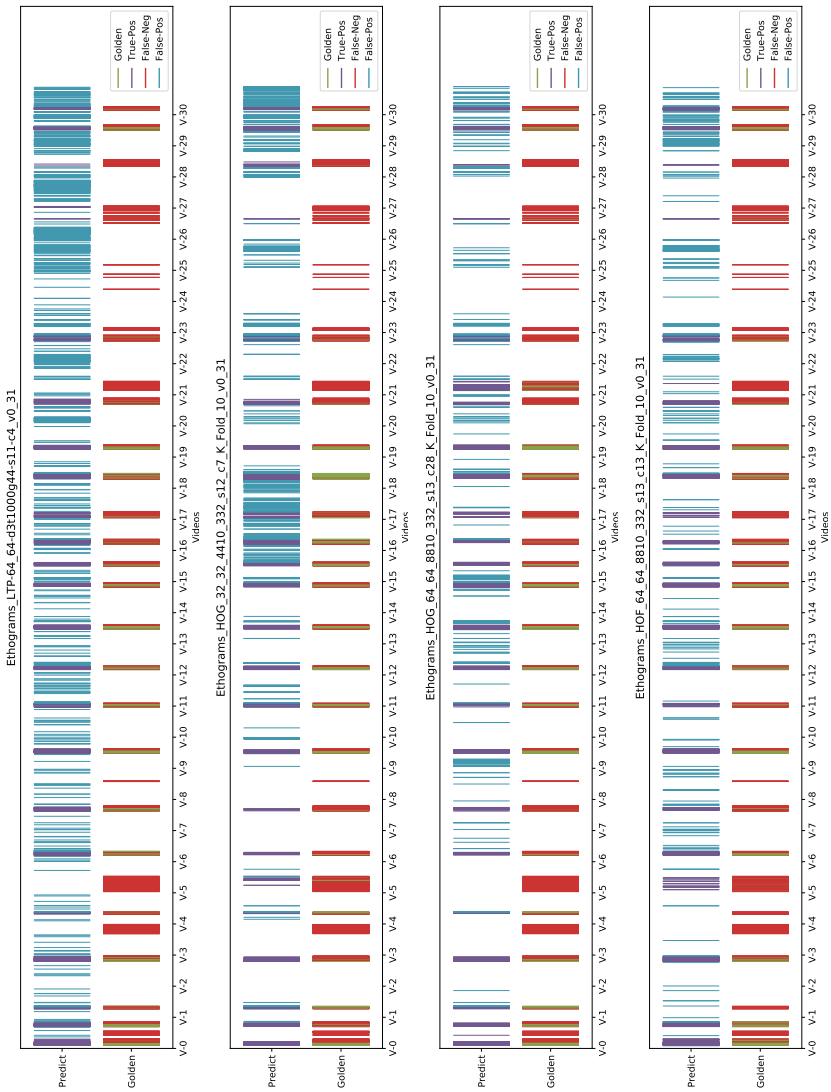

Figure 7.8: The 4 Ethograms here are the test results for 'LTP-M2', 'HOG-M2', 'HOG-M6' and 'HOF-M4', from left to right, using 10-folds crossvalidation. The settings can be found from table 7.2.

ID	MATERIAL	FEATURE TYPES	WINDOWSIZE	MODELS
Motion-win5	*Tracking BBoxes*	*Motion*	5	*SVM*
Motion-win10	*Tracking BBoxes*	*Motion*	10	*SVM*
Motion-win25	*Tracking BBoxes*	*Motion*	25	*SVM*

Table 7.3: This table shows the experiment settings I have done for tracking features.

To give a intuitive feeling of the features, I plot the normalized acceleration, velocity and the *W/H* along with the ethograms of golden standards, seen in figure 7.9. We can see some potential patterns within the figure, but still need to question. It is noteworthy that, the tracking results here is manual key-frame annotation with the auto-padding for the mid-frames, that the tracking result is not accurate and will lose details for the rapid changes. As a result, the tracking feature curves might contain more subtle changes, when we use a more stable and reliable auto-tracker, which will be more promising.

Table 7.3 shows the 3 sets of experiments I have done for the tracking features. Same order as the previous experiments, I have tested the SVM model settings, crossvalidation and the plots.

Figure 7.10 is the crossvalidation results for the best SVM settings of the three kinds of tracking features. We can see that the change of windowsize do influence the detecting performances. Among them, $windowsize = 5$ performs the best, and its Average-TPR is slightly better than HOF/HOG results, but also with high variance. In figure 7.11, there are crossvalidation results for each folds, ROC-curve and the ethogram of all the test results. We can see the prediction results are not stable for each folds, that the prediction isn't specific for some of the video chunks. Compared to the HOG/HOF results, the tracking features are not so reliable.

However, there is still space for the improvement of tracking features:

(i) the values for tracking features are not centralized, that I didn't find the way to make the values of elements comparable;

(ii) the weights in the SVM models are adjustable, that we can try different settings to optimize the performance;

(iii) more tracking features can be extracted, when we have approaches to track the specific interesting parts of mice, such as joints and forelimbs, whose movements are extremely special during the Motor seizure events.

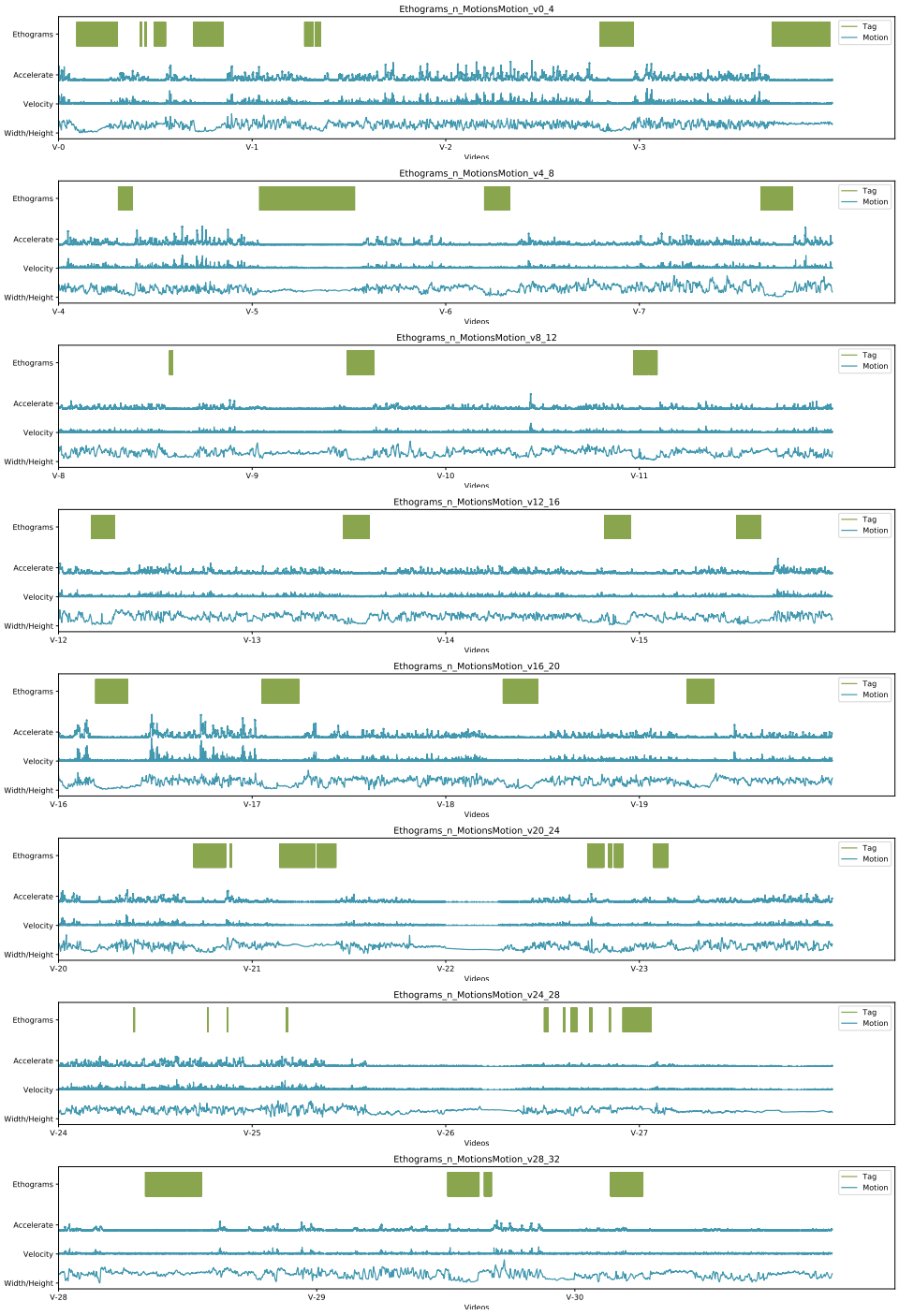

Figure 7.9: This figure shows parts of the normalized elements of Tracking Features against the Ethogram of Golden Standards.

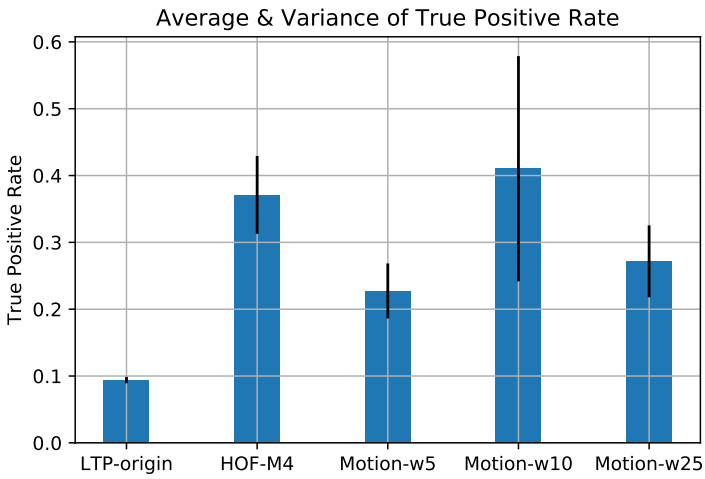

Figure 7.10: This figure shows the Average and Variance of the 10-folds crossvalidation results, for the 5 models: 'LTP-origin', 'HOF-M4'. The settings can be found in 7.1 ,7.2 and 7.3.

7.4 Prediction Enhancement

As we can see above, in the result plots and ethograms, we can already detect some of the seizure events, but the average True positive rate is still lower than 50%. In the ethograms, we can see that, the annotations (by human annotator) for seizures are continuous, but the predictions from auto-detector are sparse. Thus, we introduce a prediction enhancement method, as explained in previous chapter (section 6.3).

Here is an example using detector 'HOF-M4', that I tried different windowsize and threshold, as shown in table 7.4. The ethogram and histogram of the experiment results

ID	DETECTOR	WINDOWSIZE	THRESHOLD
Win5-T0	*HOF-M4*	5	0
Win10-T0	*HOF-M4*	10	0
Win5-T1	*HOF-M4*	5	-0.2
Win10-T1	*HOF-M4*	10	-0.2

Table 7.4: This table shows the experiment settings, I have done for Prediction Enhancement. It test the current best combination of detector system, 'HOF-M4' with different windowsizes and thresholds

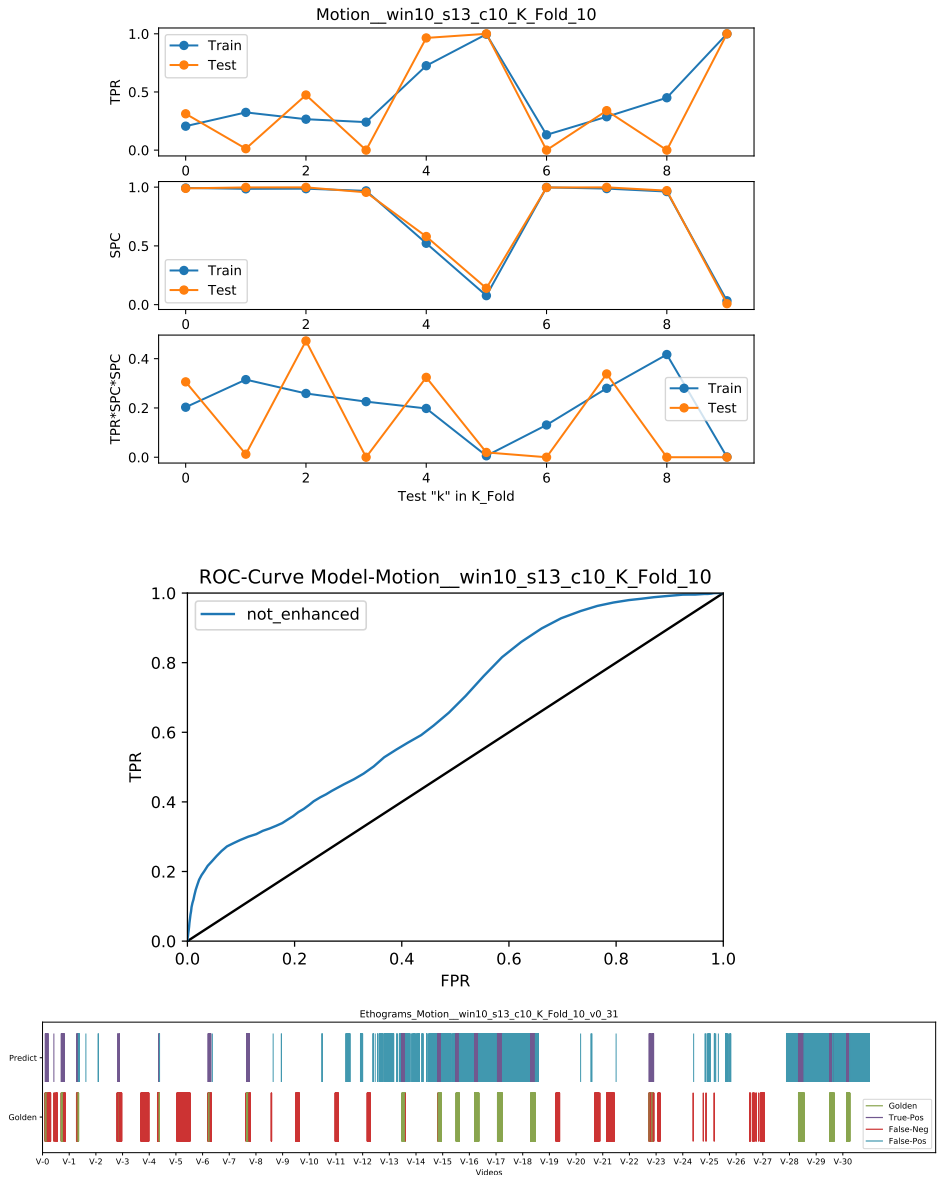

Figure 7.11: The tree figure are the crossvalidation results, ROC-Curve and ethogram for model 'Motion-w10', using tracking features with $windowsize = 10$.

Figure 7.12: These are 5 ethograms for detectors 'HOF-M4', 'Win5-T0', 'Win10-T0', 'Win5-T1' and 'Win10-T1', from the top to bottom. And the Histogram of *TPR*, *SPC* and *ACC* from crossvalidation results.

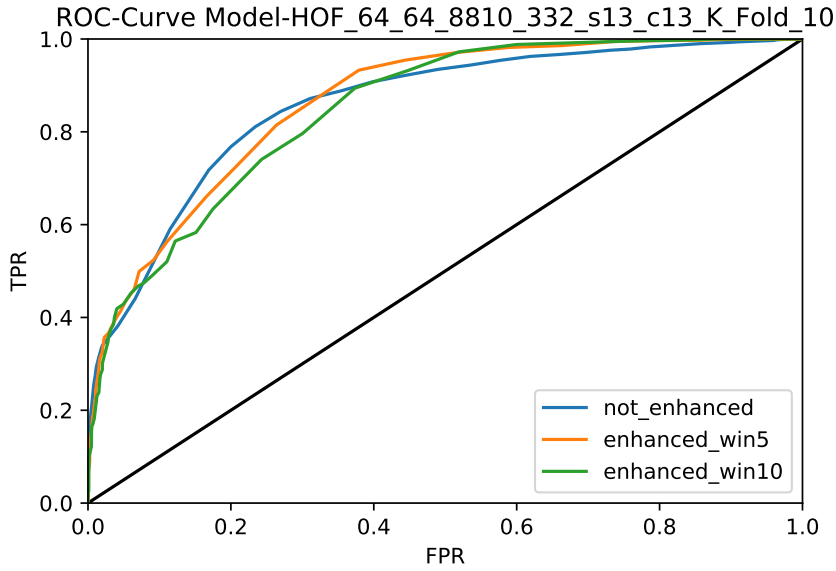

Figure 7.13: This is the ROC-Curve for 'HOF-M4' and the *enhanced* results with 'windowsize=5' and 'windowsize=10'.

are in figure 7.12. We can see that, the prediction enhancement do increase the True Positive Rate (TPR), but also reduce the Specificity and Accuracy. In the experiment, the TPR for 'Win10-T1' has already raise to 93%, compared to the origin result 34% for 'HOF-M4'; and the Accuracy (ACC) and Specificity (SPC) are greater than 50% (60% and 54%). Figure 7.13 is the ROC-Curve for 'HOF-M4' and the enhanced results with 'windowsize=5' and 'windowsize=10'.

7.5 Overfitting

In this section, I am going to talk about the overfitting and the generally analyze the experiments above.

As we don't have much training data but use a large feature space for SVM, that the final Magnitude for samples is only 10^5 and the feature space can be up to $< 1 \times 5000 >$, we need to consider the underfitting and overfitting problem.

First I re-train the models with the shuffled data, that all the videos would be partly seen in the training set, and the TPR results increase from a average of 20% to 50%, which means the models are not overfitting, using unshuffled 10-folds validation (the origin way in the experiments above). Next I tried different percentage of data as training set, and the results are much the same, one of the result for 'HOF-M4' is in

TP	TN	FP	FN	P =TP+FN	N =TN+FP	total
3252	10618	9045	212	3464	19663	23127

Table 7.5: This is the crossvalidation prediction result for model 'Win10-T1'. TP - true positive. TN - true negtive. FP - false positive. FN - false negative. The numbers are the number of samples for different class, explained in figure 3.7. P - positive samples. N - negative samples.

figure 7.14.

7.6 Final Performance Estimation

According to the above experiments, our current best model is the model 'Win10-T1', which is the detector model 'HOF-M4', enhanced with '$windowsize = 10$' and '$threshold = -0.2$'. And the detection result is shown in table 7.5. Based on the ROC curve in figure 7.13, it is impossible to detect all the seizures samples without looking through all the videos. But when we lower the standard, the detector system might still have some use.

In the appendix B, the ethograms are the zoomed-in results of model 'Win10-T1', for the 31 video chunks. The seizures are continuous, when human specialists annotate the videos; and we can assume that once we detect parts of the seizure events, we count the seizure event as 'detected', that they can find the whole seizure event manually. Under the condition, in this model, according to the ethogram in figure 7.12 compared with the plots of seizure types 7.1, it only missed the 'Absent seizures', which might be acceptable for some studies.

Considering our dataset is cleaned from a 5-hours video to $31 \times$ 5-mins video chunks, if we apply the detector system to the original dataset, we will have $23127 \times 60/31 - 23127 = 21635$ more samples, and these samples are all negative samples. Assume the non-seizure video samples have the same detection performance as the experimented ones, and they will have the same False positive rate (=46%). Thus, the detector will detect $21635 \times 46\% = 9952$ more samples as seizures for the whole 5-hours video.

As the result, the final detection result for the 5-hours video will mark

3252 (TP) + 9045 (FP) + 9952 (new FP) = 22249

samples as seizures, in a total number of $23127 + 21635 = 44762$ samples. In another

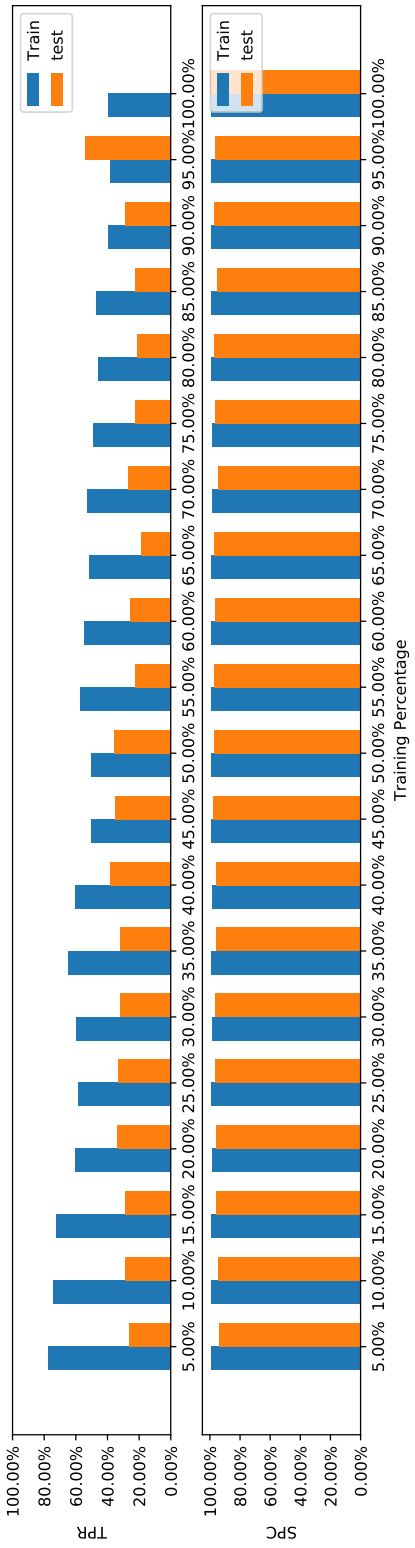

Figure 7.14: The figure shows the crossvalidation results, using different percentages of data as training data. The plot is the result using the detector system combination of 'HOF-M4'.

word, it means that the detector can reduce the video length to $23127/44762 = 49.7\%$, for specialist to check manually. Assume we have a 24-hours dataset, we can compress the length of video to only 12 hours.

However, it is the **ideal** result, only if we can implemented the auto-tracking for preprocessing and ignore the 'Absent Seizure'.

7.7 Discussion and Summary

In this chapter, I have introduced the experiments with the proposed seizure detection system. The main experiments are as below:

(i) A baseline using LTP features extracted from raw(original) video. The results is better than the 'random guessing', that some patterns of seizures might have been learned from this detector system.

(ii) SVM Models with motion features extracted from the preprocessed *sub-video*. A wild range of combinations of detector models with preprocessed sub-video are tested, using different kinds of feature settings and SVM model settings. Compared to the baseline, these models performs much better than the 'non-processed' baseline. The tracking method (preprocessing) for the mice in videos will greatly improve the detector system. And the implementation of auto-tracking will be the primary task in the future.

(iii) SVM Models using tracking Features, with different windowsize. This part of experiments show the intuition of tracking features and test the effectiveness of tracking features. Based on the preprocessed tracking result, the tracking features are proved to work, in some way; but it still need to improve before applying in actual detection task.

(iv) Prediction Enhancement. It proved to be useful to improve the true positive rate of prediction.

Overall, the results shows that the best combination of detection system is using SVM models trained with HOG/HOF features extracted from '64×64' tracking video; and the prediction enhancement can give a better performance with higher TPR in the unseen videos. Also when we train with more data, the more results seem to be more steady, that the detection system should still be underfitting.

Chapter 8

Discussion

In this thesis, I have designed an automated detection system for seizures in rodents, which includes 4 parts: preprocessing, feature extraction, classifier training and detection&evaluation. As for the actual implementation, I have developed a semi-automatic rodent-seizure detector, in which the preprocessing is adjusted to a manual processing toolkit. Besides I have done the experiments with different combination of feature types and classifiers. In this chapter, I will summarize the main points of the thesis and talk about the future work.

8.1 Contribution and Critical Analysis

This thesis is generally about the solution **design** and **implementation** for the problem,*Automated Detection of Seizures in Rodents*, which is raised from the actual research needs, based on the HCA system (Bains et al., 2016), sponsored by Actual Analytics Ltd. It is an open-ended problem, with only project requirements and the accessible data. The contributions of the thesis can be summarized as below:

(I) **Project background analysis & the Design of Detection System**. Based on the project background, I have done some research on the relative work, and finally designed the Automated Detection System, which can apply to all the seizure detection problem with 'Side-view-Video'. The system includes the 'Preprocessing', 'Feature extraction', 'Model Training' and 'Detection & Evaluation'.

(II) **Investigation of Auto-Tracking**. Auto-Tracking is a the preprocessing part in the original detection system design, and I have tested the performances of different existing 'video tracking' and 'object detection' methods on the project.

(III) **Design and Implementation** of the *Manual Video Preprocessing Toolkit*. It is a replacement for the Auto-Tracking, in order to complete the whole system function, that I have developed the toolkit to process the raw Home Cage video and obtain the *Sub-videos* for the Feature Extraction.

(IV) **Clean up and Preprocess** the videos. Use the toolkit I have developed, I manually annotated the videos for the tracking Bounding Boxes (BBox), and generate the trainable dataset in a self-proposed format.

(V) **Implementation of 'Feature extraction'**. It is based on the existing open-source code and company-supported toolkit, that I edit the existing codes to extract features into my self-proposed format dataset.

(VI) **Implementation of 'Model Training'**. It is based on the LIBLINEAR (Fan et al., 2008), an open-source toolkit, that I developed a *DataManager* as an API to match the interface of LIBLINEAR.

(VII) **Experiments on different combinations** of the extracted Features and Models, and **Evaluation** of their **performances**. I use the implemented codes of the self-proposed solution, tried different combinations of the features and SVM models; and to evaluate the results, I have implemented the scripts to analysis the model results based on the *Sensitivity and Specificity*, an statistical measurement.

Overall, I have roughly implemented the solution of detector system for rodent seizures. And the solution for the project, generally meets the requirement, that the final detection results is acceptable, that the ethogram shows the detector does work to detect the seizures. The detector is definitely better than the strong baseline - 'Random Guessing'; and the preprocessing method does increase the detection accuracy (True Positive Rate, TPR); also the *Prediction Enhancement* works in some way, that the TPR can be raised up to 90% with ACC and SPC both greater than 50%. Ideally, the detector system can reduce half of the video length for manual check (by specialists).

However, there are still some limitations in the implementation. The first is that the implemented system is not completely automated, with special requirement for manual annotation in the preprocessing part. The auto-tracking results are still too unstable for the detector system, and the ultimate version of Auto-Detector will require the implementation of auto-tracking. The second one is that, the detection results miss some 'Absent Seizures' and some obvious seizure events. The third one is that, at the current state, we didn't separate the types of seizures, which is easy to implement but might mess up the characters of different types of seizures, for example the 'motor' and 'non-motor' seizures.

Based on the experiment results, the best combination of detection system is as below:

Material	Feature	Classifier	Enhancement
resolution - 64×64	*HOF - (8,8,10),(3,3,2)*	*SVM - s13-c13*	*windowsize = 10*

Even so, I can't make the assertion but only raise the experiment conclusion:

(i) **HOG and HOF** perform similarly. The best parameter settings for the classifier (SVM) changes a lot when using different settings of features. Thus, when we change the input features, we need to search for the best SVM model again.

(ii) As for **LTP**, it predicts the same seizure events as HOG&HOF, but with slightly worse TPR; It still has an advantage that it is fast.

(iii) **Tracking features** work in some way, but still poorer than HOG&HOF. However, the logic for the tracking features is different than the motion features, and detection results do seem to be different. It is still promising once we modify the tracking features properly, or combine it with RFID data, because the acceleration is used in some other seizure detection research.

(iv) **Prediction Enhancement** is efficient to increase true positive rate for seizure detection.

8.2 Future Work

As is explained above, there is still a great space to improve, and it can be summarized as below:

- **Auto-Tracking**. The purpose of the project is achieve the automated detection system, to solve the labor-intensive problem in seizure detection, and we need the preprocessing part work automatically. In the experiment results, the preprocessed (tracked) sub-videos do improve the detection seizures a lot, compared to the raw-video. Thus, the most crucial work for the future is Auto-Tracking. Once we can achieve the auto-tracking process with stable results, the detection system can be applied into the actual usage right away.
- **Stabilization**. In most of the video-based behavior recognition tasks, stabilization and is one of the most important step to improve the classifier performance, which is the approache to stabilize the tracked video, as a denoising method for motion features. Ways of stabilization have been built up, and can be applied into this project.

- **Data Argumentation**. It is an common strategy widely used in vision-related machine learning area, and for projects lacking of training data. It aims to increase the dataset size and add in noise to prevent overtraining, by slightly adjusting (translation or rotation) the image data.

- **Classification with separated seizure types**. In my experiment results, some of the seizure types are always missed shot. And as we can see in the videos, seizure types are different in the appearance. Train the detector as multi-class task may improve the detection results in some way.

- **RFID Data and Tracking Features**. As explained in the previous section, tracking features are promising to cover the shortage of motion features. And in the HCA system, we collect the RFID data, which can be transferred into tracking features, as used in the research to detect social behavior (Redfern et al., 2017).

- **More complex classification models**. In the project, I have only used SVM as the classifier; however, there are more models can be used as classifier for the project, such as CNN, HMM or joint models of SVMs. The classifier models can be more complex than a single SVM classifier, and the more complex models may provide more information for the inner relationships between features, and seizure classes (types).

- **Introduce other activity events**. As we've done for other behavior recognition task, the continuous sequence of activity events might have some inner transformation logic. For example, the probabilities for event transformation *Rest-Walk* will be higher than *Rest-Run*. Use probabilistic models like Hidden Markov Model, we might have a more meaningful detection results for seizure events, combining the normal activities. Though this requires a lot more efforts to annotate the activity events, we are more likely to notice some more patterns for the seizure events.

All the methods for the future work can be simplified as 'Being more **Automated**, **Sensitive** and **Specific**'.

Appendix A

Appendix - Multi-Model Test

This appendix includes parts of the experiments plots, which are supported for the choose of models and the analysis of system.

MATERIAL	FEATURE	SVM-(S,C)	TPR	SPC
32×32 TRACKING VIDEO	LTP-DEFAULT	(12,1)	0.39878049	0.99276078
64×64 TRACKING VIDEO	LTPDEFAULT	(11,4)	0.51341463	0.98914116
ORIGIN	LTP-DEFAULT	(11,28)	0.16829268	0.97410685
32×32 TRACKING VIDEO	HOG-(4,4,5)	(13,25)	0.5177665	0.98244652
32×32 TRACKING VIDEO	HOG-(4,4,10)	(12,7)	0.70264766	0.93794618
32×32 TRACKING VIDEO	HOG-(8,8,5)	(12,16)	0.49441624	0.98683489
32×32 TRACKING VIDEO	HOG-(8,8,10)	(12,19)	0.52545825	0.98736958
64×64 TRACKING VIDEO	HOG-(8,8,5)	(12,4)	0.50659898	0.98601207
64×64 TRACKING VIDEO	HOG-(8,8,10)	(13,28)	0.56822811	0.98956617
32×32 TRACKING VIDEO	HOF-(8,8,5)	(12,7)	0.39898477	0.99259462
32×32 TRACKING VIDEO	HOF-(8,8,10)	(12,10)	0.49898167	0.99011532
64×64 TRACKING VIDEO	HOF-(8,8,5)	(13,16)	0.5035533	0.9917718
64×64 TRACKING VIDEO	HOF-(8,8,10)	(13,13)	0.57230143	0.98352554
BBOX INFO	MOTION-WIN5	(12,7)	0.68927126	0.96513862
BBOX INFO	MOTION-WIN10	(13,10)	0.53752535	0.99560198
BBOX INFO	MOTION-WIN25	(13,16)	0.81313131	0.92413793

Table A.1: This table shows experiments results with the best SVM model settings for different feature settings. The choices of best model settings are based on the multi-model test results, using the indicator 'TPR× SPC× SPC', shown in figure **??**.

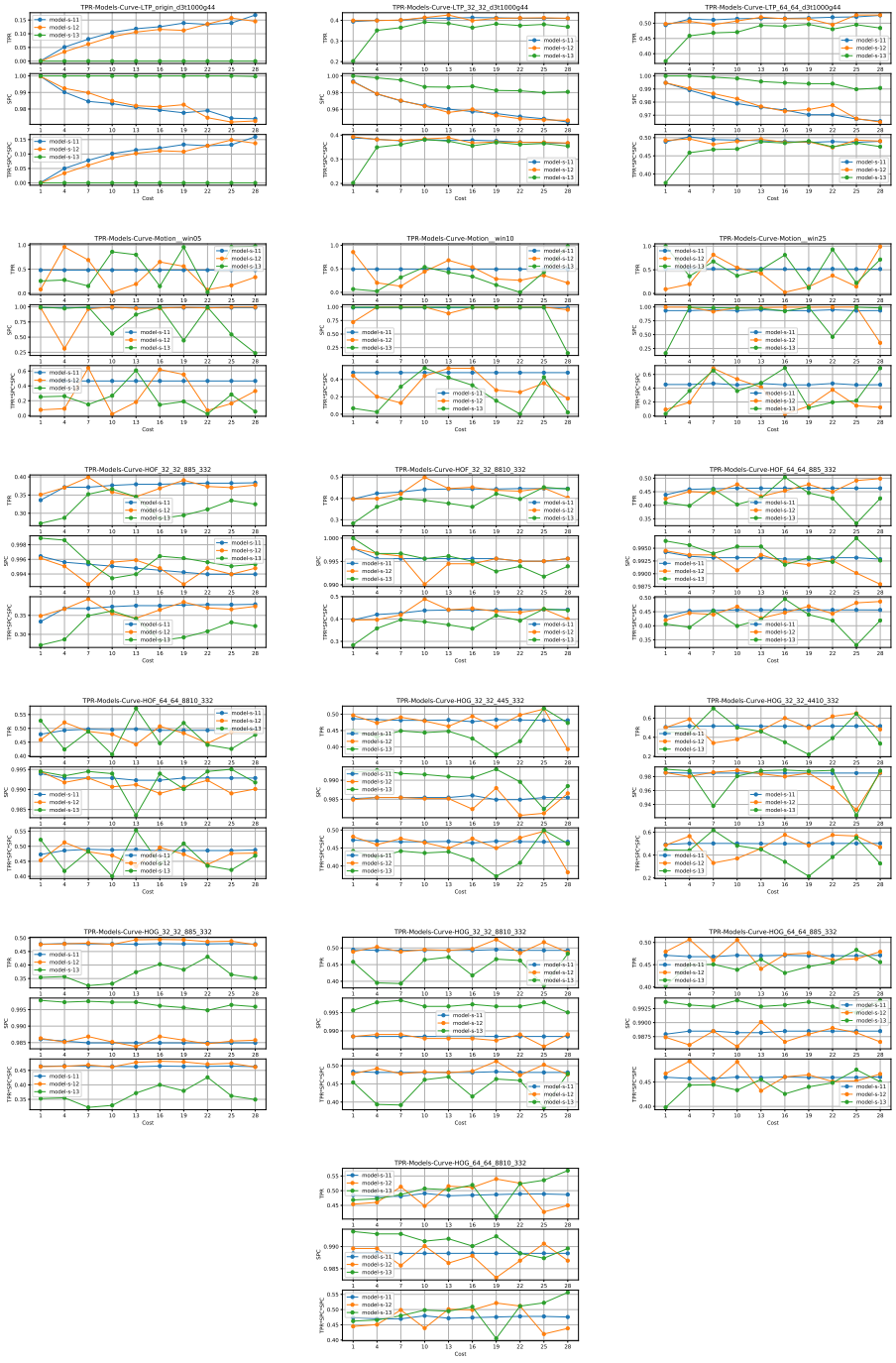

Figure A.1: These are the plots of multi-model tests, with different settings of features using different settings of SVM models. In the experiments, we use TPR$\times$SPC$\times$SPC as our indicator to choose the best combination of models for the further tests. And we can see from the figures that the best TPRs for the models are around 50%.

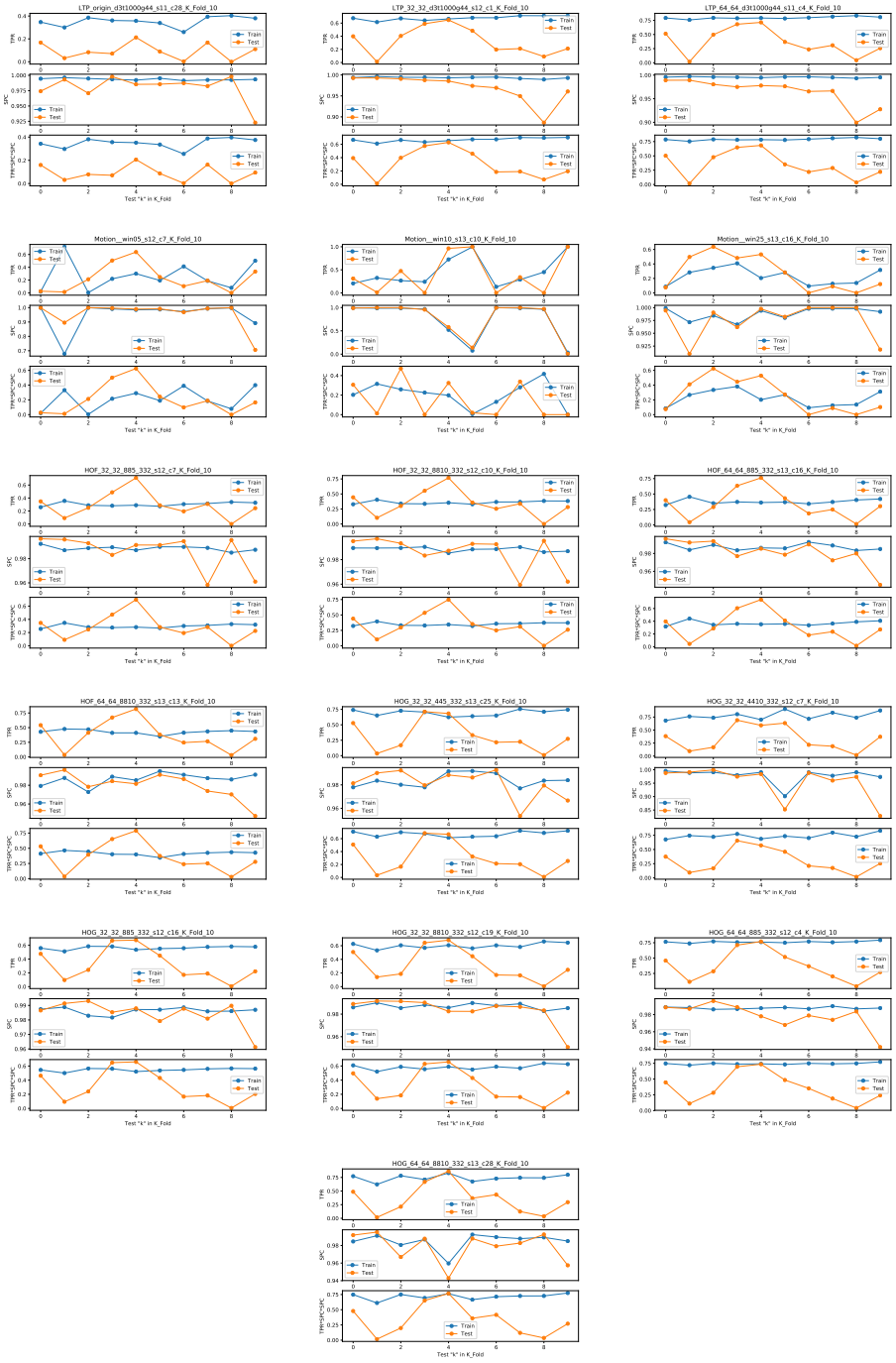

Figure A.2: These are the plots of crossvalidation results with each validation folds of different settings of features using their best SVM models. By comparing the plots, we can see that the results for different folds are with great difference,

Appendix B

Appendix - Best Detection Results

These are the zoomed-in ethograms of predicting results of model 'Win10-T1', the enhanced prediction of model 'HOF-M4'.

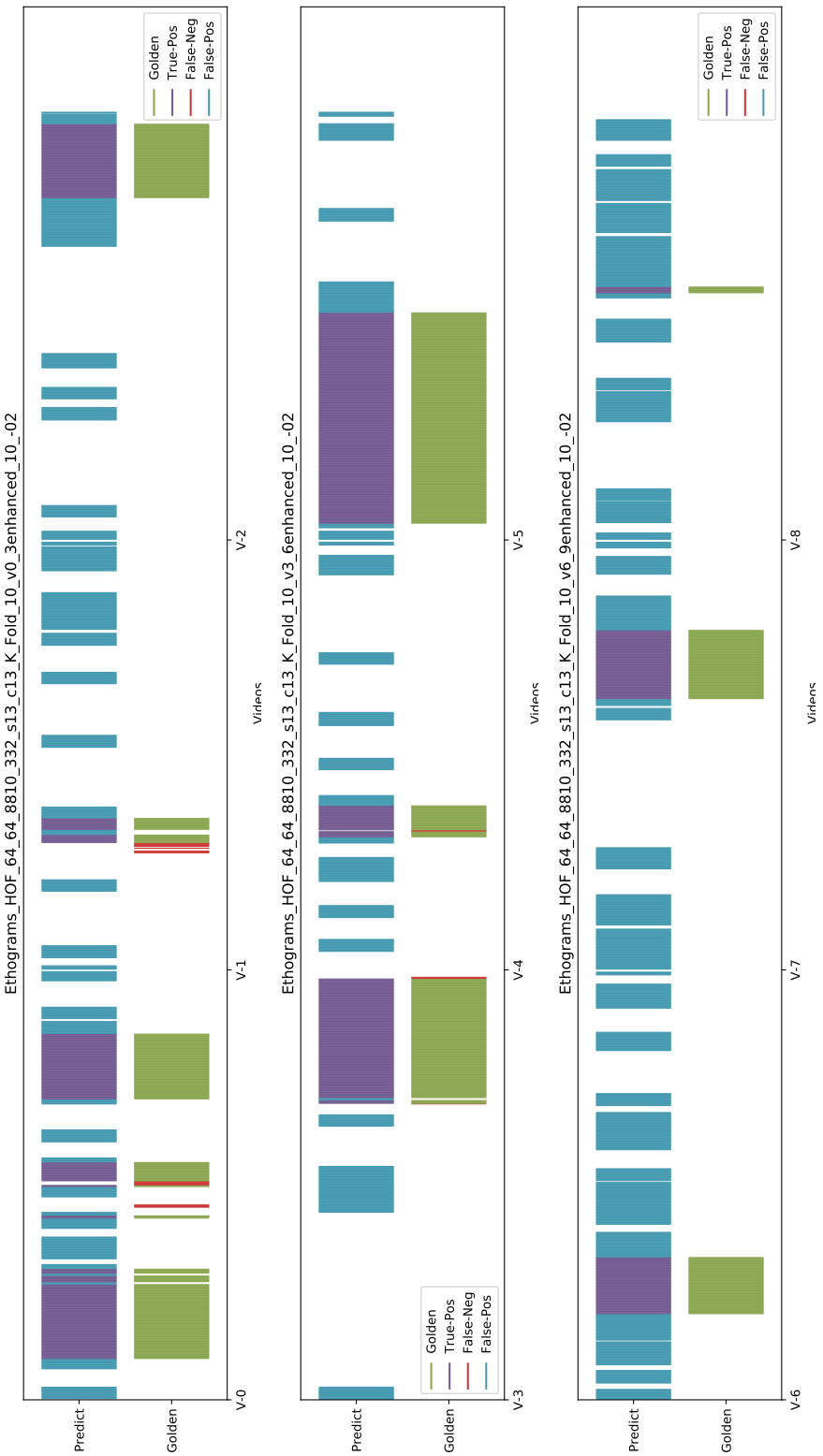
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v0_3enhanced_10_-02
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v3_6enhanced_10_-02
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v6_9enhanced_10_-02
Golden
True-Pos
False-Neg
False-Pos
Predict
Golden
Videos
V-0
V-1
V-2
V-3
V-4
V-5
V-6
V-7
V-8

Figure B.1

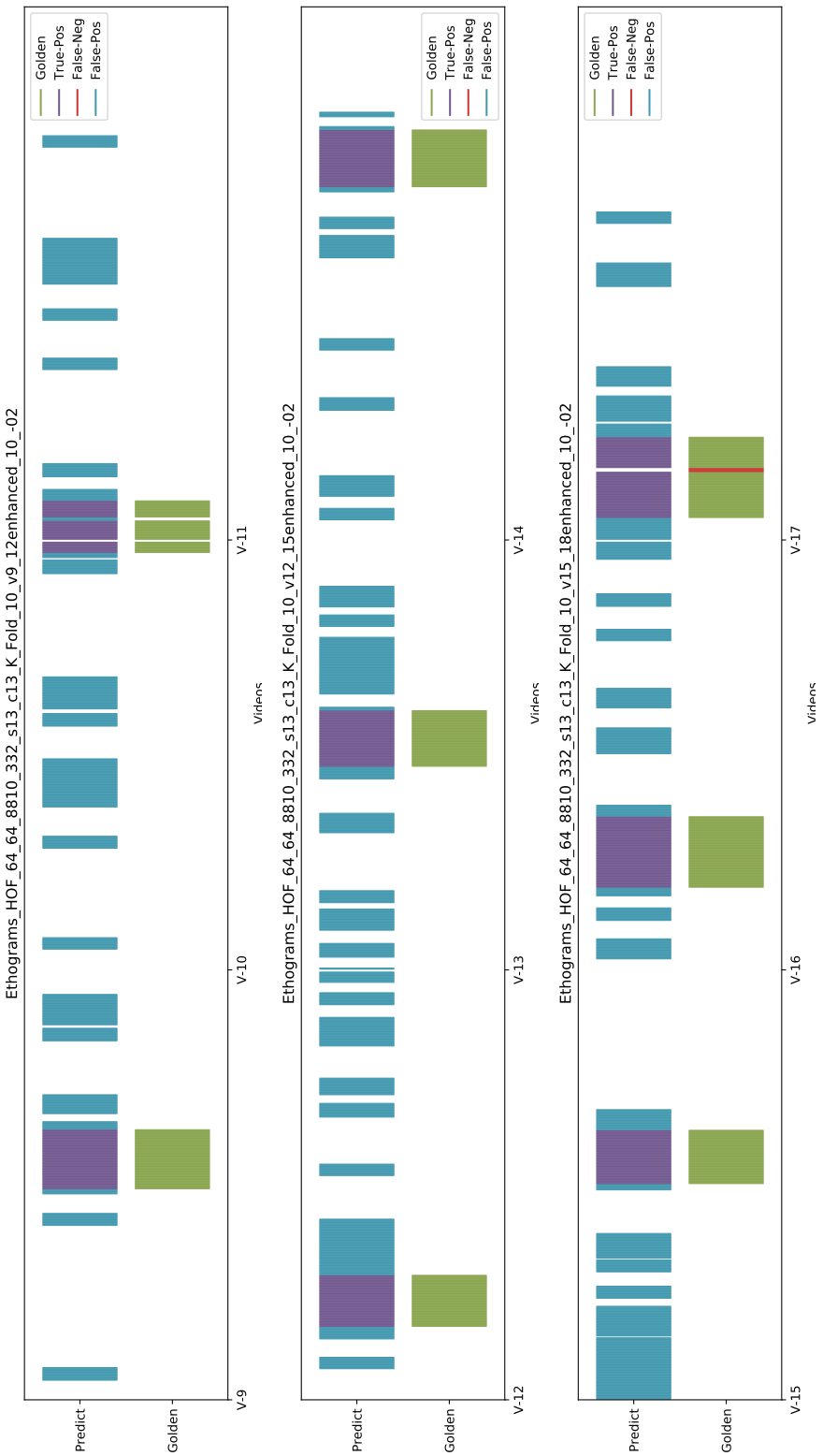
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v9_12enhanced_10_-02
Golden
True-Pos
False-Neg
False-Pos
V-9
V-10
V-11
Videos
Predict
Golden
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v12_15enhanced_10_-02
Golden
True-Pos
False-Neg
False-Pos
V-12
V-13
V-14
Videos
Predict
Golden
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v15_18enhanced_10_-02
Golden
True-Pos
False-Neg
False-Pos
V-15
V-16
V-17
Videos
Predict
Golden

Figure B.2

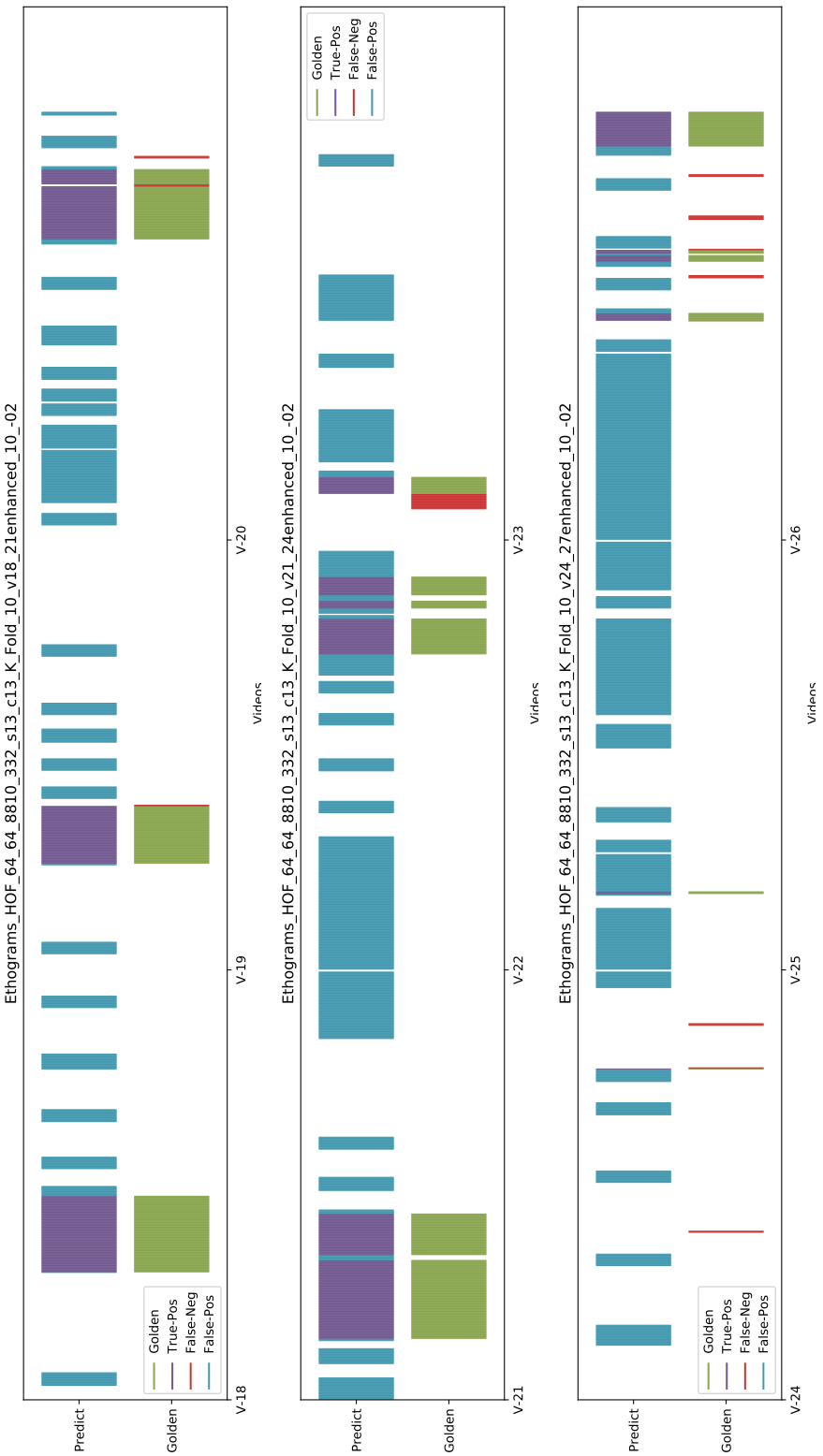
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v18_21enhanced_10_-02
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v21_24enhanced_10_-02
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v24_27enhanced_10_-02
Golden
True-Pos
False-Neg
False-Pos
Predict
Golden
V-18
V-19
V-20
V-21
V-22
V-23
V-24
V-25
V-26
Videos

Figure B.3

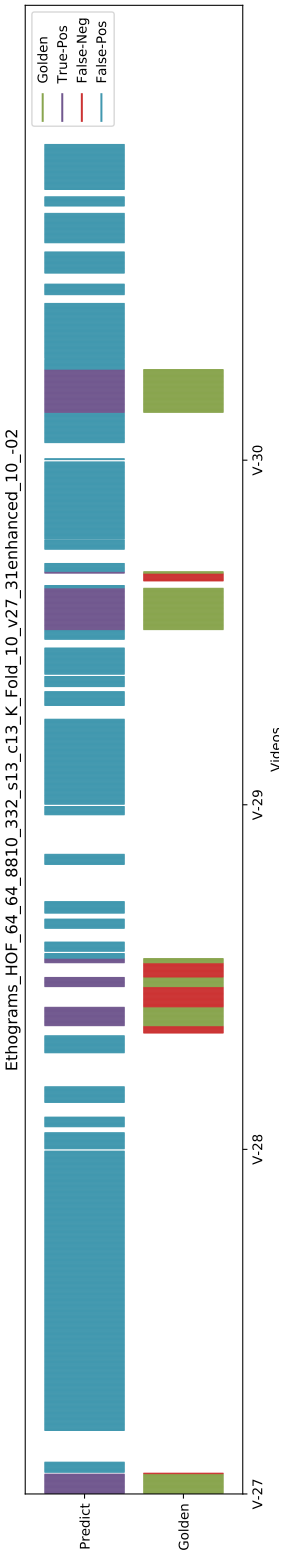
Golden
True-Pos
False-Neg
False-Pos
Ethograms_HOF_64_64_8810_332_s13_c13_K_Fold_10_v27_31enhanced_10_02
V-30
V-29
V-28
V-27
Videos
Predict
Golden

Figure B.4

Bibliography

Bains, R. S., Cater, H. L., Sillito, R. R., Chartsias, A., Sneddon, D., Concas, D., Keskivali-Bond, P., Lukins, T. C., Wells, S., Acevedo Arozena, A., et al. (2016). Analysis of individual mouse activity in group housed animals of different inbred strains using a novel automated home cage analysis system. *Frontiers in behavioral neuroscience*, 10:106.

Barnich, O. and Van Droogenbroeck, M. (2011). Vibe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image processing*, 20(6):1709–1724.

Burgos-Artizzu, X. P., Dollár, P., Lin, D., Anderson, D. J., and Perona, P. (2012). Social behavior recognition in continuous video. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1322–1329. IEEE.

Cuppens, K., Vanrumste, B., Ceulemans, B., Lagae, L., and Van Huffel, S. (2010). Detection of epileptic seizures using video data. In *Intelligent Environments (IE), 2010 Sixth International Conference on*, pages 372–373. IEEE.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE.

De Chaumont, F., Coura, R. D.-S., Serreau, P., Cressant, A., Chabout, J., Granon, S., and Olivo-Marin, J.-C. (2012). Computerized video analysis of social interactions in mice. *Nature methods*, 9(4):410.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874.

Fisher, B. (2018). Lecture slides of advanced vision, msc course in school of informatics, the university of edinburgh.

Fisher, R. S., Boas, W. v. E., Blume, W., Elger, C., Genton, P., Lee, P., and Engel, J. (2005). Epileptic seizures and epilepsy: definitions proposed by the international league against epilepsy (ilae) and the international bureau for epilepsy (ibe). *Epilepsia*, 46(4):470–472.

Fisher, R. S., Cross, J. H., French, J. A., Higurashi, N., Hirsch, E., Jansen, F. E., Lagae, L., Moshé, S. L., Peltola, J., Roulet Perez, E., et al. (2017). Operational classification of seizure types by the international league against epilepsy: Position paper of the ilae commission for classification and terminology. *Epilepsia*, 58(4):522–530.

Fonck, C., Cohen, B. N., Nashmi, R., Whiteaker, P., Wagenaar, D. A., Rodrigues-Pinguet, N., Deshpande, P., McKinney, S., Kwoh, S., Munoz, J., et al. (2005). Novel seizure phenotype and sleep disruptions in knock-in mice with hypersensitive α4* nicotinic receptors. *Journal of Neuroscience*, 25(49):11396–11411.

Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. *Artificial intelligence*, 17(1-3):185–203.

Iasemidis, L. D. (2011). Seizure prediction and its applications. *Neurosurgery Clinics*, 22(4):489–506.

Jansen, K. and Lagae, L. (2010). Cardiac changes in epilepsy. *Seizure-European Journal of Epilepsy*, 19(8):455–460.

Jhuang, H., Garrote, E., Yu, X., Khilnani, V., Poggio, T., Steele, A. D., and Serre, T. (2010). Automated home-cage behavioural phenotyping of mice. *Nature communications*, 1:68.

Jin, C., Zhang, J., Li, X., Yang, X., Li, J., and Liu, J. (2013). Injectable 3-d fabrication of medical electronics at the target biological tissues. *Scientific reports*, 3:3442.

Karayiannis, N. B., Srinivasan, S., Bhattacharya, R., Wise, M. S., Frost, J. D., and Mizrahi, E. M. (2001). Extraction of motion strength and motor activity signals from video recordings of neonatal seizures. *IEEE Transactions on medical imaging*, 20(9):965–980.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lidster, K., Jefferys, J. G., Blümcke, I., Crunelli, V., Flecknell, P., Frenguelli, B. G., Gray, W. P., Kaminski, R., Pitkänen, A., Ragan, I., et al. (2016). Opportunities for improving animal welfare in rodent models of epilepsy and seizures. *Journal of neuroscience methods*, 260:2–25.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Nijsen, T. M., Arends, J. B., Griep, P. A., and Cluitmans, P. J. (2005). The potential value of three-dimensional accelerometry for detection of motor seizures in severe epilepsy. *Epilepsy & Behavior*, 7(1):74–84.

Ohayon, S., Avni, O., Taylor, A. L., Perona, P., and Egnor, S. R. (2013). Automated multi-day tracking of marked mice for the analysis of social behaviour. *Journal of neuroscience methods*, 219(1):10–19.

Pediaditis, M., Tsiknakis, M., and Leitgeb, N. (2012). Vision-based motion detection, analysis and recognition of epileptic seizuresa systematic review. *Computer methods and programs in biomedicine*, 108(3):1133–1148.

Pitkänen, A., Buckmaster, P., Galanopoulou, A. S., and Moshé, S. L. (2017). *Models of seizures and epilepsy*. Academic Press.

Racine, R. J. (1972). Modification of seizure activity by electrical stimulation: Ii. motor seizure. *Electroencephalography and clinical neurophysiology*, 32(3):281–294.

Ramgopal, S., Thome-Souza, S., Jackson, M., Kadish, N. E., Fernández, I. S., Klehm, J., Bosl, W., Reinsberger, C., Schachter, S., and Loddenkemper, T. (2014). Seizure detection, seizure prediction, and closed-loop warning systems in epilepsy. *Epilepsy & behavior*, 37:291–307.

Redfern, W. S., Tse, K., Grant, C., Keerie, A., Simpson, D. J., Pedersen, J. C., Rimmer, V., Leslie, L., Klein, S. K., Karp, N. A., et al. (2017). Automated recording of home cage activity and temperature of individual rats housed in social groups: The rodent big brother project. *PloS one*, 12(9):e0181068.

Suykens, J. A. and Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300.

Uijlings, J., Duta, I. C., Sangineto, E., and Sebe, N. (2015). Video classification with densely extracted hog/hof/mbh features: an evaluation of the accuracy/computational efficiency trade-off. *International Journal of Multimedia Information Retrieval*, 4(1):33–44.

van Dam, E. A., van der Harst, J. E., ter Braak, C. J., Tegelenbosch, R. A., Spruijt, B. M., and Noldus, L. P. (2013). An automated system for the recognition of various specific rat behaviours. *Journal of neuroscience methods*, 218(2):214–224.

Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time-sequential images using hidden markov model. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 379–385. IEEE.

Yeffet, L. and Wolf, L. (2009). Local trinary patterns for human action recognition. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 492–497. IEEE.