DISSERTATIONES
TECHNOLOGIAE
UNIVERSITATIS
TARTUENSIS
74

KADIR AKTAS

Cosmic Ray Tomography based Object Reconstruction and Recognition

DISSERTATIONES TECHNOLOGIAE UNIVERSITATIS TARTUENSIS
74

KADIR AKTAS

Cosmic Ray Tomography based Object Reconstruction and Recognition

UNIVERSITY OF TARTU
Press
1632

Institute of Engineering and Technology, Faculty of Science and Technology, University of Tartu, Estonia.

Dissertation has been accepted for the commencement of the degree of Doctor of Philosophy (PhD) in Physical Engineering on 18.09.2023 by the Council of the Institute of Engineering and Technology, Faculty of Science and Technology, University of Tartu.

Supervisors

Prof. Gholamreza Anbarjafari
University of Tartu
Tartu, Estonia

Dr. Madis Kiisk
University of Tartu
Tartu, Estonia

Dr. Andrea Giammanco
Université Catholique de Louvain
Louvain-la-Neuve, Belgium

Opponent

Prof. Ahmet Enis Çetin
University of Illinois
Chicago
USA

The public defense will take place on 31.10.2023 at 16:15 in Nooruse 1-121.

The publication of this dissertation was financed by the Institute of Engineering and Technology, University of Tartu.

Copyright © 2023 by Kadir Aktas

ISSN 2228-0855 (print)
ISSN 2806-2620 (PDF)
ISBN 978-9916-27-364-7 (print)
ISBN 978-9916-27-365-4 (PDF)

University of Tartu Press
www.tyk.ee

ABSTRACT

Improvement of the technology and successful research in artificial intelligence (AI) field have resulted in the surge of using AI-based methods to handle low-level tasks such as object recognition. Object recognition aims to find the objects within an image or sequence of images. The goal is to localize the existing objects within the image and identify the classes of them. Due to its wide application areas in the significant real-world problems, object recognition is one of the main tasks in computer vision. Some examples of these applications include tomography systems, human-behaviour analysis, medical imaging, and sports. Although the object recognition has been a hot topic for quite some time, there are still many challenges remain due to the wide range of application areas. For example, such emerging technologies as cosmic ray tomography have a big research gap for utilizing deep learning techniques to improve the system's object recognition performance. In order to improve the reconstruction and recognition performance in a cosmic ray tomography system, a neural network based approach is presented in this thesis. A cosmic ray tomography system uses the hit positions of muons on detector plates to reconstruct the images of volume of interest (VOI). Afterwards, these images are used to recognize the objects. Therefore, the recognition performance is directly impacted by the reconstruction performance, hence also by the muon hit position estimation. A significant improvement over the conventional Center of Gravity (CoG) method is obtained by the utilisation of deep neural networks (DNN) for this task. Moreover, in this thesis, object recognition methods based on deep convolutional neural networks (DCNN) are presented. Their capabilities to recognize the objects from a single image and a time-series data are presented. Successful performances are demonstrated through the experimentation with challenging tasks on the diverse datasets.

CONTENTS

LIST OF ORIGINAL PUBLICATIONS

Publications included in the thesis

1. ***Aktas, K.**, Kiisk, M, Giammanco, A., Anbarjafari, G., Mägi, M.*: **A Comparison of Neural Networks and Center of Gravity in Muon Hit Position Estimation.** Entropy 2022, 24, 1659.

   Kadir Aktas' contributions: Conceptualization, Formal analysis, Methodology, Software, Validation, Visualization, Writing—original draft, Writing—review & editing

2. ***Aktas, K.**, Ignjatovic, V., Ilic, D., Marjanovic, M., Anbarjafari, G.*: **Deep Convolutional Neural Networks for Detection of Abnormalities in Chest X-rays Trained on the Very Large Dataset.** Signal, Image and Video Processing 17, 1035–1041 (2023).

   Kadir Aktas' contributions: Conceptualization, Formal analysis, Methodology, Software, Validation, Visualization, Writing—original draft, Writing—review & editing

3. ***Aktas, K.**, Demirel, M., Moor, M., Olesk, J., Ozcinar, C., Anbarjafari, G.*: **Spatiotemporal Based Table Tennis Stroke-type Assessment.** Signal, Image and Video Processing 15, 1593–1600 (2021).

   Kadir Aktas' contributions: Conceptualization, Formal analysis, Methodology, Project administration, Software, Validation, Visualization, Writing—original draft, Writing—review & editing

Publications not included in the thesis

1. *Pavlovs, I., **Aktas, K.**, Avots, E., Vecvanags, A., Filipovs, J., Brauns, A., Done, G., Jakovels, D., Anbarjafari, G.*: **Ungulate Detection and Species Classification from Camera Trap Images Using RetinaNet and Faster R-CNN.** Entropy 2022, 24, 353.

2. ***Aktas, K.***, Sham, A.*, Rizhinashvili, D., Kuklianov, D., Alisinanoglu, F., Ofodile, I., Ozcinar, C., Anbarjafari, G.*: **Ethical AI in Facial Expression Analysis: Racial Bias.** Signal, Image and Video Processing 17, 399–406 (2023).
   * – shared first author.

3. *Kamińska, D., **Aktas, K.**, Rizhinashvili, D., Kuklyanov, D., Sham, A. H., Escalera, S., Nasrollahi, K., Moeslund, T. B., Anbarjafari, G.*: **Two-Stage Recognition and beyond for Compound Facial Emotion Recognition.** Electronics 2021, 10, 2847.

LIST OF ABBREVIATIONS

Acronyms

AI artificial intelligence. 5, 13, 14, 28, 49

CNN convolutional neural networks. 12, 15, 16, 19, 20, 22, 25, 27, 28, 30, 31, 33, 36

CoG Center of Gravity. 5, 11, 12, 15, 16, 24–27

COVID Coronavirus Disease. 13

DCNN deep convolutional neural networks. 5, 14, 28, 29, 36, 40, 49

DNN deep neural networks. 5, 13, 20, 49

FCN fully connected network. 11, 12, 19–23, 25–27

FPGA field-programmable gate array. 15

GWO Grey Wolf Optimizer. 28

kNN K-Nearest Neighbor. 45

LSTM Long Short-Term Memory. 14, 38–40, 48, 49

MCC Matthews Correlation Coefficient. 34

MLP multilayer perceptron. 15, 19, 28

MSE Mean squared error. 21

PET positron emission tomography. 19

ReLU Rectified Linear Unit. 32

RGB Red-Green-Blue. 14, 38–40, 46, 48

ROI Region of Interest. 39, 40, 42

SiPMs silicon photomultipliers. 12, 15–18, 22, 23

SSTCNN Siamese Spatio-Temporal Convolutional Neural Network. 39

TSTCNN Twin Spatio-Temporal Convolutional Neural Network. 39

VOI volume of interest. 5, 13–15

YOLO You Only Look Once. 40, 41

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The successful research in AI field and the level of development of modern technologies created a demand for AI-based methods to handle such tasks as object recognition [1]. In general, object recognition focuses on the identification of certain objects within an image or sequence of images. It aims to localize the existing objects within the image and identify their classes [2]. Object recognition represents one of the main tasks in the computer vision field, and it can be applied to various real-world problems, for example, to tomography systems, animal recognition, human-behavior analysis, medical imaging, and sports [3–7].

Even though recently, object recognition received increased attention from researchers, many challenges still remain unresolved due to the wide range of application fields [8–10]. For instance, topics like cosmic ray tomography provide a significant research gap for utilizing deep learning methods to improve the system's object recognition performance. Another example is Coronavirus Disease (COVID) detection, where the lack of data in the previous research provides opportunities for further studies [11]. Moreover, even more popular tasks, e.g. human action recognition, present challenges due to the reasons such as the possible target variations that AI needs to differentiate from [12]. This thesis' major goal is to address the gaps in object recognition research. The primary focus in this thesis is on deep learning methods and their wide range of applications. The following chapters explore how the proposed approaches can be tailored and potentially used on a variety of data sources, including cosmic ray tomography.

Chapter 1 is dedicated to the utilization of DNN for improvement of object reconstruction and recognition in a cosmic ray tomography system. A cosmic ray tomography system usually consists of detector plates that determine the position of the muons that are passing through them [13]. This way, the trajectory of the muons can be estimated which allows observation of the scattering caused by the materials in VOI [14]. The scattering information is used by common cosmic ray tomography devices to reconstruct the VOI images which are later utilized to recognize the objects in VOI [15]. Therefore, the improvement of the reconstruction quality is crucial for the object recognition. Even though the research that utilize the deep learning techniques in cosmic ray tomography are limited, nowadays it becomes a popular topic in the field, as the hardware limitations on the muon tomography systems have been now resolved [16–18].

Tomography systems have found various applications in medical imaging. Also, AI holds a great value in the field of medical imaging due to its powerful reconstruction and recognition capabilities [19]. However, there is still a research gap in automating the identification of diseases using AI in the tomography systems. For example, the global community encountered a significant challenge in accurately and efficiently identifying individuals afflicted with COVID during the pandemic [20]. To address this issue, Chapter 2 of this work focuses on COVID detection using tomography images by utilizing deep learning techniques. We conducted

our research using X-ray chest scans. Our DCNN was trained on a dataset comprising 103,468 images and achieved an impressive accuracy of 97% in detecting the target disease. The results from our study demonstrate the successful application of our AI method to chest scan images, highlighting its potential utilization in the healthcare domain.

Various AI tasks have series-based data such as temporal data or image slices of a 3D VOI [21–23]. Human action recognition is one of them where the data is a video of a person that is performing a certain action. This video data is usually examined frame by frame to recognize the target action [24]. Because of this, action recognition tasks are a good example for analysis of data composed of a set of images. In Chapter 3, we conducted a spatio-temporal recognition study. We demonstrated how deep learning techniques, e.g. Long Short-Term Memory (LSTM) with a feature extraction backbone, can be utilized to extract and classify the spatio-temporal features in a Red-Green-Blue (RGB)-based time-series data. We investigated the effect of data augmentation as well. We utilized a table tennis dataset as an example to showcase our methods. The proposed method have shown remarkable success in the given classification task achieving 88.6% accuracy. Considering that a cosmic ray tomography produces a 3D VOI output that is a series of image slices, our method can be applied to it for extracting the spatial features in each image slice and their relationship between the other slices in the VOI to recognize certain targets e.g. a dangerous material or a harmful object. The automated detection of materials of interest can be useful in various applications such as industrial inspection and security screening.

Finally, the thesis is concluded by summarizing the contents and discussing the opportunities for future work.

1. DEEP NEURAL NETWORK-BASED MUON HIT POSITION ESTIMATION

1.1. Introduction

Cosmic ray tomography systems are created by combining several plates that are placed around the VOI [25–28]. These plates are able to detect the impact of electrically charged particles. Thanks to this, they provide an opportunity to estimate the trajectory of particles like muons when they pass through the cleverly placed plates. In [26], the authors use six detector plates by placing half of them below and remaining half above the material of interest. They improve the precision of the detection of muon trajectories through the system thanks to the multiple plates placed around the target. Commonly used detector plates are fabricated from luminescent materials. Due to the characteristics of such materials, a light is produced when a muon passes through the plate. Additionally, SiPMs are attached to the plates to collect the lights produced by the material. Thus, when a muon hits to the detector plate, the signal amplitudes of SiPMs increases. Sensors which are closer to the hit position output the highest signal amplitudes since the light is more intense near them. Finally, the hit position of the muon is estimated by processing the signal amplitudes from all the sensors together.

Estimating the hit position from signal amplitudes is performed using different processing methods, with conventional analytical techniques like the CoG being widely adopted [29, 30]. However, due to the poor performance of these methods, researchers have looked for alternatives. In [31], the authors used Anger logic to localize muon hits in a simulated plate detector. Moreover, with the advancement of technology, machine learning-based methods have recently gained popularity alongside analytical methods for hit position estimation. Their superior spatial resolution and faster processing capabilities have been explored in various studies. Additionally, there are evidences that machine learning-based methods produce promising results. For example, in [32], the authors present a more efficient real-time estimation by implementing multilayer perceptron (MLP) network with 2×8 neurons on an field-programmable gate array (FPGA). In [33], the authors used a similar network architecture as in [32], but approached the data configuration and network training differently. They split the x and y axes into 7 intervals each and trained a global network to roughly predict the x and y position, then used sub-interval networks to find the exact location. The authors trained 7 neural networks for x and 7 neural networks for y as sub-interval networks. In [34], a CNN was used to decode gamma ray interaction locations, achieving an average spatial resolution of 0.40 mm. The authors collected light distribution data as a 16-pixel image using sensors placed at the edges of the plate and used a network consisting of a CNN layer with 200 filters of size 1x1 combined with a fully connected layer.

The accuracy of muon hit position estimation is critical for cosmic ray-based tomography systems, as it affects the calculation of muon trajectories and thus the overall imaging performance. Despite some efforts to improve hit position estimation, the literature is limited, especially regarding the deep learning-based methods. In this study, we aim to improve the positional resolution by replacing conventional methods (CoG) with deep neural networks. We compare the performance of fully connected and CNN-based architectures to CoG using resolution data. The optimization of physical detection systems is not part of the scope of this study. This research presents three main contributions:

- A simulated dataset of muon hit positions on a detector plate is presented.
- Two deep learning-based methods for muon hit position estimation are introduced and their performance is compared to that of CoG. The proposed method's performance is also analyzed in detail.
- The impact of different SiPMs configurations on performance is evaluated.

The rest of the chapter is structured as follows: The dataset is described in Section 1.2, the proposed method is outlined in Section 1.3, the experimental results are presented and discussed in Section 1.4, and the conclusions of this work are summarized in Section 1.5.

1.2. Dataset

In order to create our dataset, we conducted a characterization of a muon detector that utilized a plastic scintillator slab. This was achieved through the use of the Monte Carlo simulation package GEANT4 (version 10.5) [35].

GEANT4 is a powerful software toolkit used in particle physics to simulate the interactions of particles with matter. It uses a combination of models and algorithms to calculate the probabilities of various physical processes occurring, such as particle scattering, ionization, and decay. Based on these probabilities, GEANT4 generates a set of synthetic data that represents the interactions of particles with the detector material. The synthetic data generated by GEANT4 is useful for a variety of purposes, such as testing and optimizing the performance of particle detectors, evaluating new experimental techniques, and validating theoretical models. However, it is important to keep in mind that the simulated data is not exactly the same as real-world data obtained from physical experiments.

Our simulations in GEANT4 were based on the following key parameters:

- The EJ-200 plastic scintillator is used with a light output of 10,000 photons per MeV.
- The plate has a surface area of 20 cm $\times$ 20 cm (400 cm^2) with a thickness of 10 mm.
- The SiPMs are placed on top of the surface with a spacing of 20 mm between them. The 20 mm gap is decided heuristically with the aim of keeping

the sensor number low for cost optimization. All SiPMs are the same and they have a size of 6 mm $\times$ 6 mm.

- The SiPMs have 40% detection efficiency.
- A diffusive reflection material, which is covered with TiO_2 paint, is used in the simulations as the reflective material. The reflection efficiency is 95%.

The depiction of the simulated detector plate is shown in Figure 1. This figure highlights the red squares, which are the SiPMs, and the green surface that represents the scintillator area. The arrangement of the SiPMs is facing the top of the detector plate, and a symmetrical placement is established along both axes, considering the square shape of the plate.

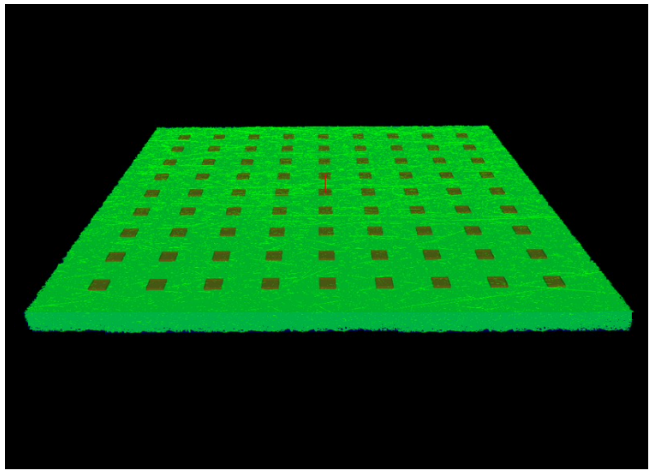


Figure 1: The simulation of detector plate.

In our simulation, we only focused on simulating muon hits in one specific area instead of the entire surface. As seen in Figure 2, the simulated hit locations are displayed as red points. We decided to focus on a smaller area than the whole surface to save the computation power and simulation time. Another option would be increasing the step size to cover the whole plate. However, analyzing how the methods' perform when the hits are located in between the sensors is one of the goals and we did not want to weaken this side of the experiments. We limited our hit locations to the specified area due to the periodic invariance by translation of the system. The muons were emitted perpendicular to the detector surface in a grid with a step of 1mm, leading to 441 different hit positions. We simulated around 10,000 hits for each hit position, resulting in a total simulated hit count of 4,416,335 ($\sim$441 $\times$ 10,000). When a muon hits the detector plate, the resulting scintillation light spreads throughout the plate and is detected by SiPMs. We recorded the signal amplitudes from the SiPMs and the hit coordinates (x, y) for each hit to create a dataset of hit coordinates and corresponding signal amplitudes. The sensors are positioned in the middle of the plate from the side view, so that

when a muon hits the exact location of a sensor, it produces light on the plate as expected. Due to this setup, the sensor at the hit location has the highest signal amplitude, as it is closest to the source. Also, the z-coordinates are kept the same for all the hits.

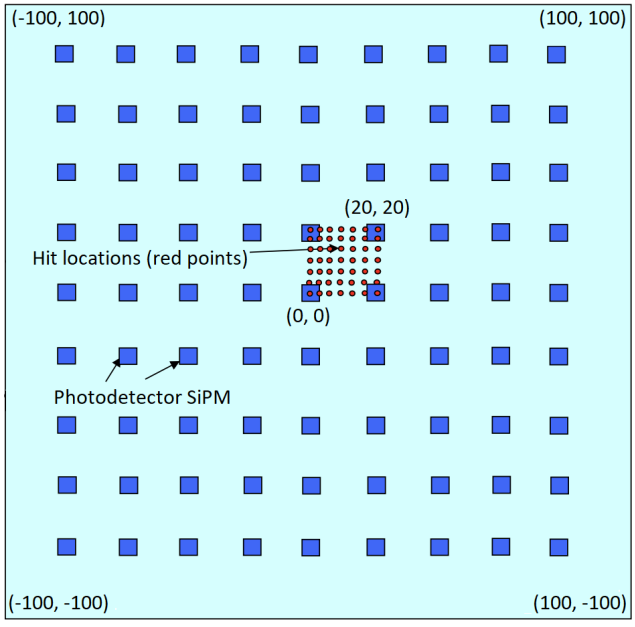


Figure 2: Hit locations selected on the simulated plate, in mm.

Finally, we reduced the dataset by excluding the signal amplitudes of outer sensors. To obtain a sub-dataset with 64 (8×8) sensors, for instance, we removed the outer readings from the data. Similarly, we generated datasets with 49 (7×7), 36 (6×6), 25 (5×5), and 16 (4×4) sensors, each by eliminating the corresponding outer signals. This allowed us to explore the impact of different SiPMs configurations on performance. Note that the hit locations and parameters remained unchanged because we created the sub-sets by cropping the 81-sensor data.

It is possible to display the illumination resulting from a single-particle hit as an image by treating each signal amplitude as a pixel. Figure 3 illustrates samples from (0, 0) and (20, 20). As can be seen from the samples, the illumination output varies with the changes in the hit coordinate. To demonstrate a clear change, outputs from hits at (0, 0) and (20, 20) coordinates are displayed. The position of brighter pixels, which correspond to higher signal amplitudes from sensors, shifts based on the hit location.

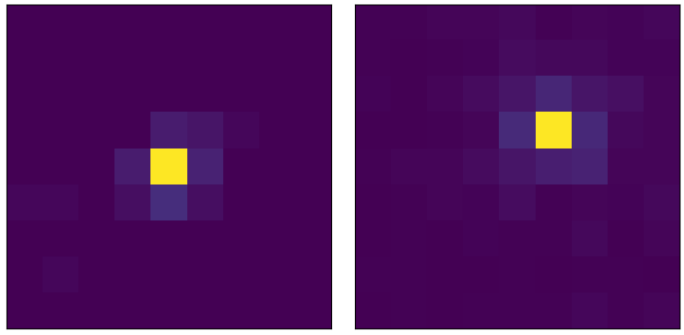


Figure 3: Signal amplitude samples for hits at (0, 0) and (20, 20) mm, respectively, with brighter color indicating higher amplitude increase.

1.3. Proposed Methods

In this chapter, we present two methods, FCN and CNN, for estimating the 2D hit position of muons on scintillation detector plates. These methods are used in previous research and have demonstrated the use of neural networks to achieve high spatial resolution in particle hit position estimation in positron emission tomography (PET) systems [32–34]. Given the similarities between our task and those previous studies, we decided to employ neural networks in our methods.

1.3.1. Fully Connected Network

In our study, we chose to build our FCN based on an MLP network, which had been shown to be successful in similar tasks in the previous research [36, 37]. The network consists of several key components: an input layer, a batch normalization layer, a flattened layer, two fully connected layers with 256 and 32 neurons, and an output layer. The purpose of the batch normalization layer is to regularize the input, while the flattened layer transforms the input into a shape suitable for the fully connected layers. The features are then extracted through the fully connected layers, and the estimated position coordinate is obtained through a single-neuron output layer.

Instead of estimating both x and y coordinates through a single model, we considered them separately, and, trained two separate FCN models, one for x and one for y. The end-to-end structure of our proposed method can be seen in Figure 4. The signal amplitudes are passed as input to each model, and the x and x coordinate predictions are obtained from the separate networks. Finally, a fusion step combines the two predictions to obtain the final (x, y) coordinate as a single output.

1.3.2. Convolutional Neural Network

A scintillation detector plate employs photosensors to convert muon impact information into a digital format. For instance, in a system with 81 (9×9) photo-

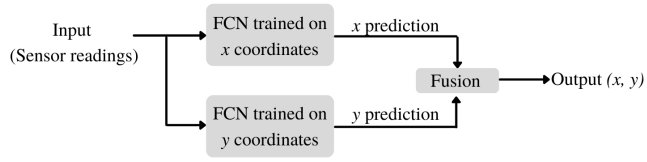


Figure 4: Structure from input to output.

sensors, the signal amplitude from each sensor is passed to a processor, creating a 9×9 image. The appearance of this image varies based on the position of the muon impact, as the signal amplitudes on each sensor differ due to variations in the plate's illumination (Figure 3 illustrates this).

A convolutional layer is the fundamental component in convolutional neural networks. In a convolutional layer, a small filter, also known as a kernel, slides or convolves across the input data, such as an image. At each position, the filter computes the element-wise product between its values and the corresponding values in the input data, and then sums up these products to produce a single value in the output, which becomes a pixel in a new representation called a feature map. The output for a single pixel from the convolutional layer is calculated using the formula provided in [38].

$$y_{i,j} = \sum_m \sum_n w_{m,n} x_{(i+a),(j+b)} \tag{1.1}$$

where x is the input, y is the output of next layer, and w is the kernel.

The process of sliding the filter and computing these products and sums is repeated across the entire input data. Thanks to this approach, a convolutional layer automatically learns and extracts features from the input data, particularly images. Using convolutional layers, DNN have been able to extract the meaningful features from the images, therefore obtain successful results in various image analysis tasks [5, 6, 39–41]. Since our input data can be viewed as an image, it is possible to handle the task as an image processing task. Therefore, a CNN can be utilized to capture visual patterns.

The proposed CNN-based architecture is built using convolutional layers and includes an input layer, a batch normalization layer for regularization, two convolutional layers with 64 and 32 depths and 3×3 kernels for feature extraction, and an output layer to reduce the network result to a single coordinate. The FCN models in the pipeline (as shown in Figure 4) were simply replaced with the CNN models in our experiments regarding this method.

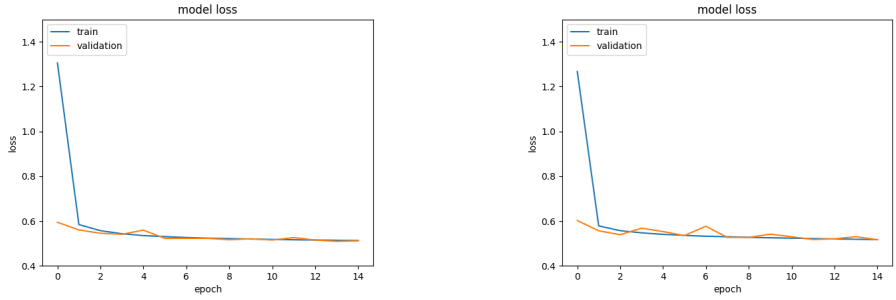
1.3.3. Training

We followed a consistent training method for all sensor configurations, only differing in the number of sensor readings used. The preparation of datasets for different configurations is described in Section 1.2. We split the data into three groups - training (60%), validation (20%), and testing (20%) - with the training

and validation data being used to train the model and the test data used solely to evaluate the model's performance after training.

In the training process, the model repeatedly processes the data and learns its characteristics over a specified number of iterations, known as epochs. During each epoch, the model updates itself after processing batches of data, with the size of the batch and number of epochs being adjustable parameters [42]. In our experiments, we set 100 epochs and a batch size of 32, which proved to be effective. To avoid overfitting, we also implemented an early stopping mechanism. The early stopping mechanism was established to keep track of the validation loss, a critical measure that reflects how well the model performs on data it hasn't encountered before. Our training procedure was devised to stop automatically if the validation loss showed no signs of enhancement over a sequence of 10 successive epochs. Due to this, our training has stopped at 14th epoch for FCN model and at 11th epoch for CNN model. In the training graph (Figure 5), evidence of overfitting is not present, however, the stagnation of the loss is visible.

We employed the Adam optimizer [43] with a learning rate of 0.001, and applied the same training process for all networks. The training curves for two models trained to predict x and y coordinates using 81-sensor data are shown in Figure 5, displaying a quick convergence.



(a) (b)

Figure 5: Training curve for FCN using 81-sensor data. (**a**) Training of the model for x coordinate prediction. (**b**) Training of the model for y coordinate prediction.

The Mean squared error (MSE) is employed as the performance metric during the training phase.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(x_i' - x_i)^2 \tag{1.2}$$

where the equation involves N, the number of samples, x, the actual value, and x', the predicted value.

The experiments were conducted on a laptop running Windows 10, using AMD Ryzen 7 5800H central processing unit, NVIDIA GeForce RTX 3070 Laptop graphics processing unit, and 32 gigabytes of memory. FCN model has 29 253

parameters, and CNN model has 19 141 parameters. Both models complete the training of one epoch approximately in 2 minutes in the specified setup.

For both FCN and CNN models, we utilized Keras library, a high-level neural networks API, for the implementation. Keras library offers compatibility with various backend frameworks, and in our case, we employed the TensorFlow backend for our implementation. This choice provided us with a versatile and efficient platform to develop and train our neural network for the intended task.

The parameters of the networks are selected heuristically. The aim was to identify a configuration that yielded a favorable performance. Also, we tried to keep the network simple while still complex enough to achieve the learning to provide a better generalization as suggested in the literature [44].

1.4. Experimental Results and Discussion

Initially, we sought to determine which model, FCN or CNN, would perform better. To do this, we used the signal amplitudes from 81 sensors as the input data. We evaluated the accuracy of the predicted hit coordinates by comparing them to the actual hit coordinates, calculating the error using the Euclidean distance and taking the average of all errors to obtain a single average error value as the result.

$$Error = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_i' - x_i)^2 + (y_i' - y_i)^2} \tag{1.3}$$

where the equation includes N, the number of samples,x and y for the real positions and x' and y' for the predicted positions on the x-axis and y-axis, respectively.

The performance of both FCN and CNN is presented in Table 1. Although both networks are able to achieve an error below 0.8 mm, FCN performs slightly better. This may be due to the fact that the CNN's method of extracting features by relating pixels to their neighbors through the convolution operation may not be effective enough with the small image size. Because, the number of pixels with significant values is limited in this case. As a result of this, FCN was selected as the preferred method for further experiments.

Table 1: Average errors for FCN and CNN using 81-sensor data.

Method	**Average Error**
FCN	0.72 mm
CNN	0.79 mm

In addition, we tested various configurations of SiPMs. Our original dataset was created from the signal amplitudes from 81 sensors. However, as described in Section 1.2, we removed the signals amplitudes from the outer sensors to assess the impact of the number of SiPMs on the performance. With the possibility of complexity in cosmic-ray tomography systems, utilizing fewer materials to

achieve comparable performance would lower costs and simplify the implementation.

In Table 2, the performance of FCN with varying sensor configurations is presented. As per the table, when comparing the average errors of 81, 36, and 25 sensors, there is a minimal difference with values of 0.72 mm, 0.73 mm, and 0.75 mm respectively. The significant increase in the average error occurs only when the number of sensors is reduced to 16. Hence, it is not necessarily required to have a high number of sensors. Beyond 25 sensors, the increase in sensor count does not significantly improve performance. Therefore, a balance must be struck between performance and cost to find an optimal solution. For our purposes, we found 25 sensors to be a good compromise as we did not want to have more than three times the number of sensors for only a small improvement of 0.03 mm in performance.

Table 2: Results for different SiPMs numbers using FCN.

Number of Sensors	**Average Error**
81 (9×9)	0.72 mm
36 (6×6)	0.73 mm
25 (5×5)	0.75 mm
16 (4×4)	0.84 mm
9 (3×3)	1.08 mm

In Table 2, the trend is clear, more sensors equate to better performance. This is due to the fact that a larger number of sensors provides a more accurate estimation of the illumination caused by the hit, leading to a more precise prediction of the hit's position. However, after a certain point, adding additional sensors has limited effectiveness. This is due to the behavior of the light produced by a hit, which becomes weaker as one moves away from the hit point. The far-positioned sensors, therefore, are exposed to weaker illumination and carry little information. As an illustration, the signal amplitudes for a hit at coordinates $(0,0)$ are shown in the Table 1.4. The sensors close to the hit position have the highest signal amplitudes, while the outer sensors' amplitudes gradually become insignificant. The readings show asymmetry due to the randomness in the simulation since the table shows the data for a single hit. It is worth noting that the average value of signal amplitudes remains the same for different hit locations, even though the pattern changes.

In order to gain a more comprehensive understanding of the performance, we delved deeper into the results for the x and y coordinates individually. Table 4 presents the quantity of predicted coordinates and the corresponding error intervals. For instance, 97.35% of x predictions are within the error interval of 0–1.2 mm. The table illustrates that the models trained on x and y coordinates sepa-

0	2	3	4	3	3	7	3	5
1	4	7	6	7	4	2	3	2
5	3	5	7	2	14	10	5	1
2	3	10	6	12	22	8	6	4
3	6	10	36	250	52	14	5	6
7	7	10	46	132	56	13	12	4
4	7	10	20	44	25	10	7	3
5	4	9	7	3	13	9	7	8
3	3	5	1	9	9	9	2	5

Table 3: Sensor readings for a single hit on (0, 0) location taken randomly from the dataset.

rately produce comparable results, which is expected given the symmetrical sensor placement in our setup. Furthermore, only a small fraction (0.01% and 0.03%) of the predictions have an error exceeding 6 mm, indicating that outliers are minimal.

Table 4: Error intervals and corresponding percentages for x and y predictions.

Error Interval	***x* Predictions**	***y* Predictions**
0–1.2 mm	97.35 %	97.28 %
1.2–2 mm	2.0 %	2.11 %
2–4 mm	0.59 %	0.53 %
4–6 mm	0.05 %	0.05 %
6 mm and higher	0.01 %	0.03 %

Lastly, to gauge the effectiveness of our method in comparison to a widely used conventional method, we compared our results to those obtained through the use of the CoG technique:

$$x = \sum_{i=1}^{N} x_i \cdot A_i (\sum_{i=1}^{N} A_i)^{-1} \tag{1.4}$$

$$y = \sum_{i=1}^{N} y_i \cdot A_i (\sum_{i=1}^{N} A_i)^{-1} \tag{1.5}$$

where x_i and y_i represent the coordinates on the plate, A_i refers to the corresponding amplitude, and x and y indicate the positions of the CoG.

As demonstrated in Table 5, our deep-learning-based methods produce significantly better results compared to the conventional method of CoG. The advantage of using analytical methods like CoG is that they do not require a large amount of data for learning, unlike deep-learning models. However, in our case, we have a sufficient dataset, making a significant improvement in performance a desirable outcome.

Table 5: Comparison of FCN, CNN, and CoG for 81-sensor data.

Method	**Average Error**
FCN	0.72 mm
CNN	0.79 mm
CoG	1.41 mm

The results of the comparison between CoG and FCN for hit location estimations are presented in Figure 6. This figure displays twenty-five hit locations that are selected with a 5 mm step. The dark blue area indicates where there are no estimations and the transition from light blue to red indicates an increasing number of hits estimated in the respective coordinates. The area in the figure represents the red dotted area shown in Figure 2. The average errors for these locations, along with the estimations, are also presented. Due to a technical issue in the simulation, which resulted with some data at the $(0,0)$ location having an unusual distribution of signal amplitudes, the results at the $(0,1)$ location are used instead of the $(0,0)$ location as it provides better clarity. Note that not all the hit locations used in the experiment are shown in the figure, but the results follow the same trend for the other locations as well.

It can be seen in Figure 6a,c, the hits at the sensor locations $(0,1)$, $(0,20)$, $(20,0)$, and $(20,20)$ are estimated more accurately for CoG, while the performance decreases as the hit locations move away from the sensors. This is because CoG weighs the closer sensors the most due to their high signal amplitudes, causing a certain pattern of estimations with a bias towards the sensors and lower performance around them. However, the estimations by FCN follow a different pattern and obtain smaller errors than CoG for all hit locations except $(0,1)$, $(0,20)$, $(20,0)$, and $(20,20)$, where the average errors are similar between the methods. Compared to CoG, the estimations by FCN are more concentrated around the actual hit locations due to the smaller errors.

It is important to discuss FCN's generalization ability. In Figure 6b, the edge estimations are more defined compared to those in the center, which we attribute to the limited nature of the data used for training the model. The model only learned about hits restricted to a specific area and therefore its estimations are limited to that same area. This issue can be resolved by using a more comprehensive dataset that encompasses hits from the entire target area. To prove this, we took the current data and multiplied it symmetrically along the x-axis and y-axis to expand the area, and then trained another model using this extended dataset. The results, shown in Figure 7, show that the average error remains unchanged at 0.72 mm, which is expected due to the symmetry of the data. In the figure, previously restricted locations, such as those along the axes, are now estimated without restriction. This experiment demonstrates that if the data is generated in a way that covers the entire plate, the method can achieve an optimal outcome.

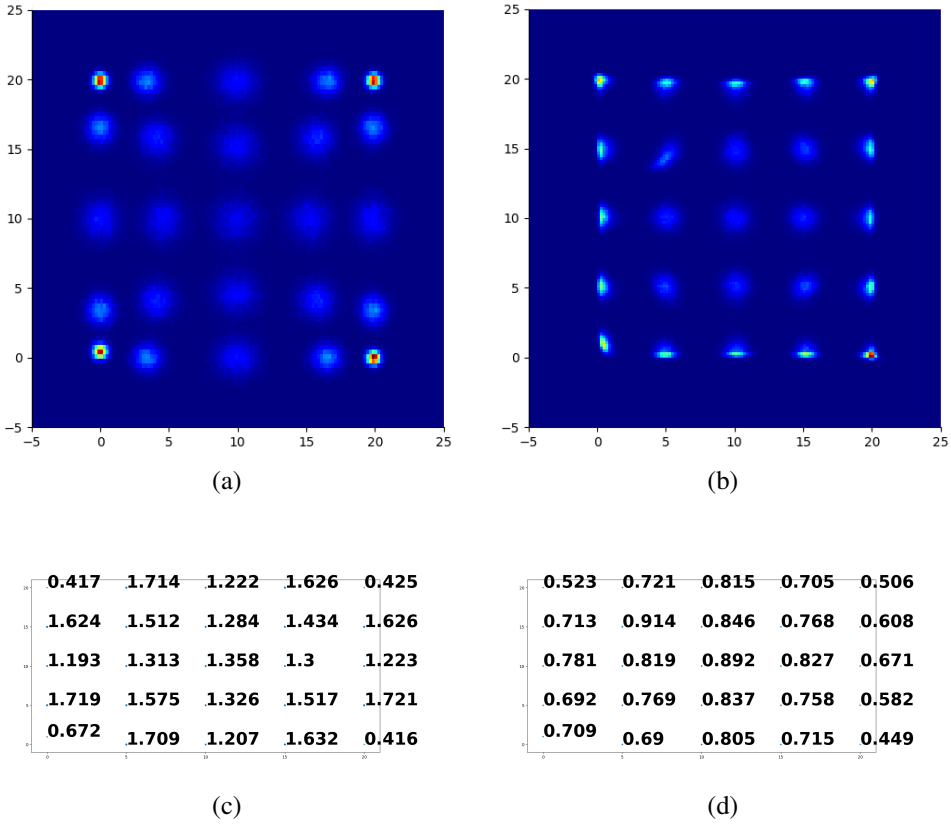


Figure 6: Comparison of CoG and FCN predictions for some locations using 50000 test samples (25×2000) (**a**) CoG predictions for the hits on the chosen locations, in mm. (**b**) FCN predictions for the hits on the chosen locations, in mm. (**c**) Average errors for CoG on the chosen locations with respect to Figure 6a. (**d**) Average errors for FCN on the chosen locations with respect to Figure 6b.

1.5. Conclusion

The quality of the images produced by cosmic-ray tomography systems is largely dependent on the accuracy of the particle hit position estimation on the detector plates. An accurate hit position estimation leads to higher quality images, whereas poor estimation results in low-quality or even meaningless images. Hence, successful hit position estimation is crucial for these systems. Historically, analytical methods such as the CoG have been used for hit position estimation. Although these methods do not require large datasets, their performance is relatively low as they do not utilize all the information available in a detector layer. As a result, researchers have explored alternative methods, such as deep learning techniques, to improve hit position estimation. While deep learning-based methods have shown promising results, there is still room for improvement and further research in this area.

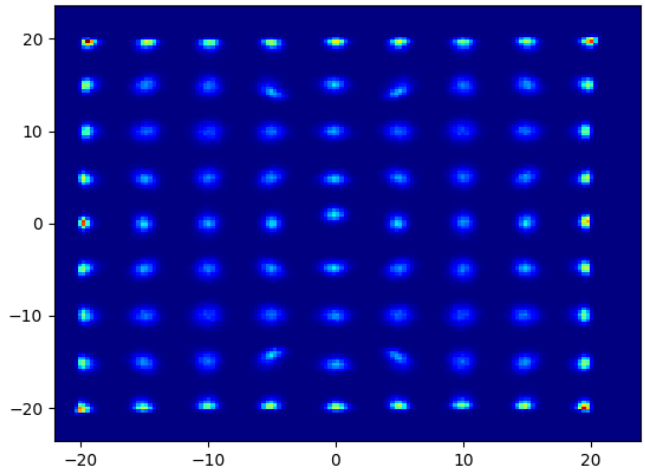


Figure 7: FCN predictions for the hits on the larger area that covers the central region of the detector plate, in mm.

In this chapter, we evaluated the effectiveness of two deep learning networks, FCN and CNN, for the task of muon hit position estimation. Our dataset was generated using GEANT4 simulation and our experiments resulted in an average error of 0.72 mm when using the FCN network with 81 sensors. This represents a significant improvement compared to the traditional analytical method of CoG. We also explored the impact of different sensor configurations and found that 25 sensors produced results comparable to those of 81 sensors. In future work, we aim to analyze the effect of different intervals of simulated hit positions on performance improvement. Also, the dataset will be expanded to include a larger plate area and varied hit angles. Our ultimate goal is to build a practical experimental setup to validate our results with a real device data.

2. ALTERNATIVE DATA SOURCE: DEEP CONVOLUTIONAL NEURAL NETWORKS FOR DETECTION OF ABNORMALITIES ON X-RAYS DATA

2.1. Introduction

Tomography systems have various applications in medical imaging, including the identification and management of illnesses [45, 46]. The employment of tomography methods in the area of COVID-19 detection is one such application [47]. A popular medical imaging method that offers useful information on the interior organs of the body is the X-ray images. They are frequently used to diagnose a wide range of illnesses, including lung ailments like pneumonia, malignancies, and bone fractures [48]. Thanks to developments in deep learning techniques AI-based automated diagnosis techniques such as DCNN have been utilized to classify X-ray images [49]. DCNN-based methods are able to recognize small variations between tissues and structures that might not be obvious to the human eye by automatically learning complicated visual features and patterns. Thanks to this, anomalies or diseases can be located and identified in X-ray images [50–52].

In this chapter, we aim to evaluate the performance of our DCNN-based method on tomography produced images. We selected COVID-19 as an example because the necessity to use assistant tools such as AI in combination with expert opinions has become more necessary to help clinicians in their work due to the recent pandemic [53, 54]. As hospitals experience a shortage of radiologists, AI-based diagnostic models could provide prompt assistance for helping the patients. In order to detect lung disorders from X-ray images, automated diagnosis approaches using DCNN have demonstrated promising results [55]. Hemdan et al. proposed seven CNN models for X-ray based COVID-19 detection, including enhanced versions of VGG19 and Google MobileNet [56]. Wang et al. used a tailored approach to identify COVID-19 images from normal and viral pneumonia patients with an accuracy of 92.4% [57]. And Ioannis et al. utilized 224 COVID-19 images and achieved a class accuracy of 93.4% [50]. An optimized CNN named Opconet was demonstrated in [51] using 2800 images with an accuracy score of 92.8%. Apostolopoulous et al. created a MobileNet CNN model using extracted features [52]. Other methods, such as InceptionV3, ResNet50, and Inception-ResNetV2, also have been used for classification [58–61]. Transfer learning-based methods were used to classify the presence or absence of COVID-19 in chest X-ray images using ResNet18, ResNet50, SqueezeNet, and DenseNet121 [62]. In [63], the key hyperparameters of the CNN were fine-tuned using MLP and Grey Wolf Optimizer (GWO) and MLP and the Whale optimization + BAT method.

The state-of-the-art approaches that employ DCNN are limited by their use of a dataset that only consists of 6,432 images at most [63]. This small dataset leads to a failure to capture real-life cases, and data augmentation techniques such

as rotation and resizing the pictures are not sufficient to cover the wide range of possible cases for COVID-19 instances, viral pneumonia, and normal chest X-ray scans. Consequently, the CNN models generated are unable to accurately differentiate between these diseases. The current limitations of the existing automated diagnostic methods in accurately recognizing COVID-19 cases can lead to confusion among medical professionals and administrators. In response to this, the current study addresses these limitations by developing an DCNN-based screening method for COVID-19 patients using chest X-rays. The presented method is trained on a large dataset of 103,468 images that covers five different classes, including COPD signs, COVID, normal, other, and pneumonia.

The structure of the remaining part of this paper is outlined as follows: Section 2.2 provides a detailed explanation of the dataset, Section 2.3 presents the proposed method, Section 2.4 showcases the experimental results, and finally, Section 2.5 concludes the paper.

2.2. Dataset

In this study, we have gathered data from four separate datasets, namely, the PADCHEST dataset [64], the BIMCV-COVID19+ dataset [65], the COVID-19 Radiography Database ([66] and [67]), and the Chest X-Ray Images (Pneumonia) [68]. By merging all these datasets, we have obtained 297,541 frontal chest X-ray images from 86,876 individuals. We have not performed any image processing during the data collection process. We used the images with the proper labeling according to the labels listed in Table 6 which added up to a dataset of 103,476 images. Therefore, we could only use a subset of the whole data. The dataset includes an average of 3-4 images per subject due to follow-up scans. To split the patients into training, validation, and test groups, patient-wise splits have been considered. Table 6 shows the number of images that each anomaly was found in. It should be noted that multiple anomalies may be present in a single image. In addition, there are 178,319 images that do not exhibit any of the anomalies listed in Table 6.

There was a significant imbalance between the different classes in the data, with some classes having significantly more samples than others. For example most of the classes have less than 1,000 samples each while "normal" and "COPD signs" have 62,115 and 23,280 samples respectively. To address this issue, similar classes were grouped together and re-labeled. Additionally, an insignificant number of images were excluded from the dataset in the experiments due to their broken file formats.

We have defined the classes for our network by combining the available abnormalities from each dataset used. Our initial experiments were conducted with a 5-class split of the dataset as shown in Table 7. The classes "normal", "COPD signs", "pneumonia", and "covid" were kept as separate classes due to the abundance of samples, while the smaller classes were grouped under the label "others".

Table 6: Distribution of anomalies in the dataset

Class	Number of samples
normal	62115
pulmonary fibrosis	760
heart insufficiency	1722
COPD signs	23280
pneumonia	7747
tuberculosis sequelae	399
emphysema	734
pulmonary artery hypertension	8
tuberculosis	152
atypical pneumonia	234
bone metastasis	150
lung metastasis	326
pulmonary edema	458
asbestosis signs	69
pulmonary hypertension	148
post radiotherapy changes	138
respiratory distress	35
lymphangitis carcinomatosa	21
lepidic adenocarcinoma	11
covid	3616
viral pneumonia	1345

In our second experiment, we divided the dataset into two categories, as shown in Table 7. The "normal" class was kept intact, while all other classes were grouped together under the label "abnormal." The "normal" label indicates a healthy lung, while "abnormal" represents an unhealthy lung.

2.3. Proposed Method

2.3.1. Network

In our approach, we extract the key features of the images that differentiate the classes using a feature extractor network. Afterwards, we use a classifier to obtain the results. In the case of X-Ray scans of lungs, the texture can provide the distinguishing features. For instance, an X-Ray scan of a lung with pneumonia displays a different texture compared to a healthy lung (as shown in Table 9). Our goal is to detect such features and identify patterns for each class.

State-of-the-art research suggests that using CNN-based deep learning methods is one of the most effective ways to extract texture features from images [6, 40, 41, 63]. These methods are successful due to their convolutional structure,

Table 7: Dataset with 5-class split

Class	Number of samples
normal	62108
COPD signs	23277
pneumonia	9092
covid	3616
others	5239

Table 8: Dataset with 2-class split

Class	Number of samples
normal	62108
abnormal	40132

which processes each pixel and its relation to neighboring pixels. Therefore, we chose to use a CNN-based network as our feature extractor.

In our approach, we use a InceptionV3 network as our backbone for feature extraction. To determine the best backbone network, we conducted experiments with different options and found that InceptionV3 performed the best in our case. The details of these experiments can be found in the results section. InceptionV3 is a CNN-based architecture consisting of symmetric and asymmetric blocks, as depicted in Figure 8. The network has a deep structure with convolution layers forming the base and fully connected layers linking to the output. Additionally, average pooling, max pooling, batch normalization, and dropouts are employed in the network to enhance its performance.

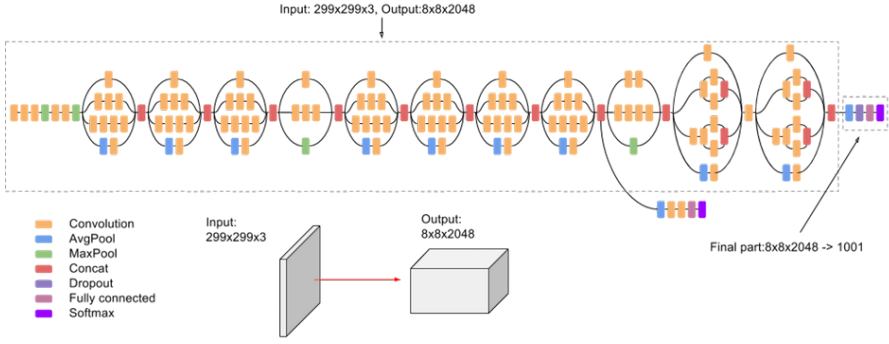
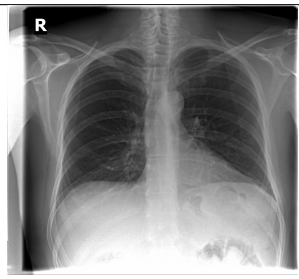
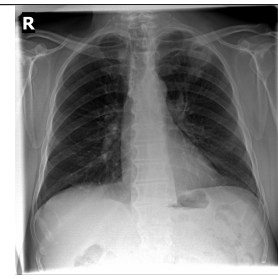


Figure 8: Architecture of InceptionV3 [69]

.

We did not modify InceptionV3 in our experiments. We used it as a feature extractor, and combined it with a classifier to obtain the final label. The features extracted from the image are first normalized through a normalization layer, which is batch normalization. This subtracts the mean and divides by the standard de-

Table 9: Example images from the dataset

Normal	Covid	Pneumonia
		

viation for each batch [70]. Then, two fully connected layers with sizes of 64 and 32 are added to model the relationship between the extracted features and the final classes. The activation function used in these layers is Rectified Linear Unit (ReLU). To prevent overfitting, dropout layers with a rate of 0.5 are included after each fully connected layer [71]. Finally, an output layer is included to produce the final prediction for each class.

2.3.2. Training

We trained the model with a batch size of 128 and utilized 20 epochs, as it provided the best performance in our scenario. It can be seen in the Figure 9, utilizing more epochs would result in overfitting. To prevent biased data splitting, we employed 5-fold cross-validation. Larger fold numbers were not utilized as they significantly increased training time. The dataset was split into training, validation, and test sets at a ratio of 0.60, 0.15 and 0.25, respectively. The training and validation sets are used during the model training, while the testing set was used solely for testing after training was completed. The experiments were conducted on a PC running Ubuntu 20.04, using an Intel® Core™ i7-5820K central processing unit, NVIDIA TITAN Xp graphics processing unit, and 96 gigabytes of memory. Respectively, VGG16, InceptionV3, ResNet50, and NasNetMobile have 14 751 813, 21 944 357, 23 729 285, and 4 343 833 parameters, and takes around 20, 25, 25, and 10 minutes to train one epoch in the specified setup.

For implementation of the models and training, we used Keras library with TensorFlow backend. Backbone networks in this work chapter been implemented using the applications feature in Keras. The choice process of hyperparameters corresponds to the procedure described in Chapter 1.3.3 of this thesis.

2.4. Results

Due to the similarities between X-ray samples in this study, the dataset has a low inter-class variance. Hence, it is crucial to extract unique features for improved performance. Our experiments focused on this concept to enhance the perfor-

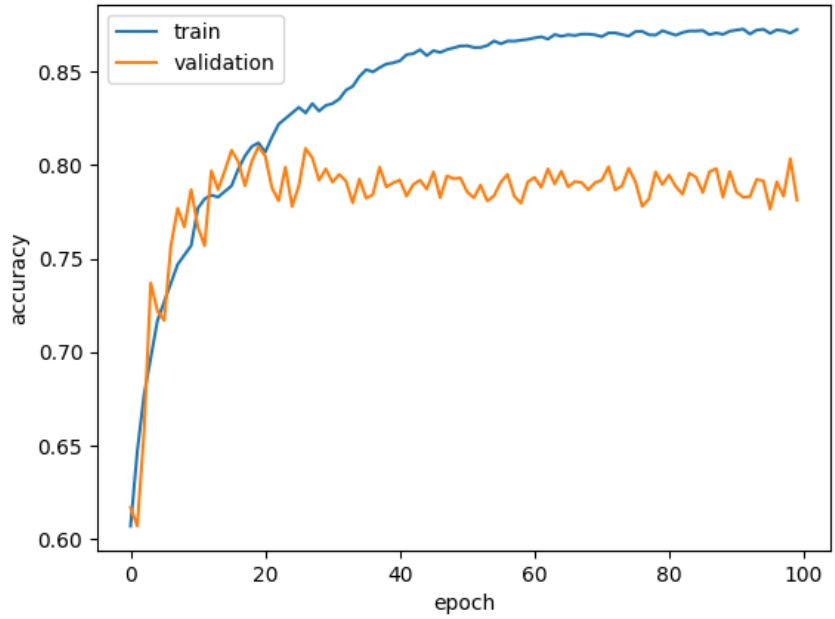


Figure 9: One of the training graphs where overfitting can be seen.

mance. We conducted experiments with different feature extraction backbones to assess their impact on the performance. As described in Section 2.3, we selected commonly used state-of-the-art CNN-based networks including VGG16 [72], InceptionV3 [69], ResNet50 [73], and NasNetMobile [74]. We conducted the same experiments for both the 5-class and 2-class dataset labeling groups.

We used accuracy as the metric to measure the model's overall performance. The accuracy can be expressed as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{2.1}$$

where the *Number of Correct Predictions* represents the count of instances that were correctly classified, and the *Total Number of Predictions* is the total count of instances in the dataset.

In addition to this, we calculated the class accuracies according to the following formula:

$$\text{Class Accuracy}_c = \frac{\text{Number of Correct Predictions for Class } c}{\text{Total Number of Instances of Class } c} \tag{2.2}$$

where the *Number of Correct Predictions for Class c* represents the count of instances that belong to class c and were correctly classified as class c, and the *Total Number of Instances of Class c* is the total count of instances in the dataset that actually belong to class c.

Table 10: Results of 2-class and 5-class splits for each backbone

Backbone	2-class	5-class
VGG16	0.84	0.79
InceptionV3	0.85	0.81
ResNet50	0.83	0.75
NasNetMobile	0.73	0.68

Table 10 shows the performance of our method when different backbones are used. We evaluated the performance on the test data for both 2-class and 5-class splits. As seen from the table, the performances of all the backbones are better for the 2-class split compared to the 5-class split. This result was expected since the 2-class split in this task is between normal and abnormal (diseased) X-ray images, which have a higher in-class variance than the variance in the 5-class dataset. The 5-class split requires the network to differentiate among four pathological classes. Additionally, the low accuracy of the "others" class in the 5-class split also contributes to the decrease in overall accuracy. As per the accuracy values in the table, VGG16, InceptionV3, and ResNet50 perform similarly for the 2-class split. However, their performance varies for the 5-class split, with InceptionV3 leading with a test accuracy of 81.03%.

Table 11: Accuracies of each class for InceptionV3 backbone.

2-class accuracies	normal 0.78
	abnormal 0.89
5-class accuracies	COPD signs 0.63
	covid 0.97
	normal 0.93
	pneumonia 0.65
	others 0.34

Table 11 presents the class accuracies of our method using the InceptionV3 backbone for both the 2-class and 5-class splits. The main differences in class accuracies are attributed to the low variance and differences in sample counts between classes. Notably, the accuracy for the "others" class is low as a result of the combination of low sample count and a challenging sample set to characterize. This is a result of using many low numbered diseases to create the "others" class. In consideration of the imbalanced dataset, we also calculated the Matthews Correlation Coefficient (MCC) [75]. Our model achieved MCC values of 0.68 and 0.65 for the 2-class and 5-class splits, respectively.

Our best results were achieved by using InceptionV3 as the backbone without any filtering. The test accuracy was 0.85 for the 2-class split and 0.81 for the 5-class split. We also achieved a test accuracy of 0.97 for predicting COVID-diagnosed X-ray images in the 5-class split. The accuracy for predicting "normal"

Table 12: Comparison with the state-of-the-art

Reference	Number of images	Database Used	Backbone Used	Covid-19 acc.
Proposed	103468	PADCHEST BIMCV-COVID19+ COVID-19 Radiography Chest X-ray	InceptionV3	0.97
[51]	2800	Chest X-ray	OptCoNet	0.98
[76]	1125	Chest X-ray	DarkCovidNet	0.98
[77]	196	Chest X-ray JSRT	VGG16	0.93
[50]	1428	Chest X-ray Covid-19 X-ray Pneumonia X-ray	MobileNetV2	0.98
[78]	502	Chest X-ray CoronaHack NLM JSRT	DenseNet103	0.92
[79]	2905	COVID-19 Radiography	A novel CNN	0.89
[63]	6432	Chest X-ray	ResNet-50	1
[63]	2905	COVID-19 Radiography	ResNet-50	1

or healthy images was 0.93 for the 5-class split and 0.78 for the 2-class split.

The comparison of our results with state-of-the-art methods is shown in Table 12. Our method performs well compared with the other methods with 0.97 accuracy in detecting COVID images. Although the accuracy is on par with the state-of-the-art, it is a remarkable achievement considering the difference in the size of the datasets. It is noteworthy that the other methods were trained with small datasets of up to 6,432 images, whereas our method was trained on a larger dataset of 103,468 images, including cases from 20 different diseases. This diversity and size of our dataset increases the challenge, but it also provides better generalization capabilities compared to the other methods. Generalization is a major issue in COVID detection, but our method has proven to be effective. Our study focuses primarily on COVID accuracy, and our method has produced exceptional results in this metric, making it a valuable contribution to the field of COVID detection. It is important to note that the results in Table 12 are produced on their respective datasets and their performance may vary in the real-world.

Table 13: Training accuracy for 2-class with InceptionV3 backbone.

	precision	recall	f1-score	support
Normal	0.82	0.78	0.80	10029
Abnormal	0.86	0.89	0.87	15531
accuracy			0.85	25560
macro avg	0.84	0.83	0.84	25560
weighted avg	0.84	0.85	0.85	25560

Table 14: Training accuracy for 5-class with InceptionV3 backbone.

	precision	recall	f1-score	support
COPD Signs	0.68	0.63	0.66	5830
Covid	0.94	0.97	0.96	843
Normal	0.84	0.93	0.89	15629
Others	0.58	0.34	0.43	1308
Pneumonia	0.92	0.64	0.76	2223
accuracy			0.81	25833
macro avg	0.78	0.70	0.74	25833
weighted avg	0.80	0.81	0.80	25833

Additionally, we present the training accuracy results of our method in Tables 13 and 14 for each class combination. The results indicate that there is only a minimal discrepancy between the training and test accuracies, suggesting that overfitting or underfitting is not an issue in our training. However, some classes such as "others" in the 5-class split show low accuracy, which is likely due to the heterogeneous nature of the samples in this class, making it more challenging to characterize than the other classes.

2.5. Conclusion

In this chapter, we used X-ray scans as an example to demonstrate the effectiveness of our proposed method on the target disease detection from the tomography images. Previous research has suggested using CNN and similar structures for automatic differentiation between normal and diseased chest X-rays. However, the previous research has suffered from limitations such as a relatively small dataset, class bias, and low generalization.

To tackle the limitations of the prior research, we carried out a DCNN analysis of pathological chest X-ray classification using a large image dataset of 103,468 images. Our experiments involved different class splits and methods, leading to an accuracy of 0.85 in classifying between normal and diseased images. Additionally, we achieved 0.81 accuracy in classifying between five classes, including

COPD signs, COVID, Pneumonia, normal, and others. Our method achieved 0.97 accuracy in detecting COVID which is a remarkable performance considering the huge size of the dataset compared to the state-of-the-art.

Our method has a drawback that the "others" category in the 5-class split encompasses images from a multitude of illnesses, making it difficult to accurately categorize. This results in a decrease in the performance. A potential solution would be to enhance the data in this category through a further study or to enhance the network's ability to learn low-performing categories better.

3. CLASSIFICATION OF TIME SERIES DATA USING LSTM AND FEATURE EXTRACTION BACKBONE

3.1. Introduction

Action recognition is a prominent and complicated subject in computer vision which has the goal to automatically recognize and categorize human actions or activities based on visual input such as images or videos [80, 81]. Image classification, as a fundamental task in computer vision, plays a critical role in action identification by finding the target class in an image or video. Action recognition has enormous potential to improve numerous fields of human-centric technologies due to its vast variety of applications, which include surveillance, video analysis, robotics, and human-computer interaction [82–85].

The classification of a data consisting of a series of images is a key approach in action recognition [81, 86, 87]. This process not only enables the extraction of meaningful spatio-temporal information but also provides valuable insights into the actions performed within the data. By effectively analyzing the sequential nature of images, action recognition algorithms can discern complex human activities, ranging from simple gestures to complex interactions, and accurately assign them to predefined action classes.

This chapter aims to demonstrate the effectiveness of spatio-temporal features that can be extracted from the time-series data and classification models through experimentation with an RGB-based time-series dataset. Specifically, we utilized a table tennis dataset called TTStroke-21 [88] obtained using the standard cameras. These data involve sequences of images, with each image containing information about the state of the performer being observed at a particular moment in time. The images are spatially correlated, meaning that nearby pixels in the image are likely to have similar values. Additionally, the videos are complex and high-dimensional, with large amounts of data being generated over time. As a result, analyzing this data requires sophisticated computational methods such as deep learning techniques.

LSTM is one of the most common deep-learning techniques that is used for temporal recognition. It has been utilized in many studies that includes RGB-based time-series data. In [89], an LSTM model was proposed to recognize actions in videos by processing the spatio-temporal features extracted from the frames. Another study [90] utilized an LSTM-based framework for event detection in surveillance videos by combining the features from both RGB and optical flow data. In [91], an LSTM model was used for gesture recognition by processing the hand and body joints extracted from the video frames. In [92], the authors proposed an LSTM-based network which is combined by a VGG16 [72] backbone to extract the spatio-temporal features from RGB images. State-of-the art methods which conducted their experiments on TTStroke-21 dataset utilizes similar techniques. A cascaded method that utilizes optical flow and RGB data is em-

ployed by [93]. Their Siamese Spatio-Temporal Convolutional Neural Network (SSTCNN) method achieved 81.3% accuracy. In [88], the same authors proposed Twin Spatio-Temporal Convolutional Neural Network (TSTCNN), which is an upgraded version of their previous work. They improved the accuracy to 91.4%. A different approach is taken by [94] by creating a method that uses 2D pose estimation data. They conducted their experiments on a custom dataset which includes more videos but less stroke types and obtained 98.72% accuracy.

In this chapter, we present a spatio-temporal recognition using LSTM-based method. Our architecture includes a backbone model that extracts the momentary features. These features are then passed into an LSTM network to perform the temporal recognition. We experimented with four different backbones namely RGB, Optical Flow, Pose, Region of Interest (ROI). Each backbone aims to extract different features and aims to understand their effect on the performance. RGB uses the initial image as the input to extract the features directly from the image. Optical flow helps to examine the velocity distribution in the image. Pose obtains keypoints in the image and extracts features based on these keypoints. ROI crops a certain part of the image and focuses only on this part. We decided to use these particular features as the above-mentioned literature proved them to be useful. However, we used different extractor networks which are explained in chapter 3.2.1. We utilized our architecture in a three stage pipeline with a late fusion to properly fit the nature of the table tennis data we have. We also investigated the effects of the data augmentation on the performance.

The rest of this chapter is organized as follows: Section 3.2 presents the proposed method. Section 3.3 shows the obtained experimental results, and Section 3.4 concludes the chapter.

3.2. Methodology

We propose a spatio-temporal recognition method based on LSTM architecture, where a backbone model extracts momentary features that are then passed into LSTM for temporal recognition. We deployed a three-stage pipeline to utilize our method in the challenging task that comes with TTStroke-21 dataset. This dataset includes 21 different stroke classes that covers different table tennis stroke techniques. Classifying table tennis strokes is a challenging task due to the high number of classes and low variance between them. In our pipeline, we address these challenges by combining the final classes into small groups. This approach reduces the number of classes to be classified at once from 20 to smaller groups that are classified individually. To further improve the classification accuracy, we employ a multi-stage approach with four different classifiers, where predictions are made in three stages and then fused using ensemble methods such as [95, 96] to obtain the final class prediction.

3.2.1. Methods

In this chapter, we explore four distinct backbone techniques —RGB-based, optical flow-based, pose estimation-based, and ROI-based methods— that can be incorporated into our pipeline to extract optimal features for classification tasks. Although they share the same principle of selecting the most appropriate features for a given task, each method prioritizes different aspects to achieve the best possible fit. Furthermore, we integrate an LSTM-based network in each of these methods to capture spatio-temporal features. Details of the backbones and the LSTM-based network is described in this section.

RGB images. Our aim with the RGB images-based method is to capture spatio-temporal features directly from the RGB images. To achieve this, we pass frames from the videos to the LSTM-based model as a sequence. By doing so, we expect the model to extract the necessary information from both the frames and the sequence of frames.

Optical flow. The optical flow method captures the distribution of velocities of movements based on brightness patterns in a frame, providing essential information about the spatial arrangement change resulting from the objects' motion relative to the viewer. To estimate the videos' motion, we used FlowNet 2.0 [97] in our work. FlowNet 2.0 incorporates a DCNN that extracts detailed motion patterns from image pairs, making it robust against complex scenes, occlusions, and large displacements. In our case, it helps to eliminate background factors and noise to focus only on the player's movement while executing a stroke. The preprocessed optical flow data is fed into our proposed model. We used the Pytorch implementation publicly available in Github [98].

Pose estimation. Pose estimation involves determining the spatial positions and orientations of objects or body parts within an image. In the context of human pose estimation, it specifically refers to the process of identifying and tracking the locations of key body joints and parts, such as the head, shoulders, elbows, wrists, hips, knees, and ankles. This information is crucial for understanding and interpreting human movement, behavior, and posture. Pose estimation methods enable us to obtain spatial and temporal information of body parts in the frames by defining their positions as keypoints. In our research, we used pose estimation to extract detailed movement information from table tennis stroke videos. We applied AlphaPose, a method for regional multi-person pose estimation which extracts 17 keypoints of different body parts for every frame [99]. The extracted keypoints are fed into our LSTM-based model to capture spatio-temporal features. We used the Pytorch impementation of AlphaPose that is available publicly available [100].

Region of interest. In this feature extraction approach, a specific area is selected and cropped from the RGB images to focus only on that area. For the given dataset, we selected to focus on the racket's motion during a stroke. Object detection is performed on each RGB image using You Only Look Once (YOLO)v5 to crop the racket. YOLOv5 a single-stage object detector known for its excel-

lent performance and faster inference time compared to two-stage object detectors [101]. It operates by dividing an input image into a grid and simultaneously predicting bounding boxes and class probabilities for each grid cell. The network architecture employs a combination of convolutional layers and a feature pyramid network to capture features at different scales, enabling it to detect objects of various sizes and aspect ratios with high precision. We used YOLOv5 implementation in [102] for our work. The detected racket's coordinates from the frames are recorded, and the initial frames' background is removed, leaving only the table tennis racket images. Finally, these images are passed on to our LSTM-based model for stroke recognition.

Architecture of the LSTM based model. The stateful structure of LSTM makes it a commonly used method to process sequence-type inputs and extract spatio-temporal features. LSTM contains memory cells with hidden states for each time step in the input data. The output of each cell is computed based on the previous cell's hidden state and the current cell's input data. This allows the previous state to provide information about the past, while the input provides information about the current data. The output vector of the cells for time step t is computed using the following equation[103, 104]:

$$h_t = o_t * tanh(c_t) \tag{3.1}$$

where,

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{3.2}$$
$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3.3}$$
$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{3.4}$$
$$\tilde{c_t} = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{3.5}$$
$$c_t = i_t * \tilde{c_t} + f_t * c_{t-1} \tag{3.6}$$

o_t, f_t, i_t, and $\tilde{C}_t$, denotes activation vectors for output gate, forget gate, input gate, and cell input respectively, c_t stands for cell state, and, x_t stands for the input of LSTM .

The proposed model's architecture is presented in Fig. 10. The input data is initially processed by a normalization layer to regularize the data, which subtracts the mean and divides by the standard deviation [70]. Following this, there are two LSTM layers that capture spatio-temporal features. The first LSTM layer has 128 units, while the second has 32 units. A fully connected layer with a size of 64 is then utilized to model the relationship between the spatio-temporal features and the output. To avoid overfitting, a dropout layer with a rate of 0.2 is inserted after these three layers [71]. Lastly, a Softmax layer is applied to the end of the model for classification purposes.

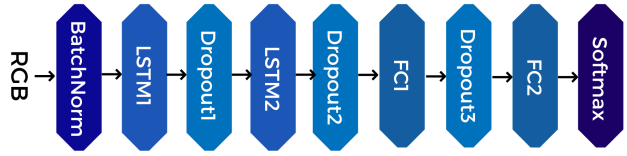


Figure 10: The proposed model architecture

3.2.2. Training

The Glorot initialization method [105] has been utilized to initialize the fully connected layers. The loss function used in the proposed model is categorical cross-entropy [106]. Additionally, RMSprop optimizer has been employed with a learning rate of 0.0001 [106]. During the training process, a batch size of 8 has been employed. For the RGB features based networks, 30 epochs were used, while 100 epochs were employed for other networks as they were sufficient for convergence. To prevent biased data split, 10-fold cross-validation has been applied. For each set of labels in Figure 11, a different model (a total of 5 models) is trained with the same training setup but using the different subsets of training data corresponding to the different sets of labels. Furthermore, the provided dataset has been split into train, validation, and test sets in the ratio of 0.7, 0.2, and 0.1, respectively. The training and validation sets were used for training the models, while the test set was used only for testing the trained model.

The experiments were conducted on a PC running Windows 10, using AMD Ryzen 5 2600 central processing unit, NVIDIA GeForce RTX 2060, and 32 gigabytes of memory. The LSTM-based network has 115,604 parameters, and takes approximately 4 minutes on average to train an epoch in the specified setup. For implementation and the training of the LSTM-based network, we used Keras library with the TensorFlow backend. For the optical flow, pose estimation, and ROI backbones, we used the publicly available implementations which are presented in Chapter 3.2.1. The logic behind choosing hyperparameters remains the same as specified in Chapter 1.3.3 of this thesis.

3.2.3. Class split

Our pipeline considers a single stroke type as a combination of 3 label groups. The first group consists of "Serve", "Offensive", and "Defensive" labels, indicating the position. The second group contains hand orientation labels, "Forehand" and "Backhand". Lastly, the third group includes hit technique labels, such as "Topspin", "Sidespin", "Backspin", "Loop", "Push", "Block", "Backspin", "Flip", "Loop", and "Hit". However, we predict the third group label based on the first group label. If the first group label prediction is "Serve", the third group labels can only be "Topspin", "Sidespin", "Backspin", and "Loop". If the prediction is "Defensive", the labels for the third group are "Push", "Block", and "Backspin". Finally. If it is "Offensive", the third group labels are "Flip", "Loop", and "Hit".

3.2.4. Pipeline

We created a pipeline consisting of three stages, where each stage predicts one component of the overall label, as depicted in Figure 11. In the first stage, we predict the initial component using the first label group. Similarly, in the second and third stages, we predict the second and third components of the label using the second and third label groups, respectively. However, the prediction of the third component is made under the condition of the outcome of the first stage prediction. For instance, if the first stage prediction is "Serve", then we use the model trained for predicting "Topspin", "Sidespin", "Backspin", "Loop" to predict the third component. Finally, we combine the outcomes of each stage to generate the final prediction.

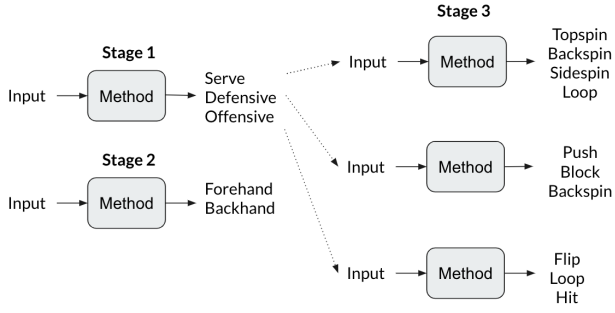


Figure 11: The proposed Spatio-temporal based stroke type assessment pipeline

The pipeline we designed for our approach operates independently in each stage, where data is not transferred between stages, and each stage receives the same input data, which are the frames of a stroke recording. Furthermore, each stage has its own unique method for classifying its assigned classes. This modular structure allows for the possibility of using different techniques for different groups of classes. In our study, we introduce four distinct methods that can be utilized as classifiers in these stages. We conduct experiments with these methods in Section 3.3 and report the most effective ones. The subsequent subsection details these methods.

3.3. Results and Discussion

3.3.1. Dataset

Data acquisition. In this chapter, we utilized a portion of the TTstroke-21 dataset, which was originally collected by Pierre et al. [88]. This dataset consists of videos of table tennis games played by 17 different athletes under real-world conditions. The recordings were captured using readily available GoPro cameras, without any markers or sensors being used. The camera was positioned high above the players and centered on them.

The table tennis game videos in the TTstroke-21 dataset used in this project were annotated by 18 experts and players at the Faculty of Sports at the University of Bordeaux [88]. The annotations were collected through a crowd sourcing method using a user-friendly web platform. The annotators labeled the stroke class and selected its starting and ending frames. To reduce the annotators' errors, annotations at the beginning and end of each video were discarded, resulting in a total of 2152 annotations.

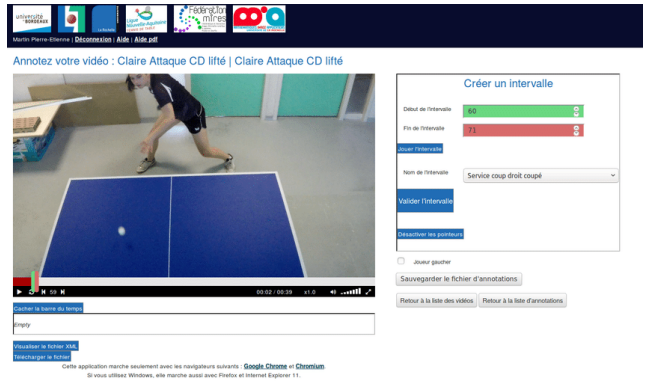


Figure 12: Annotation platform of collected samples [88].

The annotated dataset used in this study [88] includes videos with a resolution of 1920 x 1080 and a frame rate of 120 fps. Each stroke in the dataset contains at least 100 frames, and it is labeled with one of the 20 different stroke types. These 20 stroke types are combined into three groups: 6 offensive, 6 defensive, and 8 serve stroke types.

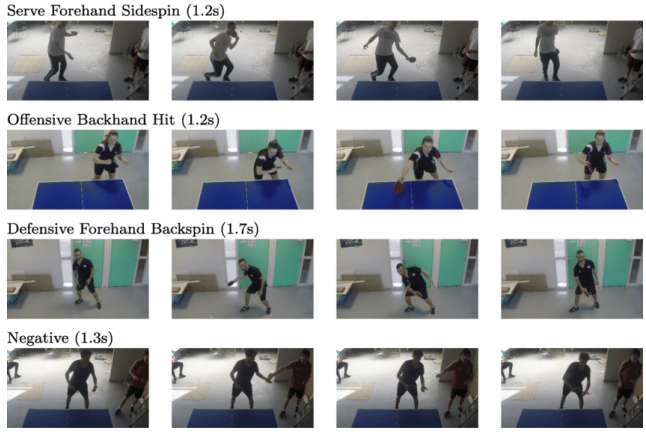


Figure 13: Samples from the dataset [88].

Data preprocessing. To avoid computational power and time-related issues caused by the high resolution and number of frames per stroke, frames are preprocessed before training the LSTM-based network. As each stroke in the dataset has a varying frame range, they are sampled at a fixed size to be used as input

to the model. In this study, we chose to sample each stroke using 21 frames as it was found to improve processing performance and provide a fixed input size. However, lower sample sizes were also tested, and it was observed that using only 7 frames per move significantly reduced the classification accuracy. The choice of 21 frames was based on the experiments.

Furthermore, in order to determine which 21 frames should be selected for each stroke, we utilized the K-Nearest Neighbor (kNN) method. This method is used to identify the most relevant frames for each stroke by calculating the centroid positions based on the nearest neighbors [107]. A centroid can be envisioned as a representative point in the data space that encapsulates the characteristics of a group of nearby points. In the context of our study, these points correspond to frames capturing different temporal instances of a stroke. The formula for calculating the centroid position for a set of k nearest neighbors is defined as:

$$x'_k = \frac{1}{k} \sum_{j=1}^{k} x_j \tag{3.7}$$

where j-th centroid is constructed by j nearest neighbors hence creating k centroids. And, $x'k$ denotes to k-nearest neighbors centroid of query points.

We compute 21 centroids for each move using the kNN method, and select the 21 frames that are closest to these centroids. To save memory, each frame is resized to (120×80), and flattened to ensure it fits into the model.

Additionally, data augmentation is experimented in this chapter. It is performed separately on each stroke in the training data. The process is applied to the whole image. Augmentation involves randomly rotating each stroke by up to 10 degrees, randomly translating it within a range of 0.1 in both horizontal and vertical directions, randomly scaling it within a range of 1 ± 0.1, and randomly flipping it horizontally with a 50% chance. Rotation, translation, scaling, and, flipping are selected because it fits the table tennis data and the values are selected because they are commonly used values in the literature [57, 88].

3.3.2. Results

We conducted experiments on the TTSTROKE-21 dataset using all the aforementioned methods. In order to comprehensively evaluate the performance, we report the results separately for each stage as well as the final results.

Our experiments on the TTSTROKE-21 dataset show that the model achieves the best performance in stage 1 when RGB images are used as input data, with a test accuracy of 97.4% (see Table 15). This indicates that the model is capable of extracting the necessary information directly from the RGB images without requiring a feature extraction method. Notably, the other methods perform less accurately, which could suggest that certain features critical to discriminating stage 1 labels are lost when using other methods.

Table 15: Accuracy for stage 1 labels (Serve, offensive defensive).

Method	Validation Accuracy	Test Accuracy
RGB	**98.9**%	**97.4%**
Optical flow	93.2%	90.7%
Pose	86.4%	88.2%
R.O.I	88.2%	84.0%

In the second stage, the model's performance is not as impressive as in the first stage. Nevertheless, it achieves a 92.6% test accuracy. Similar to stage 1, the highest accuracy is obtained when using RGB images directly. Notably, optical flow is not far behind in terms of test accuracy, with a score of 91.7

Table 16: Accuracy for stage 2 labels (Forehand, backhand).

Method	Validation Accuracy	Test Accuracy
RGB	**94.0**%	**92.6%**
Optical flow	90.1%	91.7%
Pose	91.4%	86.0%
R.O.I.	89.9%	89.8%

In the third stage, achieving high accuracy is particularly challenging due to the larger number of labels and lower variance between the classes. Nonetheless, our model utilizing optical flow data outperforms all other methods in this stage.

Table 17: Accuracy for stage 3 labels (Flip, hit, push, block, loop, topspin, backspin, sidespin).

Method	Validation Accuracy	Test Accuracy
RGB	83.8%	78.3%
Optical flow	**92.4%**	**87.2%**
Pose	72.3%	69.2%
R.O.I.	80.2%	72.8%

After obtaining predictions from the three stages, we combine them to obtain the final stroke type prediction, and the results are reported in Table 18. Optical flow performed the best among the four methods with a test accuracy of 83.6%, followed by the RGB method with 76.7% accuracy. On the other hand, the pose method showed the worst performance compared to other methods, indicating that pose estimations have limited significance for our model.

Our approach utilizes different feature extraction methods for each stage of the pipeline. Based on the performance accuracy obtained, we have implemented the RGB method for stages 1 and 2, and the optical flow method for stage 3. By doing so, the highest performance has been achieved for all three stages and the final prediction. Our proposed hybrid approach outperforms other methods in our study, achieving a test accuracy of 88.6% (see Table 18). This demonstrates

Table 18: Accuracy for final stroke type prediction.

Method	Validation Accuracy	Test Accuracy
RGB	81.8%	76.7%
Optical flow	86.0%	83.6%
Pose	66.2%	63.1%
R.O.I.	78.7%	69.4%
RGB + O.F.	90.3%	88.6%
RGB + O.F. (data aug.)	**91.9%**	**90.7%**
TSTCNN [88]	-	91.4%
SSTCNN [93]	-	81.3%

that our approach compensates for the weaknesses of each method and improves overall performance. Moreover, we have also investigated the performance of our hybrid approach using data augmentation on the training data. The results show that our model trained with augmented data achieves a test accuracy of 90.7%, which is comparable to state-of-the-art methods.

3.3.3. Discussion

There are several possible ways to improve the performance of our method in future studies. One approach is to increase the number of samples in the dataset to improve the model's understanding of the data. Additionally, while we have presented a 3-stage pipeline based on our domain knowledge, the structure of this pipeline could be modified to further enhance accuracy. Moreover, collaboration with professionals from the table tennis community to manipulate the class split could also be beneficial in improving accuracy.

It must also be pointed out that we have used the same LSTM-based network architecture for all 3 stages. The network could be individually customized for each stage, and fine-tuning could be done stage-by-stage. This might lead to a better performances, especially in the third stage. Also, for each stage, we used one feature extraction method. However, it is also possible to use a cascaded approach to combine different methods. This might lead to better extraction of relevant features and, hence, improve the overall performance.

Moreover, the methods explored in this chapter —RGB, Pose, Optical Flow and ROI— could be investigated more to get the best performance out of them. For example, we believe that the ROI approach could be further examined by expanding the region to cover the table tennis player's full body and not just the movement of the bat like it was done in our case.

3.4. Conclusion

Classification on time-series data can help to extract the spatio-temporal features from a series of images to improve the accuracy for detecting the right class. Deep

learning based method, especially LSTM have shown strong performance on similar tasks. In this study, we introduced an architecture consisting of LSTM with a feature extraction backbone. The present study aimed to demonstrate the effectiveness of this method through the experimentation with an RGB-based time-series data. We explored four different backbone methods to implement our approach and found that combining RGB images and optical flow outputs yielded the best performance on the selected dataset. Our experimental results show that our proposed method achieved a final test accuracy of 90.7%, which is among the best performances in the state-of-the-art.

4. CONCLUSION AND FUTURE DIRECTIONS

Object recognition represents one of the main problems in computer vision. It has a wide range of application areas, for example, cosmic ray tomography, disease detection, and action recognition. During the last decade, DNN has become popular in this field thanks to its good performance. However, object recognition still has challenges to overcome, which depend on the state of the research in a particular application area.

The subfield of research known as cosmic ray tomography is one of the areas that is rapidly expanding and has already demonstrated proof of concept through the application of reconstruction methods. Studies have started focusing on the detection and classification of objects using the reconstructed images that were produced as a result of the process. The amount of research that has already been carried out in this area is relatively scant due to the fact that the topic is relatively new and is limited to a particular subfield. Although more traditional methods have been applied, the opportunities presented by more contemporary approaches, such as deep learning, have not yet been explored to their full extent. In light of the fact that the precision of reconstruction has a direct bearing on the efficiency with which recognition and classification are carried out, the reconstruction process was initially analysed in order to identify potential areas in which it could be enhanced.

The topic of reconstruction is covered in the first chapter of this thesis. An approach that makes use of neural networks to estimate the location of muon hits has been proposed as a mean of improving the standard of the data that is incorporated into the reconstruction and the further steps.

In the second chapter, we tackled the problem of automatic disease detection in pathological chest images. We proposed a DCNN architecture and utilized a large dataset of 103,468 images to overcome the limitations of previous research. Our experiments showed remarkable accuracy in detecting the target disease with an accuracy of 97%, and also demonstrated the potential of our proposed method in other lung-related diseases.

In the third study, we introduced an architecture consisting of LSTM with a feature extraction backbone for time-series image data classification. We used a table tennis dataset as an example due to its low variant classes. For the selected dataset, our proposed method achieved a final test accuracy of 90.7%, which is among the best performances in the state-of-the-art. We also evaluated the effect of data augmentation and observed an improvement in the performance thanks to that.

This thesis focused on AI methodologies and their broader applications while approaching cosmic ray tomography as an example. Throughout the chapters, the versatility of DNN methodologies and their potential applicability to a wide range of data sources including cosmic ray tomography is presented.

Future directions for study in cosmic ray tomography include evaluating the ef-

fectiveness and usability of suggested deep learning techniques on actual cosmic ray data. Hit location estimation can be further improved by experimenting with various network designs, loss functions, training schemes, and transfer learning methodologies. Furthermore, it is crucial to broaden the application of automatic disease diagnosis to a wider variety of lung-related disorders using larger and more varied datasets unique to cosmic ray tomography. To capture spatial and temporal connections in cosmic ray tomography data, alternative network topologies, such as hybrid models incorporating CNN and LSTM components, can be researched.

BIBLIOGRAPHY

[1] Rahul Dev Singh, Ajay Mittal, and Rajesh K. Bhatia. "3D convolutional neural network for object recognition: a review". In: *Multimedia Tools and Applications* 78.12 (Dec. 2018), pp. 15951–15995. DOI: 10.1007/s11042-018-6912-6.

[2] Josip Stjepandić and Markus Sommer. "Object Recognition Methods in a Built Environment". In: *Springer Series in Advanced Manufacturing*. Springer International Publishing, Aug. 2021, pp. 103–134. DOI: 10.1007/978-3-030-77539-1_6.

[3] Samet Akcay and Toby Breckon. "Towards automatic threat detection: A survey of advances of deep learning within X-ray security imaging". In: *Pattern Recognition* 122 (Feb. 2022), p. 108245. DOI: 10.1016/j.patcog.2021.108245.

[4] Vaishali M Deshmukh et al. "An Overview of Deep Learning Techniques for Autonomous Driving Vehicles". In: *2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT)*. 2022, pp. 979–983. DOI: 10.1109/ICSSIT53264.2022.9716433.

[5] Abdallah Sham et al. "Ethical AI in facial expression analysis: racial bias". In: *Signal, Image and Video Processing* (May 2022). DOI: 10.1007/s11760-022-02246-8.

[6] Kadir Aktas et al. "Spatiotemporal based table tennis stroke-type assessment". In: *Signal, Image and Video Processing* 15.7 (2021), pp. 1593–1600.

[7] Sertan Serte, Ali Serener, and Fadi Al-Turjman. "Deep learning in medical imaging: A brief review". In: *Transactions on Emerging Telecommunications Technologies* 33.10 (Aug. 2020). DOI: 10.1002/ett.4080.

[8] Plamen Angelov and Alessandro Sperduti. "Challenges in Deep Learning." In: *ESANN*. 2016.

[9] Shenda Hong et al. "Opportunities and challenges of deep learning methods for electrocardiogram data: A systematic review". In: *Computers in biology and medicine* 122 (2020), p. 103801.

[10] Nicolae Sapoval et al. "Current progress and open challenges for applying deep learning across the biosciences". In: *Nature Communications* 13.1 (2022), p. 1728.

[11] Linda Wang and Alexander Wong. "COVID-Net: A Tailored Deep Convolutional Neural Network Design for Detection of COVID-19 Cases from Chest X-ray Images". In: *Scientific Reports* (2020).

[12] Yu Kong and Yun Fu. *Human Action Recognition and Prediction: A Survey*. en. Mar. 2022. DOI: 10.1007/s11263-022-01594-9.

[13] Elena Guardincerri et al. "Muon tomography at LANL". In: (Mar. 2015). DOI: 10.2172/1172833.

[14] M. D'Errico et al. "Muon radiography applied to volcanoes imaging: the MURAVES experiment at Mt. Vesuvius". In: *Journal of Instrumentation* 15.03 (Mar. 2020), p. C03014. DOI: 10.1088/1748-0221/15/03/C03014.

[15] G Jonkmans et al. "Muon Tomography for Imaging and Verification of Spent Fuel". In: (Dec. 2007).

[16] V Anghel et al. "A plastic scintillator-based muon tomography system with an integrated muon spectrometer". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 798 (2015), pp. 12–23.

[17] Chanwoo Park et al. "Design and characterization of a Muon tomography system for spent nuclear fuel monitoring". In: *Nuclear Engineering and Technology* 54.2 (2022), pp. 601–607.

[18] Oleg Kamaev et al. "Complementary non-destructive detection of nuclear materials with passive neutron and gamma-ray detectors, and a large-volume muon tomography system". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 944 (2019), p. 162503.

[19] Masayuki Tsuneki. "Deep learning models in medical image analysis". In: *Journal of Oral Biosciences* 64.3 (2022), pp. 312–320.

[20] Natasha Shaukat, Daniyal Mansoor Ali, and Junaid Razzak. "Physical and mental health impacts of COVID-19 on healthcare workers: a scoping review". In: *International journal of emergency medicine* 13 (2020), pp. 1–8.

[21] Wei Fang, Yupeng Chen, and Qiongying Xue. "Survey on research of RNN-based spatio-temporal sequence prediction algorithms". In: *Journal on Big Data* 3.3 (2021), p. 97.

[22] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. "A spatio-temporal descriptor based on 3d-gradients". In: *BMVC 2008-19th British Machine Vision Conference*. British Machine Vision Association. 2008, pp. 275–1.

[23] Jinlu Zhang et al. "Mixste: Seq2seq mixed spatio-temporal encoder for 3d human pose estimation in video". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 13232–13242.

[24] Zehua Sun et al. "Human action recognition from various data modalities: A review". In: *IEEE transactions on pattern analysis and machine intelligence* (2022).

[25] C. Morris et al. "Tomographic Imaging with Cosmic Ray Muons". In: *Science & Global Security* 16 (Oct. 2008). DOI: 10.1080/08929880802335758.

[26] Gary Blanpied et al. "Material discrimination using scattering and stopping of cosmic ray muons and electrons: Differentiating heavier from

lighter metals as well as low-atomic weight materials". In: *Nuclear Instruments and Methods in Physics Research Section A Accelerators Spectrometers Detectors and Associated Equipment* 784 (June 2015), pp. 352–358. DOI: 10.1016/j.nima.2014.11.027.

[27] G. Jonkmans et al. "Nuclear waste imaging and spent fuel verification by muon tomography". In: *Annals of Nuclear Energy* 53 (2013), pp. 267–273. ISSN: 0306-4549. DOI: https://doi.org/10.1016/j.anucene.2012.09.011.

[28] S. Riggi et al. "Muon tomography imaging algorithms for nuclear threat detection inside large volume containers with the Muon Portal detector". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 728 (Nov. 2013), pp. 59–68. ISSN: 0168-9002. DOI: 10.1016/j.nima.2013.06.040.

[29] Xianming Zhang et al. "Performance of long rectangular semi-monolithic scintillator PET detectors". In: *Medical Physics* 46.4 (2019), pp. 1608–1619.

[30] Harutyun Poladyan et al. "Gaussian position-weighted center of gravity algorithm for multiplexed readout". In: *Physics in Medicine & Biology* 65.16 (2020), p. 165003.

[31] P. Aguiar et al. "Geant4-GATE simulation of a large plastic scintillator for muon radiography". In: *2013 3rd International Conference on Advancements in Nuclear Instrumentation, Measurement Methods and their Applications (ANIMMA)*. 2013, pp. 1–5. DOI: 10.1109/ANIMMA.2013.6728082.

[32] Wang Yonggang et al. "FPGA Based Electronics for PET Detector Modules With Neural Network Position Estimators". In: *IEEE Transactions on Nuclear Science* 58.1 (2011), pp. 34–42. DOI: 10.1109/TNS.2010.2081685.

[33] Y Wang et al. "3D position estimation using an artificial neural network for a continuous scintillator PET detector". In: *Physics in medicine and biology* 58 (Mar. 2013), pp. 1375–90. DOI: 10.1088/0031-9155/58/5/1375.

[34] Peng Peng et al. "Compton PET: a simulation study for a PET module with novel geometry and machine learning for position decoding". In: *Biomedical Physics & Engineering Express* 5 (2018), p. 015018.

[35] Sea Agostinelli et al. "GEANT4—a simulation toolkit". In: *Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 506.3 (2003), pp. 250–303.

[36] Victor Babiano et al. "γ-Ray position reconstruction in large monolithic LaCl3 (Ce) crystals with SiPM readout". In: *Nuclear Instruments and*

Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 931 (2019), pp. 1–22.

[37] P Conde et al. "Determination of the interaction position of gamma photons in monolithic scintillators using neural network fitting". In: *IEEE Transactions on Nuclear Science* 63.1 (2016), pp. 30–36.

[38] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. "Understanding of a convolutional neural network". In: *2017 International Conference on Engineering and Technology (ICET)*. 2017, pp. 1–6. DOI: 10.1109/ICEngTechnol.2017.8308186.

[39] Andy LaBella et al. "Convolutional neural network for crystal identification and gamma ray localization in PET". In: *IEEE Transactions on Radiation and Plasma Medical Sciences* 4.4 (2020), pp. 461–469.

[40] Dorota Kamińska et al. "Two-Stage Recognition and beyond for Compound Facial Emotion Recognition". In: *Electronics* 10.22 (2021), p. 2847.

[41] Ilja Pavlovs et al. "Ungulate Detection and Species Classification from Camera Trap Images Using RetinaNet and Faster R-CNN". In: *Entropy* 24 (Feb. 2022), p. 353. DOI: 10.3390/e24030353.

[42] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[43] Vincent Rolfs, Matthias Kerzel, and Stefan Wermter. "Pruning Neural Networks with Supermasks". In: (1988).

[44] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016, p. 114.

[45] Erdi Çallı et al. "Deep learning for chest X-ray analysis: A survey". In: *Medical Image Analysis* 72 (2021), p. 102125.

[46] Mingyu Kim et al. "Deep learning in medical imaging". In: *Neurospine* 16.4 (2019), p. 657.

[47] Vertika Agarwal et al. "Analysis of deep learning techniques for chest x-ray classification in context of covid-19". In: *ADI Journal on Recent Innovation* 3.2 (2022), pp. 208–216.

[48] Liton Devnath et al. "Computer-aided diagnosis of coal workers' pneumoconiosis in chest x-ray radiographs using machine learning: A systematic literature review". In: *International Journal of Environmental Research and Public Health* 19.11 (2022), p. 6439.

[49] S Aslani and J Jacob. "Utilisation of deep learning for COVID-19 diagnosis". In: *Clinical Radiology* 78.2 (2023), pp. 150–157.

[50] Ioannis D Apostolopoulos and Tzani A Mpesiana. "Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks". In: *Physical and Engineering Sciences in Medicine* 43.2 (2020), pp. 635–640.

[51] Tripti Goel et al. “OptCoNet: an optimized convolutional neural network for an automatic diagnosis of COVID-19”. In: *Appl. Intell.* 51.3 (2021), pp. 1351–1366. DOI: 10.1007/s10489-020-01904-z.

[52] Ioannis D Apostolopoulos, Sokratis I Aznaouridis, and Mpesiana A Tzani. “Extracting possibly representative COVID-19 biomarkers from X-ray images with deep learning approach and image data related to pulmonary diseases”. In: *Journal of Medical and Biological Engineering* 40 (2020), pp. 462–469.

[53] Apinya Koontalay et al. “Healthcare workers’ burdens during the COVID-19 pandemic: A qualitative systematic review”. In: *Journal of Multidisciplinary Healthcare* (2021), pp. 3015–3025.

[54] Benedetta Armocida et al. “The Italian health system and the COVID-19 challenge”. In: *The Lancet Public Health* 5.5 (2020), e253.

[55] Shuo Wang et al. “A fully automatic deep learning system for COVID-19 diagnostic and prognostic analysis”. In: *European Respiratory Journal* 56.2 (2020), p. 2000775. DOI: 10.1183/13993003.00775-2020.

[56] Ezz El-Din Hemdan, Marwa A Shouman, and Mohamed Esmail Karar. “Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images”. In: *arXiv preprint arXiv:2003.11055* (2020).

[57] Wang Linda. “A tailored deep convolutional neural network design for detection of covid-19 cases from chest radiography images”. In: *Journal of Network & Computer Applications* 20 (2020), pp. 1–12.

[58] Prasoon Kumar Vinodkumar, Cagri Ozcinar, and Gholamreza Anbarjafari. “Prediction of sgRNA Off-Target Activity in CRISPR/Cas9 Gene Editing Using Graph Convolution Network”. In: *Entropy* 23.5 (2021), p. 608.

[59] Heba Elshatoury et al. “Disentangling the association between genetics and functional connectivity in Mild Cognitive Impairment”. In: *2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI)*. IEEE. 2021, pp. 1–4.

[60] Egils Avots et al. “Ensemble approach for detection of depression using EEG features”. In: *Entropy* 24.2 (2022), p. 211.

[61] Ali Narin, Ceren Kaya, and Ziynet Pamuk. “Automatic detection of coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks”. In: *Pattern Analysis and Applications* (May 2021). ISSN: 1433-755X. DOI: 10.1007/s10044-021-00984-y.

[62] Shervin Minaee et al. “Deep-COVID: Predicting COVID-19 from chest X-ray images using deep transfer learning”. In: *Medical Image Analysis* 65 (July 2020), p. 101794. DOI: 10.1016/j.media.2020.101794.

[63] Sameena Pathan, P. C. Siddalingaswamy, and Tanweer Ali. “Automated Detection of Covid-19 from Chest X-ray scans using an optimized CNN architecture”. English. In: *Applied Soft Computing* 104 (June 2021). ISSN: 1568-4946. DOI: 10.1016/j.asoc.2021.107238.

[64] Aurelia Bustos et al. “Padchest: A large chest x-ray image dataset with multi-label annotated reports”. In: *Medical image analysis* 66 (2020), p. 101797.

[65] Maria de la Iglesia Vayá et al. “Bimcv covid-19+: a large annotated dataset of rx and ct images from covid-19 patients”. In: *arXiv preprint arXiv:2006.01174* (2020).

[66] Muhammad EH Chowdhury et al. “Can AI help in screening viral and COVID-19 pneumonia?” In: *IEEE Access* 8 (2020), pp. 132665–132676.

[67] Tawsifur Rahman et al. “Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images”. In: *Computers in biology and medicine* 132 (2021), p. 104319.

[68] N Parveen and M Mohamed Sathik. “Detection of pneumonia in chest X-ray images”. In: *Journal of X-ray Science and Technology* 19.4 (2011), pp. 423–428.

[69] Christian Szegedy et al. “Rethinking the Inception Architecture for Computer Vision”. In: *CoRR* abs/1512.00567 (2015). arXiv: 1512.00567. URL: http://arxiv.org/abs/1512.00567.

[70] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015).

[71] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[72] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[73] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[74] Barret Zoph et al. “Learning Transferable Architectures for Scalable Image Recognition”. In: *CoRR* abs/1707.07012 (2017). arXiv: 1707.07012. URL: http://arxiv.org/abs/1707.07012.

[75] Brian W Matthews. “Comparison of the predicted and observed secondary structure of T4 phage lysozyme”. In: *Biochimica et Biophysica Acta (BBA)-Protein Structure* 405.2 (1975), pp. 442–451.

[76] Tulin Ozturk et al. “Automated detection of COVID-19 cases using deep neural networks with X-ray images”. In: *Computers in biology and medicine* 121 (2020), p. 103792.

[77] M.M. Gaber A. Abbas M.M. Abdelsamea. “Classification of COVID-19 in chest X-ray images using detrac deep convolutional neural network”. In: *arXiv preprint arXiv:2003.13815* (2020).

[78] Yujin Oh, Sangjoon Park, and Jong Chul Ye. “Deep Learning COVID-19 Features on CXR Using Limited Training Data Sets”. In: *IEEE Transac-*

tions on Medical Imaging 39.8 (2020), pp. 2688–2700. DOI: 10.1109/TMI.2020.2993291.

[79] Majid Nour, Zafer Cömert, and Kemal Polat. "A Novel Medical Diagnosis model for COVID-19 infection detection based on Deep Features and Bayesian Optimization". English. In: *Applied Soft Computing* 97 (Dec. 2020). DOI: 10.1016/j.asoc.2020.106580.

[80] Hritam Basak et al. "A union of deep learning and swarm-based optimization for 3D human action recognition". In: *Scientific Reports* 12.1 (2022), p. 5494.

[81] Yu Kong and Yun Fu. "Human action recognition and prediction: A survey". In: *International Journal of Computer Vision* 130.5 (2022), pp. 1366–1401.

[82] Preksha Pareek and Ankit Thakkar. "A survey on video-based human action recognition: recent updates, datasets, challenges, and applications". In: *Artificial Intelligence Review* 54 (2021), pp. 2259–2322.

[83] Jun Yin et al. "A skeleton-based action recognition system for medical condition detection". In: *2019 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE. 2019, pp. 1–4.

[84] Muhammad Attique Khan et al. "Human action recognition using fusion of multiview and deep features: an application to video surveillance". In: *Multimedia tools and applications* (2020), pp. 1–27.

[85] Isidoros Rodomagoulakis et al. "Multimodal human action recognition in assistive human-robot interaction". In: *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2016, pp. 2702–2706.

[86] AJ Piergiovanni and Michael S Ryoo. "Fine-grained activity recognition in baseball videos". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 1740–1748.

[87] Svein Arne Pettersen et al. "Soccer video and player position dataset". In: *Proceedings of the 5th ACM Multimedia Systems Conference*. 2014, pp. 18–23.

[88] Martin Pierre-Etienne et al. "Fine grained sport action recognition with Twin spatio-temporal convolutional neural networks". In: *Multimedia Tools and Applications* 79.20429-20447 (2020), pp. 85–97.

[89] Jeff Donahue et al. "Long-term Recurrent Convolutional Networks for Visual Recognition and Description". In: *CoRR* abs/1411.4389 (2014). arXiv: 1411.4389. URL: http://arxiv.org/abs/1411.4389.

[90] Amin Ullah et al. "Activity recognition using temporal optical flow convolutional features and multilayer LSTM". In: *IEEE Transactions on Industrial Electronics* 66.12 (2018), pp. 9692–9702.

[91] Ngoc-Hoang Nguyen et al. “Gesture recognition based on 3D human pose estimation and body part segmentation for RGB data input”. In: *Applied Sciences* 10.18 (2020), p. 6188.

[92] Siddharth Sriraman et al. “MediaEval 2019: LRCNs for Stroke Detection in Table Tennis”. In: *MediaEval*. 2019.

[93] Pierre-Etienne Martin et al. “Optimal choice of motion estimation methods for fine-grained action classification with 3d convolutional networks”. In: *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2019, pp. 554–558.

[94] Kaustubh Milind Kulkarni and Sucheth Shenoy. “Table Tennis Stroke Recognition Using Two-Dimensional Human Pose Estimation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2021, pp. 4576–4584.

[95] Robi Polikar. “Ensemble based systems in decision making”. In: *IEEE Circuits and systems magazine* 6.3 (2006), pp. 21–45.

[96] Hasan Demirel and Gholamreza Anbarjafari. “Data fusion boosted face recognition based on probability distribution functions in different colour channels”. In: *EURASIP Journal on Advances in Signal Processing* 2009.1 (2009), p. 482585.

[97] E. Ilg et al. “FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[98] Fitsum Reda et al. *flownet2-pytorch: Pytorch implementation of FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks*. `https://github.com/NVIDIA/flownet2-pytorch`. 2017.

[99] Hao-Shu Fang et al. “AlphaPose: Whole-Body Regional Multi-Person Pose Estimation and Tracking in Real-Time”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

[100] *AlphaPose*. `https://github.com/MVIG-SJTU/AlphaPose`.

[101] Petru Soviany and Radu Tudor Ionescu. “Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction”. In: *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*. IEEE. 2018, pp. 209–214.

[102] G. Jocher et al. *YOLOv5*. `https://github.com/ultralytics/yolov5`. 2020.

[103] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[104] F. A. Gers, J. Schmidhuber, and F. Cummins. “Learning to forget: continual prediction with LSTM”. In: *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*. Vol. 2. 1999, 850–855 vol.2. DOI: `10.1049/cp:19991218`.

[105] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, pp. 249–256.

[106] Tijmen Tieleman and Geoffrey Hinton. “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude”. In: *COURSERA: Neural networks for machine learning* 4.2 (2012), pp. 26–31.

[107] Qingjiu Zhang and Shiliang Sun. “A centroid k-nearest neighbor method”. In: *International Conference on Advanced Data Mining and Applications*. Springer. 2010, pp. 278–285.

ACKNOWLEDGEMENTS

This work was partly supported by the EU-funded project H2020-SU-SEC-2020 SilentBorder (101021812), the University of Tartu's feasibility grant (83839973, and the Estonian Centre of Excellence in IT (EXCITE) funded by the European Regional Development Fund. The author acknowledges the support of NVIDIA Corporation with the donation of GPU used for this research. The author would also like to acknowledge Anzori Georgadze for providing the GEANT4 simulation data.

SUMMARY IN ESTONIAN

Kosmilisel kiirgusel põhinev objektide rekonstrueerimine ja äratundmine

Tehnoloogia kiire areng ja valdkonnas tehtud edukad teadustööd on toonud endaga kaasa tehisintellektil põhinevate meetodite kasutamise laialdase leviku madalamatasemeliste ülesannete, näiteks objektituvastuse, lahendamisel. Objektituvastus proovib leida huvipakkuvaid objekte pildilt või pildijadadelt. Tuvastuse käigus lokaliseeritakse ja klassifitseeritakse pildil olevad objektid.

Objektituvastus leiab laialdast kasutust mitmete reaalmaailma probleemide lahendamisel ja on seetõttu üks peamisi ülesandeid, mida arvutinägemises lahendatakse. Seda rakendatakse näiteks tomograafiasüsteemide väljundite uurimises, inimkäitumise analüüsis, meditsiinis ja spordi valdkonnas.

Vaatamata objektituvastuse pikaajalisele käibel olemisele, on selles valdkonnas jätkuvalt palju väljakutsuvaid teemasid, mida uurida. Üheks selliseks näiteks on süvaõppe kasutamine müüontomograafias. Müüonid on elektroni sarnased elementaarosakesed, mis erinevaid materjale läbides neelduvad ja hajuvad erineva nurga all. Need hajumisnurgad peidavad endas informatsioon läbitud objektide kohta. Maal tekivad müüonid enamasti kosmilise kiirguse sisenemisel Maa atmosfääri. Müüontomograafias mõõdetakse osakeste sisenemis- ja väljumisnurkasid ümber huvipakkuva objekti. Nende nurkade põhjal rekonstrueeritakse objektist tomograafilised pildid. Saadud pilte kasutatakse objektituvastuses. Objektituvastuse tulemus on otseselt mõjutatud rekonstruktsioonist ja sellest kui täpselt suudetakse lokaliseerida müüonite tabamuspunkte detektori plaatidelt.

Sügavate närvivõrkude kasutamine ülesande lahendamisel parandab märkimisväärselt tulemuste täpsust võrreldes traditsioonilise raskuskeskme meetodiga. Käesolevas töös näidatakse sügavate konvolutsiooniliste närvivõrkude edukat utiliseerimist objektituvastuse ülesannete lahendamisel keerukate ja mitmekesiste andmekogumite peal.

PUBLICATIONS

CURRICULUM VITAE

Personal data

Name:	Kadir Aktas
Date of birth:	20 October 1992
Contacts:	kadiraktass@outlook.com
Citizenship:	Turkish

Education

2020–2023	PhD, Engineering and Technology, University of Tartu
2017–2019	MSc, Robotics and Computer Engineering, University of Tartu
2010–2015	BSc, Electrical and Electronics Engineering, Hacettepe University

Employment

2022–...	Machine Learning Lead, GScan, Estonia
2019–...	Chief Executive Officer, iVCV, Estonia
2019–2021	Software Developer, Singleton, Estonia
2016–2017	Engineer, Tubitak, Turkey
2015–2016	Engineer, Aselsan, Turkey

ELULOOKIRJELDUS

Isikuandmed

Nimi:	Kadir Aktas
Sünniaeg:	20 Oktoober 1992
Kontaktandmed:	kadiraktass@outlook.com
Kodakondsus:	Türgi

Haridus

2020–2023	PhD, Tehnika ja tehnoloogia, Tartu Ülikool
2017–2019	MSc, Arvutitehnika ja robootika, Tartu Ülikool
2010–2015	BSc, Elektri- ja elektroonika tehnika, Hacettepe Ülikool

Teenistuskäik

2022–...	Masinõppe Juht, GScan, Eesti
2019–...	Tegevjuht, iVCV, Eesti
2019–2021	Tarkvaraarendaja, Singleton, Eesti
2016–2017	Insener, Tubitak, Türgi
2015–2016	Insener, Aselsan, Türgi

DISSERTATIONES TECHNOLOGIAE UNIVERSITATIS TARTUENSIS

1. **Imre Mäger**. Characterization of cell-penetrating peptides: Assessment of cellular internalization kinetics, mechanisms and bioactivity. Tartu 2011, 132 p.
2. **Taavi Lehto**. Delivery of nucleic acids by cell-penetrating peptides: application in modulation of gene expression. Tartu 2011, 155 p.
3. **Hannes Luidalepp**. Studies on the antibiotic susceptibility of *Escherichia coli*. Tartu 2012, 111 p.
4. **Vahur Zadin**. Modelling the 3D-microbattery. Tartu 2012, 149 p.
5. **Janno Torop**. Carbide-derived carbon-based electromechanical actuators. Tartu 2012, 113 p.
6. **Julia Suhorutšenko**. Cell-penetrating peptides: cytotoxicity, immunogenicity and application for tumor targeting. Tartu 2012, 139 p.
7. **Viktoryia Shyp**. G nucleotide regulation of translational GTPases and the stringent response factor RelA. Tartu 2012, 105 p.
8. **Mardo Kõivomägi**. Studies on the substrate specificity and multisite phosphorylation mechanisms of cyclin-dependent kinase Cdk1 in *Saccharomyces cerevisiae*. Tartu, 2013, 157 p.
9. **Liis Karo-Astover**. Studies on the Semliki Forest virus replicase protein nsP1. Tartu, 2013, 113 p.
10. **Piret Arukuusk**. NickFects–novel cell-penetrating peptides. Design and uptake mechanism. Tartu, 2013, 124 p.
11. **Piret Villo**. Synthesis of acetogenin analogues. Asymmetric transfer hydrogenation coupled with dynamic kinetic resolution of α-amido-β-keto esters. Tartu, 2013, 151 p.
12. **Villu Kasari**. Bacterial toxin-antitoxin systems: transcriptional cross-activation and characterization of a novel *mqsRA* system. Tartu, 2013, 108 p.
13. **Margus Varjak**. Functional analysis of viral and host components of alphavirus replicase complexes. Tartu, 2013, 151 p.
14. **Liane Viru**. Development and analysis of novel alphavirus-based multifunctional gene therapy and expression systems. Tartu, 2013, 113 p.
15. **Kent Langel**. Cell-penetrating peptide mechanism studies: from peptides to cargo delivery. Tartu, 2014, 115 p.
16. **Rauno Temmer**. Electrochemistry and novel applications of chemically synthesized conductive polymer electrodes. Tartu, 2014, 206 p.
17. **Indrek Must**. Ionic and capacitive electroactive laminates with carbonaceous electrodes as sensors and energy harvesters. Tartu, 2014, 133 p.
18. **Veiko Voolaid**. Aquatic environment: primary reservoir, link, or sink of antibiotic resistance? Tartu, 2014, 79 p.
19. **Kristiina Laanemets**. The role of SLAC1 anion channel and its upstream regulators in stomatal opening and closure of *Arabidopsis thaliana.* Tartu, 2015, 115 p.

20. **Kalle Pärn**. Studies on inducible alphavirus-based antitumour strategy mediated by site-specific delivery with activatable cell-penetrating peptides. Tartu, 2015, 139 p.
21. **Anastasia Selyutina**. When biologist meets chemist: a search for HIV-1 inhibitors. Tartu, 2015, 172 p.
22. **Sirle Saul**. Towards understanding the neurovirulence of Semliki Forest virus. Tartu, 2015, 136 p.
23. **Marit Orav**. Study of the initial amplification of the human papillomavirus genome. Tartu, 2015, 132 p.
24. **Tormi Reinson**. Studies on the Genome Replication of Human Papillomaviruses. Tartu, 2016, 110 p.
25. **Mart Ustav Jr**. Molecular Studies of HPV-18 Genome Segregation and Stable Replication. Tartu, 2016, 152 p.
26. **Margit Mutso**. Different Approaches to Counteracting Hepatitis C Virus and Chikungunya Virus Infections. Tartu, 2016, 184 p.
27. **Jelizaveta Geimanen**. Study of the Papillomavirus Genome Replication and Segregation. Tartu, 2016, 168 p.
28. **Mart Toots**. Novel Means to Target Human Papillomavirus Infection. Tartu, 2016, 173 p.
29. **Kadi-Liis Veiman**. Development of cell-penetrating peptides for gene delivery: from transfection in cell cultures to induction of gene expression *in vivo*. Tartu, 2016, 136 p.
30. **Ly Pärnaste**. How, why, what and where: Mechanisms behind CPP/cargo nanocomplexes. Tartu, 2016, 147 p.
31. **Age Utt**. Role of alphavirus replicase in viral RNA synthesis, virus-induced cytotoxicity and recognition of viral infections in host cells. Tartu, 2016, 183 p.
32. **Veiko Vunder**. Modeling and characterization of back-relaxation of ionic electroactive polymer actuators. Tartu, 2016, 154 p.
33. **Piia Kivipõld**. Studies on the Role of Papillomavirus E2 Proteins in Virus DNA Replication. Tartu, 2016, 118 p.
34. **Liina Jakobson**. The roles of abscisic acid, CO_2, and the cuticle in the regulation of plant transpiration. Tartu, 2017, 162 p.
35. **Helen Isok-Paas**. Viral-host interactions in the life cycle of human papillomaviruses. Tartu, 2017, 158 p.
36. **Hanna Hõrak**. Identification of key regulators of stomatal CO_2 signalling via O_3-sensitivity. Tartu, 2017, 260 p.
37. **Jekaterina Jevtuševskaja**. Application of isothermal amplification methods for detection of *Chlamydia trachomatis* directly from biological samples. Tartu, 2017, 96 p.
38. **Ülar Allas**. Ribosome-targeting antibiotics and mechanisms of antibiotic resistance. Tartu, 2017, 152 p.
39. **Anton Paier**. Ribosome Degradation in Living Bacteria. Tartu, 2017, 108 p.
40. **Vallo Varik**. Stringent Response in Bacterial Growth and Survival. Tartu, 2017, 101 p.

41. **Pavel Kudrin**. In search for the inhibitors of *Escherichia coli* stringent response factor RelA. Tartu, 2017, 138 p.
42. **Liisi Henno**. Study of the human papillomavirus genome replication and oligomer generation. Tartu, 2017, 144 p.
43. **Katrin Krõlov**. Nucleic acid amplification from crude clinical samples exemplified by *Chlamydia trachomatis* detection in urine. Tartu, 2018, 118 p.
44. **Eve Sankovski**. Studies on papillomavirus transcription and regulatory protein E2. Tartu, 2018, 113 p.
45. **Morteza Daneshmand**. Realistic 3D Virtual Fitting Room. Tartu, 2018, 233 p.
46. **Fatemeh Noroozi**. Multimodal Emotion Recognition Based Human-Robot Interaction Enhancement. Tartu, 2018, 113 p.
47. **Krista Freimann**. Design of peptide-based vector for nucleic acid delivery in vivo. Tartu, 2018, 103 p.
48. **Rainis Venta**. Studies on signal processing by multisite phosphorylation pathways of the *S. cerevisiae* cyclin-dependent kinase inhibitor Sic1. Tartu, 2018, 155 p.
49. **Inga Põldsalu**. Soft actuators with ink-jet printed electrodes. Tartu, 2018, 85 p.
50. **Kadri Künnapuu**. Modification of the cell-penetrating peptide PepFect14 for targeted tumor gene delivery and reduced toxicity. Tartu, 2018, 114 p.
51. **Toomas Mets**. RNA fragmentation by MazF and MqsR toxins of *Escherichia coli*. Tartu, 2019, 119 p.
52. **Kadri Tõldsepp**. The role of mitogen-activated protein kinases MPK4 and MPK12 in CO_2-induced stomatal movements. Tartu, 2019, 259 p.
53. **Pirko Jalakas**. Unravelling signalling pathways contributing to stomatal conductance and responsiveness. Tartu, 2019, 120 p.
54. **S. Sunjai Nakshatharan**. Electromechanical modelling and control of ionic electroactive polymer actuators. Tartu, 2019, 165 p.
55. **Eva-Maria Tombak**. Molecular studies of the initial amplification of the oncogenic human papillomavirus and closely related nonhuman primate papillomavirus genomes. Tartu, 2019, 150 p.
56. **Meeri Visnapuu**. Design and physico-chemical characterization of metal-containing nanoparticles for antimicrobial coatings. Tartu, 2019, 138 p.
57. **Jelena Beljantseva**. Small fine-tuners of the bacterial stringent response – a glimpse into the working principles of Small Alarmone Synthetases. Tartu, 2020, 104 p.
58. **Egon Urgard**. Potential therapeutic approaches for modulation of inflammatory response pathways. Tartu, 2020, 120 p.
59. **Sofia Raquel Alves Oliveira**. HPLC analysis of bacterial alarmone nucleotide (p)ppGpp and its toxic analogue ppApp. Tartu, 2020, 122 p.
60. **Mihkel Örd**. Ordering the phosphorylation of cyclin-dependent kinase Cdk1 substrates in the cell cycle. Tartu, 2021, 228 p.
61. **Fred Elhi**. Biocompatible ionic electromechanically active polymer actuator based on biopolymers and non-toxic ionic liquids. Tartu, 2021, 140 p.

62. **Liisi Talas**. Reconstructing paleo-diversity, dynamics and response of eukaryotes to environmental change over the Late-Glacial and Holocene period in lake Lielais Svētiņu using sedaDNA. Tartu, 2021, 118 p.
63. **Livia Matt**. Novel isosorbide-based polymers. Tartu, 2021, 118 p.
64. **Koit Aasumets**. The dynamics of human mitochondrial nucleoids within the mitochondrial network. Tartu, 2021, 104 p.
65. **Faiza Summer**. Development and optimization of flow electrode capacitor technology. Tartu, 2022, 109 p.
66. **Olavi Reinsalu**. Cancer-testis antigen MAGE-A4 is incorporated into extracellular vesiclesand is exposed to the surface. Tartu, 2022, 130 p.
67. **Tetiana Brodiazhenko**. RelA-SpoT Homolog enzymes as effectors of Toxin-Antitoxin systems. Tartu, 2022, 132 p.
68. **Georg-Marten Lanno**. Development of novel antibacterial drug delivery systems as wound scaffolds using electrospinning technology. Tartu, 2022, 175 p.
69. **Liubov Cherkashchenko**. New insights into alphaviral nsP2 functions. Tartu, 2023, 171 p.
70. **Kristina Kiisholts**. Peptide-based drug carriers and preclinical nanomedicine applications for endometriosis treatment. Tartu, 2023, 138 p.
71. **Kai Rausalu**. Alphaviral nsP2 protease: From requirements for functionality to inhibition. Tartu, 2023, 175 p.
72. **Laura Sandra Lello**. Unraveling the intricate nature of the alphavirus RNA replicase. Tartu, 2023, 219 p.
73. **Houman Masnavi**. Visibility Aware Navigation. Tartu, 2023, 180 p.