

**IN BASED HEAD-MOUSE CONTROL SYSTEM FOR
PEOPLE WITH DISABILITIES**

**A THESIS SUBMITTED TO THE
INSTITUTE OF GRADUATE STUDIES
OF
NEAR EAST UNIVERSITY**

**By
MURAT ARSLAN**

**In Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy
in
Computer Engineering**

NICOSIA, 2021

**CNN BASED HEAD-MOUSE CONTROL SYSTEM
FOR PEOPLE WITH DISABILITIES**

**A THESIS SUBMITTED TO THE
GRADUATE SCHOOL OF APPLIED SCIENCES
OF
NEAR EAST UNIVERSITY**

**By
MURAT ARSLAN**

**In Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy
in
Biomedical Engineering**

NICOSIA, 2021

**Murat Arslan: CNN BASED HEAD-MOUSE CONTROL SYSTEM FOR
PEOPLE WITH DISABILITIES**

**Approval of Director of Institute of
Graduate Studies**

Prof. Dr.Hüsnü Can Be er

**We certify this thesis is satisfactory for the award of the degree of Doctor of
Philosophy in Computer Engineering**

Examining Committee in Charge:

Prof Dr.RashadAliyev
Mediterranean University, TRNC

Department of Mathematics, Eastern

Assoc.ProfDr.MusbahAqel School of Applied Sciences,
MUSAQEL Cyprus International University, TRNC

Assist.Prof Dr.Mustafa Tunay
Gelisim University, TURKEY

Department of Computer Engineering,

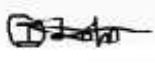
Assoc.ProfDr.KamilDimililerDepartment of Automotive Engineering,
KAMIL Near East University, TRNC

Prof Dr.RahibAbiyev Supervisor, Department of Computer

Engineering, Near East University,
TRNC

I hereby declare that all information contained in this document has been collected and presented in compliance with academic legislation and ethical standards. I also declare that, as provided by these Rules and Conduct, all materials and findings that are not original to this work have been thoroughly cited and referenced.

Name, Surname: Murat ARSLAN

Signature: 

Date:

14/04/20

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Prof. Dr. RahibAbiyev, who has supported and guided me with his vast knowledge, and also for his patience in ensuring the completion of this thesis.

I dedicate my success to the pure spirit of my father who always supported me in my studies. I would like to thank Near East University for giving me the opportunity to write my thesis.

I would also like to thank all the lecturers at Near East University who taught me during my Phd's period at the university.

Finally, I thank my friends who supported me in every possible way

ABSTRACT

The head-mouse control system for disabled people is proposed in this thesis. The designed system is a human-machine interface that is designated for disabled people injured with the spinal cord. The system controls the mouse using head movements and eye states detection. The user moves the mouse cursor to the required coordinates using head motions and transmits a command with eye blinks in the suggested system. Eye blinks are used for confirming the selected action on the user window. The developed interface is based on image processing that includes face recognition, especially eye, mouth and nose recognition. The existing methods have several limitations. Some of them require special hardware, such as certain cameras or sensor-based devices, some have additional components and the user has to wear these components. These systems use special methods to solve feature extraction and classification problems. In the thesis, deep learning using Convolutional Neural Networks (CNN) is proposed for solving the indicated problem. The proposed system integrates images' feature extraction and classification stages in the unified body of CNN, which allows simplifying the system structure used for image recognition. The convolutional layers, a pooling layer, and a fully connected network are basic blocks of CNN. The CNN converts the mouse's actual coordinates into head movements. The suggested recognition method allows low-quality images acquired by a computer's camera to be processed and recognized. A built-in computer camera captures the image of the individual in this system. The camera's image is divided in the object recognition block, and the user's head-shoulder profile is detected. Segmentation of the head-shoulder image, as well as recognition of a person's head profile and eyes, are conducted in the next stage. The CNNs perform the extraction of the features of the head profile and eyes as input and their classification. Two CNN1 and CNN2 are designed for the classification of head and eye images. The first CNN1 is utilized to detect the up, down, left, right and no-action directions of the head profile. The second CNN2 is used to detect eye states, which can be open (No-action) or closed (Ok). CNN1 and CNN2 network outputs are translated to mouse control signals and forwarded to the appropriate action. Experiment results have shown that the developed system is reliable and accurate.

The system allows the persons having disabilities to control the cursor and buttons of mouse by moving head and blinking eyes.

KEYWORDS: Disabled people, convolutional neural network, deep learning, computer vision

ÖZET

Bu tezde, omuriliktenyaralananengellileriçinbirinsan-makinearayüzüönerilmi tir.Tasarlananinsan-makinearayüzü, fare kontrolüiçinkafahareketivegözkırpmayıkullananyardımcıbirsistemdir.Önerilensistemde, kullanıcı fare imleciniistenenkoordinatlarahareketettirirvegözkırpmagöndermekomutunukullanır.Dikkate alınankafa-fare kontrolü, özelliklegözlerin, a zınveburnuntanınmasıgibiüyztanımadahilolmaküzeregörüntü lemeyedayanmaktadır.

Bununlabirlikte, mevcutyöntemlerinbazısınırlamalarıvardır.Bazıları, belirlikameralarveyasensörtabanlıgibiözeldonanımaraçlarıgerektirir.Bazılarınınekstrabile e nlerivardırvekullanıcılarınbubile enleritakmasıgerekir.Di erklasikyakla ım, özelliklerinçıkartılmasıvesınıflandırmaproblemlerininçözümüiçinözelmetodolojilerkullanır. Önerilentanımasistemi, birbilgisayarınkamasındanyakalanandü ükkaliteligörüntülerikullananevri imlisinira ların a (CNN) dayanmaktadır.Evri imliSinirA 1 (CNN), evri imlikatmanları, birhavuzkatmanınıvetamamenba lıbira içerir. CNN, kafahareketinifareningerçekkoordinatlarınadönü türür.Tasarlanansistem, engelliki ilerini fare imlecini kafahareketleriylevegözkırparak fare dü meleriylekontroletmelerineolanaktandır.

Bu sistemdeyerle ikbilgisayarkamasıki iningörüntüalımınıgerçekle tirir.Kameratarafındanalı nangörüntünesnealgılamablo undabölümlereayrılırvekullanıcınınba omuzprofilialgılanır.S onrakia amadaba omuzgörüntüsününsegmentasyonuvebirki ininba profilivegözlerinintesp itigerçekle tirilir.Ba profilivegözlerininalgılanangörüntüleri, sınıflandırmayıuygulayanCNN'leriçingirdidir.Bu görüntüleriki CNN1 ve CNN2 a ınıngiri lerinegiriyo. lk CNN1, sol, sa , yukarı, a a ıveHareketsizolabilenkafaprofilinin yönlerinintanınmasıiçinkullanılır. kinci CNN2, kapalı (Ok) veAçık (Eylemsiz) olabilengözlerindurumlarınıntanınmasıiçinkullanılır. CNN1 ve CNN2 a larınınçıkıtları fare kontrolsinyallerinedönü türülürveilgileylemiçingönderilir.

Deneilerin sonuçları, bu sistemin insanları yönlendirebilme kapasitesini göstermektedir. Bu bulgu, herhangi bir engel olmadan insanların fare imlecive fare düğmelerini özgürce kontrol etmelerine olanak tanır.

Anahtar Kelimeler: Disabled people, convolutional neural network, computer vision, deep learning.

TABLE OF CONTENTS

ABSTRACT

ÖZET

ACKNOWLEDGMENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1. Definition of Problem

2. Thesis overview

2. STATE OF ART OF DEEP LEARNING FOR THE DESIGN OF HEAD-MOUSE CONTROL SYSTEM

2.1. Overview

2.2. Review of Existing Research Works Used for Head-Mouse Control

2.3. Deep Learning Based on CNN Architectures

2.4. State of Deep Learning Architectures for the Design of Head-Mouse Control

3. HEAD MOUSE CONTROL SYSTEM

3.1. Overview

3.2. Structure of the System

3.3. Object Detection

3.4. Feature Extraction

3.5. Classification

4. DEEP LEARNING BASED ON CONVOLUTIONAL NEURAL NETWORK FOR HEAD MOUSE CONTROL

4.1. Overview

4.2. Machine Learning Systems

4.3. Artificial Neural Networks

4.3.1. Basic Concepts

4.3.2. Multilayer NNs

4.4. ANN Learning

- 4.4.1. Gradient Descent Method
- 4.4.2. Backpropagation
- 4.4.3. Regularization
- 4.5. Convolutional Neural Networksfor Head-Mouse Control
 - 4.5.1. Basic architecture
 - 4.5.2. Convolutional layer
 - 4.5.3. Pooling
 - 4.5.4. Batch normalization
 - 4.5.5. Properties of CNN

5. MODELLING OF CNN BASED HEAD MOUSE CONTROL

- 5.1.Design of CNN based head mouse control system
- 5.2.Modelling
- 5.3. Simulation Results and Discussion

6. CONCLUSIONS

REFERENCES

APPENDICES

LIST OF FIGURES

Figure 2.1. The structure of the head mouse control system

Figure 3.2. Cascade classifier approach

Figure 3.3. A simple multi-layer neural network structure

Figure 4.1: A biological neuron (a) compared to an artificial neuron (b).

Figure 4.2. Popular activation functions

Figure 4.3. A feedforward network.

Figure 4.4. Single-layer Perceptrons

Figure 4.5. A multilayer perceptron with a single hidden layer.

Figure 4.6. Backpropagation: forward (a), backward (b) pass

Figure 4.7. Convolutional Neural Networks

Figure 4.8. Adam optimizer (Kingma& Jimmy, 2015)

Figure 4.9. Pooling

Figure 4.10. CNN structure

Figure 4.11. Local receptive fields

Figure 4.12. Deep indirect interactions

Figure 5.1. Fragment of Head Pose Dataset

Figure 5.2. Detection of eyes

Figure 5.3. Training and validation results obtained for loss and accuracy of head classification

Figure 5.4. Training and validation results obtained for loss and accuracy of eyes classification

LIST OF TABLES

Table 5.1. The structure of the CNN1 model

Table 5.2. The structure of the CNN2 model

Table 5.3. Performances of the CNN models

Table 5.4. The simulation results of the selected studies obtained for the head-mouse control

Table 5.5.Comparative Results

CHAPTER 1

INTRODUCTION

Computer technologies now play a significant part in human life. People with impairments find it difficult, if not impossible, to make efficient use of modern tools. Designing an effective human-computer interface for impaired persons is extremely beneficial, and it can even lead to career prospects. For this reason, computer-aided design research is gaining traction and becoming increasingly relevant. Specific computer aids have recently been proposed for various groups of impaired individuals (Chen et al., 1999; Lin et al., 2002; Park & Lee, 1996). These methods are based on image processing or sensor-based tracking systems, which detect head motions for mouse control using image processing or physiological inputs. The optical sensors in sensor-based solutions may have limited resolution, be expensive, and be cumbersome to use over time. Furthermore, these sensors only supply a limited amount of data for subsequent calculations. Video camera solutions, on the other hand, are far less expensive and can save all visual information for future solutions. Some related applications have already made use of visual data. These include face recognition (Viola & Jones, 2001; Rowley et al., 1998; Osuna et al., 1997), face recognition (Ahonen et al., 2006; Senaratne et al., 2009; Zhang et al., 2013; Zho et al., 2003), emotion recognition (Fragopanagos & Taylor, 2005; Adolphs et al., 2000; Cohen et al., 2003), etc. Some recent studies have combined the aforementioned two groups; camera and different sensors for head-mouse control. The video camera-based real-time head tracking and eye state recognition technology has some difficulties. Lighting conditions may change while controlling the mouse, people may have varied face forms, and eyes may have varying forms and sizes depending on where they came from. To increase the performance of the intended head mouse control system, many approaches have been used. It is important to recognize the movement of the head and the states of the eye click and make an accurate decision. In this paper, to solve these problems, deep learning based on convolutional neural networks is presented to improve system performance. Here, the basic problems are extracting important features of the head and eyes and identifying their states to make an accurate decision. Traditional approaches to solving such problems are based on using feature extraction methods and feature classification methods to classify the extracted features. One of the fundamental advantages of Deep Learning structures is the ability to perform both feature extraction and classification and combine their results into

one image. Deep learning is an efficient technology for learning large amounts of data with high accuracy. This enables the construction of a compact system with high accuracy. The goal of this project is to create a head-mouse control system that uses image processing techniques and Convolutional Neural Networks (CNN) to improve head position and eye state recognition capabilities.

CNN is a class of deep learning models which is an excellent method for image recognition. CNN can automatically detect the important features of the image without any human supervision. CNN can efficiently extract distinctive features from the image sets which is very important in recognition. CNN model combines feature extraction and classification in the unified body of CNN, which allows simplifying the structure of the used image recognition system. These properties allow the designing of CNN model with the highest accuracy. CNN uses convolution and pooling operations for sharing parameters which enables CNN models to execute on any computer and this capability makes it computationally efficient. In this thesis, CNN architecture is considered for the head positions detection as well as recognition of eye states.

1.1. Definition of Problem

In this thesis, we propose a head-mouse control system for disabled people with spinal cord injuries. The developed mouse control is based on head movements and eye blinks. The user moves the mouse cursor to the required coordinates using head motions and transmits a command with eye blinks in the suggested system. The proposed head-mouse control is based on image processing, which includes facial identification, particularly of the eyes, lips, and nose.

However, the existing methods have several limitations. Some of them require special hardware, such as certain cameras or sensor-based devices. Some have additional components and the user has to wear these components. The other classical approach uses special methods to solve feature extraction and classification problems. The proposed framework based on Convolutional Neural Networks (CNN) utilizing inferior quality pictures caught by a PC's camera. The CNN comprises of fully connected layers, pooling layers and convolutional layers in the network structure. The CNN changes the head motion and eye states into the mouse movements and mouse button commands.

In this thesis, an implicit PC camera is utilized to capture the image of the individual. The captured image is segmented in order to perceive the head-shoulder profile of the person. In the following stage, the segmentation of the head-shoulder and the recognition of the head profile and eyes of an individual are performed. The identified pictures of the head profile and eyes are contributions to the CNNs. These images are inputs for CNN1 and CNN2. The main CNN1 is utilized for identifying the bearings of the head profile which can be up, down, left, right and no activity. The second CNN2 is utilized for recognizing the eyes' states, which can be shut (alright) and open (No-activity). The CNN1 and CNN2 networks' outputs are transformed to the mouse control signals.

1.2. Thesis Overview

The thesis includes an Introduction, five chapters and Conclusions.

Chapter 2 presents a state of the art of deep learning for the design of head-mouse control systems. A review on head mouse control is given. The state of the problem is presented.

Chapter 3 the structure of the proposed head mouse control based on CNN is presented. The basic elements of the designed model are explained. The object detection, feature extraction and classification stages are presented.

Chapter 4 presents the Convolutional Neural Networks used for designing head mouse control. The CNNs used for the identification of head movement and eye state detection are explained. The learning algorithm of the proposed structure has been described.

Chapter 5 gives a simulation of the head mouse control system. The modelling of the CNN based system for identification of head movement and eye state detection are presented. The stages of the algorithms are described. The performance of the designed system is compared with the performances of other models.

Finally, **Chapter 6** gives a conclusion of the thesis. The important simulation results were presented. Future recommendations of this research study have been given.

CHAPTER 2

STATE OF ART OF DEEP LEARNING FOR THE DESIGN OF HEAD-MOUSE CONTROL SYSTEM

1.1. Overview

A review and analysis of existing research studies used for head mouse control were described. The basic approaches used in mouse control for disabled people were considered. The state of the art of deep learning used in head motion identification and eye state detection is presented. The advantage of using convolutional neural networks in head movement identification and also eye click state recognition is described. The status of the research problem is given.

1.2. Review of Existing Research Works Used for Head-Mouse Control

Recently, some research studies that are based on head-controlled mouse systems have been published in different literature. Chen et al. (2003) proposed developing a head motion-controlled computer mouse system for spinal cord damage patients using image processing and microprocessor technologies (SCI). This system was created to control the movement and direction of the mouse pointer by using a marker mounted on the user's headset to capture head motion images. In comparison to the infrared operated mouse technology, they improved input speed, but users must wear the headset. Pereira et al. (2009) created a computer mouse control system for people with impairments. A video camera, computer software, and a target mounted to the front part of a cap worn by the user make up this system. The proposed system is simple to use and emulates the motion features of a computer cursor. Chen (2001) created a head-controlled computer mouse to help persons with disabilities live more independently. To determine head position, this device uses two tilt sensors in the headgear. The left-right movement is controlled by one tilt sensor, while the up-and-down movement is controlled by the second tilt sensor. A touch-switch gadget was intended to delicately contact the user's cheek to perform mouse clicks. Fouché (2017) considers the plan of head-controlled cursor control and decides if the outcomes accomplished by clients with handicaps are practically identical to those accomplished by clients without incapacities. The paper considers the ideal affectability

setting, the measure of neck weariness, learnability and the degree of client fulfilment are explored. The framework is tried on a PC round of the shooter class. For clients without incapacities, a decrease in neck weakness and a huge improvement over the long haul for specific things were found. Mishra et al (2017) introduced a human-machine interface for handicapped individuals who can't utilize their hands to control PCs. The mouse is constrained by head developments and an air sway sensor. The double-pivot accelerometer-based slant sensor is utilized to identify the development of the head. Two air bubble sensors set close to the mouth are utilized to initiate the left and right-snap of the mouse. The framework is expected to be utilized freely by incapacitated people. A gyro-mouse carrying out a human-PC interface has been created to control the mouse for crippled individuals (Eom et al., 2007). The exhibition of the framework executing the development and snap activities was assessed by click identification rate, cursor position control blunder, and snap rate each moment. A wearable head GPS beacon that can go about as a head mouse, in light of electronic parts and inertial sensors, was planned by Sim et al. (2013). The elective PC mouse was created by Gerdman et al. (2012) for engine hindered individuals. The mouse has a whirligig as a movement sensor which is excessively touchy. The planned mouse is utilized by crippled individuals to control their PC. Ruler et al. (2005) and Nguyen et al. (2006) utilized two-hub accelerometers and Neural Organizations planned a head movement grouping framework. A two-hub accelerometer was utilized to gather head movement information, Neural Organizations was utilized for grouping. Kim et al (2010) planned a sans hands mouse utilizing the sign from gyro sensors that action the precise speed of head pivots. Opto-sensors were utilized to recognize eye squints for mouse click control. All sensor-based frameworks show great outcomes and help individuals with inabilities, however we don't uphold individuals with handicaps to discover answers for issues with the extra gadgets. Arai and Mardiyanto (2010) introduced a camera-mouse framework for individuals with inabilities. In the introduced framework, the clock is utilized as a left-click occasion and the squinting is utilized as a right-click occasion. In the analyses, composing activity, left-click occasions, right-click occasions, intuitive occasions, and troubleshooting were shown with the camera-mouse framework. Su et al. (2005) introduced a visual-based PC interface for individuals with incapacities. Head developments are utilized to move the mouse to the new position. The creators recommended stay time and the spring up menu for tapping the

mouse. A head movement identification strategy is proposed by Tolle and Arai (2016). The introduced framework can decide to head present developments and is reasonable for ongoing human-PC connection. Alhamzawi (2018) introduced mouse pointer control by moving the top of a client situated before the PC. The reference point on the head, which is the focal point of the client's head, is caught by the camera. The pictures addressing the situation of the head are changed over to the directions of the mouse pointer in the wake of handling. The mouse click is changed over into seconds by standing firm on the pointer at the ideal situation. Naizhong (2015) introduced a human-PC interface that executes sans hands mouse control dependent on mouth following. The researchers utilized mouth following for mouse development and head shaking for mouse click activity. Palleja et al. (2008) introduced an overall virtual mouse dependent on head development understanding utilizing the camera for individuals with portability disabilities. Lin et al. (2007) planned an eye-GPS beacon with a head signal. Here, the client wears a gadget with a light source. The CCD camera catches the picture of the client and afterwards, the situation of the light source is controlled by an advanced picture preparing method. The user can manipulate the mouse by moving the head. Eye-following is additionally utilized in the work for mouse control. Ismail et al (2011) introduced a model framework for controlling the mouse by head development and furthermore by voice order. The voice order is utilized for a mouse click. In (Sawicki and Kowalczyk, 2018), head development is utilized to plan a touchless PC control framework. The camera on the head is utilized to dissect the situation of the screen cursor. Thus, the creators viably move the mouse cursor to the ideal position recognized by the client's facial direction. The investigation of the eye picture is performed and by squinting the framework orders were executed. The head-mounted inertial interface is produced for individuals with cerebral paralysis (Velasco et al., 2017). An empirical model of human motor performance for directed movements is presented using Fitts' law. Varona et al. (2008) introduced a vision-based user interface for those with motor disabilities to make computers more accessible. The system tracks the user's face in real-time to recognize motions. Manresa-Yee et al. identify and evaluate the important aspects of camera-based head-controlled interfaces (2014). Users features to track, initial user detection, position mapping, error recovery, profiles, and system ergonomics are all examples of these elements. The paper looks at what other systems have to offer in terms of solutions.

The appropriate recognition of a human's head is critical in human-computer interaction because it influences the accuracy of the entire system. A variety of studies have been carried out for this purpose. Jian-Zheng and Zheng (2011) described a strategy for tracking head movement using image processing techniques. To determine the feature point and tracking pattern of head motion, the Lucas-Kanade (LK) algorithm is employed, and the GentleBoost algorithm is used to detect the head direction based on the eye. For head motion detection, Zhao et al (2012) employ image processing and the LK method. Mehrubeoglu (2011) detects and tracks an eye using an image processing algorithm and a smart camera. The continuous head following and eye status framework utilizing a camcorder has a few difficulties. During the control, lighting conditions may change, individuals may have diverse face shapes, and eyes may likewise have distinctive shape and size contingent upon their starting point. In this paper, we intend to improve the presentation of a head position and eye status recognition by utilizing picture handling procedures and Convolutional Neural Organizations. In this paper, we proposed a head-mouse control framework that executes head position assessment, eye recognition and grouping. At that point, we applied the three-layer Convolutional Neural Networks (CNN) to select the state of the eyes that may be open or close. CNN is a profound learning structure roused by the regular discernment components of living creatures. As of late, various exploration papers have been distributed on the improvement of profound neural designs. One of them is CNN, which has at least one convolutional layers and max-pooling layers. The CNN engineering was proposed by LeCun et al. in 1998, it was a seven-layer convolutional network called "LeNet-5" that arranges written by hand numbers in 32x32 pixel pictures. The organization was prepared by utilizing the backpropagation calculation. The framework can perceive the pictures straightforwardly from the picture pixels (Lecun et al., 1998; Hecht-Nielsen, 1992). Reference (Russakovsky et al., 2015) presents a CNN engineering that showed upgrades in picture acknowledgement. Russakovsky et al. (2015) planned a profound design called AlexNet, which is like LeNet with a more profound construction. Many exploration works have been done to improve the exhibition of the frameworks and beat the challenges identified with preparing the CNN, like ZFNet (Zeiler and Fergus, 2014), VGGNet (Simonyan and Zisserman, 2015), GoogleNet (Szegedy et al., 2015), and ResNet (He et al., 2016). This work is reached out in more profundity to more readily inexact complex nonlinear capacities by applying nonlinear enactment capacities

and to get better component portrayals. To this end, a few strategies are created to manage these issues.

Different CNN structures were designed to solve different problems related to image recognition, classification and so on. Lawrence et al., 1997 presented CNN used for face recognition, Ciresan et al., 2001 developed a system for the classification of handwritten characters. Simard et al., 2003 presented a system for visual document analysis, Jiao et al., 2018 presented facial sketch analyses, Chudzik et al., 2018 designed a system for detection of microaneurysm, Li et al., 2018 presented fingerprint enhancement, Hussain et al., 2018 presented brain glioma tumour segmentation, Baldominos et al., 2018 presented recognition of handwritten text, Ferreira and Giraldi, 2017 developed rock tile characterization, Wachinger et al., 2018 presented neuroanatomy segmentation, Liu et al., 2018 presented change identification with heterogeneous optical and radar pictures, Liu et al., 2018 presented prediction of eye states, Salvati et al., 2018 designed acoustic source limitation improvement in boisterous and reverberant conditions, Abiyev and Ma'aitah, 2018 presented a system for detection of breast diseases. Kalchbrenner et al., 2014 presented natural language processing, Khodayar et al., 2017 presented a short-term prediction system for wind speed, Karpathy et al., 2014 presented a system for image and video recognition.

2.2. Deep Learning Based on CNN Architectures

The most popular CNN architectures were presented for solving different problems. LeNet [31] was the first successful applications of CNNs and used to read postal codes, digits, etc.

AlexNet [30]. The first well-known CNNs used in computer vision was the AlexNet. AlexNet was appeared at the ImageNet ILSVRC 1 challenge in 2012 and took first place with a top 5, error of 16% compared to second place with 26% error. The network had multiple convolutional layers stacked on top of each other without the interruption of pooling layers.

GoogLeNet [49]. GoogLeNet was the winner of ILSVRC 2014. Its main contribution was the development of an Inception Module. The inception module uses convolutional filters of different sizes and concatenates their outputs so that the model can decide which filter

size is best to capture features. It also uses Average Pooling instead of Fully Connected layers in the final network layers, eliminating a large number of parameters that arise in Fully Connected layers.

VGGNet [43]. The VGGNet was a very depth network that appeared at the ILSVRC 2014. It showed the importance of network depth. The network contains 16 layers (without pooling) and performs only 3x3 convolutions and 2x2 pooling. However, VGGNet is quite expensive to evaluate and consumes a lot of memory for its 140 million parameters.

ResNet [20]. Residual Network was the winner of ILSVRC 2015, featuring special residual blocks that improve the optimization of the cost function, and strong use of stack normalization. The structure also discarded the pattern of using fully connected layers at the end of the network.

Squeeze-and- Excitation Network [22] was the winner of ImageNet 2017, introducing the mechanism of the adaptive weighting of intermediate feature map channels in the hidden layers of the network, increasing the network's ability to learn the best representations of the input.

As can be seen from this list, the field of CNNs is changing rapidly. The best improvements to the CNN architecture are often based on simple ideas but are very effective.

2.3.State of Deep Learning Methods Used for the Design of Head-Mouse Control

A comparison of feature extraction and picture recognition methods reveals that deep learning-based CNN is an effective method for detecting head motions and recognizing eye states. The CNN-based approach for estimating head posture and detecting eye state in visual mouse control will be aimed at people with disabilities. In comparison to other image classification algorithms, CNN can intelligently restrict the architecture and requires fewer preprocessing steps. Based on the estimation of head positions, detection of eye states

and convolutional neural networks, the design of a system enabling mouse control with head motions using low-quality images acquired by a monocular camera mounted on a computer is considered (CNN). A video camera and a high-speed computer are required for the system to work. The goal of this module is to track the position of the face and the state of the eyes. The direction of movement of the mouse cursor is implemented by tracking the facial position. The eye state module will be utilized to examine the video pictures utilizing picture preparing strategies and afterwards, identifies the eyes and characterizes them as "open" or "shut" state in video pictures. The exhibitions of these modules are vital for the acknowledgement of head position, eyes and their states. These tasks ought to be acted in a brief time frame to control the mouse with head developments and eye states. The head-mouse framework ought to be utilized for individuals with disabilities to serenely utilize PCs and control objects.

Taking the above mentioned into account, in order to achieve the research goals, the design of a CNN-based head mouse control system necessitates the execution of the following procedures.

- The structure of the head mouse control system using CNN will be designed. The basic stated problems will be specified.
- The mathematical background of Convolutional Neural Networks (CNN) will be presented. The basic constituents of CNN, which are convolutional layers, a pooling layer and a fully connected network will be presented.
- The structure of CNN models will be designed for the considered problems, particular for the detection of head movement and recognition of eye states.
- The simulation of the head-mouse control system will be implemented using constructed image data sets. For this purpose, the data sets will be organized for each considered problems, i.e. head movements and eye states.
- Analysis of obtained results will be carried out. Discussion and performance analysis of the results will be performed.

Design of system using CNN will improve the performance of the head mouse control system

CHAPTER 3

HEAD MOUSE CONTROL SYSTEM

3.1. Overview

For physically challenged patients with motor neuron disease or severe cerebral palsy, the head-mouse control system was created. The technology calculates the head's position and converts it to mouse positions. In this section, the structure of the system is designed. The functions of its main elements are explained. The object recognition problem and its solution methods are presented in detail. The steps and details of feature extraction are explained, the use of CNN to solve this problem is described. The classification unit is described in detail. In the paper, CNN is proposed for solving the above problems.

3.2. Structure of the System

The considered head-mouse control system is designated for physically disabled people having motor neuron disease or severe cerebral palsy. The technology calculates the head's location and transforms it into mouse movements. The presented system also detects and analyzes eye conditions, which it then uses to operate mouse buttons. The architecture of the designed system is given in Figure 3.1. The system includes a set of modules for the acquisition of images, recognition of objects, extraction and segmentation of features, classification, conversion of the output signal to the control signal for moving the mouse to the indicated position and finally converting eye blink signal to the mouse on/off signal. As shown in the figure, the person's image is acquired by the computer camera and is transformed and integrated into the system. The camera's image is divided in the object recognition block, and the user's head-shoulder profile is detected. The segmentation of the head-shoulder image, as well as the recognition of a person's head profile and eye states, are conducted in the next stage. The CNNs that perform classification use separately the images of head profile and eyes as inputs. These input images are fed into the first and second CNN blocks depicted as CNN1 and CNN2. The first CNN1 is utilized to detect the directions such as up, down, left, right and no-action of the head-profile. The second CNN2 is utilized to detect the closed and open states of the eyes. Here closed state means

clicking Ok button, open state continuation of the action, that is no-action of Ok button. CNN1 and CNN2 network outputs are translated to the control signals of mouse and the forwarded for determination of appropriate actions.

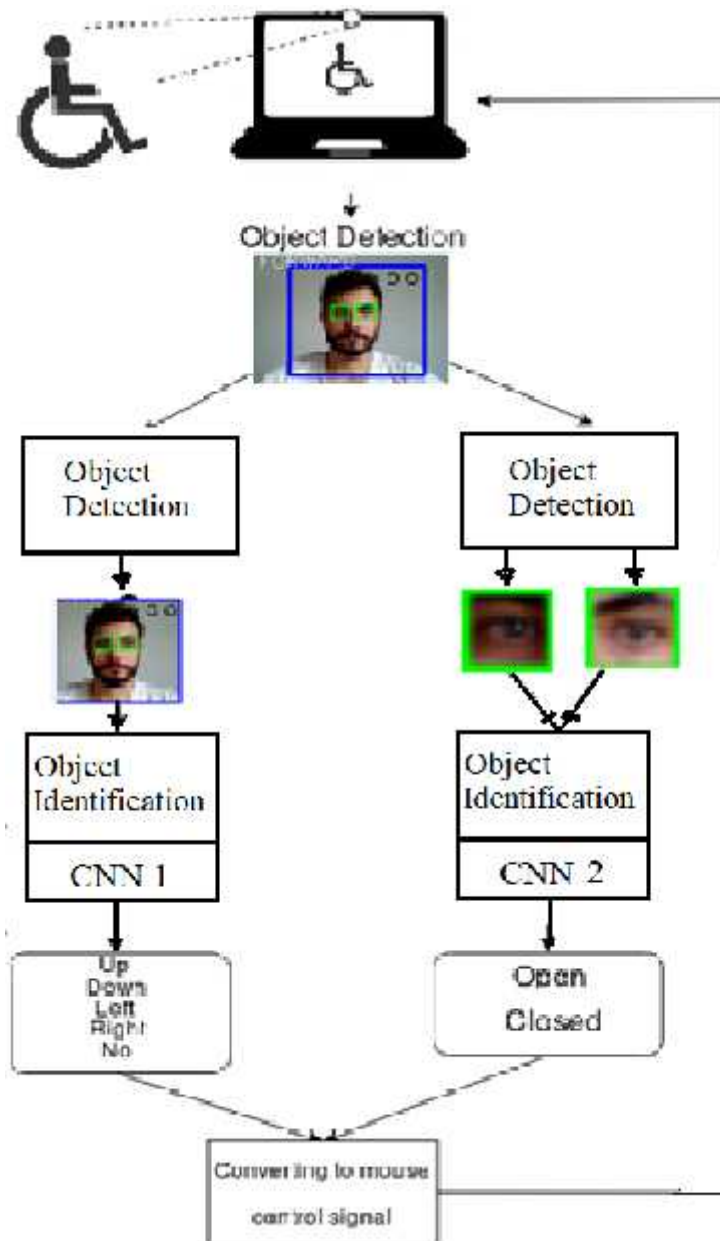


Figure 3.1. The architecture of the head mouse control system

3.3. Object Detection

Object recognition, which is utilized to recognize the head profile and eyes, is one of the essential building modules of the head-mouse control system. In this thesis, we use the hair

cascade classifier approach for object recognition described by Viola and Jones (2001) for this purpose. Simple features are used to achieve object categorization in this technique. The features can encode ad hoc knowledge domains and operate considerably more quickly than a pixel-based system. The features are similar to those found in the Haar basis feature. The total of pixels in the rectangles is the features that are being looked for (Figure 3.2). The set of rectangle characteristics provide a rich visual representation of images and allows the system to be learned effectively. As shown in Figure 2, the value of any input feature is the sum of the pixels taken from the pointed (black) rectangles and the pixels in the clear (white) rectangles. The cascade function is trained using negative and positive images before being used to detect the other objects on images. Those featuring a face are considered positive, whereas images without a face are considered negative. The classification function is trained using the feature and training sets of negative and positive images.

All human faces have comparable traits, which can be used to create hairstyles. The eyelids, for instance, are darker than the top cheekbones. Using the equation, we can determine the gradient value based on the histogram:

$$\text{Value} = (\text{pixels of black region}) - (\text{pixels of white region}) \quad (1).$$

We utilized a variation of Adaboost to extract useful features from photos. The algorithm determines the proper threshold for each feature in order to classify the faces. At first, the weights of each image are equal. New weights and errors are computed and they are updated in each step until the error rate and detection accuracy are satisfactory.

The input space of CNNs is the detected pictures of the head and eyes. These images are used by CNNs to create classes, which are then utilized to determine mouse control signals.

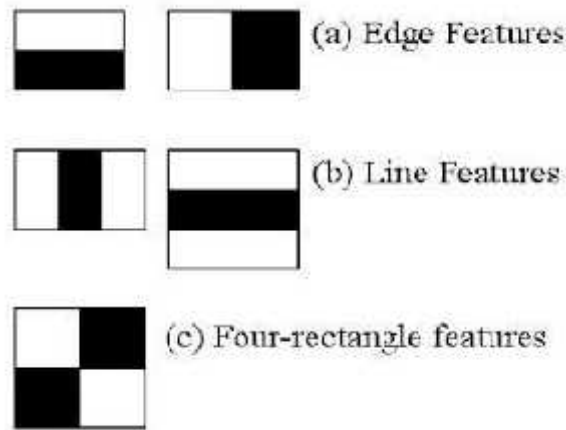


Figure 3.2. Cascade classifier approach

3.4. Feature Extraction

Each image object is characterized by a set of features. The detection and obtaining of these features are important steps in image recognition. In the thesis, the convolutional layers of CNN are employed for the detection of these features. The output signal of the object detection module is the input for CNN. The convolution is mathematically based on integral transform operation on functions that includes input signals using a particular operator. It uses the original function and transforms or reshapes it in order to get new representations. In image processing, convolution is started widely used to detect and extract features without removing the spatial relationships between image pixels. Using an appropriate operator searches for a particular feature in a much large set of pixels. A convolution operation is mathematically constructed by first inverting the operator, called the convolution kernel, both row-wise and column-wise. The convolution kernel filter is shifted by a fixed step over the entire original image. Here the kernel size should be less than the image area. For each operation, the result of convolution will be the integration of original elements weighted by the corresponding inverted kernel. The convolution is used as an effective technique for extraction of the features that cleverly reduces data dimensions and produces a less redundant dataset. The extracted set of features are called a feature map. Each kernel filters out or determines the position of the features in the original image. Ultimately, a map is generated whose height reveals the distribution of these features. Since the direct analysis of the original data requires a lot of preprocessing operations, and can deconstruct the symbolic information of the images hardly, the use of

general feature description methods is necessary. This feature extraction is a convolution. Convolution aims automatically extract important features and make them “visible” to the system. The use of convolution not only fully describes the features of the objects, but also reduces the need for manual intervention. The kernels have lower-order and smaller in size than the original images. They extract local features of the input objects. However, higher-order convolutions allow for extensions throughout the receptive field, gradually transforming local features into global features, which is also the case of how humans look at and recognize an object. In addition, the parameters of kernels are constant during the recognition. This leads to the weight-sharing property of the individual receptor neurons, which results in large computation time savings. Moreover, pooling further samples the features and reduces the computational operations for the computer

3.5. Classification

A set of methods have been designed for the classification of various objects. Neural Networks that are composed of large numbers of dense neurons is one of the adaptive classification techniques with the learning ability. Neural Networks learn its parameters using examples. NN main goal is to replicate the performance of biological neural systems, learn from the way humans recognize the world and thus use the network structure to solve complex problems such as image recognition or object classification. NN includes a set of connected neurons characterized set of weight coefficients. These weight coefficients modify coming input signals. These signals are modified by the nonlinear activation function of neurons. The basic problem in NN classification is finding proper values of these parameters. Different learning algorithms are employed for this purpose. By updating these layer parameters, NN tries to classify input images into the output signal. By selecting appropriate values of weight coefficients of the NN classification model is designed. The learning ability, generalization and parallel processing property allows to develop a high performance classification model.

Figure 3.3 depicts a neural network structure consisting of a set of neurons. As shown the network includes three layers, that are input, hidden and output layers. Each layer includes a set of connected neurons. There are connections between neurons of input, hidden and output layers. The strengths of each connection are presented by weight coefficients. Neurons of the input layer receive signals and distribute them between neurons of the

hidden layer. In hidden neurons, these signals are processed, modified by the weight parameters and activated. After activation, they are transmitted to the neurons of output layers. The same processing and activation operations are used for the output signal of hidden neurons.

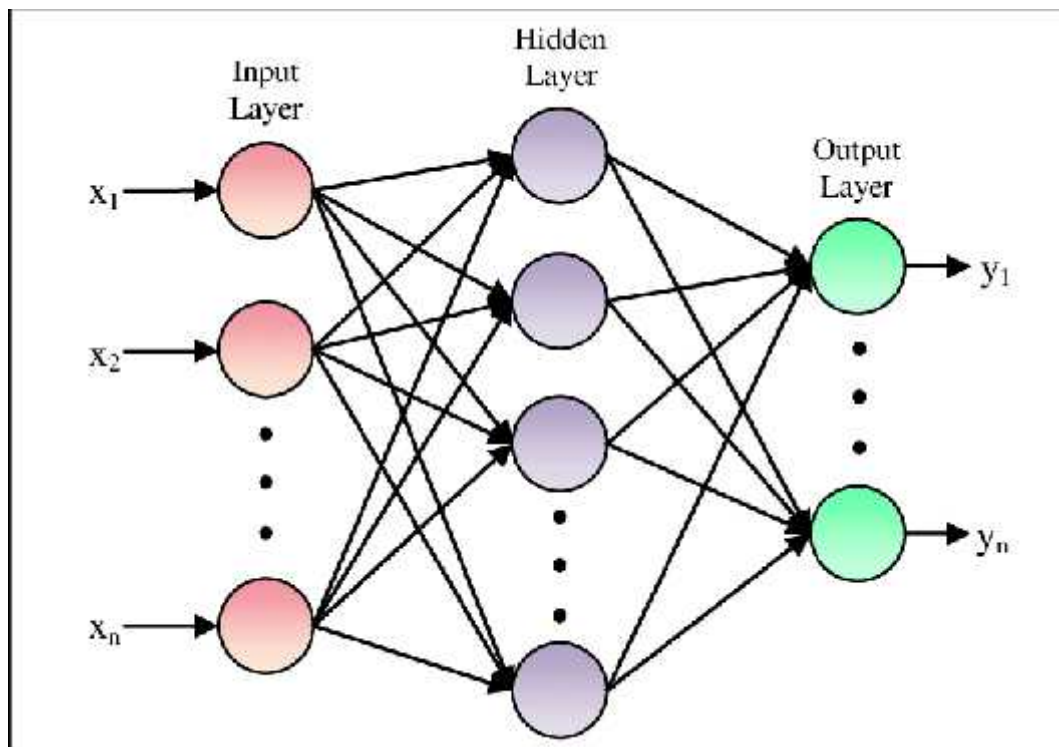


Figure 3.3.A simple multi-layer neural network structure

Finally, on the output of neurons of the output layer, the output signals are determined. The considered NN is a feedforward structure. CNN is also a feedforward structure that includes multiple layers and the CNN structures were designed for solving image processing problems. It is applied for processing one-, two and three- dimensional data. A basic neuron of a CNN collects the activations of kernel filters and generates a feature map. CNNs provide satisfactory recognition results for practical use, but the accumulation of layers also creates a black box of the feature extraction process.

CHAPTER 4

DEEP LEARNING BASED CONVOLUTIONAL NEURAL NETWORK FOR HEAD MOUSE CONTROL

4.1. Overview

This chapter introduces the design of the classification system based on Deep Learning which is a branch of Machine Learning (ML). The basic problem of Machine Learning is the development of algorithms that use statistical methods and data to learn computer systems [25]. It covers topics essential for understanding complex learning systems such as neural networks. Deep learning based on learning and statistics allows the design of a more efficient decision-making system. In this section, deep learning based on CNN structure will be described. Structure and constituent elements of CNN, its operation principles will be presented.

4.2. Machine Learning Systems

In ML, the goal is to create a system that can perform a particular task without being explicitly programmed [41]. Instead, it should suffice to provide such a system with empirical data of the process whose behaviour is to be studied. Formally, this can be formulated as follows [50]:

"A computer program is said to learn by experience E with respect to a class of tasks T and a performance measure P if its performance on tasks in T , as measured by P , improves with experience E ."

This definition summarizes all the important aspects of an ML problem.

T , the task - the problem that an ML system is supposed to solve. The most common tasks are regression and classification. Regression models are trained to estimate unknown continuous variables based on given inputs. In classification, an ML model learns to determine to which of k categories an object belongs.

P , the performance measure is a quantitative measure of how well the model performs. Optimizing the performance measure is one of the main goals of machine learning. Usually, the choice of metric depends on the type of task the machine learning system is

dealing with. A common practice is to use multiple metrics that give different perspectives on the model performance.

E, the experience. In most ML algorithms, experience is represented by a dataset - a collection of data points represented by real-valued vectors $x \in \mathbb{R}^m$. In general, ML problems require the collection of experience data and its transformation into values that can be accepted by the learning algorithm. This process is commonly referred to as feature engineering. Although some branches of machine learning, such as Deep Learning, deviate from this approach and use as much information as possible. For example, a grayscale image might be represented by a feature equal to the sum of the pixel values of the left part, subtracted from the sum of the pixel values of the right part. Deep Learning systems would use the entire image as input. When possible, it is convenient to represent the data set by a matrix. A dataset consisting of n m -dimensional samples forms a matrix $X \in \mathbb{R}^{n \times m}$. The i th row of X then denotes a data point x_i .

A learning algorithm is an algorithm that is able to generate models from the statistical dataset. The used learning algorithms are conditionally divided into supervised and unsupervised algorithms [25], although this distinction is sometimes too imprecise. There is also reinforcement learning, where learning algorithms interact "in real-time" with dynamic systems and receive feedback to evaluate the success of their actions [48].

In supervised learning, each data point is associated with a target value that provides information for learning and that the model tries to estimate when it receives a new unseen input. In unsupervised learning problems, there are no target values. The presence of the target value strongly affects the learning methodology. This paper focuses on algorithms for supervised learning.

4.2. Artificial Neural Networks

ANN is a parallel distributed connectionist system made up of simple processing units (commonly referred to as neurons) [19]. ANN is a model consisting of multiple computational layers stacked on top of each other. Successive layers are thought to learn features of increasing abstraction, looking "deep" into the data. At the origins of deep learning, biological neural networks have been a great source of inspiration for artificial neural network research. ANNs are an important area of deep learning research and have

been successfully applied in many fields, such as image recognition [11], speech recognition [17], natural language processing [14], and bioinformatics [55]. Two types of models can be distinguished based on their topology: Feedforward and recurrent neural networks. In feedforward networks, information flows from input to output through computational layers consisting of processing units, and there are no feedback connections. If these are present and intermediate outputs are fed back to previous layers or stored for later use, the network is said to be recurrent. Thus, a feedforward neural network can be represented by directed acyclic graphs, while recurrent networks are represented by directed cyclic graphs. In this text, only feedforward networks are discussed. This chapter covers concepts that form the foundation of ANNs, followed by multilayer perceptrons - the cornerstone of Deep Learning. The rest of the chapter describes the process of learning with neural networks.

4.3.1 Basic Concept

ANNs originated as systems inspired by biological neural networks. They were designed to simulate the brain of an animal. So, by analogy, they consist of neurons and connections between them.

Artificial neurons were designed as a mathematical model of the biological neuron. It is a function that converts n real input "stimuli" into a single "response value". The most common definition of the neuron is

$$o = f(x) = f(\sum w + b),$$

where $w \in \mathbb{R}, b \in \mathbb{R}$ are the neuron's weights and biases respectively, and f is the activation function described below. Weights and bias are the parameters of the neuron, and their values are determined in the learning process. Bias makes this transformation affine, which allows us to learn a wider variety of functions.

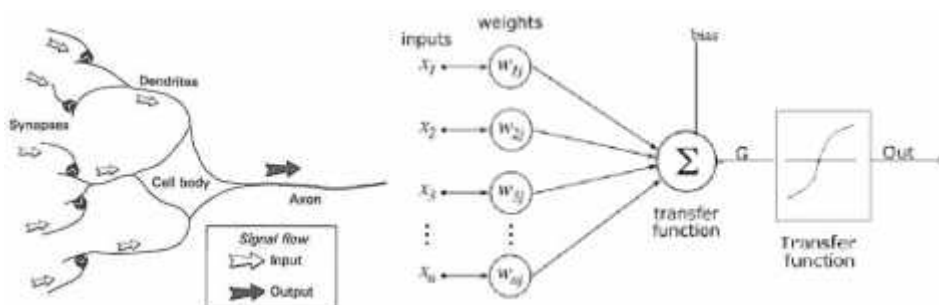


Figure 4.1: A biological neuron (a) compared to an artificial neuron (b). The analogy is clear: both neurons receive multiple signals and aggregate them to a single output that is transferred further on [18].

The activation function transforms the output of a neuron. It is applied element-wise on the output vector. The purpose of an activation function is to introduce non-linearity to the model in order to learn complex nonlinear functions. Without activation functions, hypothesis space of a neural network would be constituted only from linear functions. Popular examples of activation function include **ReLU** (Rectified Linear Unit) $f(x) = x^+ = \max(0, x)$ and **sigmoid** $f(x) = \frac{1}{1 + e^{-x}}$

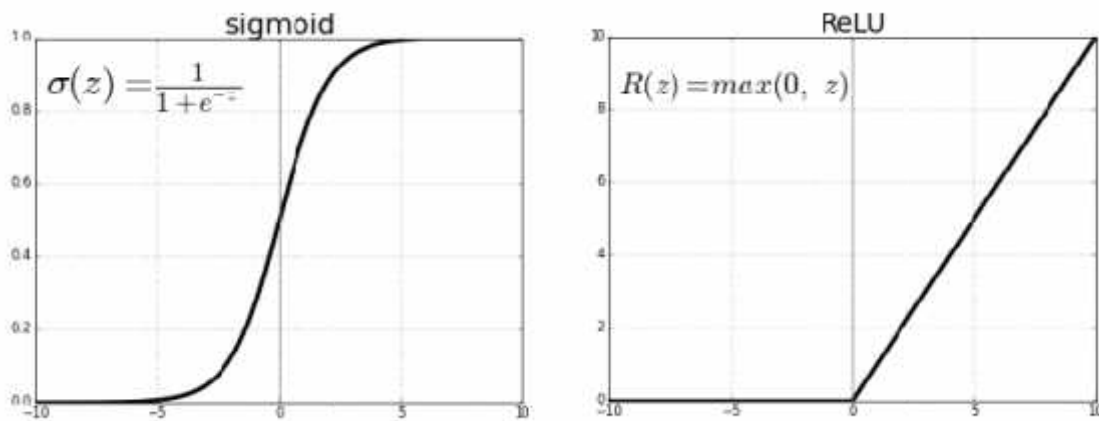


Figure 4.2. Popular activation functions

The layer is a collection of neurons that are grouped to process multidimensional inputs to multi-dimensional outputs. ANNs are usually described in terms of layers since it is more convenient due to the bigger scale and abstraction compared to artificial neurons. A notable example of a layer is a fullyconnected layer. In fullyconnected layer neurons receive input from each neuron of the previous layer. This allows for convenient simultaneous computation of all neuron outputs of the layer via matrix multiplication:

$$f(x) = f(d\mathbf{w} + b)$$

where \mathbf{W} and \mathbf{b} are individual neuron's weights and biases collected into a matrix and a vector respectively. Network topology is a configuration of multiple chained connected layers. Two special layers are always present in the network: input and output ones. An

input layer holds values of the input vector and is connected to the next layer only to pass it forward. An output layer is the last layer in the chain and its outputs are the final result of the input processing done by the network. Layers between them are called hidden because their outputs are only intermediate and are not used outside of the model. The number of hidden layers is called the network's depth. The number of neurons in the layer is referred to as its width.

Signal directions in NN can be feedforward and feedback. In a feedforward neural network (Figure 4.3) the connections between the nodes do not form a cycle. These NNs have feedforward connections only. The information moves from input nodes to output nodes.

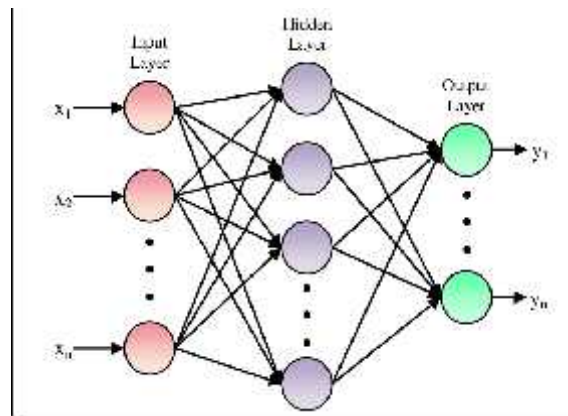


Figure 4.3.A feedforward network.

Feedforward NNs may be single-layer and multi-layer architectures. Single-layer consists of one artificial neuron. It can be extended with additional units for each output element to produce multi-valued outputs, but it would still have no hidden layers. This type of neural network has very limited learning potential. Due to the lack of hidden layers, the outputs are only a linear combination of the inputs, so the single-layer perceptron is only able to approximate linear relationships [35].

$$y = \varphi \left(\sum_{i=1}^n w_i x_i + b \right) = \varphi(w^T x + b)$$

where w denotes the vector of weights, x is the vector of inputs, b is the bias and φ is the non-linear activation function.

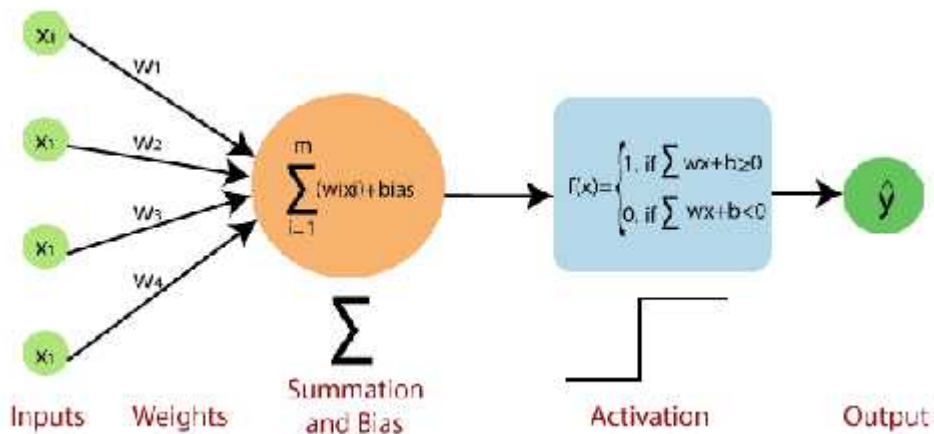


Figure 4.4. Single-layer Perceptrons can learn only linearly separable patterns. For classification, we use the Activation function as a threshold to predict class. And for Regression, we need not need the Activation function (Thresholding) or we can use a linear function to predict continuous value.

4.3.2 Multilayer NNs

The multilayer NNs sometimes are called multilayer perceptron (MLP). It has at least one hidden layer and all layers are fully connected. MLP is the fundamental tool and the basis of many advanced techniques in DL. It is a cornerstone of Deep Learning.

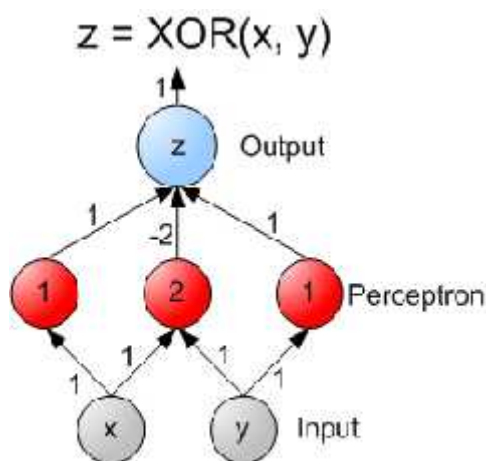


Figure 4.5. A multilayer perceptron with a single hidden layer.

Mathematically, MLP is a composite function that performs alternating affine and non-linear transformations to reflect the graphical structure of the network []. Figure 3.3 depicts a multilayer perceptron with a single hidden layer.

MLPs can be used in a wide range of tasks - according to the universal approximation theorem [21], an MLP with at least one hidden layer and nonlinearity, containing a finite number of units, can approximate any continuous function with an arbitrary nonzero error, given suitable parameters. However, the theorem is not constructive and does not provide a way to find such a network configuration and its parameters. In practice, optimization algorithms are not able to find parameters that correspond to the desired function. This means that designing and training a network may not result in the desired approximation quality. Nevertheless, this is an important result that shows the great potential of ANN.

Multiple Hidden layers are used to find the nonlinearity of the data. This instruction is also called a feed-forward network.

Multilayer perceptrons can be trained using input-output training pairs. Training operation includes adjusting the weights and biases of the network in order to minimize error. In practice, the backpropagation algorithm is widely used for this purpose.

In practice, MLP has a few problems that outweigh its appeal as a universal approximator. One of the major drawbacks of this model is the high number of trainable parameters due to the fact that all layers are fully connected. The number of parameters may become impractical to train for high dimensional data or wide hidden layers. It also limits the depth of the model. Deep and wide perceptrons are not only computationally difficult to train, but due to their high capacity, they also tend to overfit if the dataset is not large enough.

3.2.ANN Learning

The weights of NN are adjusted during the learning process. The weights of NN are trained using statistical data existed about considered problems. These data are called training input-output pairs. Training of networks is implemented using input-output training pairs.

NN has a set of parameters that affect the learning process. Sometimes these parameters are called hyperparameters. Learning rate, number of hidden layers, number of parameters, stack size are included hyperparameters [46]. Learning rate is used to define the size of correction of weights. The large value increases learning time or speed but leads to

oscillation of learning process, small value lead to decrease learning time. The optimal value of the learning rate speed up the learning of NN [47]. The momentum rate is also used to speed up and stabilize the learning process of the network.

The cost function is formulated using errors in order to organize the learning process of the NN model and to measure its performance. In the learning process, the measured error on the output of NN is propagated back and used for the correction of weight coefficients. This process is called Backpropagation. The values of error are used to correct weight coefficients [48-52].

Learning can be organised as supervised, unsupervised and reinforcement learning. It's called supervised learning the training dataset as a teacher supervising the learning process. Here input-output training pairs are used for training. If learning is carried out using input data only this learning process is called unsupervised. Only input data (X) is used for weight corrections. In reinforcement learning, the goal is to weigh (strategize) the network to perform actions that minimize the long-term (expected cumulative) cost. During learning, if the error is minimized then +1 sent to the system in another case 0 sent to the system.

A number of learning algorithms have been designed. More used are the gradient descent algorithm and backpropagation.

4.4.1 Gradient Descent Method

The optimization problem of training a neural network can be stated formally:

$$\text{Minimize } L(\theta, x, y), \theta \in \Theta, (x, y) \in D \quad (3.4)$$

where D denotes the set of training samples, θ denotes all parameters of the model in the parameter space Θ , and L is the real loss function. The evaluation of L naturally involves the computation of the model performance for each training sample $x \in D$. In the case of ANNs, this makes the power function nonlinear, leading to complex and often non-convex optimization problems. Such problems can be solved with iterative gradient-based algorithms.

The gradient of a real continuous function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is the column vector of its partial derivatives:

$$\theta = \theta - \alpha \frac{\partial L(\theta, x^{(i)}, y^{(i)})}{\partial \theta} \quad (3.5)$$

The pseudocodes of stochastic gradient decent method is preseted as

```

for i in range (epochs ):
    np.random.shuffle (data )
    for example in data :
        params_grad = evaluate_gradient (loss_function , example , params)
        params = params - learning_rate * params_grad

```

4.4.2 Back - propagation Algorithm

Any type of gradient descent algorithm requires multiple computations of the gradient. Evaluating the gradient of composite functions can be computationally intensive, and a loss function can potentially be quite complex since it involves evaluating the entire neural network on many data samples. The back-propagation algorithm (BP) provides an efficient way to evaluate the gradient [33].

BP uses the chain rule to compute the loss function. The algorithm can be considered as a specialized version of automatic differentiation by reverse accumulation [13]. Reverse accumulation traverses the expression of the chain rule from the outside in, computing all

Neural networks can be represented as a computational graph of back propagation. This is designed to visit each node of the network once, avoiding repetitive computations that arise when evaluating the gradient of a composite function. The computational effort required for BP scales linearly with the size of the computational graph of the network.

Back-propagation always starts with a forward pass that computes the error for a training sample. It must keep all intermediate computation results as they are used to compute the gradient. As soon as the error is evaluated, the backward pass begins. It is so-called because it traverses from the output of the network to the inputs. From the point of view of symbolic computations, the expression of the chain rule is traversed from the outermost subexpression to the inner ones.

Learning happens in two ways, Forward propagation and backward propagation

1. **Forward Propagation:** In the *forward pass*, the direction of signals from input nodes to output nodes through the hidden neurons. The value of output error will be calculated against ground truth and true label.
2. **Back Propagation:** Below are the steps mentioning how the back prop works.

```

Require:  $d$ , network depth
Require:  $\mathbf{W}_i, i \in 1, \dots, d$ , individual layer weights
Require:  $\mathbf{b}_i, i \in 1, \dots, d$ , individual layer biases
Require:  $\mathbf{x}$ , the input vector
Require:  $\mathbf{y}$ , the target vector
 $\mathbf{h} \leftarrow \mathbf{x}$ 
for  $l = 1, \dots, d$  do
     $\mathbf{a}_l \leftarrow \mathbf{W}_l \mathbf{x} + \mathbf{b}_l$ 
     $\mathbf{h}_l \leftarrow \phi(\mathbf{a}_l)$ 
end for
 $\hat{\mathbf{y}} \leftarrow \mathbf{h}_d$ 

```

(a)

```

Require:  $d$ , network depth
Require:  $L(\hat{\mathbf{y}}, \mathbf{y})$ 
 $\mathbf{g} \leftarrow \nabla_{\hat{\mathbf{y}}} L$ 
for  $l = d, d-1, \dots, 1$  do
     $\mathbf{g} \leftarrow \mathbf{g} \odot \phi'(\mathbf{a}_l)$ 
     $\nabla_{\mathbf{b}_l} L = \mathbf{g} + \alpha \nabla_{\mathbf{b}_l} \Omega(\theta)$ 
     $\nabla_{\mathbf{w}_l} L = \mathbf{g} \mathbf{h}_{l-1}^T + \alpha \nabla_{\mathbf{w}_l} \Omega(\theta)$ 
     $\mathbf{g} \leftarrow \nabla_{\mathbf{h}_{l-1}} L = \mathbf{W}_l^T \mathbf{g}$ 
end for

```

Figure 4.6. Backpropagation: forward (a), backward (b) pass

4.4.3 Regularization

Besides L1 and L2 losses, the two most popular regularization techniques in ANN are dropout and early stopping.

Dropout [46] is a regularization technique specifically for neural networks. During training, network units are randomly removed along with all incoming and outgoing connections. Dropout is controlled by the hyper-parameter $p \in [0, 1]$, which indicates the probability that a neuron is kept in the network. After training, all nodes are reinserted into the network. Dropout reduces overfitting since the dropped nodes have learned from only a portion of the data.

Early stopping [4] is a regularization method that can be used in any iterative optimization algorithm. Usually, the stochastic gradient descent runs for a certain predefined number of epochs. As soon as a stopping criterion is true, the training process is stopped immediately. The stop criterion monitors the training error and indicates if its value does not change on successive iterations, which means that the algorithm is stuck in a local minimum and further parameter adjustments are unnecessary and lead to overfitting.

4.4. Convolutional Neural Networks for Head-Mouse Control

4.4.1 Basic architecture

The CNN is a deep learning structure with one or more convolutional, pooling, and feedforward layers (Figure 4.7). In MLPs, each neuron has its own weight vector; however, in CNNs, neurons share weight vectors, limiting the number of weights that may be taught. Using the weight sharing technique, neurons use convolution filters to perform convolutions on the input data. The pooling layer receives the output features from the convolutional layers. The generated feature map is activated using the activation function $f(x)=\max(0,x)$ in a ReLU layer that sits between the convolution and pooling layers. The image size is lowered after numerous convolutional and pooling layers, and more complex characteristics are extracted. The contents are then shifted into a one-dimensional vector with a tiny enough feature map, which is then supplied into the fully linked layer. The CNN's output is computed by the fully linked layer.

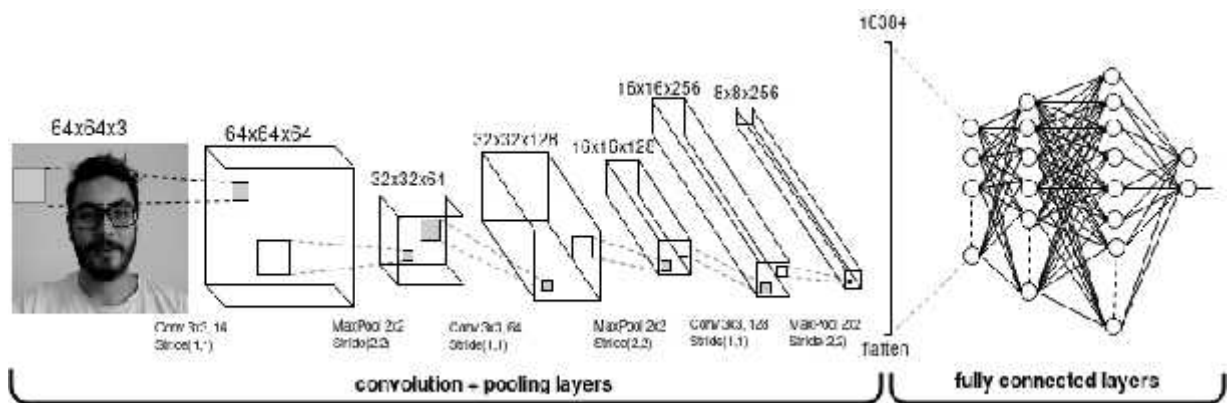


Figure 4.7. Convolutional Neural Networks

Convolutional layers are found in CNNs and are defined by an input map I , a bank of filters K , and biases b . Here $I \in R^{H \times W \times C}$, $K \in R^{k_1 \times k_2 \times C}$ for input image with height H , width W , and channels (red, blue, and green) $C = 3$ and for a bank of D filters and biases $b \in R^D$ one for each filter.

The convolutional output is:

$$(I * K)_i = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \sum_{c=1}^C K_{m,n,c} \cdot I_{i+m,j+n,c} + b \quad (2)$$

Assume that the first layer is $l = 1$ and the last layer is $l = L$, and that x is the input with $aH \times W$ dimension with iterators i and j . The iterators in the kernel ω with the $k_1 \times k_2$ dimension are m by n . $\omega_{m,n}^l$ is the weight matrix l connects the neurons of the l layer with the neurons of the $l-1$ layer. At layer l , b^l the bias unit, the convolved input vector $x_{i,j}^l$ plus bias is represented at layer l as:

$$x_{i,j}^l = \sum_m \sum_n \omega_{m,n}^l o_{i+m,j+n}^{l-1} + b^l \quad (3)$$

If $l=1$, then $o_{i,j}^{l-1}$ becomes the CNN input vector. At l layer, $o_{i,j}^l$ is the output vector while the activation function $f(\cdot)$ is given by:

$$o_{i,j}^l = f(x_{i,j}^l) \quad (4)$$

The fully connected layer's nonlinearity is incorporated into its neurons, rather than in distinct layers like the convolutional and pooling layers. The convolutional and pooling layers' output is calculated as:

$$o_{i,j}^l = p \left(f \left(\sum_m \sum_n \omega_{m,n}^l o_{i+m,j+n}^{l-1} + b^l \right) \right) \quad (5)$$

Following pooling, the flatten operation was used to concatenate the acquired features using $o_f^l = \text{flatten}(o_{i,j}^l)$. A fully-connected layer $y = F(o_f^l)$ is used to turn the given feature vector into model outputs.

The learning of weight coefficients which are unknown parameters of CNN begins after the output signals have been obtained. Let's call the CNN unknown parameters as θ . In order to determine the precise values of parameters, an appropriate loss function is created. This is accomplished by using input-output training pairs $\{(x^{(i)}, y^{(i)}); i \in [1, \dots, N]\}$ to minimize the loss function. The i -th input data is $x^{(i)}$, and the matching output goal data is $y^{(i)}$. If we represent the current output of CNN as $o^{(i)}$, we can calculate CNN's loss as follows:

$$L = \frac{1}{N} \sum_{i=1}^N l(\theta; y^{(i)}, o^{(i)}) \quad (6)$$

The loss function is minimized when the CNN is being trained. The exact values of the parameters are obtained as a result of the training. The learning algorithm Adam optimizer (adaptive moment estimation) is utilized for adjusting the unknown parameters and finding their appropriate values (Kingma & Jimmy, 2015). In this learning algorithm the first-order gradient of Loss function is used for updating the parameter values. The approach is a stochastic optimization that computes individual adaptive learning rates for different parameters using the first and second moments of the gradients (Kingma & Jimmy, 2015). Adam optimizer algorithm used in this thesis is presented in Figure 4.8.

Require: α : Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0 : Initial parameter vector
 $m_0 \leftarrow 0$ (Initialize 1st moment vector)
 $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
 $t \leftarrow 0$ (Initialize timestep)
while θ_t not converged **do**
 $t \leftarrow t + 1$
 $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
 $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
 $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
 $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
 $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
 $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ (Update parameters)
end while
return θ_t (Resulting parameters)

Figure 4.8. Adam optimizer (Kingma & Jimmy, 2015).

Deep CNN requires a large amount of training data to be trained effectively. Data augmentation is used to resolve this issue and offset the relative paucity of data compared to the amount of parameters in CNNs. Existing data is turned into new data without changing its character in data augmentation. For data augmentation, geometric transformations such as rotation, translation, shearing, zooming, and flipping are used.

4.5.2 Convolutional Layer

In CNN, convolution is used to generate a feature map from the input, which can be the original image or another feature map. The main goal of using convolution in ANNs is to exploit the special structure of the input and learn how to transform it into the most informative form.

In practice, the behaviour of the convolutional layer is controlled with a set of hyperparameters, which brings flexibility to the design of neural networks and allows them to be adapted to different problems:

The kernel size defines the dimensions of the convolutional kernel. It controls the range of input to which the neurons are sensitive. The choice of the appropriate value for this parameter almost always depends on the data set. One possibility is to set the kernel shape of the first layer according to the scale of the images to capture important details, such as edges. However, for deeper layers, there is no rule of thumb and the optimal kernel size is determined experimentally. In the case where the input contains multi-channel images or arbitrary three-dimensional data, the kernel itself is often three-dimensional.

The number of kernels controls the number of dimensions of the layered output since each kernel would generate its own feature map. Increasing the number of kernels can help reduce information loss in architectures where the feature map size decreases with each layer. It also controls the capacity of the model - as the number of kernels increases, so does the total number of parameters that can be trained.

Padding: the convolution is undefined at the input boundaries since part of the kernel cannot match any input values. To overcome this problem and apply convolution in corner cases, the input can be framed with zeros. The number of input values for which

convolution is defined directly affects the size of the output, so padding can be used to control it.

Stride controls the "step" of the convolution filter. A Stride value of two would mean that after applying convolution to some pixels, two pixels are skipped in all dimensions. By manipulating Stride, we can regulate the overlap of different receptive fields and the reduction of the output size. Let n_{in} , n_{out} , k , p , s be the total number of inputs and outputs, the total kernel size, the padding size, and the stride, respectively. Then the following relation holds [15]:

$$n_{out} = \lfloor \frac{n_{in} + 2p - k}{s} \rfloor + 1. \quad (4.4)$$

3.5.3 Pooling

A typical convolutional layer consists of three stages:

- 1) Applying convolutions to produce intermediate results.
- 2) Passing intermediate results through a non-linear activation function, like in the standard multilayer perceptron. It is sometimes called the detection *stage*.
- 3) Applying a **pooling function**.

The pooling function replaces rectangular areas of the input with their summary. It can be viewed as a non-linear downsampling method.

Let $[a_{i,j}] = A \in \mathbb{R}^{n \times m}$ be a real matrix that represents a feature map region that is being passed to a pooling function. Most commonly used pooling methods include:

- **Max pooling**, which replaces the input region with its maximum value:

$$(\) = (\) = (\{ \ , \mid \in I, \dots, \ , \in I, \dots, \}). \quad (4.5)$$

- **Average pooling** and **weighted average pooling** aggregate the input region by taking its average or a weighted sum, which can be based on the distance from the region's center:

$$p(A) = \frac{1}{nm} \sum_{i=1}^n \sum_{j=1}^m a_{i,j}. \quad (4.6)$$

- **L2 pooling** computes the L2 norm of the vector constructed by “unfolding” the input region.

$$p(A) = \sqrt{\sum_{i=1}^n \sum_{j=1}^m a_{i,j}^2}. \quad (4.7)$$

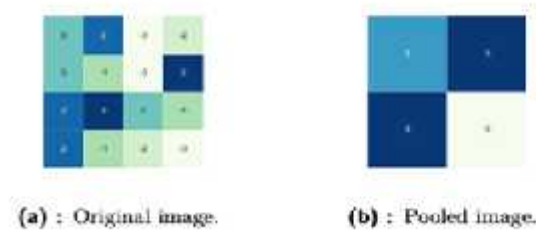


Figure 4.9. Pooling

Pooling is sometimes considered a separate layer, but in most popular architectures it always follows the convolution and can be considered part of it. However, it can be situation-dependent and there are successful architectures that use pooling following several stacked convolution operations or that do not use pooling at all [45]. It can be set with padding and stride hyperparameters, analogous to convolution.

Pooling is used to reduce the number of variables in the model to prevent overfitting and improve computational efficiency. However, heavy use of pooling can lead to aggressive information loss and underfitting, so it should be treated with caution. Another purpose of using a pooling function is to introduce translation invariance into the model [15]. By considering entire regions instead of separate input values, pooling helps to emphasize the value of the feature more, regardless of its position. As a result, the network becomes resistant to small perturbations in the input.

3.5.4 Batch Normalization

The **batch normalization layer** [24] normalizes the output of the layer before it. Let $\mathbf{X} \subseteq \mathcal{D}$ be a batch of inputs, then:

Algorithm 4.1 Compute the output \mathbf{y} of the batch normalization layer.

$$\begin{aligned} \mu &\leftarrow \frac{1}{|\mathbf{X}|} \sum_{\mathbf{x} \in \mathbf{X}} \mathbf{x} \\ \sigma^2 &\leftarrow \frac{1}{|\mathbf{X}|} \sum_{\mathbf{x} \in \mathbf{X}} (\mathbf{x} - \mu)^2 \\ \hat{\mathbf{x}} &\leftarrow \frac{\mathbf{x}_i - \mu}{\sqrt{\sigma^2 + \epsilon}} \\ \mathbf{y} &= \gamma \hat{\mathbf{x}} + \beta \end{aligned}$$

▷ Scale and bias

By centring and scaling the feature maps batch normalization makes the gradient computation more robust because. The main goal of the layer is to discard the change in the distribution of hidden layer outputs, which can happen in the process of training. This simplifies learning and provides faster convergence toward the minimum. The values of γ and β are determined in the process of learning.

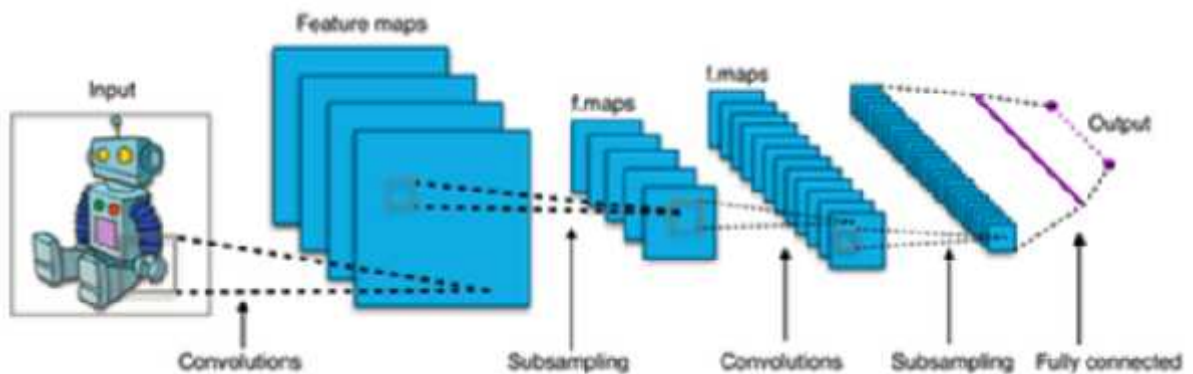


Figure 4.10. CNN structure

4.5.5. Properties of CNN

In traditional fully connected layers, every output is connected to every input. In CNN, neuron connections are restricted to only the subset of adjacent inputs through the kernel. This subset of neurons is sometimes referred to as the neuron's local receptive field.

This approach aims to detect local patterns and small details in the input.

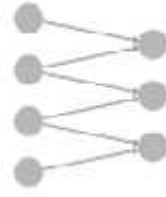


Figure 4.11. Local receptive fields

Despite the typically small size of the receptive field, neurons located in the deeper layers interact indirectly with the larger area of input. It is in the nature of convolution that when the output of the layer is defined to be smaller than the input, its feature map consists of aggregated input regions. Following this principle, feature maps in the deepest layers of the network will assemble "dense" information describing interactions between different regions of the input.

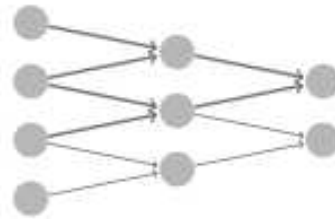


Figure 4.12. Deep indirect interactions

One of the major drawbacks of the multilayer perceptron is the large increase in the number of trainable parameters with each new layer added to the model, making the models impractical to train and limiting the breadth and depth of the network. Meanwhile, convolutional layers have far fewer parameters distributed throughout the layer.

Let m , n be the number of neurons in two consecutive layers, respectively, then a traditional fully connected layer would have $n(m + 1)$ trainable parameters (with the addition of biases). But convolutional layers with kernel size $w \times h$ end up having the sum of $wh + n$ weights and biases. This number can be sufficiently smaller than that of the fully linked layer.

Thus, parameter sharing leads to a smaller number of model parameters and positively affects the computational and storage costs during training and prevents overfitting.

However, it is worth noting that for some problems it is useful to have non-shared parameters. For example, in the face recognition problem, where the dataset contains centred and normalized images, different filters would capture different information, but at a higher computational cost.

CHAPTER 4

MODELLING OF CNN BASED HEAD MOUSE CONTROL

5.1. Design of CNN based head mouse control system

Haar Cascade classifiers and CNN are used to construct a mouse control system with head movements and eye blinking for persons with disabilities. The system receives images from the camera in front of the user as input. Haar Cascade Classifier uses input image to segment and extract the head profile and eyes of user. Two CNN models are employed for extracted images. The first CNN1 model is used for user's head profile, the second CNN2 model for eyes of user. The CNN takes the user's head profile as input and identifies the user's face direction (left, right, forward, down, and up). The mouse pointer is moved in the correct direction using this information. The second CNN structure takes the eyes retrieved from the head-shoulder profile as input. The cropped eye image is fed into the second CNN structure, which detects whether the eye is open or closed. This data is utilized to make a mouse click. The left mouse button is triggered by the left eye, whereas the right mouse button is triggered by the right eye.

The four parameters of height, width, depth, and number of classes are used to define CNN models in this thesis. The input photos' width and height are specified. The number of channels in the incoming images determines the depth. The user's head-shoulder profile input is 64x64x3, with width 64, height 64, and depth 3 in standard RGB. 24x24x3 is the size of the retrieved eye input.

As the layers are added one by one, the CNN models are defined sequentially. (See Figure 3.) The structure of the developed CNN1 used for head direction detection is shown in Table 1. The first convolutional layer employs 64 convolutional filters with 5x5 inputs each. When the input is less than 0, the Rectified Linear Unit (ReLU) activation function outputs 0; otherwise, it outputs 1. Following these steps, 2x2 max pooling is applied in both the x and y directions. The second and third convolutional layers used 128 and 256 convolutional filters, respectively, to execute the same tasks.

The initial CNN structure employs 64 filters in the first convolutional layer, 128 filters in the second convolutional layer, and 256 filters in the third convolutional layer, as previously mentioned. The ReLU activation layers are employed between the convolutional layers to prevent over-matching. A flattening layer is performed after the

convolutional layers to retrieve the output of the last convolutional layer, and the output space is smoothed into a single vector. 1000 input nodes and 5 dense output nodes (since we have 5 classes - left, right, up, down, and no action) are utilized in the dense fully linked layer for classification in the first CNN model after flattening. The same processes are applied and employed in fully linked networks with two outputs for classification purposes in the second CNN structure (because the classification is binary whether "open" or "closed"). The structure of CNN2, which is utilized to detect eye states, is shown in Table 2.

Table5.1. The structure of the CNN1 model

Table5.2. The structure of the CNN2 model

	Layer Type	Output Shape	Param
Input Layer	Input	64 x 64 x 3	0
Conv 1	Conv1	64 x 64 x 64	4864
	ReLU	64 x 64 x 64	0
	Pool1	32 x 32 x 64	0
Conv 2	Conv2	32 x 32 x 128	204928
	ReLU	32 x 32 x 128	0
	Pool2	16 x 16 x 128	0
Conv 3	Conv3	16 x 16 x 256	295168
	ReLU	16 x 16 x 256	0
	Pool3	8 x 8 x 256	0
Classification layer	Flatten	16384	0
	Dense1	1000	16385000
	ReLU	1000	0
	Dense2	5	5005
	Softmax	5	0

	Layer Type	Output Shape	Param
Input Layer	Input	24 x 24 x3	0
Conv 1	Conv1	24 x 24 x 64	4864
	ReLU	24 x 24 x 64	0
	Pool1	12 x 12 x 64	0
Conv 2	Conv2	12 x 12 x 128	204928
	ReLU	12 x 12 x 128	0
	Pool2	6 x 6 x 128	0
Conv 3	Conv3	6 x 6 x 256	295168
	ReLU	6 x 6 x 256	0
	Pool3	3 x 3 x 256	0
Classification layer	Flatten	2304	0
	Dense1	1000	2305000
	ReLU	1000	0
	Dense2	2	2002
	Softmax	2	0

5.2. Modelling

A "head-mouse control system" for people with disabilities is being designed using the CNN-based classifier system. With the parameters provided in Figure 5.1, the Tensorflow framework is employed. Our hand-crafted dataset was utilized to predict head posture. The dataset contains 853 photos of 7 persons from our university who are of various nations, genders, and ages. A hair cascade classifier is used in the first stage to recognize the head profile and eyes. The process of head and eye recognition is depicted in Figure 5.2. Image categorization is done when CNN is used to recognize the head profile. 90% of the photos are used for training, while 10% are used for testing in the system design. This also aids in the reduction of overfitting and the enhancement of performance. To acquire more accurate findings from tiny datasets, several generating techniques such as cropping, shifting, random rotations, and so on were used in the data augmentation phase. The system's design includes 50 training epochs, a $1e-3$ learning rate, and 64 batch sizes. The loss function values and accuracy obtained for training and validation are shown in Figure 6. The system is tested after it has been trained. For the test data, an RMSE of 0.0076 and an accuracy of 99.76 percent were attained.



Figure 5.1. Fragment of Head Pose Dataset

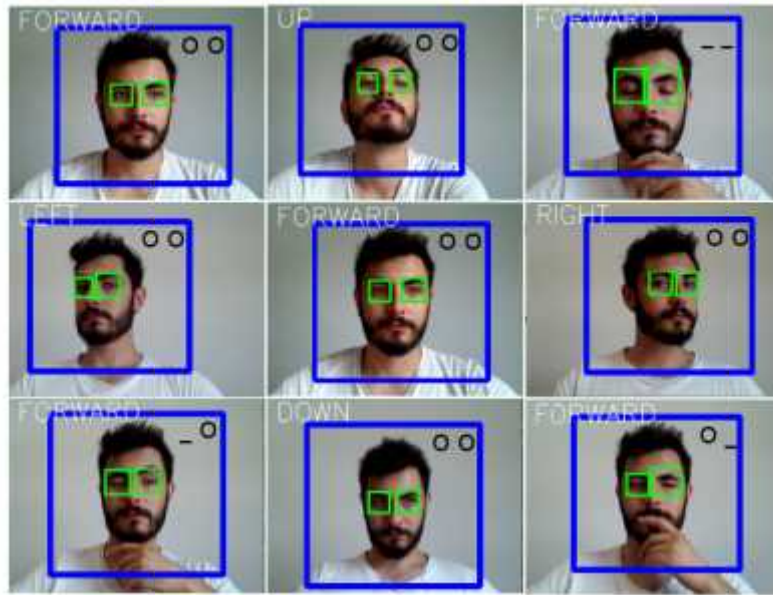


Figure 5.2. Detection of eyes

5.3.Simulation Results and Discussion

The CEW dataset (Song et al., 2014) is utilized for eye classification, and it contains 2385 closed and 2463 open 24x24 pixel eye pictures. Eye classification is conducted after recognizing the eyeballs with the Haar cascade classifier. Figure 5.3 depicts the plots of loss functions and accuracies for training and validation data sets for head classification, correspondingly. CNN2 uses 90% of the photos in its design, just as CNN1. 70 percent of this 90 percent is used for training, and 30 percent is used for CNN2 validation. The remaining 10% of the information is used for testing. With 50 epochs, the CNN2 model is trained. Figure 5.4 shows the training and validation results for eye classification obtained for loss and accuracy. The system is tested after it has been trained. For the test data, an RMSE of 0.0575 and an accuracy of 97.42 percent were found.

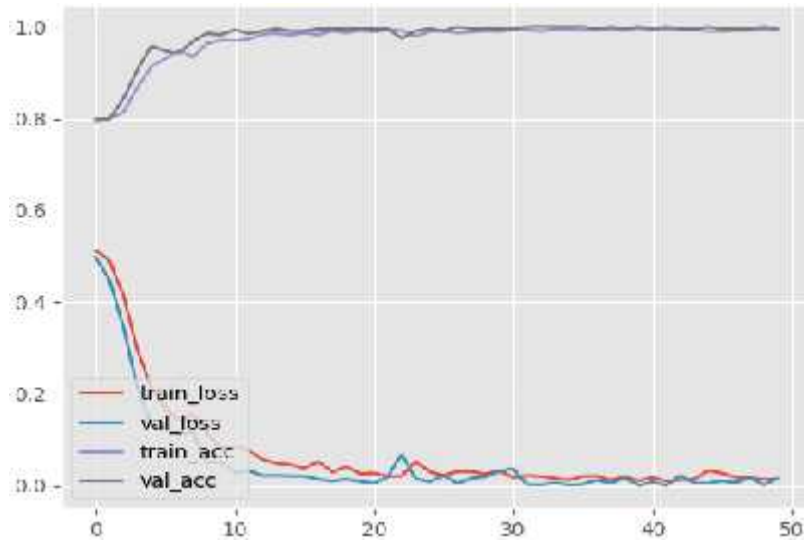


Figure5.3. Training and validation results obtained for loss and accuracy of head classification

The performance of the head mouse control system is illustrated for the train, assessment, and testing stages using CNN1 and CNN2 models. RMS, ACC, and AUC values are determined during the simulation. Table 5.3 shows the training, evaluation, and testing stages of each model's outcomes.

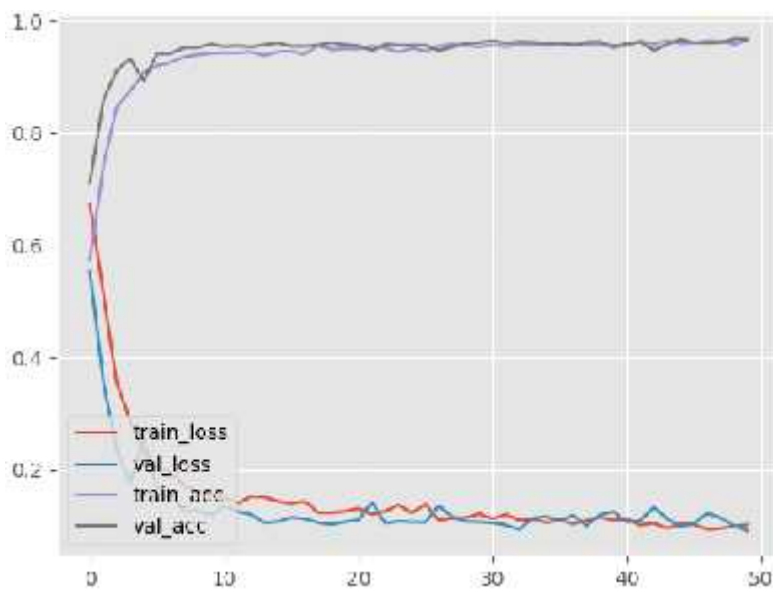


Figure5.4. Training and validation results obtained for loss and accuracy of eyes classification

Table 5.4 shows the results of the performance of the most competitive models tested for head-mouse control. The papers that offered the accuracy results were taken into consideration. The simulations were run using the authors' personal databases. As indicated, some studies use a sensor-based strategy for head-mouse control, while others utilize a vision-based approach. The introduction section contains analyses of several of these works. When authors reported various findings using the same approach, only the best was used in the table. The performance of the CNN models described in this paper is shown in Table 5.5. The findings of CNN models are clearly superior to those of the approaches based on vision-based techniques shown in Table 5.4.

Table 5.3. Performances of the CNN models

Model	Training			Validation			Testing		
	RMSE	AUC	ACC	RMSE	AUC	ACC	RMSE	AUC	ACC
Eye	0.0585	0.9925	96.09%	0.0593	0.9891	95.21%	0.0575	0.9883	97.42%
Head	0.0196	0.9996	99.72%	0.0094	1.0000	98.84%	0.0076	0.9999	99.76%

Table 5.4. The simulation results of the selected studies obtained for the head-mouse control

Author (Year)	Technique	Performance
Arai and Mardiyanto (2010)	Vision-based	90%
Alhamzawi H.A. (2018).	Vision-based	95% (Detection) 84.86% (Clicking)
Su et al. (2005)	Vision-based (Eye tracking)	99%
Eom et al. (2007).	Gyro-sensor based	93%
Tolle and Arai (2016)	Sensor based	80 % (average)
Sawicki and Kowalczyk (2018).	Sensor and vision based	95.14% (Total error rate 4.86%)

Chen (2001)	Sensor based	97.8% (nondisabled) 95.1% (disabled)
King et al. (2005)	Gyro-sensor based	99.85%
Ngyen et al. (2006)	Gyro-sensor based	93.75%
Naizhong et al. (2015)	Vision-based (Mouth tracking)	97.09% (average)
Varona et al. (2008)	Vision based	97.3%
Jian-Zheng and Zheng(2011)	Vision-based and LK algorithm, Gentleboost algorithm (Eye detection)	97.7%
Zhao et al.(2012)	Vision-based and LK algorithm (Eye detection)	92.6% (average)
Zhao and Yan (2011)	Vision-based (Eye detection)	90%
Mehrubeoglu et al. (2011)	Vision-based (Eye detection)	95.44% (ideal eye) 90.13%(moving eye)
Li et al. (2016)	Vision-based (Eye detection)	85.3%
Wu et al. (2008)	Vision-based (Head pose estimation, KDA with Gaussian kernel)	94.0%

The performance of the CNN models employed for head-mouse control is compared to the performance of the MLP and HOG -SVM models for comparison (Table 5). A series of experiments were carried out with MLP in the first phase in order to determine its ideal structure. The structure of MLP1 used for head classification has been determined as a consequence of the tests (2352, 500, 5). The number of input nodes is 2352, the number of hidden nodes is 500, and the number of output nodes is 5. (clusters). The structure of MLP2, which is used to classify eyeblinks, is as follows: (1728, 500, 2).The number of input, hidden, and output nodes is 1728, 500, and 2, respectively. The loss function and head classification accuracy are calculated to be 0.2340 and 92.15 percent, respectively. For eyeblink classification, the loss function and accuracy were determined to be 0.2353 and 91.29 percent, respectively. SVM is used to replicate head and eye blink classifications in the next stage. The histogram gradient technique (HOG) was utilized to extract features

during the simulation. SVM is fed the features that represent the images for classification. The loss function values (RMSE value) for head and eyeblink classification were found to be 0.2019 and 0.0652, respectively, as a consequence of the simulation. For head and eye lens categorization, the accuracy scores were 98.12 percent and 93.48 percent, respectively. The CNN models perform better than the MLP and HOG-SVM models, as seen in the table. The comparative result indicates the effectiveness of CNN in head-mouse control.

Table 5.5.Comparative Results

Model	Head		Eye-blink	
	RMSE	ACC	RMSE	ACC
MLP	0.1108	92.15%	0.5353	91.29%
HOG+SVM	0.2019	98.12%	0.0652	93.48%
CNN	0.0076	99.76%	0.0575	97.42%

CHAPTER 6

CONCLUSIONS

- It is difficult, if not impossible, for disabled people with spinal cord injuries to make efficient use of computer technologies. Designing an effective human-computer interface for impaired persons is extremely beneficial, and it can even lead to career prospects. The analysis of existing head-mouse systems that used a human-computer interface has shown they have several limitations. Some of them require special hardware, such as certain cameras or sensor-based devices, some have additional components and the user has to wear these components. These systems use special methods to solve feature extraction and classification problems and they are expensive, inconvenient, and provide limited data for future computers. Therefore, in the thesis, deep learning using Convolutional Neural Networks (CNN) is proposed for solving this problem. The proposed system does not use special hardware for wearing and integrates images' feature extraction and classification stages in the unified body of CNN, which allows simplifying the system architecture used for image recognition.
- The structure of the head-mouse control system is proposed using CNN. The proposed system includes two subsystems that are head movements recognition and eye states detection and each one includes image capturing, processing and classification stages. The performance of the proposed system was improved due to its designed architecture and learning property of CNN.
- The learning algorithm of CNN based head mouse control system has been designed using adaptive moment estimation (Adam optimizer).
- The design of the head-mouse control system is carried out by implementing the design CNN based system for recognition of head movements and the design of CNN based system for detection of eye states.
- For experimental evaluation of the proposed system, the CEW data sets that include 4848 eye images and 853 head images designed in the Applied Artificial Intelligence Research Centre of NEU were taken. The results obtained from experiments have shown CNN based head-mouse control system requires few preprocessing steps and provides high accuracy both for head movement recognition and eye states detection.

The accuracy of head movement recognition was obtained as 99.76%, accuracy of eye states detection- 97.42%.

- Without any additional hardware, the designed system can identify the head direction and eye states from the camera image and utilize them to operate the mouse pointer. The system is quite robust and effective, according to the findings of the experiments. The described idea can be used to manipulate a computer mouse by disabled people.

REFERENCES

1. Chen Y.L., Chang W.H., Tang F.T., Wong M.K., Kuo T.S. (1999). The new design of an infrared-controlled human-computer interface for the disabled. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 7, 474-481.
2. Lin C.S., Chang K.C., Jain Y.J. (2002). A new data processing and calibration method for an eye-tracking device pronunciation system, *Optics and Laser Technology*, 34 (5), 405-413.
3. Park K.S., Lee K.T. (1996). Eye-controlled human-computer interface using the line-of-sight and the intentional blink, *Computer Engineering*, 30(3), 463-473.
4. Viola P. & Jones M.J. (2001). Rapid object detection using a boosted cascade of simple features. In Proceed. of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR , 1, 511-518.
5. Rowley H., Baluja S., & Kanade T. (1998). Neural network-based face detection. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 20(1), 23-38. doi: 10.1109/34.655647
6. Osuna E., Freund R., Girosit F. (1997). Training support vector machines: an application to face detection. In Proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997. <https://doi.org/10.1109/CVPR.1997.609310>
7. Ahonen T., Hadid A., & Pietikainen M. (2006). Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 28(12), 2037-2041. doi: 10.1109/tpami.2006.244
8. Senaratne, R., Halgamuge, S., & Hsu, A. (2009). Face Recognition by Extending Elastic Bunch Graph Matching with Particle Swarm Optimization. *Journal Of Multimedia*, 4(4). doi: 10.4304/jmm.4.4.204-214
9. Zhang, H., Zhang, Y., & Huang, T. (2013). Pose-robust face recognition via sparse representation. *Pattern Recognition*, 46(5), 1511-1521. doi: 10.1016/j.patcog.2012.10.025
10. Zhao, W., Chellappa, R., Phillips, P., & Rosenfeld, A. (2003). Face recognition. *ACM Computing Surveys*, 35(4), 399-458. doi: 10.1145/954339.954342

11. Fragopanagos, N., & Taylor, J. (2005). Emotion recognition in human-computer interaction. *Neural Networks*, 18(4), 389-405. doi: 10.1016/j.neunet.2005.03.006
12. Adolphs, R., Damasio, H., Tranel, D., Cooper, G., & Damasio, A. (2000). A Role for Somatosensory Cortices in the Visual Recognition of Emotion as Revealed by Three-Dimensional Lesion Mapping. *The Journal Of Neuroscience*, 20(7), 2683-2690. doi: 10.1523/jneurosci.20-07-02683.2000
13. Cohen, I., Sebe, N., Garg, A., Chen, L., & Huang, T. (2003). Facial expression recognition from video sequences: temporal and static modelling. *Computer Vision And Image Understanding*, 91(1-2), 160-187. doi: 10.1016/s1077-3142(03)00081-x
14. Chen, Y., Chen, W., Kuo, T., & Lai, J. (2003). A head movement image (HMI) controlled computer mouse for people with disabilities Analysis of a time-out protocol and its applications in a single server environment. *Disability And Rehabilitation*, 25(3), 163-167. doi: 10.1080/0963828021000024960
15. Pereira C., BolligerNeto R., Reynaldo A., Luzo M., & Oliveira R. (2009). Development and evaluation of a head-controlled human-computer interface with mouse-like functions for physically disabled users. *Clinics*, 64(10). doi: 10.1590/s1807-5932200900100000
16. Al-Rahayfeh A., Faezipour M. (2013). Eye tracking and head movement detection: a state-of-art survey. *IEEE Journal of Translational Engineering in Health and Medicine*, 1, doi:http://dx.doi.org/10.1109/JTEHM.2013.2289879.
17. Chen Y-L. (2001). Application of tilt sensors in human-computer mouse interface for people with disabilities. *IEEE Transactions On Neural Systems And Rehabilitation Engineering*, 9(3), 289-294. doi: 10.1109/7333.948457
18. Fouché R.C. (2017). Head mouse: generalisability of research focused on the disabled to able bodied users. In *Proceed. of the South African Institute of Computer Scientists and Information Technologists, SAICSIT '17*, 1-10, 10.1145/3129416.3129442.
19. Mishra M., Bhalla A., Kharad S., Yadav D. (2017). HMOS: Head Control Mouse Person with Disability. *International Journal on Recent and Innovation Trends in Computing and Communication* ISSN: 2321-8169, 5(5).
20. Eom G.-M., Kim K.-S., Kim C.-S., Lee J., Chung S.-C. (2007). Gyro-Mouse for the Disabled: 'Click' and 'Position' Control of the Mouse Cursor. *International Journal of Control, Automation, and Systems*, 5(2), 147-154.

21. Sim N., Gavriel C., Abbott W.W., Faisal A.A. (2013). The Head Mouse – Head Gaze Estimation "In-the-Wild" with Low-Cost Inertial Sensors for BMI Use. In proc. of 6th Annual International IEEE EMBS Conference on Neural Engineering, 735-738
22. Gerdman C., Backlund Y., Linden M. (2012). A gyro sensor based computer mouse with a USB interface: A technical aid for motor-disabled people. *Technology and Disability* 24, 117–127 DOI 10.3233/TAD-2011-0340
23. King L. M., Nguyen H. T., and Taylor P. B.(2005). Hands-free head-movement gesture recognition using artificial neural networks and the magnified gradient function,' in Proc. 27th Annu. Conf. Eng. Med. Biol., 2063-2066.
24. Nguyen S. T., Nguyen H. T., Taylor P. B., and Middleton J.(2006). Improved head direction command classification using an optimised Bayesian neural network, in Proc. 28th Annu. Int. Conf. EMBS, 5679-5682.
25. Kim S, Park M., Anumas S., Yoo J. (2010). Head mouse system based on gyro- and opto-sensors. Proc. 3rd International Conference on Biomedical Engineering and Informatics (BMEI), 4, 1503–1506, doi:<http://dx.doi.org/10.1109/BMEI.2010.5639399>.
26. Arai K., Mardiyanto R.(2010). Camera as Mouse and Keyboard for Handicap Person with Troubleshooting Ability, Recovery, and Complete Mouse Events. *International Journal of Human Computer Interaction (IJHCI)*, 1(3), 46-56.
27. Su M-C, Su S-Y, Chen G-D.(2005). A low-cost vision-based human-computer interface for people with severe disabilities. *Biomedical Engineering Applications, Basis & Communications*. 17(6), 284-292.
28. Tolle H. and Arai K. (2016). Design of Head Movement Controller System (HEMOCS) for Control Mobile Application through Head Pose Movement Detection. *International Journal of Interactive Mobile Technologies (IJIM)*, 10(3), 24-28
29. Alhamzawi H.A.(2018). Control Mouse Cursor by Head Movement: Development and Implementation. *Applied Medical Informatics Original Research*, 40(3-4), 39-44.
30. Naizhong Z, Jing W., Jun W..(2015). Hand-Free Head Mouse Control Based on Mouth Tracking. In Proceed. 10th International Conference on Computer Science & Education (ICCSE 2015), 707-713, Fitzwilliam College, Cambridge University, UK
31. Palleja T.,Rubion E., Teixido M., Tresanchez M., del Viso A.F., RebatíC., PalacinJ.(2008). Simple and robust implementation of a relative virtual mouse

- controlled by head movements. In *Proceed. of IEEE Conference on Human System Interactions*, 221-224.
32. Lin C-S, Ho C-W., Chan C-N., Chau C-R., Wu Y.-C., Yeh M-S. (2007). An eye-tracking and head-control system using movement increment-coordinate method. *Optics & Laser Technology*, 39, 1218–1225
 33. Ismail A., AL HAJJAR AES. and HAJJAR M. (2011). A prototype system for controlling a computer by head movements and voice commands. *The International Journal of Multimedia & Its Applications (IJMA)*, 3(3), 15-25.
 34. Sawicki D., Kowalczyk P. (2018). Head Movement Based Interaction in Mobility, *International Journal of Human-Computer Interaction*, 34:7, 653-665, DOI: 10.1080/10447318.2017.1392078
 35. Velasco M.A., Clemotte A., Raya R., Ceres R., Rocon E. (2017). Human-Computer Interaction for Users with Cerebral Palsy Based on Head Orientation. Can Cursor's Movement Be Modeled by Fitts' Law? *Int. Journal of Human-Computer Studies*, 106,1-9
 36. Varona J., Manresa-Yee C., Perales F.J.(2008). Hands-free vision-based interface for computer accessibility. *Journal of Network and Computer Applications*. 31. 357–374
 37. Manresa-Yee C., Varona J., Perales F.J., Salinas I. (2014). Design recommendations for camera-based head-controlled interfaces that replace the mouse for motion-impaired users. *Univ Access InfSoc*, 13:471–482
 38. King D. E. (2009). *Dlib-ml: A machine learning toolkit*. JMLR.
 39. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.(1998). Gradient-based learning applied to document recognition. In *Proceed. of the IEEE*. 86(11), 2278-2324.
 40. Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. *Neural networks for perception*, 2, 65-93.
 41. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A., Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115, 211-252.
 42. Zeiler M.D., Fergus R. (2014). Visualizing and Understanding Convolutional Networks. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) *Computer Vision –*

- ECCV 2014. *Lecture Notes in Computer Science*, 8689, 818-833, doi: https://doi.org/10.1007/978-3-319-10590-1_53
43. Simonyan K, Zisserman A. (2015). Very deep convolutional networks for large-scale image recognition, In Proceedings of the International Conference on Learning Representations (ICLR 2015), doi: 10.1.1.740.6937
 44. Szegedy C., Liu W., Jia Y., Sermanet P., Reed S., Anguelov D., Erhan D., Vanhoucke V., Rabinovich A. (2015). Going deeper with convolutions, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015), 1–9. doi: 10.1109/CVPR.2015.7298594
 45. He K, Zhang X, Ren S., Sun J. (2016). Deep residual learning for image recognition, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (CVPR 2016), 770–778, doi: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90)
 46. Lawrence, S., Giles, C., Ah Chung Tsoi, & Back, A. (1997). Face recognition: a convolutional neural-network approach. *IEEE Transactions on Neural Networks*, 8(1), 98-113. doi: 10.1109/72.554195
 47. Ciresan, D., Meier, U., Gambardella, L. and Schmidhuber, J. (2011). Convolutional Neural Network committees for Handwritten Character Classification. In Proceedings of 2011 International Conference on Document Analysis and Recognition, 3207 - 3220 doi: [10.1109/ICDAR.2011.229](https://doi.org/10.1109/ICDAR.2011.229)
 48. Simard P, Steinkraus D and Platt J. (2003). Best practices for convolutional neural networks applied to visual document analysis. In Proceedings of Seventh International Conference on Document Analysis and Recognition, 958 - 963 doi: [10.1109/ICDAR.2003.1227801](https://doi.org/10.1109/ICDAR.2003.1227801)
 49. Jiao, L., Zhang, S., Li, L., Liu, F., & Ma, W. (2018). A modified convolutional neural network for face sketch synthesis. *Pattern Recognition*, 76, 125-136. doi: 10.1016/j.patcog.2017.10.025
 50. Chudzik, P., Majumdar, S., Calivá, F., Al-Diri, B., & Hunter, A. (2018). Microaneurysm detection using fully convolutional neural networks. *Computer Methods And Programs in Biomedicine*, 158, 185-192. doi: 10.1016/j.cmpb.2018.02.016

51. Li, J., Feng, J., &Kuo, C. (2018). Deep convolutional neural network for latent fingerprint enhancement. *Signal Processing: Image Communication*, 60, 52-63. doi: 10.1016/j.image.2017.08.010
52. Hussain, S., Anwar, S., & Majid, M. (2018). Segmentation of glioma tumours in brain using deep convolutional neural network. *Neurocomputing*, 282, 248-261. doi: 10.1016/j.neucom.2017.12.032
53. Baldominos, A., Saez, Y., &Isasi, P. (2018). Evolutionary convolutional neural networks: An application to handwriting recognition. *Neurocomputing*, 283, 38-52. doi: 10.1016/j.neucom.2017.12.049
54. Ferreira, A., &Giraldi, G. (2017). Convolutional Neural Network approach to granite tiles classification. *Expert Systems With Applications*, 84, 1-11. doi: 10.1016/j.eswa.2017.04.053
55. Wachinger, C., Reuter, M., & Klein, T. (2018). DeepNAT: Deep convolutional neural network for segmenting neuroanatomy. *Neuroimage*, 170, 434-445. doi: 10.1016/j.neuroimage.2017.02.03
56. Liu, J., Gong, M., Qin, K., & Zhang, P. (2018). A Deep Convolutional Coupling Network for Change Detection Based on Heterogeneous Optical and Radar Images. *IEEE Transactions on Neural Networks And Learning Systems*, 29(3), 545-559. doi: 10.1109/tnnls.2016.2636227
57. Liu, N., Han, J., Liu, T., & Li, X. (2018). Learning to Predict Eye Fixations via Multiresolution Convolutional Neural Networks. *IEEE Tran. on Neural Networks And Learning Systems*, 29(2), 392-404.
58. Salvati, D., Drioli, C., &Foresti, G. (2018). Exploiting CNNs for Improving Acoustic Source Localization in Noisy and Reverberant Conditions. *IEEE Transactions on Emerging Topics In Computational Intelligence*, 2(2), 103-116. doi: 10.1109/tetci.2017.2775237
59. Khodayar, M., Kaynak, O., &Khodayar, M. (2017). Rough Deep Neural Architecture for Short-Term Wind Speed Forecasting. *IEEE Transactions on Industrial Informatics*, 13(6), 2770-2779.
60. Kalchbrenner N, Grefenstette E, and Blunsom E. (2014). A Convolutional Neural Network for Modelling Sentences. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, doi: 10.1.1.636.1565

61. Abiyev R., Ma'aitah M.K.S. (2018). Deep Convolutional Neural Networks for Chest Diseases Detection. *Journal of Healthcare Engineering*, Volume 2018, 1-11.
62. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R and Fei-Fei L. (2014). Large-Scale Video Classification with Convolutional Neural Networks. In *Proceeding of 2014 IEEE Conference on Computer Vision and Pattern Recognition*, 1725 – 1732, doi: 10.1109/CVPR.2014.223
63. Abiyev R., Arslan M., Gonsel I. and Cagman A. (2017). Robot Pathfinding Using Vision-Based Obstacle Detection. *2017 3rd IEEE International Conference on Cybernetics (CYBCONF 2017)*, 1 - 6 doi: 10.1109/CYBConf.2017.7985805
64. Kingma D. P., Jimmy B.. A. (2015). A Method for Stochastic Optimization. *International Conference on Learning Representations (ICLR 2015)*, <https://arxiv.org/abs/1412.6980>
65. Song F., Tan X., Liu X. & Chen S. (2014). Eyes Closeness Detection from Still Images with Multi-scale Histograms of Principal Oriented Gradients, *Pattern Recognition*. 47(9), 2825–2838
66. Jian-Zheng L. and Zheng Z.(2011). Head movement recognition based on LK algorithm and Gentleboost, in *Proc. 7th Int. Conf. Netw. Comput. Adv. Inf. Manag.*, pp. 232-236.
67. Zhao Z., Wang Y., and Fu S.(2012). Head movement recognition based on Lucas-Kanade algorithm,' in *Proc. Int. Conf. CSSS*, 2303-2306.
68. Zhao Y. and Yan H. (2011). Head orientation estimation using neural network. in *Proc. ICCSNT*, 3, 2075-2078.
69. Mehrubeoglu M., Pham L. M., Le H. T., Muddu R., and Ryu D. (2011). Realtime eye tracking using a smart camera. in *Proc. AIPR Workshop*, 1-7.
70. Li Y., Dou Y., Liu X., Li T. (2016). Localized region context and object feature fusion for people head detection .2016 IEEE International Conference on Image Processing (ICIP), 594 – 598
71. Wu J. and Trivedi M. (2008). A Two-Stage Head Pose Estimation Framework and Evaluation,” *Pattern Recognition*, 41(3), 1138-1158.
72. Ma B., Zhang W., Shan S., Chen X., and Gao W. (2006). Robust Head Pose Estimation Using LGBP, *Proc. 18th Int’l Conf. Pattern Recognition*, 512-515.

73. Rahib H. Abiyev, Murat Arslan. Head mouse control system for people with disabilities. *Expert Systems*, 37, 2020, <https://doi.org/10.1111/exsy.12398>
74. Arslan M., Bush I.J., Abiyev R.H. (2019) Head Movement Mouse Control Using Convolutional Neural Network for People with Disabilities. In: Aliev R., Kacprzyk J., Pedrycz W., Jamshidi M., Sadikoglu F. (eds) 13th International Conference on Theory and Application of Fuzzy Systems and Soft Computing — ICAFS-2018. ICAFS 2018. *Advances in Intelligent Systems and Computing*, vol 896. Springer, Cham. https://doi.org/10.1007/978-3-030-04164-9_33
75. RahibAbiyev and Murat Arslan, Vision-Based Drowsiness Detection System Using Convolutional Neural Networks, ICECCE-2020, June 11-14, 2020, Istanbul. <https://doi.org/10.1109/ICECCE49384.2020.9179207>
76. RahibAbiyev, Murat Arslan, Irfan Günsel, Ahmed Çagman. Robot Pathfinding Using Vision Based Obstacle Detection. 3rd IEEE International Conference on Cybernetics (CYBCONF), Book Series: IEEE International Conference on Cybernetics-CYBCONF, 29-34, Exeter, ENGLAND Date: JUN 21-23, 2017
77. RahibAbiyev, John Idoko and Murat Arslan. Application of Reconstructed Convolutional Neural Network Structure to Sign Language Translation, ICECCE-2020, June 11-14, 2020, Istanbul. <https://doi.org/10.1109/ICECCE49384.2020.9179356>
78. Rahib Abiyev ,Murat Arslan ,John Bush Idoko ,Boran Sekeroglu and Ahmet Ilhan. Identification of Epileptic EEG Signals Using Convolutional Neural Networks. *Appl. Sci.* 10(12), 2020, 4089; <https://doi.org/10.3390/app10124089>
79. Rahib H. Abiyev, John Bush Idoko, Murat Arslan. Sign Language Translation Using Deep Convolutional Neural Networks. *KSII Transactions on Internet and Information Systems*, Vol.14, No.2, 2020. <https://doi.org/10.3837/tiis.2020.02.009>.
80. John Bush Idoko, Rahib H. Abiyev, Murat Arslan. Impact of Machine Learning Techniques on Hand Gesture Recognition. *Journal of Intelligent & Fuzzy Systems*, Volume: 37 Issue: 3 Pages: 4241-4252 Published: 2019, <https://doi.org/10.3233/JIFS-190353>
81. JB Idoko, M Arslan, R Abiyev. [Fuzzy Neural System Application to Differential Diagnosis of Erythemato-Squamous Diseases](https://doi.org/10.5152/cjms.2018.576). *Cyprus Journal of Medical Sciences* 3 (2), 2018, 90-97, **DOI:** 10.5152/cjms.2018.576

82. John Bush Idoko, MuratArslan, Rahib H. Abiyev. Intensive Investigation in Differential Diagnosis of Erythematous-Squamous Diseases.13th International Conference on Application of Fuzzy Systems and Soft Computing - ICAFS-2018 Book Series: Advances in Intelligent Systems and Computing Volume: 896 Pages: 146-153 Published: 2019, Location: Warsaw, POLAND Date: AUG 27-28, 2018, https://doi.org/10.1007/978-3-030-04164-9_21

APPENDICES

APPENDIX 1.
Program Listing

APPENDIX 2

APPENDIX 2: Ethical Approval letter



ETHICAL APPROVAL DOCUMENT

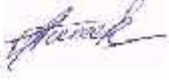
Date:01/06/2021

To the Graduate School of Applied Sciences

For the thesis project entitled as "CNN based head mouse control system for people with disabilities", the researchers declare that they did not collect any data from human/animal or any other subjects. Therefore, this project does not need to go through the ethics committee evaluation.

Title: Prof. Dr.

Name Surname: Rahib Abiyev

Signature: 

Role in the Research Project: Supervisor

APPENDIX 3

Similarity Report

ABOUT THIS PAGE
This is your assignment page. To view a paper select the paper's title. To view a similarity report, select the paper's similarity report icon in the similarity column. A message box indicates that the similarity report has not yet been generated.

paper

TURNITIN ASSIGNMENT NEW PAPERS

checkbox	checkbox	checkbox	checkbox	checkbox	checkbox	checkbox	checkbox	checkbox
similarity	match	approved	date	similarity	match	approved	date	similarity
0%	—	—	15/01/2021	0%	—	—	15/01/2021	0%
3%	—	—	16/01/2021	3%	—	—	16/01/2021	3%
0%	—	—	16/01/2021	0%	—	—	16/01/2021	0%
15%	—	—	15/01/2021	15%	—	—	15/01/2021	15%
17%	—	—	16/01/2021	17%	—	—	16/01/2021	17%
17%	—	—	16/01/2021	17%	—	—	16/01/2021	17%
18%	—	—	16/01/2021	18%	—	—	16/01/2021	18%

Supervisor.Prof.Dr.RahibAbiyev