UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Karl-Kristjan Luberg

Human Body Poses Recognition Using Neural Networks with Class Based Data Augmentation

Master's Thesis (30 ECTS)

Supervisor: Amnir Hadachi, PhD
Supervisor: Artjom Lind, MSc

Tartu 2018

Human Body Poses Recognition Using Neural Networks with Class Based Data Augmentation

Abstract: Information technologies and continuous development of information technologies have made it possible for computers to see and learn. What we see with our eyes, can be divided into pixels and fed to a computer, giving the computer the ability to see and learn based on the pixel values. Based on the input values computers can learn to recognize different objects depending on the examples taught to them. There are many possible applications for computers to see and learn in order to solve new tasks. In this thesis, we propose a framework, capable of automatically recognizing human body poses from a single image, obtained with a traditional low-cost camera. Our approach combines computer vision with neural networks to detect a human from an image. This process starts by extracting the silhouette from an image and then using a neural network to recognize body poses based on the extracted silhouettes. In order to match detected silhouettes with body poses, the neural network was trained with an already classified augmented dataset of preprocessed images depicting silhouettes. According to our results, we show that the proposed method provides promising results with acceptable accuracy.

Keywords: computer vision, machine learning, human detection, silhouette extraction, human body pose classification

CERCS: P170

Inimese keha pooside tuvastamine neurovõrkude abil kasutades klassipõhiste andmete augmenteerimist

Lühikokkuvõte: Infotehnoloogia ja selle pidev arenemine on võimaldanud arvutitel näha ja õppida. Mida meie näeme oma silmadega, on võimalik jagada piksliteks ja anda pikslid sisendiks arvutile. Pikslite väärtuste põhjal saab arvuti näha ja õppida tundma ära erinevaid objekte, mida neile tundma õpetatakse. Arvutinägemisel ja õppimisel on palju võimalikke rakendusi. Antud lõputöös pakume välja raamistiku, mis suudab automaatselt tuvastada inimese keha poose traditsioonilise odava kaameraga tehtud pildilt. Meie lähenemisviis ühendab arvutinägemise ja neurovõrgud, et tuvastada inimene pildilt. Antud protsess algab silueti eraldamisega pildilt ning jätkub neurovõrgu kasutamisega. Neurovõrk tuvastab keha poosi eraldatud silueti põhjal. Suutmaks siluetti seostada keha poosiga, treeniti neurovõrku eelnevalt töödeldud piltidega siluettidest. Saadud tulemuste põhjal pakub antud raamistik paljulubavaid tulemusi aktsepteeritava täpsusega.

Võtmesõnad: arvutinägemine, masinõpe, inimese tuvastamine, silueti eraldamine, inimese keha poosi klassifitseerimine

CERCS: P170

Acknowledgements

I am very grateful to Dr Amnir Hadachi for supervising and guiding me during the writing of master's thesis and master's program in general. He has always been incredibly supportive, provided proper guidance and endless encouragement. I would like to acknowledge his unbelievably positive attitude that is addictive and helped push forward with the thesis writing.

Furthermore, I'd like to thank Artjom Lind, my other advisor, for his expertise and assistance that has been of great value.

Abbreviation and Acronyms

This section clarifies some terms used in the paper.

HOG - histogram oriented gradients

RF - random forest

SVM - support vector machine

GMM - Gaussian Mixture Model

RGB - red, green and blue color model

NN - neural network

ANN - artificial neural network

CNN - convolutional neural network

Contents

1 Introduction

1.1 General view

The pedestrian recognition has been making considerable progress with the advancement in using deep learning [4] and preprocessing techniques [18]. This aspect is very important for future intelligent systems and autonomous vehicles. In addition, it is not only important to make the detection of pedestrian but it is also crucial to understand in which poses the pedestrian is such as standing, walking, running, etc. Since this knowledge will help a lot in decision making and provide the autonomous vehicles with valuable information for their artificial intelligence to process and make decisions. Besides, it can be also applied to detect poses of the drivers and support the advanced driver-assistance systems for decision-making regarding the safety of the driver and passengers [20]. Therefore, recognizing body pose is a key step in solving the problem of detecting distracted drivers.

To solve such problem, we are proposing a framework that combines computer vision with neural networks to recognize human body poses from images taken by a low-cost camera. The process starts by applying a preprocessing stage to facilitate the human silhouettes extraction. Then, detecting the human body and its extraction is based on using support vector machine (SVM) pretrained with histogram oriented gradients (HOG). After the human body is extracted it is used as input for the neural networks in order to detect the human body poses.

1.2 Objectives

The aim of this thesis work is to propose a framework that combines computer vision with neural networks to recognize human body poses from images taken by a low-cost camera. To that extent, the following steps should be done:

1) Investigate computer vision methods for human detection;

2) Investigate algorithms for human silhouette extraction from images;

3) Investigate machine learning methods for effective image-based classification of body poses;

4) Implement a program that detects human, extracts human silhouette and classifies human body pose;

5) Analyze and compare results with different neural network models;

6) Based on the obtained results, draw conclusions and propose a set of future improvements.

While working on this thesis, limitations were found. The biggest limitation is that the neural network requires huge training dataset to obtain highly accurate results. For the task at hand it is impossible to find labelled dataset and therefore a labelled dataset is created manually. Manually going through the silhouettes and labelling them is a tedious task and therefore the dataset used in this thesis has a limited size. Another limitation is the computational resources. Working with a large dataset and resource hungry algorithms requires powerful machines. It is especially critical when working with a neural network - training and testing it.

1.3 Contribution

The main contribution of this research is resulting framework for human body pose detection based on a single image taken with a low-cost camera. For this purpose, multiple methods are utilized to create a framework for human body pose detection. The framework includes detection of human from an image, extraction of human silhouette and classification of a human silhouette to determine the human body pose. Therefore framework consists of three steps utilizing different methodologies.

The aim of the thesis is to prove that chaining multiple methods together to create a framework for training human body pose classifying neural networks can achieve good accuracy given reasonable dataset for training. This work contributes to the development of scientific methods that solve human body pose detection problem and therefore solve many real-world problems like detecting distracted drivers.

1.4 Road Map

The rest of the thesis has four more chapters and is organized as follows.

Section 2: Describes different state-of-the-art methods of computer vision and machine learning. Literature review gives an overview of popular algorithms utilized in image processing and human detection. The section also introduces innovative classification methods based on machine learning like neural networks.

Section 3: Lists the methods and steps used to achieve the goal of the thesis. Gives an in-depth overview of how the different methods work and can be utilized together to achieve the goal of human body pose detection.

Section 4: Describes dataset and testing strategy. Explains data augmentation and different neural network models used. Shares experimentation results and steps taken to improve the results while experimenting with different models.

Section 5: Conclusion and future research perspectives are presented in this chapter.

2 State-of-the-art

2.1 Introduction

For years researchers have been investigating and innovating different applications related to people detection [38] and recognition [8]. The main application for this research has been in advanced driver-assistance systems. In addition to detection and recognition of people, the focus has also been on recognition of human body poses such as standing or walking. In recent 2017 paper [15] the authors presented a model that includes both human body poses recognition and also the lateral speed. The recognition part was achieved by hierarchically separating all features that were extracted from the spatial body language ration. Another example is in 2013 paper [13], which based its approach on performing a simple classification of human body pose based on HOG and SVM. The focus in the paper was to provide an algorithm that could be used in real-time applications. Due to that constraint, the algorithm couldn't be complex with respect to time taken to compute. The performance of the system was acceptable, but not satisfying - mainly due to the small training dataset.

There is a general manner of using HOG feature by introducing randomized trees for human body pose detection. This manner was illustrated in 2008 paper [25], where the effort was put on aligning the training images using 3D Mocap data. The aligned image and training class definition was injected into random forest (RF) generation algorithm. This process allowed performing human detection and human classification. In the end regression learning on the training data was combined with human detection to estimate the poses. The results were encouraging and techniques used could be used in many different scenarios.

In recent years in addition to recognition based on SVM there have been many papers that introduce machine learning to the computer vision problems and for the recognition, the task uses deep learning methods. In 2017 paper [17] the authors presented two methods for human body pose recognition based on deep learning. First methods concentrated on extracting and detecting body parts using a convolutional neural network (CNN). The second method incorporated an optimized neural network for mapping human body poses to the right estimation by using a similarity evaluation between dimensions. Both models had different cases where the outperformed each other. In another 2017 paper [34] the authors again demonstrated the use of deep learning to solve human body pose recognition task. The authors' method used binarized normed gradient features to extract objectiveness using saliency map. For making the final recognition, a CNN learnt the hierarchies of the features and made predictions based on learnt.

2.2 Computer vision

Computer vision, in general, is the venture of automating and incorporating an extensive variety of procedures and representations utilized for vision perception [16]. Knowledge from different fields is joined together in one activity with an end goal of giving computers high-level understanding of digital images. A digital image itself is just a numeric representation of a normally binary 2-dimensional image. Computer vision task here is to implement methods that identify and pass on the information that constitutes an image by processing the image using mathematical operations. The output of these mathematical operations can be a set of image characteristics or simply an image.

Computer vision continues to grow, in the past years more than ever, in useful applications. The phrase *computer vision* was originally formulated to describe the general goal, which is to enable computers with attached cameras to intelligently see images and recognize the same things from images that humans do. In an example, humans are able to visually distinguish banana from apple in a picture of a fruit basket.

To see like humans, computers need algorithms. Algorithms enable computers to process the pixel values and identify the most important features from an image. This is a big challenge and many people spend their whole careers trying to teach computers to find faces in photos automatically or solve similar tasks.

Fortunately, much progress has been made in computer vision recently. Competent algorithms exist for many types of problems. These algorithms can now perform reasonably fast given advances in heterogeneous computing. However, computer vision is still far from being fully developed. Computers are unable to see as well as humans in most real-life scenarios. Computer vision is most powerful when it is coupled with machine learning to find objects in incoming imagery (computer vision) as well as match those objects against a large body of information (machine learning). This is also what this thesis does - couples computer vision with machine learning to classify human body poses from input images.

Computer vision has found applications in many different fields. The innovation of computer vision has automated processes in electronics component manufacturing, quality textile production, metal product finishing, glass manufacturing, machine parts, printing products and many more. There are many real-life applications of which one of the most important is object detection [21]. Most recently computer vision has solved the problem of quality inspection of fruits and vegetables by automating the inspection processes. In 2014 paper [37], the authors utilized the most important appearance characters of fruits and vegetables to set up a computer vision system that can monitor these characteristics and decide on the quality of the fruit or vegetable based on that information. The generic setup of the computer vision system can be seen in figure 1. Another big challenge under object detection is detecting humans.

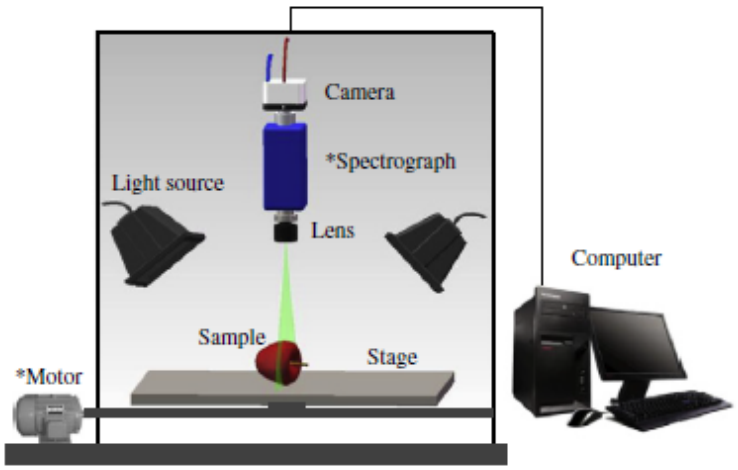


Figure 1. Generic computer vision system. [37]

2.2.1 Human detection

Human detection is one of the better-known problems that computer vision tries to solve. The problem solution consists mainly of the design and training of a human model based on characteristics, which are in example dimensions and silhouette, and the adjustment the human model to the possible candidates to be humans on the given image. Candidates found from the image that adjust to the human model are detected as human while all the other candidates are identified as not human. Almost all of the state-of-the-art approaches are based on appearance information [30] [33], but some also utilize motion information to increase detection accuracy. They get the motion information through different tracking algorithms. The general human detection approach is visualized on figure 2.

Different appearance-based methods can be divided into groups based on the model used. The simplest of methods use simplified human models, which only process a region or shape [35]. Due to the very simplistic human model used, these tend to have low complexity and don't support partial occlusions or human body poses variations. These models also lead to low accuracy. More complex models [33], however, support also partial occlusions and pose variations. The more complex models also make decisions based on a larger variety of different findings, so they are more reliable than simpler human models.

In addition to human body/shape information, a very helpful part of the human body for human detection is the face. Face recognition, using an example neural networks, is a very accurate technique for human identification in the case of cooperative behaviour. There are, however, problems with face recognition - in an example when faces cannot

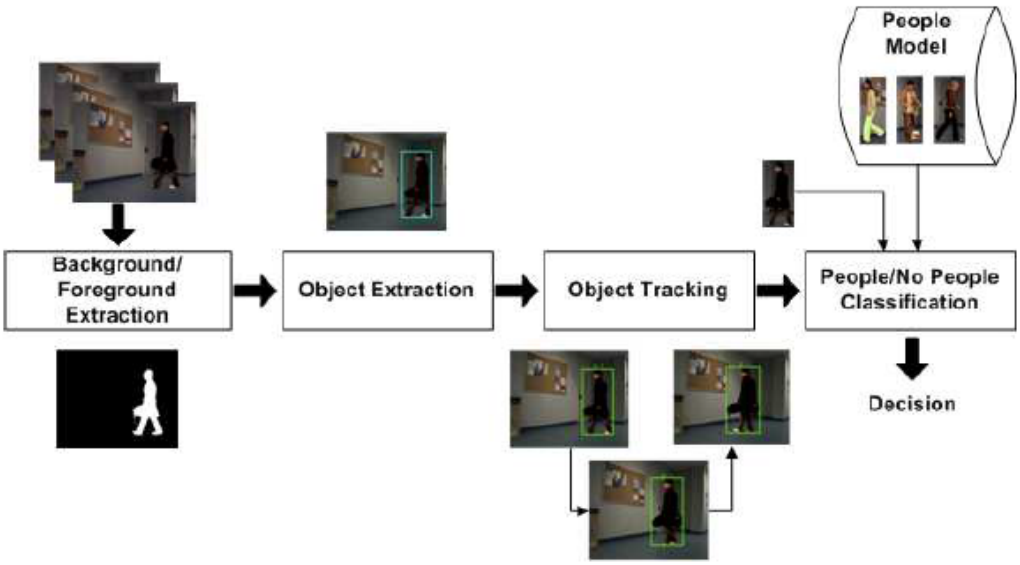


Figure 2. Human detection system visualized. [12]

discriminate people due to the small size and significant changes the recognition can fail. However, it is a very active field and receives constant improvements. In an example to overcome the difficulties of detecting faces of small sizes, recognition by gait analysis can be used as shown by the results that achieve an excellent accuracy of 97.4%, which favours its applicability in an access control context [10].

2.3 Machine learning

Machine learning is one of the most powerful ways of performing analysis in computer vision with methods like SVM [32] and RF [7] receiving some of the highest success rates. These methods receive high success rates in many cases [29], but their overall performance often falls short of the very high accuracy required for fully automated systems. Neural networks and deep learning try to solve this problem by learning features themselves. This approach increases accuracy given large enough dataset for training.

Machine learning techniques in computer vision mainly fall into two categories - older SVM or RF methods and newer neural network methods.

2.3.1 Support vector machine and random forest

Classifiers like SVM and RF fall into the category of linear or non-linear classifiers based on how they have been implemented. Figure 3 gives an overview of data classification

by linear and non-linear classifiers.

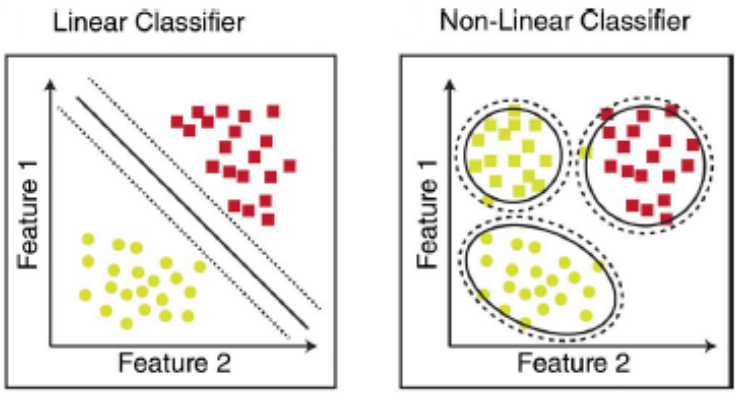


Figure 3. Schema representation of linear and non-linear classifiers. [14]

Linear classifiers work when the decision boundaries are linear as the name suggests. The classifiers model the boundaries directly by predicting the class of a feature without giving any consideration to the joint probabilities of classes and features themselves [23]. Linear SVM is a good example of a linear classifier as it uses a hyperplane that simply separates two classes by trying to maximize the distance between hyperplane and chosen data points. Chosen data points are made up of the data points from the opposite classes that are closest to each other. The hyperplane itself is generated only based on manually classified sample data points, images, that the SVM classifier assigns a class based on their positions relative to the hyperplane [31].

Non-linear classifiers are used when decision boundaries are not linear. In this case, simple linear classifier won't work and complex non-linear classification algorithms are required. For example, the SVM classifier described above can be used to produce non-linear decision boundaries if simply a non-linear kernel function is used. The idea of a kernel function is to transform the feature space and after the transformation fit the SVM model for classification. This enables the creation of non-linear decision boundaries while maintaining the original feature space [31]. As another example, RF is also a non-linear classifier. RF uses decision trees where sample features have been mapped to classes. The decision tree, in this case, is called a classification tree as it forms branches of features where a combination of features will end up pointing to one class in the tree. RF often faces the issue of over-fitting. To counter over-fitting the RF classifiers uses random feature combinations with uncorrelated decision trees.

2.3.2 Neural network

A general consensus is that the information contained in images raw images is too much for processing with machine learning methods. For this reason, much effort in computer vision neural networks involves precomputation of image features. This involves an example the preprocessing of images with filters. This aids the computer in the detection of high contrast areas on an image and should make enough information available for representing classes of objects whilst drastically reducing the amount of information on the image pixels compared to the original set of pixels [29].

The output from the preprocessing is passed into the classifier, where classes are separated from each other efficiently. When creating the classifier, the choice of the dataset is left to the user and is often limited to existing sets, which are used in scientific literature. These datasets might not provide the classifier's learning algorithm with best data description, which can lower the accuracy of the model in whole. The main issue with the learning algorithms is that they mainly learn to solve one task. They will eventually perform very well with one task but fail with another. Due to that, there is a motivation in the scientific community to produce more general learning methods.

One of the earliest general learning methods includes the artificial neural network (ANN). ANN simulates neuron-like connections where inputs are transferred through learnt functions to outputs. Similar to the structure of human brain these transferrals represent a set of activations propagating through a network like structure. In general, ANN has three layers - input layer, hidden layer and output layer. Modern deep learning utilizes neural networks built on this simple structure. Different advanced neural network structures are all mainly extensions of the ANN, where additional layers have been added. As many more layers of artificial neurons are added, the list of layers becomes deep and this is where the term deep learning originates from. The additional layers increase the ability of the neural network to discriminate between classes with better results [22]. CNN advances the general structure further by replacing additional neuron layers with convolutional layers instead. The convolutional layers detect features from input images by using different filters and then feed these found features into the traditional ANN based neural network layers for classification. For example, while the initial layer of convolution might simply compute features like edges and corners, while deeper layers might contain complex features representing real-world objects [36]. An example of CNN being used to detect an ear of wheat from an input image is visualized on figure 4.

CNNs have quickly gathered wider popularity in computer vision scientific communities [39] and modern CNNs used by the scientific community normally includes many layers making the training very complex requiring a large dataset to achieve acceptable results [19]. However, if successfully trained, then CNN's accuracy is unrivalled [28].

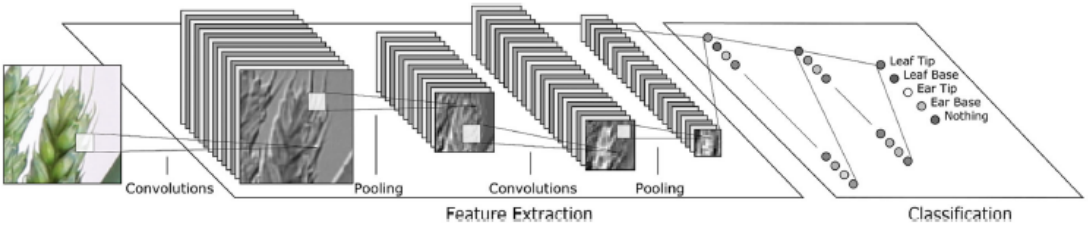


Figure 4. A simplified example of a CNN architecture operating on a fixed size image of part of an ear of wheat. [24]

2.4 Conclusion

This chapter provides a brief explanation of state-of-the-art methods and algorithms used in computer vision and computer vision based machine learning. Computer vision mainly tackles feature detection problem - how to make computer detect most important features from images. A big research challenge is people detection. There are many methods for detecting people from images, but most approaches involve detecting features like hands, head or legs to detect human. These approaches also face big obstacles like detecting obscured people or people that are only partially on the image.

In recent past machine learning coupled with computer vision has been gaining wider popularity in the scientific community. Machine learning can be used to train either linear/non-linear classifiers or neural networks to detect features from input images. Preprocessed images can be fed to classifiers are 2-dimensional pixel arrays and the classifiers make a decision as to what features exist on the input image. The fields face multiple challenges as in general the images are quite data-heavy. Therefore processing them in machine learning is time- and resource-consuming task. Main challenge resides in finding a balance between accuracy and speed. With speedier classifiers, more simple architecture is used which leads to lower accuracy. For higher accuracy, more resources are used and the training is more time-consuming, which doesn't enable the real-time use of the classifiers.

Computer vision coupled with machine learning has come a long way and is gaining more and more popularity in the scientific community. However, there are many challenges to be tackled multiple of which are discussed in this thesis.

3 Methodology

3.1 Introduction

The proposed approach for human body poses recognition is based on shape analysis of human silhouette. The method can be divided into three main parts as highlighted in figure 5:

a) Detecting human on input image;

b) Extracting human silhouette from image;

c) Classifying human body pose based on silhouette using neural networks.

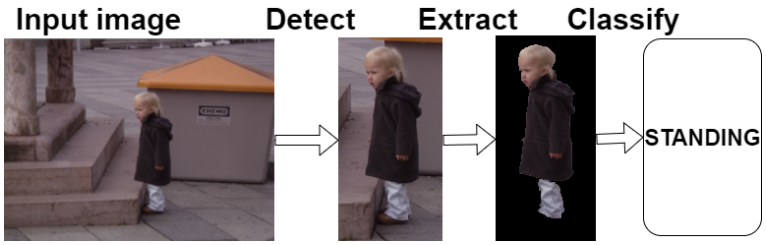


Figure 5. Methodology step by step.

First part is built using OpenCV package [6], which contains a pretrained HOG and linear SVM model that can be applied to detect human on individual images.

The second part uses GrabCut image segmentation method to cut out a human silhouette from the image.

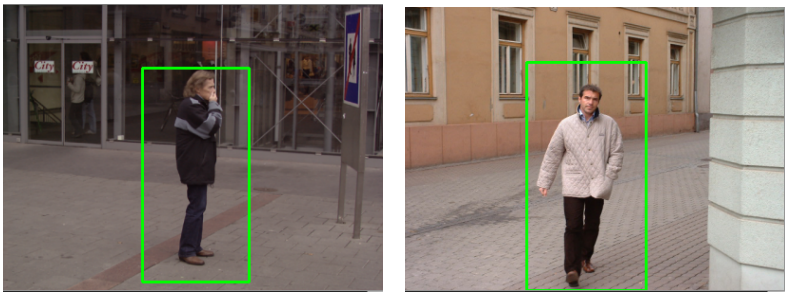
The third part consists of training and testing neural networks to classify human body poses with acceptable accuracy. In addition, PyBrain library is used to aid in creation and optimization of a simple neural network [27]. More advanced convolutional neural network is created using TensorFlow with Keras library.

The first and second part recognize and cut out silhouettes from images, which creates input training data for the third part which is classification. In order to create training data, a dataset of images is compiled, where for each image class is given based on human's body pose. After training the neural network with the silhouettes and corresponding body poses, the neural network is tested with a new dataset of images. After testing, an error rate is computed in order to evaluate the accuracy of the neural network in classifying and recognizing the human body poses.

3.2 Detection

At a structural level, a human has a head, arms, torso and legs. Computer vision can be used to exploit these basic traits to detect humans from a random image.

Concerning detecting human from an image, we rely on OpenCV package, which ships with pretrained HOG and linear SVM method. Despite the fact that idea of applying HOG descriptor for object recognition is nearly a decade old, it is still used a lot and shows good results. The HOG method was suggested in [9], where it was explained that HOG image descriptor and a linear SVM could be used to train highly accurate object classifiers. Therefore, for our preliminary step of detecting pedestrians, we used OpenCV library, which has pretrained descriptors to be applied for human detection. The process starts by using images of interest that are all resized to 480x640 pixels if necessary. As HOG uses sliding windows and image pyramids, the detection would become very slow with high-resolution images.



(a) Example of standing human detected and highlighted with a green rectangle.

(b) Example of walking human detected and highlighted with green rectangle.

Figure 6. Overview of human being detected on image.

An issue arises with above method – sometimes multiple and overlapping bounding boxes are detected for each human. To effectively use the found bounding boxes in our steps of problem resolving, we need to extract a clear bounding box for each human on the image. Hence, we introduce non-maxima suppression method [11] in order to combine multiple bounding boxes into one proper bounding box. This method suppresses bounding boxes that overlap with a significant threshold. This ends up with one bounding box that is drawn onto the original image to highlight the detected human on an image. The figures 6(a) and 6(b) illustrate a good example of the resulting images after applying the above mentioned methodology. The bounding boxes found on the images will be used in next step for extracting the silhouettes of the humans.

3.2.1 Histogram oriented gradients descriptor

HOG is a feature descriptor used to detect objects in computer vision and image processing. What the descriptor essentially does is counts occurrences of gradient orientation in a detection window. The implementation of the HOG descriptor algorithm is visualized on figure 7 and works as follows:

1) Input image is split into cells;
2) Each cell has its HOG directions computed, which are essentially edge orientations for the pixels in the cell;
3) Each cell is discretized into angular bins according to the gradient orientation;
4) Each cell's pixel adds a weighted gradient to its corresponding angular bin;
5) Adjacent cells are grouped into blocks, which is the basis for grouping and normalization of histograms;
6) Normalized group of histograms represents block histogram and set of these block histograms represents the descriptor.

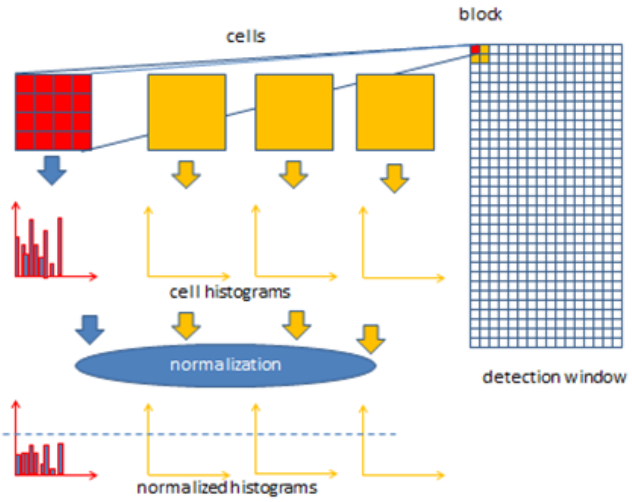


Figure 7. HOG descriptor algorithm visualized. [2]

3.2.2 Linear support vector machine for human detection

OpenCV provides pretrained linear SVM that has been trained with HOG descriptors to detect humans from images. The linear SVM uses hard-negative mining. For each image and each possible scale of each image in the negative training dataset, the sliding window technique is applied and a window is slid across the image. At each window, HOG descriptors are computed and classifier applied. Moreover, in case the classifier incorrectly marks a given window as an object, the feature records vector associated it with the false-positive patch along with the probability of the classification. This approach is called hard-negative mining.

The false-positive samples found during the hard-negative mining stage are taken into account with their sorted confidence. The classifier is then re-trained using these hard-negative samples. The trained classifier can be applied to test dataset. For each image in the test dataset, and for each scale of the image, the sliding window technique is applied. In addition, for each window HOG descriptors are extracted and classified. Next, if the classifier detects an object with sufficiently large probability, the bounding box of the window is recorded.

3.3 Extraction

The human silhouette extraction from video feeds can be a straightforward process with the assumption that the environment is under control. We can rely on the moving object when comparing the frames and detect and extract the human body. However, this task becomes a little bit challenging when working with single images. Hence, in our case we used GrabCut algorithm [26] because it has good results rate. After detection phase, once we are sure that the human is in the segmented image, the image can be divided into two parts – background and foreground. The foreground is the bounding box from the previous step where human has been detected. The background is the part of the image outside of the bounding box. GrabCut utilizes this information.

The algorithm is introduced as a solution for foreground extraction. The user draws a bounding rectangle as done in the previous section in this paper. The foreground object (human) must be completely in the rectangle. The algorithm segments foreground iteratively to get the best result. The algorithm segments pixels into foreground and background. Background pixels are coloured black and the end result is an image where the foreground object is nicely brought out. On the images produced by previous human detection step, where bounding rectangle has been found, GrabCut compares the colours and structure of background to try to remove background elements from foreground leaving only human silhouette onto the image to work with.

This is well shown by the examples highlighted by figures. 8(a) and 8(b).

(a) Example of standing human silhouette extracted.

(b) Example of walking human silhouette extracted.

Figure 8. Overview of human silhouette extracted from image.

3.3.1 GrabCut algorithm

The GrabCut algorithm is an extension of the graph-cut algorithm that proposes a more powerful and iterative version of the optimization. The graph-cut algorithm addresses the segmentation given an initial trimap T that consists of T_B (set of background pixels), T_F (set of foreground pixels) and T_U (unlabeled pixels).

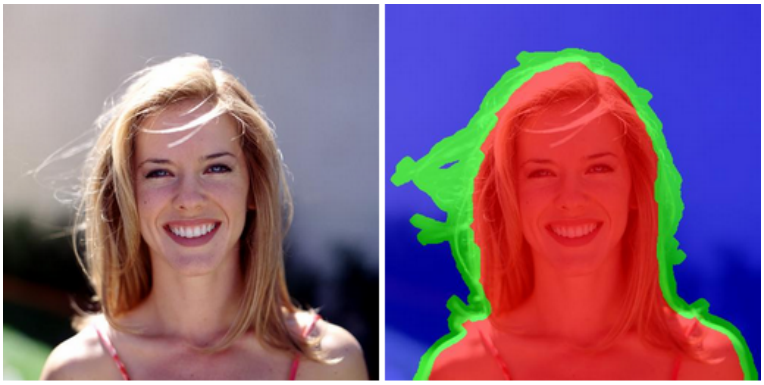


Figure 9. Image (left) and trimap overlaid on image (right). [3]

Figure 9 shows an example of a trimap overlaid on a photo, where pixels have been assigned colours accordingly - blue for the background, red for foreground and green for unlabeled. As seen and proven in [5], creating a trimap and then running the min-cut algorithm yields two groups of vertices with labelling that minimizes the cost function. The algorithm itself takes an input image that is defined as an array of grey values $z = (z_1, ..., z_n, ..., z_N)$ and the segmentation of the image is expressed as an array $\alpha = (\alpha_1, ..., \alpha_n, ..., \alpha_N)$, which gives to every pixel a value in a range from 0 to 1

(where 0 means background and 1 means foreground). The parameter θ consists of two histograms of grey values both of which represent pixel value distribution for foreground and background labels respectively:

$$\theta = h(z; \alpha), \alpha = 0, 1 \tag{1}$$

The histograms are built from labelled pixels from trimap regions T_B, T_F and the histograms are normalized to sum to 1 over the grey level range.

The segmentation task is to infer the unknown opacity variables α from the given image data z and the model θ. To solve the task, an energy function E is defined (minimizing it corresponds to good segmentation):

$$E(\alpha, \theta, z) = U(\alpha, \theta, z) + V(\alpha, z) \tag{2}$$

In the energy function, the data term U evaluates the fit of the opacity distribution α to the data z given the histogram of the model θ. The smoothness term V encourages coherence in regions of a similar level of grey. The segmentation is estimated as a global minimum of the energy function and computed by the min-cut algorithm. GrabCut enhances the graph-cut mechanism summarized above by mainly using colour instead of a grey level by replacing the monochrome image model by a Gaussian Mixture Model (GMM) and replacing the min-cut algorithm by a more powerful iterative procedure that alternates between estimation and parameter learning. With the GrabCut algorithm, the input image consists of pixels z_n in the RGB colour space. To enable the use of RGB colour space, GMMs are used (one for the background and one for the foreground). This introduces an additional vector $k = k_1, ..., k_n, ..., k_N$ in the optimization framework, assigning to each pixel a unique GMM component (either from the background or from the foreground, depending on α = 0 or 1). With these changes, the energy function for segmentation becomes:

$$E(\alpha, k, \theta, z) = U(\alpha, k, \theta, z) + V(\alpha, z) \tag{3}$$

The data term $U(\alpha, k, \theta, z)$ is defined taking into account the colour GMM models, encouraging pixels to belong to the foreground or the background depending on how well they fit in the corresponding GMM. The smoothness term $V(\alpha, z)$ is equivalent to the one in the graph-cut algorithm, only using the Euclidean distance in RGB colour space instead of in the monochrome space. The smoothness term encourages similar and neighbouring pixels to belong to the same region (just like in graph-cut). The minimization scheme in GrabCut works iteratively, allowing an automatic refinement of the opacities α (which define the pixels belonging to foreground and background) as newly labelled pixels from the T_U region of the initial trimap are used to refine the colour GMM parameter θ.

3.4 Classification

The previous steps – detecting a human from an image and extracting silhouette are applied to hundreds of images. In this thesis, two poses are classified – standing and walking. The dataset for training classifiers consists of 226 images of silhouettes, which are augmented to form a dataset of 2260 images of silhouettes in total.

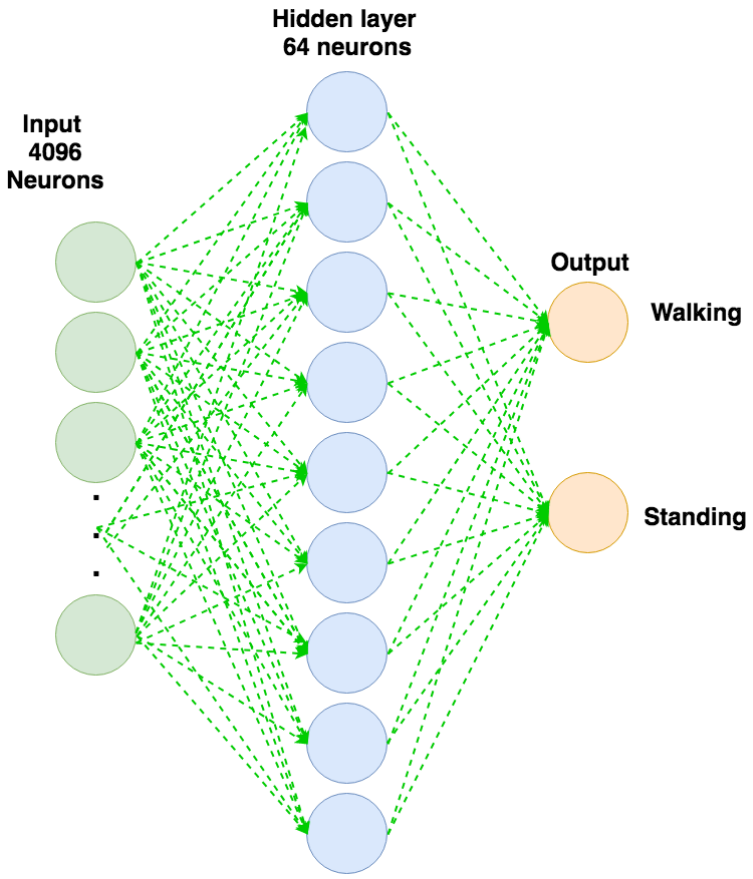
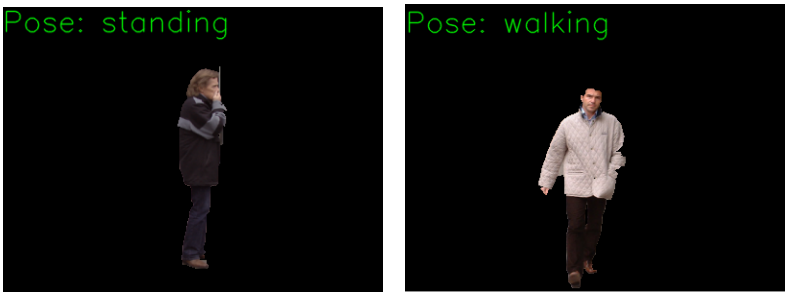


Figure 10. Neural network structure visualized.

The simple feed-forward neural network is created using PyBrain after the creation of the initial dataset of 4096 inputs.

The network has one hidden layer consisting of 64 neurons. The output has 2 binary neurons with a float value categorizing the image based on the pose (one neuron for walking and second for standing). The neural network is visualized on figure 10. We resize input silhouette images to a size of 64x64 pixels and make the RGB images into grey-scale images. The training data is fed to neural network inputs (4096-pixel values). The neural network is trained for varied number of epochs after which it is tested with

test dataset. Figures 11(a) and 11(b) give example of neural network output. The neural network outputs classification value, which is translated into text and written on the input image of a human silhouette.



(a) Example of standing human body pose correctly being classified by the neural network.

(b) Example of walking human body pose correctly being classified by the neural network.

Figure 11. Overview of human pose being classified based on silhouette.

After initial tests the simple feed-forward neural network is replaced with a CNN and same classification steps run using CNN.

3.4.1 Convolutional neural network

Trying to improve the results of a simple neural network, a convolutional layer is added to the simple neural network. This essentially forms a CNN. The CNN involves four major steps:

a) Convolution;

b) Pooling;

c) Flattening;

d) Full connection.

The CNN is a sequential network containing multiple layers. The first step is to perform convolution on the training images, which is done by the first convolutional layer. The layer is 2-dimensional as the images are also 2-dimensional pixel data arrays. After the convolutional layer, a pooling layer is used, which performs the pooling operation using a max-pooling function. Max-pooling is used because for each region of interest, the maximum pixel value is needed. After the first convolution and pooling, there is the second layer of convolution and pooling with the same parameters. The output from

pooling is flattened from the 2-dimensional array into a 1-dimensional array, which can then be fed into the feed-forward neural network accepting 4096 value array.

The convolutional layer uses 32 filters, where each filter is in the shape of 3x3. The input to the convolutional layer is a 64x64 pixel coloured image in an RGB format and the layer uses rectifier function for processing.

The pooling layer performs pooling operation. Convolutional operation outputs multiple feature maps per image and pooling operation is run on this output. Pooling layer takes in the feature maps from convolutional operation and uses a 2x2 matrix to mimize the pixel loss while getting a precise region around feature locations. The output from pooling layer is finally flattened to get a 1-dimensional single vector, which is then fed to the hidden layer just like in simple feed-forward network introduced before.

3.5 Parameter tuning

Each method used takes input parameters. The input parameters greatly decide the effectiveness and accuracy of the method itself and in general, depend on the task and input at hand. The parameters have been tuned by trial and error to achieve the best results.

3.5.1 Detection parameters

Human detection leverages pretrained linear SVM and HOG detector that ships with OpenCV. The module consists of a function *detectMultiScale*. This function takes an input image and different parameters that can be given to it. The function outputs coordinates of rectangles where it potentially detected people. Tuning the parameters according to current task's needs is therefore very important. The parameters given to this function can do the following:

- Increase the number of false-positive detections by reporting a rectangle on an image contains a human although it does not;
- Result in getting no detection at all given an image with a human on it;
- Greatly affects the speed of the detection process.

The general trade-off with the parameter tuning is between speed and accuracy. In this thesis, speed is not that important as the application is not real-time. The greater importance goes to accuracy and for that greater time for processing can be given.

The main parameters affecting the people detection in current task are *winStride*, *padding* and *scale*. Below explains their meaning:

- **winStride** is the step size in the x and y coordinates of the sliding window used (explained in 3.2);

- **padding** controls the amount of pixels the region of interest is padded with prior to HOG feature vector extraction and SVM classification;
- **scale** controls the size of the image pyramid, which allows detection of people in images at multiple scales.

In this thesis, the raw images input to detection are all from GRAZ-01 [1] dataset, where all images are uniformly 480x640 pixels. This is already a decent height/width and ratio. With bigger images, one should consider resizing as the bigger the image, the bigger the number of sliding windows and therefore the processing of a single image can take a big chunk of time. In this case, no resizing is needed.

For *winStride* parameter value of 4x4 pixels were used. This guarantees that the sliding window will find the smaller people in the background and has a lesser chance of not detecting a human on the image. Having a small steps increases the processing time, but increases the accuracy.

For *scale* parameter value 1.05 is used. This value was found by trial and error as with higher values the detector didn't notice smaller people in the background of an image. With smaller values, however, the detector caused too many unwanted rectangles of interest to appear.

For *padding* parameter of 8x8 pixels were used. This causes the detector to detector people that are partially on the image and also help reduce cases where a human doesn't get detected because only part of him/her was initially found inside the region of interest.

As noted, often the detector finds overlapping rectangles of interest on the same human. These should be merged into one region of interest and this can be done by non-maxima suppression explained in 3.2. Non-maxima suppression is run with overlap threshold 0.65, which means it will merge together all rectangles of interest where their space overlaps 65%.

3.5.2 Extraction parameters

In the detection phase, the regions of interest on an image that includes a human are extracted. Knowing the region of interest we can compare the pixel values in the image overall with the pixel values in the region of interest. Essentially it is comparing a rectangle on an image including human with everything outside the rectangle. It is important that the human is contained in the rectangle completely for perfect silhouette extraction. The GrabCut algorithm is used for the extraction purpose.

The GrabCut algorithm takes as input the region of interest rectangle coordinates *x1, y1, x2, y2* and the times to run the algorithm. A value of 10 iterations is used as it is getting best results without giving away too much performance.

For each region of interest, the algorithm runs 10 iterations and each time compares the region of interest (containing human) pixels to background pixels. All pixels deemed

to be background pixels are changed to value 0, which is black. The end result is an image with a black background with the silhouette itself being the only colourful part (pixel values other than 0).

3.5.3 Classification parameters

The classification task first involves a simple feed-forward neural network that is trained with back-propagation. The neural network is trained with a training dataset of 1800 images of silhouettes as explained in subsection 4.2. Once the neural network is trained, it is tested with a dataset of 460 images of silhouettes and the error rate is recorded.

The first task is creating a classification dataset for training. Each of the 1800 silhouettes is resized to 64x64 pixels and turned into grey-scale images. Next, each sample is added to the classification dataset with a label indicating the body pose of the silhouette (in current case 0 for standing and 1 for walking).

Next, the feed-forward neural network is built. The neural network is initialized with 4096 input neurons as it corresponds to the 64x64 pixels of the input image. Through trial-and-error it was observed that larger input image with the larger amount of input neurons doesn't increase accuracy, but takes a heavy toll on the processing power of the neural network.

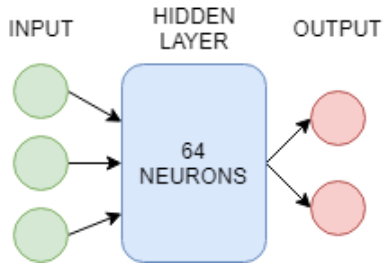


Figure 12. Structure of simple feed-forward neural network used.

Figure 12 shows the neural network structure. The neural network has one hidden layer with 64 neurons. Tests were carried out with two hidden layers consisting of 64 neurons and with one hidden layer consisting of 128 neurons. The simple approach of 64 neurons and one hidden layer proved to offer similar accuracy compared to the other tested neural network structures, but being more efficient. The output from the 64 neurons in hidden layer goes to two output neurons. Count of output neurons is based on the amount of body pose classes. In the current case, two body pose classes are observed and therefore the count of binary output neurons is two as well. The neural network uses softmax layer as it is best fitted for multi-classification in a logistic regression model. Added value is that the probabilities sum will be 1.

The created neural network is trained with a back-propagation trainer with the classification dataset created previously. Back-propagation trainer trains the parameters of a module according to a supervised dataset (potentially sequential) by back-propagating the errors (through time). The input parameters to the trainer are very important as they decide the effectiveness of training the neural network. The trainer takes as input the *momentum*, *learning rate* and *weight decay*. The parameters affect the trainer as follows. The learning rate gives the ratio of which parameters are changed into the direction of the gradient. The learning rate decreases by a factor which is used to multiply the learning rate after each training step. The parameters are also adjusted with respect to momentum, which is the ratio by which the gradient of the last timestep is used. Weight decay corresponds to the weight decay rate, where 0 is no weight decay at all. The trainer is run for 100 epochs as through testing it was deemed that after 100 epochs of training the error rate doesn't increase greatly anymore.

The trained neural network is tested with a labelled dataset of test data and the neural network results are compared with the labelled results to find the error rate of the neural network.

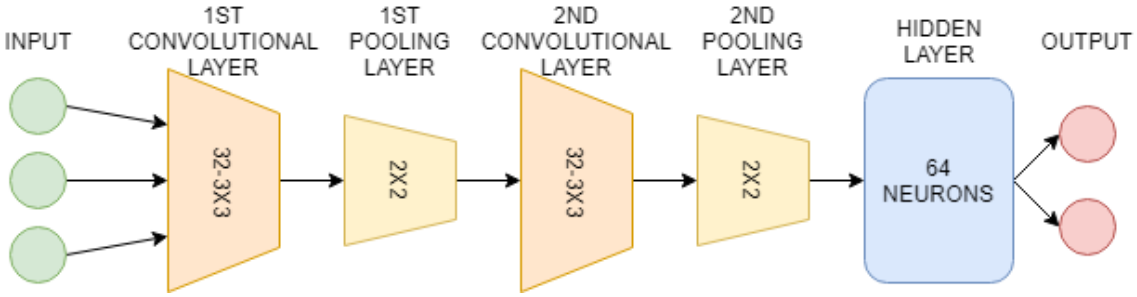


Figure 13. Structure of CNN used.

To try to improve the results further, the convolutional layer is added to the simple neural network as explained in section 3.3. Figure 13 highlights the CNN structure. The neural network consists of two sets of convolution and pooling. After the two sets of steps are completed, the resulting values are fed to the hidden layer and outputs provided.

3.6 Conclusion

The main part of this chapter illustrates details of the methods used for detecting people, extracting their silhouettes and classifying their body poses based on the silhouettes acquired. The chapter covers the methods used in each step and explains the used algorithms in detail.

The methodology includes the use of linear SVM, GrabCut algorithm and neural networks. Linear SVM that has been pretrained with images of people, is used to detect

people from input images. The GrabCut algorithm is then used to extract silhouettes of detected people by comparing the human pixel values with background pixel values and cropping out all detected background pixels. This way all background around the silhouette is blacked out. The silhouettes are manually then labelled according to body poses and the neural network is trained with the manually labelled training data. Once trained the neural network is tested with test images to find out the real-world accuracy of the trained network.

4 Experimentation

4.1 Introduction

As discussed under methodology, the framework outlined in this thesis for human body pose detection consists of three different steps with each step using different algorithms/methodologies. For human detection task, we are using OpenCV with built-in pretrained linear SVM that has been trained to detect people from images. For silhouette extraction, we use GrabCut algorithm implemented in Python to extract the background from foreground (human silhouette) and black out the background. For body pose detection initially, PyBrain library is used to create a simple feed-forward neural network that is trained with the silhouettes and later tested with test data to find the accuracy. In addition to the simple feed-forward neural network, we are utilizing TensorFlow with Keras to implement a CNN essentially adding convolutional layers to the existing simple neural network. CNN is best suited for image processing tasks. The results achieved with a CNN are compared to the simpler feed-forward neural network. To increase the success of training the neural networks, data augmentation is used. For each individual silhouette, augmented data is generated using different methods of cropping, resizing, etc. All results achieved by the different models of neural networks are compared to each other and the most accurate model is found out.

4.2 Dataset and testing strategy

We use classic popular people dataset GRAZ-01 [1] to apply the methods. GRAZ-01 is a popular people dataset consisting of images often used in research papers dealing with human detection.

The dataset contains a variety of different people in different situations providing a good case for training and testing computer vision algorithms. Each image from the dataset passes first two steps: detection and extraction of a human silhouette. The end result is a processed dataset of images of human silhouettes that can be used for training and testing different neural networks. Figure 14(a) gives an example of raw image from GRAZ-01 dataset and figure 14(b) shows the same image after passing the detection and extraction steps.

The output images of silhouettes are manually classified based on if the human on the image is standing or walking. The dataset of processed images of silhouettes contains 226 images. The neural networks are trained to classify two poses - standing and walking. The silhouettes dataset respectively contains 125 images of standing humans and 101 images of walking humans.

The dataset is split with 80/20 ratio to training and test datasets meaning the 226 images are divided 180 images for training and 46 images for testing the neural network.

For each tested neural network structure and case of parameters, the testing strategy

(a) Image from GRAZ-01 dataset. (b) Imgage from dataset of processed silhouettes.

Figure 14. Example of generating silhouette data from images of GRAZ-01 dataset.

consists of validating the neural network after each epoch of training. After all epochs of training are finished, the neural network is validated against the test dataset. It is important to note that as 180 images for training are far too little, data augmentation is used to generate a larger dataset of training data. However, during validation with testing data, no data augmentation is used on the test data to provide fair results.

4.3 Data augmentation

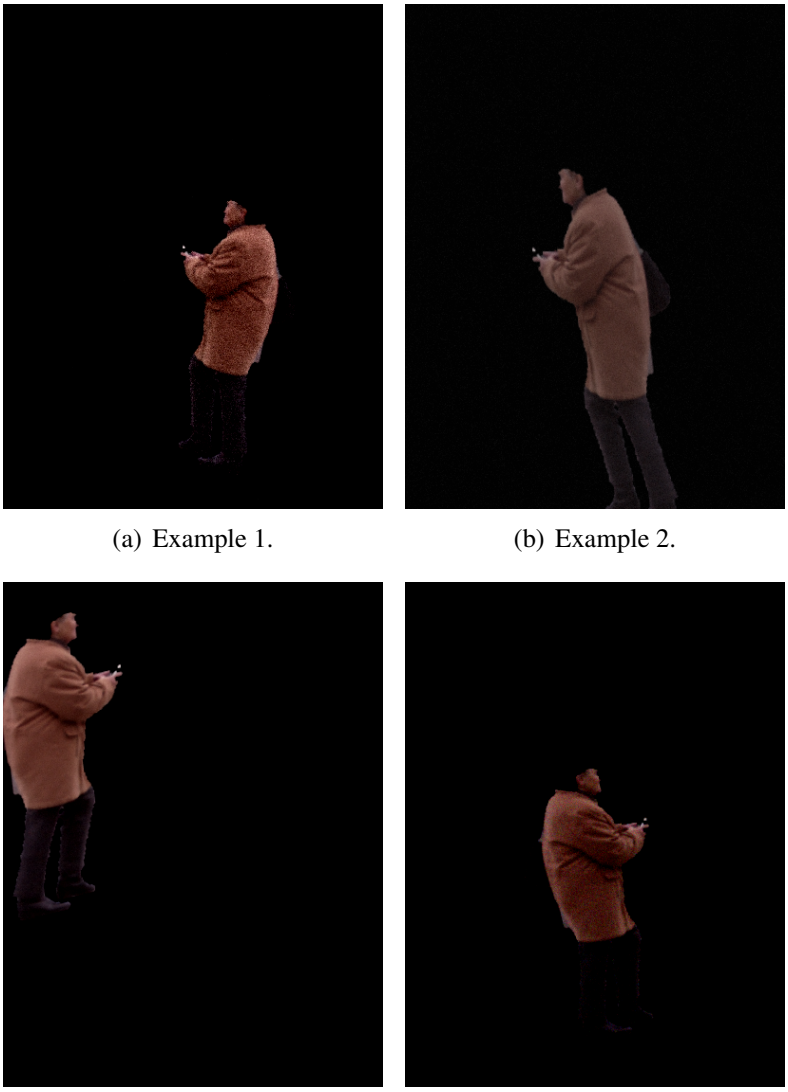
As the dataset of 226 images, in general, is too small for properly training a neural network, data augmentation is used to make the dataset more varied. To each silhouette artificial noise is added before being fed to the neural network. A random combination of 10 augmented images is created of each silhouette image. Figures 15(a), 15(b), 15(c) and 15(d) give an example of data augmentation. Data augmentation makes the 226 image dataset into a 2260 images dataset, which means there are over 1000 images for both classes. This is considered a good amount of data for initial neural network training.

The data augmentation applied involves the following process:

a) Horizontally flipping image with 50% probability;

b) Randomly cropping image;

c) Blurring image with 50% probability and adding random blur factor between 0 and 0.5;



(a) Example 1. (b) Example 2.

(c) Example 3. (d) Example 4.

Figure 15. Examples of single silhouette being used to generate augmented data for neural networks to learn upon.

d) Strengthening or weakening the contrast in image with 50:50 probability;

e) Adding Gaussian noise with 50:50 probability of applying only noise per pixel or applying noise per pixel and channel;

f) Making image darker or brighter by random factor;

g) Change color of image with 20% probability;

h) Apply affine transformations to image by scaling/zooming, translating/moving or rotating.

4.4 Results

Multiple neural network models were tested to achieve different results and compare them. Two datasets were used for testing - raw dataset of 226 images and augmented dataset of 2260 images. Two neural network structures were used for testing - simple NN with hidden layer and CNN where the convolutional layer is added to the simple NN. Different counts of epochs were tried when training the models. Based on the options above, 7 different combinations were generated and 7 different models were trained as per below:

- Simple NN with raw data and 10 epochs of training;
- simple NN with raw data and 25 epochs of training;
- Simple NN with raw data and 100 epochs of training;
- simple NN with augmented data and 10 epochs of training;
- Simple NN with augmented data and 25 epochs of training;
- CNN with augmented data and 25 epochs;
- CNN with augmented data and 100 epochs of training.

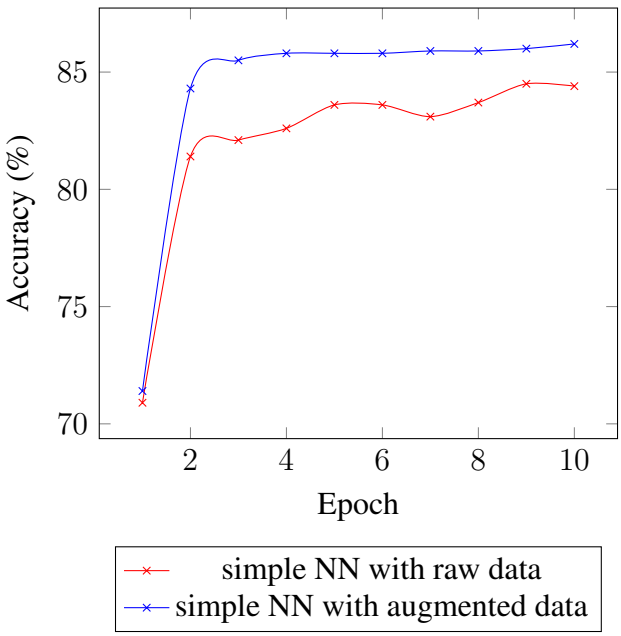


Figure 16. Accuracy of models based on validation after each epoch of training over 10 epochs.

Figure 16 highlights difference between raw data and augmented data. Two models of simple NN were both trained for 10 epochs and accuracy recorded upon validation after each epoch. The only difference between the models was the dataset used. Using raw data the accuracy upon validation was mainly between 81-83%. Using augmented data the accuracy upon validation was between 85-86%. Based on that using 10 times larger augmented dataset compared to smaller raw dataset increased model accuracy around 3%.

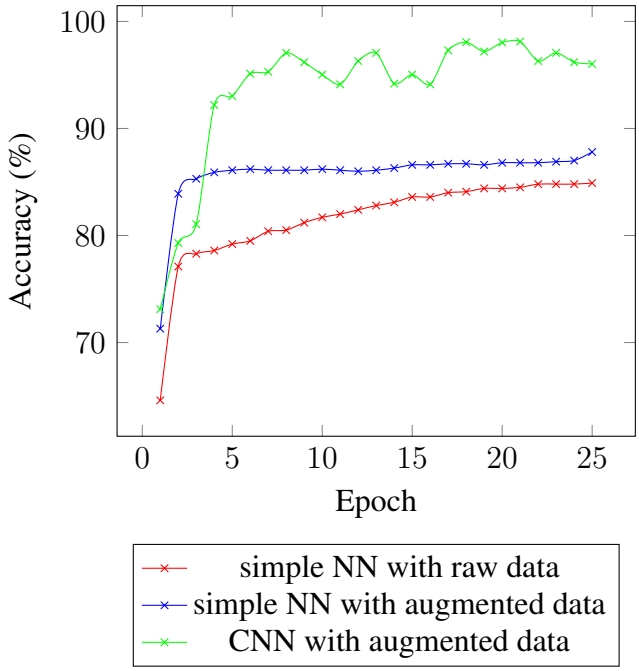


Figure 17. Accuracy of models based on validation after each epoch of training over 25 epochs.

Figure 17 compares the three most different models over 25 epochs of training. The simple NN with augmented data performs better than simple NN with raw, but the accuracy difference upon validation doesn't differ more than 5% between these models. Having introduced convolutional layer to the simple NN forming a CNN, the model outperforms other two models by far when trained with augmented data. The CNN reaches an accuracy of 93-97% upon validation. The other two models don't manage to get over 90% error rate even after 25 epochs of training. Difference between accuracy of simple NN and convolutional NN is at some points as high as 10%.

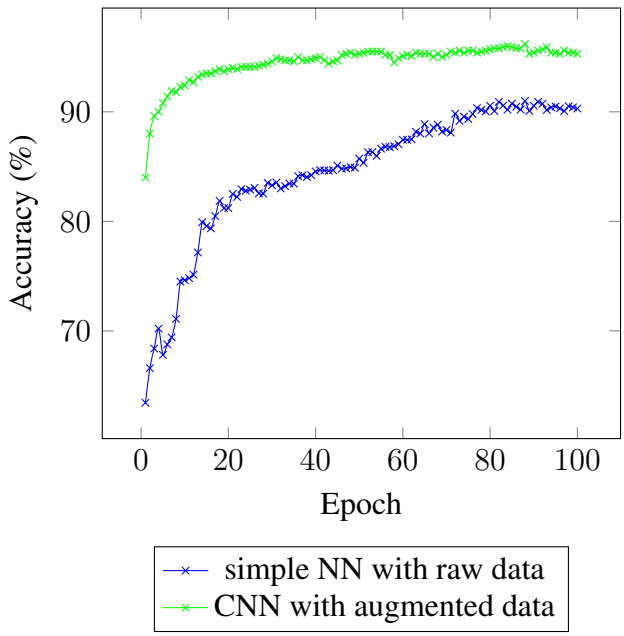


Figure 18. Accuracy of models based on validation after each epoch of training over 100 epochs.

Based on previous results the worst model is simple NN with raw data and the best model is CNN with augmented data. Next, the best and worst model were both trained for 100 epochs to see what accuracy upon validation can be achieved after multiple times more epochs of training compared to the earlier training of 10 and 25 epochs long. The simple NN with raw data almost reached 90% accuracy upon validation after around 80 epochs of training. After first epoch, the validation accuracy was however only 60%. CNN with augmented data outperformed the previous model by a large margin every epoch of training. The model reached an accuracy of 85-95% in the first 10 epochs of training and seemed to reach convergence after already 40 epochs of training. The last 60 epochs of training the CNN didn't improve the results drastically.

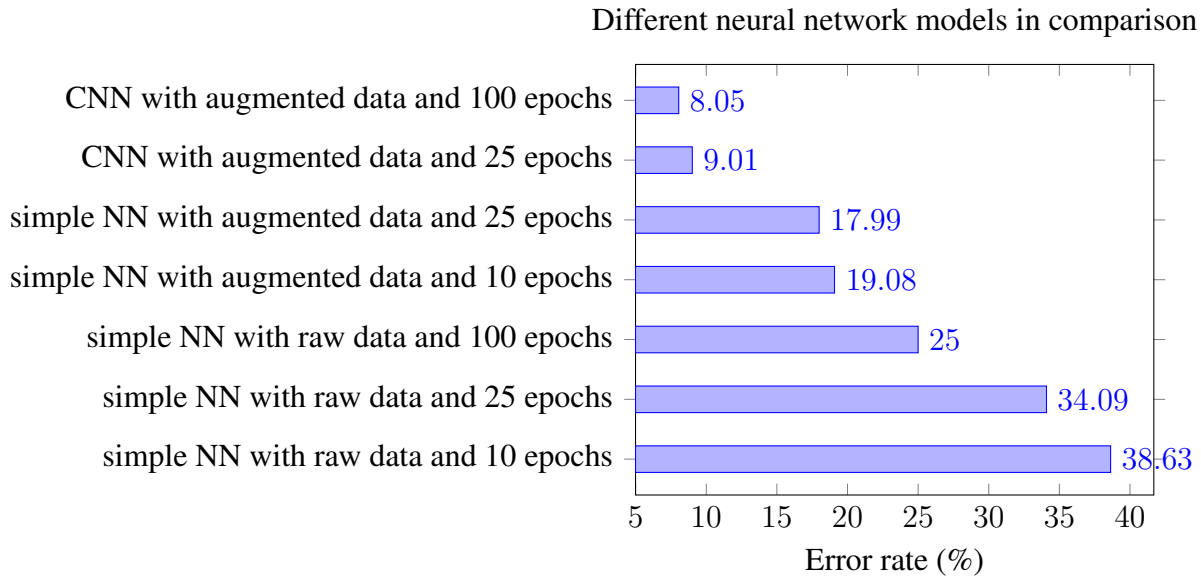


Figure 19. Different neural network models in comparison based on the error rate achieved on test data after fully training the model.

After fully training all models, they were tested on test dataset and error rate recorded. The worst neural network model using simple NN with raw data and only 10 epochs of training achieved an error rate of 38%. Then, after changing the neural network model step by step to finally use convolutional layer and augmented data with 100 epochs of training managed to bring the error rate down to 8%. The success criterion we defined to be the neural network reaching an error rate of under 10% on test images. The error rate hereby is defined as the sum of false positives and false negatives divided by the total number of samples. As seen from figure 19 the error rate of the best model is 8.05%, which is lower than what was noted in success criterion. We consider this result a success. It has become clear that to reach error rate even lower than 8% with a problem as complex as classifying human body poses, thousands upon thousands of test images are required and most likely key-point detection must be incorporated into the model.

4.5 Conclusion

This chapter illustrates details of the algorithms and methods used during experimentation. The chapter also covers the reasoning behind parameters used for each algorithm and method. The chapter explains in detail the different neural network models tested and the results achieved. Through extensive parameter optimization in all steps of methodology, we have achieved 8.05% error rate with detecting human body pose from a single image obtained through a low-cost camera.

5 Conclusion

5.1 Conclusion

Computer vision became under more active research in the late 1990s and it is still an actively researched field with many challenges. In the 2000s researchers started to actively combine machine learning with computer vision to try to solve different challenges like object recognition. An active challenge in this category in human body poses recognition. A solution for human body pose recognition that is both fast and very accurate would solve many real-life problems like detecting distracted drivers.

In this thesis, we presented techniques capable of detecting and recognizing human body poses using a neural network with class-based data augmentation. The approach includes three major steps that are related to the detection of the human, extraction of the human silhouette and classification and recognition of the human body pose. The proposed framework uses pre-trained SVM for human detection, GrabCut algorithm for human silhouette extraction and for human body pose classification a CNN that has been trained with an augmented dataset of silhouettes.

The results obtained by our method are very encouraging since we produced accuracy over 90% with the best neural network model. Best neural network model was created by using a CNN that was trained with an augmented dataset consisting of 2260 silhouettes created by the detection and extraction steps. The CNN was trained to detect if a human was walking or standing still based on the silhouette so the dataset had over 1000 silhouette examples per class. The trained model of the CNN with forward and backward propagation achieved an error rate of 8%.

The recommendations for improving the performance of the classifier are described in next section. The recommendations need to be further checked and are therefore introduced under the subsection of future work.

5.2 Future work

As future work, we should increase the number of the images and the classes used in our dataset, which will have an impact on the recognition. In addition, through this investigation, we came to the conclusion that deep learning techniques heavily rely on the input parameters. By tuning the CNN input parameters and layers, the accuracy might be improved. The future aim should be to bring the human body pose recognition accuracy over 99%.

References

[1] Graz-01 persons dataset. http://www-old.emt.tugraz.at/~pinz/data/GRAZ_01/persons.zip. Accessed: 2017-12-01.

[2] Intel hog descriptor example. https://software.intel.com/en-us/ipp-dev-reference-histogram-of-oriented-gradients-hog-descriptor. Accessed: 2018-02-01.

[3] Robustmatting trimap example. http://www.juew.org/projects/RobustMatting/index.html. Accessed: 2018-02-05.

[4] Anelia Angelova, Alex Krizhevsky, Vincent Vanhoucke, Abhijit S Ogale, and Dave Ferguson. Real-time pedestrian detection with deep network cascades. In *BMVC*, volume 2, page 4, 2015.

[5] Yuri Boykov and Marie-Pierre Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *IEEE Internation Conference on Computer Vision.*, volume 1, pages 105–112. IEEE, 2001.

[6] Gary Bradski and Adrian Kaehler. Opencv. *Dr. Dobb's journal of software tools*, 3, 2000.

[7] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[8] Shinko Yuanhsien Cheng and Mohan M Trivedi. Turn-intent analysis using body pose for intelligent driver assistance. *IEEE Pervasive Computing*, 5(4):28–37, 2006.

[9] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[10] A. Derbel, D. Vivet, and B. Emile. Access control based on gait analysis and face recognition. *Electronics Letters*, 51(10):751–752, 2015.

[11] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[12] Alvaro Garcia-Martin and Jose M. Martinez. Robust real time moving people detection in surveillance scenarios. *ResearchGate*, 2010.

[13] Alexander Gepperth, Michael Garcia Ortiz, and Bernd Heisele. Real-time pedestrian detection and pose classification on a gpu. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 348–353. IEEE, 2013.

[14] Ben T. Grys, Data S. Lo, Nil Sahin, Oren Z. Kraus, Quaid Morris, Charles Boone, and Brenda J. Andrews. Machine learning and computer vision approaches for phenotypic profiling. *Journal of Cell Biology*, 2016.

[15] Joko Hariyono and Kang-Hyun Jo. Detection of pedestrian crossing road: A study on pedestrian pose recognition. *Neurocomputing*, 234:144–153, 2017.

[16] Tushar Jain. Automation and integration of industries through computer vision systems. *International Journal of Information and Computation Technology*, 3(9):963–970, 2013.

[17] Nataraj Jammalamadaka, Andrew Zisserman, and CV Jawahar. Human pose search using deep networks. *Image and Vision Computing*, 59:31–43, 2017.

[18] Tulga Kalayci and Ozcan Ozdamar. Wavelet preprocessing for automated neural network detection of eeg spikes. *IEEE engineering in medicine and biology magazine*, 14(2):160–166, 1995.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. pages 1097–1105, 2012.

[20] AS Kulkarni and SB Shinde. Monitoring driver distraction in real time using computer vision system. 2017.

[21] Arzoo Lakhwani, Krupali Shah, Ankita Vaghela, Disha Panchal, and Saurabh Rathod. Review on basics of computer vision and its applications. *Journal of Computational Biology*, 6(2):33–40, 2017.

[22] Yann Lecun, Leon Bottou, Y Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. 86:2278 – 2324, 12 1998.

[23] Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. pages 841–848, 2002.

[24] Michael P. Pounds. Deep machine learning provides state-of-the-art performance in image-based plant phenotyping. *GigaScience*, 6:1–10, 2017.

[25] Grégory Rogez, Jonathan Rihan, Srikumar Ramalingam, Carlos Orrite, and Philip HS Torr. Randomized trees for human pose detection. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[26] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23, pages 309–314. ACM, 2004.

[27] Tom Schaul, Justin Bayer, Daan Wierstra, Yi Sun, Martin Felder, Frank Sehnke, Thomas RĆ1/4ckstieĆ, and JĆ1/4rgen Schmidhuber. Pybrain. *Journal of Machine Learning Research*, 11(Feb):743–746, 2010.

[28] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, April 2017.

[29] Arti Singh, Baskar Ganapathysubramanian, Asheesh Singh, and Soumik Sarkar. Machine learning for high-throughput stress phenotyping in plants. 21, 12 2015.

[30] Roberto Vezzani and Rita Cucchiara. Annotation collection and online performance evaluation for video surveillance: The visor project. *IEEE Xplore*, 4:227–234, 2008.

[31] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for svms. pages 647–653, 2000.

[32] Peter Wilf, Sharat Chikkerur, Stefan Little, Scott Wing, and Thomas Serre. Leaf architecture: Computer vision cracks the leaf code. 07 2013.

[33] Bo Wu and R Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. *Proceedings of the IEEE International Conference on Computer Vision*, 1:90 – 97, 2005.

[34] Dao-Xun Xia, Song-Zhi Su, Li-Chuan Geng, Guo-Xi Wu, and Shao-Zi Li. Learning rich features from objectness estimation for human lying-pose detection. *Multimedia Systems*, 23(4):515–526, 2017.

[35] Fengliang Xu and Kikuo Fujimura. Human detection using depth and gray images. pages 115– 121, 08 2003.

[36] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.

[37] Baohua Zhang, Wenqian Huang, Jiangbo Li, Chunjiang Zhao, Shuxiang Fan, Jitao Wu, and Chengliang Liu. Principles, developments and applications of computer vision for external quality inspection of fruits and vegetables: A review. *Food Research International*, 62:326–343, 2014.

[38] Liang Zhao and Charles E Thorpe. Stereo-and neural network-based pedestrian detection. *IEEE Transactions on intelligent transportation systems*, 1(3):148–154, 2000.

[39] Jian Zhou and Olga G Troyanskaya. Predicting effects of noncoding variants with deep learning-based sequence model. 12, 08 2015.

Appendix

I. License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Karl-Kristjan Luberg**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

   1.1 reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

   1.2 make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

   of my thesis

   Human Body Poses Recognition Using Neural Networks with Class Based Data Augmentation

   supervised by Dr Amnir Hadachi and Mr Artjom Lind

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, 21.05.2018