“ALEXANDRU IOAN CUZA” UNIVERSITY OF IAȘI

FACULTY OF COMPUTER SCIENCE

MASTER’S THESIS

Human Gait Recognition using Deep Neural Networks

proposed by

Rareș-Alexandru Stan

Session: *July, 2019*

Scientific Coordinator

Lect. Dr. Ignat Anca

“ALEXANDRU IOAN CUZA” UNIVERSITY OF IAȘI
FACULTY OF COMPUTER SCIENCE

Human Gait Recognition using Deep Neural Networks

Rareș-Alexandru Stan

Session: *July, 2019*

Scientific Coordinator
Lect. Dr. Ignat Anca

Contents

1 Introduction

Gait is the movement pattern of the limbs during walking over a solid surface. It varies based on speed, terrain, maneuvering or efficiency of energy. This movement is unique for each human and can be used for recognizing persons from afar, without the need of their cooperation or physical contact, whereas fingerprint, iris or facial do need the physical access or their cooperation [1].

There are three main categories in which recognition could be classified, Machine Vision (MV), floor sensors and wearable sensors. MV is preferred because it is effective in continuous authentication and is the most non-intrusive approach.

We will create a system for human gait recognition using machine Vision and Convolutional Neural Networks, that accept a series of frames with the person walking.

2 State of the Art

Human gait is the movement pattern of the limbs during walking. It can vary depending on the persons age, weight, how tired he is and if he is carrying extra weight. A system for recognizing persons by their walking should take all of the situations from above, to correctly identify them.

There are three main approaches for identifying people by their gait, Machine Vision (MV), floor sensors and wearable sensors. Each of the three approaches have some disadvantages and advantages:

- MV:
  - it is cheaper to implement, no need to install extra sensors, just some video cameras;
  - can cover a wide area;
  - it is affected if the people are wearing voluminous clothes;
- Floor Sensors:
  - are not affected by the clothes worn by the user;
  - are more expensive to implement than MV;
  - limited area for recognizing people;
- Wearable Sensors:
  - are not limited by a specific area;
  - are not affected by the clothes worn by the user;
  - you need to have physical access or to have their cooperation.

In Machine Vision there are tow main approaches, model-free and model-based, where the first approach uses direct image sequences, whereas the latter needs more processing of the input sequence.

2.1 Model-Based Machine Vision

Molhema Mohualdeen and Magdi Baker [2] have proposed a model-based approach for the Gait Recognition problem using Region os Interest (ROI), Discrete Wavelet Transform (DWT), Edges, Gait Cycle and Neural Networks. ROI was used in the preprocessing phase to reduce data and extract the exact silhouette from each frame, by cropping.

Next, in the feature extraction phase, they used DWT for multi-scale analysis, using diagonal, horizontal and vertical details of the three levels low pass and high pass filters on two dimensions DWT. Beside 3L-2D-DWT they used Edge Detection for magnitude and orientation and box technique for step and cycle length, using the with of the bounding box. Estimating the Gait Cycle was done by combining the silhouettes between the tow main phases of Gait and combining them together for each person and measuring the combination area and the width of the white shape boundary represents the step length. Classification was done using a Back Propagation Neural Network (BPNN).

Munif Alotaibi and Ausif Mahmood [3] propose a different type of preprocessing with a Convolutional Neural Network for classification. The processioning is done using the Gait Energy Image (GEI), defined as: $GEI(x, y) = 1/s \sum_{t-1}^{s} F^t(x, y)$, where s is the total number of frames representing the Gait Cycle and $F^t(x, y)$ is the silhouette of the subject at the time interval t. For determining the Gait Cycle it is used the bounding box changes method and the silhouettes are then resized to $140 * 140$ pixels. The Neural Network has 4 pairs of Convolution and Pooling layers, each with eight $5 * 5$ filters and eight sub-sampling maps with pooling factor 2. For the activation function of the Convolutional layers it is used the Hyperbolic Tangent function. After the last Convolution and Pooling pair a Dense Layer with 124 nodes and SoftMax activation functions is used, to classify the data. For adding a new user to be recognized by the system, the old model is taken and froze the Convolutional and Pooling layers, so they are not changed during the new training period, and just the Dense Layer is modified, by adding a new node, and retrained.

Hazem El-Alfy, Ikuhisa Mitsugami and Yasushi Yagi [4] build a system in which the preprocessing is done using the Gauss Map of the silhouettes and classification they use Euclidean Distance on the feature vectors between the person to be recognized and the existing database. In more details, the Gauss Maps wore done on the silhouette's surface, evaluated locally, to overcome the lack of the third dimension and made all the normal vectors point outwards the silhouette. Gauss Mapping was done on a silhouette with it's boundary extracted then smoothed using a parametric cubic spline interpolation, for it's continuity at zero, first and second order with control points being every fifth pixel of the boundary. All of the silhouettes pixels are then Distance Transformed where the distance is calculated as follows $d = max(|x_1 - x_2|, |y_1 - y_2|)$, where (x_1, y_1) and (x_2, y_2) are two distinct pixels from the image and then are computed the contour lines of the distance map. After this the image is divided in a regular grid and for each cell in computed a histogram of all the normal vectors in that contour cell. All of the cell are combined in a feature vector and it is repeated for all contours in that image. Last all the feature vectors for that image are merged into the final feature descriptor, the NDM. All of the NDMs from a full gait cycle are integrated together, using their average for the aggregate cycle descriptor. Over this aggregated feature vector the Euclidean Distance is calculated.

2.2 Model-Free Machine Vision

3 Contributions

The two classes of methods used for Gait Recognition using Machine Vision are: Model-Base and Model-Free. In the former class there are methods that preprocess the input data or extract some other information from it, which is used for classification, whereas the latter uses the base data, only with preprocessing meant to clean and sanitize the input, for classification. In this theses we have tried both Model-Based and Model-Free approaches for Human Gait Recognition and obtained the best results with one of the Model-Free approaches.

For training and testing the CASIA-B database [5][6][7] was used, as it already has the silhouettes extracted from the videos, which gave us the possibility to focus on the recognition methods. We have used the first part of the database, which included silhouettes for 62 people, of the 124 total. The database has been split in three categories: training, nm01-nm04, validation nm-05 and testing nm-06.

We have tried both Model-Based and Model-Free approaches. The first model we tries had the input images cropped, smoothed and then a surface curvature metric extracted from the silhouette and then classified using a Convolutional Neural Network. After that the rest of the models had the input images centered around the silhouette and cropped, then feed to either classic Convolutional Neural Networks (CNN) or combinations of CNN and Recurrent Neural Network (RNN) for classification. Lastly a comparison between all the models described.

Through surface curvature metric we refer to a function that can map the shape, position and altitude, from an image with three dimensional objects, to a numerical value, for each pixel in the original image.

Smoothing refers to the process of removing noise or fine-scale structures from images, resulting in a less jagged edges and sharp curves/transitions.

4 Approach

In the first part of this chapter we will explain what the shape index represents and how Convolutional Neural Networks and Recurrent Neural Network classify data. In the second part of the chapter the architectures and preprocessing used for the Model-Based and Model-Free approaches will be presented.

4.1 Shape Index

The shape index measures the local curvature, derived from the eigen values of the Hessian, defined by Koenderink and van Doorn [8]. It can be used to find structures based on their apparent local shape. It maps to values in the range $[-1, 1]$, representing different shape types. It is defined as follows:

$$s = \frac{2}{\pi} * \arctan \frac{k_2 + k_1}{k_2 - k_1} \quad (k_1 \geq k_2)$$

Here (k_1, k_2) are polar coordinates described by $k_{1,2}^2 - 2H_{k_{1,2}} + K = 0$, where H is the mean curvature, measuring the spread of normals for the points of infinitesimal arcs, divided by the arch length and averaged over all surfaces, and K is the Gaussian curvature, specifying the spread of normals.

4.2 Neural Networks

Neural Networks (NN) are a collection of connected nodes, name neurons, that for a node j, at the time interval t, with input $p_j(t)$, consist of:

- an activation $a_j(t)$,
- a threshold θ_j,
- an activation function f that computes the new activation at time $t+1$ from $a_j(t), \theta_j$ and the input $p_j(t)$, resulting $a_j(t+1) = f(a_j(t), p_j(t), \theta_j)$,
- and an output $o_j(t) = f_{out}(a_j(t))$

and with a propagation function, that computes the input $p_j(t)$, for node j, from $o_i(t)$ of predecessor neurons and has the form $p_j(t) = \sum_i o_i(t) * w_{ij} + w_{0j}$, where w_{0j} is a bias.

Learning in Neural Networks is done by firstly feed forward the input, to compute the output of the network, for the input $x, x \in \mathbb{R}^n$, where n is the size of the input vector. Secondly calculate the error of the entire network for the known input and lastly propagate the error and update the weights of each layer. The weight updating is described throw the following:

$$w_{ij}(t+1) = w_{ij}(t) - \eta\frac{\partial C}{\partial w_{ij}} + \xi(t)$$

where η is the learning rate, C is the cost function, and $\xi(t)$ a stochastic term. The cost function depends on the learning type and the activation function, usually in supervised learning the cost function is Cross Entropy. Back-Propagation is done for each training sample for the desired number of epochs.

4.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) beside the classic dense layers they have convolution and pooling layers and are primarily used for image and video related learning. Convolutional layers have multiple filters of fixed dimensions, randomly initialized with a uniform distribution. The $a * b$ filter is applied over the input x, using the dot product between the filter and sub-matrix of x, with a stride s, and the output is added with the bias term and then the activation function is applied, this happens for all the defined filters of a convolution layer independently. Pooling layers reduce the size of the input by a specified factor, using functions like: max, average, min, etc. Pooling layers are used for enabling the following filters to look at bigger feature of the input.

4.2.2 Recurrent Neural Networks

Recurrent Neural Networks (RNN) are mostly the same as standard Neural Networks, with the added bonus of each neuron having its owen memory. The activation function is time-varied. Thanks to having internal memory for each node, RNNs can process time sequences of inputs, which makes them great for video processing, speech recognition, and generating sequences.

One such king of RNN is Long Short-Term Memory (LSTM) which avoids the vanishing and exploding gradient. Besides the standard RNN model, they have recurrent gates, called forget gates.

4.3 Model-Based Approach

For the Mode-Based approach we used a Convolutional Neural Network which has as input a set of frames that have applied the shape index on each of them.

Model-Based Machine Vision approaches have a preprocessing phase, in which features are extracted from the input and passed to a Machine Learning algorithm for classification.

Because gait happens over time, we needed a way to capture the passing of time. In order to do that we used 3D Convolution, with the third dimension being time, using multiple consecutive frames.

Before any preprocessing we sanitized the input images af follows:

1. select the first 80 frames from the database, for each angle of all the 62 people, if there are ath least that many frames in that set, otherwise take all the frames and add empty frames to the end of the series, until 80 images are reached,

2. center the silhouette in each frame,

3. crop all images so that only the silhouette is in the frame, with a small padding,

4. resize the each input to 120 × 60 pixels.

After the input is sanitized we applied the shape index algorithm for each frame, transforming it from a matrix with values $\{0, 1\}$ into a matrix with values in the range $[-1, 1]$. Then all the resulting matrixes for a persons angle are put together, in chronological order, to obtain a tensor (3D matrix), with shape $(80, 120, 60)$, 80 frames, 120 pixels heigh and 60 pixels wide frames. Each tensor is a training sample for the CNN.

Last step is the Convolutional Neural Network, which takes as input a tensor of shape $(80, 120, 60)$ and outputs a list with confidences, for each of the 62 subjects and we will say that the CNN thinks that the input is a recording of the person with the highest confidences. The Model used here has four 3D convolutional layers, each followed by a 3D max pooling layer and finally a densely connected layer, of size 62, for output labeling.

When doing convolution, values nearby are combined and evolved to express new features. When moving from a convolution layer to another, the features from the previous layer are used as input for the next one and previous features are combined to express more complex features in the current layer. The pooling layers are used to select between the features from the previous layer so after it only the important features are passed down.

For each convolution layer, there is specified a number of filters, which will convert the input in an output with that many channels, each one having the same size as the input. Pooling happens for each chanel independently.

In Table 1 all the layers are described with each ones size, number of channels and activation functions.

After training, this model proved unsuccessful, having the accuracy on both the validation and text set of inputs $\approx 1.62\%$, which is equivalent to assigning a random person to each input.

Table 1: Description of the CNN Model used with the shape index preprocessing

Layer name	**Size**	**No. Filters**	**Activation**
3D Convolution	10, 10, 10	32	elu
3D Max Pooling	2, 2, 2	–	–
3D Convolution	10, 10, 10	16	elu
3D Max Pooling	2, 2, 2	–	–
3D Convolution	5, 10, 10	8	elu
3D Max Pooling	2, 2, 2	–	–
3D Convolution	3, 10, 5	4	elu
3D Max Pooling	2, 2, 2	–	–
Dense	62	–	softmax

4.4 Model-Free Approach

As stated before, Model-Free approaches have little to no preprocessing, mostly to sanitize the input or reduce its size, without losing many details of features. For all the models created in this approach have little preprocessing, to normalize the size of the input and get rid of the huge empty spaces in the images as follows:

1. select the first 80 frames from the database, for each angle of all the 62 people, if there are ath least that many frames in that set, otherwise take all the frames and add empty frames to the end of the series, until 80 images are reached,
2. center the silhouette in each frame,
3. crop all images so that only the silhouette is in the frame, with a small padding,
4. resize the each input to 120 × 60 pixels.

After the cleaning and normalizing of the input data, it is feed to the Neural Network (NN). We created multiple models for the NN, to test how the size and type of layers affect the performance of the classification.

Firstly we can se two main path in the model definition, based on how time is defined, using:

- 3D Convolution,
- Recurrent Neural Networks, more exactly Long Short-Term Memory (LSTM).

When using 3D Convolution the third dimension represents time and so when applying filters and pooling, features from different frames, different time intervals, are combined in a unique way for each person, and the dense layers, that come after the convolution, will classify each person, based on those features.

Recurrent Neural Networks have memory, which remembers features from the past and are taken into consideration alongside with the input at the current time step.

4.4.1 3D Convolution

Seeing that the Model-Based approach gave so poor results, we tried to identify the problem and reached to the conclusion that the preprocessing with shape index somewhat transformed the input in something that the network did not find features with which to uniquely identified the people. So we dropped the shape index and used only the input normalization and the neural network, which game promising results on the validation and testing input sets of $\approx 59\%$. We iterated through a couple more models and reach a satisfactory result of 91.7%, on the validation and testing sets, after adding an additional dense layer of 200 nodes, right after the last max pooling layer, to the original network.

This improvement of $\approx 32.7\%$ is based on the power to extract more complex features and accentuate or remove features from the input received from the convolutional and pooling layers, of the new dense layer.

In Table 2 all the layers are described with each ones size, number of channels and activation functions.

Preventing overfitting is a very real problem when using the CASIA-B dataset [5][6][7] due to the very few training data available, only four examples for each filming angle for all the people. A couple of examples of measures to reduce or prevent overfitting are:

- add more training data,
- use dropout, so that all the nodes in the network are activated,
- early stopping,
- use regularization, to control how much the weights update.

Out of the ones listed above we have used early stopping and adding dropout to all the layers, except the input layer. If the network did not obtain a better accuracy on the validation set than the best recorded yet, in 20 epochs, we stopped the training and run the best model on the testing set. Dropout picks at random a number of nodes, in the current layer, which will be used, so that all the nodes have equal chances to have their output evaluated and not overridden by other nodes on the same layer. In our models we chose to use a dropout on half of our nodes on each layer.

When learning, neural networks need a function that calculates the value with which the weights of the nodes are updated, the most popular function is stochastic gradient descent, that has a fixed learning rate, used during the entire training. In later years other functions appeared that improved the stochastic gradient descent, to reduce the learning time, called AdaGrad, RMSProp and Adam, that combines the advantages of both AdaGrad and RMSProp. We chose to also add the Adam optimization for the learning process.

Table 2: Description of the CNN Model with additional dense layer highlighted

Layer name	**Size**	**No. Filters**	**Activation**
3D Convolution	10, 10, 10	32	elu
3D Max Pooling	2, 2, 2	–	–
3D Convolution	10, 10, 10	16	elu
3D Max Pooling	2, 2, 2	–	–
3D Convolution	5, 10, 10	8	elu
3D Max Pooling	2, 2, 2	–	–
3D Convolution	3, 10, 5	4	elu
3D Max Pooling	2, 2, 2	–	–
Dense	**200**	–	**elu**
Dense	62	–	softmax

4.4.2 LSTM

After seeing the success of the model using 3D Convolution we tried using RNN, with LSTM nodes to treat the temporality of gait. RNNs wore chosen because their success in natural language processing and generation, being able to recall events happened in the text quite early one, hoping to recall specific features that appeared in previous frames of the input.

For doing so initially a Convolution Neural Network was created, to process and extract features for one frame, using 2D Convolution and 2D Max Pooling layers. The CNN was then used as time distributed input for a LSTM layer of 128 nodes, followed by a dense layer of 256 nodes and a dense output layer of 62 nodes. The LSTM layer was used for combining and selecting features across time and the first dense layer for extracting more complex features and accentuate or remove features from the input. Lastly the output dense layer labels the output, based on the features received as input.

One of the CNN models created for the RNN consists of 4 sets of 2 2D Convolution layers followed by a 2D Max Pooling layer. The whole RNN network, with layer sizes, is described in Table 3.

Table 3: Description of the CNN+RNN Model with many Convolution layers

Layer name	**Size**	**No. Filters**	**Activation**
2D Convolution	3, 3	64	elu
2D Convolution	3, 3	64	elu
2D Max Pooling	2, 2	–	–
2D Convolution	3, 3	128	elu
2D Convolution	3, 3	128	elu
2D Max Pooling	2, 2	–	–
2D Convolution	3, 3	256	elu
2D Convolution	3, 3	256	elu
2D Max Pooling	2, 2	–	–
2D Convolution	3, 3	512	elu
2D Convolution	3, 3	512	elu
2D Max Pooling	2, 2	–	–
LSTM	128	–	tanh, sigmoid
Dense	256	–	elu
Dense	62	–	softmax

This network obtained better results than the initial CNN, with just an output dense layer, obtaining 73.7% accuracy on the test and validation sets, but adding another dense layer to the CNN obtains better results, with 18% higher accuracy. LSTM shows potential in classifying human gait.

Taking from the previous result, maybe the CNN is to big and does not select all the relevant information for identifying people, so we changed the CNN with one that proved to have good results, from the 3D Convolution and adapted it to work with 2D input, removing the time dimension. The resulting CNN+RNN is described in Table 4.

Using a smaller CNN improved the accuracy of the entire network with 1.3%, to 76%, on validation and testing sets. To further improve the accuracy we can change the activation function of the LSTM layer or reduce de size of the dense layer. Further tests will be done.

To prevent overfitting we have used early stopping and adding dropout to all the

Table 4: Description of the CNN+RNN Model with fewer Convolution layers

Layer name	**Size**	**No. Filters**	**Activation**
2D Convolution	10, 10	32	elu
2D Max Pooling	2, 2	–	–
2D Convolution	10, 10	16	elu
2D Max Pooling	2, 2	–	–
2D Convolution	10, 10	8	elu
2D Max Pooling	2, 2	–	–
2D Convolution	10, 5	4	elu
2D Max Pooling	2, 2	–	–
LSTM	128	–	tanh, sigmoid
Dense	256	–	elu
Dense	62	–	softmax

layers, except the input layer. If the network did not obtain a better accuracy on the validation set than the best recorded yet, in 20 epochs, we stopped the training and run the best model on the testing set. Dropout picks at random a number of nodes, in the current layer, which will be used, so that all the nodes have equal chances to have their output evaluated and not overridden by other nodes on the same layer. In our models we chose to use a dropout on half of our nodes on each layer. Learning was speedup using the Adam optimization.

5 Conclusions

6 Bibliography

References

[1] H. Srivastava, “A comparison based study on biometrics for human recognition,” *IOSR Journal of Computer Engineering*, vol. 15(1), pp. 22–29, 2013.

[2] M. Mohualdeen and M. Baker, “Gait recognition based on silhouettes sequences and neural networks for human identification,” *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, vol. 6, p. 110 117, March 2018.

[3] M. Alotaibi and A. Mahmood, “Improved gait recognition based on specialized deep convolutional neural network,” *Computer Vision and Image Understanding*, vol. 164, pp. 103–110, 2017.

[4] H. El-Alfy, I. Mitsugami, and Y. Yagi, “Gait recognition based on normal distance maps,” *IEEE Transactions on Cybernetics*, vol. 48, pp. 1526–1539, May 2018.

[5] S. Zheng, J. Zhang, K. Huang, R. He, and T. Tan, “Robust view transformation model for gait recognition,” in *International Conference on Image Processing(ICIP)*, (Brussels, Belgium), 2011.

[6] S. Yu, D. Tan, and T. Tan, “A framework for evaluating the effect of view angle, clothing and carrying condition on gait recognition,” in *Proc. of the 18'th International Conference on Pattern Recognition (ICPR)*, (Hong Kong, China), August 2006.

[7] R. He, T. Tan, and L. Wang, “Robust recovery of corrupted low-rank matrix by implicit regularizers,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, pp. 770–783, April 2014.

[8] J. JKoenderink and A. J. van Doorn, “Surface shape and curvature scales,” *Image and Vision Computing*, vol. 10, pp. 557–564, October 1992.