

RESEARCH

Open Access



# Analysis and best parameters selection for person recognition based on gait model using CNN algorithm and image augmentation

Abeer Mohsin Saleh<sup>1,2\*</sup>  and Talal Hamoud<sup>1</sup>

\*Correspondence:  
abeersaleh5988@gmail.com  
<sup>1</sup> Damascus University,  
Damascus, Syria  
Full list of author information  
is available at the end of the  
article

## Abstract

Person Recognition based on Gait Model (PRGM) and motion features is indeed a challenging and novel task due to their usages and to the critical issues of human pose variation, human body occlusion, camera view variation, etc. In this project, a deep convolution neural network (CNN) was modified and adapted for person recognition with Image Augmentation (IA) technique depending on gait features. Adaptation aims to get best values for CNN parameters to get best CNN model. In Addition to the CNN parameters Adaptation, the design of CNN model itself was adapted to get best model structure; Adaptation in the design was affected the type, the number of layers in CNN and normalization between them. After choosing best parameters and best design, Image augmentation was used to increase the size of train dataset with many copies of the image to boost the number of different images that will be used to train Deep learning algorithms. The tests were achieved using known dataset (Market dataset). The dataset contains sequential pictures of people in different gait status. The image in CNN model as matrix is extracted to many images or matrices by the convolution, so dataset size may be bigger by hundred times to make the problem a big data issue. In this project, results show that adaptation has improved the accuracy of person recognition using gait model comparing to model without adaptation. In addition, dataset contains images of person carrying things. IA technique improved the model to be robust to some variations such as image dimensions (quality and resolution), rotations and carried things by persons. Results for 200 persons recognition, validation accuracy was about 82% without IA and 96.23 with IA. For 800 persons recognition, validation accuracy was 93.62% without IA.

**Keywords:** Person recognition, Convolution neural network, Gait model, Deep learning, Image augmentation

## Introduction

Gait is a kind of behavioral biometric feature, it is defined as the way a person moves and the movement of every person has no typical form or scenarios.

There are other behavioral biometrics like face and iris but they are limited by the distance between the person and the used camera. There are two types of biometric features soft and hard; the hard ones are also considered classic or traditional, such as

faces, fingerprints or signatures. The soft ones are related to faces, as skin color, hair color or facial measurements, other soft are related to bodies, like height or weight, and to accessories, such as glasses or hats [1]. The unique characteristics of gait, such as unobtrusive, non-contactable and non-invasive, have a powerful potential to apply in the scenarios including criminal investigation, access security and surveillance [2]. In general, gait can be used for the human recognition task. Not only person recognition can be estimated by gait, gender and age can also be estimated by it [3].

In addition, PRGM has a many application in real time life, it offers a useful tool for non-invasive biometric validation, and human-robot interaction in a broad range of applications from crowd traffic management to personalized health care. Considering the gait as a set of poses and movements, gait information can be extracted from images or videos from static cameras, the information represent the gait of person [4].

PRGM also has important role in video surveillance. It is particularly challenging because observed pedestrians undergo significant variations across camera views, and there are a large number of pedestrians to be distinguished given small pedestrian images from surveillance videos [5].

Some researches work on person recognition and assume that observations of persons are captured in relatively short periods, such that clothes and body shapes do not change much and can be used as cues to recognize identity. In video stream surveillance, the captured persons are often small in size, facial or iris features are indistinguishable in images and face or iris recognition techniques are not applicable. Therefore person recognition techniques with gait models become important [6].

There are two ways to get gait features for any person cameras and sensors. Surveillance cameras may observe tens of thousands pedestrians by images in a public area in one day and many of them look similar in appearance. cameras images might be small images and contain sometimes noise or it could be blurry image, so we need a powerful tool to to be robust to these conditions. Our designed model treat person recognition regardless the large variations of lightings, poses, viewpoints, blurring effects, image resolutions, camera settings, and background across camera views, the designed model has many enhancements as we will see.

Other method for gait model is recording using sensors; motion sensors are available widely such as accelerometers and gyros, these sensors are located on the human body; in which gait analysis is performed based on the readings of the sensors, sensors might be built into the smart-phones. In this regard, especially important is the problem of recognizing the person using his gait [7].

Using sensors (as mentioned above) might be undesirable by the person, cameras can be captures images or videos of person without the attention of person, so using cameras are suitable for gait analysis and person recognition. Images from cameras differs in some issues according to some conditions:

- Cameras resolution.
- Cameras position.
- Weather conditions (night, day, light issues).

To handle these conditions, we need a powerful tool, deep learning tool (CNN as example) to deal with all condition and search deeply to find all possible features for each person. After mentioning many application for PRGM and with the increasing demand for person recognition in the era of big data and artificial intelligence, the research and development of many algorithms attracted broad attention from both academia and industry, e.g., the fingerprint, iris, face, and voice etc., have been implemented commercially. All examples in this field will output huge amount of data ( 1,2 or dimensions), therefore we need a Convolution neural network to handle features that can be concluded from the input data.

In addition to data in our hand; the persons images, we also worked on image augmentation and thus will increase the amount of data multi times. So this research addresses the problem of person recognition depending on gait model with main research points:

- Best selection of both parameters and the design of CNN
- Using Image Augmentation to increase person features and to make trained model robust to some variations in the image
- Understand the structure, layers and parameters of CNN to implement it and be able to understand how to change CNN properties.

### **Background and related work**

In [8], authors introduce a simulation-based methodology and a subject-specific dataset to for generate synthetic video frames and sequences for data augmentation to help in gait recognition. They generated a multi-modal dataset. In addition, they supply simulation files that provide the ability to simultaneously sample from several confounding variables; these variables are about brightness, rotation, color saturation. The basis of the data is real motion capture data of subjects walking and running on a treadmill at different speeds. Results from gait recognition experiments suggest that information about the identity of subjects is retained within synthetically generated examples. This study has applied image augmentation using render program to generate an accurate sequence of motion as real as possible, this is costing processing time, they also did not consider body form (fat, this, short and tall).

In [9], the authors used a virtual environment, which enabled them to present the same type of gait across different identities. Using this setting, they assessed the accuracy and distance at which identities are recognized based on their gait, as a function of gait distinctiveness. Furthermore, the virtual environment also enabled them to assess. They find that the accuracy and distance at which people were recognized increased with gait distinctiveness. Overall these findings highlight an important role for gait in real life person recognition and stress that gait contributes to recognition independently from the face and body. The virtual environment helped them a lot in their research.

In [10], authors proposed a gait recognition approach for person re-identification. The proposed approach starts with estimating the angle of gait first, and this is then followed with the recognition process, which is performed using convolution neural networks. Here in, multi-task convolution neural network models and extracted Gait Energy Images (GEI) are used to estimate the angle and recognize the gait. GEIs are extracted by

first detecting the moving objects, using background subtraction techniques. The proposed gait recognition method showed an accuracy of more than 98% for almost all used datasets. The authors worked on GEIs, there is a feature engineering process before re-identification process.

In [4], the authors devoted to the problem of the recognition of a person by gait using a video recorded in the optical range depending on detection of a moving person on a video sequence with the subsequent size normalization and dimension reduction using the principal component analysis (PCA) technique. The person classification was carried out using the support vector machine (SVM). Authors have determined the best values of the method parameters; their method for person recognition has next steps: detection and segmentation of a moving person in the video sequence, normalization of the frame size of the selected video sequence fragment, dimensionality reduction of the selected video sequence fragment and classification of video sequences. The obtained results showed high classification accuracy with small number classes.

In [11], The authors have studied recognition performance when handling confounding variables, such as clothing, carrying and view angle. A novel method was proposed to explicitly disentangle pose and appearance features from RGB imagery to get pose features over time produces. In addition, authors focused on gait recognition from frontal-view walking, which is a challenging problem since it contains minimal gait cues compared to other views. The method demonstrated superior performance compared with other researches. The cases of tests focused on rotations and variations from only front view walking.

In [12], The author has applied a gait recognition depending on the reality that every human has a distinctive walking style; which is proposed to be used in gait recognition as an identification criterion, Author has applied CNN with help of center-of-pressure (COP) trajectory that is sufficiently unique to identify a person with high certainty. Using a platform to record COP for a period then using these records to classify only 30 persons, this method requires a platform for each person so it is expensive.

In [13],The authors developed a specialized deep CNN architecture for Gait Recognition. The proposed architecture is less sensitive to several cases of the common variations and occlusions that affect and degrade gait recognition performance. It can also handle relatively small data sets without using any augmentation or fine-tuning techniques. The majority of previous approaches to gait recognition have used subspace learning methods which have several shortcomings that we avoid. Their specialized deep CNN model can obtain competitive performance when tested on the CASIA-B large gait data set; CASIA data set has only 20 persons.

The usage of neural network (the basic thing in CNN model) has increased remarkably lately, Many studies on neural networks in new fields is are rising every day. In Compacting Concrete as in [14], In navigation fields as in [15]. In quantum physics as in [16].

In this study, the tests were on data set that has images for 200 persons, most of the studies tested on data sets with small number of persons. In addition, the data set images have low resolution, only 64\*128. Some studies depend on ready models with some modification or tuning. In this study, CNN model is built step by step as it will be explained in next paragraphs.

The main benefit of our study can be summarized in two main points:

*First point:* Image augmentation for gait recognition no matter what used algorithm was. In our case, CNN was used.

*Second point:* Using CNN algorithm and best selection for:

- CNN parameters.
- CNN design and structure.

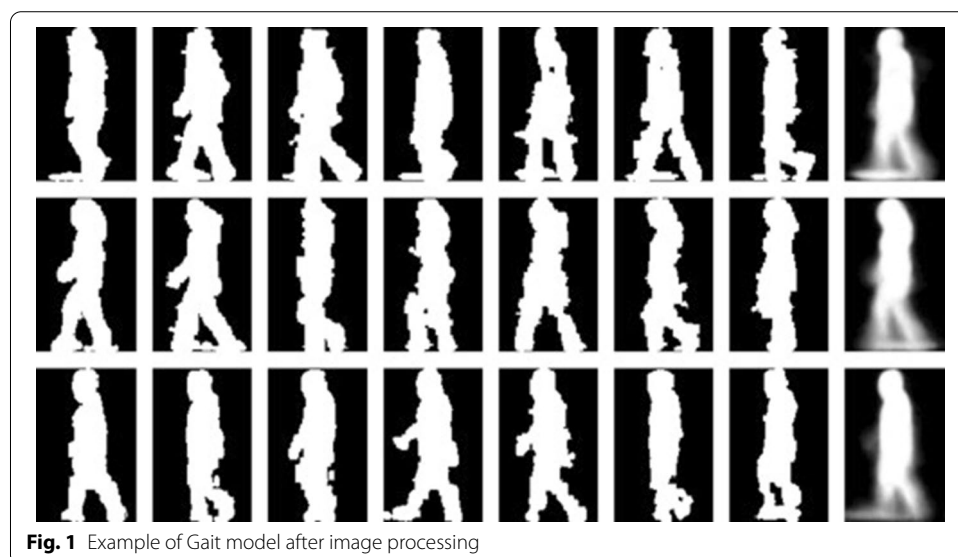
### Objective of the paper

Actually, the main intention of this work in general is to master the person recognition based on gait features and improve the recognition as possible as we can. The other goal is to apply improved recognition system in application such as tracking COVID spread and recognition people in cases where there is no fingerprint, no iris detection and no clear view of the face. Examples of Gait model is presented in the Fig. 1 below [17]

In practice, gait model recognition is often used when cameras are installed around the target subjects (persons), which may be unaware of the observation for prisoners or criminals [5].

New reports about gait models are collected for corona-virus states. Under this title “Corona-virus gives China more reason to employ biometric”, tablets (containing cameras) have recently been installed by the driver’s seat in public buses and in many places. Passengers are expected to put their foreheads close to the tablet so that their temperatures and photos can be taken. The photos taken by the cameras (more than 200 millions cameras) can then be used in person recognition depending on many terms including gait model that might be ill with Corona-Virus. The number of CCTV1 cameras in China in 2019 is 200 and it will be increase to 626 million by 2020 [18]. Gait model might have some carrying objects which is considered as noise in the image [19] as shown in Fig. 2.

According to the design of Convolution Neural Network, one image with dimension  $100 \times 100$  may be during the CNN stages as thirty images with less dimension (supposed





**Fig. 2** Gait model with carrying objects

50\*50), so a data 10000 will convert to 75000 (7.5 times) for one image. for a dataset of 5000 images, data during CNN may be about 375000000 that is considered as big data issue. The term “big data” refers to data that is so large, fast or complex that it is difficult or impossible to process using traditional methods.

Image augmentation is one useful way for building deep learning algorithm that can increase the size of the training set without acquiring new images. The idea is to duplicate images with some kind of transformations so the model can learn from more examples. Ideally, image can be augmented in a way that preserves the features key to making predictions, but rearranges the pixels enough that it adds some noise. Augmentation will be counterproductive if it produces images very dissimilar to what the model will be tested on, so this process must be executed with care [20].

## Methods

Convolution Neural network model is an important type of feed-forward neural network with special success on applications where the target information can be represented by a hierarchy of local features [21]. A CNN is defined as the composition of several convolution layers and several fully connected layers [22]; it is helpful tool for recognizing people through their gaits; neural networks analysis gait model to extract multiple complex features. Convolution neural networks (CNNs) have been used with great success for video-based gait recognition. CNN is well used in BigData field with many application, that image is extracted many times through the CNN process [23].

CNNs are especially well suited for working with images as a result of their strong spatial dependencies in local regions and a substantial degree of translation invariance. Similarly, time series can exhibit locally correlated points that are invariant with time shifts. The successful use of deep CNNs for the classification of unidimensional or multidimensional time series has been attested [24]. Like for image classification, CNNs can extract deep features from a signal’s internal structure. CNNs are potent tools for bypassing feature engineering in signal processing tasks (end-to-end learning) [25]. However, CNNs, like other artificial neural networks, require hundreds of examples in each class for efficient learning; therefore, they have not been applied in footstep recognition studies so far due to the difficulty of collecting many strides using sensing floors or force platforms.

There are a lot of algorithms that people used for recognition problems which their input is images before CNN became popular. In general, we need to create features from images and then feed those features into machine learning algorithm like SVM. Some algorithm also used the pixel level values of images as a feature vector too. As an example, SVM could be trained with 784 features where each feature is the pixel value for a  $28 \times 28$  image.

CNN work so much better

- CNNs can be thought of automatic feature extractors from the image. While if I use a algorithm with pixel vector I lose a lot of spatial interaction between pixels; feature engineering is not required in CNN [26].
- CNN effectively uses adjacent pixel information to effectively down-sample the image first by convolution and then uses a prediction layer at the end[27].
- CNN includes the multiple uses of the convolution operator in image processing.
- The CNN architecture implicitly combines the benefits obtained by a standard neural network training with the convolution operation to efficiently handling the requested tasks; classification, recognition and identification.
- CNN is also scalable for large datasets.

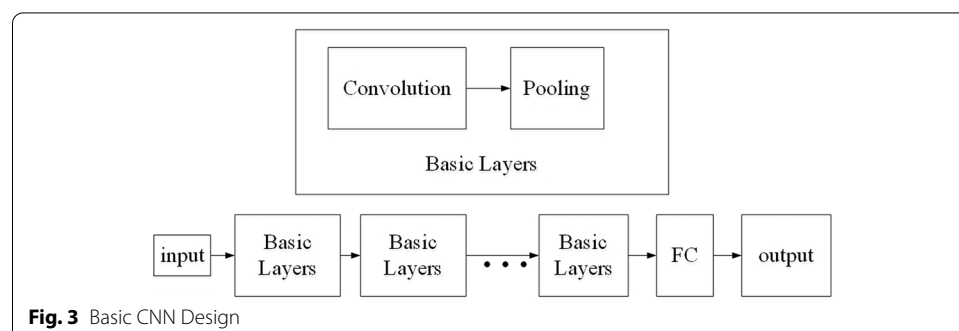
For this study, the typical deep CNN is consisted of [28]:

- Input
- Convolution layer
- Pooling Layer
- Fully connected layer (FC)
- Output

Basic CNN design is shown in Fig. 3

Each part has its own parameters and properties, convolution layer has:

- nb filter: The number of convolution filters.
- filter size: Size of filters (kernel size).
- Strides: Strides of convolution operation.
- Padding: is simply a process of adding layers of zeros to input images to avoid problem related to size after convolution process.
- Activation: Activation function applied to this layer (Default is linear).
- Bias: If True, a bias is used.
- Weights init: Weights initialization.
- Trainable: If True, weights will be trainable.
- Regularizer: Regularization is commonly used for alleviating over-fitting in machine learning. For CNN, regularization methods, such as DropBlock and Shake-Shake, have illustrated the improvement in the generalization performance [29].



There are other parameters for pooling, regression and fully connected layer.

For Pooling layer:

- Kernel size: kernel size.
- Strides: Strides of pooling operation.
- Padding: Same as in convolution layer.

Fully connected layer has:

- Neurons
- Activation function

Regression Layer:

- Loss functions: are a key part of any machine learning model: they define an objective against which the performance of the model is measured.
- Learning Rate: helps to converge faster. Choosing a wrong learning rate, either too small or too large, can have a huge impact on the output.
- Optimizer: to minimize the provided loss function 'loss' (which calculate the errors).

There is also a dropout that refers to dropping out units (both hidden and visible) in a neural network. Dropout refers to ignoring neurons during the training phase of certain set of neurons by a term named Keep Probability, which is chosen at random. By “ignoring”; i.e. these units are not considered during a particular forward or backward pass [30].

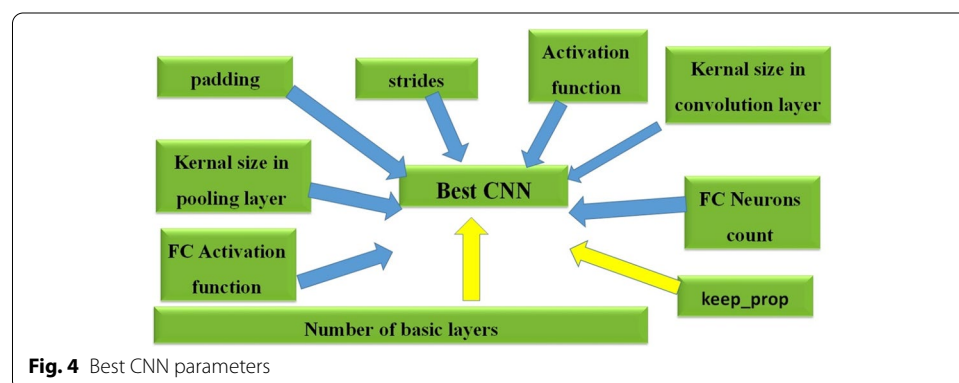
The main work is to improve CNN depending on these parameters as shown in Fig. 4:

### Market dataset

#### *Data set preparing*

First of all, Data set was downloaded from [31], this dataset contains images that have been captured in front of supermarket for Tsinghua University:

- Six cameras (5 cameras with high resolution and one with low resolution).
- General environment (in front of supermarket).



**Fig. 4** Best CNN parameters



- 1501 identity with more than 12900 images.
- Every person is existed in two images of two cameras at least.
- Image name contains: The person number, camera number, and image sequence number within the scene which is a random number Example 0001c1s100230100: person number 0001, camera c1, and image scene s1.

### Feature extraction

For convolution neural network, it automatically detects the important features without any human supervision. The convolution layer detects features such as head, long ears, legs, hands, trunk and so on. The fully connected layers then act as a classifier on top of these features. The sequence of images for the same person with will explored as temporal features for the person. There is no definition for gait imprint. somehow, we can say that gait imprint is a collection of features that represent body parts and their changes with time.

The convolution layers in CNN are the basic and important powerhouse of any CNN model. They automatically detect significant features. The convolution layers learn such complex features by building on top of each other. The first layers detect edges in the image, the next layers combine edges and lines to detect shapes or objects, to following layers merge this information to infer that this is a body or hands or legs etc. To be clear, the CNN is blind for these things; it does not know what a face is. By seeing a lot of them in images, it learns to detect that as a feature. The fully connected layers learn how to use these features produced by convolutions in order to correctly classify the images.

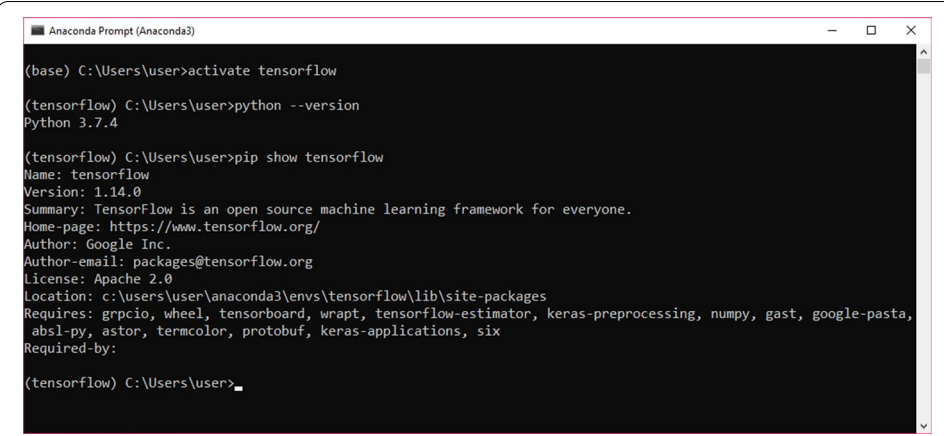
## Implementation and results

### Programming environment

Anaconda was used as programming environment. According to many references, Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012. Details about python version is shown in Fig. 5.

With tensorflow version 1.14.0 and python version 3.7.4. Tflearn was used for implementation, Tflearn is a modular and transparent deep learning library built on top of Tensorflow. It was designed to provide a higher-level API to TensorFlow in order to facilitate and speed-up experimentations, while remaining fully transparent and compatible with it. Keras is also can be used, but Tflearn was preferred because of:

- TfLearn allows to use Python arrays directly.
- TfLearn allows to save models as checkpoint, index, and meta files, these files are used to create a frozen version models easily. Frozen models are very important to be used in Android apps or C++ programs.



```

(base) C:\Users\user>activate tensorflow

(tensorflow) C:\Users\user>python --version
Python 3.7.4

(tensorflow) C:\Users\user>pip show tensorflow
Name: tensorflow
Version: 1.14.0
Summary: TensorFlow is an open source machine learning framework for everyone.
Home-page: https://www.tensorflow.org/
Author: Google Inc.
Author-email: packages@tensorflow.org
License: Apache 2.0
Location: c:\users\user\anaconda3\envs\tensorflow\lib\site-packages
Requires: grpcio, wheel, tensorboard, wrapt, tensorflow-estimator, keras-preprocessing, numpy, gast, google-pasta,
  absl-py, astor, termcolor, protobuf, keras-applications, six
Required-by:

(tensorflow) C:\Users\user>

```

**Fig. 5** Python version

- Keras saves models as HDF5 (The Hierarchical Data Format version 5) files, using which requires new skills again. Additionally, h5py library is need to be installed.
- TFLearn API is closer to that of TensorFlow.

### Model improvement

200 person were used from Market data set, 3104 images for training set and 840 images for testing set. For validation and testing, only 50 epochs were ran because training process might take too much time, all results in next paragraphs were for validation accuracy, since there are overall accuracy and validation accuracy. Model improvement was applied by testing many terms as follows to select best values of simulated terms.

Our main work is most like a genetic algorithm work. For each term we tested some values and monitored the accuracy and at which epoch the model reached high accuracy. In addition to that, IA was implemented to improve recognition accuracy. Edits on design and structure were done for best improvement.

### Testing learning rate

First of all, Define next terms: CL: Convolution Layer, PL: Pooling Layer, FC: Fully Connected. Tests details are:

*Input* → CL1 → PL1 → CL2 → PL2 → CL3 → PL3 → CL4 → PL4 → CL5 → PL5 →

FC1 → Dropout → FC2 → Regression

CL1: 32 filters, Kernal size = 3, activation function: relu, bias:true

CL2: 64 filters, Kernal size = 3, activation function: relu, bias:true

CL3: 128 filters, Kernal size = 3, activation function: relu, bias:true

CL4: 64 filters, Kernal size = 3, activation function: relu, bias:true

CL5: 32 filters, Kernal size = 3, activation function: relu, bias:true

relu: Rectified linear unit function

PL1-2-3-4-5: Kernal size = 5, Pooling type: max

FC1: 1024 neurons, activation function: relu

FC2: number of persons for neurons counts, activation function: relu

Dropout: keep probability = 0.5

Regression:optimizer = 'adam',loss ='categorical crossentropy'

by changing learning rate, and watching the validation accuracy: (Table 1)

From this results, LR is  $8e-4$

Adam is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments [32].

#### **Testing dropout**

Choosing LR =  $8e-4$ , by changing learning dropout, and recording the validation accuracy (Table 2):

From this results, best dropout value is 0.8

#### **Testing pooling Kernal size**

Choosing LR =  $8e-4$ , and dropout = 0.8, by changing Kernal size of pooling layer, and watching the validation accuracy (Table 3):

**Table 1 Results of changing LR**

LR	Validation accuracy (%)
$1e-5$	Too bad
$1e-4$	37
$5e-4$	55
$7e-4$	58.45
$8e-4$	59.17
$9e-4$	59
$1e-3$	56
$2e-3$	44
$5e-2$	Bad

**Table 2 Results of changing dropout**

Dropout	Validation accuracy (%)
0.3	59.29
0.5	59.17
0.7	63.57
0.8	64.29
1	55.12

**Table 3 Results of changing pooling kernal size**

Dropout	Validation accuracy (%)
3	63.69
5	64.29

From results above, we can choose the value 5 to be the best Pooling Kernel size. From the tests, we have figured out that high kernel size will result in fast learning. There are two types of pooling MAX and AVERAGE, we tested two types but the average type was better than other.

**Testing other parameters**

Other parameters have not add any new improvements:

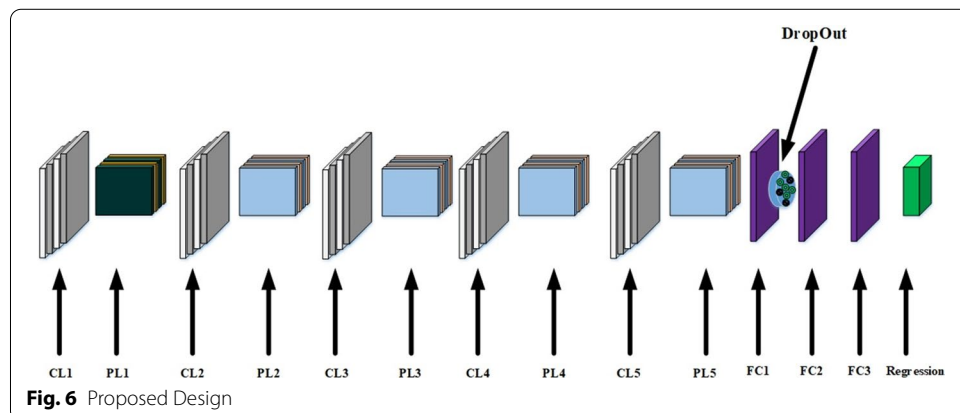
- Convolution layer kernel size: the value 3 was best for model.
- Padding and strides: keep the default values.
- loss function for regression layer: categorical cross-entropy since the labels are the Identities of persons and there are many labels (not binary issue).
- Adaptive Moment Estimation (Adam) optimizer works better (faster and more reliably reaching a global minimum) when minimizing the cost function in training [32].

**Model improvement with new design**

After choosing some parameters, changes of design was adopted as shown in Fig. 6, these edits were implemented to improve accuracy:

*Input* → CL1 → PL1 → CL2 → PL2 → CL3 → PL3 → CL4 → PL4 → CL5 → PL5 → FC1 → Dropout → FC2 → FC3 → Regression

CL1: 50 filters, Kernal size = 3, activation function: relu, bias:true, regularizer = 'L2'  
 CL2: 100 filters, Kernal size = 3, activation function: relu, bias:true, regularizer = 'L2'  
 CL3: 200 filters, Kernal size = 3, activation function: relu, bias:true, regularizer = 'L2'  
 CL4: 100 filters, Kernal size = 3, activation function: relu, bias:true, regularizer = 'L2'  
 CL5: 50 filters, Kernal size = 3, activation function: relu, bias:true, regularizer = 'L2'  
 PL1: Kernal size = 5, Pooling type: max  
 PL2-3-4-5: Kernal size = 5, Pooling type: average  
 FC1: 4096 neurons, activation function: relu  
 FC2: 4096 neurons, activation function: relu  
 FC3: number of persons for neurons counts, activation function: relu  
 Dropout: keep probability = 0.8



Regression:optimizer = 'adam',loss = 'categorical\_crossentropy', Learning rate = 8e-4  
 Results are shown in the Fig. 7:

The overall accuracy has improved but still slow in converging. At epoch 28 has the accuracy about 99%.

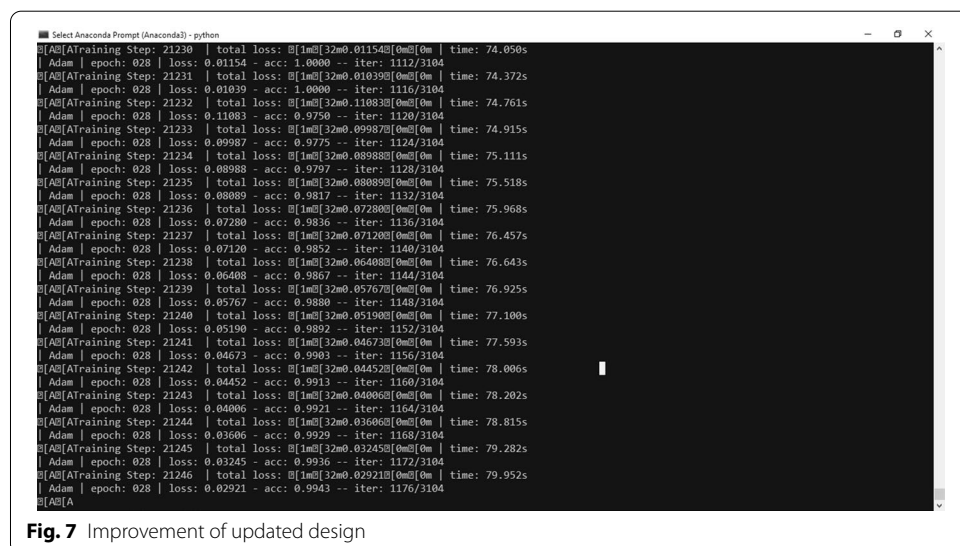
In L2 regularization, the sum of all the parameters squared is calculated and added it with the square difference of the actual output and predictions, the value of the parameters will decrease as L2 will penalize the parameters.

**Batch normalization**

Batch normalization is a used technique to evolve the design of trained model in machine learning and improve the speed, performance, and stability of artificial neural networks. It is used to normalize the input layer by re-centering and rescaling [33]. batch normalization allows each layer of a network to learn by itself a little bit more independently of other layers. There are other normalization methods, local response normalization (LRN). LRN is a non-trainable layer that square-normalizes the pixel values in a feature map in a within a local neighborhood. Batch Normalization (BN): is a trainable layer normally used for addressing the issues of Internal Covariate Shift (ICF); it increases the stability of a neural network generally. ICF arises due to the changing distribution of the hidden neurons/activation. The output for normalization for some of pixels centered in X is calculated as follows where Y is the output.

$$\mu = \frac{1}{m} \sum_{i=1}^m X_i \tag{1}$$

$$\sigma = \frac{1}{m} \sum_{i=1}^m (X_i - \mu)^2 \tag{2}$$



**Fig. 7** Improvement of updated design

$$\hat{X} = \frac{X - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{3}$$

$$Y = \lambda * \hat{X} + \beta \tag{4}$$

$\lambda$  and  $\beta$  are trainable parameters to get best performance.

The updated design is shown in Fig. 8: BM: stands for Batch Normalization

*Input* →

CL1 → PL1 → BN1 → CL2 → PL2 → BN1 →

CL3 → PL3 → BN1 → CL4 → PL4 → BN1 →

CL5 → PL5 → BN1 → FC1 → Dropout → FC2 → Dropout → FC3 →

*Regression*

CL1: 50 filters, Kernel size = 3, activation function: relu, bias:true, regularizer = 'L2'

CL2: 100 filters, Kernel size = 3, activation function: relu, bias:true, regularizer = 'L2'

CL3: 200 filters, Kernel size = 3, activation function: relu, bias:true, regularizer = 'L2'

CL4: 100 filters, Kernel size = 3, activation function: relu, bias:true, regularizer = 'L2'

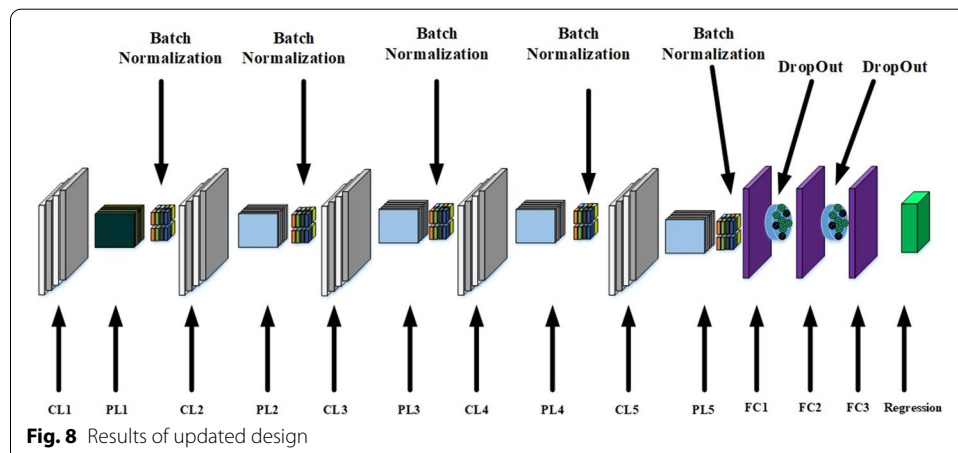
CL5: 50 filters, Kernel size = 3, activation function: relu, bias:true, regularizer = 'L2'

Without regularizing convolution layers, they may learn a over-fitted feature extraction which is not generalizable. It means that the features would be very distinctive for training set while they are not for the test set. If features are over-fitted, the model also may be over-fitted.

The overall accuracy has improved very well to became 100 % at epoch 54 as shown in Fig. 9. and validation accuracy is about 82 %. Moreover, the speed of converging is increased as show in Fig. 10.

Figure 11 explains BN and other modes of Normalization for image (In convolution layers) with dimensions H: Height, W: Width [33] There are:

- Batch norm
- Layer norm
- Instance norm
- Group norm



```

Select Administrator: Anaconda Prompt (Anaconda3) - python
| Adam | epoch: 053 | loss: 0.00681 - acc: 1.0000 -- iter: 3086/3104
| AB|A|Training Step: 82248 | total loss: 0[1m]32m0.006138[0m]0[0m] | time: 458.877s
| Adam | epoch: 053 | loss: 0.00613 - acc: 1.0000 -- iter: 3088/3104
| AB|A|Training Step: 82249 | total loss: 0[1m]32m0.005510[0m]0[0m] | time: 459.188s
| Adam | epoch: 053 | loss: 0.00551 - acc: 1.0000 -- iter: 3090/3104
| AB|A|Training Step: 82250 | total loss: 0[1m]32m0.004960[0m]0[0m] | time: 459.526s
| Adam | epoch: 053 | loss: 0.00496 - acc: 1.0000 -- iter: 3092/3104
| AB|A|Training Step: 82251 | total loss: 0[1m]32m0.004470[0m]0[0m] | time: 459.861s
| Adam | epoch: 053 | loss: 0.00447 - acc: 1.0000 -- iter: 3094/3104
| AB|A|Training Step: 82252 | total loss: 0[1m]32m0.004020[0m]0[0m] | time: 460.220s
| Adam | epoch: 053 | loss: 0.00402 - acc: 1.0000 -- iter: 3096/3104
| AB|A|Training Step: 82253 | total loss: 0[1m]32m0.003620[0m]0[0m] | time: 460.562s
| Adam | epoch: 053 | loss: 0.00362 - acc: 1.0000 -- iter: 3098/3104
| AB|A|Training Step: 82254 | total loss: 0[1m]32m0.004200[0m]0[0m] | time: 460.954s
| Adam | epoch: 053 | loss: 0.00420 - acc: 1.0000 -- iter: 3100/3104
| AB|A|Training Step: 82255 | total loss: 0[1m]32m0.003780[0m]0[0m] | time: 461.326s
| Adam | epoch: 053 | loss: 0.00378 - acc: 1.0000 -- iter: 3102/3104
| AB|A|Training Step: 82256 | total loss: 0[1m]32m0.003470[0m]0[0m] | time: 500.971s
| Adam | epoch: 053 | loss: 0.00347 - acc: 1.0000 | val_loss: 1.72696 - val_acc: 0.8238 -- iter: 3104/3104
s
Training Step: 82257 | total loss: 0[1m]32m0.003130[0m]0[0m] | time: 0.300s
| Adam | epoch: 054 | loss: 0.00313 - acc: 1.0000 -- iter: 0002/3104
| AB|A|Training Step: 82258 | total loss: 0[1m]32m0.003720[0m]0[0m] | time: 0.561s
| Adam | epoch: 054 | loss: 0.00372 - acc: 1.0000 -- iter: 0004/3104
| AB|A|Training Step: 82259 | total loss: 0[1m]32m0.003380[0m]0[0m] | time: 0.876s
| Adam | epoch: 054 | loss: 0.00338 - acc: 1.0000 -- iter: 0006/3104
| AB|A|Training Step: 82260 | total loss: 0[1m]32m0.003040[0m]0[0m] | time: 1.170s
| Adam | epoch: 054 | loss: 0.00304 - acc: 1.0000 -- iter: 0008/3104
| AB|A|Training Step: 82261 | total loss: 0[1m]32m0.002740[0m]0[0m] | time: 1.481s
| Adam | epoch: 054 | loss: 0.00274 - acc: 1.0000 -- iter: 0010/3104
| AB|A|Training Step: 82262 | total loss: 0[1m]32m0.002470[0m]0[0m] | time: 1.811s
| Adam | epoch: 054 | loss: 0.00247 - acc: 1.0000 -- iter: 0012/3104
| AB|A|Training Step: 82263 | total loss: 0[1m]32m0.002220[0m]0[0m] | time: 2.107s
| Adam | epoch: 054 | loss: 0.00222 - acc: 1.0000 -- iter: 0014/3104
| AB|A|

```

Fig. 9 Final Results

```

Select Anaconda Prompt (Anaconda3) - python
| AB|A|Training Step: 1026 | total loss: 0[1m]32m2.287260[0m]0[0m] | time: 287.164s
| Adam | epoch: 001 | loss: 2.28726 - acc: 0.4391 -- iter: 3078/3104
| AB|A|Training Step: 1027 | total loss: 0[1m]32m2.302200[0m]0[0m] | time: 287.433s
| Adam | epoch: 001 | loss: 2.30220 - acc: 0.4619 -- iter: 3081/3104
| AB|A|Training Step: 1028 | total loss: 0[1m]32m2.114930[0m]0[0m] | time: 287.692s
| Adam | epoch: 001 | loss: 2.11493 - acc: 0.5157 -- iter: 3084/3104
| AB|A|Training Step: 1029 | total loss: 0[1m]32m1.922910[0m]0[0m] | time: 287.955s
| Adam | epoch: 001 | loss: 1.92291 - acc: 0.5641 -- iter: 3087/3104
| AB|A|Training Step: 1030 | total loss: 0[1m]32m1.854660[0m]0[0m] | time: 288.209s
| Adam | epoch: 001 | loss: 1.85466 - acc: 0.5744 -- iter: 3090/3104
| AB|A|Training Step: 1031 | total loss: 0[1m]32m1.815620[0m]0[0m] | time: 288.503s
| Adam | epoch: 001 | loss: 1.81562 - acc: 0.5836 -- iter: 3093/3104
| AB|A|Training Step: 1032 | total loss: 0[1m]32m1.791130[0m]0[0m] | time: 288.760s
| Adam | epoch: 001 | loss: 1.79113 - acc: 0.5919 -- iter: 3096/3104
| AB|A|Training Step: 1033 | total loss: 0[1m]32m1.824520[0m]0[0m] | time: 289.020s
| Adam | epoch: 001 | loss: 1.82452 - acc: 0.5994 -- iter: 3099/3104
| AB|A|Training Step: 1034 | total loss: 0[1m]32m1.951860[0m]0[0m] | time: 289.279s
| Adam | epoch: 001 | loss: 1.95186 - acc: 0.5395 -- iter: 3102/3104
| AB|A|Training Step: 1035 | total loss: 0[1m]32m1.982050[0m]0[0m] | time: 309.266s
| Adam | epoch: 001 | loss: 1.98205 - acc: 0.5188 | val_loss: 2.62824 - val_acc: 0.3869 -- iter: 3104/3104
--
Training Step: 1036 | total loss: 0[1m]32m2.033950[0m]0[0m] | time: 0.298s
| Adam | epoch: 002 | loss: 2.03395 - acc: 0.4670 -- iter: 0003/3104
| AB|A|Training Step: 1037 | total loss: 0[1m]32m2.043960[0m]0[0m] | time: 0.599s
| Adam | epoch: 002 | loss: 2.04396 - acc: 0.4203 -- iter: 0006/3104
| AB|A|Training Step: 1038 | total loss: 0[1m]32m1.893030[0m]0[0m] | time: 0.886s
| Adam | epoch: 002 | loss: 1.89303 - acc: 0.4782 -- iter: 0009/3104
| AB|A|Training Step: 1039 | total loss: 0[1m]32m1.765750[0m]0[0m] | time: 1.156s
| Adam | epoch: 002 | loss: 1.76875 - acc: 0.5304 -- iter: 0012/3104
| AB|A|Training Step: 1040 | total loss: 0[1m]32m1.680390[0m]0[0m] | time: 1.415s
| Adam | epoch: 002 | loss: 1.68039 - acc: 0.5440 -- iter: 0015/3104
| AB|A|Training Step: 1041 | total loss: 0[1m]32m1.560860[0m]0[0m] | time: 1.657s
| Adam | epoch: 002 | loss: 1.56086 - acc: 0.5896 -- iter: 0018/3104
| AB|A|Training Step: 1042 | total loss: 0[1m]32m1.748860[0m]0[0m] | time: 1.899s
| Adam | epoch: 002 | loss: 1.74886 - acc: 0.5307 -- iter: 0021/3104

```

Fig. 10 Speed is improved

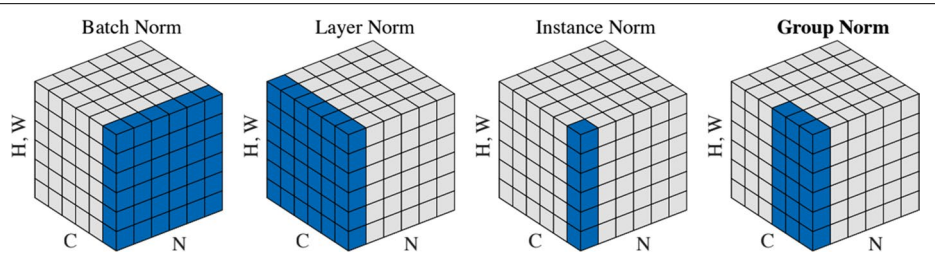


Fig. 11 Batch explain

- Weight norm
- Batch-Instance norm
- Switchable norm

**Table 4** Image augmentation

---

```
datagen = ImageDataGenerator(  
Rotation-range = 40,  
Shear-range = 0.2,  
Zoom-range = 0.2,  
Horizontal-flip = True,  
Brightness-range = (0.5, 1.5))
```

---

**Fig. 12** Simple image for augmentation**Image augmentation**

Image augmentations have become a common implicit regularization technique to combat over-fitting in deep learning models and are ubiquitously used to improve performance. Common transformations that are typically used: flipping, rotating, scaling, and cropping. In our case, we used the next Image augmentation function (Table 4)

- Rotation-range is a value in degrees (0–180), a range within which to randomly rotate pictures.
- Shear-range is for randomly applying shearing transformations.
- Zoom-range is for randomly zooming inside pictures.
- Horizontal-flip is for randomly flipping half of the images horizontally –relevant when there are no assumptions of horizontal asymmetry.
- Brightness-range: Tuple or list of two floats. Range for picking a brightness shift value from.

for an image in the Fig. 12:

the resulted augmentation is shown in Fig. 13:

So the training dataset will be increased six times.





**Table 5 Results comparison**

Without IA	With IA (%)	Notes
82	96.23	200 persons recognition
93.62	NA	800 persons recognition (With IA is Not Available due to memory issue because of huge amount of data for 800 persons images with augmentation)

**Performance analysis**

Performance was validated using Market dataset, we have choose samples from the dataset to work on. For 200 persons in recognition, training dataset was increased by augmentation to 21557 images and validation accuracy has remarkable increased to reach 96.23

For 800 person, the accuracy was 93.62% without image augmentation. Image augmentation for 800 persons was not applied because memory issue, since 800 persons are represented by more than 11000 images (more than 66000 images with augmentation).

Comparison of applied augmentation Machine Learning models is in Table 5.

Parameters were selected depending on some goals. For each parameters, two main goals were considered:

- High accuracy.
- Speed of convergence.

So we can summarize the main advantage of this work as follows:

- Image augmentation for gait recognition was applied with CNN algorithm.
- Best CNN parameters.
- Best CNN design and structure.

The proposed model consists of typical CNN layers with some specific properties:

- First pooling layer with max function as pooling type
- Rest pooling layers in CNN structures with average function as pooling type.
- Batch normalization after every pooling layer.
- Convolution layers with regularizer L2.

After selection best parameters for our proposed model, image augmentation techniques were implemented to make the final model robust as possible.

Comparing to [10] in selected hyper parameters

Ours: LR:8e−4, Epochs:50, Batch size: 10, Optimizer: Adam.

Them: LR:1e−3, Epochs:50, Batch size: 20, Optimizer: Adam.

The average recognition rate reaches 97%, while in ours it reaches 96.23.

There are not many studies with image augmentation to compare with. In addition, No studies are worked on Market dataset.

### Conclusion and future scope

In summary, This work proposed simple and robust model for person recognition using gait model features based on CNN algorithm, this model resulted with edits in the design of CNN model and choosing hyper parameters for some parts of CNN model. This study also introduce Image augmentation in recognition; that helps to make the simple models robust to some changes in images of persons, it generate images of the same frame or view with different conditions. The final model was validated and it performed well and better than background studies.

We can think about some points:

- Improve person recognition by rebuilding implemented methods can be rebuilt with other batch normalization modes and with some pre-processing steps for data-set.
- Using genetic algorithm for some parameters; it needs more processing time and high performance processors.
- Search deeply in Fully connected layer to figure out the validity of changing activation function, or manually select activation function.
- For IA, new conditions can be added to generate more images that handling other variation in images.

The main effective scope is to implement it in real time system for real useful goals.

### Abbreviations

CNN: Convolution Neural Network; SVM: Support Vector Machine; NN: neural network; PCA: Principal Component Analysis; PRGM: Person Recognition based on Gait Model.

### Acknowledgements

Thanks for Mr. AA, for his co-operation and help.

**Authors' contributions**

AS took the role of performing the literature review, working on CNN algorithm. TH took on a supervisory role and oversaw the completion of the work, Proposed data set for implementation and validation. All authors read and approved the final manuscript.

**Funding**

The authors declare that they have no funding.

**Availability of data and materials**

The data set is available to public and can be found in <https://drive.google.com/file/d/0B8-rUzbwVRk0c054eEozWG9COHM/view>.

**Ethics approval and consent to participate**

The authors Ethics approval and consent to participate.

**Consent for publication**

The authors consent for publication.

**Competing interests**

The authors declare that they have no competing interests.

**Author details**

<sup>1</sup> Damascus University, Damascus, Syria. <sup>2</sup> Syrian Private University, Damascus, Syria.

Received: 3 September 2020 Accepted: 26 November 2020

Published online: 03 January 2021

**References**

- Ghaleb AEK, Amara NEB. Soft and hard biometrics for the authentication of remote people in front and side views. *Int J Appl Eng Res.* 2016;11(14):8120–7.
- Sun J, Wang Y, Li J, Wan W, Cheng D, Zhang H. View-invariant gait recognition based on kinect skeleton feature. *Multimedia Tools Appl.* 2018;77(19):24909–35.
- Zhang Y, Huang Y, Wang L, Yu S. A comprehensive study on gait biometrics using a joint cnn-based method. *Pattern Recogn.* 2019;93:228–36.
- Strukova O, Shiripova L, Myasnikov E. Gait analysis for person recognition using principal component analysis and support vector machines. *CEUR Workshop Proc.* 2018;2210:170–6.
- Wang X, Zhao R. Person re-identification: System design and evaluation overview. In: *Person Re-Identification*, Springer 2014;351–370.
- Hahnel M, Klunder D, Kraiss K-F. Color and texture features for person recognition. In: *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*, IEEE 2004;1:647–652.
- Zou Q, Wang Y, Wang Q, Zhao Y, Li Q. Deep learning-based gait recognition using smartphones in the wild. *IEEE Trans Inform Forensics Security.* 2020;15:3197–212.
- Charalambous CC, Bharath AA. A data augmentation methodology for training machine/deep learning gait recognition algorithms. 2016; arXiv preprint [arXiv:1610.07570](https://arxiv.org/abs/1610.07570)
- Simhi N, Yovel G. Dissociating identity from gait: A virtual reality study of the role of dynamic identity signatures in person recognition 2019;
- Elharrouss O, Almaadeed N, Al-Maadeed S, Bouridane A. Gait recognition for person re-identification. *J Supercomput.* 2020. <https://doi.org/10.1007/s11227-020-03409-5>.
- Zhang Z, Tran L, Yin X, Atoum Y, Liu X, Wan J, Wang N. Gait recognition via disentangled representation learning. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019;4710–4719.
- Terrier P. Gait recognition via deep learning of the center-of-pressure trajectory. 2019; arXiv preprint [arXiv:1908.04758](https://arxiv.org/abs/1908.04758)
- Alotaibi M, Mahmood A. Improved gait recognition based on specialized deep convolutional neural network. *Comput Vision Image Understanding.* 2017;164:103–10.
- Nguyen CH, Tran LH, Ho KN. Application of neural network to predict the workability parameters of self-compacting concrete. In: *CIGOS 2019, Innovation for Sustainable Infrastructure*, Springer, 2020;1161–1166.
- Assad A, Khalaf W, Chouaib I. Radial basis function kalman filter for attitude estimation in gps-denied environment. *IET Radar, Sonar & Navigation.* 2020;14(5):736–46.
- Krenn M, Zeilinger A. Predicting research trends with semantic and neural networks with an application in quantum physics. *Proceedings of the National Academy of Sciences.* 2020;.
- De Marsico M, Mecca A. Gait recognition: The wearable solution. In: *Human Recognition in Unconstrained Environments*, Elsevier 2017:177–195.
- Kawakami T. Coronavirus gives China more reason to employ biometric tech. *Nikkei Asian Review* 2020; <https://asia.nikkei.com/Business/China-tech/Coronavirus-gives-China-more-reason-to-employ-biometric-tech>
- Makihara Y, Suzuki A, Muramatsu D, Li X, Yagi Y. Joint intensity and spatial metric learning for robust gait recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017:5705–5715.
- Shorten C, Khoshgoftaar TM. A survey on image data augmentation for deep learning. *J Big Data.* 2019;6(1):60.
- Yamashita R, Nishio M, Do RKG, Togashi K. Convolutional neural networks: an overview and application in radiology. *Insights Into Imaging.* 2018;9(4):611–29.

22. Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, Liu T, Wang X, Wang G, Cai J, et al. Recent advances in convolutional neural networks. *Pattern Recogn.* 2018;77:354–77.
23. Castro FM, Marín-Jiménez MJ, Guil N, De La Blanca NP. Automatic learning of gait signatures for people identification. In: *International Work-Conference on Artificial Neural Networks*, Springer 2017:257–270.
24. Shiraga K, Makihara Y, Muramatsu D, Echigo T, Yagi Y. Geinet: View-invariant gait recognition using a convolutional neural network. In: *2016 International Conference on Biometrics (ICB)*, 2016:1–8. IEEE
25. Terrier P. Gait recognition via deep learning of the center-of-pressure trajectory. *Appl Sci.* 2020;10(3):774.
26. Sagawa R, Shiba Y, Hirukawa T, Ono S, Kawasaki H, Furukawa R. Automatic feature extraction using cnn for robust active one-shot scanning. In: *2016 23rd International Conference on Pattern Recognition (ICPR)*, 2016:234–239. IEEE, New York.
27. Li Y, Liu D, Li H, Li L, Li Z, Wu F. Learning a convolutional neural network for image compact-resolution. *IEEE Trans Image Processing.* 2018;28(3):1092–107.
28. Lawrence S, Giles CL, Tsoi AC, Back AD. Face recognition: a convolutional neural-network approach. *IEEE Trans Neural Networks.* 1997;8(1):98–113.
29. Wang Y, Bian Z-P, Hou J, Chau L-P. Convolutional neural networks with dynamic regularization. 2019; arXiv preprint [arXiv:1909.11862](https://arxiv.org/abs/1909.11862)
30. Gal Y, Ghahramani Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: *International Conference on Machine Learning*, 2016:1050–1059
31. Market Dataset. <https://www.aitribune.com/dataset/2018051063>
32. Kingma DP, Ba J. Adam: A Method for Stochastic Optimization 2017; [arXiv:1412.6980](https://arxiv.org/abs/1412.6980)
33. Wu Y, He K. Group normalization. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018:3–19

### Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Submit your manuscript to a SpringerOpen<sup>®</sup> journal and benefit from:**

- ▶ Convenient online submission
- ▶ Rigorous peer review
- ▶ Open access: articles freely available online
- ▶ High visibility within the field
- ▶ Retaining the copyright to your article

---

Submit your next manuscript at ▶ [springeropen.com](https://www.springeropen.com)

---