Human-Centric Intelligent Systems (2023) 3:13–24
https://doi.org/10.1007/s44230-022-00013-z

RESEARCH ARTICLE

Convolutional Auto-Encoder and Independent Component Analysis Based Automatic Place Recognition for Moving Robot in Invariant Season Condition

Md. Tariqul Islam[1] · Khan Md. Hasib[2] · Md. Mahbubur Rahman[3] · Abdur Nur Tusher[4] · Mohammad Shafiul Alam[2] · Md. Rafiqul Islam[5]

Received: 21 July 2022 / Accepted: 20 November 2022 / Published online: 5 December 2022

Abstract
Building up a map is essential for mobile robots to localize their position and perfect autonomous navigation which is known as Simultaneous Localization and Mapping (SLAM). The map has become very important when the weather is inappropriate for the robot. However, the map becomes inconsistent when the robot moves in the environment and detects errors in its detection accuracy. The robot had difficulty identifying its previously visited path, which is called loop-closure detection when the climate changed immensely e.g. seasonal changes. The main goal of this work is to apply Independent Component Analysis (ICA) and Auto-Encoder (Convolutional Auto-Encoder and Fundamental Auto-Encoder) to understand the route through the robot. During the operation of robots across a wide range of environmental changing conditions, the ICA has auspicious potential to extract descriptors of condition-invariant images. On the other hand, Auto-Encoder has the capability to differentiate condition variant and condition invariant characteristics of a site and identify the most possible route for the robot. In order to complete this work perfectly, we used three seasonal datasets, they are Summer–Fall, Spring–Fall, and Summer–Spring datasets. This work uses the baseline method with a precision-recall curve and evaluates the performance of our proposed algorithm, especially the ICA algorithm. In short, the proposed algorithm ICA showed a 91.05% accuracy rate which is better than the baseline algorithm.

Keywords Auto-encoder · Independent Component Analysis · Machine learning · Deep learning · Principle Component Analysis · SLAM

1 Introduction

With the advancements of modern technology, a moving robot moves through an unknown environment and uses a map for autonomous navigation. As the localization and mapping are done concurrently, the process is known as Simultaneous Localization and Mapping (SLAM). In recent years, the SLAM algorithms have been applied in various environments (indoor and outdoor). Loop-closure detection, map estimation, visual odometry, and sensor data processing are the main module in the SLAM system. Though loop-closure detection is one of the great modules among the modules that consist of the Simultaneous Localization and Mapping (SLAM) visual system, the main problem of the loop-closure detection module is unable to detect frequent addresses and places. While it has the capability to assist the visual SLAM module because it can diminish the position defects significantly

✉ Khan Md. Hasib
khanmdhasib.aust@gmail.com

✉ Md. Rafiqul Islam
rafiqulislam.cse24@gmail.com

[1] Department of Electronics and Communication Engineering, Khulna University of Engineering and Technology, Khulna, Bangladesh

[2] Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh

[3] Department of Civil Engineering, European University of Bangladesh, Dhaka, Bangladesh

[4] Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh

[5] Data Science Institute (DSI), University of Technology Sydney (UTS), Sydney, Australia

over time and has the power to provide help to the SLAM module to construct an environmental map, it is necessary to introduce a model which has the capability to recognize place and address frequently.

The state-of-the-art algorithms for autonomous navigation are generally used based on the appearance of a scene. However, correct loop-closure detection is a challenging task in complex environments, where the environment experiences enormous variation in the outlook. In different seasons when the weather faces extreme change, the outlook of a place can be looked at differently. Appearance-based image matching is a popular technique for building a loop-closure detection model. The idea of the image matching technique is to find the match of a current view of a moving robot with the memory of the previously visited locations. This process is accomplished mainly in two steps; image description and similarity measurement.

Image description refers to representing a scene that is more discriminating than the original image. The success of the SLAM algorithm relies heavily on the robust representations of the scenes; hence the condition of constant feature learning is a crucial part the loop-closure detection. For example, Lowry et al. [24] showed that the Principal Component Analysis (PCA) could be used to retrieve robust representations of the scenes by directly applying PCA to the intensity images. This work is chosen as a baseline algorithm in this work. The basic images that are obtained from PCA depend on second-order statistics, but most of the perceptually distinguishable information is associated with higher-order statistics. Independent Component Analysis (ICA) is one technique that deals with the fourth-order moment [7]. This work intends to use ICA to obtain the condition-invariant representations of the scenes across different seasons. Although the hand-crafted features succeed in appearance-based loop closure detection, they are still not suitable in situations, e.g., illumination or seasonal changes. From the success the neural network-based feature learning, an unsupervised auto-encoder has been proposed to learn robust features across different seasons.

Visual localization performs robustly in outdoor environments where the environment does not have extreme perceptual changes. Vision-based systems struggle to identify the same place under different lighting changes, seasons, or weather conditions. Traditional approaches use key point-based descriptors, which do not remain stable in different seasons. As a result, the local descriptors, such as SIFT, SURF, etc., do not perform well in seasonal changes in the environment. The main focus of this work is to learn the condition-invariant features across extreme seasonal changes. Instead of using hand-engineered features, this work aims at discovering the condition-invariant image representation. The contributions of this paper are divided into four key points and these points are included below as follows:

- First, we collected the data-processed cessing the data so that the dataset becomes suitable for our proposed models. We also split the dataset into two sections called testing (20%) and training (80%) for training the models.
- Second, we provided an in-depth description of Independent Component Analysis (ICA) in order to obtain the condition-invariant representations of the scenes across different seasons.
- Third, we described Auto-encoder (FAE and CAE) precisely for distinguished two features sight variant condition and sight invariant condition in order to discover the possible route.
- Fourth, we provided a comparative analysis between the existing baseline algorithm (PCA) and our proposed models (ICA and Auto-encoder) so that we can easily and appropriately identify, justify and assess our models. Additionally, we provide some challenges, scopes, and future work for the upcoming researchers so that during further research, they can get help from previous research and avoid pitfalls.

The rest of the paper is organized as follows: Sect. 2 and Sect. 3 present related work and research Methodology respectively. The experimental outcomes and run-time analysis of the research work are presented in Sect. 4. Opportunities, challenges, and future directions are described in Sect. 5. Finally, the conclusion is presented in Sect. 6.

2 Literature Review

In the machine vision business, object recognition is frequently utilized for inspection, registration, and manipulation. However, modern commercial object recognition systems rely nearly entirely on correlation-based template matching. While template matching is very effective in certain engineered environments with tightly controlled object pose and illumination, it becomes computationally infeasible when object rotation, scale, illumination, and 3D pose are allowed to vary, and even more so when dealing with partial visibility and large model databases. Instead of examining all picture locations for matches, extracting features from the image that are at least somewhat invariant to the image generation process and matching solely to those features is an option. Many possible feature types have been presented and investigated, including line segments [7], edge groups [17], and regions [1], among many others. While these traits have performed effectively for particular object classes, they are typically not identified frequently or consistently enough to provide the foundation for accurate identification.

Recent work has focused on creating significantly denser collections of visual features. One method has been to utilize a corner detector (or, more precisely, a detector

of peaks in local image variance) to find recurrent picture regions around which local image attributes may be monitored. The Harris corner detector was used by Zhang et al. [25] to locate feature sites for the unipolar alignment of pictures collected from different perspectives. Rather than trying to correlate areas from one picture against all conceivable regions in a second image, merely matching regions cantered at corner points in each image resulted in significant time savings. Schmid and Mohr [22] employed the Harris corner detector to identify interest sites in the object identification issue and then built a local image descriptor at each interest point using an orientation-invariant vector of derivative-of-Gaussian image measurements.

These image descriptors were employed for robust object detection by searching for numerous matching descriptors that met object-based orientation and placement limitations. This study was outstanding in terms of both the speed with which it recognized photographs in a vast database and its ability to handle crowded images.

The corner detectors employed in the preceding techniques have a key flaw in that they only evaluate a picture at a single scale. These detectors react to distinct picture spots when the scale shift gets considerable. Furthermore, since the detector does not offer an indication of the object size, picture descriptors must be created and a vast number of scales must be attempted to match. This study presents a fast way to find stable key positions in scale space. This implies that differing picture scaling have no influence on the set of key places chosen. Again, Handcrafted feature-based approaches collect various aspects from face photos to build strong feature vectors that are then used to train classifiers such as SVM [9], LDA, BPNN, and others. These techniques investigate textural, picture quality, and motion-based characteristics of face photographs in order to distinguish between real and fake face photos. The motion-related properties demonstrated by the movement of the eyes, face, and head motions in motion-based techniques. Various scholars have put forth a lot of effort to investigate motion-related aspects. Similarly, image quality characteristics are being investigated in order to extract quality-related aspects from pictures for face ant spoofing techniques [6]. The techniques based on textural features are explained in the following paragraphs.

Textural feature information is collected from face photos in texture feature-based techniques, and these features are utilized to distinguish real face photographs from false ones. In the literature, texture feature descriptors such as Local Binary Patterns (LBP), HoG, LPQ, Gabor wavelet, SURF, and others are utilized to solve face PAs. Among all, the LBP and its derivatives are the most commonly used descriptors; this might be owing to improved identification results or algorithm efficiency. Maatta et al. [18] and Chingovska et al. [5] first investigated the multi-scale LBP descriptor to fight face picture and replay assaults. However, a paradigm change began in 2012, when a deep learning-based model, AlexNet [15], easily won the ImageNet competition. Deep learning models have since been applied to a variety of challenges in computer vision and Natural Language Processing (NLP), with promising results. Not unexpectedly, biometric identification systems were among the first to be supplanted by deep learning models (with a few year's delay). Models based on deep learning offer an end-to-end learning system capable of learning feature representations while doing classification/regression This is accomplished by using multi-layer neural networks, also known as Deep Neural Networks (DNNs) [8], to learn several layers of representations that correspond to various degrees of abstraction, which is more suited to uncovering underlying data patterns. A multi-layer neural network was first proposed in the 1960s [26].

However, its feasibility was an issue in and of itself, since the training period would be prohibitively long (because to a lack of powerful computers at the time). Scientists were able to train very deep neural networks much faster thanks to advances in processor technology, particularly the development of General-Purpose GPUs (GPGPUs), as well as the development of new techniques (such as Dropout) for training neural networks with a lower chance of over-fitting [16]. The basic principle behind a neural network is to route (raw) input through a network of linked neurons or nodes, each of which emulates a linear or nonlinear function depending on its own weights and biases. These weights and biases would change during training via backpropagation of the gradients from the output [21], which usually resulted from differences between the expected and actual current outputs, with the goal of minimizing a loss function or cost function (difference between the predicted and actual outputs according to some metric) [3] (Table 1).

Table 1 Related work of EEG Channel prediction using various techniques

References	Key purposes	Model
Bellekens et al. [2]	Analyze numerous for place recognition	Cloud coarse registration methods
Marcel et al. [19]	Detect the motion of the objects	SVM, LDA, BPNN
Chen et al. [4]	Place classification task	AMOSNet and HybridNet
Noh et al. [21]	Landmark recognition task	Global descriptor adaptation
Wang et al. [24]	Place detection	Omnidirectional CNNs

All of these studies have shown the most efficient and accurate categorization approach used in object recognition systems. The performance comparison of the ml and dl classifiers with other kinds of classification algorithms is very interesting. However, no studies have been discovered that examine the best technique to apply the automated location identification method in object recognition systems. In this paper, the main motive of this work is to promote a model which is capable to understand the route through the route presented robustly. This algorithm can also work during seasonal changes (Summer to Fall, Fall to Winter, Winter to Summer) and is capable to helps loop-closure detection accurately.

3 Materials and Methods

In this section, we discuss the data collection and data preprocessing, an in-depth description of the baseline algorithm (PCA), and proposed algorithms (ICA and auto-encoder). Figure 1 precisely represent the overall system structure.

3.1 Data Collection and Processing

In this work, we have used the northland dataset which is released by the Norway television company in 2012. This dataset is an open dataset, recording a 728 km long train journey between Trondheim and Bodo in north Norway. The train took 10 h to complete the journey and has been recorded data in three different seasons: summer, fall, and spring through the same route. The resolution of the data is 1920 × 1080 and 25 frames per second.

In order to enhance the quality of the image and get an accurate result, data augmentation and data enhancement techniques are used. Data are extracted from the original dataset by data augmentation techniques including horizontal flip, width shift, height shift, and rotation. All the parameters of augmentation used in this work are shown in Table 2. After image augmentation, the dataset is increased.

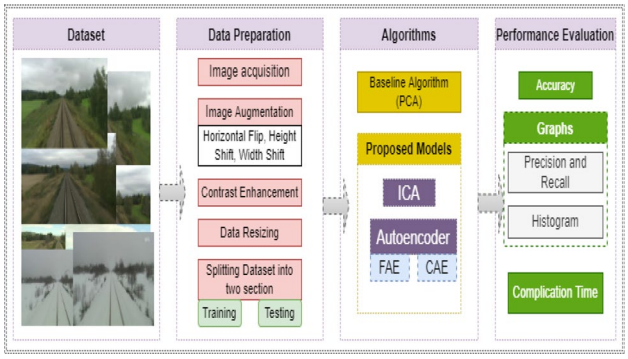

Fig. 1 Working diagram of the system

Table 2 Data augmentation parameters

Augmentation technique	Range
Horizontal flip	True
Width shift	0.2
Height shift	0.2
Rotation	False
Vertical flip	False

The image frames are extracted 1 frame per second from the original videos which provide 35,768 images from each season. Then the image is down-sampled into 32 × 64 pixels and each image is converted to a greyscale image.

The dataset is partitioned into two sections one is training and the other one is testing according to the baseline paper [7]. The test dataset is consisted using 3569 images in each season which is previously unseen when the train went through the tunnels and waited in the train station. The total dataset description is provided in Table 3.

3.2 Selected Method

3.2.1 Baseline Algorithm

Lowry et al. [24] showed that the PCA can be used to retrieve robust representations of the scenes by directly applying the PCA to the intensity images. The idea of this algorithm is demonstrated in Fig. 2.

Figure 2 shows that the condition-dependent features of the scenes are associated with the first few principal components. So, by discarding the first few principal components and by choosing the subsequent principal components, the condition-independent or condition-invariant features can be learned. This state-of-the-art algorithm has been chosen as a baseline algorithm in this thesis work. We used three combinations of datasets (summer–fall, spring–fall, and summer–spring). For completing the loop, the robot used the same route which means in the summer–fall dataset, during moving the robot in the summer season the images

Table 3 Details of dataset

Feature	Value
Total number of images	143,072
Total number of training images	128,796
Total number of test images	14,276
Dimension (pixels)	32 × 64
Colour	Grey scale
Total number of images from each season	35,768
Total number of training images from each season	32,199
Total number of test images from each season	3569

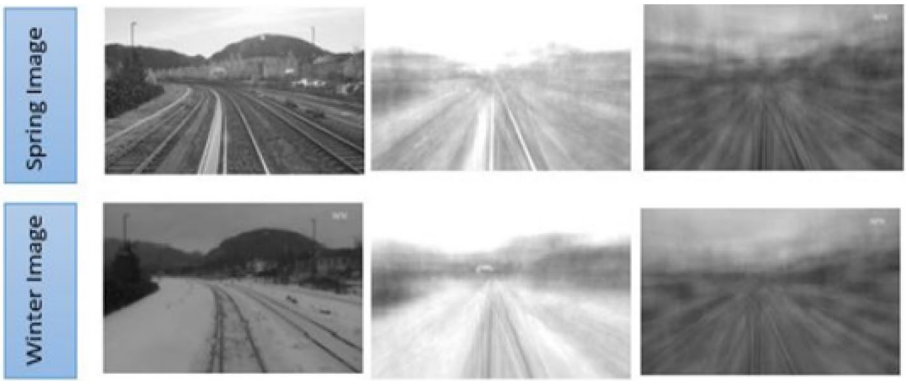

Fig. 2 First column: Two images from the same locations of the two different seasons [Northland dataset]. Second column: Images are projected on the first 100 principal components (PC's) Third column: Images are projected on the second 100 principal components (PC's)

of the route are stored in internal memory, during the fall season, when the robot was coming back following the same route visited in the summer season, it also stores the images of the same route. In this algorithm, the training dataset is constructed with 5000 images, where each season (spring, summer, and fall) contributed 2500 images equally. For instance, during the summer–fall dataset if the summer season contributed 2500 images then the fall season also contributed the same amount of images. Similarly, the test dataset is constructed with 900 images and each season contributed 450 images in a similar way. The principal components are learned from this training dataset and those principal components are used for the test dataset. After the PCA decomposition, the test images are projected onto the learned PCA dimensions and removed from the first 50 PCA dimensions [13]. Then the image is converted back to the pixel space. These converted images are the condition-invariant representations of the scenes in the loop-closure detection problem.

Principle Component Analysis (PCA): Principal Component Analysis (PCA) is a dimension reduction algorithm where a huge amount of variables are described by a smaller number of variables without the major loss of information, i.e., the low dimensional data will be still able to explain the original data. For example, an object can be described by numerous properties [14]. PCA tries to summarize the properties of an object by finding a low-dimensional linear subspace onto which the data can be projected. PCA reduces data set dimensions while attempting to keep as many variations in the data set as possible when there is an imbalanced issue [11] in the dataset. To this end, PCA tries to preserve the maximum possible variance in the data.

From Fig. 3, the first principal component holds the largest possible variance direction, the second component holds the second largest variance direction, and so on, and this scene is presented in Fig. 3. Consider an n-dimensional dataset, $X = x^1, x^2, x^3, \ldots\ldots, x^p$, where P is the number of samples or observations. Each sample x^i, where $i = 1,2,3,..,P$ is an n-dimensional column vector, called a feature vector. The dataset can be represented in terms of $n \times P$ matrix X. Each

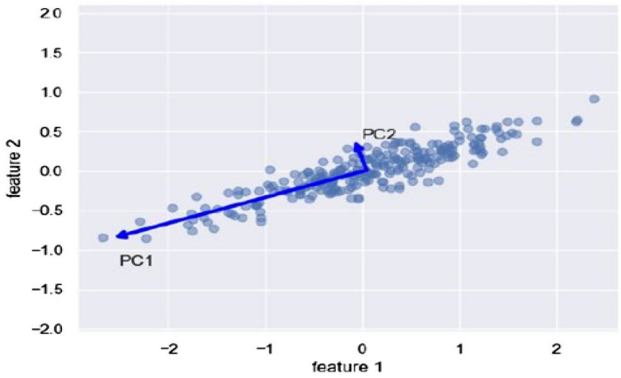

Fig. 3 Illustration of the principal component analysis. PC1 is the direction of the maximum variance of the data points and PC2 is the direction of the second maximum variance

column of matrix X refers to the feature vector of a sample. We define a new matrix $W(n \times n)$ which transforms X to Y:

$$\mathrm{Y} = \mathrm{W}\ \mathrm{X}. \tag{1}$$

This equation transforms the coordinates of the original data X; geometrically, W stretches and rotates the X. Let $w_1, w_2, \ldots, w_n$ be the row vectors of W and $x^1, x^2, \ldots, x^P$ be the column vectors of the X. The result of the dot product between W and X is

$$\mathbf{WX} = \begin{matrix} \mathbf{W_1X^1}\ \mathbf{W_1X^2} \ldots\ldots\ldots\ldots\ldots\ldots \mathbf{W_1X^P} \\ \mathbf{W_2X^1}\ \mathbf{W_2X^2} \ldots\ldots\ldots\ldots\ldots\ldots \mathbf{W_2X^P} \\ \mathbf{W_3X^1}\ \mathbf{W_3X^2} \ldots\ldots\ldots\ldots\ldots\ldots \mathbf{W_3X^P} \end{matrix} = \mathbf{Y} \tag{2}$$

The data matrix X is projected onto the rows of W. Hence, the rows of W form a new basis for the columns of X, which are denoted as the principal component directions. We look for a transformation matrix W, which keeps as much as possible variance of the data. We did it by estimating the covariance matrix of the original data and getting the directions in which the variance is maximized, then used these directions as a new orthonormal basis for our data. Estimation of a

covariance matrix of $n \times P$ dimensional data X, assuming X to be zero-mean, can be calculated as:

$$\mathbf{R_{XX}} = \frac{1}{\mathbf{N}-1}\mathbf{XX^T} = \frac{1}{\mathbf{N}-1}\begin{bmatrix} \mathbf{X^1X^{1T}} & \mathbf{X^1X^{2T}} \ldots\ldots\ldots\ldots & \mathbf{X^1X^{nT}} \\ \mathbf{X^2X^{1T}} & \mathbf{X^2X^{2T}} \ldots\ldots\ldots\ldots & \mathbf{X^2X^{nT}} \\ \mathbf{X^nX^{1T}} & \mathbf{X^nX^{2T}} \ldots\ldots\ldots\ldots & \mathbf{X^nX^{nT}} \end{bmatrix}$$

The diagonal entities of R_{XX} represent the variance of the elements x^i, where $i = 1,2,3, \ldots, n$, and the off-diagonal elements represent the cross-covariance between x^i and x^j (with i is not equal to j). The matrices with this structure (which are square symmetric by their nature) are called Variance–Covariance matrices. Since our main goal is dimensionality reduction, i.e., reducing the redundancy of the data, we should keep the cross-correlation between off-diagonal elements as small as possible. Another way of saying this is that; the covariance matrix of the transformed space Y is diagonal. Therefore, one needs to find the W that diagonalizes R_{YY}, where R_{YY} is a covariance matrix of the transformed data. Using Eq. (1) for Y, R_{YY} can be written as:

$$\mathbf{R_{YY}} = \frac{1}{\mathbf{N}-1}\mathbf{YY^T}\frac{1}{\mathbf{N}-1}\mathbf{WX(WX)^T} = \frac{1}{\mathbf{N}-1}\mathbf{WR_{xx}W^T}$$

Note that R_{XX} is an $n \times n$ symmetric square matrix, therefore it can be orthonormal diagonalized as follows: $R_{XX} = U \Lambda U^T$, where U = square matrix, R_{XX} = Eigenvectors, and A = Diagonal matrix. This diagonalization step is called Eigen Value Decomposition (EVD). If we choose rows of W being Eigenvectors of R_{XX}, then we can write $W = U^T$

$$\mathbf{R_{YY}} = \frac{1}{\mathbf{N}-1}\mathbf{WR_{xx}W^T} = \frac{1}{\mathbf{N}-1}\mathbf{UU^TAUU^T} = \frac{1}{\mathbf{N}-1}\mathbf{A}$$

The inverse of an orthogonal matrix is its transpose, i.e., $U^{-1} = U^T = U^T U = I$, where I is the identity matrix. As a result, the eigenvectors of R_{XX} are the proper choice for W, as diagonalization of R_{YY} is the goal of PCA and $W = U^T$ satisfies the condition. In practice, PCA can be used on a data set X in the following way:

- Subtract the mean of the dataset to prepare the zero-centered data.
- Compute EVD of R_{XX} and sort the eigenvalues
- Collect finite amount $n^-(n^- < n)$ of eigenvectors of R_{XX} in matrix W, where n^- is a new dimensionality of the projected data Y.
- Matrix W represents a new basis for our data, so project the data onto this basis by multiplication, $Y = W_X$.

3.2.2 Proposed Models

1. Independent Component Analysis (ICA): In the Independent Component Analysis (ICA), we observed n scalar random variables $x_1, x_2, x_3, \ldots, x_n$, which are assumed to be a linear combination of $m(m \leq n)$ independent components (IC's) $s_1, s_2, s_3, \ldots, s_m$. The IC's are mutually statistically independent (i.e. one variable does not give any information about the other variables), and have zero mean. In a matrix form, we can write this relationship as follows:

$$x = As \tag{3}$$

where x is the n-dimensional observed signal vector, s contains the independent components, and A is called the mixing matrix where the matrix is,

$$\begin{matrix} \mathbf{x_1} \\ \mathbf{x}_2 \\ \vdots \\ \vdots \\ \mathbf{x_3} \end{matrix} = \begin{matrix} \mathbf{a_{11}} & \mathbf{a_{12}} \ldots\ldots\ldots\ldots & \mathbf{a_{1n}} \\ \mathbf{a}_{21} & \mathbf{a}_{22} \ldots\ldots\ldots\ldots & \mathbf{a}_{2n} \\ \vdots & & \vdots \\ \vdots & & \vdots \\ \mathbf{a_{n1}} & \mathbf{a_{n2}} \ldots\ldots\ldots\ldots & \mathbf{a_{nn}} \end{matrix} = \begin{matrix} \mathbf{s_1} \\ \mathbf{s}_2 \\ \vdots \\ \vdots \\ \mathbf{s_3} \end{matrix} \tag{4}$$

For mathematical convenience, we have assumed that the independent components have unit variance [24]. So far, the most popular application areas of ICA have been the blind source separation problem and feature extraction [12]. In the feature extraction case, the columns of A represent the feature vectors and s_i $(i = 1,2,3,\ldots, n)$ represents the contribution of the ith feature vector to construct the observed signal. The goal of the ICA is to estimate the full rank mixing matrix A and the independent components s from the observed original signal x. The ICA component is shown in Fig. 4.

FastICA Algorithm: The summary of the ICA algorithm is presented in this section. Suppose we have a data matrix $X \in R^{n \times P}$ where each column of the data matrix X represents an n-dimensional sample vector. We desire to extract

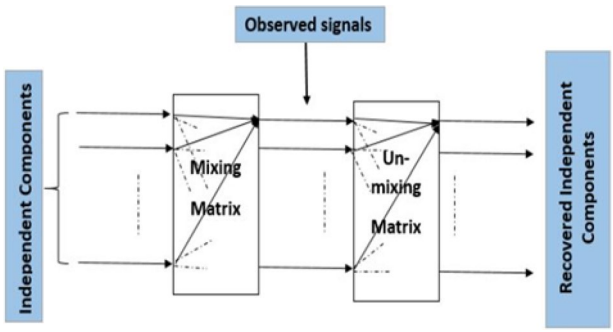

Fig. 4 Overview of the ICA problem. The independent components s and the mixing matrix A both are unknown. The goal is to estimate A and s from the known x

C numbers of independent components, where $C \leq n$. The task is to find the un-mixing matrix $W \in R^{n \times C}$ onto which the observed data matrix X is projected to obtain the independent component matrix $S \in R^{C \times P}$.

Algorithm: FastICA
1: for p = 1 to C do
2: $w_p \leftarrow$ random vector of length n
3: while w_p changes do
4: $w^{p+1} \leftarrow X_g\left(w_P^T X\right) - g^{-}\left(w_P^T X\right) w$
5: $w^{p+1} \leftarrow W_{p+1} - \left(\sum_{j=1}^{p} w_P^T w_j w_j^T\right).^T$
6: $w^{p+1} \leftarrow \frac{w_{p+1}}{\|w_{p+1}\|}$
7: end while
8: return w_p
9: end for
10: return $W = [w_1; w_2; w_3; \ldots\ldots\ldots, w_C]$

2. Auto-Encoder: In the auto-encoder, by applying the non-linear activation function in the encoder and decoder part, it is possible to extract the non-linear hidden pattern of the input space as a code vector. The term code vector is defined as the output of the encoder, it is the compressed or the low dimensional representation of the original data space. In this work, two variations of auto-encoders have been used. They are mentioned below:

- Fundamental auto-encoder with one hidden layer.
- Convolutional auto-encoder with deep convolutional layers.

Fundamental Auto-Encoder (FAE): An auto-encoder is an unsupervised learning algorithm, which takes unlabelled training data as input and re-generates the input data as output data. A two-layer auto-encoder is shown in Fig. 5. An input data (n-dimensional vector), $X = [x_1, x_2, x_3, x_4, x_5, \ldots\ldots, x_n]^T$ is mapping to output layer values, $Y = [y_1, y_2, y_3, y_4, y_5, \ldots\ldots, y_n]^T$ through a hidden layer space representation, where $y \approx x$. An auto-encoder has two main parts, one is the encoder part, which projects the input data to the internal hidden layer space, i.e., which encodes higher dimensional data into lower dimensions. The second part is the decoder, which tries to reconstruct the original dimension from the encoded dimension. The encoder part gives the inherent information of the data which can be very useful for various applications.

The fundamental autoencoder architecture is shown in Fig. 5. All the hidden layer and output layer neurons have the same working principle. If the number of hidden neurons is equal to the number of input neurons, then the network fails to extract the useful hidden pattern of the data points [10]. By assigning the constraints on the number of hidden neurons of the hidden layer, e.g., if the number of hidden neurons is less than the number of inputs. Then it might be possible to discover the hidden useful patterns of the data.

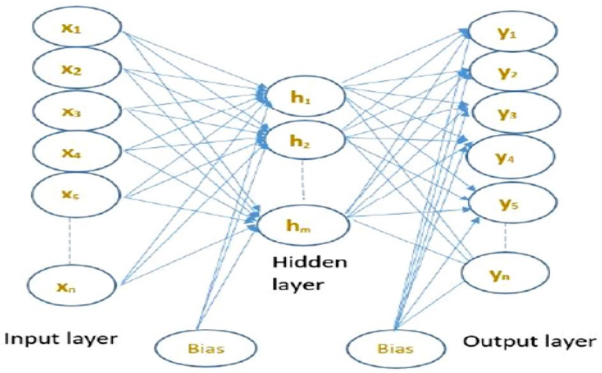

Fig. 5 An architecture of a fundamental auto-encoder with a hidden layer and an output layer

Convolutional Auto-Encoder (CAE): Convolutional auto-encoder (CAE) is very similar to the fundamental auto-encoders except that an input of the CAE is an image. By using Machine Learning, we enable strong issue solutions by optimizing computer resources and implementing ML-based optimizations on the pre-post processing side of things. In CAE, there are two basic types of layers in the architecture of the convolutional layer and pooling layers:

- Convolutional layers
- Max-pooling layers
- Up-pooling layers

From Fig. 6, it is clear that, unlike the fundamental auto-encoder, in CAE the hidden layers will be replaced by the convolution layers and the pooling layers. It enables a huge reduction in the number of parameters (e.g., weights, bias) to be learned for the network, as the convolutional layer's filter dimensions are predefined and independent of the input image size.

4 Result

In this section, first, we provide the experimental setup, followed by experimental outcomes and runtime analysis in Sects. 4.2 and 4.3 accordingly.

4.1 Experimental Setup

The test environment is Ubuntu 16.04.4 with 16 GB of RAM, Intel(R) Core(TM) i7-8086 K CPU @ 4.00 GHz processor with GeForce GTX 1060 graphics card. The

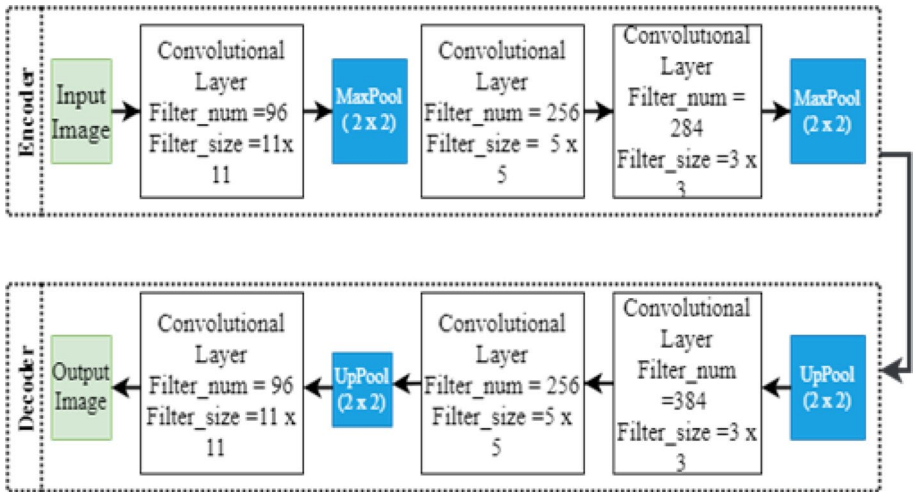

Fig. 6 An architecture of the convolutional auto-encoder. Filter number stands for the number of filters of that associated layer. Filter size refers to the size of the convolutional filter

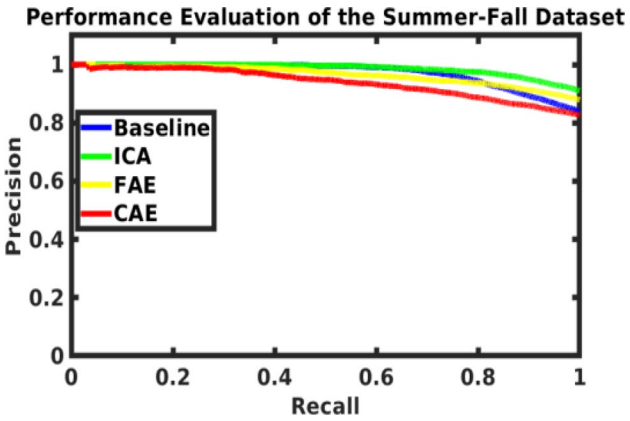

Fig. 7 Precision-recall curves of the baseline and proposed algorithms on the summer-fall dataset

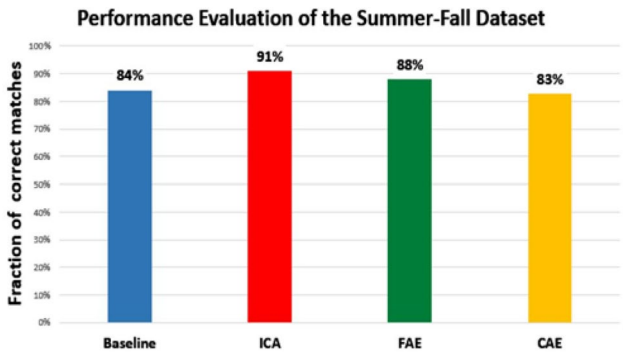

Fig. 8 Fraction of correct matches of the baseline and proposed algorithm on the summer-fall dataset

development environments and programming languages that were used for this purpose include MATLAB (R2018a), Python (version 3.6.1) with TensorFlow 1.0, and Kara's 2.2 frameworks. The programs ran on the system with the GPU2 acceleration mode.

4.2 Experimental Outcomes

4.2.1 Outcomes of the Summer–Fall Dataset

Figure 7 shows the precision-recall curves of the proposed methods and the baseline method on the summer–fall test dataset. It shows that the performance of the ICA algorithm shows better results than the baseline algorithm. The baseline algorithm and the ICA hold the precision rate around 100% until the 67% recall rate, but after that, the precision rate is decaying quickly for the baseline algorithm than the ICA algorithm. ICA has a 91.05% precision rate at 100% recall, whereas the baseline algorithm achieves an 84% precision rate at 100% recall rate. The auto-encoders (fundamental auto-encoder (FAE) and convolutional auto-encoder (CAE)) also follow a trend like ICA. They both drop the 100% precision rate earlier than the baseline and ICA, but the decaying rate of the precision rate is very slow. Even for the FAE, the precision rate gets higher than the baseline method an 82% recall rate, and for CAE, the precision rate is almost the same as the baseline algorithm at a 100% recall rate.

In Fig. 8, the results of the algorithms are shown using a fraction of correct matches as an evaluation metric. It shows that the ICA algorithm performs better than the other three algorithms with a 91% correct loop detection rate. The FAE shows better results compared to the baseline with 88% correct matches. CAE achieves 83% correct matches, which is very close to the baseline algorithm of 84% correct matches.

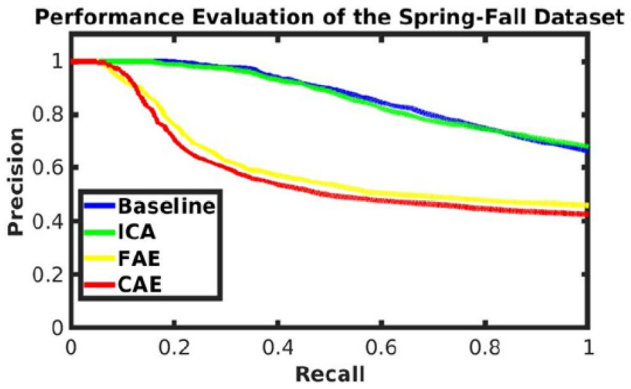

Fig. 9 Precision-recall curves of the baseline and proposed algorithms on the Spring–fall dataset

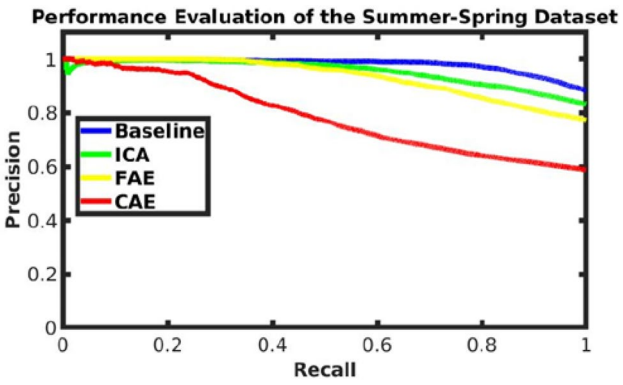

Fig. 11 Precision-recall curves of the baseline and proposed algorithm on the summer-spring dataset

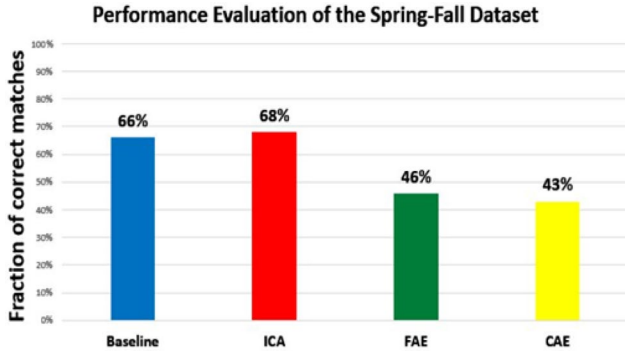

Fig. 10 Fraction of correct matches of the baseline and proposed algorithm on Spring-fall dataset

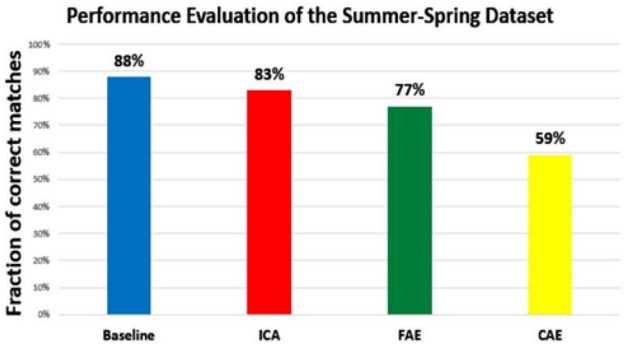

Fig. 12 Fraction of correct matches of the baseline and proposed algorithms on the summer–spring dataset

4.2.2 Outcomes of the Spring–Fall Dataset

Figure 9 shows the precision-recall curves of the baseline and the proposed algorithms on the spring–fall test dataset. This figure indicates that all four algorithms fail to hold 100% precision at a 50% recall rate. The baseline and the ICA algorithm follow the same trend in performance, they hold the 100% precision rate to till around 20% recall rate. After that, the precision rate decays very slowly for both of them throughout the graph. The FAE and the CAE model start to drop the precision rate at the beginning at a 5% recall rate and very quickly. FAE and CAE reach a precision rate from 100% to 60% at only a 32% recall rate. After that, the precision rate is smoothly going down, hence they have below 50% precision rate at a 100% recall rate.

Figure 10 represents the fraction of correct matches of the baseline and proposed algorithms on spring-fall dataset. It illustrates that our proposed ICA algorithm shows a little bit of improvement in detecting the correct loops than the baseline algorithm. The baseline model achieves 66% of correct matches while the ICA model achieves 68%. FAE and CAE also obtain 46% and 43% of correct matches, respectively. The reason for falling the precision rate quickly in the spring–fall dataset can be the perceptual changes between the spring seasons images and the fall seasons images. In the spring season, the sky is less cloudy compared to the fall season. Moreover, nature is greener in the spring season compared to the fall season.

4.2.3 Outcomes of the Summer–Spring Dataset

Figure 11 shows the precision-recall curves of the baseline and the proposed algorithms on the summer–spring test dataset. In this figure, we can see that our proposed algorithms do not outperform the baseline algorithm. We notice a drop in the precision rate in the ICA algorithm at around 1% recall, due to finding the false positive matches. After that, the precision rate gets higher and, reaches 100% precision rate. The baseline, ICA, and FAE models hold around 100% precision rate at 40% recall, after that the precision rate starts to drop for the ICA and FAE compared to the baseline method. The CAE algorithm shows poor precision values compared to the other three algorithms [20]. Its precision rate starts to decay from the beginning, it achieves a

Fig. 13 Few examples of the test images from all three seasons, i.e., summer, fall, and spring. The first row refers to the image sequence of the summer season, the second row refers to the fall season, and the third row the spring season

precision rate of 78% and 58% at the recall rate of 50% and 100%, respectively.

Figure 12 shows the fraction of correct matches of the different algorithms on the summer–spring dataset. It shows that the baseline algorithm detects the correct loops at 88%, whereas the proposed ICA, FAE, and CAE methods detect 83%, 77%, and 59% respectively. The ICA and the FAE algorithm have the second and third-best loop detection rates among the presented algorithms in this work.

4.2.4 Overall Result and Exception Analysis

The performance of all the algorithms for the summer–fall dataset is very good because the true perception changes between summer and fall images are very low which we can see from the test images presented in Fig. 13. In the spring–fall dataset, the precision rate falls very quickly due to large perceptual changes between the spring and fall seasons images. In the spring season, the sky remains less cloudy and nature is greener than in the fall season. As a result, the perceptual change among images is high. That is why all algorithms showed low performance.

The baseline algorithm presents the scene based on a smaller number of the variable of an image and ICA used complete images as the global extractor and image patch as the local feature extractor [27]. Hence, the perceptual changes increases during the ICA algorithm application compare to the baseline algorithm. As a result, the ICA performance becomes lower than the baseline algorithm. In every test dataset, it has been seen that the auto-encoders do not perform up to the mark. As neural network algorithms demand a high volume and high variety of training data. We have the intuition that in the training dataset the variations in the training images are not that much enough so that the network is able to learn the generalized parameters. As a result, the auto-encoders fail to learn the condition-invariant representation of the unseen test data across the seasonal changes.

Table 4 Compilation times of the models

Models	Required times
Baseline	1 h 12 min 5 s
ICA	1 h 57 min 32 s
FAE	3 h 43 min 46 s
CAE	7 h 56 min 9 s

Table 5 Required time to extract the condition-invariant features of a test image

Models	Required times (ms)
Baseline	0.1
ICA	0.3
FAE	3.26
CAE	17.5

4.3 Runtime Analysis

The required time to compile the models and to extract the condition-invariant features of a test image is shown in Tables 4 and 5 respectively. The feature extraction time can be reduced using powerful GPUs. The CAE model takes high time to produce the output, because of the high number of convolutional filters in the encoder and decoder part. As in the ICA model, all the ICs have been used without reducing the dimensions, hence the PCA-based baseline algorithm performs slightly fast compared to the ICA model.

5 Discussion

The aim of this research is to understand the robust representation of the scene. The autonomous robot will be able to recognize the previously visited route though the perceptual changes of the scene go very high [23]. The issue of finding an appropriate route when a robot comes back through its previously visited path is called a loop-closure detection problem. A significant role is played by accurate loop-closure detection which corrects the errors of the map of an environment in an autonomous navigation model. During this research, to understand the condition-invariant features of the scene, we proposed two possible methods called Independent Component Analysis (ICA) and the Convolutional auto-encoder. For finding the image patch or detecting the loop, this model used learned condition invariant features. This work perceives that the proposed algorithms (especially the ICA algorithm) outperform or perform up to a considerable level in the summer, fall, and spring seasons, i.e., where there are no extreme perceptual changes in the appearance of the scenes. The proposed ICA algorithm performs better than the baseline algorithm in the summer–fall dataset with 91% of correct loop-closure detections.

6 Conclusion

Visual place recognition is becoming a trendy research area in computer vision, especially in SLAM. This area has achieved significant progress. Still, it is a very challenging area due to the change in the appearance of a scene in different ways, such as weather, seasons, illumination, etc. Auto-encoders can be handy algorithms to learn the robust features of the locations in challenging environments with the appropriate amount of training data. The minor variation in the training dataset is mentioned as one of the possible reasons for the poor performance of the proposed auto-encoders. The training dataset can be updated by gathering sufficient data covering various environments, including seasons, viewpoint changes, illumination, etc.

Acknowledgements We acknowledge Khulna University of Engineering & Technology and Ahsanullah University of Science and Technology for providing good infrastructure to contribute significantly to this work.

Authors Contributions All authors are contributed equally.

Funding This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Data Availability Statement And this manuscript has no associated data.

Declarations

Conflict of Interest We declare that we do not have any conflict of interest.

References

1. Basri R, Jacobs DW. Recognition using region correspondences. Int J Comput Vision. 1997;25:145–66.
2. Bellekens B, Spruyt V, Berkvens R, Penne R, Weyn M. A benchmark survey of rigid 3d point cloud registration algorithms. Int J Adv Intell Syst. 2015;8:118–27.
3. Bottou L, et al. Stochastic gradient learning in neural networks. Proc Neuro-Nımes. 1991;91:12.
4. Chen Z, Jacobson A, Sunderhauf N, Upcroft B, Liu L, Shen C, Reid I, Milford M. Deep learning features at scale for visual place recognition. In: 2017 IEEE international conference on robotics and automation (ICRA). IEEE; 2017. p. 3223–3230.
5. Chingovska I, Anjos A, Marcel S. On the effectiveness of local binary patterns in face anti-spoofing. In: 2012 BIOSIG-proceedings of the international conference of biometrics special interest group (BIOSIG). IEEE; 2012. p. 1–7.
6. Deb S, Islam SMR, Robaiat Mou J, Islam MT. Design and implementation of low cost ECG monitoring system for the patient using smart device. In: 2017 International conference on electrical, computer and communication engineering (ECCE). IEEE; 2017, February. p. 774–778.
7. Grimson WEL, Lozano-Perez T. Localizing overlapping parts by searching the interpretation tree. IEEE Trans Pattern Anal Mach Intell. 1987;9:469–82.
8. Hasib KM, Habib MA, Towhid NA, Showrov MIH. A novel deep learning based sentiment analysis of twitter data for US airline service. In: 2021 international conference on information and communication technology for sustainable development (ICICT4SD). IEEE; 2021. p. 450–455
9. Hasib KM, Iqbal M, Shah FM, Mahmud JA, Popel MH, Showrov M, Hossain I, Ahmed S, Rahman O. A survey of methods for managing the classification and solution of data imbalance problem. 2020. arXiv preprint arXiv:2012.11870.
10. Hasib KM, Tanzim A, Shin J, Faruk KO, Mahmud JA, Mridha MF. BMNet-5: a novel approach of neural network to classify the genre of Bengali music based on audio features. IEEE Access. 2022;10:108545–63. https://doi.org/10.1109/ACCESS.2022.3213818.
11. Hasib KM, Towhid NA, Islam MR. Hsdlm: a hybrid sampling with deep learning method for imbalanced data classification. Int J Cloud Appl Comput (IJCAC). 2021;11(4):1–13.
12. Islam MT, Islam SMR. A new image quality index and it's application on MRI image. Int J Image Graph Signal Process (IJIGSP). 2021;13(4):14–32.

13. Islam SMR, Islam MT, Huang X. A new approach of image quality index. In: 2017 4th international conference on advances in electrical engineering (ICAEE). IEEE; 2017.
14. Islam MR, Razzak I, Wang X, Tilocca P, Xu G. Ucbvis: understanding customer behavior sequences with visual interactive system. In: 2021 international joint conference on neural networks (IJCNN). IEEE; 2021. p. 1–8.
15. Ivakhnenko A. Cybernetic predicting devices. Technical report.
16. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst. 2012;5:1106–14.
17. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521:436–44.
18. Lowe DG. Three-dimensional object recognition from single twodimensional images. Artif Intell. 1987;31:355–95.
19. Marcel S, Nixon MS, Li SZ. Handbook of biometric anti-spoofing, vol. 1. Berlin: Springer; 2014.
20. Md H, Islam R, Haque MA, Hossain MS, Ul-Haq A, Sawan JJ. Automated face detection, recognition and gender estimation applied to person identification. J Comput Sci. 2019;15:395–415.
21. Noh H, Araujo A, Sim J, Weyand T, Han B. Large-scale image retrieval with attentive deep local features. In: Proceedings of the IEEE international conference on computer vision; 2017. p. 3456–3465.
22. Rumelhart D, Hinton G, Williams R. Learning representations by backpropagating errors, cognitive model. Cogn Model. 1988;5:1.
23. Schmid C, Mohr R. Local grayvalue invariants for image retrieval. IEEE Trans Pattern Anal Mach Intell. 1997;19:530–5.
24. Wang TH, Huang HJ, Lin JT, Hu CW, Zeng KH, Sun M. Omnidirectional cnn for visual place recognition and navigation. In: 2018 IEEE international conference on robotics and automation (ICRA). IEEE; 2018. p. 2341–2348.
25. Waterborg JH. The lowry method for protein quantitation. In: The protein protocols handbook. Berlin: Springer; 2009. p. 7–10.
26. Zarini H, Khalili A, Tabassum H, Rasti M, Saad W. AlexNet classifier and support vector regressor for scheduling and power control in multimedia heterogeneous networks. In: IEEE Transactions on Mobile Computing, 2021.
27. Zhang Z, Deriche R, Faugeras O, Luong QT. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. Artif Intell. 1995;78:87–119.