Heliyon 9 (2023) e13636

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon

Review article

AUTO-HAR: An adaptive human activity recognition framework using an automated CNN architecture design

Walaa N. Ismail [a,b,*], Hessah A. Alsalamah [c,d,**], Mohammad Mehedi Hassan [c], Ebtesam Mohamed [b]

[a] Department of Management Information Systems, College of Business Administration, Al Yamamah University, 11512, Riyadh, Saudi Arabia
[b] Faculty of Computers and Information, Minia University, 61519, Minia, Egypt
[c] Information Systems Department, College of Computer and Information Sciences, King Saud University, 4545, Riyadh, Saudi Arabia
[d] Computer Engineering Department, College of Engineering and Architecturen, Al Yamamah University, 11512, Riyadh, Saudi Arabia

ARTICLE INFO

Keywords:
Human activity recognition
Deep learning
Convolution neural networks
CNN topology
Genetic algorithms
Evolutionary neural network search

ABSTRACT

Convolutional neural networks (CNNs) have demonstrated exceptional results in the analysis of time- series data when used for Human Activity Recognition (HAR). The manual design of such neural architectures is an error-prone and time-consuming process. The search for optimal CNN architectures is considered a revolution in the design of neural networks. By means of Neural Architecture Search (NAS), network architectures can be designed and optimized automatically. Thus, the optimal CNN architecture representation can be found automatically because of its ability to overcome the limitations of human experience and thinking modes. Evolution algorithms, which are derived from evolutionary mechanisms such as natural selection and genetics, have been widely employed to develop and optimize NAS because they can handle a blackbox optimization process for designing appropriate solution representations and search paradigms without explicit mathematical formulations or gradient information. The Genetic optimization algorithm (GA) is widely used to find optimal or near-optimal solutions for difficult problems. Considering these characteristics, an efficient human activity recognition architecture (AUTO-HAR) is presented in this study. Using the evolutionary GA to select the optimal CNN architecture, the current study proposes a novel encoding schema structure and a novel search space with a much broader range of operations to effectively search for the best architectures for HAR tasks. In addition, the proposed search space provides a reasonable degree of depth because it does not limit the maximum length of the devised task architecture. To test the effectiveness of the proposed framework for HAR tasks, three datasets were utilized: UCI-HAR, Opportunity, and DAPHNET. Based on the results of this study, it has been found that the proposed method can efficiently recognize human activity with an average accuracy of 98.5% (∓1.1), 98.3%, and 99.14% (∓0.8) for UCI-HAR, Opportunity, and DAPHNET, respectively.

* Corresponding author at: Department of Management Information Systems, College of Business Administration, Al Yamamah University, 11512, Riyadh, Saudi Arabia.
** Corresponding author at: Information Systems Department, College of Computer and Information Sciences, King Saud University, 4545, Riyadh, Saudi Arabia.
E-mail address: walaa.ismail@mu.edu.eg (W.N. Ismail).

https://doi.org/10.1016/j.heliyon.2023.e13636
Received 1 June 2022; Received in revised form 4 December 2022; Accepted 6 February 2023

Available online 13 February 2023

0. Introduction

Human activity recognition (HAR) involves analyzing video or sensor data to determine human behavior [1,2]. It is becoming increasingly important to analyze human behavior, human-computer interaction, and human behavioral monitoring systems. Several human behaviors, including walking, sitting, riding a bike, jogging, eating, reading, and washing, are considered static or dynamic [3]. Training data for HAR are primarily derived from nonvisual wearable sensors and ambient assisted cameras (visual sensors) and a combination of both [4,5].

Over the last few years, deep learning (DL) has been widely used to aid in the analysis of sensor-collected data and recognition of patterns. Since the introduction of deep learning in recent years, convolutional neural networks (CNNs) have gained considerable attention in many computer vision applications owing to their established theory and high performance in modeling and classifying collected data [2,6]. Recommendation systems, medical image analysis, and natural language processing are areas in which CNNs are utilized [7–9]. CNNs have been demonstrated to produce outstanding results in time-series data analysis [10,11]. However, it is generally accepted that deep learning approaches, especially CNNs, are characterized by a large number of trainable parameters, a large amount of training data, extensive parameter tuning, and a high level of resource exhaustion during training and inference [12]. In addition, practitioners of deep learning encounter difficulty when it comes to manually building deep models and determining suitable configurations (e.g., model layers and operation types) through trial and error. Various steps are involved in feeding domain knowledge into DL, including Feature Engineering (FE) [13], model generation [14], and model deployment [15,16]. Because CNNs are based on layers, they allow the flexibility of adding or removing layers based on the training phase, which is then used in inference (classification) to classify the data. It may not be necessary to use the entire architecture from the training phase in the inference phase, as most of the data can be correctly classified using only the first few layers [17,18]. Consequently, if classification can be performed using only an optimum portion of the network at a time, we can avoid redundant operations in the CNN, leading to the enhancement of the classification/prediction abilities for a given application. Additionally, the challenge is to define the set of constraints of CNN layers that will result in an effective and reliable performance. The set of constraints may include the time limit, number of inputs, model depth, maximum number of neurons, and number of filters. Owing to the difficulty of acquiring expert knowledge to define all such constraints, the development of a CNN is challenging, and its configuration can greatly influence the classification/prediction efficiency and performance [9,6,12].

The Neural Architecture Search (NAS) technique, which automates the design of deep learning networks, has been shown to perform equally well or better than manually designed architectures in a variety of computer vision tasks compared to conventional deep learning techniques that rely heavily on expert or domain knowledge [19,59]. Therefore, several different search strategies have been proposed, including evolutionary deep learning (EDL) [12], random search methods [20], gradient-based methods [21], and Bayesian optimization methods [22] Another factor to consider in architectural search is the scope of the search space. Two search spaces are available: the entire architecture for a given set of inputs and outputs [20,23], or the inner structure (operations and connections) of a fixed macro architecture [24,25], which is not optimized during the architecture search and functions as an exoskeleton. Although these innovative search spaces display a combination of innovation and efficiency, they are difficult to comprehend, design, and train. However, the models can be designed structurally efficiently [12,26] and their size reduced by removing unnecessary inner structure parameters. Process optimization is largely driven by evolutionary mechanisms. The family of Evolutionary Algorithms (EAs) is a robust and flexible heuristic that may be considered when there is a problem that is difficult to solve but for which evaluating solutions is easy [27,17]. EDL may therefore be used to identify the most accurate and efficient configurations of deep models through evolutionary optimization. Human activity recognition is achieved through EDL by automatically designing a deep model through an evolutionary process [23,28] to provide an easy-to-use human activity learning tool. Therefore, the model configuration is viewed as an individual, and the goal of the model is viewed as the performance of the model. Based on the analysis above, EDL tries to improve the adaptability of a deep model to learning tasks by automating construction (from a DL viewpoint) and striving to generate the best model possible within the set restrictions or objectives (from an evolutionary computation viewpoint). Mainly, our research aims to develop and evaluate an Automated CNN (AUTO-CNN) evolutionary model for the design of a CNN-based architecture for human activity recognition purposes by addressing the previously discussed limitations. AUTO-CNN applies a Genetic Algorithm (GA) to identify the performant topology constraints for CNN-based classification algorithms; however, it is easy to evaluate its quality (though perhaps computationally challenging). There is no requirement for users to have any previous knowledge of CNN design and dataset parameter selection before they can generate many different architectures, analyze their effectiveness, and select the best-fitting architecture. The proposed algorithm can be used to directly analyze any activity without recomposition, preprocessing, or postprocessing if new activities are to be examined. Additionally, AUTO-CNN will use an efficient layer-based CNN architecture through the proposed novel variable-length encoding scheme and incorporate a crossover operator and a mutation operator integrated into a GA to search for the optimal depth of the CNN. The proposed algorithm achieves impressive performance, yet users are not expected to be experts in these design building blocks to use it. The contribution can be summarized as follows:

1. A new framework, Automatic HAR (AutoHAR), is proposed for human activity recognition from streaming movies. AutoHAR will evaluate the performance of Evolutionary Algorithms (EA) in the context of CNN architecture search to recognize human activity.
2. A new algorithm, AUTO-CNN, is developed and evaluated to choose the best CNN architecture for human activity recognition tasks through an effective variable-length encoding strategy. In the AUTO-CNN algorithm, the optimal CNN architecture is established with sufficient depth, without defining the maximum length of the architecture.

3. A series of experiments are conducted to assess the efficiency and accuracy of the proposed method to automatically design CNN architectures compared to current state-of-the-art CNN-based approaches. Three datasets are used: the first dataset is obtained from UCI; the second dataset is formed from the opportunity dataset, and the third dataset is the DAPHNET dataset. According to the results of the experiments, the proposed approach is correct and effective. Furthermore, the method has a significantly higher classification accuracy than state-of-the-art, CNN-based models that are built manually, non-evolutionary, or evolutionary-based models.

Other sections of the paper are organized as follows: Background information is presented in section 1. Documentation of the proposed algorithm can be found in section 2. The experimental environment design and numerical results are presented in Section 3 and Section 4. An overview of the conclusions reached and the work to be undertaken in the future can be found in 5.

1. Related work

Researchers have been experimenting with DL based methods to develop tools for detecting normal or abnormal human movements using sensor-collected data or images [41]. The literature varies in its categorization of human actions based on whether they are produced by handwork (conventional-based) [29] or algorithmic recognition (artificial intelligence) using Deep neural networks (DNNs) [18] or other techniques [30]. DNNs and their main types have received a vast amount of effort for computer vision problems. In [31], an automatic lab annotation generation algorithm based on deep learning (DL) was utilized using a CNN with an average recognition rate of 95.58%. The framework starts its work by modeling the characteristics of human body motion. Subsequently, an efficient DL algorithm that can automatically generate the dance spectrum platform is proposed. In [32], the application area was changed to recognize the human actions from the collected data using the promising performance of the CNN network. The authors trained a 3D convolutional neural network to train the model over frames of data. Wang et al. [33] utilized a spatiotemporal CNN to segment the human-collected videos into subactivities.

In [34], the authors proposed a hybrid model using a DNN for the recognition and prediction of human actions. Their model consists of four pretrained hybrid layers: AlexNet [35], VGG-16 [35], GoogleNet [36], and ResNet [37]. Additionally, the GA is utilized to approximate optimum feature selection. Their model achieved approximately 98.5%. Using dynamic Bayesian networks with domain knowledge, Zeng and Ji [22] proposed solutions for human action recognition. This approach presents generic Bayesian dynamic network models for human activity classification and models that incorporate multiple features. In this context, the authors propose a framework for learning domain knowledge using Deep Belief Networks (DBNs). With the proposed framework, the authors claim to not have to consider insufficient training data for human action recognition. Despite the great performance achieved through deep learning models, the architecture of such models still greatly affects the performance and final prediction accuracy. Many attempts have been made to enhance the architecture design as follows: the weights of the architecture initialization and selection, such as greedy layerwise [38] and fast learning algorithms and neural network training [39–41]. Moreover, different techniques have been conducted to improve architecture generalization [42,43]. Leong et al. [44] employed a fusion of ResNets [70], DenseNets [45], and VGGNets [46] with skip connections between individuals to learn the spatio-temporal features more efficiently for video action recognition. An important goal is to develop an early stopping to avoid deep neural network overfitting or adding randomly dropping neural units during training [47,48]. Additionally, unsupervised pretraining was employed to initialize the weights in deep neural networks [49]. Recently, researchers have conducted a variety of methods to initialize a well-designed architecture, including randomly chosen traits, grid search, and manual search [48,49]. Each method can effectively confirm a search, and certain methods provide a less/more promising configuration space for hyperparameter optimization [48,49]. In the current work, we address these challenges through the use of evolutionary algorithms GAs for the selection of the best convolution neural network structure. Researchers have used the GAs to enhance neural network performance from different perspectives, including feature selection optimization [50,51], topology selection [4,52], or parameter weight selection [53,26].

GA are based on the idea of multidimensional convolution. The initial value and threshold of block-based neural networks are adjusted and optimized using state-of-the-art architecture to arrive at the optimal solution and to reduce the redundancy rate in behavior recognition [54,26]. The authors in [54] employed a GAs to train their network by optimizing the learning weight of the autoencoder. By combining genetic algorithms with a multidimensional convolution method, the authors in [54] proposed a model that can more accurately and quickly identify and predict human behavior, with calculations that are basically consistent with the actual requirements. The proposed model starts its work by clustering the redundant data by a genetic algorithm to produce fragments of data. Afterward, the genetic algorithm clusters the data to form subgenetic particles with different dimensions and performs conevolution and optimal location sharing. Researchers have emphasized the importance of Neural Architecture Search (NAS) methods, which will be used to reduce the search space and automate the design of CNN architectures by transforming it into a complex optimization problem [55,56].

A fog-assisted cloud computing architecture proposed in [4] that is able to recognize human activity from cloud data. A hybrid algorithm is used to solve the activity recognition problem by combining the benefits of YOLOv3 [57] and GA. YOLOv3 is used to detect the changes in the frames using an accelerated GAs paradigm on the generated subblocks. To ensure the highest degree of accuracy, GAs parameters are randomly selected with the aim of reducing the response time and accuracy.

In [20], a chain neural network structure is used to identify the block that constructs the architecture in sequence, which optimizes the architecture parameters, sequence of construction, and depth of the network model. Additionally, to reduce the amount of computational resources required to train the architecture generations, a random forest-based performance predictor is utilized.

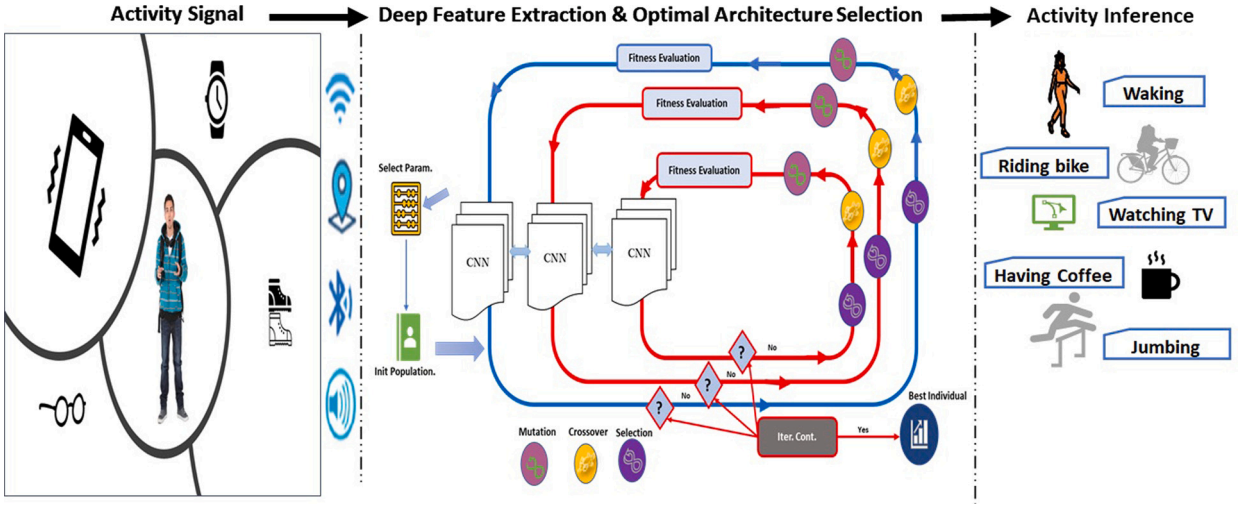

Fig. 1. The Proposed AUTO-HAR Deep Learning Architecture.

Loni et al. [53] reduced the search space by limiting the range of convolutional hyperparameters (such as the number and size of convolutional layers, the filter size, and the activation function) and by using a multiobjective search with NSGA-II to balance network accuracy with network size.

In a study by Zhang et al. [58], nine different types of interactions among nodes were analyzed to identify connections and computations. When the architecture is generated, the efficient building units of the architecture can ensure its efficacy, so the algorithm can identify an architecture with good performance.

Parkr et al. [59] introduced Once-For-All (OFA), a NAS technique in which GAs are used in order to find the optimal CNN architecture. The network is designed to consider diverse layers (i.e., depth), input image channels and kernel sizes in multiple units.

In Baldominos et al. [60] an activity recognition task is conducted by proposing a DL CNN. By using a grammatical evolution method, a topology is calculated based on an evolutionary algorithm. Based on the GE's definition of a grammar in normal form, each network design is derived. An efficient variable-length, genetic coding strategy was developed in AE-CNN [23] and CNN-GA [28] to adaptively search for the optimal depth of CNNs. The above models utilized search methods using a block-based search algorithms that enable skip connections for training deep architectures.

In this work, inspired by the great ability of the GAs in current research environments to simultaneously optimize both weights and topology, we propose a new approach for selecting the best topology of a convolutional neural network classifier by exploiting the efficient global and local search abilities of the GAs for the prediction of human actions. The AUTO-CNN utilized in this work selects and optimizes a very broad number of features for the estimation and selection of an appropriate CNN architectural design. These features, including the number of output channels, CNN kernel size, model building components, activation function selection, fully connected layer parameters, and dropout rate, are incorporated into state-of-the art models. The novelty of our work is therefore in modeling convolutional neural network topologies as a GA chromosome to expedite the search for finding the optimum parameters in a given search space, which will help overcome the limitation of fixed-based classifier topologies. Table 1 compares the most recent research on NAS. Based on this review, the following can be inferred as the primary limitation of the present literature.

- Most investigations have concentrated on a rigid encoding strategy for the devised CNN architecture, which has a limited architectural search space. In this study, a novel variable-length encoding schema for a CNN was devised.
- Studies have focused on the use of graphics processing units (GPUs), which can be expensive to implement on devices with limited resources. However, the devised method is very light and can be implemented in a limited-resource environment.
- Numerous studies have focused on improving a small number of model parameters; however, we addressed this problem by focusing on all parameters.
- Earlier research focused on the application of enhancement processes to image datasets to obtain the desired results. However, in this study, we used acceleration data related to human activity detection.

2. Proposed framework

In this work, we present an automatic architecture, AUTO-HAR, for human action recognition by designing an optimized CNN architecture. The proposed architecture is illustrated in Fig. 1. The framework consists of four main phases. The process starts with parameter identification, design of the CNN architecture, model training and optimization and a final evaluation phase. The framework begins its work by preprocessing video-based or sensor-collected data. The user should select the parameters pertaining to the search space, including the training parameters and termination criteria. As the performance of a trained classifier depends heavily on its initial parameter weights, in the second section, the design of the convolution and fully connected layers of the CNN

Table 1
Overview of the Neural Architecture Search State-of-the art related works. ISAR: the space target inverse synthetic aperture radar, PSO: Particle Swarm Optimization, AMBO: Adaptive Monarch Butterfly Optimization.

Ref	Task	Model	Interpretations	Limitations
[52]	Human Activity Recognition	A Search strategy for an architecture combining CNNs and LSTM networks using the non-dominated sorting genetic algorithm (NSGA-II).	A lightweight and fast algorithm has been used to eliminate the need for manual effort in the network architecture search process.	There is an increase in complexity of the model as well as a rise in memory access costs as the number of channels increases.
[20]	Images object recognition	Automated design strategy for CNN architectures based on an evolutionary algorithm while using a random forest-based performance predictor, with a triplet attention mechanism.	By incorporating a random forest-based evaluation into the fitness function, the classification performance of the devised architecture is significantly improved.	The encoding strategy used for the devised CNN architecture has reduced the search space of the architecture, which prevents the structure from expressing the diversity of the assigned tasks.
[15]	Human Activity Recognition	A simple grid search algorithm has been explored how to optimize the models' hyper-parameters.	To achieve a good trade-off between classification score and memory occupancy, a combination of sub-byte and mixed precision quantization was used.	Applying this strategy to CNNs with several layers (more than 10) would need the use of a NAS tool, which is not feasible.
[61]	Human Activity Recognition	A PSO based CNN architecture search methodology.	The use of global and local exploration capabilities of PSO and gradient descent back-propogation has improved the classification accuracy efficiently.	The algorithm fails to recognize various activities, such as WALKING_UPSTAIRS, WALKING, LAYING, and SITTING, due to an insufficient amount of training data.
[13]	Face Recognition	An evolutionary discriminant feature extraction algorithm based on a combination of GA and subspace analysis over a static-based CNN.	The use of the Convolution search paradigm results in a reduction in the complexity of the search space and an improved classification method.	There is a greater difficulty in developing an individually optimized architecture due to the static CNN configuration.
[62]	ISAR image recognition	The generation of incremental architecture using Random Forest Learning (RL) based methods.	The use of Random Forest- based Learning framework makes the proposed model applicable for effective diagnosis of HAR.	Considering the large state/action space involved, substantial computational resources are required. Furthermore, there are many hyperparameters associated with each RL-based NAS.
[21]	Remote Sensing Image.	The generation of neural architecture using gradient-based algorithms.	By treating NAS problems as continuously differentiable problems and not discrete, the model is able to efficiently search architectures with appropriate weight parameters.	The gradient-based algorithms require the update of a large number of parameters, which will significantly increase the GPU costs.
[14]	Human Activity Recognition	Three optimization search algorithms are used: PSO, Greedy,and Genetic algorithms with KNN classifier.	The model achieved excellent classification results based on smartphone embedded sensors to recognize six basic human activities.	This process involves pruning a portion of a large network randomly, retraining it, and repeating the process until it reaches its optimal performance.
[63]	Human Activity Recognition	AMBO is employed to enhance the performance of traditional deep learning algorithms.	The AMBO algorithm significantly improves recognition accuracy by providing the best error rate and the highest level of performance accuracy.	Optimization algorithms are used to determine only the optimal hyperparameters weights for the predefined network structure.
[16]	Human Activity Recognition	An adaptive CNN architecture that uses an output block predictor to select portions of the baseline architecture based on statistical features.	A novel adaptive architecture that selects a portion of the baseline architecture for inference based on an output block predictor	When the earlier layer misclassifies a segment with a higher degree of certainty than the baseline design, such systems may suffer from performance loss. In addition,it is important to ensure that less frequent knowledge is not forgotten and those early classifiers are correctly initialized.

is developed. Due to the limited computational resources, preliminary experiments revealed that a CNN-Convolutional (CN) Layer with 4 layers followed by a dropout layer and Fully Connected (FC) layer could produce a better overall sensitivity. We used two FC layers and a softmax layer with random dropout. The details of these CNN blocks are described in Subsection 2.2. Note that it may be possible for the classification results to be obtained by millions of CNN parameter constraints for the collected set of data in each application. Therefore, selecting the best topological constraints for the CNN following static or predefined values may yield a solution that is stuck in a local minimum. Therefore, the third step involves estimation and selection of an appropriate CNN architecture, including *the output channels, CNN kernel size, block number, activation functions, FC layer features*, and *dropout rate* (more details will be given in Subsection 2.3). In this phase, the effective search capability implementations of genetic algorithms were utilized in the human action recognition processes. The GA population of the proposed algorithm will be the CNN structure, followed by the fitness evaluation step, in which the chromosome is decoded to initialize the topology of a CNN classifier. The selected structure is tested and evaluated. In the upcoming sections, we will explain each part of the proposed model in detail.

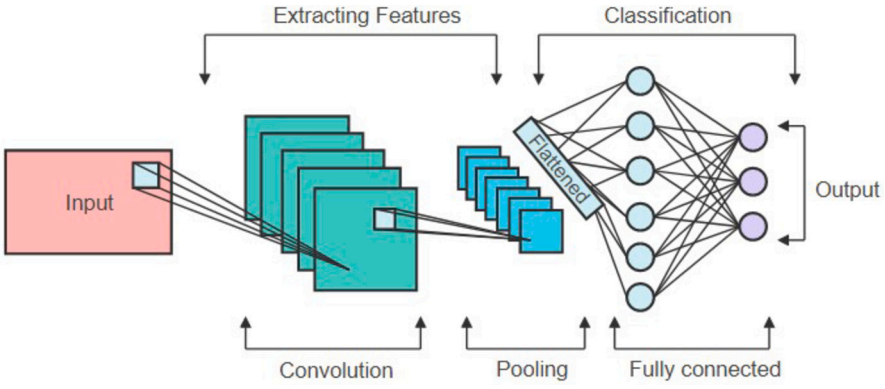

Fig. 2. Basic CNN Architecture.

2.1. CNN classifier for human action recognition

CNNs are a class of Artificial Neural Networks (ANNs) that were developed in the 1980s to address visual pattern recognition tasks; they were modeled on the visual cortex of vertebrates [64]. The CNN structure (shown in Fig. 2) can be considered a two-dimensional matrix of pixels represented as a set of L - main blocks located between convolutional part Cn and classification part Cl and as set of their P-parameters, $S(Cn(L_{Cn}, C_{Cn}, P_{Cn}), Cl(L_{Cl}, C_{Cl}, P_{Cl}))$. The model structures for these double parts are described using Keras neural network libraries [65]. The CNN structure employed in the current study follows the same basic architecture S with the search space of the main blocks for the convolution part Cn, including combinations of Conv1D, Pool, AveragePoooling2D, and ReLU layers with different activations and counts of filters. A convolution layer Cl defines a function $f : Ra \rightarrow Rb$ to map the given input matrix to K "kernels" or "filters" [64], where $0 < k < kb - 1$ with dimensions $a \times h' \times w'$, with $h' \leqslant h$ and $w' \leqslant w$ (where $h' = w$ as an integer odd positive number). Furthermore, convolution layers can be characterized by a number of parameters, such as "stride" and "dilation"; we use a stride of 1 and a dilation of 0 in our implementation. A dense layer with different activations and numbers of neuron combinations forms the main block search space for the classification part (Cl). We initialized all activation functions in Keras while employing the same set of possible connection layers for both Cl and Cn with dropout layers, which means that no additional layers between the main building blocks are needed. For convolution layers, kernel sizes are set to 3×3, and strides are set to 1×1. To avoid overfitting, the CNN classifier is trained using a backpropagation algorithm for a relatively small number of epochs ($p1$).

The CNN structure can be represented by several neurons represented numerous layers. Therefore, encoding the best structure CNN should define a number of T constraints to pertain to an optimal classification result as this condition is satisfied. Otherwise, a CNN's architecture may be arbitrary and could adversely affect classification performance. The set of constraints may include the time limit, number of inputs that it must have to attach to a neuron, model depth, maximal number of neurons, and number of filters. In the current study, the CNN structure identification, its encoding, training, and optimization are demonstrated in Fig. 3. Mathematical-based model optimization can be conducted; however, such models are restricted and do not allow a complete examination of the selected structures included in the model search space. EAs, as metaheuristic methods, are based on smart-exploration techniques, which can handle a wider range of constraints [19]. Furthermore, EAs facilitate representation of the solution.

2.2. AUTO-CNN search space

A search space is a collection of all possible solutions that can be searched for a specific problem. Several hyperparameters, such as feature maps and convolutions size, must be defined in each CNN architecture design. Having a relatively small search space may reduce the time that is required for the algorithm to construct a satisfactory CNN model, but the limited number of models within this space may cause the algorithm to become overly constrained. In addition to being more time-consuming, a larger search space requires more computational power. The majority of the previous works do not include the number of blocks (depth of network) in their search space. A part of the search space in our experiments was:

- The number of input feature maps in each block.
- Activation function type.
- Pooling type.
- The number of output feature maps in each CN.
- The inclusion of batch normalization layer.
- Layer number.

The default search space for our experiments is described in Table 2. Hyperparameters are listed in the left column, and their possible values are listed in the right column.

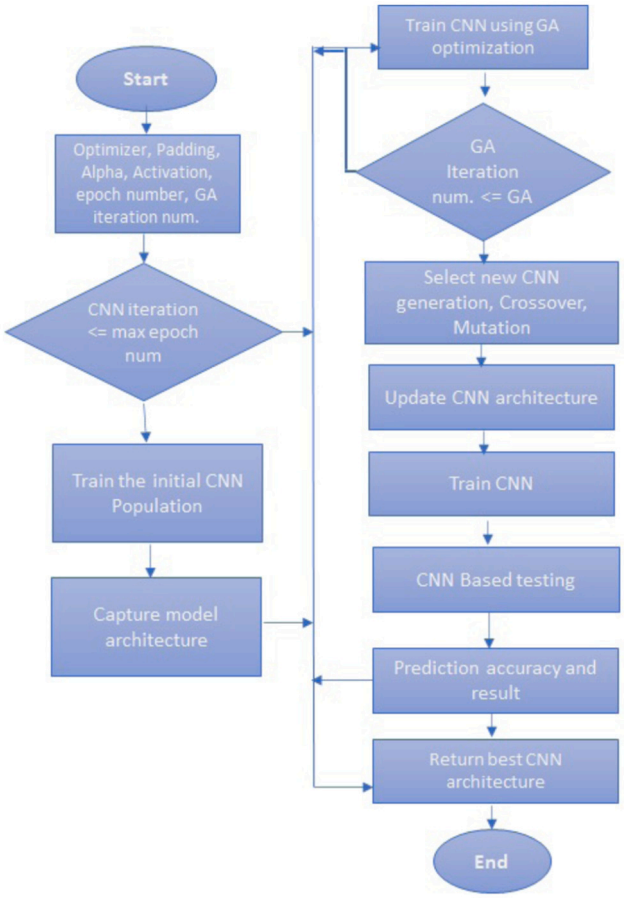

Fig. 3. AUTO-CNN architecture design and optimization.

Table 2
Search space parameters and their possible values.

Parameters	Values
nb_filters	[32, 64, 128, 256, 512]
nb_blocks	[3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
Activation	['relu', 'elu', 'tanh', 'sigmoid']
Optimizer	['adam', 'adamax', 'nadam']
Padding	['same', 'valid']
Alpha	[0.01 - 0.5]

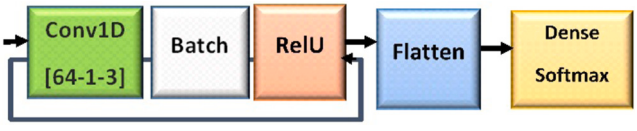

Fig. 4. Addition of a basic block to a fixed FC part.

The convolution part Cn is based on a block named **CBL**. The CBL block includes **three layers**: Conv1D (conv), BatchNormalization (batch) and LeakyReLU(*ReLU*) layers. The fully connected part FC includes a flatten layer and a dense layer with a softmax activation function.

GAs use encoding to model real-world problems to be directly solved by GAs. The first step in employing GAs is to figure out what encoding strategy to use. The selected encoding strategy is based on the problem itself. This approach seeks to develop an effective way to model CNN with different architectures by devising a new encoding strategy.

There is a strong correlation between the performance of a CNN and its depth; therefore, the CBL block is automatically repeated using GAs, and then the fixed FC part is added to the best-selected blocks, as illustrated in Fig. 4. In the selected CBL basic block, the configuration of state-of-the-art CNNs, number of filters, padding and activation function are set according to Table 2. The kernel

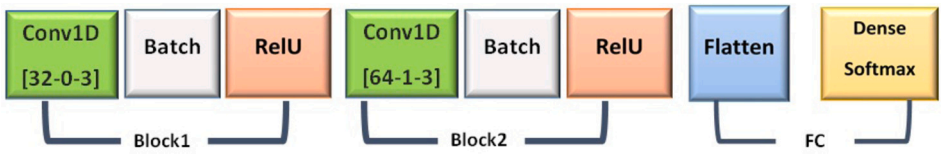

Fig. 5. Example of encoding CNN architecture with a depth of two.

Table 3
AUTO-CNN Algorithm Symbols and Their Descriptions.

Name	Purpose
Block	CNN Subnetwork Topology.
Gene	Specifies the convolution and fully connected part constraints, including (similar to the kernel size of a convolutional block).
Genomes	Sequence of gene parameters (CNN topology description) that can be converted to a new individual following the "crossover" value. To avoid the computation complexity of such a calculation, these values are cached for each genome. Therefore, if the same topology reappears in the current or next generation, we do not have to recalculate it.
Mutation	Changing the parameters of a current CNN topology to a new genome.
Population	Collection of CNNs of genomes.

size of conv is 3 x 3, and the stride is set as the step size to 1x1. In addition, the ReLU layers used in the proposed encoding strategy alpha parameter are set to random numbers between [0.01, 0.5]. The parameter encoded in a batch layer is only 0 or 1 (exist or do not exist). Fig. 5 illustrates how the proposed encoding strategy is used to encode a CNN. There are two CBL blocks in this CNN, one flat layer and one dense layer. An entire CNN can be represented as a string of sequentially connected codes. For the CNN shown in this example, where the layers' codes are listed above each other, the total CNN code is "32 - 0 - 3 - 1 - 0.1 - 64 - 1 - 3 - 1 - 0.15" to represent a CNN with two blocks in addition to the FC fixed blocks.

2.3. Automated CNN architecture selection

Algorithm 1 illustrates the scheme for searching the best CNN architectures using the GAs with the symbols mentioned in Table 3. Genetic algorithms are heuristic search methods that are based on evolutionary principles proposed by Darwin [66]. The evolutionary ideas are based on natural selection and genetics considering a population's survival of the fittest (survival of the fittest). A candidate solution consists of GAs chromosome selection, crossover, and mutation that occur as a result of selection. Changing the chromosomes to create a modified set at the end of the process. At the end of every GA cycle. During each GA cycle, the GAs chromosome that captures the system characteristics and a quality measure of the solution (fitness function) is calculated to be improved successively over multiple generations of chromosomes. A curious user may refer to [67] for more details. In the current study, the algorithm will search the dataset after defining the search space and loading the layers pool. At the first iteration, for the selected population, the CNN structure will be randomly generated. Additionally, any pretrained architecture can be initialized to improve the convergence of the search process. Subsequently, the main parameters of the GA algorithm, including crossover and mutation operators, are used to refine the defined architecture. Crossover is used to combine perspective parts by permuting between two solutions, while mutation is used to explore a new direction of the search space. In the testing process, every individual represents a CNN architecture, which is then transformed into the Keras framework. The fitness function of the evolutionary algorithms is applied to describe how well an individual adapts to new conditions related to the specified problem in question. Using the predicted values of a test sample, a new CNN model is fit to a new training sample and evaluated using selected metrics. It is more likely that the offspring of a person who has adapted well to his environment will also be able to thrive in it. The GAs require a method for generating a solution without the need for gradient evaluation as part of its search procedure. A problem relating to the design of neural networks specifically requires this advantage. A parallel algorithm can also be utilized to accelerate the evaluation of solutions in a population. The mutation and crossover operators are used in the CNN structure design to randomly generate the convolution or fully connected part of the network; therefore, the mutation process will replace the network parts with the new part.

The searched space is denoted as G by training the model as outlined in Algorithm 1, and the first phase will train G for several epochs. For each training iteration, AUTO-CNN will generate a random CNN architecture (Line 2-3, Algorithm 1). Starting from the first generation of CNN individuals to generation 100 (note: the maximum number of generations is 100). The algorithm then checks whether the number of the current generation reaches a predetermined number of generations or whether we are still in the initial generations (Line 4-7, Algorithm 1), such as checking whether the current generation exceeds the maximum number. (Line 8-17, Algorithm 1), the Make POPULATION function is called and returns a two-dimensional array with a size equal to the number of initial populations. A candidate architecture will be sampled, and a set of linear feature outputs will be selected for each dataset as the initial population. The remaining topological selection parameters, including the input channels, fully connected layer, activation functions, kernel sizes, and dropout rates, are randomly selected from [32; 64; 128; 256], [tanh; relu; leakyRelu], [3; 5; 7], [0:1; 0:2;

Algorithm 1 An algorithm to search for the optimal the CNN architecture.

Ensure: gene = 0, stage = 0; Input shape, CNN-shapes
Ensure: I_In- > input_shapes
Ensure: L_out -> layer output shapes ▷ for each upcoming Gene
1: $G \leftarrow gene$
2: **while** ($G \leq 100$) **do**
3: $train(generator)$
4: **if** ($G\% \leftarrow 0$) **then**
5: Save the best architecture
6: $stage \leftarrow stages + 1$
7: **end if**
8: **if** ($FLAG$ is True) **then**
9: $POPULATION = new(POPULATION)$ ▷ by calling Function: Make POPULATION
10: $FLAG \leftarrow False$
11: **end if**
12: **for** (index I in CNN-shapes indices) **do**
13: **if** ($index = 1$) **then**
14: $CNN - shapes \leftarrow I\_In[I]$
15: **else** $I\_in \leftarrow L\_out[I]$
16: **end if**
17: **end for**
18: Define Block_val ← the convolution layer's specific parameters
19: Block.val [I] ← I_in
20: Block[i] ← Block_val
21: **for** I **do**
22: Create new block[i]
23: block[i].output_indices ← out_indices
24: block[i].out_feature depth ← out_feature_depth
25: block[i].Tail ← a layer that pools global averages followed by fully connected layers
26: Initialize the weights of the fully-connected layer
27: $Iters \leftarrow Iters + 1$
28: **end for**
29: **end while**

0:3; 0:5]., respectively. After completing training iterations, the best architecture design (with the highest classification results) will be selected (Line 18-20, Algorithm 1). A new architectural design search of the next stage will be started (Line 21-28, Algorithm 1). Note that a new architecture CNN will not be reinitialized until the training of the current generation is completed.

Algorithm 2 Function: Make POPULATION.

Ensure: $In\_shape, Out\_shape, trainingset, validationset, modelhyperparameters$
Ensure: $iters \leftarrow 0$
Ensure: $index \leftarrow 0$
Ensure: $Flag \leftarrow False$
Ensure: $genomes$ => $array$
1: **for** (each $genome$ in $Genomes$) **do**
2: $genome$ = new Genomes [$in\_shape, out\_shape, hyper\_min, hyper\_max, n\_genes$]
3: Evaluate the population fitness function using ($MEAN(losses)$, $STANDARD - DEVI(losses)$, $n\_parameters$)
4: **if** ($Genomes[i]$ is the best) **then**
5: Flag ← True
6: **end if**
7: **end for**
8: **return** ($Flag, Genomes[i], trainingset, validationset, hyperparameters$)

Evaluate all current generation members (CNN model under this setting) against the given fitness function (Lines 1-3, Algorithm 2), with the cross-entropy loss with a lower value indicating higher fitness. Depending on which criteria are met (i.e., the training loss mean and standard deviation drop below a threshold, a dynamic FLAG is set to True, and the AUTO-CNN training process is immediately terminated. Otherwise, the fittest individuals of the previous generation are selected to form a new population by applying genetic operators. If the given prediction accuracy (called a fit threshold as defined by the lack of improvement in fitness over a certain number of generations) of the current population is the best, then terminate and return the current architecture (Lines 4-6, Algorithm 2). Alternatively, if the conditions are not met, breed a new generation of CNN architecture from the current generation and the newly bred generation and repeat step 2.

3. The experimental setting

The purpose of this study is to find the optimal CNN architecture in the designed space automatically for the task of human activity recognition. It is necessary to maintain a good classification accuracy while using a full automated architecture design. A number of experiments are designed and compared against state-of-the-art peer competitors in order to quantify the performance of

the suggested method using three datasets. In the upcoming section we will go through each dataset and the overall experimental analysis.

The parameter values of Auto-CNN are based on empirical conventions to ensure the effectiveness of the construction blocks and training. The population size in evolutionary progress is set to 100, the number of evolutionary generations is set to 100, mutation probability is 0.003, and The initial generation's minimum number of Genes in each randomly generated Genome are set to 10:15. Mean and Standard deviation threshold is 0.2, and 0.02 respectively. In this study, experiments were conducted using the Google CoLab platform and the Intel Core i7 9900 kf processor. The software programming environment is Python 3:7, the framework is TensorFlow.

3.1. Our peer competitors

To illustrate the superiority of the proposed AUTO-CNN, we compare its performance to different peer competitors that have been incorporated and implemented with the standard mentioned public datasets. The peer competitors were chosen from different categories. The first covers hand-crafted architectures that have been widely adopted in HAR with extensive domain expertise *(Manual)*, including: AlexNet [68], GoogleNet [69], ResNet-152 [70] and VGGNet [46].

The second set represents the ensemble-based classifiers' architecture design that achieve high performance (accuracy) in HAR task without using any evolutionary *(Non-evolutionary)* neural network search algorithms, including CNN [71], CNN_LSTM [72], a conventional Vanilla- LSTM [73], Stacked-LSTM [73], Bi_LSTM [74], and iSPLInception [73].

The third category includes deep learning algorithms with evolutionary neural network search *(Evolutionary)*, Neuroevolutionary [60], AE-CNN [23], and CNN-GA [28].

3.2. Benchmark datasets

The dataset used in this study to model human activity is collected from different sources including *UCI-HAR, Daphnet* and *Opportunity*. As benchmark datasets in the experiment, we used three publicly available datasets that are utilized in many state-of-the-art CNNs architectural design algorithms for human activity recognition.

3.2.1. UCI dataset

Anguita et al. [75] propose that the UCI dataset is derived from 30-person video recordings of participants performing basic physical activities with an inertial sensor-enabled smartphone mounted on their waist. As part of this dataset, there are three static postures **(Standing, Sitting, Lying)**, as well as three dynamic activities **(Walking, Walking downstairs, and Walking upstairs)**. A total of 128 readings were taken in sliding windows with a fixed width of 2.56 seconds and 50% overlap. The noise was reduced by applying a median filter and a Butterworth filter with a cutoff frequency of 20 Hz. The acceleration signal was divided into body acceleration and gravity signals using a Butterworth lowpass filter.

3.2.2. Opportunity dataset

Roggen et al. [76] propose the Opportunity dataset, which contains naturalistic activities that were collected using 72 body and environmental and body sensors in an environment with rich sensor information. 12 subjects have been recorded using 15 networked sensory systems that contain 72 sensors of 10 modalities. During our analysis, only data from inertial measurement units that are included in columns 38 to 134 were taken into account. A triaxial accelerometer, a gyroscope, a magnetometer, as well as other measurements were used, but the quaternion measurements were excluded. Thus, we were able to receive 77 signals (channels) as input. A 30-Hz sampling rate was used to sample the data, and a 3-second window was extracted that consisted of 90 samples per window. A multiclass classification problem was originally posed on the Opportunity dataset with 18 classes, but we excluded the null class, adding a new label, and therefore used 17 classes instead. There is a disparity in the distribution of data amongst the classes in this dataset, which leads to an imbalance in the dataset.

3.2.3. DAPHNET dataset

In an article published in 2010, Bachlin et al. [77] proposed a dataset, Daphnet Freezing of Gait (Daphnet). In this dataset, automatic algorithms are benchmarked against wearable acceleration sensors placed on the thighs and hips for the detection of gait freeze. Freezing of Gait (FOG) is a sudden and transient inability to walk that affects approximately half of the patients with Parkinson's Disease (PD). Falling is often the result of this condition, it interferes with everyday activities, and it has a significant negative impact on quality of life. There has been an increased interest in nonpharmacological treatments for gait deficits in patients suffering from Parkinson's disease (PD) due to their resistance to pharmacological treatment. The team developed an innovative wearable system that detects fog in real-time and automatically provides a sound of signaling until it walks again. An evaluation of this wearable assistive technology was performed with 10 patients with Parkinson's disease. Two activities are contained in this dataset **(freeze and No Freeze)**. Based on the sampling rate of 64 Hz, the data was sampled in 3-second sliding windows with a 50% overlap, resulting in 192 readings per window. It is apparent from this dataset that there is a significant imbalance between the No Freeze and Freeze classes due to the disparity in percentages.

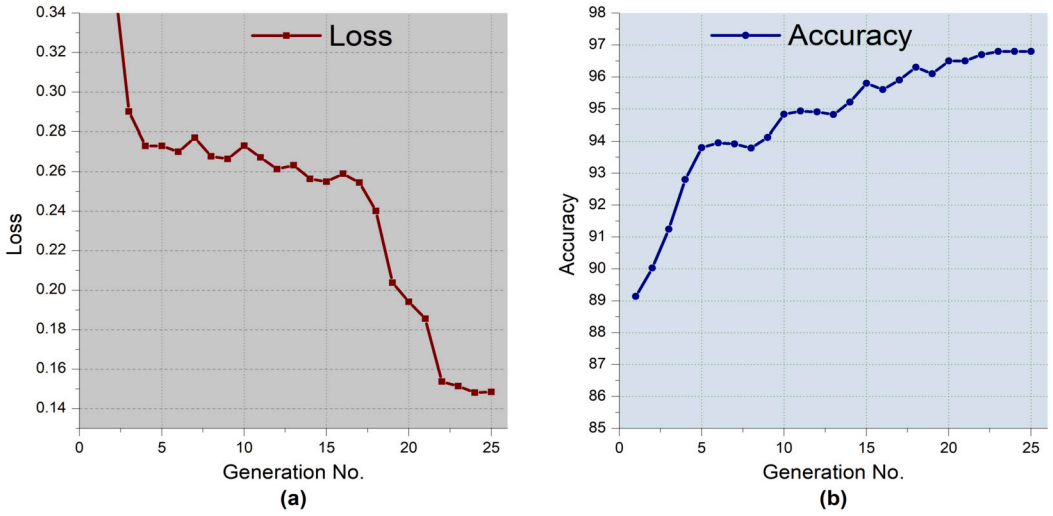

Fig. 6. The performance of the proposed model for 25-generations of UCI dataset (a) loss and (b) mean accuracy.

3.3. Evaluation metrics

For evaluating the proposed AUTO-CNN algorithm, we used $accuracy$, $Sensitivity$ (Recall), $Specificity$ (True Negative Rate), FDR (False Discovery Rate), and $F-score$ measures. The $Accuracy$ value reflects the proportion of samples that were correctly classified to the total number of samples. The $Sensitivity$ indicates the percentage of records correctly identified in relation to the total number of activities. A $specificity$ calculated by dividing the total number of correctly classified activities by the percentage of such activities records in the dataset FDR measures the probability that an activity that has been classified as significant is actually null. The $F-score$ parameter is the balanced weight of the precision and recalls values of the tested data. The equations (1), (2), (3), (4), and (5) illustrate how these evaluation metrics were calculated as shown below. Where, TP is true positive, TN is true negative, FP is false positive and FN is false negative.

$$Accuracy = \frac{(TP+TN)}{(TP+FP+FN+TN)} \tag{1}$$

$$Sensitivity = \frac{TP}{(TP+FN)} \tag{2}$$

$$Specificity = \frac{TP}{(TP+FP)} \tag{3}$$

$$FDR = \frac{FP}{(FP+TP)} \tag{4}$$

$$F-score = \frac{(2*TP)}{(2*TP+FP+FN)} \tag{5}$$

4. Experimental results and discussion

Using the proposed CNN architecture design and the AUTO-CNN algorithm, the experiment is designed to verify whether it is feasible to achieve promising performance on the classification task of Human Activity Recognition. In this section, as a first step, we will introduce the proposed model performance using three datasets, followed by a comparison of the peers' performance versus the proposed algorithm. The three used datasets are splitted into 70% for training and 30% for testing. To test scalability, cross-validation methods were used with a ten-fold increase in the learning rate, with one-third (40%) of the data being used for validation and the remainder being used for training. The data split occurred only once for each dataset with a fixed seed number (random state = 2) to ensure that different subjects participated in the classification process with different activities. Furthermore, there was no overlap among the selected activities.

4.1. UCI dataset experiments

As we used the AUTO-CNN algorithms to address HAR classification problem, it should illustrate whether they have converged or not when they terminate. As a result, the evolutionary curve of the proposed algorithm in terms of the investigated benchmark datasets is discussed and explained.

A statistical analysis is conducted for each individual in each generation by recording their classification accuracy. Fig. 6 (a) illustrates the evolutionary curve of the proposed algorithm on UCI dataset, where the vertical axis represents the mean accuracy (mean accuracy of the individuals in the same generation) of test set and the horizontal axis represents the generation

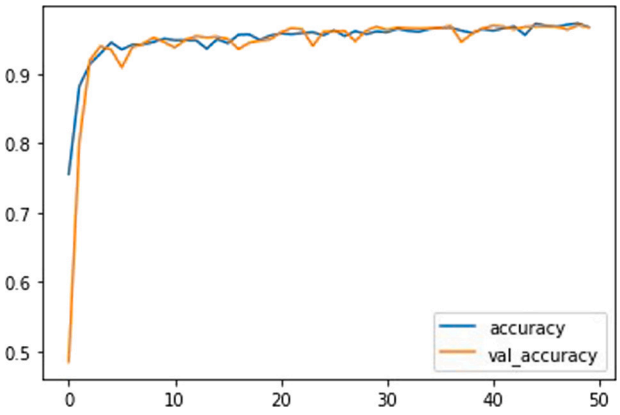

Fig. 7. Accuracy for 50 epochs from UCI dataset.

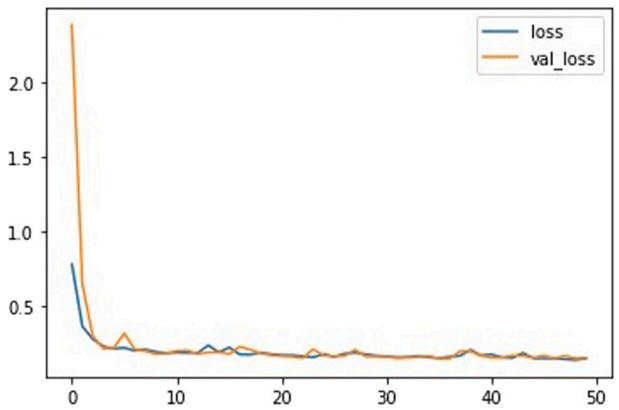

Fig. 8. Loss for 50 epochs from UCI dataset.

number. Fig. 6(b) shows the loss of the proposed AUTO-CNN through 25 generations, where the vertical axis donates the sparse_categorical_crossentropy loss and horizontal axis is generation number.

We can observe from Fig. 6(a) that, from the 1st to the 5th generation, the mean classification accuracy increases sharply; and then increased bit by bit as during evolution process until the 19th generation. Finally, the mean classification accuracy of the proposed algorithm converges in steady state when it terminates.

From Fig. 6(a), it can be observed that the mean classification accuracy in the first two generations is 89%, this due to the randomly initialized architecture cannot run because of the out-of-memory problem; As a result of their noncompetitive fitness, individuals with out-of-memory architectures will be eliminated from the population in the third generation, resulting in a steady increase in classification accuracy until the algorithm has been terminated.

After obtaining the best architecture of CNN for UCI dataset using AUTO-CNN, we can illustrate the performance of the best CNN architecture of the training and validation accuracy and loss for AUTO-CNN in Fig. 7 and Fig. 8. In order to reduce the learning rate iteratively when the training plateaus, the model was trained for 50 epochs and used a learning rate of 0.001.

In Fig. 9, we demonstrate that our model performs better at separating the different classes based on the confusion matrices. Where the classes are labeled from 0 to 5 to represent the six classes **'laying', 'walking', 'walking_upstairs', 'walking_downstairs', 'sitting', 'standing'** respectively. Although the [sitting] and [standing] classes are easily confused due to their similarities, AUTO-CNN overcomes this and classifies them apart with more precision, where it achieves 97.3% for setting and 87.3% for standing class.

Additionally, we evaluated the performance of AUTO-CNN in the recognition of each activity in the UCI-HAR dataset using 10-fold cross-validation with a number of evaluation metrics, such as accuracy, loss, precision, sensitivity, specificity, and false discovery rate (FDR). A comparison of the FDR and specificity is shown in Figs. 10 and 11. Tables 4 and 5 show the sensitivity and precision of the AUTO-CNN model trained on the UCI-HAR dataset. It is clear from both tables that the average precision and sensitivity for the 10-fold comparison were 98.5% and 98.6%, respectively. The sensitivities for sitting and standing were 96% and 95.7%, respectively. Additionally, the sensitivity for all activities was 100%, while that for sitting and standing was 95.0% and 96.8%, respectively. We can conclude that, with the exception of the two confused classes of sitting and standing, the Auto-CNN model achieves an average accuracy is 98.5% ($\mp$ 0.8) for all activities. This is because standing and sitting classes have a number of characteristics in common, which makes them easily mixed. This misclassification is caused by the sensor's placement on the waist, which causes the signals obtained from those actions to be quite similar.

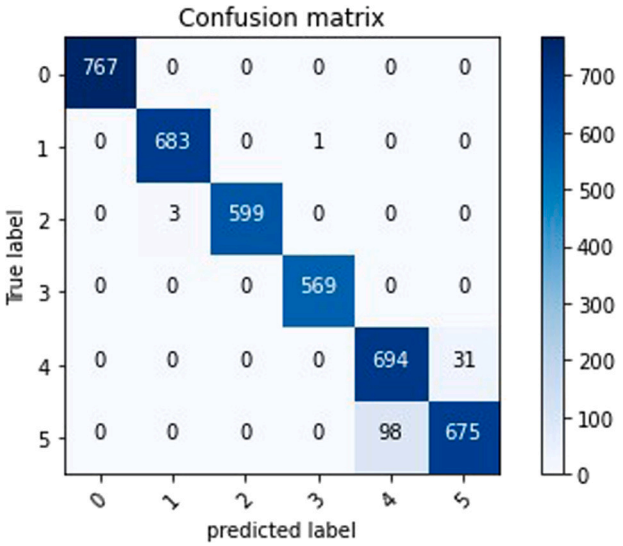

Fig. 9. Confusion matrix for UCI dataset of proposed model.

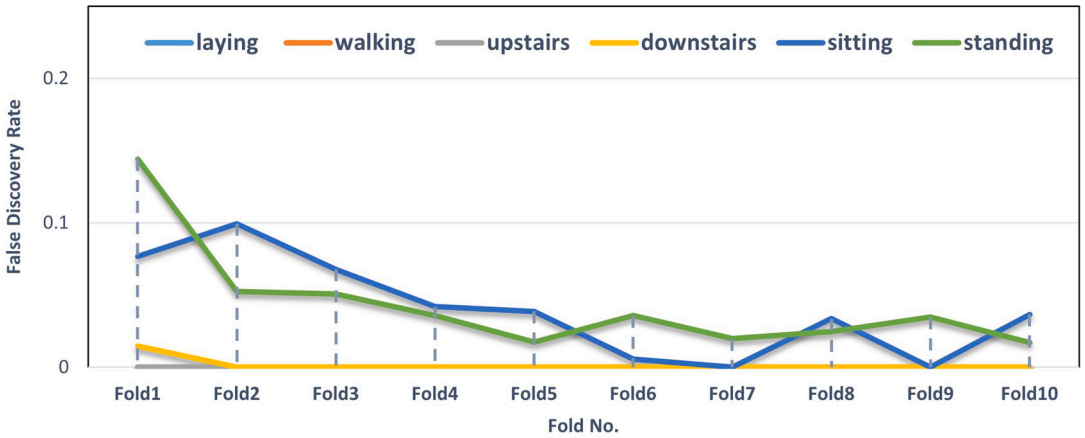

Fig. 10. *UCI_HAR* 10-Fold False Discovery Rate.

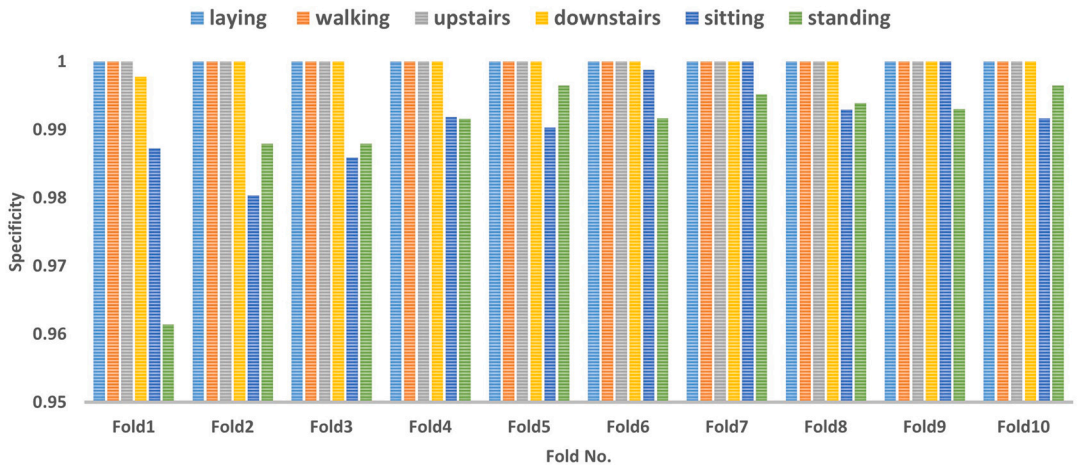

Fig. 11. *UCI_HAR* Specificity.

Table 4
UCI_HAR 6-activities 10-Fold cross validation (Precision).

Activity	Fold1	Fold2	Fold3	Fold4	Fold5	Fold6	Fold7	Fold8	Fold9	Fold10	Average
Laying	1	1	1	1	1	1	1	1	1	1	**1**
Walking	1	1	1	1	1	1	1	1	1	1	**1**
Upstairs	1	1	1	1	1	1	1	1	1	1	**1**
downstairs	0.985	1	1	1	1	1	1	1	1	1	**0.998**
Sitting	0.923	0.901	0.932	0.958	0.961	0.994	1	0.966	1	0.963	**0.959**
Standing	0.856	0.947	0.949	0.964	0.982	0.964	0.980	0.975	0.965	0.982	**0.956**
Average	**0.960**	**0.974**	**0.980**	**0.987**	**0.990**	**0.993**	**0.996**	**0.990**	**0.994**	**0.991**	**0.985**

Table 5
UCI_HAR 6-activities 10-Fold cross-validation (Sensitivity).

Activity	Fold#1	Fold#2	Fold#3	Fold#4	Fold#5	Fold#6	Fold#7	Fold#8	Fold#9	Fold#10	Average
laying	1	1	1	1	1	1	1	1	1	1	**1**
Walking	0.989	1	0.993	1	1	1	1	1	1	1	**0.998**
Upstairs	1	1	1	1	1	1	1	1	1	1	**1**
downstairs	1	1	1	1	1	1	1	1	1	1	**1**
Sitting	0.804	0.939	0.958	0.985	0.962	0.975	0.971	0.968	0.983	0.949	**0.95**
Standing	0.945	0.913	0.969	0.955	0.994	1	0.970	1	0.960	0.967	**0.968**
Average	**0.956**	**0.975**	**0.986**	**0.990**	**0.992**	**0.996**	**0.990**	**0.995**	**0.990**	**0.986**	**0.986**

Table 6
Comparison of performance of the AUTO-CNN and the peer competitors on UCI dataset.

Category	Model	Accuracy	Loss	F1-Score
Manual	AlexNet [68]	90.6	0.3201	86.6
Manual	ResNet-152 [70]	93.4	0.2985	93
Manual	VGGNet [46]	87.6	0.4005	82.4
Manual	GoogleNet [69]	88.8	0.4236	84.8
Non-evolutionary	CNN [71]	91.67	0.2977	92
Non-evolutionary	CNN-LSTM [72]	94.48	0.2137	94
Non-evolutionary	Vanilla- LSTM [73]	90.80	0.3296	91
Non-evolutionary	Stacked-LSTM [73]	91.82	0.3995	92
Non-evolutionary	BiLSTM [74]	93.91	0.2777	94
Non-evolutionary	iSPLInception [73].	95.09	0.1761	95
Evolutionary	Neuroevolutionary [60]	-	-	-
Evolutionary	AE-CNN [23]	95.3	0.1912	95
Evolutionary	CNN-GA [28]	96.02	0.1661	96
Evolutionary	**Proposed model**	**96.8**	**0.1485**	**96**

Table 6 shows the results of a comparison of the suggested algorithm with peer competitors using the UCI dataset. The first column lists the many types of architectures. The names of the architectures are listed in the second column. The third, fourth, and fifth columns pertain to the $Accuracy, Loss$, and $F1-Score$, respectively. All of the competitors' results in the table are taken from their published papers and compared with AUTO-CNN with the implementation of AE-CNN, and CNN-GA considering UCI dataset.

AUTO-CNN gets the highest test accuracy of 96.8% and F1 score of 97% compared with the other models. Table 6 illustrates the obtained results of the different models for the UCI dataset. The *iSPLInception* and the *CNN-GA* models results are close to AUTO-CNN. Where *iSPLInception* achieves a 1.71% lower accuracy and 95% F1 score. The $sparse\_categorical\_crossentropy$ loss for AUTO-CNN is 0.1485 which is better than the best of either the evolutionary-based models (CNN-GA) that achieves 0.1661, or non-evolutionary models. AUTO-CNN gets highly significant improve where its test accuracy of 96.8% and F1 score of 97% compared with the handcrafted CNN models. It is observed that the worst model is VGGNet handcrafted model that gets loss of 0.4005. The ResNet-152 follows AUTO-CNN. Where ResNet-152 achieves a 3.4% lower accuracy and 93% F1 score. For evolutionarily based algorithms, AE-CNN, and CNN-GA which utilize an efficient building blocks in their design space, the proposed method outperforms the classification accuracy results and F1-Score when compared to these techniques on UCI, by 1.08% and 1.8%, respectively.

4.2. Opportunity dataset experiments

After obtaining the best architecture of CNN for Opportunity dataset using AUTO-CNN, we can illustrate the performance of the best CNN architecture of the training and validation accuracy and loss for AUTO-CNN. In this model, the learning rate is 0.001 for 100 epochs, which is iteratively reduced as the training progresses.

Table 7
Comparison of performance of AUTO-CNN and the peer competitors on Opportunity dataset.

Category	Model	Accuracy	Loss	F1-Score
Manual	AlexNet [68]	-	-	-
Manual	ResNet-152 [70]	-	-	-
Manual	VGGNet [46]	-	-	-
Manual	GoogleNet [69]	-	-	-
Nonevolutionary	CNN [71]	80.24	0.8037	80
Nonevolutionary	CNN-LSTM [72]	81.41	0.5734	81
Nonevolutionary	Vanilla- LSTM [73]	76.79	0.7988	77
Nonevolutionary	Stacked-LSTM [73]	80.82	0.6405	81
Nonevolutionary	BiLSTM [74]	79.90	0.6559	80
Nonevolutionary	iSPLInception [73]	88.14	0.4790	88
Evolutionary	Neuroevolutionary [60]	93	0.91250	0.9047 - 0.9271
Evolutionary	AECNN [23]	98.8	0.1112	99
Evolutionary	CNNGA [28]	96.3	0.1531	96
Evolutionary	**Proposed model**	**98.3**	**0.1405**	**98**

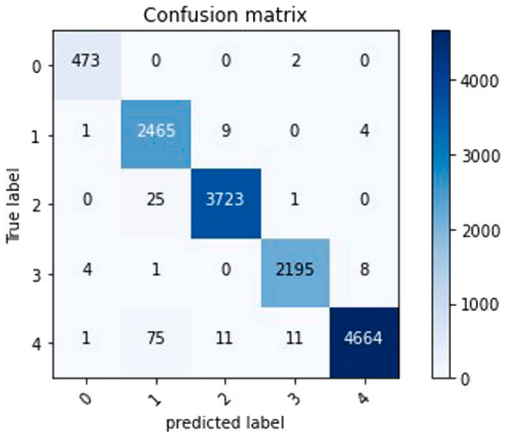

Fig. 12. Confusion matrix for Opportunity dataset of proposed model.

Table 7 summarizes the obtained results of the different models for the Opportunity dataset. The first column lists the many types of architectures. The names of the architectures are listed in the second column. The third, fourth, and fifth columns pertain to the Accuracy, Loss, and F1-Score, respectively. All of the competitors' results in the table are taken from their published papers except the evolutionary-based models which are implemented and evaluated based on Opportunity dataset, and '-' indicate that no results published with the mentioned model based on Opportunity dataset.

AUTO-CNN gets highly significant improve on this dataset compared to other models. AUTO-CNN gets accuracy of 98.3% and F1-score of 98%. However its performance results was lower than $AE-CNN$ by 0.5% on Opportunity HAR. The non-evolutionary $iSPLInception$ models come after $AUTO-CNN$. Where $iSPLInception$ achieves a 10.16% lower accuracy and 88% F1 score.

In Fig. 12, the confusion matrices AUTO-CNN show that the proposed model performs better in distinguish the different classes apart. The classes are labeled from 0 to 4 to represent the five highest-level classes of the Opportunity dataset, which include (lie, relax), (stand, coffee time), (walk, early morning), (sit, clean up), and (sandwich time).

4.3. Daphnet dataset experiments

To further test the effectiveness of the classification accomplished by the automatic CNN design, in this section we apply our AUTO-CNN architecture on Daphnet dataset. The performance of the best CNN architecture of the training and validation accuracy for AUTO-CNN is illustrated in Fig. 13. It achieved high performance where test accuracy of 94.8%. In Fig. 14, the confusion matrix of AUTO-CNN shows that however, Daphnet is imbalanced dataset, the proposed model performs better in distinguish its different classes apart. Table 8 and 9 illustrate how the model recognizes the two activities of the Daphnet dataset using sensitivity and precision metrics and a 10-fold cross-validation process. The AUTO-CNN architecture increased average precision and sensitivity by 10-folds, to 98.4% and 98.2%, respectively. The precision for the freeze activity (99.4%) is higher than that for the no-freeze activity (97.3%). Furthermore, the sensitivity for freezing was 99.6%, which is higher than no-freeze activity at 96.7%. Fig. 15 and 16 illustrate the FDR and specificity of each activity, respectively. We found that the proposed model could accurately identify the behavior of both Daphnet activities with an accuracy of 99.14% ($\mp$1.2) and a loss of 0.052.

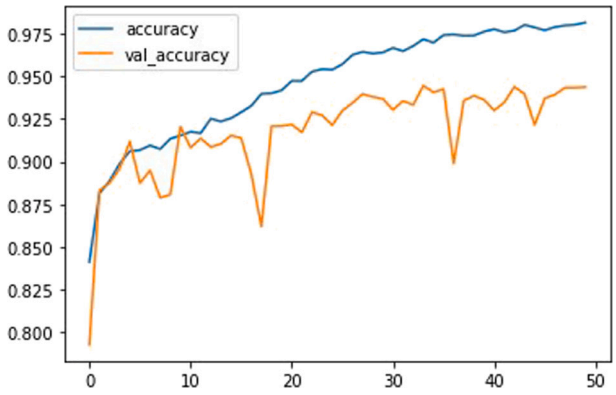

Fig. 13. Accuracy of AUTO-CNN on Daphnet dataset.

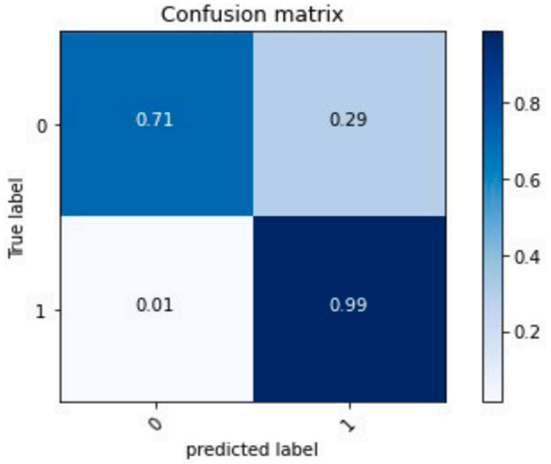

Fig. 14. Confusion matrix for Daphnet dataset of proposed model.

Table 8
Cross-validation of Daphnet 2-activities by 10-fold (sensitivity).

Activity	Fold1	Fold2	Fold3	Fold4	Fold5	Fold6	Fold7	Fold8	Fold9	Fold10	Average
Freeze	0.793	0.950	0.981	0.987	0.990	0.993	0.985	1	1	0.990	0.967
No-freeze	0.986	0.987	0.995	0.997	0.997	0.998	0.999	0.998	0.998	0.999	0.996
Average	0.889	0.969	0.988	0.992	0.994	0.996	0.992	0.999	0.999	0.994	0.982

Table 9
Cross-validation of Daphnet 2-activities by 10-fold (Precision).

Activity	Fold1	Fold2	Fold3	Fold4	Fold5	Fold6	Fold7	Fold8	Fold9	Fold10	Average
Freeze	0.909	0.923	0.972	0.982	0.984	0.993	0.995	0.988	0.987	0.995	0.973
No-freeze	0.964	0.992	0.996	0.998	0.998	0.998	0.997	1	1	0.998	0.994
Average	0.937	0.958	0.984	0.990	0.991	0.996	0.996	0.994	0.994	0.996	0.983

Table 10 summarizes the obtained results of the different models for the Daphnet dataset while implementing and evaluating the evolutionary based models (AUTO-CNN, CNN-GA).

The first column lists the many types of architecture. The names of the architectures are listed in the second column. The third, fourth, and fifth columns pertain to the Accuracy, Loss, and F1-Score, respectively. '-' indicate that no results published with the mentioned model based on Daphnet dataset. AUTO-CNN gets highly significant improve on classification accuracy using this dataset compared to other models. AUTO-CNN gets accuracy of 94.8% and F1-score of 95%. AUTO-CNN is better than CNN-GA model which achieves a 0.78% lower accuracy and 94% F1 score.

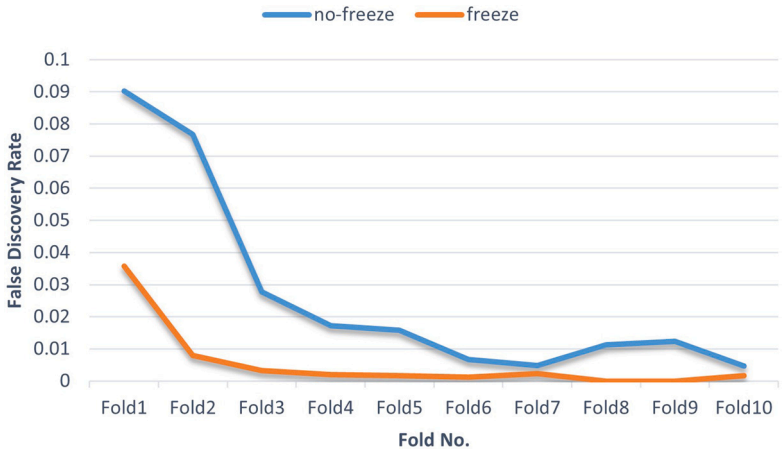

Fig. 15. *Daphnet* 10-Fold False Discovery Rate.

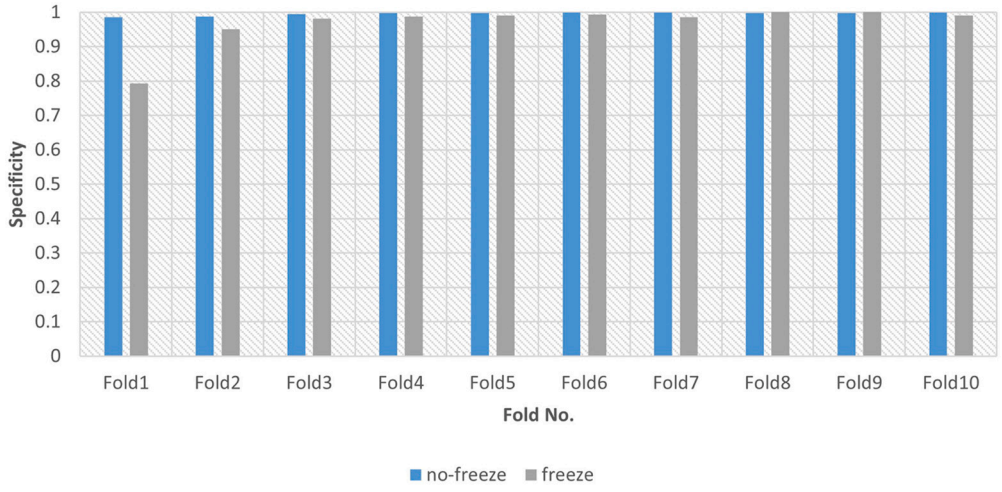

Fig. 16. *Daphnet* Specificity.

Table 10
Comparison of performance of the proposed model and the peer competitors on Daphnet dataset.

Category	Model	Accuracy	Loss	F1-Score
Manual	AlexNet [68]	-	-	-
Manual	ResNet-152 [70]	-	-	-
Manual	VGGNet [46]	-	-	-
Manual	GoogleNet [69]	-	-	-
Non-evolutionary	CNN [71]	92.97	0.2768	93
Non-evolutionary	CNN-LSTM [72]	92.97	0.2558	93
Non-evolutionary	Vanilla- LSTM [73]	93.22	0.2392	93
Non-evolutionary	Stacked-LSTM [73]	87.65	0.2969	88
Non-evolutionary	BiLSTM [74]	92.41	0.2479	92
Non-evolutionary	iSPLInception [73]	93.52	0.2771	94
Evolutionary	Neuroevolutionary [60]	-	-	-
Evolutionary	AE-CNN [23]	93.9	0. 2269	94
Evolutionary	CNN-GA [28]	94.02	0. 2012	94
Evolutionary	**Our model**	**94.80**	**0. 1889**	**95**

4.4. The designed architectures

Fig. 17 and Fig. 18 illustrates the best searched architectures discovered by the proposed algorithm for two datasets. The best CNN architecture for UCI dataset consists of four convolutional blocks, whereas the best CNN architecture designed on the Daphnet dataset consists of seven convolutional blocks. In contrast to UCI, Daphnet's architecture has three more convolutional blocks,

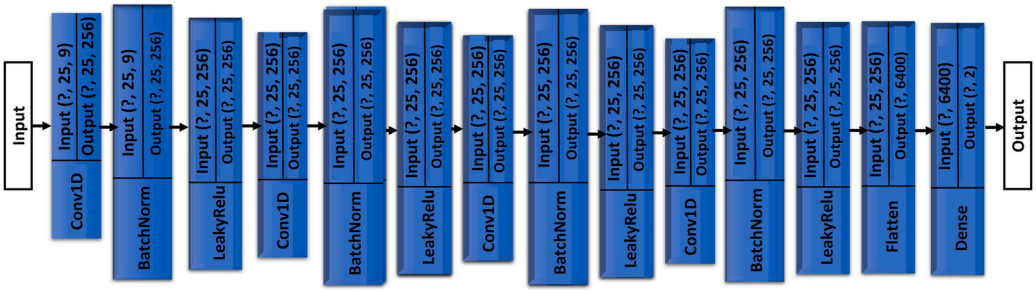

Fig. 17. The best architecture for UCI dataset.

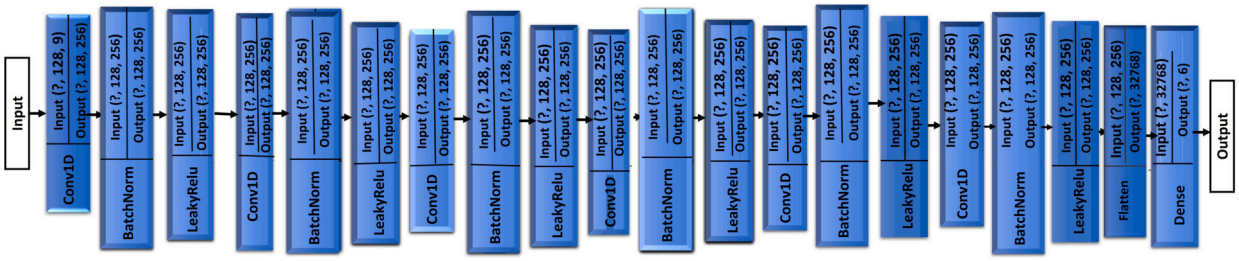

Fig. 18. The best architecture for Daphnet dataset.

indicating its more complex nature. Therefore, the proposed algorithm is capable of identifying the number and parameters of network building-blocks as well as estimating architectures according to the complexity of each task.

5. Conclusion

In this paper, we propose an efficient AUTO-HAR framework that can be used to design CNN architectures automatically and efficiently for the task of human activity recognition. Considering the used datasets, a large number of labeled observations are collected from heterogeneous sensors (ambient, wearable, and object sensors). The use of traditional machine learning techniques will have many obstacles because of the variety of ranges and mixed frequencies. Additionally, various dimensionality, heterogeneity, and features numbers must be selected and adjusted. Considering all the mentioned complexity of the selected datasets, the AUTO-CNN method for selecting the best CNN architecture is proven to be successful in the classification of human activities recognition process. By designing a CNN-based building blocks and designing the evolutionary operators corresponding to them which reduce the search space of the architectures. The application of the proposed approach to real-world problems has quite a lot of potential. AUTO-HAR can be used to develop practical applications with higher complexity, and it can be modified based on specific tasks, such as vehicle reidentification or trajectory prediction.

6. Limitations and challenges for future work

We highlight a few key aspects that require continued efforts in order to make AUTO-CNN more competitive than state-of-the-art CNN-based models:

- One of the key drawbacks of the suggested AUTO-CNN in this work is that the initial CNN architecture must be pre-defined using best state-of-the-art architecture, which implies that the structure of CNN models cannot be changed after the GA has begun searching for the optimal answer. Moreover, the best design found for each dataset in this study may not perform well when applied to a problem with a different sensor configuration. Therefore, deep learning can be applied to a wide range of domains and concerns by using a system that can infer optimal topologies automatically.
- Another point is the mutation operation, where AUTO-CNN utilized a greedy strategy to shift the order of Convolution layer output dimensions in ascending order. We are doing this because we are dedicated to achieving a promising result as quickly as possible. In many cases, with adequate processing power, the maximum generation can be set to more than 100 layers at a time. Future work should improve the accuracy of performance predictions by improving the fitness evaluation and the search space algorithm efficiency
- A further direction for future research would be to incorporate an efficiency measure (FLOPs [78] or estimated GPU time [20]) to find an optimal model that balances accuracy and latency.
- In addition, it is necessary to have mathematical methods for calculating the complexity of the problem domain and setting appropriate evolutionary conditions.

Funding

This work received funding from the Deanship of Scientific Research at King Saud University.

Declaration of competing interest

The authors declare no conflict of interest.

References

[1] S. Zhang, Y. Li, S. Zhang, F. Shahabi, S. Xia, Y. Deng, N. Alshurafa, Deep learning in human activity recognition with wearable sensors: a review on advances, Sensors 22 (4) (2022) 1476.
[2] W.N. Ismail, M.M. Hassan, H.A. Alsalamah, Context-enriched regular human behavioral pattern detection from body sensors data, IEEE Access 7 (2019) 33834–33850.
[3] G. Gao, Z. Li, Z. Huan, Y. Chen, J. Liang, B. Zhou, C. Dong, Human behavior recognition model based on feature and classifier selection, Sensors 21 (23) (2021) 7791.
[4] R.R. Subramanian, V. Vasudevan, A deep genetic algorithm for human activity recognition leveraging fog computing frameworks, J. Vis. Commun. Image Represent. 77 (2021) 103132.
[5] M.A.K. Quaid, A. Jalal, Wearable sensors based human behavioral pattern recognition using statistical features and reweighted genetic algorithm, Multimed. Tools Appl. 79 (9) (2020) 6061–6083.
[6] A. Allegra, A. Tonacci, R. Sciaccotta, S. Genovese, C. Musolino, G. Pioggia, S. Gangemi, Machine learning and deep learning applications in multiple myeloma diagnosis, prognosis, and treatment selection, Cancers 14 (3) (2022) 606.
[7] H. Lee, J. Lee, Scalable deep learning-based recommendation systems, ICT Express 5 (2) (2019) 84–88.
[8] Y. Zhou, Natural language processing with improved deep learning neural networks, Sci. Program. (2022).
[9] S.A. Solla, Y. Le Cun, Constrained neural networks for pattern recognition, in: Neural Networks: Concepts, Applications and Implementations, Prentice Hall, 1991.
[10] M.F. Aslan, A. Durdu, K. Sabanci, M.A. Mutluer, Cnn and hog based comparison study for complete occlusion handling in human tracking, Measurement 158 (2020) 107704.
[11] H.H. Aghdam, E.J. Heravi, Guide to Convolutional Neural Networks, vol. 10(978-973), Springer, New York, NY, 2017, p. 51.
[12] N. Li, L. Ma, G. Yu, B. Xue, M. Zhang, Y. Jin, Survey on evolutionary deep learning: principles, algorithms, applications and open issues, arXiv preprint, arXiv:2208.10658.
[13] Q. Zhao, D. Zhang, L. Zhang, H. Lu, Evolutionary discriminant feature extraction with application to face recognition, EURASIP J. Adv. Signal Process. 2009 (2009) 1–12.
[14] A. Al-Taei, M.F. Ibrahim, N.J. Habeeb, Optimizing the performance of KNN classifier for human activity recognition, in: International Conference on Advances in Computing and Data Sciences, Springer, 2021, pp. 373–385.
[15] F. Daghero, A. Burrello, C. Xie, M. Castellano, L. Gandolfi, A. Calimera, E. Macii, M. Poncino, D.J. Pagliari, Human activity recognition on microcontrollers with quantized and adaptive deep neural networks, ACM Trans. Embed. Comput. Syst. 21 (4) (2022) 1–28.
[16] N. Rashid, B.U. Demirel, M.A. Al Faruque Ahar, Adaptive CNN for energy-efficient human activity recognition in low-power edge devices, IEEE Int. Things J. (2022).
[17] S. Mirjalili, Evolutionary Algorithms and Neural Networks, Studies in Computational Intelligence, vol. 780, 2019.
[18] L. Guo, H. Zhang, C. Wang, W. Guo, G. Diao, B. Lu, C. Lin, L. Wang, Towards CSI-based diversity activity recognition via LSTM-CNN encoder-decoder neural network, Neurocomputing 444 (2021) 260–273.
[19] N. Sinha, K.-W. Chen, Novelty driven evolutionary neural architecture search, arXiv preprint, arXiv:2204.00188.
[20] Y. Xie, H. Chen, Y. Ma, Y. Xu, Automated design of CNN architecture based on efficient evolutionary search, Neurocomputing 491 (2022) 160–171.
[21] Z. Zhang, S. Liu, Y. Zhang, W. Chen, Rs-darts: a convolutional neural architecture search for remote sensing image scene classification, Remote Sens. 14 (1) (2021) 141.
[22] Z. Zeng, Q. Ji, Knowledge based activity recognition with dynamic Bayesian network, in: European Conference on Computer Vision, Springer, 2010, pp. 532–546.
[23] Y. Sun, B. Xue, M. Zhang, G.G. Yen, Completely automated CNN architecture design based on blocks, IEEE Trans. Neural Netw. Learn. Syst. 31 (4) (2019) 1242–1254.
[24] W. Gu, F. Gao, R. Li, J. Zhang, Learning universal network representation via link prediction by graph convolutional neural network, J. Soc. Comput. 2 (1) (2021) 43–51.
[25] F. Liu, Z. Zhang, R. Zhou, Automatic modulation recognition based on CNN and GRU, Tsinghua Sci. Technol. 27 (2) (2021) 422–431.
[26] Y. Liu, Y. Sun, B. Xue, M. Zhang, G.G. Yen, K.C. Tan, a survey on evolutionary neural architecture search, IEEE transactions on neural networks and learning systems.
[27] I. Younas, A. Naeem, Optimization of sensor selection problem in IoT systems using opposition-based learning in many-objective evolutionary algorithms, Comput. Electr. Eng. 97 (2022) 107625.
[28] Y. Sun, B. Xue, M. Zhang, G.G. Yen, J. Lv, Automatically designing CNN architectures using the genetic algorithm for image classification, IEEE Trans. Cybern. 50 (9) (2020) 3840–3854.
[29] A.A. Alani, G. Cosma, A. Taherkhani, Classifying imbalanced multi-modal sensor data for human activity recognition in a smart home using deep learning, in: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE, 2020, pp. 1–8.
[30] M. Islam, S. Nooruddin, F. Karray, G. Muhammad, et al., Human activity recognition using tools of convolutional neural networks: a state of the art review, data sets, challenges and future prospects, arXiv preprint, arXiv:2202.03274.
[31] J. Yao, Y. Chen, A motion capture data-driven automatic labanotation generation model using the convolutional neural network algorithm, Wirel. Commun. Mob. Comput. (2022).
[32] K.K. Verma, B.M. Singh, Deep multi-model fusion for human activity recognition using evolutionary algorithms, Int. J. Inter. Multimed. Artif. Intell. 7 (2) (2021).
[33] L. Wang, Y. Xu, J. Cheng, H. Xia, J. Yin, J. Wu, Human action recognition by learning spatio-temporal features with deep neural networks, IEEE Access 6 (2018) 17913–17922.
[34] A.A. Yilmaz, M.S. Guzel, E. Bostanci, I. Askerzade, A novel action recognition framework based on deep-learning and genetic algorithms, IEEE Access 8 (2020) 100631–100644.
[35] C. Alippi, S. Disabato, M. Roveri, Moving convolutional neural networks to embedded systems: the AlexNet and VGG-16 case, in: 2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), IEEE, 2018, pp. 212–223.

[36] S.M. Sam, K. Kamardin, N.N.A. Sjarif, N. Mohamed, et al., Offline signature verification using deep learning convolutional neural network (CNN) architectures googlenet inception-v1 and inception-v3, Proc. Comput. Sci. 161 (2019) 475–483.
[37] A.J. Suresh, J. Visumathi, Inception ResNet deep transfer learning model for human action recognition using LSTM, Materials Today: Proceedings.
[38] S. Roychowdhury, M. Diligenti, M. Gori, Regularizing deep networks with prior knowledge: a constraint-based approach, Knowl.-Based Syst. 222 (2021) 106989.
[39] H.-G. Han, M.-L. Ma, H.-Y. Yang, J.-F. Qiao, Self-organizing radial basis function neural network using accelerated second-order learning algorithm, Neurocomputing 469 (2022) 1–12.
[40] L. Chambers, M.M. Gaber, Deepstreamos: fast open-set classification for convolutional neural networks, Pattern Recognit. Lett. (2022).
[41] K. Muralidharan, A. Ramesh, G. Rithvik, S. Prem, A. Reghunaath, M. Gopinath, 1d convolution approach to human activity recognition using sensor data and comparison with machine learning algorithms, Int. J. Cogn. Comput. Eng. 2 (2021) 130–143.
[42] A. Alhudhaif, K. Polat, O. Karaman, Determination of Covid-19 pneumonia based on generalized convolutional neural network model from chest x-ray images, Expert Syst. Appl. 180 (2021) 115141.
[43] Y.A. Andrade-Ambriz, S. Ledesma, M.-A. Ibarra-Manzano, M.I. Oros-Flores, D.-L. Almanza-Ojeda, Human activity recognition using temporal convolutional neural network architecture, Expert Syst. Appl. 191 (2022) 116287.
[44] M.C. Leong, D.K. Prasad, Y.T. Lee, F. Lin, Semi-CNN architecture for effective spatio-temporal learning in action recognition, Appl. Sci. 10 (2) (2020) 557.
[45] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.
[46] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint, arXiv:1409.1556.
[47] J. Huang, S. Wang, G. Zhou, W. Hu, G. Yu, Evaluation on the generalization of a learned convolutional neural network for MRI reconstruction, Magn. Reson. Imaging 87 (2022) 38–46.
[48] A.Z. Selvaggio, F.M.M. Sousa, F.V. da Silva, S.S. Vianna, Application of long short-term memory recurrent neural networks for localisation of leak source using 3D computational fluid dynamics, Process Saf. Environ. Prot. (2022).
[49] M.V. Narkhede, P.P. Bartakke, M.S. Sutaone, A review on weight initialization strategies for neural networks, Artif. Intell. Rev. 55 (1) (2022) 291–322.
[50] M.H. Abdelhafiz, M.I. Awad, A. Sadek, F. Tolbah, Sensor positioning for a human activity recognition system using a double layer classifier, Proc. Inst. Mech. Eng., H J. Eng. Med. 236 (2) (2022) 248–258.
[51] J.W. Kasubi, D. Manjaiah, Feature selection strategy for multi-residents behavior analysis in smart home environment, in: Data Management, Analytics and Innovation, Springer, 2022, pp. 11–26.
[52] X. Wang, X. Wang, T. Lv, L. Jin, M. He, Harnas: human activity recognition based on automatic neural architecture search using evolutionary algorithms, Sensors 21 (20) (2021) 6927.
[53] M. Loni, S. Sinaei, A. Zoljodi, M. Daneshtalab, M. Sjödin, Deepmaker: a multi-objective optimization framework for deep neural networks in embedded systems, Microprocess. Microsyst. 73 (2020) 102989.
[54] Q. Wang, S. Liu, A prediction model analysis of behavior recognition based on genetic algorithm and neural network, Comput. Intell. Neurosci. (2022).
[55] K.O. Stanley, R. Miikkulainen, Evolving neural networks through augmenting topologies, Evol. Comput. 10 (2) (2002) 99–127.
[56] K.O. Stanley, J. Clune, J. Lehman, R. Miikkulainen, Designing neural networks through neuroevolution, Nat. Mach. Intell. 1 (1) (2019) 24–35.
[57] J. Redmon, A. Farhadi, Yolov3: an incremental improvement, arXiv preprint, arXiv:1804.02767.
[58] H. Zhang, Y. Jin, R. Cheng, K. Hao, Efficient evolutionary search of attention convolutional networks via sampled training and node inheritance, IEEE Trans. Evol. Comput. 25 (2) (2020) 371–385.
[59] G. Park, Y. Yi, Condnas: neural architecture search for conditional CNNs, Electronics 11 (7) (2022) 1101.
[60] A. Baldominos, Y. Saez, P. Isasi, Evolutionary design of convolutional neural networks for human activity recognition in sensor-rich environments, Sensors 18 (4) (2018) 1288.
[61] P.G. Devarakonda, B. Bozic, Particle swarm optimization of convolutional neural networks for human activity prediction, in: Optimisation Algorithms and Swarm Intelligence, IntechOpen, 2022.
[62] H. Yang, Y.-s. Zhang, C.-b. Yin, W.-z. Ding, Ultra-lightweight CNN design based on neural architecture search and knowledge distillation: a novel method to build the automatic recognition model of space target ISAR images, Defence Technol. 18 (6) (2022) 1073–1095.
[63] A. Deshpande, K.K. Warhade, Hybrid features enabled adaptive butterfly based deep learning approach for human activity recognition, in: Machine Vision and Augmented Intelligence-Theory and Applications, Springer, 2021, pp. 341–363.
[64] K.-T. Yang, Artificial neural networks (ANNs): a new paradigm for thermal science and engineering, J. Heat Transf. 130 (9) (2008).
[65] A. Gulli, S. Pal, Deep Learning with Keras, Packt Publishing Ltd, 2017.
[66] A. Blansché, P. Gançarski, J.J. Korczak, Genetic algorithms for feature weighting: evolution vs. coevolution and Darwin vs. Lamarck, in: Mexican International Conference on Artificial Intelligence, Springer, 2005, pp. 682–691.
[67] S. Forrest, Genetic algorithms, ACM Comput. Surv. (CSUR) 28 (1) (1996) 77–80.
[68] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inf. Process. Syst. 25 (2012).
[69] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[70] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[71] D. Van Kuppevelt, C. Meijer, F. Huber, A. van der Ploeg, S. Georgievska, V.T. van Hees, Mcfly: automated deep learning on time series, SoftwareX 12 (2020) 100548.
[72] R. Mutegeki, D.S. Han, A CNN-LSTM approach to human activity recognition, in: 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), IEEE, 2020, pp. 362–366.
[73] M. Ronald, A. Poulose, D.S. Han, isplinception: an inception-resnet deep learning architecture for human activity recognition, IEEE Access 9 (2021) 68985–69001.
[74] N.T.H. Thu, D.S. Han, Utilization of postural transitions in sensor-based human activity recognition, in: 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), IEEE, 2020, pp. 177–181.
[75] D. Anguita, A. Ghio, L. Oneto, X. Parra, J.L. Reyes-Ortiz, Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine, in: International Workshop on Ambient Assisted Living, Springer, 2012, pp. 216–223.
[76] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, et al., Collecting complex activity datasets in highly rich networked sensor environments, in: 2010 Seventh International Conference on Networked Sensing Systems (INSS), IEEE, 2010, pp. 233–240.
[77] M. Bächlin, M. Plotnik, D. Roggen, N. Giladi, J.M. Hausdorff, G. Tröster, A wearable system to assist walking of Parkinson s disease patients, Methods Inf. Med. 49 (01) (2010) 88–95.
[78] M.E. Paoletti, J.M. Haut, X. Tao, J. Plaza, A. Plaza, Flop-reduction through memory allocations within CNN for hyperspectral image classification, IEEE Trans. Geosci. Remote Sens. 59 (7) (2020) 5938–5952.