A Comparative Study of ANN Tuning Methods for Multiclass Daily Activity and Fall Recognition

Tevfik Aktay[1(✉)] and Mehmet Önder Efe[2]

[1] Department of Industrial Engineering, Hacettepe University, Beytepe, Ankara, Turkey
tevfik.aktay12@hacettepe.edu.tr

[2] Department of Computer Engineering, Hacettepe University, Beytepe, Ankara, Turkey
onderefe@hacettepe.edu.tr

Abstract. Smart phones and other sensor-enabled devices are very frequently used daily life devices. Movement data obtained by sensors from these devices can be interpreted by artificial intelligence algorithms and this may be critically helpful in some daily life issues. Such a daily activity and fall classification mechanism is particularly important for rapid and accurate medical intervention to the elderly people who live alone. In addition, the real time human activity recognition (HAR) is important for healthcare solutions and better assistance of intelligent personal assistants (IPAs). In this study, the dataset is obtained from 6 different wearable sensors. It contains 20 daily activities and 16 fall motions on the 3060 observations. To classify these movements separately, 3 different Artificial Neural Network (ANN) training algorithms were chosen as the basis. These are gradient descent, momentum with gradient descent and Adam algorithms. Dropout and L2 regularization techniques are used to obtain better results for the test data. The results have shown that the ANN based approach correctly recognizes the daily activities and falls with 94.58% accuracy score on the test set.

Keywords: Multiclass classification · Human activity recognition · Fall detection · Artificial neural networks · Optimization · Regularization

1 Introduction

In recent years, development of hardware technology on sensors, GPS-enabled devices and miniaturized accelerometers have enabled the data collection and analysis of different kinds of data [1]. This has led to enormous amounts of real-time data, which allows for a variety of analytical insights [2]. With the development of these technologies and the rise in the living standards, human activity recognition and fall detection have gained more interest in applications,

C. Djeddi et al. (Eds.): MedPRAI 2019, CCIS 1144, pp. 24–38, 2020.
https://doi.org/10.1007/978-3-030-37548-5_3

such as health, entertainment services and technology. Today, most smart phones and wearable devices like smart watches have sensitive embedded sensors. These sensors include pedometer, accelerometer, gyroscope and magnetometer, thus they continuously gather movement data. Because of the growing volume of this data and easy to access to it, many research studies are devoted to recognize daily activity as well as falls using machine learning solutions.

Fall detection and classification are getting more important especially for elderly people for early and accurately treatment. According to [3], every one-third of elderly people who are 65 or older falls at least one time in a year. 20% of people who fall need medical treatment [4]. Therefore, fall detection and early alert systems are critical to prevent permanent injuries or saving people's lives. In these systems, false negative alarm is another noteworthy fact. In this case, a fall occurs but the learning algorithm does not label it as a fall. This means that the model does not perform one of the most critical tasks. This is an issue handled in this paper. Also, identification of the 16 different fall types can be helpful to manage accurate first aid.

Human daily activity recognition is a popular research area in machine learning. Some research studies have focused on the energy expenditure during the daily exercises [5]. Recognition of daily activities opens the data driven analysis for personalized health care programs. Also, it improves the quality of physical rehabilitation that is applied to elderly people [6]. Calculating daily activity type and their iteration counts provide information to doctors for patient-based care plans [7]. They can use the data as a feedback mechanism to follow their treatment effects. Another approach is to exploit the internet and mobile technologies for IPAs. Next generation smart assistants could merge the human activity recognition and daily routines for serving specific recommendation and reminder systems. This systems collects the user/patient location, heart rate, and sensor data [8]. Furthermore, collected information is forwarded to a caretaker IPA, in real time, that will manage a set of human activity recognition (HAR) and alarms appropriately. In this case, accuracy and reliability expectations on the daily activity classifiers get higher.

During the last two decades, the computational capacities of computing platforms have increased dramatically while the size and cost have been decreasing. These platforms have also provided an increasing number of different sensor data at increasing rates of sampling. Typically, the movement data is gathered by the wearable sensors, mobile-based sensors and recorded visual data [9]. However, personal privacy could be an issue in visual system solutions. Therefore, sensor based researches and developments have less security and privacy risks for the users. Machine learning and especially ANN algorithms are preferred to build models to improve the daily activity recognition and fall detection. A HAR study is considered in University of California Irvine (UCI) HAR dataset to classify 6 different daily activities [10]. Authors examined three models, namely, single layer neural networks, multi layer neural networks and reservoir computing with multi layer neural networks. The last model achieved to 96.2% accuracy rate on the test set. Another study uses video data to recognize totally 6 falls and daily

activities and accuracy of approximately 98% is reported in [11]. However, the experiments in [11] consider only indoor recordings. Lack of outdoor recordings is a major issue, further, video based recordings have several drawbacks, such as the motion of the other persons may result in incorrect decisions. Another study and its dataset, which we used in this paper is Simulated Falls and Daily Living Activities Dataset. It is acquired and considered by Özdemir et al. [12]. The dataset is available at UCI Repository and [12] covers 6 machine learning algorithms to improve binary classifier in between falls and daily activities. The work in [12] achieves the best results with the k-nearest neighbor (k-NN) classifier and least squares method (LSM). These two algorithms have scores above 99% in sensitivity, specificity, and accuracy measurements. A recent study [13], a convolutional neural network model is proposed to detect fall from accelerometer data. They achieved significant results on binary classification over 99% accuracy score and precision.

When the current studies are examined, it is seen that a small number of daily activities and fall movements are generally classified. However, a broad spectrum of daily activity and fall recognition is likely to bring more focused healthcare services and personal products. For example, a accurate prediction of the fall type can be provide a specific first aid for the elderly peoples. On the other hand, a successful classification of the personalized daily activities increase the recommendation performance of the IPAs in calculating the burned calories according to the activity type. In this context, the aim of the proposed article is to serve both groups that have been noted above by classifying more daily life activities and falls with high accuracy. To do that, we used different optimization and regularization techniques on ANN to achieve best classification results. Our motivation in using neural networks is to be successful in learning the patterns in the time series sensor data and the network-based approaches are suitable for classification problems.

The rest of the paper is organized as follows. In Sect. 2, we provide details of the dataset and also data preprocessing techniques to prepare it for the learning algorithms. Then, in Sect. 3, we describe the generic classification algorithm and its dependencies for our research. Also, the section includes details of 3 optimization algorithms and 2 regularization techniques. Section 4 provides a technical description about tests and analysis. Afterwards, Sect. 5 discusses the similarities and differences between the proposed model and the reference studies. Finally, Last section explains the conclusion and the remarks for future work.

2 Dataset and Preprocessing

In this section, we explain the dataset and its structure. The preprocessing steps as well as the feature extraction and regularization issues on the raw dataset are discussed in the sequel.

2.1 Dataset

In the laboratory environment, 20 falls and 16 daily living activities are examined by 17 volunteers. Each volunteer suits 6 wearable sensors and then repeats the 36 simulated movements 5 times. The experiments are foldered for 17 volunteers separately to the raw data folder. At the end of the data collection procedure, folder stores totally 18360 instance .txt files (17 volunteers $\times$ 36 movements $\times$ 5 trials $\times$ 6 sensors). All of the trials, which are realized by volunteers are collected by 6 wearable sensors, which are located at different points of the subjects' body. These wireless sensor units fit to the subjects' heads, chests, waists, right wrists, right thighs and right ankles. Recording of the data has been accomplished by using 6 Xsens MTw Motion Tracking Kit sensors and its MTw Software Development Kit made by Xsens Technologies [14]. With the recording capacity of the sensor unit, any trial data is tracked with 6 sensor units simultaneously with sampling frequency of 25 Hz [12]. This also means that data is received every 40 ms. There are approximately 375 rows of record, which corresponds to a sequence of sensory information that lasts 15 s for any sensor instance. Each sensor unit collects 22-columns recorded features, however 9 of them are used for the next steps. These features are the 3 perpendicular axes (x, y, z) of accelerometer, gyroscope and magnetometer. These axes are called as $A_x, A_y, A_z, G_x, G_y, G_z, M_x, M_y, M_z$. A, G, and M stand for the linear acceleration data from accelerometer, rotational acceleration from the gyros and the magnetometer data, respectively.

To preserve the reference dataset file structure, it is preferred to enumerate the motions same as in [12]. According to Table 1, fall motions are enumerated between 1 and 20, and also daily human activities are enumerated between 21 to 36.

2.2 Data Preprocessing

According to the dataset described above, a total of 6 different sensor-instances were collected simultaneously for each trial. Each instance consists 9 column features and 375 rows data. To decrease the computational burden and extracting the persistent information from the raw data, firstly, feature extraction and then dimensional reduction techniques are used. Furthermore, some daily activities and falls can be easily misclassified because they share similar motion patterns. Therefore, extraction of the new features is considered from the instance groups. The total acceleration of the waist accelerometer is selected for the further feature extraction steps as a reference. The magnitude of the measured acceleration is given in (1).

$$A_T = \sqrt{A_x^2 + A_y^2 + A_z^2} \tag{1}$$

The expression above is used to identify the peak acceleration value of the waist accelerometer instance in each trial. After that, the data is separated into 2-seconds (50 samples) blocks centering the peak instant. Thus, waist sensor file is reduced to the 101 rows which are placed before and after the peak value

Table 1. Fall and daily activities according to the dataset.

Motion ID	Label	Detailed description
1	Front-lying	From vertical falling forward to the floor
2	Front-protecting-lying	From vertical falling forward to the floor with arm protection
3	Front-knees	From vertical falling down on the knees
4	Front-knees-lying	From vertical falling down on the knees and then lying on the floor
5	Front-quick-recovery	From vertical falling on the floor and quick recovery
6	Front-slow-recovery	From vertical falling on the floor and quick recovery
7	Front-right	From vertical falling down on the floor, ending in right lateral position
8	Front-left	From vertical falling down on the floor, ending in left lateral position
9	Back-sitting	From vertical falling on the floor, ending sitting
10	Back-lying	From vertical falling on the floor, ending lying
11	Back-right	From vertical falling on the floor, ending lying in right lateral position
12	Back-left	From vertical falling on the floor, ending lying in left lateral position
13	Right-sideway	From vertical falling on the floor, ending lying
14	Right-recovery	From vertical falling on the floor with subsequent recovery
15	Left-sideway	From vertical falling on the floor, ending lying
16	Left-recovery	From vertical falling on the floor with subsequent recovery
17	Rolling-out-bed	From lying, rolling out of bed and going on the floor
18	Podium	From vertical standing on a podium going on the floor
19	Syncope	From standing falling on the floor following a vertical trajectory
20	Syncope-wall	From standing falling down slowly slipping on a wall
21	Walking-fw	Walking forward
22	Walking-bw	Walking backward
23	Jogging	Running
24	Squatting-down	Squatting, then standing up
25	Bending	Bending about 90 degrees
26	Bending-pick-up	Bending to pick up an object on the floor
27	Limp	Walking with a limp
28	Stumble	Stumbling with recovery
29	Trip-over	Bending while walking and then continuing walking
30	Coughing-sneezing	Coughing or sneezing
31	Sit-chair from vertical	To sitting with a certain acceleration onto a chair (hard surface)
32	Sit-sofa from vertical	To sitting with a certain acceleration onto a sofa (soft surface)
33	Sit-air from vertical	To sitting in the air exploiting the muscles of legs
34	Sit-bed from vertical	To sitting with a certain acceleration onto a bed (soft surface)
35	Lying-bed	From vertical lying on the bed
36	Rising-bed	From lying to sitting

generated by (1). Data from the remaining axes of each sensor unit are reduced according to the time index of the waist sensor used as a reference for the related trial. Each movement file collected by six sensors are reduced six 101 × 9 array data. Then, new features are extracted from 9 measured signals by using minimum, maximum, mean, skewness, kurtosis, the first 11 values of the autocorrelation sequence and the first 5 frequencies and 5 amplitudes with maximum magnitude of the Discrete Fourier Transform (DFT) for every column vectors (10 DFTs features × 9 columns + 11 autocorrelation features × 9 columns + first 5 statistical metrics × 9 columns). In this way, 234 features are extracted from each sensor unit. After that, remaining 5 sensors' data are processed same way and all six are merged as a feature vector of dimension 1404 × 1 (234 features × 6 sensors) for each trial.

There are totally 3060 trials in the dataset. Each trial is labeled according to the its movement type from 1 to 36. So our extracted dataset dimension is 1440 × 3060 array of data. Firstly, the dataset is normalized to the interval [0, 1] for reducing the computational complexity. Also, we reduced the number of features from 1440 to 60 components using Principal Component Analysis (PCA). PCA is an approach for deriving a low-dimensional set of features from a large set of variables [15]. After the implementation of PCA, 60 components represent 90% of the total variance. Their cumulative explained variance distributions are illustrated with all features and principal components in Figs. 1 and 2 respectively.

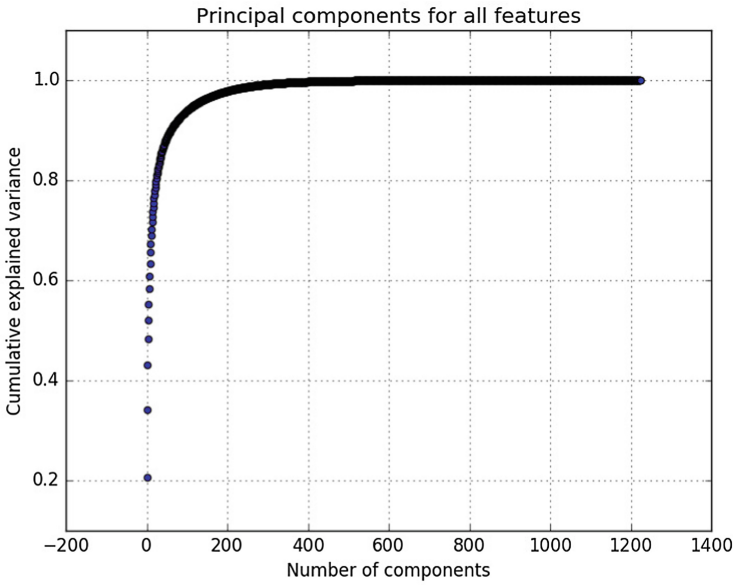

Fig. 1. The illustration of the all features with their cumulative explained variance.

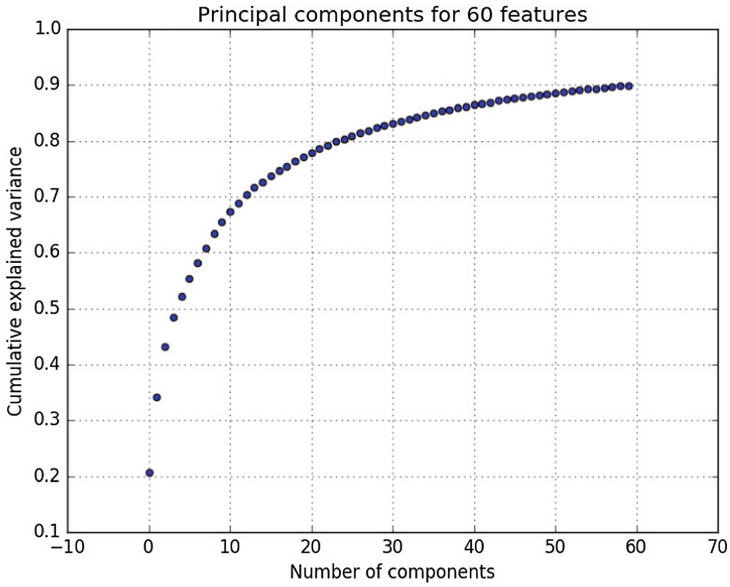

Fig. 2. The illustration of the 60 principal components after PCA with their cumulative explained variance.

3 Architecture of Neural Networks

In this study, a two-layer ANN structure is used in all experiments and it is outlined in Fig. 3. Firstly, hidden layer and output layer weights are initialized using the initialization approach discussed in He et al. [16] to improve of the performance of the model. This method remembers the previous layer size and that helps to reach minimum of the cost function quickly and efficiently. Afterwards, inputs x_j are multiplied by the weights W_{ij}^L in hidden layer and bias B_i^L values are added as shown in (2)–(3). Next, new values y_i are processed by the hyperbolic tangent activation function in (4) and the evaluated values are sent to the output layer weights W_{ij}^R and bias B_i^R in (5). Then, sigmoid activation function is used at the output layer. Output layer produces a matrix that contains probabilities between 0 and 1. According to (6)'s output matrix, a value larger than 0.5 is equal to the predicted class for the movement. At the end of the forward propagation using equations (2)–(7), the Mean Squared Error (MSE) is computed by using the output matrix of the network and desired output values of the dataset (See (7)).

$$\tau = \text{sigmoid}\left(W^R \tanh(W^L x + B^L) + B^R\right) \tag{2}$$

where W^L is the leftmost weight matrix, W^R is the rightmost weight matrix, B^L and B^R are the corresponding bias vectors respectively. The neural network has m inputs and n outputs, and there are M_1 hidden neurons. With these

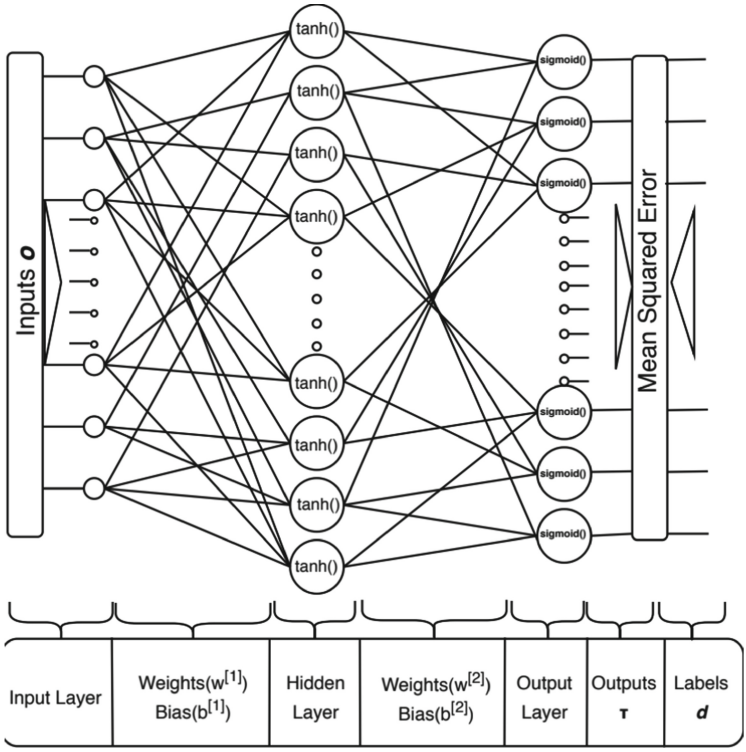

Fig. 3. A generic form of the two-layer neural network structure.

definitions, in the above input/output relation, $W_L \in \Re^{M_1 \times m}$, $B_L \in \Re^{M_1 \times 1}$, $W_R \in \Re^{n \times M_1}$ and $B_R \in \Re^{n \times 1}$.

The above equation can be written explicitly as follows

$$y_i^L = \sum_{j=1}^{m} W_{ij}^L x_j + B_i^L, \quad i = 1, 2, \ldots, M_1 \tag{3}$$

$$o_i^L = \tanh\left(y_i^L\right) \quad \text{where} \quad i = 1, 2, 3, \ldots, M_1 \tag{4}$$

$$y_i^R = \sum_{j=1}^{M_1} W_{ij}^R o_j^L + B_i^R \quad \text{where} \quad i = 1, 2, 3, \ldots, n \tag{5}$$

$$o_i^R = \text{sigmoid}(y_i^R) = \frac{1}{1 + e^{-y_i^R}} \quad \text{where} \quad i = 1, 2, 3, \ldots, n \tag{6}$$

$$E := \frac{1}{2n} \sum_{i=1}^{n} (d_i - \tau_i)^2, \quad \tau_i = o_i^R \tag{7}$$

To decrease the training time and computational burden, number of neurons in hidden layers and input batch size are determined on the entire dataset. Then,

3 different optimization algorithms are used to compare classification models' performances on test set using 10-fold cross validation technique. The optimization algorithms used in this study are gradient descent, gradient descent with momentum and Adam algorithm. In order to prevent overtraining and memorization, all training algorithms are regularized by using dropout and L2 cost regularization techniques. A total of 9 different models are examined in this study and these models are shown in Table 2.

Table 2. 9 Models which are examined in this study.

Optimization algorithm	Optimization algorithm with regularization technique
Gradient descent	Gradient descent without regularization
	Gradient descent with dropout
	Gradient descent with L2 cost regularization
Gradient descent with momentum	G.D. Momentum without regularization
	G.D. Momentum with dropout
	G.D. Momentum with L2 cost regularization
Adam optimization algorithm	Adam without regularization
	Adam with dropout
	Adam with L2 cost regularization

As given in (2)–(7), forward propagation step is followed by gradient descent algorithm. This is given in (8)–(11).

$$W_{ij}^{L}(k+1) = W_{ij}^{L}(k) - \alpha \frac{\partial E}{\partial W_{ij}^{L}(k)} \tag{8}$$

$$W_{ij}^{R}(k+1) = W_{ij}^{R}(k) - \alpha \frac{\partial E}{\partial W_{ij}^{R}(k)} \tag{9}$$

$$B_{i}^{L}(k+1) = B_{i}^{L}(k) - \alpha \frac{\partial E}{\partial B_{i}^{L}(k)} \tag{10}$$

$$B_{i}^{R}(k+1) = B_{i}^{R}(k) - \alpha \frac{\partial E}{\partial B_{i}^{R}(k)} \tag{11}$$

Second optimization algorithm used in this work is gradient descent algorithm with momentum. Gradient descent with momentum is an algorithm that increases the gradient descent velocity in the related direction and reduces oscillations [17]. According to (12), a scaled value of the last update amount is used in the current update equation. The momentum term β is generally used as

0.9 or close values. After the calculation, standard weight update procedure is implemented, (See (13)).

$$v(k+1) = \beta v(k) + (1-\beta)\left.\frac{\partial E(W)}{\partial W}\right|_k, \quad W \in \{W^L, W^R, B^L, B^R\} \tag{12}$$

$$W(k+1) = W(k) - \alpha v(k+1) \tag{13}$$

Adaptive Moment Estimation (Adam) [18] is an optimization algorithm that computes adaptive learning rates for each parameter. It stores the previous gradients v_t, like momentum. Also it keeps exponentially decaying average of the previously squared gradients as m_t. At the beginning, v_t and m_t are initialized to zero vectors of appropriate dimensions. In (14) and (15), the mean moment and the uncentered variance moment of the gradients are defined respectively. Also the algorithm includes bias correction steps, which are given in (16) and (17). Then, these variables are used for updating the parameters (See (18)). Default Adam configuration parameters are 0.9 for β_1, 0.999 for β_2, and 10^{-8} for ϵ selected for the training [18].

$$v(k+1) = \beta_1 v(k) + (1-\beta_1)\left.\frac{\partial E(W)}{\partial W}\right|_k, \quad W \in \{W^L, W^R, B^L, B^R\} \tag{14}$$

$$m(k+1) = \beta_2 m(k) + (1-\beta_2)\left.\frac{\partial E(W)}{\partial W}^2\right|_k, \quad W \in \{W^L, W^R, B^L, B^R\} \tag{15}$$

$$v(k+1) = \frac{v(k+1)}{1-\beta_1} \tag{16}$$

$$m(k+1) = \frac{m(k+1)}{1-\beta_2} \tag{17}$$

$$W(k+1) = W(k) - \alpha \frac{v(k+1)}{\sqrt{m(k+1)} + \epsilon} \tag{18}$$

Neural network models can memorize the training data. This causes a high variance problem. To overcome this problem, a remedy is to try regularization techniques. Adding regularization generally helps to prevent overfitting. In this study, we applied two different regularization techniques to each one of three different training algorithms. In addition, all 3 major models are trained without any regularization too.

Dropout [19] is a technique that prevents the model from overfitting. Basically, the dropout technique randomly closes the nodes in the hidden layer according to a specified rate, which prevents to overfitting to the training dataset.

L2 regularization is another approach to prevent the model from overtraining. It affects weights and cost function at the same time. L2 regularization method uses lambda parameter to penalize the weights and the cost function.

4 Test and Analysis

During the tests, an ANN with 2 hidden layers is used in 9 different classification models. As a result of the PCA implementation in data preprocessing step, the number of the features are reduced to the 60 principal components. Therefore, the weight matrix dimension in the hidden layer is in a generic form size (N × 60). The number of neurons in hidden layer is expressed by the N parameter in the general notation because its optimal values are found in the further tests. Lastly, in the output layer, 36 neurons are used for the recognition for 20 falls and 16 daily activities, so their generic weight matrix size is (36 × N).

All dataset contains 3060 observations. Each of the 36 movements is homogeneously distributed on the dataset and they are iterated 85 times. In order to obtain a good performance for the test set, the training steps are started directly with 10-fold cross validation. This means that the model is trained with 2754 observations and tested with 306 observations in every cycle. Also, each major algorithm is executed without regularization as well.

Firstly, we try to define optimal number of hidden layer units in major learning algorithms. Each trial is worked with 10-fold cross validation approach. Basically, the whole dataset is divided into 10 units and 9 of them are used for training and 1 of them used for testing the model. This scheme is continuously repeated for changing the test dataset.

Table 3. Mean accuracy score for the selections of optimization algorithm and number of hidden layer neurons.

Number of hidden layer neurons								
Optimization algorithm	10	20	40	60	100	150	200	250
Gradient descent	67.7%	87.7%	**90.5%**	91.0%	91.1%	91.1%	91.2%	91.3%
G.D. with momentum	71.8%	88.2%	90.4%	**91.7%**	91.6%	91.9%	91.7%	91.6%
Adam	89.4%	93.2%	**94.1%**	94.2%	94.1%	94.3%	94.4%	94.6%

As shown in the Table 3, number of neurons in the hidden layer is defined for three cases. For the gradient descent algorithm, gradient descent with momentum and Adam optimization algorithm, the number of neurons are selected 40, 60 and 40 respectively. The data listed in the table suggests better accuracy scores yet those cases come up with unaffordable computational burden.

In order to find the optimal batch sizes for the feedforward and backpropagation steps, we try 2754, 2048, 1024, 512, 256, 128, 64 batch sizes for the optimization algorithms. During these attempts, 10-fold cross validation is used and Table 4 shows the optimal learning rates and their batch sizes for each one of the algorithms considered. 3000 iterations are found to be sufficient in both parameter searches.

After the determining of the hidden layer units and optimal batch sizes for the three major models, they are analyzed for improving the test set accuracy score. In order to accomplish that, dropout and L2 cost regularization techniques are

Table 4. Mean accuracy scores for selections of optimization algorithms, learning rates and optimal batch sizes.

Optimization algorithm	Learning rate	Batch size	Accuracy score of test set
Gradient descent	0.7	2754	90.56%
G.D. with momentum	0.7	2754	91.83%
Adam	0.07	2754	**93.63%**

used for all discrete models. The selected hyperparameters of the regularization are defined during the experiments. Thus, λ is set 0.0009 for the L2 regularization and ratio for dropout is selected 0.85.

According to the result given Table 5, all models' mean accuracy performances on the test set are improved after the regularization techniques.

Table 5. Performance comparison of the nine models.

Optimization algorithm with regularization technique	Mean accuracy score of test set
Gradient descent without regularization	90.56%
Gradient descent with dropout	92.29%
Gradient descent with L2 cost regularization	90.70%
G.D. momentum without regularization	91.83%
G.D. momentum with dropout	92.16%
G.D. momentum with L2 cost regularization	93.46%
Adam without regularization	93.63%
Adam with dropout	**94.58%**
Adam with L2 cost regularization	94.25%

To visualize and analyze the performance of the classified daily activities and falls separately, a normalized confusion matrix is illustrated in Fig. 4. The matrix is created by using the model that executed with Adam optimizer with dropout via optimized hyperparameters. The vast majority of activity misclassifications take place in the front-lying fall which is numbered by 1st. Additionally, except for 7 and 9 fall types, other falls are classified correctly. Therefore, it can be said that the false negative alarm is kept at the minimum level.

The proposed model classified some daily activities as front-lying fall as seen in Fig. 4. When the definitions of the 18th, 21th, 22nd, 24th and 29th movements are examined according to the Table 1, these have similar action patterns with falling forward. Moreover, the rate of these misclassifications does not exceed 30%.

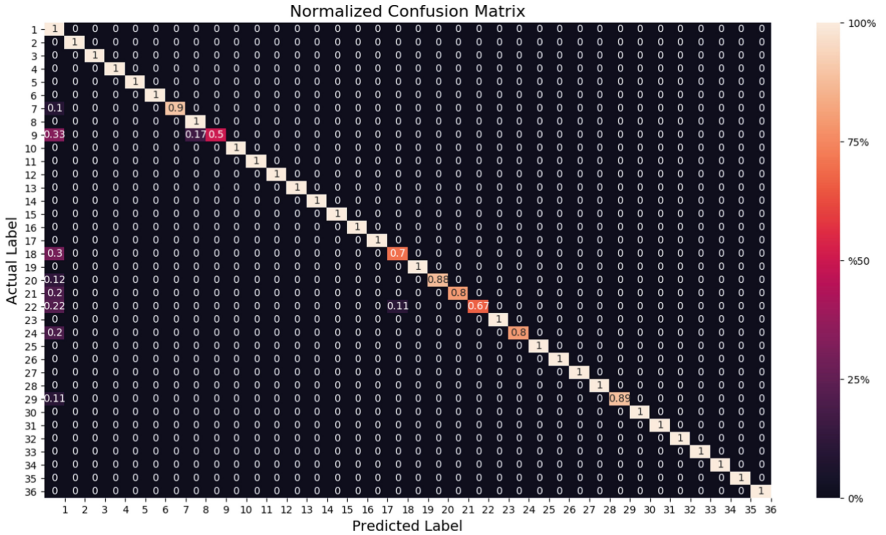

Fig. 4. The normalized confusion matrix is formed using the samples of the test set. The rows in the matrix show the actual class of the sample and the columns represent the class assigned by the classifier. The main diagonal shows the percentage of the movement types are predicted correctly. The numbers on the edges represent the falls and daily activities as explained in Table 1

To enable early intervention of IPAs, the real-time classification is very important. Proposed model is capable to predict a movement approximately at 0,000006 s. Time can be maximum 1 s on embedded platforms with data acquiring from sensors, data preprocessing and classification steps.

All training experiments are deployed on a Ubuntu 16.04 LTS linux-based computer with 64-bit Intel Core i7-3630QM processor, CPU @ 2.40 GHz × 4 for the CPU measurements and 8 GB of RAM.

5 Discussion

When we evaluate the results of our model with reference articles, we can say that the developed model is resulted in a slightly lower accuracy. One of the main reasons for that the proposed model aims to classify more falls and daily activities than other studies. When the number of classification capabilities of the model increases, a decreasing on the overall accuracy is also expected. For instance, while the studies [12] and [13] only perform binary fall recognition, thus they achieved highly accurate results. However, [10] and [11] classify 6 different movements with 96.2% and 98% accuracy scores on the test set respectively. As a result, When the model is evaluated according to these results, it has 6-times wider movement classification spectrum than the existing studies with 94.58% accuracy score.

6 Conclusion

In this paper, we focus on classification of 36 different movement types in the domain of falls and daily activities. Three different learning algorithms and two different regularization techniques are compared and all of the 10-fold cross validation mean accuracy scores are observed above 90%. The most successful classifier obtained at the end of the study is the Adam optimization technique with dropout, which produced 94.58% mean test accuracy. When the result is compared with exist studies, the fact that 16 daily human activities and 20 falls can accurately be defined at such a rate is remarkable for sensory data and the scheme it is obtained. ANN based classifiers are found versatile in processing the information considered here and in generating useful outputs. In the future, smart phones are aimed to be used for data acquisition and classification.

References

1. Sankarasubramaniam, Y., Akyildiz, I.F., Su, W., Cayirci, E.: A survey on sensor networks. IEEE Commun. Mag. **40**(8), 102–114 (2002)
2. Aggarwal, C.C.: Managing and Mining Sensor Data. Springer, New York (2009)
3. Sherrington, C., Menz, H.B., Close, J.C., Lord, S.R.: Falls in Older People: Risk Factors and Strategies for Prevention. Cambridge University Press, Cambridge (2007)
4. Gillespie, L.D., et al.: Interventions for preventing falls in older people living in the community. Cochrane Database Syst. Rev. (9) (2012). ISSN 1465-1858. https://doi.org/10.1002/14651858.CD007146.pub3
5. Crouter, S., Bassett, D., Freedson, P., Staudenmayer, J., Pober, D.: An artificial neural network to estimate physical activity energy expenditure and identify physical activity type from an accelerometer. J. Appl. Physiol. **107**(8), 1300–1307 (2009)
6. King, R., Yang, G.Z., Atallah, L., Lo, B.: Sensor placement for activity detection using wearable accelerometers. In: Proceedings of International Conference on Body Sensor Networks, pp. 24–29 (2010)
7. Oliveira, N.H., et al.: Relationship between pulmonary function and physical activity in daily life in patients with COPD. Respir. Med. **102**(8), 1203–1207 (2008)
8. Rodrigues, J., Silva, B., Casal, J., Saleem, K., Denisov, V., Santos, J.: An iot-based mobile gateway for intelligent personal assistants on mobile health environments. J. Netw. Comput. Appl. **71**(8), 1203–1207 (2016)
9. Labrador, M., Delahoz, Y.: Survey on fall detection and fall prevention using wearable and external sensors. Sensors **14**, 19806–19842 (2014)
10. Morgül, Ö., Çatalbaş, B., Çatalbaş, B.: Human activity recognition with different artificial neural network based classifiers. In: 25th Signal Processing and Communications Applications Conference, pp. 1–4 (2017)
11. Conde, I.G., Sobrino, X.A.V., Olivieri, D.N.: Eigenspace-based fall detection and activity recognition from motion templates and machine learning. Expert Syst. Appl. **39**(5), 5935–5945 (2012)
12. Barshan, B., Özdemir, A.T.: Detecting falls with wearable sensors using machine learning techniques. Sensors (Basel, Switzerland) **14**, 10691–10708 (2014)

13. Monteiro, K., Rocha, E., Silva, I., Lynn, T., Leoni, G., Endo, P.: Accelerometer-based human fall detection using convolutional neural networks. Sensors **19**, 1644 (2019)
14. Xsens Technologies B.V.: MTw Awinda User Manual and Technical Documentation (2019). http://www.xsens.com/. Accessed 2 Mar 2019
15. Hastie, T., Tibshirani, R., James, G., Witten, D.: An Introduction to Statistical Learning: With Applications in R. Springer, Heidelberg (2014). https://doi.org/10.1007/978-1-4614-7138-7
16. Ren, S., Sun, J., He, K., Zhang, X.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Washington, DC, USA, pp. 1026–1034. IEEE Computer Society (2015)
17. Qian, N.: On the momentum term in gradient descent learning algorithms. Neural Netw.: Off. J. Int. Neural Netw. Soc. **12**, 145–151 (1999)
18. Ba, J., Kingma, P.: Adam: a method for stochastic optimization. In: International Conference in Learning Representations (2015)
19. Krizhevsky, A., Sutskever, I., Salakhutdinov, R., Srivastava, N., Hinton, G.E.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)