World Academy of Science, Engineering and Technology
International Journal of Mechanical and Materials Engineering
Vol:11, No:3, 2017

# Genetic Algorithm Based Deep Learning Parameters Tuning for Robot Object Recognition and Grasping

Delowar Hossain, Genci Capi

Open Science Index, Mechanical and Materials Engineering Vol:11, No:3, 2017 publications.waset.org/10006712/pdf

***Abstract*—**This paper concerns with the problem of deep learning parameters tuning using a genetic algorithm (GA) in order to improve the performance of deep learning (DL) method. We present a GA based DL method for robot object recognition and grasping. GA is used to optimize the DL parameters in learning procedure in term of the fitness function that is good enough. After finishing the evolution process, we receive the optimal number of DL parameters. To evaluate the performance of our method, we consider the object recognition and robot grasping tasks. Experimental results show that our method is efficient for robot object recognition and grasping.

***Keywords*—**Deep learning, genetic algorithm, object recognition, robot grasping.

## I. Introduction

DL is a vastly growing research field. It tries to mimic the human brain. For example, the feature hierarchies in the human visual cortex represent the objects at the different level of abstraction. The more abstract features go up in the hierarchy; the objects become more visible to the human. DL works in the same way as our human brain organizes ideas in hierarchical fashion. The objective of DL is to bring the machine learning research to Artificial Intelligence.

DL research was started by Hinton et al. [1] in 2006. After then, many researchers are working to improve the performance of DL. DL has many parameters which have influence on performance. For this reason, recently researchers are working to integrate evolutionary programming with DL. Some research works are already presented. The first effort was done by Lamos-Sweeney [2]. He integrated GA with a multilayer DL network for data compression and object classification. He showed that his proposed method enhanced the flexibility and reduced the computational burden of the algorithm. Levy et al. [3] proposed a hybrid approach to integrating GA and deep restricted Boltzmann machines (RBMs) for painter classification problem. They extracted features using generic image processing function and deep RBMs. Tirumala [4] studied the evolutionary computation (EC) implantation possibility with the deep architectures. He argued EC could solve the Deep Neural Networks (DBNs) overfitting problem. David and Greental [5] proposed a GA-assisted method for a deep autoencoder to improve the performance and produced a sparser neural network. Verbancsics and Harguess [6] investigated a neuro-evolution (NE) based DL method. They applied the Hypercube-based NeuroEvolution of Augmenting Topologies (HyperNEAT) to training a feature extractor process in backward propagation learning. Shao et al. [7] developed an evolutionary learning methodology based on multiobjective genetic programming (MOGP) for image classification. It generated domain-adaptive global feature descriptor automatically.

DL has demonstrated state-of-art performance on robotic applications. There have been many works presented for robot real-time object recognition and grasping [8]-[13]. The goal is that robots can learn through the interaction with the environment and the human subjects. It reduces the stress of human subjects in various home style and industrial tasks.

In this paper, we present an evolutionary learning method that combined GA and Deep Belief Neural Networks (DBNNs) for robot object recognition and grasping. This method optimizes the DBNN parameters, such as the number of hidden units, the number of epochs, learning rates and momentum in learning procedures of the hidden layers. It reduces the error rate and network training time of object recognition. The objects are recognized in the different orientation, positions, and lighting conditions using optimized DBNN method, after then the robot picks up the recognized objects and places in a predefined position.

This paper is organized as follows. In Section II, combined GA and DBNN are described. In Section III, GA parameters and results are presented. In Section IV, the experimental results of optimized DBNN for robot object recognition and grasping are shown. In Section V, we conclude and mention the future works.

## II. Combined GA with Deep Belief Neural Network (GADBNN)

The problem with DBNN is that the DBNN parameters need to set before the training begins. Although there are no any fixed rules how to set these parameters, these parameters have influenced on the success of the training. By combining GA with Deep Belief Neural Network (GADBNN), the DBNN parameters are optimized using GA. The flowchart of GADBNN is shown in Fig. 1.

### *A. Structure of Deep Belief Neural Network (DBNN)*

A DBNN is a probabilistic generative model, which is constructed by a stack of RBMs. An RBM consists of a visible layer and a hidden layer, or a hidden layer and another hidden layer. In RBM, the neurons of each layer are completely connected with the neurons of another layer, but the neurons of

Delowar Hossain is with the Graduate School of Science and Engineering for Education, University of Toyama, 3190 Gofuku, Toyama 930-8555, Japan (e-mail: delowar.hossain.38@hosei.ac.jp).

Genci Capi is with the Department of Mechanical Engineering, Hosei University, Tokyo 184-8584, Japan (corresponding author, phone: +81-42-387-6148, fax: +81-42-387-6148, e-mail: capi@hosei.ac.jp).

Open Science Index, Mechanical and Materials Engineering Vol:11, No:3, 2017 publications.waset.org/10006712/pdf

the same layer are not internally connected with each other. RBMs are stacked on the top of each other to build a DBNN. DBNN extracts feature in the hierarchical fashion, where lower level features form higher levels features.

The energy function between visible layer and hidden layer $\{v,h\}$ in RBM is given as follows:

$$E(v,h) = -\sum_{i=1}^{n_v}\sum_{j=1}^{n_h} a_i b_j w_{ij} - \sum_{i=1}^{n_v} a_i v_i - \sum_{j=1}^{n_h} b_j h_j \quad (1)$$

where $n_v$ is the number of visible units, $n_h$ is the number of hidden units, $a$ is the bias term for visible units, $b$ is the bias term for hidden units, $w$ is the weights between visible and hidden units, $v$ is the visible units with $v \in \{0,1\}$, and $h$ is the hidden units with $h \in \{0,1\}$.

The structure of our DBNN is shown in Fig. 2. It consists of a visible layer, three hidden layers, and an output layer. The optimal number of hidden units in the three layers was found by applying GA [14]. Visible units are set to the activation probabilities; on the other hand, hidden units are set to the binary values. We use two types of sampling method, such as Contrastive Divergence (CD) and Persistent Contrastive Divergence (PCD). The first hidden layer is sampled by PCD because PCD explores the entire searching domain for the input features. It is much better in representing the log-likelihood of the pixels. The second and third hidden layers are sampled using CD because CD explores the better near the input images. CD is not aware of spurious modes of the input images and better for extracting features. For this reason, we combine CD and PCD sampling methods and it outperforms for object recognition purpose.

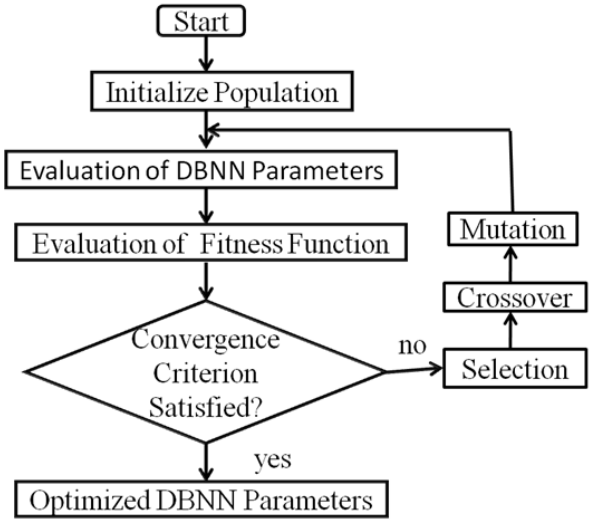



Fig. 1 Flowchart of combined GA and DBNN

Backpropagation error derivatives are used to reduce the discrepancies between the original features and its reconstruction for fine-tuning weights in order to better object recognition. It refers the whole procedure encompassing the calculation of the gradient and uses in stochastic gradient descent. We use softmax function for classifying objects. Backpropagation will terminate, if it satisfies one of the following conditions: (1) reach to best performance, i.e. mean square error (MSE), (2) reach to maximum epoch number, i.e. 200, (3) reach to minimum gradient value, or (4) reach to maximum validation checks, i.e. 6.

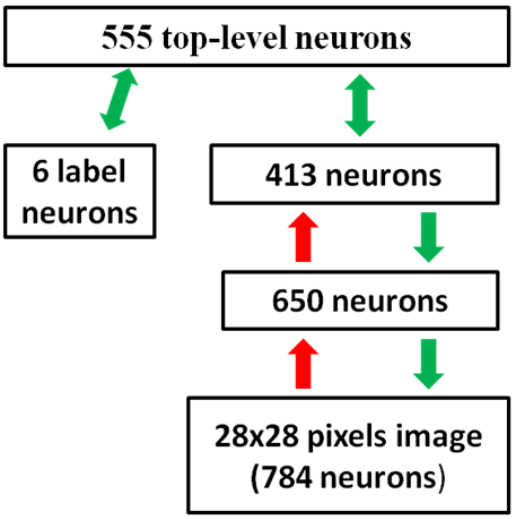



Fig. 2 Structure of our DBNN

### *A. Genetic Algorithm (GA)*

GA is the heuristic search algorithm based on the natural evolutionary ideas. It is the best way to solve a problem that has little information to know. It is a very general algorithm and suitable for any search space. GA takes advantages of historical information for searching of better solution among the search space. The basic technique is to design an evolution process to simulate the survival of the fittest among individuals for solving a problem in generations.

The evolution process of DBNN is shown in Fig. 1. In GADBNN, the GA is used to find the optimal number of DBNN parameters, such as the number of hidden units, the number of epochs, learning rates and momentum in learning procedures for hidden layers. DBNN information is encoded in the genome of the GA. Initially, the random number of individuals is generated. Then, the number of DBNN parameters is evaluated and ranked. After then, the fitness function is evaluated. If the convergence criteria are not satisfied, then crossover and mutation creates new individuals and replaces the worst members of population. If the convergence criteria are satisfied, then evolution process is terminated, and the optimal number of DBNN parameters is generated.

## III. GA Parameters

In this paper, we are deployed a real-valued GA [14], which generates better solutions with respect to the quality of the solution. It uses real values as parameters of the chromosome in populations. It is implemented with the selection, crossover and

Open Science Index, Mechanical and Materials Engineering Vol:11, No:3, 2017 publications.waset.org/10006712/pdf

mutation operators.

The GA function and parameters are mentioned in Table I. Population is divided into four subpopulations, and each subpopulation uses different mutation rates. Migration is selected based on the number and size of the subpopulations. Crossover probability is same for all subpopulations. The total number of individuals is 100, and the maximum number of generation is 30. The best subpopulation has received the resources, and worst individuals are removed from less successful generations.

TABLE I
GA FUNCTIONS AND PARAMETERS

| Function Name | Parameters |
|---|---|
| Number of subpopulations | 4 |
| Initial number of individuals (subpopulation) | 25, 25, 25, 25 |
| Crossover probability | 0.8 |
| Mutation rate (subpopulation) | 0.1, 0.03, 0.01, 0.003 |
| Isolation time | 10 generations |
| Migration rate | 10% |
| Results on screen | Every 1 generation |
| Competition rate | 10% |
| Termination | 30 generations |

The fitness function is defined as to minimize the error rate and minimize the network training time in order to optimize the number of hidden units, the number of epochs, learning rates and momentum in hidden layers. The fitness function used in our implementation is shown as follows:

$$Fitness = 100 \times (E_{BBP} + E_{ABP}) + \frac{(T_{BBP} + T_{DBP})}{40} \quad (2)$$

where $E_{BBP}$ is the total number of misclassification divided by the total number of test data before backpropagation, $E_{ABP}$ is the total number of misclassification divided by total number of test data after backpropagation, $T_{BBP}$ is the network training time in second before fine-tuned operation using backpropagation, $T_{DBP}$ is the network training time in second during fine-tuned operation using backpropagation operation.

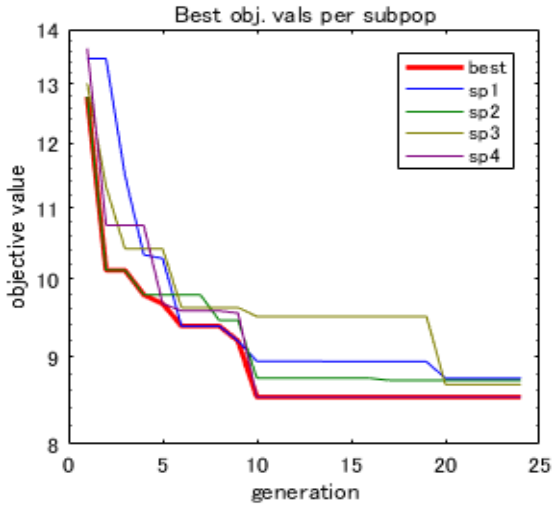



Fig. 3 Best objective values per subpopulation

## IV. Experimental Results

In this section, we describe the experiment and results of our approach. We divide the experimental results into two sections: (1) Experimental result for GA based Deep Belief Neural Network (GADBNN), (2) Experimental result for Optimized DBNN for robot object recognition and grasping.

### *A. Experimental Results for GADBNN*

In order to optimize DBNN parameters, we build a database, which consists of 1200 train images of six different types of objects (200 images of each object) and 600 test images of 100 images of each object. This training and test images were taken randomly in different oriental, positions, and lighting conditions in our experimental environment to make our system robust. In our dataset, all images are grayscale images.

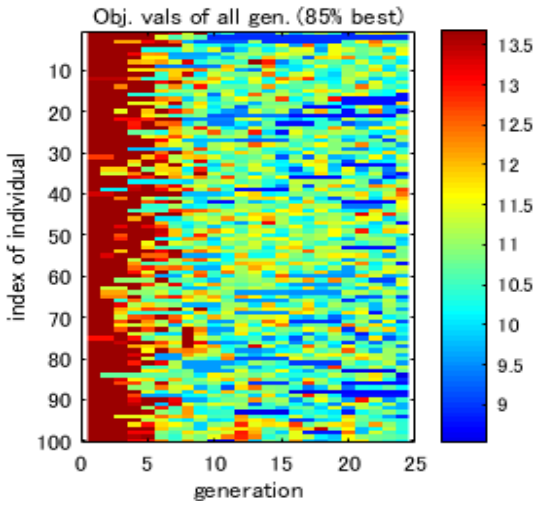



Fig. 4 Objective values of all individuals of generations

In order to evaluate GADBNN performance, we plot the best objective value per subpopulation in Fig. 3. Four subpopulations (sp1, sp2, sp3, sp4 where sps are the subpopulations) are mentioned with four different colors. The best object value is mentioned by bold red color. In addition, objective values of individuals through all generation during evolution process are shown in Fig. 4. The fitness value was started from 13.6250. In first four generations, the strategy applied in subpopulation 2 is successful. On the fifth generation, strategy 4 is successful. From six to nine generations, strategy 1 is successful. The individual's convergence is most successful in the tenth generation and objective value is 8.52528. At the end of the optimization (tenth generation), the strategy applied on subpopulation 4 is the most successful. The individual's convergences were terminated on 24th generation.

### *B. Experimental Results for Object Recognition and Robot Grasping*

In order to demonstrate the performance of optimized DBNN parameters, we consider the object recognition and robot grasping tasks. For this purpose, we build an image dataset that is consisted of 1200 images (200 images of each object) of six

Open Science Index, Mechanical and Materials Engineering Vol:11, No:3, 2017 publications.waset.org/10006712/pdf

robot graspable objects in different orientations, positions, and lighting conditions in our experimental environment.

1. Object Recognition Results

When a user requests for an object by clicking on Graphical User Interface (GUI), then a snapshot is taken by USB camera (shown in Fig. 6). This snapshot is converted to the grayscale image. We apply a morphological structuring element operation, and all existing objects in the environment are extracted and separated based on the center of the objects of the size of 28x28 pixels. In the visible layer, 784 neurons are used as input. From optimized DBNN, we found 650, 413, and 555 neurons of hidden units, 100, 88, and 184 neurons of epochs in three hidden layers. Learning rate for first hidden layer was 0.084859 and 0.1664 for second and third hidden layers. Momentum values of first five epochs in each hidden layer are 0.28644, 0.28088, 0.15208, 0.55015, and 0.0258 in order to make learning procedures more efficient. Momentum value for the remaining epochs in each hidden layer is 0. After passing through the three hidden layers, DBNN generates six probability values as output, because we trained optimized DBNN with six different types of object. Each probability belongs to each object. The highest probability value is considered the recognized object. If the highest probability value is smaller than the threshold value (0.7), then we consider that the requested object does not exist in the experimental environment. For example, the blue screwdriver is used as input of DBNN, as outputs, DBNN generates six probability values, such as 0.0001, 0.0028, 0.0037, 0.9999, 0.0000, and 0.0000 (shown in Fig. 5). From this result, the maximum probability is 0.9999, which belongs to the fourth object that we defined as the blue screwdriver. At the same way, other objects can be recognized using optimized DBNN method.

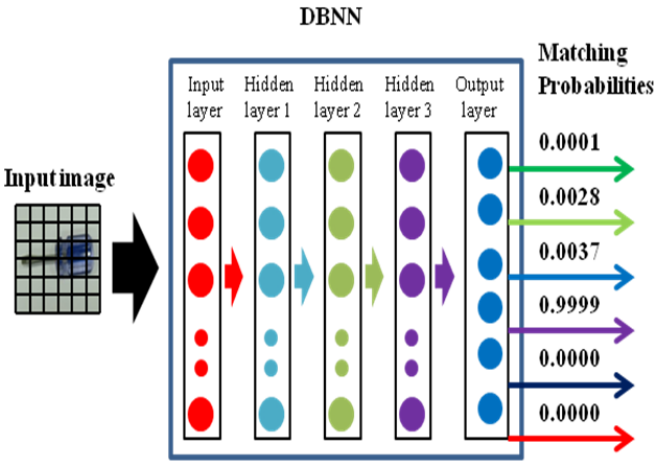



Fig. 5 Object recognition process

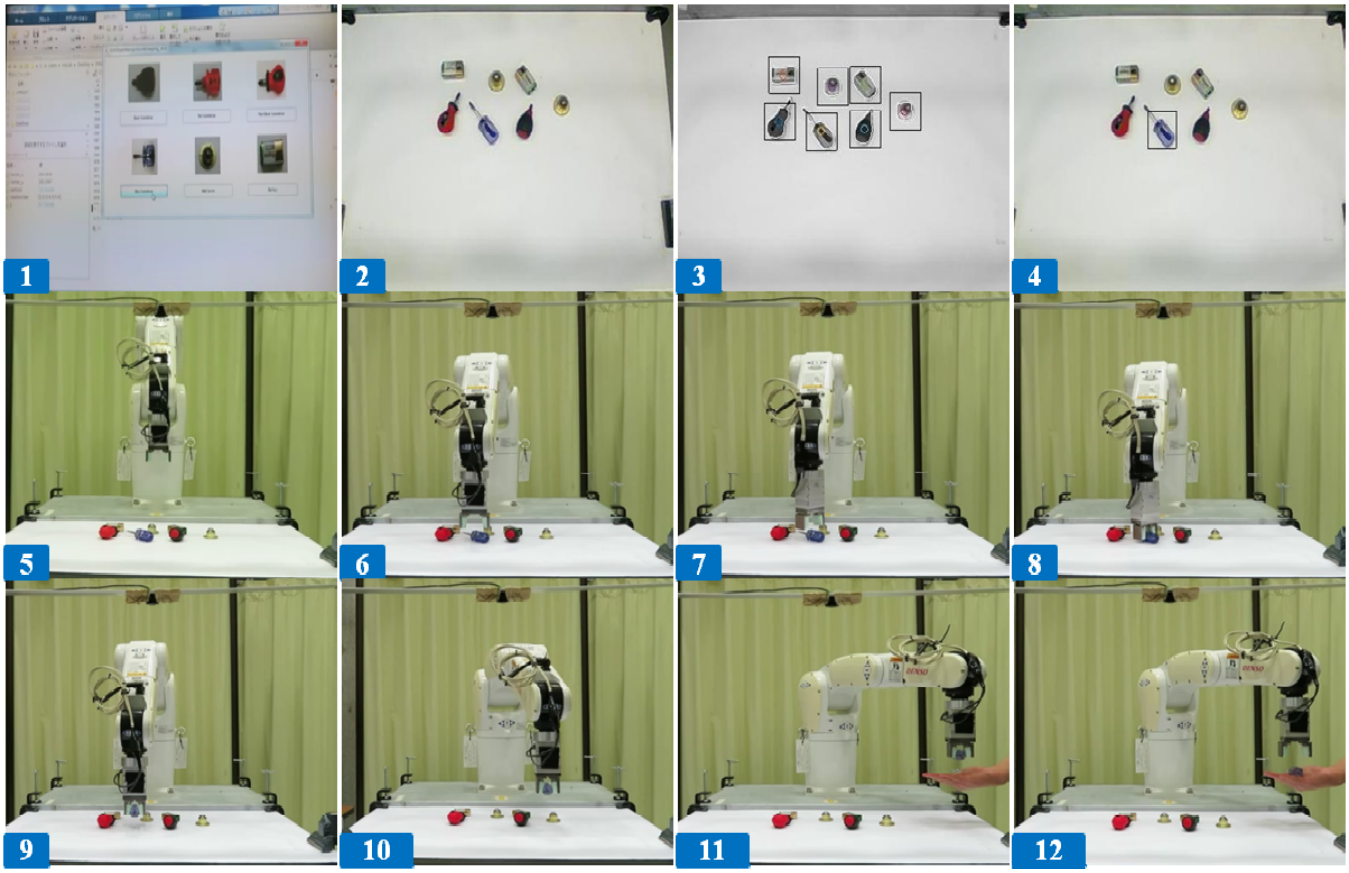



Fig. 6 Snapshots of robot object recognition and grasping using optimized DBNN

Open Science Index, Mechanical and Materials Engineering Vol:11, No:3, 2017 publications.waset.org/10006712/pdf

2. Robot Grasping Results

For robot grasping purpose, we used a Programmable Universal Machine for Assembly (PUMA) robot from Denso Corporation. It has a gripper that can grasp any objects of the maximum width of 95 mm. When a user requests for a specific object, then the requested object is recognized using optimized DBNN method. The robot finds the grasping position based on the recognized object. The robot generates a motion trajectory from the initial position to the object grasping position. After reaching to the object grasping position, the robot adjusts its gripper orientation based on the object orientation in order to grasp objects in different orientations. After grasping the object, the robot generates another motion from object position to the predefined object placing position. After placing the object, the robot returns to the initial position and waits for next requests from the user. The snapshots of robot object recognition and grasping process are shown in Fig. 6.

## V. Conclusions and Future Works

The goal of this paper is to propose a method to optimize DL parameters using evolutionary algorithms. Our proposed method optimized the DL parameters, such as the number of hidden units, the number of epochs, learning rates and momentum of learning procedures in hidden layers. The error rates and network training time were minimized. We evaluated our optimized method on real-time object recognition and robot grasping process. The results showed that our optimized method outperformed at the assign tasks.

In future, we will integrate Multi-objective Evolutionary Algorithms (MOEA) because the coefficients of single object evolutionary algorithms in the objective function are very difficult to determine. In addition, we want to consider the scaling factors of the objects.

## References

[1] G. Hinton, S. Osindero, and Teh, Y.-W, “A fast learning algorithm for deep belief nets”, Neural Computation, vol. 18, no. 7, pp. 1527-1554, 2006.
[2] J. Lamos-Sweeney, “Deep learning using genetic algorithms”, Master’s Thesis, Rochester Institute of Technology, NY, USA, 2012.
[3] E. Levy, O. E. David, and N. S. Netanyahu, “Genetic algorithms and deep learning for automatic painter classification”, Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation, 2014, pp. 1143-1150.
[4] S. S. Tirumala, "Implementation of Evolutionary Algorithms for Deep Architectures", AIC, pp. 164-171 2014.
[5] O. E. David, and I. Greental, "Genetic algorithms for evolving deep neural networks", In Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation, pp. 1451-1452. ACM, 2014.
[6] P. Verbancsics, and Josh Harguess, "Generative neuroevolution for deep learning", arXiv preprint arXiv:1312.5355 (2013).
[7] L. Shao, L. Liu, and X. Li, “ Feature learning for image classification via multiobjective genetic programming”, IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 7, pp.1359-1371, 2014.
[8] I. Lenz, H. Lee, and A. Saxena, “Deep Learning for Detecting Robotic Grasps”, International Journal of Robotics Research, vol. 34, no. 4-5, pp. 705-724, 2015.
[9] L. Pinto, and Gupta, “Supersizing Self-supervision: Learning to Grasp from 50K Tries and 700 Robot Hours”, arXiv:1509.06825 (cs.LG), 2015.
[10] D. Hossain, and G. Capi, “Application of Deep Belief Neural Network for Robot Object Recognition and Grasping”, The 2nd IEEJ International Workshop on Sensing, Actuation, and Motion Control (SAMCON 2016), Tokyo, Japan, March 2016.
[11] D. Hossain, G. Capi, and M. Jindai, “Object Recognition and Robot Grasping: A Deep Learning based Approach”, The 34th Annual Conference of the Robotics Society of Japan (RSJ 2016), Yamagata, Japan, September 2016.
[12] J. Redmon, and A. Angelova, “Real-Time Grasp Detection Using Convolutional Neural Networks”, arXiv:1412.3128, 2015.
[13] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, D. “Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection”, arXiv:1603.02199, 2016.
[14] G. Capi, and K. Doya, “Evolution of recurrent neural controllers using an extended parallel genetic algorithm”, Robotics and Autonomous System, vol. 52, no. 2-3, pp. 148-159, 2005.

**Delowar Hossain** is working toward the Ph.D. degree at Graduate School of Science and Engineering for Education, University of Toyama, Japan, also working as visiting researcher at Hosei University, Japan. He received the B.Sc. and M.Sc. degrees in Computer Science & Engineering from University of Rajshahi, Rajshahi, Bangladesh, in 2010 and 2012, respectively. He was a Lecturer at the Department of Computer Science and Engineering, Dhaka International University, Dhaka, Bangladesh from 2012 to 2014.

His research interests include the industrial robot, deep learning, artificial intelligence, computer vision, image processing, intelligent robotics, learning, and evolution.

**Genci Capi** received the B.E. degree from Polytechnic University of Tirana, in 1993 and the Ph.D. degree from Yamagata University, in 2002. He was a Researcher at the Department of Computational Neurobiology, ATR Institute from 2002 to 2004. In 2004, he joined the Department of System Management, Fukuoka Institute of Technology, as an Assistant Professor, and in 2006, he was promoted to Associate Professor. In 2010, he was joined as a Professor at the Department of Electrical and Electronic Systems Engineering, University of Toyama, Toyama, Japan. He is currently a Professor at the Department of Mechanical Engineering, Hosei University, Tokyo, Japan.

His research interests include intelligent robots, BMI, multi-robot systems, humanoid robots, learning, and evolution.