Learning Reaching Tasks Using an Arm Robot Equipped with MACMSA

Learning Reaching Tasks Using an Arm Robot Equipped with MACMSA

Motohiro Akikawa and Masayuki Yamamura

School of Computing, Tokyo Institute of Technology, J2 bldg. room1706, 4259 Nagatsuta-cho, Midori-ku, Yokohama, JAPAN

Email: my@c.titech.ac.jp

Abstract. In recent years, models that integrate multimodal information to control robots have been actively developed. Memorizing and Associating Converted Multimodal Signal Architecture (MACMSA) was proposed to integrate multimodal information obtained from robots with Hopfield networks as associators and independent feed-forward neural networks as encoders and decoders. The performance of MACMSA has thus far been investigated only using pseudo-data. Notably, MACMSA exhibits high resistance to noise. However, it cannot generate signals for robot control. The purpose of this study was to improve MACMSA to generate signals for robot control and optimize it using real data on reaching tasks. The results of the generated control signals on a real machine are presented to demonstrate that the improved model can be effectively used in a real environment. The results also show that the proposed model can perform well with real data.

1. Introduction

Studies on robot controls to interact with humans have been conducted since a long time [1,2]; however, no robot capable of interacting with humans in any given situation has yet been developed. In the early stages of robot controls, the methods using which developers embedded programs to control robots were adapted [3]. These methods could not solve the well-discussed problems, such as those associated with complex actions and the frame problem, although they often occur in the day-to-day functioning of industries.

In recent years, techniques to control robots using deep learning have been developed to increase the accuracy of recognition [4,5]. Some robot control systems have also been developed to process multimodal information to obtain higher recognition [6-10]. Systems that process multimodal information face two challenges [11]. The first is the increase in the computational cost. Simply integrating multimodal information will bloat the network and increase computational costs [12]. The second factor is the resistance to environmental noise. A slight change in the position of the light source can cause the robot to lose control [13].

Memorizing and Associating Converted Multimodal Signal Architecture (MACMSA) [11], which is used for integrating multimodal information obtained from robots by adopting the Hopfield network as an associator and independent feed-forward neural networks as encoders and decoders, was proposed to resolve the two challenges. MACMSA uses independent encoders for each modal, which contributes to reducing network bloats even when the number of modals increases. Because the Hopfield network, whose ability is strong noise resistance, is adapted as the associator at which multimodal information is integrated, MACMSA recalls the correct output from an input exceeding 30% noise.

MACMSA lacks two important features that are necessary to ensure the efficiency of a robot control system. First, the performance of the model was evaluated using only pseudo-data, not real data. In [11], the pseudo-data that were generated had a normal distribution. However, real data obtained from robots are not always normally distributed. Thus, the performance must also be evaluated using real data for a complete verification of the system's performance. Second, it is not possible to generate control signals for robot control. Generation of control signals is the most important function for controlling robots. Therefore, every robot control system must have this function.

The current study aimed to address the aforementioned limitations of MACMA by enabling the function of generating control signals and conducting learning experiments using real data on reaching tasks. A key point was to increase the number of neurons in the Hopfield network to store and recall current states and control signals. Reaching tasks are simple benchmarks in which the controlled robots reach the arms to the goal. The benchmarks are efficient for evaluating the performance of a controlled robot with real data. We also present the obtain results of the generated control signals on a real arm robot to show that the proposed model can perform in a real environment.

2. Material & Method

2.1. Outline of the Proposed Model

The most crucial difference between MACMSA and the model proposed in this study lies in the number of neurons that are present in the Hopfield network. The model proposed in this study uses a relatively larger network and an increased number of neurons to generate the control signals. The associator stores and recalls the current states and control signals in the Hopfield network. An outline of the proposed model is shown in Figure 1. The model consists of four three-layer feed-forward neural networks and one Hopfield network. Three feed-forward neural networks are used as the encoders to convert real data into binary values, which are then to processed by the Hopfield network. The Hopfield network functions as an associator that stores the converted and concatenated multimodal signals and recalls the control signals. The feed-forward neural network connected to a part of the associator performs as a decoder that converts the control signal, which is composed of binary values, into the real values for the control signal that is input to the actuator. In this study, images, sounds, and actuator positions at time *t* are given as inputs, and an actuator position that should be at time *t+1* is output as the control signal.

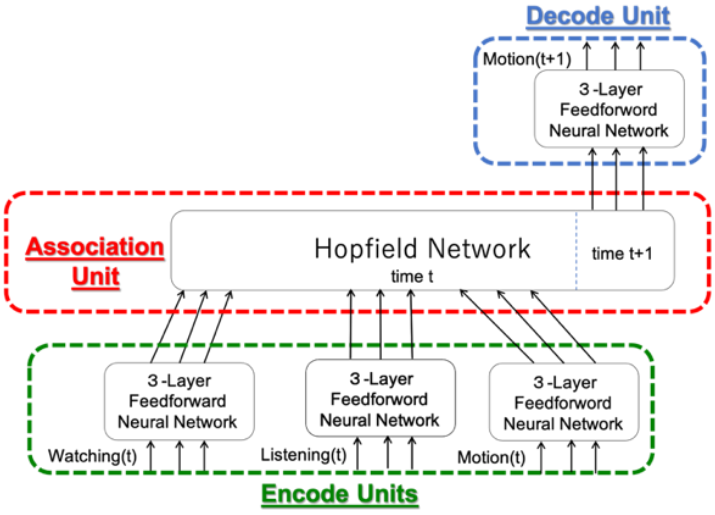

Figure 1. Outline of the proposed model.

The entire system can be divided in three steps. The first involves the optimization of the autoencoders. The second involves storing patterns into the Hopfield network. In the third step, the decoder is tuned.

2.1.1. Optimization of the Autoencoders. In MACMSA, all the activation functions are Sigmoid functions. However, in our study, we used a hyperbolic tangent (Equation (1)) as the activation function of the third layer, which is the output layer of an encoder. Here, x denotes the data and α denotes the gain.

$$Tanh(\boldsymbol{x}) = \frac{e^{\alpha x} - e^{-\alpha x}}{e^{\alpha x} + e^{-\alpha x}} \cdots \quad \cdots\cdots\cdots (1)$$

In MACMSA, it is necessary to replace the value of 0 with -1 during the inference process because the value of the Sigmoid function lies in the range [0,1]. By using a hyperbolic tangent whose output ranges between [-1,1], in the third layer, the output from the encoder can be input directly to the associator. In MACMSA, the gain in the Sigmoid function of the third layer is gradually increased during the learning process, so that the final value is close to binary. Likewise, in this study, the gain, which is α in Equation (1), is increased from *ginitial* to *greach* at each *step* to make the outputs from encoders as close to binaries as possible. The flowchart of the aforementioned procedure is presented in Figure 2.

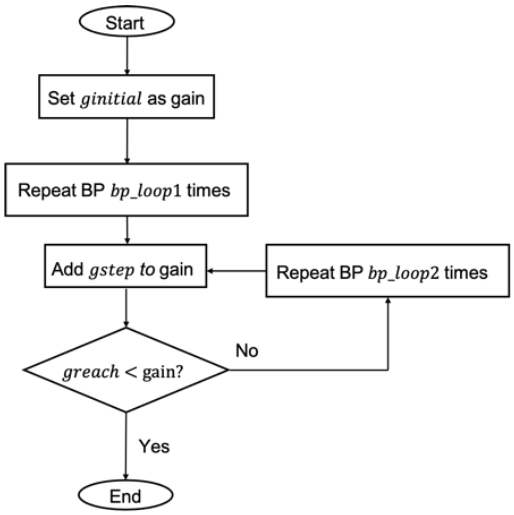

Figure 2. Flowchart of the procedure for optimizing the autoencoders.

Apart from the third layer, the Sigmoid function with gain set to 1 was used as the activation function. The encoder and decoder for each mode were optimized as a five-layer autoencoder. The first three layers, including the input layer, were used as the encoders. The second three layers were used as the decoders. In other words, the encoder and decoder shared a single hidden layer [11]. Note that, in the current study, although all the modes of the encoders and decoders were optimized as autoencoders, and the decoder was used for the actuator positions. The optimizer used was Adam [14]. Notably, the loss function is the sum of squares added to the Kullbacker divergence (Equation (2)) [11].

$$E(w) = \sum_n \|\boldsymbol{y}_n - \boldsymbol{t}_n\|^2 + \beta \sum_j KL(\rho||\widehat{\rho_j}) \quad (2)$$

Here, $\boldsymbol{y}_n$ is the output, $\boldsymbol{t}_n$ is the label, n is the index of the data, β is a hyper-parameter, ρ is a target value of sparse, $\widehat{\rho_j}$ is the measured value of sparse, and j is the index of the neuron.

2.1.2 Storing Patterns into The Hopfield Network. In this study, we assume that the training data are time-series data. The 0th data point is the initial state of all the modal information (time t=0), and the next data point is the state of time, t=1. All the training data were encoded by each optimized encoder. Subsequently, all the encoded training data were replaced with {-1,1} by the threshold, which was set to 0 (Figure 3). This was done to ensure that the output from the encoder is approximately binary, although not exactly binary. During the inference process, replacement is not performed, and a pattern that is concatenated with approximate binary values is given to the associator as input. The stored patterns are composed of two parts of time, t and $t+1$, which forms the control signal for the robot; t is combined with the image, sound, and actuator position corresponding to time, and $t+1$ is the state of $t+1$ of the actuator position. In other words, if the pattern encoded by the encoder for the images is W[i], the pattern encoded by the encoder for the sound is L[i], and the pattern encoded by the encoder for the actuator position is M[i], a pattern that is stored as a single pattern in the Hopfield network is a vector that is concatenated W[i], L[i], M[i], and M[i+1]. i is index of the data.

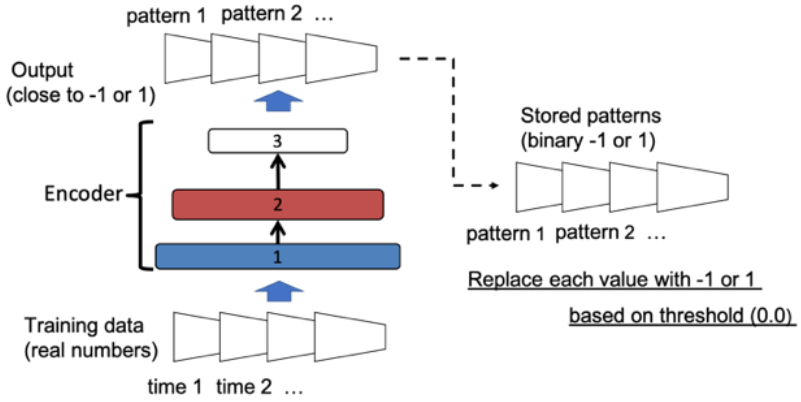

Figure 3. Replacement of patterns.

2.1.3. Tuning the Decoder. In this study, the second three layers of the autoencoder for the actuator position served as the decoder. In the optimization process of the autoencoder, the input of the hidden layer, which is the input layer of the decoder, are real numbers. However, the recalled result from the associator is binary. Thus, errors occur between the optimization and inference processes. To reduce the number of errors, the decoder is tuned with the weight of the second three layers of the optimized autoencoder as the initial weight. The training data are values in the range {-1,1}, and the labels are the actuator positions corresponding to the data index. The loss function is the sum of squares, and the number of iterations of backpropagation is *bp_loop1*, which is used as a parameter to optimize the autoencoder for the actuator position.

2.2. Data Flow in the Inference Process

In the inference process, the system outputs the control signal using the following three steps. First, for each mode, the system encodes the input such that it is in the range [-1,1], using an encoder. Second, the patterns of all the modes of time t are concatenated. A part of the pattern that becomes the state of time $t+1$ is filled with 0s. The concatenated pattern is given as the initial state of the associator. The associator recalls the entire stored pattern, including the state at time $t+1$. Third, the state of time $t+1$ is cut out from the entire recalled pattern and converted to [0,1] using the decoder for the actuator position.

3. Experiment

In this study, nine datasets consisting of images, sounds, and actuator positions were used. A web camera (Sanwa Supply, 400-CAM083) was used to obtain the images. A microphone, which was the built-in

microphone of the web camera, was used to obtain the sound. The product information of the web camera is available at [15]. The actuators were a KXR-A5 arm-type Ver.2. KXR-A5 consists of five servo motors connected to an arm robot. The product information of KXR-5A is available at [16]. The experimental equipment is shown in Figure 4.

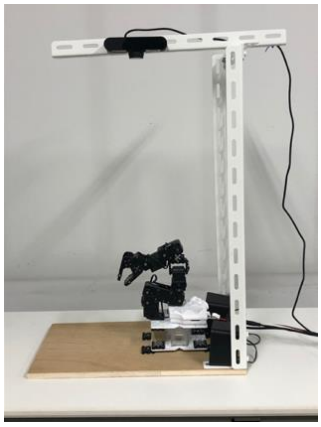

Figure 4. Overview of the experimental equipment.

The robot stretches the arm in the center, right, and left directions. In each direction, the arm passes through two points. The actuator positions are obtained from five servomotors and each such position is defined as an integer value in the range [3500,10500]. The image is resized to 600 × 750 pixels from the original image to obtain the region of interest. Subsequently, it is converted into 8-bit grayscale and reduced to 20 × 25 pixels. In the optimization process, three data points, which denote the initial states of the images and actuator positions, lie in the same situation. Thus, three data points are merged into one and trained as seven data points. All the images that were used as training data in the experiment are shown in Figure 5.

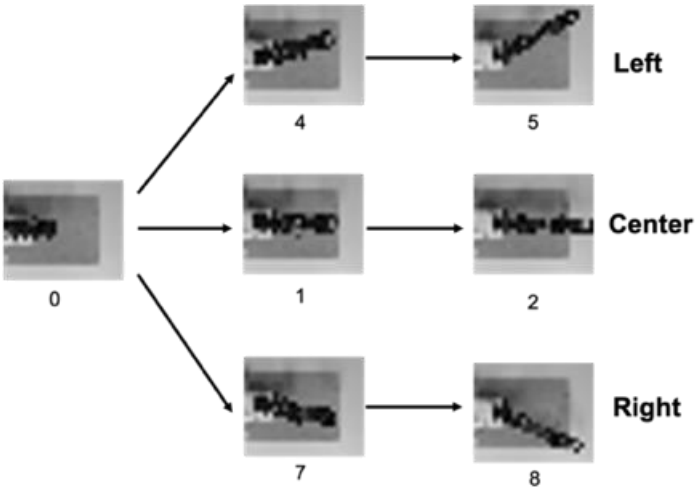

Figure 5. Training data comprising pictures and the indexes of data.

Nine sounds (cymbal, ring bell, snare drum, fingerstyle bass, marimba, harp, cathedral organ, koto, and trumpet) were played by two speakers behind the arm robot. Simultaneously, these sounds were recorded and Fourier transformation was applied. The recording length was set to 1 s. The quantization bit was 16 bits. Frequency is at 16 kHz. For the Fourier transform, the first approximately 0.08 seconds is deleted and the next 500 points are sampled. All the mode data were normalized using Equation 3

along the dimension, and the value range was set to [0,1].

$$x' = \frac{x - min\,(x)}{max(x - min\,(x))} \ldots\ldots\ldots \quad \ldots (3)$$

x' denote the normalized data, x are the values obtained from the sensors, min () is the minimum value along the dimensions, and max () is the maximum value along the dimensions.

The parameters for network optimization are presented in Table1. In MACMSA, to compress the dimension, the number of neurons in all the encoders is reduced from the input layer to the output layer. However, in the current study, the number of neurons in the encoder for the actuator position was increased to expand the dimension toward the output layer from the input layer, because the actuator position had only five dimensions. Further, if the dimension is compressed further, the decoder will not be able to represent nine pieces of data. Table 2 presents the other parameters required to optimize the autoencoders. After the optimization processes, the learning data were input to the system again, and the generated control signal was obtained as the output. We performed the results on a real arm robot to demonstrate the results. Note that the system outputs were normalized to range between [3500,10500], to correspond to the actuator position that ranges from [0,1], which is the range of outputs from the system obtained by the reverse procedure of Equation (3).

Table 1. Number of neurons in each autoencoder and loop values.

Autoencoder structure	Input layer	Hidden layer 1	Hidden layer 2	Hidden layer 3	Output layer	bp_loop1	bp_loop2
Picture	500	375	250	375	500	10 000	5 000
Sound	500	375	250	375	250	10 000	5 000
Position	5	10	20	10	5	2 000	1 500

Table 2. Parameters for optimizing the autoencoders.

$ginitial$	$greach$	$gstep$	α		ρ	β
			Picture & Audio	Position		
1.0	10.0	1.0	0.00001	0.005	0.4	1.8

4. Result

The results in this study are for the results of the highest recall rate on the associator based on 100 trials with different initial weights of the neural network. Figure 6 shows a comparison of the trajectories of the ideal control signal played on the robot and the actual output from the system played on the robot.

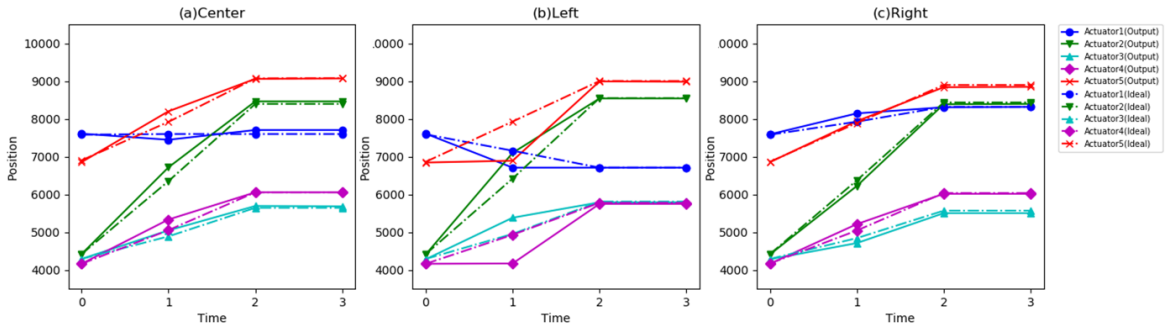

Figure 6. Comparison of the ideal trajectory and the recalled output.

The x-axis indicates the time and y-axis indicates the actuator position. In Figure 5, the solid lines of time 1-3 represent the trajectories of the control signal recalled by the system. There was a difference of approximately 1000 on the left at time 1 on actuators 4 and 5 The recall rate of the associator was 98.3%

in total and 98.9% in the part; this directly affected the output. We obtained these results using the actual arm robot to observe the movements. We found that the movement were not inferior when the ideal motion was compared. If there was a defect, it was observed that actuator 4 did not stretch enough at time 1 on the left. (Video is available at https://www.youtube.com/watch?v=cjD4s03hNaY).

5. Discussion

As mentioned in the previous section, the recalled control signal could be played, and the movements were not inferior when compared to the ideal ones. We believed that the most suitable way to verify the behavior of the proposed system would be to execute it on a real robot. Notably, slight differences occurred in the trajectories. Further, it is not possible to set the actuator to exactly the same position as that in the case of ideal data because the actuators used in the experiment were analog. Therefore, the model proposed in this study was found capable of adapting to the reaching task because the system was able to generate the control signal to the reaching destination in all the cases, even if actuator 4 did not stretch adequately at time 1 on the left.

It appears that the reason for the large error on actuator 4 on the left was the recall rate, which was not 100%, even when ideal data were input. In other words, the associator could not store all the patterns. The recall rate depends significantly on the encoder's ability of represent. In this study, the values obtained from the sensors were converted to the range [0,1] via simple normalization (Equation (3)). An approach to increase the recall rate may be used to address the whitening of data [17], thereby providing better accuracy in training autoencoders.

In this study, real-time control was not conducted because more than 20% of the noise occurred in exactly the same situation owing to hardware limitations. According to a previous study [11], a system of the size used in this study can absorb up to 20% or less of the environmental noise. To operate the proposed model for real-time control, it is necessary to carefully select the hardware. In particular, the selection of cameras and microphones is crucial. For the actuator, the error was approximately 0.2% in the same situation. Thus, the KXR-A5 arm type Ver.2 can be used. As for system component customization, a suitable neural network for each sensor and modal must be chosen. For example, CNN [18] was used for images, and LSTM [19] for sound. Autoencoders were considered adequate for the actuator position.

6. Conclusion

We proposed an improved version of MACMSA. We trained the model using real data obtained from a real arm robot in a reaching task. After training, we provided ideal data to the system as input and obtained results for the recalled control signal on the arm robot. The movements that were controlled by the recalled signals were not inferior as compared to the ideal movements. As a result, we confirmed that our proposed model performed well even with real data. A method to obtain a higher recall rate on the associator was also discussed. In the future, we will attempt to select a suitable neural network model for each modal and control the real robot in real time.

References

[1] Sakagami Y, Watanabe R, Aoyam C, Matsunaga S, Higaki N and Fujimura K 2002 The intelligent ASIMO: System overview and integration *IEEE Int. Conf. Intell. Robots Syst.* pp 2478–2483
[2] Tanaka F, Isshiki K, Takahashi F, Uekusa M, Sei R and Hayashi K 2015 Pepper learns together with children: Development of an educational application *15th Int. Conf. on Humanoid Robots* pp 270–275.
[3] Bekey GA 2005 Autonomous Robots: From Biological Inspiration to Implementation and Control The MIT Press
[4] Ramachandram D and Taylor G 2017 Deep Multimodal Learning, *IEEE Signal Process. Mag.* **34** 96-108.

[5] Sünderhauf N, Brock O, Scheirer W, Hadsell R, Fox D, Leitner J, Upcroft B, Abbeel P, Burgard W, Milford M and Corke P 2018 The limits and potentials of deep learning for robotics. *Int. J. Rob. Res.* **37** 405–420.
[6] Sergeant J Sünderhauf N, Milford M and Upcroft B 2015 Multimodal deep autoencoders for control of a mobile robot. *Australas. Conf. Robot. Autom.* pp 1–10.
[7] Yang C Sasaki K Suzuki K Kase K Sugano S and Ogata T 2017 Repeatable folding task by humanoid robot worker using deep learning *IEEE Robot. Autom. Lett.* **2** 397–403.
[8] Gamal O Cai X and Roth H 2020 Learning from Fuzzy System Demonstration: Autonomous Navigation of Mobile Robot in Static Indoor Environment using Multimodal Deep Learning *24th Int. Conf. Syst. Theory Control Comput.* pp 218-225
[9] Saito N, Ogata T, Funabashi S, Mori H and Sugano S 2021 How to Select and Use Tools?: Active Perception of Target Objects Using Multimodal Deep Learning *IEEE Robot. Autom. Lett.* **6** 2517-2524
[10] Yang L, Yan W and Wu H 2021 Comparison of deep learning-based methods in multimodal anomaly detection: A case study in human–robot collaboration Science Progress. **2** 1-24
[11] Akikawa M and Yamamura M 2021 Materializing Architecture for Processing Multimodal Signals for a Humanoid Robot Control System *JACIII* **25** 335-345
[12] Ngiam J, Khosla A, Kim M, Nam J, Lee H and Ng AY 2011 Multimodal deep learning *Proceedings of the 28th Int. Conf. on Machine Learning* pp 689–696.
[13] Noda K, Arie H, Suga Y and Ogata T 2014 Multimodal integration learning of robot behavior using deep neural networks. *Rob. Auton. Syst.* **62** 721–736
[14] Kingma D and Ba j 2014 Adam: A method for stochastic optimization arXiv (*Preprint* abs/1412.6980)
[15] https://direct.sanwa.co.jp/ItemPage/400-CAM083 (accessed on September 20, 2021)
[16] https://kondo-robot.com/product/03157 (accessed on September 20, 2021)
[17] Okatani T 2015 *Deep Learning* Sugiyama S Tokyo Bunkyo-ku Kodansya pp 67-72
[18] Zhang K, Zuo W, Chen Y, Meng D and Zhang Y 2017 Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising *IEEE Trans. Image Process.* **26** 3142-3155
[19] Graves A Jaitly N 2014 Towards End-To-End Speech Recognition with Recurrent Neural Networks *PMLR* **32(2)** 1764 -1772