Acquisition of Context-based Word Recognition by Reinforcement Learning Using a Recurrent Neural Network

by

Ahmad Afif Mohd Faudzi

10E2041

A thesis submitted to the

Faculty of the Graduate School of the

Oita University in partial fulfillment

of the requirements for the degree of

Masters in Engineering

Department of Electrical and Electronics Engineering

March 2012

Acquisition of Context-based Word Recognition by

Reinforcement Learning Using a Recurrent Neural Network

The eye movement and recognition in humans seem very flexible and intelligent. The flexible and intelligent recognition is not only depending on the information that belonged by target object, but also is supported by other information, including past knowledge and contextual information. For an example, when we read a book, we do not usually seem to read it by recognizing each character one by one. We would predict a word from the first two or several characters, and also would utilize the story context to expect the next character or word. In order to make flexible recognition, we consider many things simultaneously, and move our eyes, recognize and understand the context flexibly. Such parallel consideration and flexibility must be achieved by our parallel and flexible brain, and learning plays an important role in it.

In our laboratory, the coupling of a neural network (NN) and reinforcement learning (RL) is considered useful because of its autonomous, parallel and flexible learning ability. In the previous research, it was verified through simulations and also using a real camera that the appropriate camera motion, recognition and recognition timing were successfully acquired. However, since a regular layered neural network was used, the recognition and movement functions were limited to the case where the whole pattern is in the visual field. Furthermore, only two patterns were used.

In this thesis, context-based word recognition learning system was developed. 6 words that need context-based recognition function for the words to be recognized were chosen. The learning system was trained to recognize all the chosen words. As a learning method, the combination of Reinforcement Learning and a Recurrent Neural Network (RNN) was applied. The developed learning system has a 4-layered RNN and it was trained by BPTT method based on teaching signal that was generated by Q-Learning algorithm. The learning system was trained on several tasks using simulations or real time learning in order to verify whether flexible recognition could emerge through this context-based word recognition learning.

There are two types of problem tasks in this thesis. The first type using ideal images as

learning data and the second one using real camera captured images. For the first type, two tasks were done through simulations. The one is a fixed initial position task, and the second one is a random initial position task. As results, for both simulations the system manages to recognize all the prepared words. Here, the relation between the parameter of discount factor γ and the value of teaching signal was also discussed and how to choose the appropriate settings was proposed. In the second type of task, both simulations were trained using real camera captured images that were prepared beforehand as samples. After the learning was successfully verified, finally, the system was trained for the same task in real time. After the real time learning, learning was successful and system manages to recognize the entire prepared patterns.

All these results show that by applying a combination of Reinforcement Learning and a Recurrent Neural Network learning method, the context-based word recognition can be achieved. Flexible recognition function in the appropriate timing is mostly acquired.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Acquiring external information is one of the most important functions for biological subjects including humans. The external information is necessary for human to recognize their surrounding environment and interacts with them. Human can obtain that information through various sensory organs and interacts with the information by moving their body. Among the human sensory organs, vision is probably the most informative perception.

The eye movement and pattern recognition in human seem very flexible and intelligent. However, the performance of these abilities could be declined at certain situation such as in a situation where only intrinsic information is provided. Intrinsic information is information that belongs naturally to the target object. Besides intrinsic information, the flexible and intelligent recognition is supported by other information including past knowledge and contextual information.

In some situations, contextual information can provide more relevant information for the recognition of an object rather than the intrinsic object information[20]. For example, Fig. 1.1

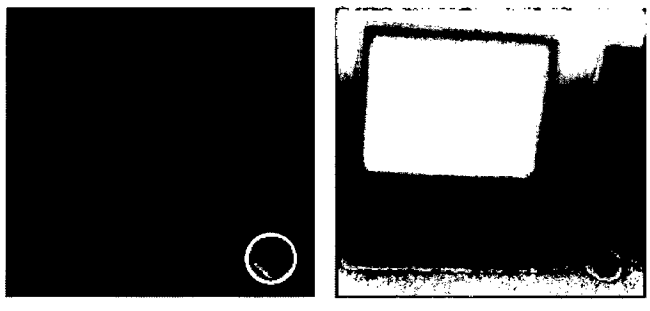

Figure 1.1: Object recognition in presence of image degradation.

shows the same image of a place. Both images are blurred (due to distance or fast scene scanning) and except the target recognition object, the left image was covered. When images in Fig. 1.1 are asked to be observe, recognition based on intrinsic features provides poor performances. However, when the object is immersed in its typical environment, we can recognize that the target object is a computer mouse based on provided contextual information; monitor, keyboard and others. We consider many things simultaneously, moving our eyes, recognizing and understanding the context flexibly. Such parallel consideration and flexibility must be achieved by our parallel and flexible brain, and learning plays an important role in it.

The author would like to ask the reader to imagine one situation where several new patterns or words were presented to us repeatedly. We were exposed to these patterns or words repeatedly until we know their size, shape and others in detail. At one time, we could recognize all the patterns correctly even if only some of parts of these recognition objects that can distinguish each other are visible to us. This explains that, human has the ability to memorize past experience and ability to recognize the patterns or words based on the contextual information.

Research on character or word recognition has been going on for a long-time[6]. Nowadays, hand-written character recognition systems are practically used already in post offices and so on. The recognition rate is high. However, these systems were usually limited to the cases when only one character is written in a predetermined area. Therefore, character segmentation still remains as one of the big problems in image recognition[9]. When a pattern on an image is shifted, the image signals are completely different from the previous signals. Fukushima *et al.* show that through neural network system learning, position shifting and small size change can be absorbed[5]. However, it is limited to the problem where the whole pattern is shown in the visual field. It is more flexible if the system could recognize more patterns, including patterns that are larger than visual field by holding some information while moving the camera to the position where it can recognize the target pattern.

Recently, one of the machine learning methods, the coupling of a neural network (NN) and reinforcement learning (RL) is considered useful because of its autonomous, parallel and flexible

learning ability. However, many researchers are still positioning RL as a learning specific for actions in the total process, and the NN as a non-linear function approximator. On the other hand, there is an approach by Shibata *et al.*, where by applying RL to a neural network that bears the entire process from sensors to motors, the necessary functions including recognition and motor control emerge[12]. Flexible sensor motion also can be expected to emerge through this framework of learning.

Shibata *et al.* also proposed Actor-Q learning method and applied it on an active perception and recognition learning system[13]. They verified through simulations and also using a real camera[4] that the appropriate camera motion, recognition and recognition timing were successfully acquired. However, because it was trained by a regular layered neural network, the recognition and movement function were limited to the case where the whole pattern is in the visual field. Furthermore, in[4] only two patterns were used.

One class of NN is known as Recurrent Neural Network where connections between neurons form a directed cycle. When introducing a recurrent NN, functions that need memory or dynamics are expected to emerge. The combination of a Recurrent Neural Network and Reinforcement Learning has been successfully applied to several different memory-required tasks. For example, a wheel-type robot was trained by given a reward when it went to the correct one of two possible goals[15]. Utsunomiya *et al.*[16] present a combination of a RNN and actor critic to enable a system to discover pattern meaning from delayed rewards.

With all that written above, the author believed that through learning, human is able to understand the contextual information that will support the flexible recognition and movement. In order to apply such function into the computer world, a parallel and flexible learning method, the combination of a RNN and RL seem promises. In this paper, context-based word recognition task was trained. Several simulations were done. There are simulations by using ideally made image data and there are also simulations that by using image data that were captured by the real camera. A combination of reinforcement learning and a RNN method is applied. Here the author would like to verify whether flexible recognition emerges through this context-based word

recognition learning.

This thesis consists of 5 chapters. Chapter 2 describes the learning system. In this chapter, several learning methods that were used in this learning system are highlighted. Chapter 3 describes several simulations that were done using ideal images. In this chapter, there are two types of tasks. In the first type of task, only decision making problem is trained while for the second type, decision-making and camera motion problems are trained. The task and the result analysis are discussed in detail here. Chapter 4 is similar like chapter 3 but here the images that were captured by the real camera were used. Finally, chapter 5 concludes with the summary of the overall research and suggestion for future work.

Chapter 2

Learning System for Context-based Word Recognition

The learning method for context-based word recognition is a combination of Reinforcement Learning (RL) and a Recurrent Neural Network. The signals from a sensor are inputted into the RNN and the teaching signals for output are generated by reinforcement learning algorithm. The teaching signals are used in network training that is executed by Back Propagation Through Time (BPTT) method.

In this chapter, firstly in section 2.1, reinforcement learning and its learning algorithms will be introduced. Then, neural network and recurrent neural network will be introduced in section 2.2 and 2.3 respectively. Finally, the learning system that combine both learning methods introduced in section 2.4.

2.1 Reinforcement Learning

Reinforcement learning is a very adaptive learning framework that allows an intelligent agent to improve its behavior through active interaction with its environment [18]. A reinforcement learning agent learns how to react in different situations to maximize a numerical reinforcement reward. The agent does not need an instructed action to learn, but instead, by trial-and-error, it finds the most valuable actions in different situations.

As shown in Fig. 2.1, reinforcement learning, as one of the popular learning methods in machine learning domain, has a formal mathematical foundation in which a learning agent manipulates its environment through a set of actions. A new environmental state and reward are

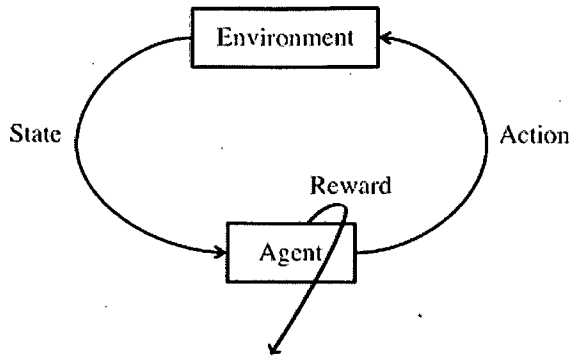

Figure 2.1: Interaction between agent and environment in reinforcement learning

perceived as the results of the executed actions. The agent stores the information as knowledge about how to find and select the action that will maximize the reward. The goal of the agent is adapt to its environment and to maximize its reward gained from the environment. Next, some of reinforcement learning algorithms; Q-learning, Actor-Critic and Actor-Q that were used in this research will be explained.

2.1.1 Q-Learning

Q-Learning [21] is one of the most widely applied reinforcement learning algorithms. It is based on Temporal-Difference(TD) methods. The benefit of using Q-learning is that the agent equipped with the algorithm can directly learn an approximation of action (state) value for a task. Hence, when learning terminates, an optimal policy can be generated from the learned action state value function. Q-Learning is an off-policy learning algorithm, which means the actions that the agent performs, can be different from the actions the agent selects to update its action state value function. An important feature for Q-learning is that it does not require a model of the environment.

In Q-learning, the action state function is stored in a Q(**s**,a) table (also called Q-value table). Q-learning generates the Q-table by performing as many actions in the environment as possible. The initialization values of this table are set to optimistic initial value before learning. The update

rule for setting values in the table is as follows in Equation 2.1 and its flow chart is as Fig. 2.2.

$$Q(\mathbf{s}_t, a_t) \Leftarrow Q(\mathbf{s}_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(\mathbf{s}_{t+1}, a) - Q(\mathbf{s}_t, a_t) \right] \tag{2.1}$$

In the value update function, $r_{t+1} + \gamma \max_a Q(\mathbf{s}_{t+1}, a)$ is the new estimation of the old $Q(\mathbf{s}_t, a_t)$. Parameter alpha (α) is a learning rate $0 < \alpha < 1$, which determines how fast the new estimation is updated to the old $Q(\mathbf{s}_t, a_t)$ value. Gamma γ is a discount factor also defined in the range

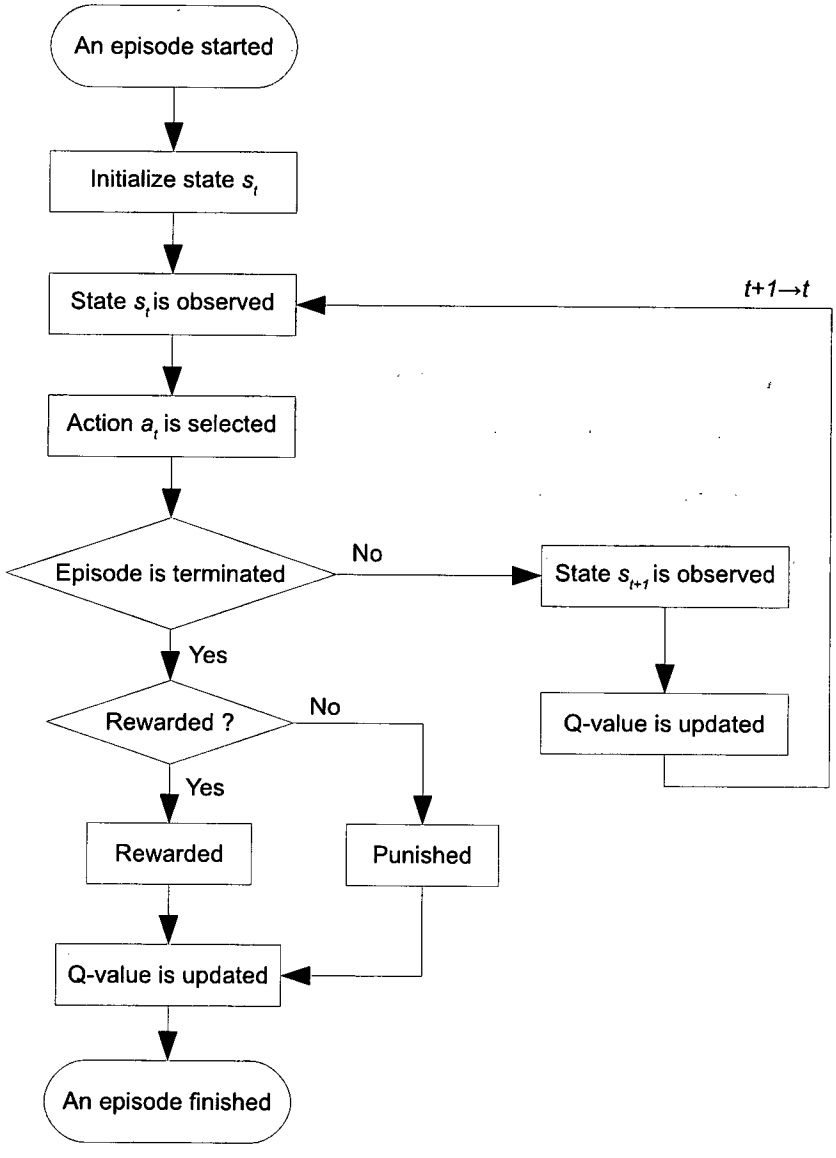

Figure 2.2: Flowchart of Q-learning algorithm in one episode

of $0 < \gamma < 1$ and is used to weight near term reinforcement more heavily than distant future reinforcement. The closer γ is to 1, the greater is the weight of future reinforcements. In this equation, the newly estimated Q-value is used to update the existing value of $Q(\mathbf{s}_t, a_t)$ which makes the old value grow toward the new value. By updating the $Q(\mathbf{s}_t, a_t)$ table through continually interacting with the environment under some conditions, the Q-value will converge toward the optimization. The convergence of Q-value can be reached when all states and actions are continually updated infinitely with exploration.

For exploration, the agent repeatedly selects an action based on a policy, such as $\varepsilon - greedy$ or *boltzman selection*. In $\varepsilon - greedy$, we call the action with the maximum action value *greedy action*. When the agent selects the greedy action at a state, it is *exploiting* its knowledge of that state. Exploiting current knowledge means making the best use of current understanding. However, greedy exploitation prevents the agent from gaining new knowledge. For learning purposes, it is necessary for an agent to select non-greedy actions occasionally. This is the *exploring* aspect for the agent, which enables the agent to discover new states and to estimate the condition of environment. An agent employed with a $\varepsilon - greedy$ policy will behave greedily most of the time, but occasionally explore other non-greedy actions with a probability ε. Overall, all actions will be tried, but the greedy ones will be executed most. Continual exploration is absolutely necessary when the agent learns in a dynamic environment that evolves with time. In this research, ε is reduced according to the progress of learning using a exponential function in following equation.

$$\varepsilon = \exp(-trial \times P) \tag{2.2}$$

where *trial* is the number of the current episode and P is the constant value.

2.1.2 Actor-Critic

Actor-critic [18] is another commonly employed reinforcement algorithm. Actor-critic method also based on TD methods that have a separate memory structure to explicitly represent the policy independent of the value function. The architecture of Actor-critic can be shown as Fig. 2.3. The policy structure is known as an *actor*, because it is used to select actions, and the estimated value function is known as a *critic*, because it criticizes the action made by the actor. Learning is always on-policy: the critic must learn about and critique whatever policy is currently being followed by the actor. The critique takes the form of TD error.

Typically, the critic is a state-value function. After each action selection, the critic evaluates the new state to determine whether things have gone better or worse. The evaluation of TD error is described as the following equation.

$$\hat{r}_t = r_{t+1} + \gamma C(\mathbf{s}_{t+1}) - C(\mathbf{s}_t) \tag{2.3}$$

where $C(\mathbf{s}_t)$ is the current value function implemented by the critic in the state $\mathbf{s}_t$, r_{t+1} is the given reward and γ is a discount factor. This TD error can be used to evaluate the action selected, the action $\mathbf{A}_t$ taken in state $\mathbf{s}_t$. If the TD error is positive, it suggests that the tendency to select $\mathbf{A}_t$ should be strengthened for the future, whereas if TD error is negative, it suggests the tendecy should be avoided. Then, by using a TD error, the critic value is updated using the following

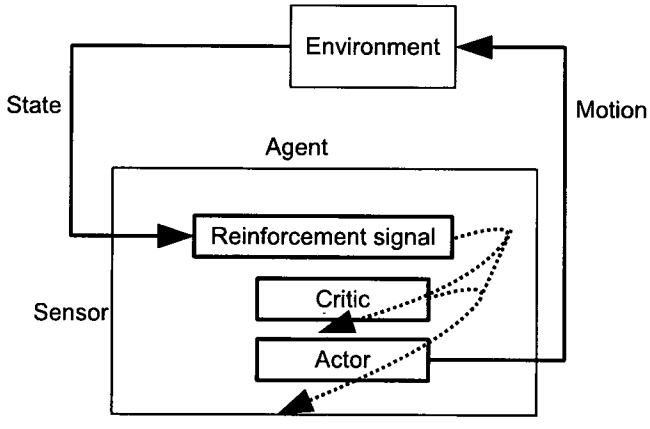

Figure 2.3: The actor-critic learning

equation.

$$\Delta C(\mathbf{s}_t) = \alpha \times \hat{r}_t \tag{2.4}$$

where α is a learning rate.

On the other hand, the actor value is updated by adding uniform random number to actor output $\mathbf{A}(\mathbf{s}_t)$ by the following equation.

$$\Delta \mathbf{A}(\mathbf{s}_t) = \alpha \times \mathbf{rnd}_t \times \hat{r}_t \tag{2.5}$$

where random numbers $\mathbf{rnd}_t$ as trial and error factors.

2.1.3 Actor-Q

Actor-Q[8] is also one of the reinforcement learning method. The algorithm of this learning method is the a combination of Q-Learning and Actor-Critic. It enables a learner to learn simultaneously both discrete decision making and continuous motions generation. For example, in this research (in section 3.3), the outputs of the learning system are divided into two types; "action", which are discrete intentions, and "motion", which is a continuos value. For every step, "action" is determined, and if the action required "motion", "motion" is also determined. "Action" is chosen based on Q-values and Q-learning is used for training. On the other hand, actor, which is usually used in Actor-Critic learning, is employed to generate a "motion". In this system, the Q-value corresponding to the "motion" is used to train the actor on behalf of the critic.

The update equation for Q value is same as equation for Q-Learning. However, TD-error in the update equation for actor is generated from the Q-value instead of critic. The equation is given as

$$\hat{r}_t = r_{t+1} + \gamma \max_a Q_a(\mathbf{s}_{t+1}) - Q_{a_t}(\mathbf{s}_t). \tag{2.6}$$

2.2 Neural Network

The human brain is capable of computationally demanding perceptual acts (e.g. recognition of faces, speech) and control activities (e.g. body movements and body functions). The advantage of the brain is its effective use of massive parallelism, the highly parallel computing structure, and the precise information-processing capability. The human brain is a collection of more than 10 billion interconnected neurons[22]. Fig. 2.4 shows a cell of each neuron that uses biochemical reactions to receive, process, and transmit information.

Typical neurons collects signal from other neurons through a host of fine structures called *dendrites*. The neuron sends out spikes of electrical activity through a long, thin fiber known as an *axon*. The end axon may split into thousands of branches. At the end of each branch, a structure called *synapse* sends the electric signals to other neuron through chemical interaction. When a neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes.

The first wave of interest in neural networks emerged after the introduction of simplified neurons by McCulloch and Pitts[7]. Artificial Neural Networks (ANN) have been developed as simplification of mathematical models of biological nervous systems.

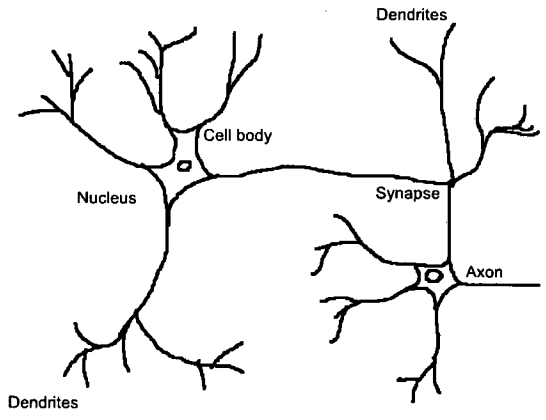

Figure 2.4: A model of a biological neuron

A typical artificial neuron and the modeling of a multi-layered neural network are illustrated in Fig. 2.5. Once modeling an artificial functional model from the biological neuron, we must take into account three basic components. First, the synapses of the biological neuron are modeled as weights. Let's remember that the synapse of the biological neuron is the one that interconnects the neural network and gives the strength of the connection. For an artificial neuron, the weight is a number, and represents the strength of the connection. The following components of the model represent the actual activity of the neuron cell. All inputs are weighted and summed altogether. This activity is referred as a linear combination. Finally, an activation function controls the amplitude of the output.

Mathematically, this process is described in the Fig. 2.5. From this model the internal activity of the neuron can be shown to be

$$v_k = \sum_{j=1}^{p} w_{kj} \cdot x_j \tag{2.7}$$

The output of the neuron, y_k, would therefore be the outcome of activation function on the value of v_k. The activation function acts as a squashing function, such that the output of a neuron in a neural network is between certain values (usually 0 and 1, or -1 and 1). The most common activation function is the sigmoid function that can be calculated by following equation.

$$y_k = f(v_k) = \frac{1}{1 + e^{-v_k}} \tag{2.8}$$

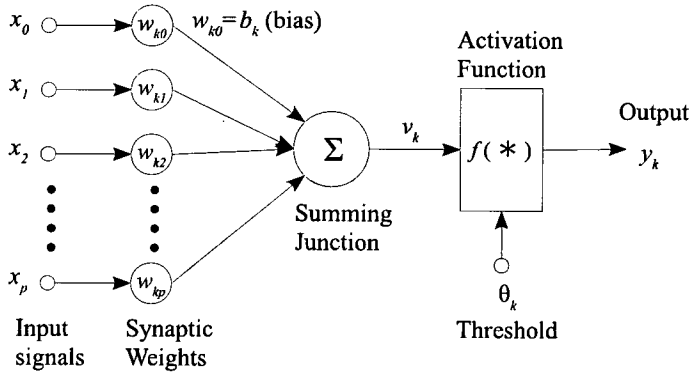

Figure 2.5: Architecture of an artificial neuron

2.2.1 Multi-layer feed-forward Neural Network

As shown in Fig. 2.6, a feed-forward network has a layered structure. Each layer consists of units which receive their input from units from a layer directly below and send their output to units in a layer directly above the unit. There are no connections within a layer. The N_i inputs are fed into the first layer of $N_{h,1}$ hidden units. The input units are merely 'fan-out' units, no processing takes place in these units. Consider $x_j^{(l)}$ is the output of j-th neuron at the (l)-th layer, $U_k^{(l+1)}$ is the net value of k-th neuron at the $(l+1)$-th layer and $w_{kj}^{(l+1)}$ is the connection weight between j-th neuron at (l)-th layer and k-th neuron at $(l+1)$-th layer. Then, the net value of the k-th hidden neuron, $U_k^{(l+1)}$ is given by

$$u_k^{(l+1)} = \sum_{j=1}^{n_1} w_{k,j}^{(l+1)} \cdot x_j^{(l)} + \theta_k \tag{2.9}$$

Here, n_1 is the neuron number at (l)-th layer and θ_k is a bias. However, in this research, bias, θ was set to 0. In order to get the output of k-th neuron at the $(l+1)$ layer, the sigmoid function that was mentioned in Eq. (2.8) is used. However, in this research, as shown in Eq. (2.10), the outcome of activation function is deducted by 0.5 to shift the original range to between -0.5 and 0.5.

$$x_k^{(l+1)} = \frac{1}{1+e^{-u_k^{(l+1)}}} - 0.5 \tag{2.10}$$

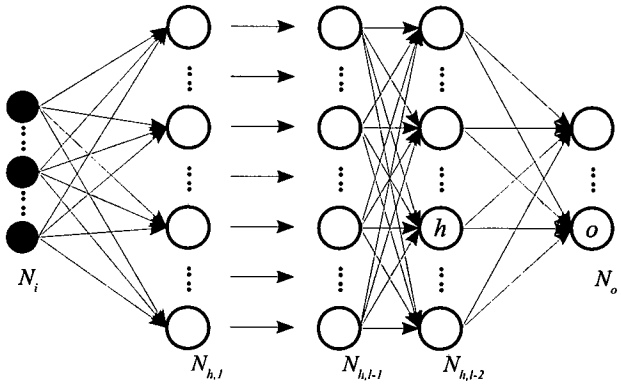

Figure 2.6: Architecture of an artificial neuron

2.2.2 Back Propagation (BP) learning method

After calculating the feedforward computation as shown above, the value of the connection weights need to be updated in order to reduce the difference between the network output with the *desired output* or *target output*.

In this section, Back-propagation (BP)[11], currently the most popular algorithm to perform the supervised learning task will be explained. This algorithm gives a prescription for changing the connection weights in any feed-forward networks based on the error function E. The error function E is defined as the total quadratic error. Consider a network that have 3 layer where the input, hidden and output layer is first (1), second (2) and third(3) layer respectively and d_k is a teaching signal for the k-th output of the neuron at the output layer. Here, E can calculated by

$$E = \frac{1}{2}\sum_{k=1}^{N_o}\left(d_k - x_k^{(3)}\right)^2 \tag{2.11}$$

where N_o is neuron number at the output layer. Since we want to minimize the error function E according to the steepest descent method, the amount of correction in connection weights is calculated by the following equation.

$$\Delta w_{kj}^{(3)} = -\eta\frac{\partial E}{\partial w_{kj}^{(3)}} = -\eta\frac{\partial E}{\partial x_k^{(3)}}\cdot\frac{dx_k^{(3)}}{du_k^{(3)}}\cdot\frac{\partial u_k^{(3)}}{\partial w_{kj}^{(3)}} \tag{2.12}$$

where η is a learning rate. Here, the partial difference of E with the output $x_k^{(3)}$ at k-th node in the output layer is given by

$$\frac{\partial E}{\partial x_k^{(3)}} = \frac{\partial}{\partial x_k^{(3)}}\left(\frac{1}{2}(d_k - x_k^{(3)})^2\right) = -\left(d_k - x_k^{(3)}\right) \tag{2.13}$$

The second term of Eq.(2.12) is calculated using the derivative of sigmoid function. From Eq. Eq.(2.10), the derivative of sigmoid function, $f'(u_k)$ is calculated as

$$\frac{dx_k^{(3)}}{du_k^{(2)}} = (0.5 + x_k^{(3)})(0.5 - x_k^{(3)}) \tag{2.14}$$

Then, the net value, U_k is derived by the following equation.

$$\begin{aligned}\frac{\partial u_k^{(3)}}{\partial w_{kj}^{(3)}} &= \frac{\partial}{\partial w_{kj}^{(3)}}\left(x_1^{(2)}w_{k1}^{(3)} + x_2^{(2)}w_{k2}^{(3)} + \cdots + x_j^{(2)}w_{kj}^{(3)}\right) \\ &= x_j^{(2)}\end{aligned} \tag{2.15}$$