A Neuromorphic Categorization System with Online Sequential Extreme Learning

Ruoxi Ding[1], Bo Zhao[2] and Shoushun Chen[1]

[1]VIRTUS IC Design Centre of Excellence, School of EEE, Nanyang Technological University, Singapore
[2]Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore

***Abstract*—This paper presents an event-driven categorization system which processes the address events from a Dynamic Vision Sensor. Using neuromorphic processing, cortex-like spike-based features are extracted by an event-driven MAX-like convolutional network. The extracted spike patterns are then classified by an Online Sequential Extreme Learning Machine with Auto Encoder. Using a Lookup Table, we achieve a virtually fully connected system by physically activating only a very small subset of the classification network. Experimental results show that the proposed system has a very fast training speed while still maintaining a competitive accuracy.**

I. Introduction

Primates' vision is very accurate and efficient in object categorization. The computation in the primates' visual cortex is based on asynchronous sparse and spike-based signaling and processing. Address Event Representation (AER) Dynamic Vision Sensors (DVS), which encodes dynamic transition of the scene into asynchronous address events (spikes), is therefore the ideal visual frontend to implement a biomimetic vision processing system.

To fully utilize the power of AER vision sensors, event-driven vision processing algorithms should be utilized. In the literature, lots of research work have been done on event-driven vision processing, such as real-time object tracking through event-based clustering [1], event-based convolution in a convolutional network for object recognition [2], and event-driven feed-forward categorization using cortex-like features and a spiking neural network [3]. However, designing a real-time categorization system with the capability of a fast online training speed remains a challenge.

In this paper we propose an event-driven categorization system with a fast online training speed. The system takes the address events data from a DVS. It extracts cortex-like spike-based features through a neuromorphic feature extraction unit and performs classification on the extracted feature spike patterns using an Online Sequential Extreme Learning Machine (OSELM) [4] with Auto Encoder (AE) and Lookup Table (LUT). A Motion Symbol Detector in the system detects a burst of input AER events and then trigger the successive processing parts. We target a fast online training speed and a hardware friendly architecture. The major contribution of this work lies in two areas: 1) the seamless integration of an online sequential learning method into the AER categorization system; 2) the use of an LUT to achieve a virtually fully connected system by physically activating only a very small subset of the classification network.

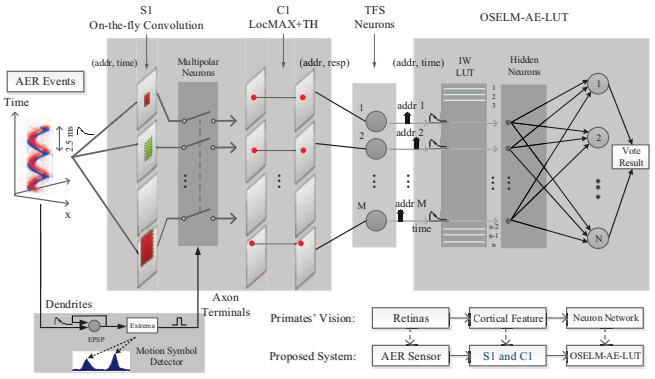

Fig. 1. Architecture of the proposed AER categorization system. The lower right part of the figure illustrates the neuromorphic strategy of the system.

The rest of this paper is organized as follows. Section II explains the system architecture. Section III-V illustrates the building modules and the corresponding algorithms. Experimental results are reported in Section VI and conclusions are drawn in Section VII.

II. System Architecture

Fig. 1 shows the architecture of the proposed system. The flow of information processing is as follows.

1) **Feature Maps and Neural Competition:** The event stream from the AER sensor is sent in parallel to a batch of S1 feature maps [5], where on-the-fly convolution with Difference of Gaussian (DoG) filters are performed. A forgetting mechanism is introduced in the convolution to forget the impact of very old events. Four scales (3, 5, 7 and 9) of DoG filters are used after a trade-off between the complexity and the performance. Each S1 neuron competes with its local neighbors (i.e. the neurons that locate within its receptive field) and can only survive in the C1 layer if it is the maximum among those local neighbors (LocMAX) [3].
2) **Motion Symbol Detector and Feature Spike Conversion:** Note that the convolution maps (S1) are updated for each incoming AER event, however, the classification decision does not need to be made in such a high speed. A Motion Symbol Detector is thus introduced in the system to detect a burst of input AER events and

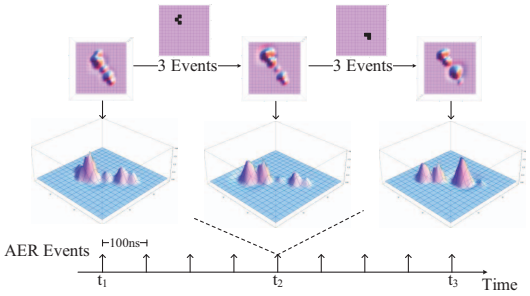

Fig. 2. Reconstructed images illustrating the on-the-fly Convolution with a forgetting mechanism.

then trigger the successive processing stages. It consists of a leaky integrate neuron and an Extrema detector. Survived C1 responses after the LocMAX operation are sent to a small set of Time-to-First-Spike (TFS) neurons, where they are converted into spikes.

3) **OSELM with Auto Encoder and Lookup Table:** The generated feature spike patterns are then fed to the OSELM-AE-LUT for classification. Each feature spike is associated with an address, which can be used to access the LUT and fetch a corresponding weight. In this way, we achieve a virtually fully connected system by physically activating only a very small subset of the classification network. The final classification decision is made by applying Majority Voting to the auto encoder outputs of the OSELM.

III. Feature Maps and Neural Competition

S1 Maps are built by convolving each input event with a bank of DoG filters on the fly. Each filter models a neuron cell that has a certain size of receptive field and responds best to a feature with a certain extent. The DoG filter used in this system is the approximation of Laplacian of Gaussian (LoG), which is defined as:

$$DoG_{\{s,l_c\}}(l) = G_{\sigma(s)}(l - l_c) - G_{3\cdot\sigma(s)}(l - l_c) \quad (1)$$

$$G_{\sigma(s)}(l) = \frac{1}{2\pi \cdot \sigma(s)^2} \cdot \exp\left(-\frac{||l||^2}{2 \cdot \sigma(s)^2}\right) \quad (2)$$

where $G_{\sigma(s)}$ is the 2-D Gaussian with variance $\sigma(s)$ which depends on the scale s. l_c is the center position of the filter.

The on-the-fly convolution is illustrated in Fig. 2. Each input event adds the convolution kernel to the S1 map. The center of kernel is aligned to the map according to the address of the input event. In order to eliminate the impact of very old events on current response map, a forgetting mechanism is adopted. Each pixel in the response map will decrease (or increase) toward the resting potential (usually set as 0) as time goes by.

Through the convolution with 4 scales of DoG filters, we get 4 S1 maps. C1 maps are then obtained by performing the LocMAX operation on S1 maps. As shown in Fig. 3, each S1 neuron competes with its local neighbors (i.e. the neurons that locate within its receptive field) and can only survive in the C1 layer if it is the local maximum. Neurons in different S1 maps have different sizes of receptive field, which vary from 3×3 to 9×9. After the LocMAX operation, each survival neuron in C1 maps represents a feature, i.e. a blob with certain extent.

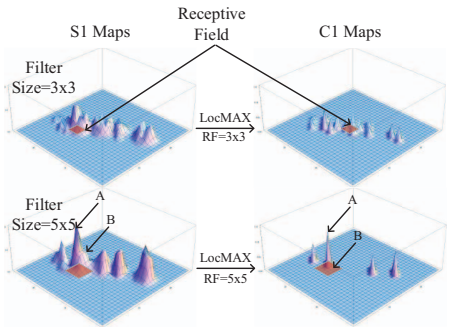

Fig. 3. The LocMAX operation. Each S1 neuron competes with its local neighbors and can only survive in the C1 layer if it is the local maximum.

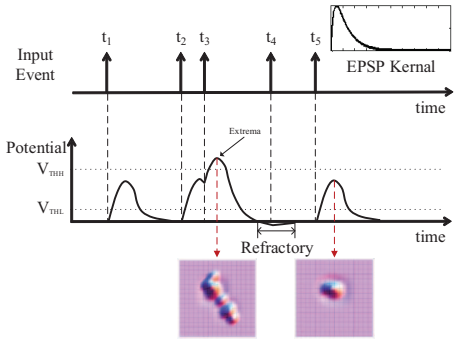

Fig. 4. Motion Symbol Detector. Each input event initiates an EPSP to the neuron. The total potential of the neuron is obtained by superposition of all EPSPs. This figure also shows two reconstructed images of an S1 map at two particular timings (the time of Extrema and post refractory).

IV. Motion Symbol Detector and Feature Spike Conversion

The proposed system utilizes a Motion Symbol Detector to detect a burst of input AER events and then trigger the successive processing stages. The Motion Symbol Detector consists of a leaky integrate neuron and an Extrema detector. As shown in Fig. 4, each input event contributes an Excitatory Post-Synaptic Potential (EPSP) to the leaky integrate neuron. The total potential of this neuron is obtained by superposition of all EPSPs. If the neuron's potential exceeds V_{THH} and the curve has reached a peak, an Extrema can be identified and a pulse will then be generated to turn ON the Multipolar Neurons (Switches) between S1 and C1 in Fig. 1. After this, the neuron will go through a short refractory period, during which new input events will not be responded. Fig. 4 also shows two reconstructed images at different timings to testify the effectiveness of our algorithm. Note that the Motion Symbol Detector has another small threshold V_{THL} to filter out noise.

The survived C1 responses after the LocMAX operation are sent to a small set of Time-to-First-Spike (TFS) neurons, where C1 responses are converted into spikes. Each spike is again in the form of AER; it has an address and a timestamp.

V. OSELM WITH AUTO ENCODER AND LOOKUP TABLE

The extracted feature spike patterns are then fed to an OSELM for classification.

A. ELM and OSELM

The extreme learning machine (ELM) is an emerging and efficient learning technique provides unified solutions to generalized feed-forward neural networks. A Single Hidden Layer Feed-forward Neural Network (SLFN) with nonlinear activation functions and N hidden layer neurons can classify N distinct features. In addition, the Initial Weights (IW) and hidden layer biases do not require tuning and can be randomly generated [6]. ELM has been proven to have a fast training speed and competitive classification performance. It transforms learning into determining output weights β through a generalized inverse operation of the hidden layer weight matrices. For N arbitrary distinct samples (x_i, t_i), the standard ELM with L hidden neurons and the activation function $g(x)$ is modeled by:

$$f_{\tilde{N}}(X_j) = \sum_{i=1}^{\tilde{N}} \beta_i g(a_i, b_i, x) = t_j, j = 1, ..., N. \quad (3)$$

where a_i and β_i represent the weight vectors from input neurons to the ith hidden neuron and from the ith hidden neuron to output neurons, respectively. b_i is the bias for ith hidden node. The above ELM is proved to be able to approximate N samples with zero error.

The smallest training error is achieved by using the above model since it represents the least-square solution of the linear system of $H\beta = \Gamma$ as $||H\hat{\beta} - \Gamma|| = ||HH^{\dagger}H - \Gamma|| = min||H\beta - \Gamma||$, where $H^{\dagger}$ represents a Moore-Penrose generalized inverse of the hidden layer output matrix H. However, the above solution assumes that all N distinct training observations are available (Batch Learning), which is not the case for actual real-time learning. Hence, modifications are required to suit it to the scenario of online sequential/increamental learning [4]. Under the condition of $rank(H) = L$:

$$\beta = H^{\dagger}\Gamma; H^{\dagger} = [H^T H]^{-1} H^T = (\Psi)^{-1} H^T \quad (4)$$

Non-singularity of $\Psi = H^T H$ can be ensured by decreasing the number of hidden layer neurons or increasing the number of training samples N in the initialization phase.

Suppose that we have a set of initial training set $(x_i, t_i, 1 \leq i \leq N_0)$. The ELM learning for these data is equivalent to the problem of minimizing the error $||H_0\beta - \Gamma_0||$ for $N_0 \geq L$ where

$$H_0 = \begin{bmatrix} G(a_1, b_1, x_1) & \cdots & G(a_L, b_L, x_1) \\ \vdots & \cdots & \vdots \\ G(a_1, b_1, x_{N_0}) & \cdots & G(a_L, b_L, x_{N_0}) \end{bmatrix}_{N_0 \times L} \quad (5)$$

and Γ_0 represents the set of targets $t_i (1 \leq i \leq N_0)$.

One solution to minimize $||H\beta - \Gamma_0||$ is $\hat{\beta}^{(0)} = (\Psi_0^{-1} H_0^T \Gamma)$. Suppose that we have another set of training data containing

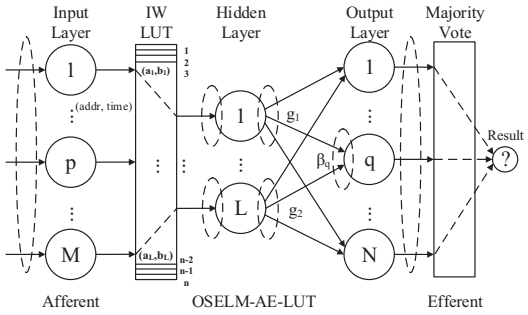

Fig. 5. OSELM-AE-LUT. Each input feature spike has an address, which is used to access the Initial Weight LUT and fetch its corresponding weight. The timestamps of the spikes are used as the input. Recursive β update only occurs between Hidden Layer and Output Layer. The classification decision is made by Majority Voting of the Auto Encoder outputs of ELM.

N_1 number of samples. Now the error minimization problem is transformed into two sets of data:

$$\epsilon = \left|\left|\begin{bmatrix} H_0 \\ H_1 \end{bmatrix}\beta - \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \end{bmatrix}\right|\right| \quad (6)$$

Solving the error minimization problem for both sets of data, the output weight matrix $\beta^{(1)}$ becomes:

$$\beta^{(1)} = (\Psi_1)^{-1} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \end{bmatrix} \quad (7)$$

Online sequential learning needs to represent $\beta^{(1)}$ as a function of $\beta^{(0)}, \Psi_1, H_1$, and Γ_1. Note that:

$$\begin{aligned} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \end{bmatrix} &= H_0^T\Gamma_0 + H_1^T\Gamma_1 \\ &= \Psi_1\beta^{(0)} - H_1^T H_1 \beta^{(0)} + H_1^T\Gamma_1 \end{aligned} \quad (8)$$

So, combining (7) and (8), we have

$$\beta^{(1)} = \beta^{(0)} + (\Psi_1)^{-1}\Psi_1^T\left[\Gamma_1 - H_1\beta^0\right] \quad (9)$$

The recursive expression for updating the nonlinear model can then be achieved:

$$\beta^{k+1} = \beta^k + \Psi_{k+1}^{-1} H_{k+1}^T \left(\Gamma_{k+1} - H_{k+1}\beta^k\right) \quad (10)$$

$$\Psi_{k+1}^{-1} = \Psi_k^{-1} - \Psi_k^{-1} H_{k+1}^{-1}[I + H_{k+1}\Psi_k^{-1}H_{k+1}^T]^{-1} \quad (11)$$

The β recursive update algorithm shown above calculates the error of every training sample, gets the error weight $\beta = \Psi_{k+1}^{-1} H_{k+1}^T(\Gamma_{k+1} - H_{k+1}\beta^k)$ and update the previous β^k. The OSELM does not require the number of training samples (Block Size) to be identical. The recursive learning approach presented in (10) and (11) consists of two phases: (1) Initialization Phase; (2) Sequential Learning.

B. OSELM with Auto Encoder and Lookup Table

In principle, we need all the C1 responses for classification. Let $m \times n$ denotes the resolution of feature maps and let N denote the number of classes, the size of the fully connected ELM network is: $4 \times m \times n$-input and N-output. The size of the network is quite large. However, thanks to the LocMAX operation and the AER nature of feature spikes, we can

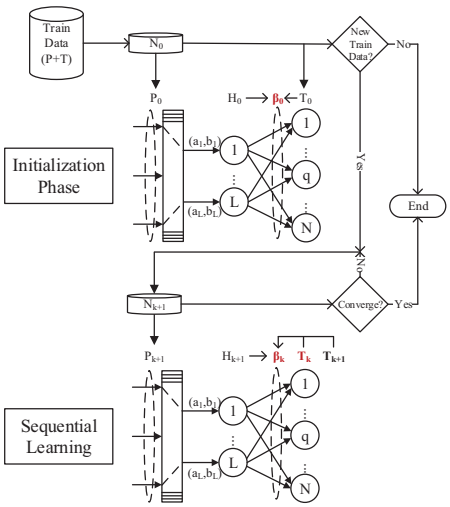

Fig. 6. The recursive training process of OSELM has two phases: Initialization and Sequential Learning. The Training process terminates when there is no new training data or β converges.

achieve the same results as the full network using a very small network that has only a few inputs (100 in our case). This will tremendously reduce the hardware cost. As shown in Fig. 5, we use a LUT to store the Initial Weights between the input layer and the hidden layer of ELM. Each input feature spike has an address, which is used to access the Initial Weight LUT and fetch its corresponding weight. The timestamps of the feature spikes are used as the input. We utilizes OSELM to auto encode the output of each class to a vector $\vec{t}$. Each class is represented by 10 bits. The final classification decision is made by applying Majority Voting to the outputs. Fig. 6 shows the flow diagram of the recursive training process in the OSELM-AE-LUT.

VI. Experiment Results

A. Training Block Size and Hidden Neuron Number

The OSELM-AE-LUT classifier has two major parameters to be tuned: the sequential training block size and the number of hidden neurons. As shown in Fig. 7, the classification accuracy almost remains the same as the block size increases, however, the training time decreases dramatically. When the number of hidden neurons increases, the accuracy improves, but the training time also gets longer. We set the number of hidden neurons to be 400 after a trade-off between the accuracy and the training speed. Considering the actual real-time processing scenario, we set the block size to be 2.

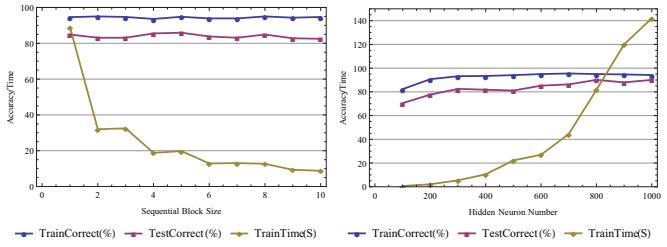

Fig. 7. Parameter selection for the OSELM-AE-LUT classifier: sequential training block size (left) and numbers of hidden neurons (right).

B. Performance Evaluation

The proposed system was evaluated on the AER posture dataset captured by Zhao et. al [3]. This dataset contains three human actions, namely ($BEND$), ($SITSTAND$) and ($WALK$). Fig. 8 shows a few reconstructed images. We compared our work with Zhao [3] and another two biologically inspired algorithms: HMAX [7] and Serre et. al [8]. The results in Table I show that the propose system has a very fast training speed while still maintaining a competitive accuracy.

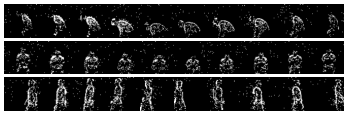

Fig. 8. Reconstructed images from the AER posture dataset, which contains three kinds of human actions. Each row corresponds to one action.

TABLE I
Performance on Posture Dataset

Model	TestCorrect(%)	TrainTime(s)
This Work (Sigmoid)	90.1 ± 3.5	21.1
This Work (Hardlim)	88.0 ± 3.4	26.8
Zhao et. al [3]	98.5 ± 0.8	4620
HMAX + SVM [7]	78.1 ± 4.1	–
Serre + SVM [8]	93.7 ± 1.9	–

VII. Conclusion

This paper presents an event-driven AER categorization system with an extreme online learning speed. Cortex-like spike-based features are extracted by a neuromorphic architecture and then classified by an online sequential extreme learning machine with auto encoder and lookup table, which provides a very fast training speed. Experimental results show that the proposed system provides a huge reduction of the training time while still maintaining a competitive accuracy.

References

[1] M. Litzenberger, C. Posch, D. Bauer, A. Belbachir, B. K. P. Schon, and H. Garn, "Embedded vision system for real-time object tracking using an asynchronous transient vision sensor," in *12th Signal Processing Education Workshop*, Sept. 2006, pp. 173–178.

[2] J. Perez-Carrasco, B. Zhao, C. Serrano, B. Acha, T. Serrano-Gotarredona, S. Chen, and B. Linares-Barranco, "Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate-coding. Application to feed forward ConvNets," *IEEE Trans. Pattern Anal. Mach. Intell*, 2013.

[3] B. Zhao, Q. Yu, H. Yu, S. Chen, and H. Tang, "A bio-inspired feedforward system for categorization of AER motion events," in *IEEE BioCAS*, Oct 2013, pp. 9–12.

[4] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. Neural Networks*, vol. 17, pp. 1411–1423, 2006.

[5] S. Chen, P. Akselrod, B. Zhao, J. Perez-Carrasco, B. Linares-Barranco, and E. Culurciello, "Efficient feedforward categorization of objects and human postures with address-event image sensors," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 34, no. 2, pp. 302–314, Feb. 2012.

[6] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst. Man Cybern, B Cybern.*, vol. 42, no. 2, pp. 513–529, April 2012.

[7] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, 1999.

[8] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust object recognition with cortex-like mechanisms," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 29, no. 3, pp. 411–426, 2007.