2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Bio-inspired Categorization using Event-driven Feature Extraction and Spike-based Learning

Bo Zhao
Institute for Infocomm Research
Agency for Science, Technology and Research (A*STAR)
Singapore 138632
Email: zhaob@i2r.a-star.edu.sg

Shoushun Chen
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore 639798
Email: eechenss@ntu.edu.sg

Huajin Tang
Institute for Infocomm Research
Agency for Science, Technology and Research (A*STAR)
Singapore 138632
Email: htang@i2r.a-star.edu.sg

***Abstract*—This paper presents a fully event-driven feedforward architecture that accounts for rapid categorization. The proposed algorithm processes the address event data generated either from an image or from Address-Event-Representation (AER) temporal contrast vision sensor. Bio-inspired, cortex-like, spike-based features are obtained through event-driven convolution and neural competition. The extracted spike feature patterns are then classified by a network of leaky integrate-and-fire (LIF) spiking neurons, in which the weights are trained using tempotron learning rule. One appealing characteristic of our system is the fully event-driven processing. The input, the features, and the classification are all based on address events (spikes). Experimental results on three datasets have proved the efficacy of the proposed algorithm.**

I. Introduction

A. Feedforward Architecture for Rapid Categorization

Primates vision is extremely accurate and efficient in object categorization. This is usually ascribed to the ventral pathway processing in visual cortex [1]. It has been for decades a hot research topic to model the feature representations in visual cortex and design systems that mimic cortical information processing. Until today, our understanding of visual cortex has been boosted by massive research works in neurobiology and neurophysiology.

The studies on processing speed in the visual cortex have constrained the “immediate” object recognition to be mainly feedforward. Thorpe et al. [2] performed rapid categorization experiments on human subjects. The subjects were shown with previously unseen images flashed on for just 20ms and were then asked to decide whether an animal existed or not in the image. Event-related potentials (ERP) analysis revealed that humans could perform rapid categorization tasks in natural images approximately 150ms after stimulus onset. Thus the visual processing from V1 to IT (in human cortex) must be within 150ms. The short processing time along ventral visual stream strongly suggest that the flow of information is mostly feedforward [3].

Among many neurophysiologically plausible feedforward models of information processing in visual cortex, HMAX proposed by Riesenhuber and Poggio [4] is one of the most popular theories. HMAX extends the Hubel and Wiesel classical model of simple cells and complex cells [5]. It summarizes the basic facts about the ventral visual stream ($V1$-$V2$-$V4$-IT). HMAX consists of a hierarchy of “S” layers and “C” layers (“S” and “C” follow the notation in Fukushima’s Neocognitron [6]). The “S” layer cells increase feature complexity by linear weighted summation of inputs; while “C” layer cells increase invariance through nonlinear MAX pooling operations, which select the maximum input as the response of the cell.

HMAX model was further extended by Serre et al. [7], [8]. The whole feedforward architecture remained the same ($S1$-$C1$-$S2$-$C2$). $S1$ and $C1$ layer correspond to simple and complex cells in primary visual cortex $V1$, while $S2$ and $C2$ are roughly related to $V2$ and $V4$, respectively. The first two layers of Serre’s model are mostly consistent with the original HMAX (differences exist in the adoption of Gabor filters rather than difference of Gaussians). The last two layers ($S2$ and $C2$) are where Serre et al. has made significant modifications, with learning introduced at stage $S2$ [7]. He first randomly extracts a number of patches from $C1$ maps of training images, and uses these patches as radial basis function (RBF) centers. Then for each image, Gaussian RBF is applied to the distance between $C1$ maps and patches, followed by a MAX operation to generate the shift- and scale- invariant $C2$ features. Promising results comparable to state-of-the-art computer vision systems have been achieved in object recognition tasks using natural images [8].

B. Spiking Neural Network

The aforementioned bio-inspired models (HMAX [4] and Seree model [8]) put large efforts on building the hierarchical feedforward architecture, however, they seem to omit another very important factor, i.e. the spike-based representation and computation.

Neurons in the brain communicate with each other by short electrical pulses, the so-called action potentials or spikes [9]. Various models have been proposed in the literature to describe the dynamics of a spiking neuron, such as leaky integrate-and-fire (LIF) model [9], Hodgkin-Huxley model [10], and Izhikevich model [11]. Among these models, LIF has the simplest structure and thus has been widely used. By combining multiple spiking neurons and storing weight information in synapses, we can construct a spiking neural network

(SNN) to learn and discriminate spatiotemporal spike patterns. Experimental studies in neuroscience have revealed a phenomenon namely spike-timing-dependent plasticity (STDP). The synaptic strength will be regulated by the relative timing of presynaptic and postsynaptic spike. It has been observed long term potentiation (LTP) of synaptic strength when a presynaptic neuron fires shortly before a postsynaptic neuron, and long term depression (LTD) when the presynaptic neuron fires shortly after [12]. STDP-based rules have been studied in [12]–[14] for unsupervised learning of spike patterns. In addition to unsupervised STDP rules, supervised learning schemes such as tempotron [15] and ReSuMe [16] have also been widely exploited. Compare to ReSuMe, which specifies a desired firing time, tempotron learning rule only needs to label the status of firing or not, and thus it is more suitable for real-world stimuli categorization.

C. Event-driven Neuromorphic Processing

Both the spike-based representation and the feedforward hierarchy should be considered in order to mimic the biological processing architecture that accounts for rapid categorization.

Event-driven neuromorphic processing is such a research area that aims to build electronic systems that have the same efficiency of brains, by mimicking the biological use of the asynchronous, sparse, spike-based representation and computation.

Address-Event-Representation (AER) [17] sensors naturally provide a way to incorporate event-driven computation. AER sensors generally have an output-by-demand nature. Hardware-based pixel-level computation is performed on chip to reduce output redundancy. For example, AER temporal contrast vision sensors allow pixel-parallel image processing at the focal plane. Each pixel in the sensor can individually monitor the relative change in light intensity and output an event if the change is larger than a user-defined threshold [18], [19]. Therefore, a lot of data redundancy is removed in the scene, and only the relevant information is outputted. The output of an AER sensor is an asynchronous stream of digital address events. Each event has an address and a timestamp. The address indicates which pixel the event is from, and the timestamp represents the event's time of occurrence.

In order to fully utilize the power of AER sensors, the concept of "event-driven processing" should be applied to every computing stage. Event-based moving object tracking is studied in [20]. An embedded vision system is designed and combined with an AER temporal contrast vision sensor to achieve real-time object tracking through efficient event-based clustering. Event-based convolution for feature extraction has been exploited in [21]. AER 2D convolution chips for neuromorphic spike-based cortical processing have been designed to accelerate convolutions of programmable kernels over the AER visual input. Moreover, event-based convolution is also applied to Convolutional Networks (ConvNets) in [22] to generate a frame-free AER-based ConvNet for categorization of AER visual event stream. The AER-based ConvNets have a similar architecture to conventional frame-based ConvNets [23], where convolution and subsampling modules interlace. Due to the frame-less event-based processing, AER-based ConvNets have a significant advantage in terms of input-output latency. However, the learning of the AER-based ConvNets is based on mapping from frame-based ConvNets but not naturally spike-based learning.

In this paper, we introduce a fully event-driven feedforward architecture that accounts for rapid categorization after stimulus onset. The proposed algorithm processes the address event data generated either from an image or from AER temporal contrast vision sensor. Bio-inspired, cortex-like, spike-based features are obtained through event-driven convolution and neural competition. The extracted spike feature patterns are then classified by a network of LIF spiking neurons, in which the weights are trained using tempotron learning rule.

Our major contribution resides in a fully event-driven architecture that can emulate the biological use of asynchronous, sparse, spike-based signaling and computation. In our algorithm, the input, the features, and the classification are all based on address events (spikes).

The rest of this paper is organized as follows. Section II describes system architecture. Sections III and IV illustrate the system building modules. Experimental results are reported in Section V and conclusions are drawn in Section VI.

II. System Architecture

Fig. 1 shows the architecture of the proposed event-driven categorization system. Our system takes AER stream as input. The flow of information processing is outlined in Algorithm 1. The details of the algorithm will be illustrated in the following several sections.

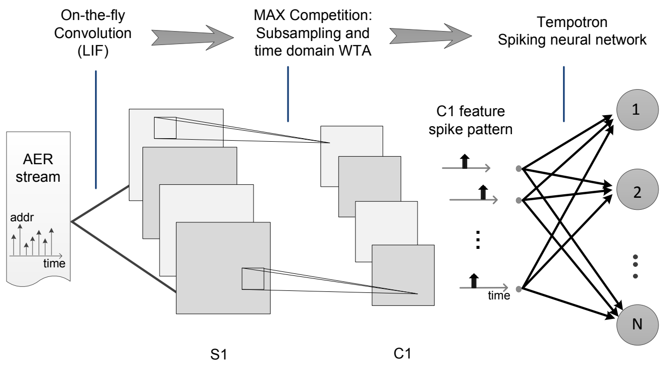
On-the-fly Convolution (LIF)
MAX Competition: Subsampling and time domain WTA
Tempotron Spiking neural network
AER stream
addr
time
C1 feature spike pattern
time
1
2
N
S1
C1

Fig. 1. Architecture of the proposed event-driven categorization system. The system takes AER stream as input. Each address event is projected onto a set of Gabor filters to build $S1$ feature maps through on-the-fly event-driven convolution. $S1$ cells are modeled as simplified LIF neruons. MAX-like neural competition are performed on $S1$ output spikes to extract $C1$ feature spike pattern. This is achieved by using subsampling and time domain WTA operations. A tempotron spiking neural network is then utilized to classify different $C1$ spike patterns.

III. Feature Extraction by Event-driven Convolution and Neural Competition

Inspired by feedforward models of cortical information processing [4], [8], we propose a two-layer hierarchical con-

Algorithm 1 The flow of information processing in our categorization system

1) $S1$ **layer: on-the-fly convolution and simplified LIF neurons.** Each address event in the AER stream is projected onto a set of Gabor filters to build $S1$ feature maps. The pixel in $S1$ feature maps is modeled as a simplified LIF neuron. Each $S1$ neuron's potential is changing dynamically due to the on-the-fly convolution with a forgetting mechanism. An output spike (event) will be generated if this potential is larger than a threshold. The output event has the same AER format as the input.
2) $C1$ **layer: MAX-like Neural Competition by subsampling and time domain winner-take-all (WTA).** From $S1$ layer to $C1$ layer, a neural competition mechanism that is similar to the max pooling in HMAX is adopted. This is performed by subsampling (merging) the $S1$ spikes, followed by a time domain WTA operation. As a result, only the first (earliest) $S1$ spike in each local neighborhood is fedforward to $C1$ layer. Therefore, $C1$ layer largely reduces the number of output spikes and generates a time-to-first-spike (TFS) pattern.
3) **Classification by a Spiking Neural Network.** The extracted $C1$ feature spike pattern is further fed to a LIF spiking neural network for classification. In this single-layer fully-connected network, each neuron receives afferent spikes from all $C1$ neurons. Spike-based synaptic learning is performed using the tempotron learning rule.

volutional network to extract features from input AER stream. The overall data flow can be summarized as *address events* $\rightarrow$ *S1 layer* $\rightarrow$ *C1 layer*.

A. S1 Layer: On-the-fly Convolution and LIF Neurons

The $S1$ cells are modeled as Leaky integrate-and-fire (LIF) neurons. The "leaky" part is defined in a postsynaptic-potential (PSP) kernel, the "integrate" process is related to the event-driven convolution, and the "fire" action is achieved by thresholding the $S1$ neurons' potentials.

Each input event is convolved on the fly with a set of Gabor filters to build $S1$ feature maps. Each filter models a "neuron" cell with a certain size of receptive field and shows best responses to a basic feature of a certain orientation [24], [25]. The function of Gabor filter can be described as:

$$G(x,y) = \exp\left(-\frac{X^2+\gamma^2Y^2}{2\sigma^2}\right) \times \cos\left(\frac{2\pi}{\lambda}X\right) \quad (1)$$

where $X = x\cos\theta + y\sin\theta$ and $Y = -x\sin\theta + y\cos\theta$. The filter parameters (orientation θ, aspect ratio γ, effective width σ and wavelength λ) have been well tuned in pioneering work [8], and here we adopt a similar set of these parameters. For hardware implementation consideration, we use totally four Gabor filters with two scales (3×3 and 5×5) and two orientations (horizontal and vertical). The parameters of Gabor filters we used are listed in Table I.

TABLE I
PARAMETERS OF GABOR FILTERS

Filter sizes	3×3	5×5
effective width σ	1.2	2.0
wavelength λ	1.5	2.5
aspect ratio γ	0.3	
Orientations θ	0; $\frac{\pi}{2}$	

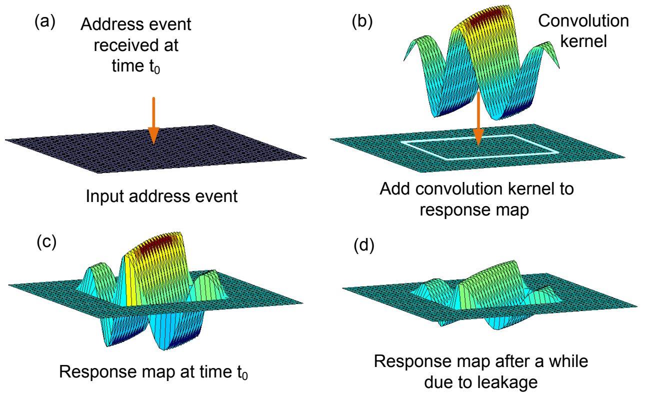
(a)
Address event received at time t_0
Input address event
(b)
Convolution kernel
Add convolution kernel to response map
(c)
Response map at time t_0
(d)
Response map after a while due to leakage

Fig. 2. On-the-fly convolution with a forgetting mechanism.

The event-driven on-the-fly convolution is illustrated in Fig. 2. When an input address event comes in, a convolution kernel is overlaid onto the response map (feature map) at the position specified by the input event's address. Each element of the convolution kernel is then used as a gain (weight) to the event's PSP. The scaled PSP kernels are finally added with corresponding original responses to update the feature map. As seen from Fig. 2(c) and (d), the responses in the feature map decreases (or increases, for negative responses) toward 0 due to the leakage defined in the PSP kernel. Fig. 3 shows an example of the event-driven on-the-fly convolution. Two input events are received at time $100ns$ and $200ns$, respectively. Assume that the responses in convolution map ("conv map") are all 0 at the beginning. The first event, which has an address of (3,3), adds the convolution kernel ("conv kernel") to the "conv map", with the kernel's center aligned at position (3,3) of the map. At time $200ns$, the second input event comes in. The leakage needs to be updated first before adding the kernel to the map at position (2,3).

Note that each input event only affects a small region of $S1$ neurons. The size and position of this region is determined by the scale of convolution kernel and the input event's address, respectively. In other words, each $S1$ neuron can be affected by input events that come from a certain region, which is called the receptive field of this neuron.

The convolution illustrated above contributes to the leaky integration process of $S1$ LIF neurons. In order to "fire" output spikes, a threshold is further needed. During the integration, if the potential of an $S1$ neuron crosses the threshold, then an output spike will be generated. In the meantime, this $S1$ neuron's potential will also be reset.

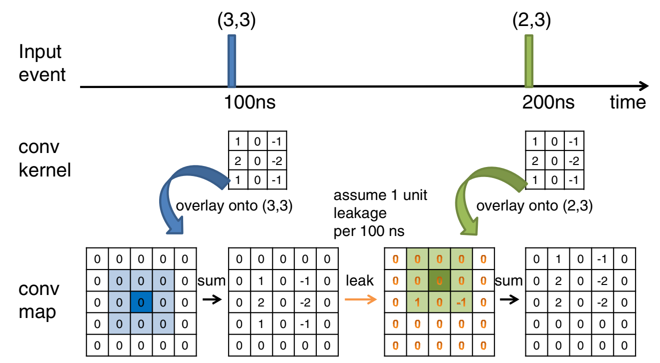

Fig. 3. An example of on-the-fly convolution with a forgetting mechanism.

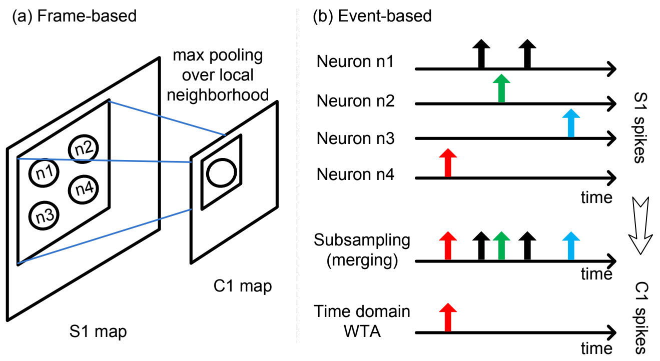

Fig. 5. MAX over local neighborhood. Event-based neural competition (through subsampling and time domain WTA) is equivalent to max pooling over local neighborhood used in the frame-based HMAX algorithm

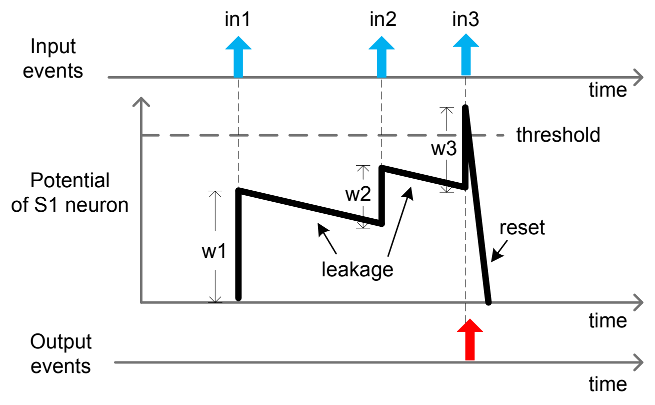

Fig. 4. Dynamics of an LIF neuron in S1 layer.

The dynamics of an $S1$ LIF neuron are illustrated in Fig. 4. Here we only show the input events that come from the receptive filed of this $S1$ neuron. Each input event contributes a certain potential increment (“w1”, “w2”, and “w3” in Fig. 4) determined by corresponding filter value. The leakage factor leads the neuron’s potential towards the resting level. An output spike is fired when the neuron’s potential crosses the threshold. After firing, the neuron resets its potential.

The output $S1$ spikes have the same AER format as the input. Each spike has a timestamp (that records its time of occurrence) and an address (that indicates the firing neuron’s position).

B. $C1$ Layer: MAX-like Competition by Subsampling and Time Domain Winner-Take-All (WTA)

$C1$ cells are further obtained by performing a MAX-like neural competition over simple $S1$ units. As shown in Fig. 5(a), in the frame-based HMAX algorithm [4], a $C1$ cell is calculated by max pooling over $S1$ cells that reside in its receptive field. Max pooling is a form of nonlinear subsampling. The input image is first partitioned into a set of non-overlapping regions. Then, for each region, only the maximum value is fedforward. Max pooling not only reduces subsequent computational complexity but also provides a slight position invariance.

In our event-based architecture, the max pooling computation is achieved by subsampling ($2 \times$ 2-to-1 merging) the $S1$ spikes, followed by a time domain WTA operation. This procedure is illustrated in Fig. 5(b). Assume that “n1”, “n2”, “n3” and “n4” are the four $S1$ neurons located in the receptive field of a $C1$ neuron. First, subsampling is performed by simply merging the spikes from all four afferent $S1$ neurons into one channel. Then, a time domain WTA operation selects only the first spike and inhibits all subsequent ones. As a result, only the earliest $S1$ spike in each local neighborhood is fedforward to $C1$ layer. This event-based max pooling operation largely reduces the number of output spikes. It not only subsamples the feature map, but also converts multiple spikes per channel into a single spike. The final output of $C1$ layer is thus a time-to-first-spike (TFS) pattern.

Fig. 6 visualizes the intermediate results of the feature extraction process (event-driven convolution and competition) for a sample input. A gray level image, which contains a handwritten digit 0, is first converted into a stream of AER events using a rate-coding algorithm [26]. The number of events produced by each pixel is proportional to its gray value. The input AER events are then fed to the convolution module to generate $S1$ output spikes. The second row of Fig. 6 shows the four $S1$ feature maps reconstructed from $S1$ spikes. From left to right, they are corresponding to the feature maps of scale-3 0^o, scale-3 90^o, scale-5 0^o, and scale-5 90^o, respectively. The reconstruction from address events back to an image also follows the rate-coding scheme. The pixel value in the reconstructed image is proportional to the number of events from that address. Note that the background (which means no events happened there) is shown as gray color but not black just for the purpose of better visualization. The third row of Fig. 6 depicts the impacts of event-based subsampling (merging) on the four feature maps. In the fourth row, the $C1$ TFS patterns after time domain WTA are illustrated. Here the gray levels represent $C1$ spikes’ times of occurrence. The earlier the spike occurs, the smaller the spike time is, and thus the darker it is shown in the image. Finally, the spatiotemporal spike pattern in the last row of Fig. 6 shows another illustration

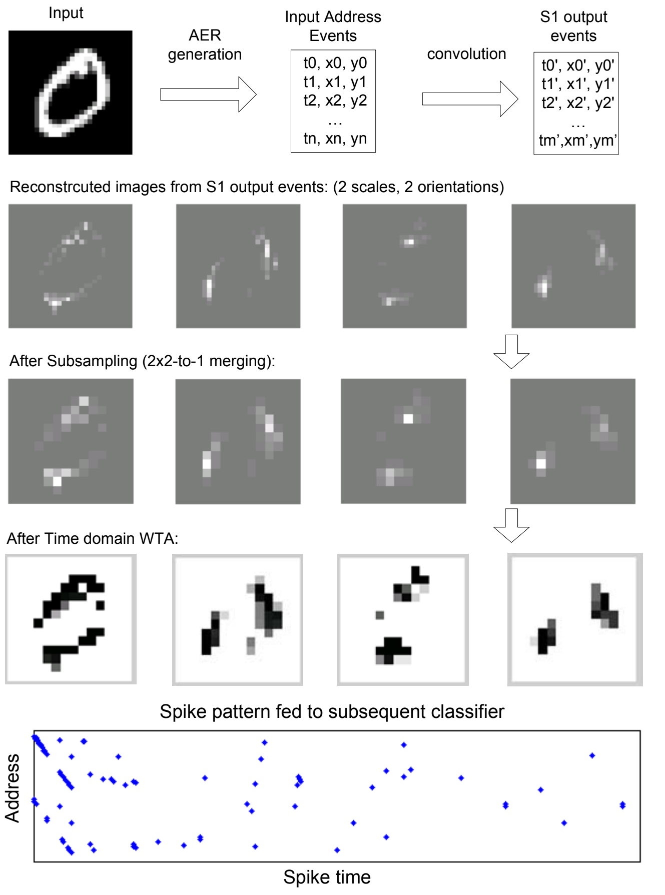

Fig. 6. The feature extraction process (convolution and competition) for a sample input. See the context for more illustrations.

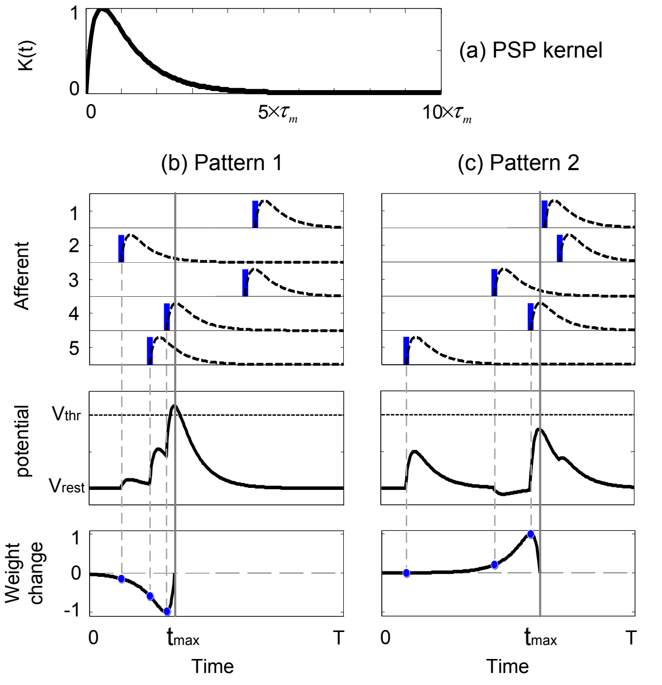

Fig. 7. The dynamics and learning rule of tempotron neuron. (a) shows the PSP kernel. (b) and (c) illustrate the operations of tempotron using two spatiotemporal patterns. The vertical thick bars stand for spikes, and the dash curve beside each bar denotes the PSP kernel generated by corresponding spike. For $pattern1$ in (b), the total potential crosses the threshold, which means the neuron would fire for this input. If this is an error (the neuron should not fire for this input), then we find all the spikes before t_{max} and decrease the weights of corresponding afferents. $Pattern2$ in (c) does not make the neuron fire, if this is an error, the weights of those afferents which have spikes before t_{max} will be increased. Note that the curve of weight change is just the mirror of PSP kernel.

of $C1$ spikes.

IV. Classification by a Spiking Neural Network

In this section, we will illustrate how we perform classification on extracted $C1$ feature spikes using a network of tempotron spiking neurons. Tempotron is a supervised temporal learning model that allows a spiking neuron to efficiently discriminate spatiotemporal spike patterns [15]. It utilizes spike time information and integrates postsynaptic potentials from afferent spikes with different addresses. These properties make tempotron by nature a perfect match for our extracted feature spikes.

A. Tempotron Learning Rule

Tempotron uses the LIF neuron model. Each input spike initiates a PSP kernel which has an exponential decaying shape. For an input event received at time t_i, the normalized PSP kernel K is defined as:

$$K(t-t_i) = V_0 \times (\exp(\frac{-(t-t_i)}{\tau_m}) - \exp(\frac{-(t-t_i)}{\tau_s})) \quad (2)$$

where τ_m and τ_s denote decay time constants of membrane integration and synaptic currents. For simplicity, τ_s is set to be $\tau_m/4$. V_0 normalizes PSP so that the maximum value of the kernel is 1.

The neuron's total membrane potential is the weighted summation of PSP from all input spikes:

$$V(t) = \sum_i \omega_i \sum_{t_i} K(t-t_i) + V_{rest} \quad (3)$$

where ω_i and t_i are the synaptic efficacy and the firing time of the ith afferent, respectively. V_{rest} is the resting potential of the neuron. K denotes the normalized PSP kernel.

If the neuron's potential is higher than a specified threshold, the neuron will fire an output spike. After firing, the neuron shunts all the following input spikes and the potential decreases to the resting level. In other words, the spikes arrive after the firing time have no impact on the postsynaptic potential any more. If the membrane potential fails to cross the threshold then the neuron will not fire.

The tempotron learning rule aims to train the weights of afferent synapses so that the output neuron can fire or not according to its class label. If the neuron is supposed to fire (or not fire, on the other hand) but it actually fails to do so (or does fire, vice versa), then the weights of afferent synapses should be modified in the following way: first, we find the peak potential during the effective period and label the corresponding time stamp as t_{max}; second, update the weights using the following equation:

$$\Delta\omega_i = \lambda \sum_{t_i < t_{\max}} K(t_{\max} - t_i) \tag{4}$$

B. Classification Strategy

For an N-class categorization task, a network of N tempotron neurons is adopted. During the training process, the output targets of these N tempotron neurons are labeled using a one-hot coding scheme, with "1" standing for firing and "0" for not firing. For example, in a three-class categorization problem, for spike patterns that belong to the first, the second, and the third categories, the output targets are defined as 001_b, 010_b, and 100_b, respectively. During testing, the decision making for each input pattern is easy. One just needs to check which neuron has fired.

There are chances that multiple neurons fire together or even no neuron fires at all. Therefore, this coding scheme is susceptible to the fluctuation existing in a single neuron's signal. In order to make the system immune from such fluctuations, the population coding scheme can be adopted. Population coding is a method to represent the stimuli by using the joint activities of a number of neurons [27]. Each neuron in the population has a distribution of responses over the inputs, and the responses of many neurons can be combined to determine the final output. In our categorization problem, we can use a population of M neurons instead of only one for each category. Since the initial synaptic weights are set randomly, these neurons will eventually have different synaptic weights after training. We then use a majority voting method to make the final decision: to check which category has the largest number of firing neurons [28]. For example, we can use 10 neurons for each category in a three-class categorization problem. Assume that the numbers of firing neurons in each category are 2, 8, and 3, respectively. Then, the input pattern will be classified to the second category since the second number 8 is the largest.

Besides using the firing status, we can also use the potentials (P) of tempotron neurons to make the decision, where P is the peak potential throughout the integration period of a tempotron neuron. In the case of single neuron coding per category, the decision is made by checking which output neuron has the largest P value. As for the population coding, a mean value $\bar{P}$ is first calculated for each population of neurons, then the population that has the largest $\bar{P}$ determines which category the input pattern should belong to.

V. Experimental Results

A. Datasets

The proposed event-driven categorization system has been evaluated on three datasets. The first one is the MNIST hand-written digit dataset [29]. It has ten groups (from digit 0 to digit 9) of gray level images. Each image has a resolution of 28×28. Some sample images of MNIST dataset are shown in Fig. 8. We use a total number of 500 images for evaluation, with 50 images per group. Each image has to be converted into an AER stream before being fed into our system. As mentioned earlier in the example of Section III-B, the image-to-AER conversion adopts a rate-coding algorithm [26]. The number of events produced by each pixel is proportional to its gray value.

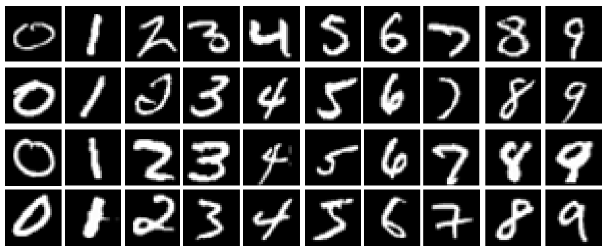
Fig. 8. Some sample images from MNIST hand-written digit dataset.

The second dataset is a posture dataset [24]. It has six groups of postures, namely "bend", "hand1", "hand2", "squat", "stand", and "swing". These binary images are captured by a temporal-different image sensor, with "1" standing for a moving foreground pixel and "0" for the static background. Fig. 9 shows some sample images of this posture dataset. A total number of 300 images are used, with 50 images per group. Each image has a resolution of 64×64. Similar image-to-AER conversion algorithm is performed on these images.

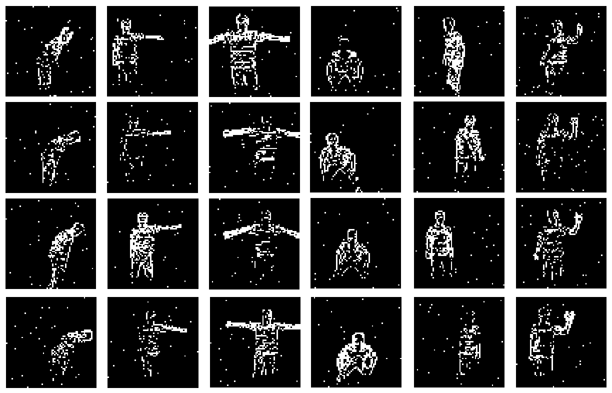
Fig. 9. Some sample images from the posture dataset. There are six kinds of postures: bend, hand1, hand2, squat, stand, and swing.

The third dataset is a poker card symbol dataset [22]. It contains four types of poker card symbols, namely "club", "diamond", "heart", and "spade". These card symbols, in the data form of raw AER streams, were captured by a temporal contrast AER vision sensor [19]. These AER data have a spatial resolution of 32×32. They can be directly processed by our event-driven categorization system. Some reconstructed images from these AER streams are shown in Fig. 10. There are totally 100 pieces of AER stream, with 25 streams per category.

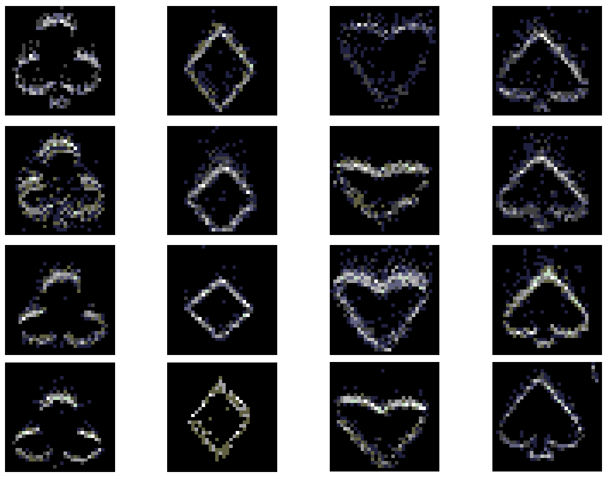

Fig. 10. Some reconstructed images from raw AER streams in the poker card symbol dataset. Four types of symbols: club, diamond, heart, and spade.

B. Number of Tempotron Neurons per Category

As mentioned earlier in Section IV-B, using a population of tempotron neurons for each category makes the system immune from fluctuations of a single neuron's signal. In order to determine the size of this population, we run our categorization system on the MNIST dataset several times, with each time using a different number of neurons per group. The results are reported in Fig. 11. We finally choose to use 5 neurons per group.

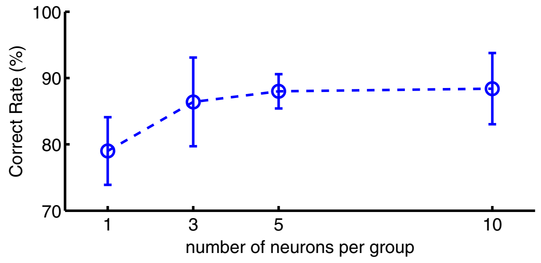

Fig. 11. Performance versus number of tempotron neurons per group, on MNIST dataset. Neurons' firing statuses and the majority voting scheme are used to make the decision. We finally choose to use 5 neurons per group.

C. Performance

We have evaluated the performance of the proposed algorithm on the aforementioned three datasets. For each dataset, we randomly selected 90% for training and the remaining 10% were used for testing. By repeating this evaluation process ten times, we obtained the average performances. Population coding was used, with 5 tempotron neurons for each category. We tried two classification strategies: 1) using the firing status of tempotron neurons and the majority voting scheme to make the decision, and 2) using the mean potential $\bar{P}$ of tempotron neurons for decision making. The results (correct rates for testing) are summarized in Table II. Note that the second strategy (using potential) achieves better performances than the first one (using firing status), especially for the poker card symbol dataset. The reason could be as follows. Some neurons are trained to fire for specific input patterns. Due to the input variances, these neurons may fail to fire for a testing pattern. However their potentials may be already very close to the threshold. Therefore, using the potential information of tempotron neurons for decision making can generally improve the correct rate.

TABLE II
PERFORMANCES OF THE PROPOSED ALGORITHM ON THREE DATASETS

classification strategy	Correct Rate: mean ± std (%)		
	MNIST	posture	poker card
1) using firing status	88.00 ± 2.58	96.00 ± 3.65	81.67 ± 8.61
2) using potential	89.14 ± 1.57	97.33 ± 2.79	95.83 ± 5.89

We also compared our approach with other biologically inspired algorithms: the original HMAX scheme (HMAX'99) [4], the model by Serre et. al. (Serre'07) [8], and the algorithm by Chen et. al. (Chen'12) [24]. The first two models, HMAX'99 and Serre'07, use the Support Vector Machine (SVM) as the classifier. To perform multi-class categorization, the one-versus-one (OVO) SVM scheme [30] is employed. The third algorithm, Chen'12, extracts line features and uses a nearest neighbor classifier based on line segment Hausdorff distance [24]. The comparison results for all three datasets are shown in Fig. 12. For MNIST and posture dataset, the proposed system, wether using firing status or potential for decision making, has better performance than others. For the poker card symbol dataset, the proposed system has a lower performance than others when using firing status for classification, while it still has the best performance when using the potential.

VI. Conclusion

This paper presents a feedforward categorization system on AER stream. The system uses event-based processing at every computing stage. Cortex-like features are extracted by a two-layer hierarchical convolutional network. Due to the MAX-like competition, features are encoded into a limited number of feature spikes. A tempotron spiking neural network efficiently discriminates the feature spike patterns. Promising results have been achieved on three datasets. Future work includes the design and implementation of algorithms for categorizing continuous AER motion events without periodic reset as required in current stimulus onset scenario.

References

[1] L. G. Ungerleider and J. V. Haxby, "'what' and 'where' in the human brain," *Current Opinion in Neurobiology*, pp. 157–65, 1994.
[2] S. Thorpe, D. Fize, and C. Marlot, "Speed of processing in the human visual system," *Nature*, vol. 381, pp. 520–522, june 1996.
[3] T. Serre, "Learning a dictionary of shape-components in visual cortex: Comparison with neurons, humans and machines," Ph.D. dissertation, MIT, Apr. 2006.
[4] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, 1999.

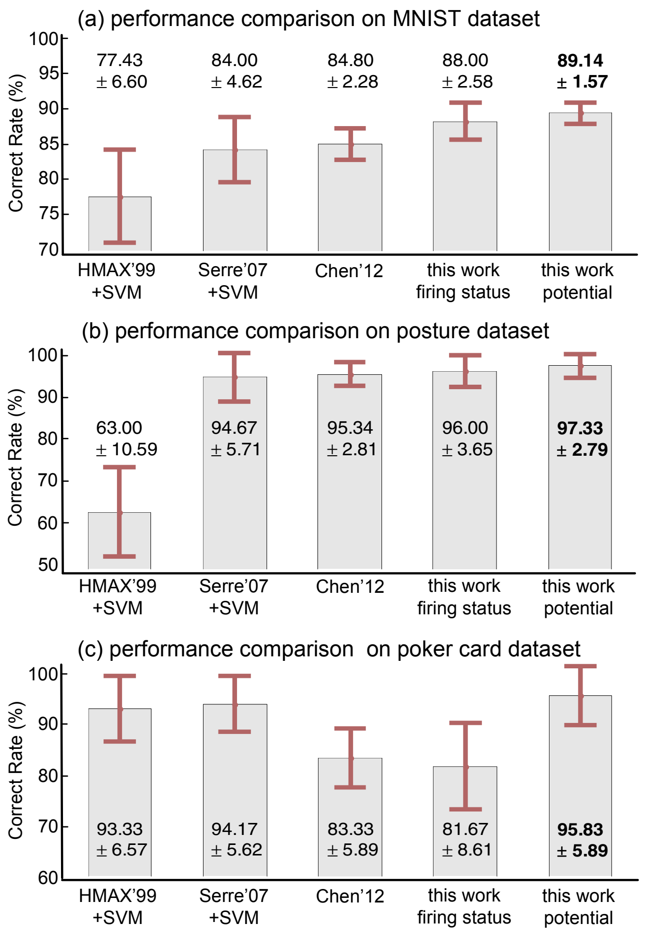

Fig. 12. Performance comparison on three datasets. For MNIST and posture dataset, the proposed system, wether using firing status or potential for decision making, has better performance than others. For the poker card symbol dataset, the proposed system has a lower performance than others when using firing status for classification, while it still has the best performance when using the potential.

[5] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architectures in cats visual cortex," *Journal of Physiology*, vol. 160, pp. 106–154, 1962.

[6] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[7] T. Serre, L. Wolf, and T. Poggio, "Object recognition with features inspired by visual cortex," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR'05)*, June 2005, pp. 994–1000.

[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.

[9] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, 1st ed. Cambridge University Press, Aug. 2002.

[10] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, pp. 500–544, Aug. 1952.

[11] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003.

[12] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Spike timing dependent plasticity finds the start of repeating patterns in continuous spike trains," *PLoS ONE*, vol. 3, no. 1, 2008.

[13] T. Masquelier and S. J. Thorpe, "Unsupervised learning of visual features through spike timing dependent plasticity," *PLoS Comput Biol*, vol. 3, no. 2, p. e31, Feb. 2007.

[14] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Competitive STDP-based spike pattern learning," *Neural computation*, vol. 21, no. 5, pp. 1259–1276, May 2009.

[15] R. Gutig and H. Sompolinsky, "The tempotron: a neuron that learns spike timing-based decisions," *Nature Neuroscience*, vol. 9, no. 3, pp. 420–428, Feb. 2006.

[16] F. Ponulak and A. Kasiński, "Supervised learning in spiking neural networks with ReSuMe: Sequence learning, classification, and spike shifting," *Neural Computation*, vol. 22, no. 2, pp. 467–510, Feb. 2010. [Online]. Available: http://dx.doi.org/10.1162/neco.2009.11-08-901

[17] K. Boahen, "Point-to-point connectivity between neuromorphic chips using address events," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 5, pp. 416–434, 2000.

[18] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128$\times$128 120dB 15μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, feb. 2008.

[19] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128 $\times$ 128 1.5% contrast sensitivity 0.9% fpn 3 μs latency 4 mw asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 3, pp. 827–838, Mar. 2013.

[20] 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*, Sep. 2006, pp. 173–178.

[21] R. Serrano-Gotarredona, T. Serrano-Gotarredona, A. Acosta-Jimenez, C. Serrano-Gotarredona, J. Perez-Carrasco, B. Linares-Barranco, A. Linares-Barranco, G. Jimenez-Moreno, and A. Civit-Ballcels, "On real-time AER 2D convolutions hardware for neuromorphic spike based cortical processing," *IEEE Trans. Neural Networks*, vol. 19, no. 7, pp. 1196–1219, Jul 2008.

[22] 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*, vol. 35, pp. 2706 – 2719, Nov. 2013.

[23] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS'10)*, june 2010, pp. 253 –256.

[24] 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.

[25] B. Zhao and S. Chen, "Realtime feature extraction using max-like convolutional network for human posture recognition," in *IEEE ISCAS'11*, May 2011, pp. 2673–2676.

[26] A. Linares-Barranco, G. Jimenez-Moreno, B. Linares-Barranco, and A. Civit-Balcells, "On algorithmic rate-coded aer generation," *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 771–788, 2006.

[27] S. Wu, S. ichi Amari, and H. Nakahara, "Population coding and decoding in a neural field: A computational study." *Neural Computation*, vol. 14, no. 5, pp. 999–1026.

[28] Q. Yu, H. Tang, K. C. Tan, and H. Li, "Rapid feedforward computation by temporal encoding and learning with spiking neurons," *IEEE Trans. Neural Netw. Learning Syst.*, pp. 1539–1552, Oct. 2013.

[29] Y. LeCun and C. Cortes, "The MNIST database of handwritten digits," URL: http://yann.lecun.com/exdb/mnist/, 2012.

[30] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: A stepwise procedure for building and training a neural network," in *Neurocomputing: Algorithms, Architectures and Applications*, ser. NATO ASI Series. Springer-Verlag, 1990, vol. 68, pp. 41–50.