Deep Spiking Neural Network Using Spatio-temporal Backpropagation with Variable Resistance

Xianglan Wen
College of Computer Science
Sichuan University
Chengdu, China
wwwenxla@163.com

Pengjie Gu
College of Computer Science
Sichuan University
Chengdu, China
gupj1202@gmail.com

Rui Yan
College of Computer Science
Zhejiang University of Technology
Hangzhou, China
ryan@zjut.edu.cn

Huajin Tang
College of Computer Science and Technology
Zhejiang University
Hangzhou,China
htang@zju.edu.cn

***Abstract*—In recent years, the learning of deep spiking neural networks(SNN) has attracted increasing researchers' interest, and has also made important progresses in theories and applications. It is desired to choose a neuron model with biological features and suitable for SNN training. Currently, Leaky Integrate-and-Fire(LIF) model is mainly used in deep SNN and some factors that can express the spatio-temporal information are ignored in the model. In this work, inspired by the Hodgkin-Huxley(H-H) model, we propose an improved LIF neuron model, which is an iterative current-based LIF model with voltage-based variable resistance. The improved neuron model is closer to the characteristics of the biological neuron model, which can make use of the spatio-temporal information. We further construct a new SNN learning algorithm that uses spatio-temporal back propagation by defining a loss function. We evaluated the proposed methods on single-label and multi-label data sets. The experimental results show that the variable resistance of the neuron model will affect the performance of the model. Choosing the appropriate relationship between the variable resistance and the membrane voltage can effectively improve the recognition accuracy.**

***Index Terms*—neuromorphic computing, spatio-temporal, backpropagation, spiking neural network**

I. Introduction

Neuromorphic computing is one of the frontier fields. It is a set of technologies that simulates the biological neural system to improve the robustness of computing system and hardware efficiency. In the neuromorphic computing, spikes are believed to transmit and process information [1]–[3]. Currently, some different models are proposed to emulate the neuron's behavior in brain, such as Hodgkin-Huxley(H-H) model [4], Leaky Integrate-and-Fire(LIF) model [1], Izhikevich model [5], Spiking Response Model(SRM) [6], etc..

This work was supported by the National Natural Science Foundation of China under grant 61673283, and the Major Scientific Research Project of Zhejiang Lab(No. 2019KC0AD02).

Inspired by the biological neuron, H-H model describes the electrical behavior in the membrane of giant squid axons and most biological spiking neuron models are based on it [5], [7]. In H-H model it is proposed that there are three different kinds of ion current in the membrane, including sodium current, potassium current and a leak current. The conductance of sodium and potassium is inconstant and related to the membrane potential. H-H model considers the membrane as a capacitor [2], and it can be simulated by the equivalent circuit diagram shown in Fig. 1(a). Although this model can keep the characteristics of the biological neuron, its computational complexity limits applications [8]. As a simplified model, LIF model has been widely used to simulate the large-scale neural networks [9]. LIF model integrates the presynapses' spikes. When the membrane potential is up to the threshold, the neuron will fire and reset potential to resting potential [10]. Fig. 1(b) shows the equivalent circuit diagram.

Compared with traditional Artificial Neural Networks(ANN), Spiking Neural Networks(SNN) based on spiking neuron model are more biological and have better power efficiency in neuromorphic hardware [11]–[13]. The discrete and binary spike trains can be regarded as the events happening on precise time, so SNN can capture the temporal dynamics of neural behaviors. Although there exist many models [14]–[16], the training of deep SNNs using backpropagation is still difficult because of the complex temporal dependencies and non-differentiable of spike activity [17]. Now there are three main categories to train deep SNNs. The first is ANN-to-SNN, which transforms the trained ANN into the same structure's SNN, usually based on LIF neurons [18]–[21]. The conversion process inevitably causes the loss of accuracy, and some approaches have also been proposed to improve the accuracy, such as scaling weights [18], [19], defining equivalent transfer function [20] and so on. However, in the transformation approach the rate-based encode methods without temporal features are only supported [22], which limits its applications in dynamic tasks. The second approach is incomplete-BP category by ignoring some information in backpropagation. For example, SpikeProp only considers the backpropagate error at spike times [23], [24], and the temporal dependencies of spiking neurons is ignored in [25], [26]. Thus, in these kinds of algorithms, the spatio-temporal features cannot be extracted efficiently. The third way is spatio-temporal-BP category, which approximates the derivative of spike function to overcome its non-differentiability and backpropagates error both in the spatial field and temporal field. In this category, SLAYER [17] based on SRM neurons uses the Probability Density Function to approximate the derivative. STBP [27] proposes different derivative functions to approximate and combines spatial dependencies and temporal dependencies to backpropagate, and it has high performance without any training skills. Inspired by STBP, STCA [22] changes the loss function to voltage-driven function and can achieve the state-of-the-art algorithms. Existing experiments show that this category has the best performance.

For many of SNN algorithms, they are based on different LIF neurons, such as the simple iterative LIF models [20], [27], [28], current-based LIF(C-LIF) models [22], [29]–[31], and threshold-adaptive LIF models [32]. In these kinds of LIF models, the resistance is considered as a constant. Compared with the H-H model in Fig.1 (a), we can discover that the LIF model merges three types of current and regards their resistances as a resistance. Thus the resistance is non-constant, which actually includes the spatio-temporal information. In this work, we firstly propose an iterative C-LIF model with voltage-based variable resistance. This model is closer to the characteristics of the biological neuron model, which can make use of the spatio-temporal information. Then we further construct a new deep SNN algorithm that uses spatio-temporal backpropagation by defining a temporal-like loss function, which is a voltage-driven loss function like the algorithm Tempotron [28]. The proposed algorithm can capture the spatio-temporal features from spike trains. Due to the variable resistance, our deep SNN is non-linear weighted. To evaluate the influence and the performance of the proposed methods, we do some experiments both on single-label(MedleyDB) and multi-label(MAPS) data sets, which contain spatio-temporal features. These experiments demonstrate that variable resistance has different influences in the performance, the recognition accuracy can be effectively improved by choosing the appropriate relationship between the variable resistance and the membrane voltage, and compared with other methods, the proposed algorithm shows the state-of-the-art performance.

II. Methods

In this section, we will describe the approach in detail. Firstly, we propose the spiking neuron model: the iterative C-LIF model with voltage-based variable resistance. Then, we define the tempotron-like loss function and show the deep SNN structure with the proposed spiking model. we also display the iterative derivation process using error backpropagation in the spatio-temporal field.

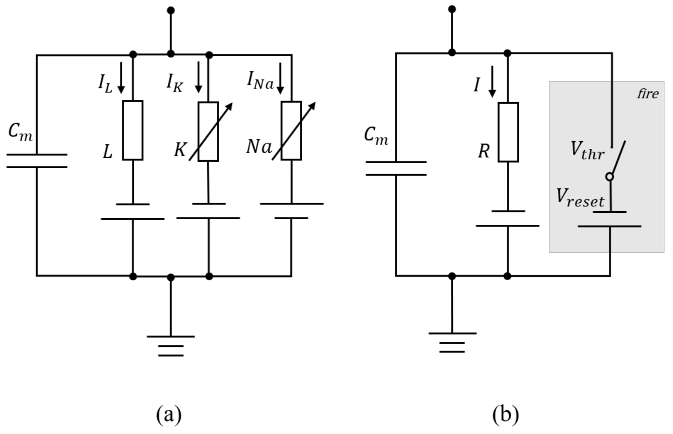

Fig. 1. The diagram of spiking neuron models.(a)The diagram of Hodgkin-Huxley model. It has three types of current corresponding to three resistances, two of them is voltage-based variable resistances. (b)The diagram of Leaky Integrate-and-Fire model. It has a resistance, and a fire mechanism(shows in the gray frame: when the voltage up to V_{thr}, the switch off, the voltage will be equal to V_{reset}).

A. The Spiking Neuron Model

Our model is based on C-LIF model [30]. The membrane voltage of this model is affected by decay and spiking inhibition. The membrane voltage $U(t)$ in the model can be expressed as:

$$\begin{aligned} U(t) =& N_0 * \sum_{j=1}^{N} W_{ij} \sum_{t_j^s<t} exp(-(t-t_j^s)/\tau_m) * R \\ & -N_0 * \sum_{j=1}^{N} W_{ij} \sum_{t_j^s<t} exp(-(t-t_j^s)/\tau_s) * R \\ & -V_{thr} \sum_{t_i<t} exp(-(t-t_i)/\tau_m) \end{aligned} \tag{1}$$

where t_j^s means the time of s-th spike in j-th presynaptic neuron and t_i is the i-th spike in this neuron; τ_m and τ_s are the decay constants; R is the membrane resistance; V_{thr} denotes the threshold of the neuron; N_0 is a normalization factor, follow as:

$$N_0 = \frac{1}{f_\tau - 1} * f_\tau^{\frac{f_\tau}{f_\tau - 1}} \tag{2}$$

where F_τ is a factor of τ_m and τ_s.When $U(t)$ is up to V_{thr}, the neuron will fire.

In the traditional C-LIF model, R is regarded as constant and $R = 1M\Omega$. From the formula of the H-H model, we can discover that variable resistance is related to the membrane voltage. To reduce computational complexity and keep up with the changes in LIF model(the change of membrane voltage decreases with the increase of membrane voltage.), the current variable resistance $R(t)$ can be represented as:

$$R(t) = \left(1 - \alpha \frac{U(t-1)}{V_{thr}}\right) \cdot R_0 \tag{3}$$

where $U(t-1)$ is membrane voltage at the last time; α is constant, which controls the relationship between $R(t)$ and $U(t-1)$, and avoids the condition that the neuron is hard to fire when the membrane approaches the threshold; R_0 is the fixed resistance.

To build deep SNN, the $R(t)$ should be included in the iterative C-LIF model, which is presented to capture the spatio-temporal features according to the event-driven property of C-LIF model [22]. The iterative C-LIF model with voltage-based variable resistance is shown as:

$$T_n^i(t) = \left[N_0 \sum_{j=1}^{L(n-1)} W_n^{ij} \cdot P_{n-1}^j(t)\right]\left(1 - \alpha \frac{U_n^i(t-1)}{V_{thr}}\right) R_0, \tag{4}$$

$$Dm_n^i(t) = \gamma_m Dm_n^i(t-1) + T_n^i(t), \tag{5}$$

$$Ds_n^i(t) = \gamma_s Ds_n^i(t-1) + T_n^i(t), \tag{6}$$

$$Re_n^i(t) = \gamma_m E_n^i(t-1) + V_{thr} \cdot P_n^i(t-1), \tag{7}$$

$$U_n^i(t) = Dm_n^i(t) - Ds_n^i(t) - Re_n^i(t), \tag{8}$$

$$P_n^i(t) = S(U_n^i(t)), \tag{9}$$

where $U_n^i(t)$ denotes the membrane voltage at time t of the i-th neuron in n-th layer; γ_m and γ_s are the decay constants, which is associated to τ_m and τ_s ; $P_n^i(t)$ is the binary spike train, and $S(x)$ is the spike function: $S(x) = 1$ (if $x > V_{thr}$) or 0 (otherwise). The relations in $U_n^i(t)$ and R_0 are the same as in $U_n^i(t)$ and W_n. If R_0 is changed, scaling W_n without training still keeps the result, for simple computation, $R_0 = 1M\Omega$.

There is a simple condition in Fig. 2. We can find that the same input spike trains can get the different output in our model and C-LIF model. In C-LIF model, if the spikes have same synpase strength, each spike has the same influence in membrane voltage. In our model, due to variable resistance, the influence of input spike is not same for different membrane voltage and same synapse strength. When the membrane voltage is not equal to 0, the change in membrane voltage caused by input spike in our model is less than that of C-LIF model, and the degree of this change is associated with the distance in membrane voltage and threshold. The closer the voltage is to the threshold, the less influence the input spike has.

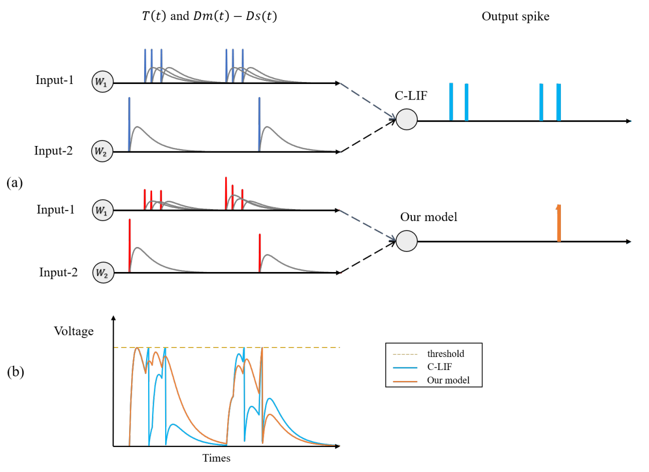

Fig. 2. A simple comparison of C-LIF and our model. (a)The condition of two models. The top is C-LIF model, and the bottom is our model. Two models receive the same spike trains, through the same synapse strength of two input neuron, they have different $T(t)$, $Dm(t) - Ds(t)$, and output spike. (b)The membrane voltage of the two models.

B. The Algorithm of Deep Spiking Neural Network

Tempotron is the classical algorithm in monolayer SNN. When training, it does not spike and change the synaptic strength according to the distance in max voltage and threshold, it is a voltage-driven algorithm and has good performance. Here, we bring the training idea and define a Tempotron-like Loss Function, which can be expressed as:

$$L = \begin{cases} V_{thr} - V_{max}, & \text{if } target = 1 \ \& \ V_{max} < V_{thr} \\ -(V_{thr} - V_{max}), & \text{if } target = 0 \ \& \ V_{max} > V_{thr} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where $V_{max} = U_N(t^*)$, and t^* is the time of max membrane voltage of a neuron in the last layer. There are two conditions that cause loss in this loss function. The first condition indicates that the neuron spikes and reaches the desire one but the V_{max} is not up to the threshold in actual. The second condition suggests that the V_{max} is up to the threshold in actual but the neuron does not spike in desire. Due to the last layer does not fire when training, the membrane voltage is different, which is expressed as:

$$U_N^i(t) = Dm_N^i(t) - Ds_N^i(t), \tag{11}$$

The spike function $S(x)$ is non-differentiable. Its derivation only has an infinite value (when it is up to threshold) and 0 (otherwise). To derivation, we can approximate the derivative of it by $g(x)$. the feature of $g(x)$ is that it only has values in an interval closed to threshold, which can replace the infinite value. There are many choices of $g(x)$, such as the derivative of the rectangular function, sigmoid function, or probability density function. Here, for simplicity computation, we use the rectangular function, its derivative follows as:

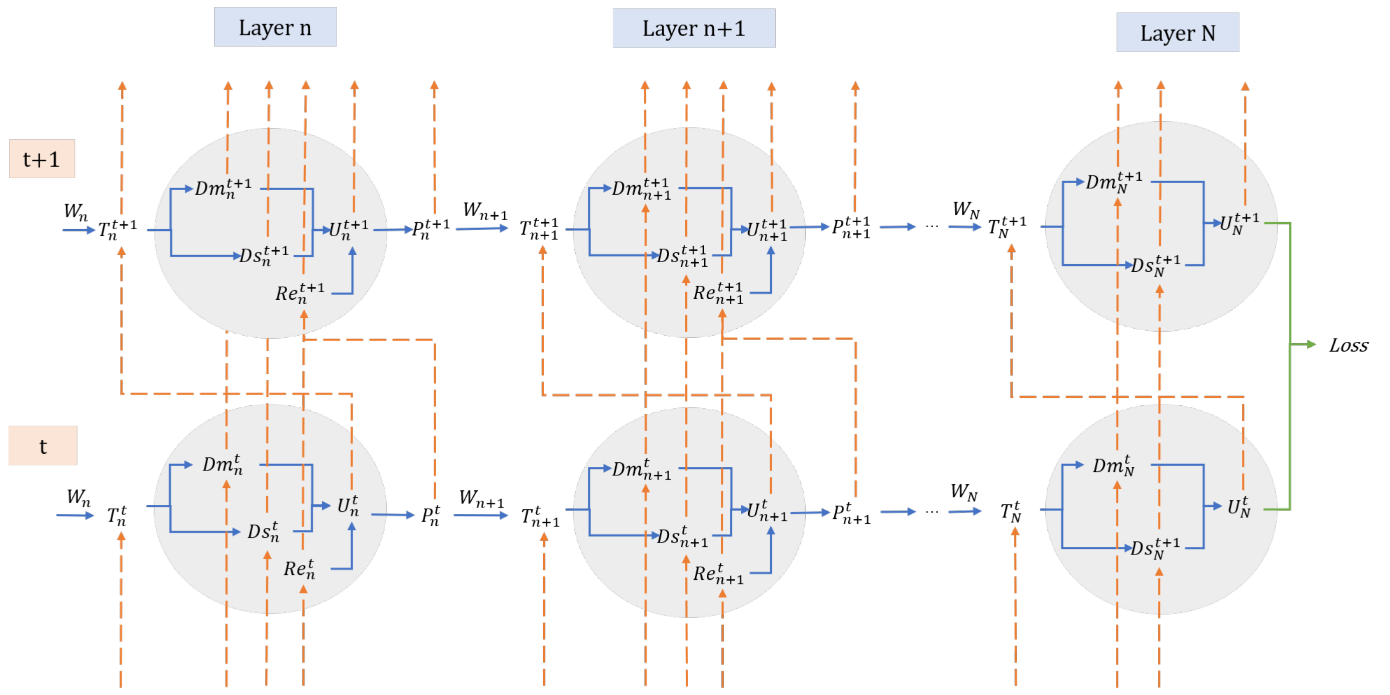

Fig. 3. The spatio-temporal dependencies of the algorithm. Here shows two time points of t and $t+1$, layer N is the last layer. Each gray circle represents a neuron, including voltage and its directive variable. There is mainly two types of lines: the orange dashed line represents the temporal dependencies, and the blue full line represents the spatial dependencies. All of dependencies described by the related equations can be found.

$$g(x) = \begin{cases} \frac{1}{q}, & \text{if } |x-\theta| < \frac{q}{2} \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

Based on the Tempotron-like loss function and the iterative C-LIF model with variable resistance, we can construct a new algorithm of deep SNN, the spatio-temporal dependencies of the algorithm is showed in Fig. 3.

The Tempotron-like loss function has a feature: it only backpaprogate error when $t < t*$ in temporal field. Due to the W_n affects L through T_n at each time step when $t < t^*$, the gradient of W_n is as follow:

$$\nabla W_n = \sum_{t=0}^{t=t^*} \frac{\partial L}{\partial T_n^t} \cdot \frac{\partial T_n^t}{\partial W_n}. \tag{13}$$

In order to a neat layout, we denote the subscript n as the layer of neuron and the superscript t as time in these parts. Because the relation of ∂T_n^t and ∂W_n is distinct by (4) - (9) . The gradient can divided into two part:

a) $\frac{\partial T_n^t}{\partial W_n}$: It can be calculated using the derivative rule, we can get:

$$\frac{\partial T_n^t}{\partial W_n} = -\frac{\alpha}{V_{thr}} \cdot \frac{\partial U_n^{t-1}}{\partial W_n} \cdot N_0 W_n P_{n-1}^t + \left(1 - \alpha \cdot \frac{U_n^{t-1}}{V_{thr}}\right) \cdot N_0 P_{n-1}^t \tag{14}$$

and

$$\frac{\partial U_n^{t-1}}{\partial W_n} = \left[\gamma_m \frac{\partial Dm_n^{t-2}}{\partial W_n} + \frac{\partial T_n^{t-1}}{\partial W_n}\right] - \left[\gamma_s \frac{\partial Ds_n^{t-2}}{\partial W_n} + \frac{\partial T_n^{t-1}}{\partial W_n}\right] - \left[\gamma_m \frac{\partial Re_n^{t-2}}{\partial W_n} + V_{thr} \cdot g(U_n^{t-2}) \frac{\partial U_n^{t-2}}{\partial W_n}\right] \tag{15}$$

When iteration begins from t= 0, all of the derivative can be calculated.

b) $\frac{\partial L}{\partial T_n^t}$: It can be calculated using the iterative mechanism and the chain rule to derive the gradients recursively. By combining with Fig.3, we can find that the T_n^t affects L through Dm_n^t and Ds_n^t. So we have:

$$\frac{\partial L}{\partial T_n^t} = \frac{\partial L}{\partial Dm_n^t} + \frac{\partial L}{\partial Ds_n^t}. \tag{16}$$

Since the Dm_n^t affects L through Dm_n^{t+1} and U_n^t, we have:

$$\frac{\partial L}{\partial Dm_n^t} = \frac{\partial L}{\partial Dm_n^{t+1}} \cdot \gamma_m + \frac{\partial L}{\partial U_n^t} \tag{17}$$

.

In the similar way, we can get $\frac{\partial L}{\partial Ds_n^t}$ and $\frac{\partial L}{\partial Re_n^t}$.

As for $\frac{\partial L}{\partial U_n^t}$, there are different values for different layers:

- In the last layer N, when $t < t^*$, because of the relation between the variable resistance and voltage, L can be influenced by U_N^t through T_N^{t+1}, so the derivative can be denoted as:

$$\frac{\partial L}{\partial U_n^t} = \frac{\partial L}{\partial T_N^{t+1}} \cdot \left(-\frac{\alpha}{V_{thr}} \cdot N_0 W_N P_{N-1}^t\right) \tag{18}$$

When $t = t^*$, it is obvious that the derivative is 1.

- In other layer n, due to the variable resistance, L can be influenced by U_n^t through T_n^{t+1}, and due to the firing influence, L can be influenced by U_n^t through P_n^t, the derivative can be denoted as:

$$\begin{aligned}\frac{\partial L}{\partial U_n^t} &= \frac{\partial L}{\partial T_n^{t+1}} \cdot \frac{\partial T_n^{t+1}}{\partial U_n^t} + \frac{\partial L}{\partial P_n^t} \cdot \frac{\partial P_n^t}{\partial U_n^t} \\ &= \frac{\partial L}{\partial T_n^{t+1}} \cdot \left(-\frac{\alpha}{V_{thr}} \cdot N_0 W_n P_{n-1}^t\right) + \frac{\partial L}{\partial P_n^t} \cdot g(U_n^t)\end{aligned} \tag{19}$$

Because P_n^t can affect L through Re_n^{t+1} and T_{n+1}^t, we have:

$$\begin{aligned}\frac{\partial L}{\partial P_n^t} &= \frac{\partial L}{\partial Re_n^{t+1}} \cdot \frac{\partial Re_n^{t+1}}{\partial P_n^t} + \frac{\partial L}{\partial T_{n+1}^t} \cdot \frac{\partial T_{n+1}^t}{\partial Dm_n^t} \\ &= \frac{\partial L}{\partial Re_n^{t+1}} \cdot V_{thr} + \frac{\partial L}{\partial T_{n+1}^t} \cdot \left(1 - \alpha \cdot \frac{U_n^{t-1}}{V_{thr}}\right) \cdot N_0 \cdot W_n\end{aligned} \tag{20}$$

When iteration begins from $t = t^*$, all of the derivative can be calculated.

Up to now, all of the derivative can be calculated in the proper iterative order, and we can get the gradient of W_n. The Algorithm 1 describes the training procedure of our algorithm in a simple way with binary target and represents some special values.

III. Experiment

To demonstrate our model's performance, we test it in different datasets and different parameters. Here, we divide into two parts: the first is the experiment in single-label's dynamic music dataset(MedleyDB); the second is the experiment in multi-labels' music dataset(MAPS).

A. Experiment on Instrument Recognition

The MedleyDB dataset [33] is a dataset of annotated multi-track recordings. It contains 122 multitrack recordings of songs with melody annotations and instrument activations. Here, we extract the monophonic stems of 10 instruments and randomly select 160 pieces as training samples and 80 pieces as test samples for each instrument, the time of each pieces is 1.5s. the 10 instruments contain cymbal, tube, drum, fiddle, woman, violoncello, guitar, string bass and Erh-hu.

Firstly, we test the influence of variable resistance. We set the experiment in instrument recognition which only changes the parameter α. The results are shown in Fig. 4. All of these experiments use the spikegram as encoding method and has the same network structure.when $\alpha < 0$, the performance shows lower, because in the rate of membrane voltage's change,

Algorithm 1 the training procedure of our algorithm.

```
Input: target T, input spike trains S.
Output: $\nabla W$ in each layer.
1: //Feedforward
2: for each $n \in [1, N]$ do
3:     $Dm_n(0)$, $Ds_n(0)$, $Re_n(0)$, $P_n(0) \leftarrow 0$. //initialize
4: end for
5: for $t = 1$ to $T$ do
6:     get input spike $P_0(t)$. // According to S.
7:     for $n = 1$ to $N-1$ do
8:         compute $U_n(t)$, $P_n(t)$. //use (4)-(8),(9).
9:     end for
10:    compute $U_N(t)$. // use (4)-(6),(11).
11:    $V_{max}, t^* \leftarrow max(V_{max}, U_N(t))$.
12: end for
13: compute Loss. // use (10)
14: //Backpaprogation
15: for $n = N$ to 1 do
16:    //Part 1
17:    $\frac{\partial T_n(0)}{\partial W_n}$, $\frac{\partial Dm_n(0)}{\partial W_n}$, $\frac{\partial Ds_n(0)}{\partial W_n} \leftarrow N_0 \cdot P_{n-1}(t)$.
18:    $\frac{\partial Re_n(0)}{\partial W_n} \leftarrow 0$.
19:    for $t = 1$ to $t^*$ do
20:        compute $\frac{\partial U_n(t)}{\partial Wn}$ , $\frac{\partial T_n(t)}{\partial W_n}$. // use (15),(14).
21:    end for
22:    // Part 2
23:    for $t = t^*$ to 1 do
24:        if $n = N$ then
25:            compute $\frac{\partial L}{\partial U_n(t)}$. //use(18).
26:        else
27:            compute $\frac{\partial L}{\partial P_n(t)}$, $\frac{\partial L}{\partial U_n(t)}$. //use (20), (19).
28:        end if
29:        compute $\frac{\partial L}{\partial Dm_n(t)}$, $\frac{\partial L}{\partial Ds_n(t)}$, $\frac{\partial L}{\partial Re_n(t)}$. //use (17).
30:        compute $\frac{\partial L}{\partial T_n(t)}$. //use (16).
31:    end for
32:    compute $\nabla W_n$. //use (13).
33: end for
```

variable resistance is opposite to LIF model. When $\alpha < 0.3$, the performance gets better as the α increase and the F-measure can up to 98.20% best when $\alpha = 0.3$. But when $\alpha > 0.3$, the current is affected too more by variable resistance and it can be difficult to fire, the performance decrease. The results demonstrate that variable resistance can effect the performance, and a optimal α can effectively improve the recognition accuracy, on the contrary, some α will decrease the accuracy .

Secondly, we compare the performace of our model with other models, including LSTM, temporal CNN [34], TD-SNN [25], and STCA [22]. Because the input form for ANN and SNN is not the same, we use different encoding methods for them. In these models, LSTM and temporal CNN are traditional ANN and use spectrogram to process data, and TD-SNN, STCA, and our model use spikegram [35] to encode. Except the temporal CNN has 76.9k trainable parameters, the

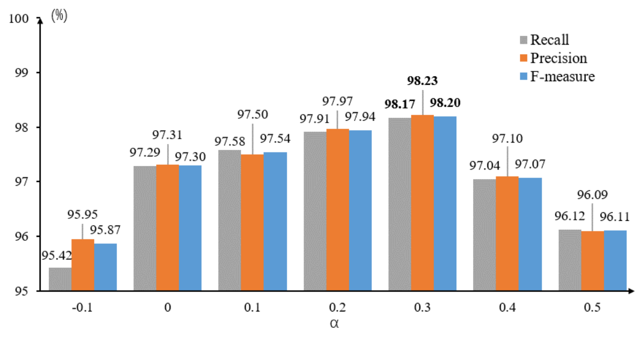

Fig. 4. The influence of variable resistance

other models have 27.6k each. LSTM is a recurrent network and has a simple structure to keep the same values of trainable parameters. the temporal CNN has 76.9k trainable parameters and is designed for modeling temporal features, whose convolutional kernels' structures are complex. The TD-SNN is a deep SNN using incomplete-BP category: it ignores the temporal dependencies. Table I shows the result of instrument recognition, we choose recall, precision and F-measure to evaluate performance. The result of Our model showed is $\alpha = 0.3$. Though it uses simple structure and relatively few trainable parameters, it gets the best performance. The MedleyDB dataset contain abundant spatio-temporal features, and our models use variable resistance, which can make the membrane voltage affect input spike trains and strengthen the temporal dependencies, so it can be better to use the temporal information to extract these features.

B. Experiment on Multipitch Estimation

In piano music, it's general to play multiple notes simultaneously, named piano multi-pitch. The MAPS dataset [36] is a dataset of piano multi-pitch estimation, which is a multi-labels and 88 classes' dataset(because the piano has 88 notes). The number of each sample's labels depends on polyphonic levels, for example, P3 means play 3 notes simultaneously, so the sample has 3 labels. Here, we select the P1-P6 levels' samples and randomly divided them into the training set and test set. These samples come from I60-68 set in UCHO and RAND and NO set in ISOL, total 14841.

This dataset still contains abundant spatio-temporal features, we train the model and test on different polyphonic levels.

TABLE I
THE RESULT OF INSTRUMENT RECOGNITION

Models	Structure	Precision	Recall	F-measure
LSTM	96-218-10	96.08%	93.31%	94.62%
Temporal CNN	\	95.94%	99.21%	97.51%
TD-SNN	384-700-10	75.62%	86.56%	80.73%
STCA	384-700-10	97.23%	97.29%	97.25%
proposed	384-700-10	98.23%	98.17%	98.20%

TABLE II
THE INFLUENCE OF VARIABLE RESISTANCE IN MAPS

α	0	0.1	0.2	0.3	0.4
P1	96.69	96.86	96.86	96.30	92.56
P2	95.37	95.29	95.34	94.58	92.48
P3	95.22	95.78	96.44	95.76	94.10
P4	93.19	94.05	94.32	93.77	92.80
P5	92.50	93.14	93.10	92.69	91.50
P6	68.04	67.97	69.66	68.06	66.24
All	90.75	91.30	91.59	91.00	89.47

To test the influence of variable resistance is general or not, we only change the parameter α as same as the experiment on Instrument Recognition. The result of the F-measure on different polyphonic levels is in Table II. It shows that when $\alpha = 0.2$, the performance gets best. Variable resistance affects the temporal dependencies, if the intensity of temporal dependencies is different, the optimal choices of α will change.

In Table III, we display the results of MAPS dataset in different polyphonic levels. Here, we list the F-measure of this dataset. In these compared models, Emiya's model [36] is an adaptive method, which can match the frequencies' inharmonic distribution adaptively and model the spectral envelope autoregressively. Benetos' model [37] is a joint multiple estimation system transcribe piano music automatically. The MPEnet [38] is a deep ANN based on multimodal sparse incoherent non-negative matrix factorization, whose structure is very complex compared with ours(1745-800-88). Our model shows a stable performance when the level lower than P6, and it has the best performance when the level is P3, P4, and P5. For all polyphonic levels, it still has the state-of-the-art performance. But it lacks in P6 to compare with MPENet. Other methods extract temporal feature by converting them into spatial field, but our model extracts features in both spatial field and temporal field, which make it show the best performance.

IV. CONCLUSION

In this work, we first proposed an iterative current-based LIF model with variable resistance. The proposed model mainly has the following advantages: (1) the proposed model is close

TABLE III
THE F-MEASURE OF MAPS

	Emiya	Benetos	Proposed	MPENet
P1	93	91.86	96.86	\
P2	93	88.61	95.34	97.11
P3	88	91.3	96.44	94.25
P4	80	88.83	94.32	90.08
P5	75	88.14	93.10	86.89
P6	63	69.55	69.66	83.65
All	63	88.54	91.59	90.39

to H-H model, which is inspired by the biological neuron; (2) it has temporal dependencies. Variable resistance builds a new relationship in the temporal field, which strengthen the dependencies; (3) it is a nonlinear-weighted network, the weighted operation is influenced by the state of the neuron. Then we construct a deep SNN algorithm by defining a voltage-driven loss function and using spatio-temporal backpropagation. We do some experiments to demonstrate that the proposed method is efficient and has a high performance in spatio-temporal fields.

References

[1] R.-M. Memmesheimer, R. Rubin, B. P. Ölveczky, and H. Sompolinsky, "Learning precisely timed spikes," *Neuron*, vol. 82, no. 4, pp. 925–938, 2014.

[2] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[3] Q. Kang, B. Huang, and M. Zhou, "Dynamic behavior of artificial Hodgkin–Huxley neuron model subject to additive noise," *IEEE transactions on cybernetics*, vol. 46, no. 9, pp. 2083–2093, 2015.

[4] 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, 1952.

[5] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on neural networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[6] W. Gerstner, "Time structure of the activity in neural network models," *Physical review E*, vol. 51, no. 1, pp. 738–758, 1995.

[7] M. Haeusser, "The Hodgkin-Huxley theory of the action potential," *Nature Neuroscience*, vol. 3, no. 11, pp. 1165–1165, 2000.

[8] Z. Wang, L. Guo, and M. Adjouadi, "A generalized leaky integrate-and-fire neuron model with fast implementation method," *International journal of neural systems*, vol. 24, no. 5, p. 1440004, 2014.

[9] Q. Yu, H. Tang, K. C. Tan, and H. Li, "Precise-spike-driven synaptic plasticity: Learning hetero-association of spatiotemporal spike patterns," *Plos one*, vol. 8, no. 11, pp. 65–87, 2013.

[10] A. N. Burkitt, "A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input," *Biological cybernetics*, vol. 95, no. 1, pp. 1–19, 2006.

[11] S. Ghosh-Dastidar and H. Adeli, "Spiking neural networks," *International journal of neural systems*, vol. 19, no. 4, pp. 295–308, 2009.

[12] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza *et al.*, "A low power, fully event-based gesture recognition system," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7243–7252.

[13] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[14] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "STDP-based spiking deep convolutional neural networks for object recognition," *Neural Networks*, vol. 99, pp. 56–67, 2018.

[15] Q. Xu, J. Peng, J. Shen, H. Tang, and G. Pan, "Deep CovDenseSNN: A hierarchical event-driven dynamic framework with spiking neurons in noisy environment," *Neural Networks*, vol. 121, pp. 512–519, 2020.

[16] T. Zhang, Y. Zeng, D. Zhao, and B. Xu, "Brain-inspired Balanced Tuning for Spiking Neural Networks." in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, 2018, pp. 1653–1659.

[17] S. B. Shrestha and G. Orchard, "SLAYER: Spike layer error reassignment in time," in *Advances in Neural Information Processing Systems*, 2018, pp. 1412–1421.

[18] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2015, pp. 1–8.

[19] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in neuroscience*, vol. 11, p. 682, 2017.

[20] E. Hunsberger and C. Eliasmith, "Spiking deep networks with LIF neurons," *arXiv preprint arXiv:1510.08829*, 2015.

[21] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, "Real-time classification and sensor fusion with a spiking deep belief network," *Frontiers in neuroscience*, vol. 7, p. 178, 2013.

[22] P. Gu, R. Xiao, G. Pan, and H. Tang, "STCA: spatio-temporal credit assignment with delayed feedback in deep spiking neural networks," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2019, pp. 1366–1372.

[23] S. M. Bohte, J. N. Kok, and H. La Poutre, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, no. 1-4, pp. 17–37, 2002.

[24] O. Booij and H. tat Nguyen, "A gradient descent rule for spiking neurons emitting multiple spikes," *Information Processing Letters*, vol. 95, no. 6, pp. 552–558, 2005.

[25] A. Samadi, T. P. Lillicrap, and D. B. Tweed, "Deep learning with dynamic spiking neurons and fixed feedback weights," *Neural computation*, vol. 29, no. 3, pp. 578–602, 2017.

[26] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers in neuroscience*, vol. 10, p. 508, 2016.

[27] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Frontiers in neuroscience*, vol. 12, p. 331, 2018.

[28] R. Gütig and H. Sompolinsky, "The tempotron: a neuron that learns spike timing–based decisions," *Nature neuroscience*, vol. 9, no. 3, pp. 420–428, 2006.

[29] R. Gütig, "Spiking neurons can discover predictive features by aggregate-label learning," *Science*, vol. 351, no. 6277, p. 1041, 2016.

[30] Q. Yu, H. Li, and K. C. Tan, "Spike timing or rate? Neurons learn to make decisions for both through threshold-driven plasticity," *IEEE transactions on cybernetics*, vol. 49, no. 6, pp. 2178–2189, 2018.

[31] R. Xiao, Q. Yu, R. Yan, and H. Tang, "Fast and accurate classification with a multi-spike learning algorithm for spiking neurons," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2019, pp. 1445–1451.

[32] G. Bellec, F. Scherr, A. Subramoney, E. Hajek, D. Salaj, R. Legenstein, and W. Maass, "A solution to the learning dilemma for recurrent networks of spiking neurons," *bioRxiv*, p. 738385, 2019.

[33] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "Medleydb: A multitrack dataset for annotation-intensive mir research." in *ISMIR*, vol. 14, 2014, pp. 155–160.

[34] J. Pons, O. Slizovskaia, R. Gong, E. Gómez, and X. Serra, "Timbre analysis of music audio signals with convolutional neural networks," in *2017 25th European Signal Processing Conference (EUSIPCO)*. IEEE, 2017, pp. 2744–2748.

[35] E. Smith and M. S. Lewicki, "Efficient coding of time-relative structure using spikes," *Neural Computation*, vol. 17, no. 1, pp. 19–45, 2005.

[36] V. Emiya, R. Badeau, and B. David, "Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1643–1654, 2009.

[37] E. Benetos and S. Dixon, "Joint multi-pitch detection using harmonic envelope estimation for polyphonic music transcription," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1111–1123, 2011.

[38] X. Li, Y. Guan, Y. Wu, and Z. Zhang, "Piano multipitch estimation using sparse coding embedded deep learning," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2018, no. 1, pp. 1–23, 2018.