IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 11, 2021, accepted June 30, 2021, date of publication July 2, 2021, date of current version July 14, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3094262

Automated Adaptive Threshold-Based Feature Extraction and Learning for Spiking Neural Networks

HESHAM H. AMIN
Department of Computer Science in Jamoum, Umm Al-Qura University, Makkah 25371, Saudi Arabia
Electrical Engineering Department, Faculty of Engineering, Computers and Systems Section, Aswan University, Aswan 81542, Egypt

e-mail: hhabuelhasan@uqu.edu.sa

This work was supported by the Deanship of Scientific Research at Umm Al-Qura University, Makkah, Saudi Arabia, under Project 43308002.

ABSTRACT Over the past years Spiking Neural Networks (SNNs) models became attractive as a possible bridge to enable low-power event-driven neuromorphic hardware. SNNs have a high computational power due to the implicit employment of the biologically inspired input times. SNNs employ various parameters such as neuron threshold, synaptic delays, and weights in their structures. However, SNNs applications are still limited and elementary compared with other neural network architectures such as the Convolution Neural Networks (CNNs). In this research, a new SNN-based model named Adaptive Threshold Module (ATM) and its algorithm are proposed. The proposed ATM and algorithm depend on the adaptation of the internal spiking neuron threshold level. Adapting the threshold of the neurons is employed to control the spiking neuron firing rate to uniquely extract the main features of the input pattern that is in the shape of spike trains. It is shown that this technique works as an automated feature extraction method of input patterns in an efficient and faster way than other methods. The proposed method can preserve all information of the input spike trains. Simulations of the proposed model and the algorithm, using the challenging speech TIDIGITS dataset, sound RWCP dataset, and Poisson distribution spike trains, show encouraging results. The ATM can make SNN provide an accuracy surpassing that of the current state-of-the-art SNN algorithms and conventional non-spiking learning models.

INDEX TERMS Spiking neural networks, adaptive threshold, features extraction, speech encoding.

I. INTRODUCTION

For more than half of a century, numerous models and structures of Artificial Neural Networks (ANNs) were considered as the cores of data processing automation in the fields of artificial intelligence and machine learning. ANNs passed to remarkable progress in the recent years after the milestone advances in the Deep Neural Networks (DNNs) with their learning algorithms [1], [2]. Moreover, ANNs are expected to be progressively engaged in many real-world applications to solve and ease human's daily life problems. Generally, all ANNs models are computational models that emulate the real biological neural networks in the brains of different living organisms. Mainly, an ANN structure is represented by a network contains processing units which are called artificial neurons. The neurons are interconnected via weighted connections that representing artificial synapses.

In the last two decades, more biological and realistic models of ANNs called Spiking Neural Networks (SNNs) emerged powerfully in research [3]. SNNs are considered the third generation of ANNs and have high computational and biological likely properties [4]–[6]. Recently, distinct SNN models were proposed to suit the requirements of real-world applications [6]–[9]. However, few models are fully biological because of their expensive computations as the Hodgkin–Huxley model [3]. On the other hand, many models are biologically realistic with fewer computations in different level degrees [5], [6], [10]. Moreover, SNNs

The associate editor coordinating the review of this manuscript and approving it for publication was Maurizio Tucci.

have high computational capabilities due to the employment of various biologically inspired parameters like the inputs which are in the shape of spike times. Additional effective parameters are employed as the neuronal threshold, synaptic delays, and weights which are implicit in SNN structures [6]. However, various SNN models and structures are utilized as multi-layers or even state-of-the-art deep neural networks.

On the other hand, over the past few years, DNNs models and algorithms had arisen very fast. Generally, DNNs give a massive push to the whole field of artificial intelligence and have become a standard for solving many of the most complex real-world machine learning problems [1], [2]. However, to solve more advanced real-world problems using DNNs, growing demand for computing and power resources is necessary and unavoidable. Therefore, to employ DNNs on embedded systems still a long way dream for scientists, where available resources are still limited. Thus, SNNs attract widespread interest as powerful models and lower computation neural networks due to their event-driven and low-powered characteristics, especially in hardware and neuromorphic implementations [11]–[16]. Unfortunately, the utilization of SNNs still limited to simple applications comparing to DNNs despite that SNNs have various benefits over other ANN such as the famous Convolution Neural Networks (CNNs). Moreover, research is scant for a full deep SNNs which can work head-to-head with CNNs. Thus, mixing both SNN models with the concepts of DNNs can develop high advances for ANNs research.

Therefore, researchers proposed various models to accomplish deep SNNs models in two main ways. First, converting the traditional CNN partially into an SNN by changing several parameters of the CNN [9], [17]–[22]. However, the generated neural networks are not pure SNN. That leads to the loss of much of SNN capabilities. Therefore, these models still suffer from the complexity of the computations inherited from the original CNN models. Second, building fully SNN-based models and algorithms for feature extraction and learning may have the same functionality and benefits as deep learning models while consuming fewer computations.

In this research, a new SNN-based model mainly composes of the Adaptive Threshold Module (ATM) with its adaptation algorithm is proposed. The ATM and its algorithm are based on the structure and parameters of SNNs. The adaptive algorithm is based on the adaptation of the internal threshold of the ATM spiking neurons. Thus, the main idea is that adaptation of the threshold level is employed in multiple stages of the ATM to control the firing rate of spiking neurons. Consequently, by adapting the threshold levels of the neurons, the output firing frequency is controlled. This leads to the extraction of the input pattern features and information. The real-world input patterns are preprocessed and introduced into the ATM module in the shape of spike trains in the time domain.

Simulations show that this model not only works as a feature extraction of the input pattern, in a way almost similar to the CNN convolution and pooling layers but also with fewer computations. Moreover, the ATM preserves all information of the input spike train by exploiting the adaptation of threshold levels of spiking neurons. Simulations of the proposed SNN and its learning algorithm are done and show that the algorithm has encouraging results.

The following sections of the paper are arranged as follows: Section II contains a general explanation of the SNN models. Section III contains related models for the research. Section IV contains a detailed explanation of the proposed adaptive threshold and the learning modules. In Section V, the simulation results of the proposed model and analysis of these results are discussed.

II. SPIKING NEURAL NETWORKS MODELS

The structure of an ANN and the functions of its neurons have the main effect on the learning algorithm and the accuracy of results. ANNs have various algorithms and models that range from simple networks to the most complex and powerful DNNs [23], passing through the famous multi-layer ANNs. From the learning point of view, ANNs may be divided into supervised and unsupervised learning. Furthermore, ANNs may be categorized according to their dependency on biological structures such as the SNNs or non-biological such as the multi-layer ANN. However, the biologically plausible neural network models get rising trends in the last two decades [6].

Understanding the SNNs models, including the spiking neurons' function, structure, and way of communication is a vital step for successful modeling of any related model as in this research. In the SNN, pre-synaptic neurons send information to post-synaptic neurons through electrical pulses known as action potentials or spikes. These spikes travel from neurons to others through synapses that connect pre- and post-synaptic neurons. The synapse is considered one of the main players in the learning rules of all neural network models. The learning process is mainly done by changing the synaptic strengths which are known as weights of the synapses. The main important characteristic of the SNNs is the implicit usage of spike times as the main input information. Therefore, the time of a spike that passes from a pre-synaptic neuron carries information through one or more synapses towards post-synaptic neurons. Thus, for every spike received by a post-synaptic neuron, it accumulates its membrane potential according to the SNN model.

However, various SNNs structures and algorithms were proposed to adapt the requirements of theoretical research and real-world applications [3], [5], [6], [8], [9]. SNN neurons have many computational models as the Leaky Integrate and Fire (LIF) neuron [3], the Spike Response Model (SRM) [5], and the Izhikevich model [10], among many others. These models are regularly used for real-world applications due to their realism in both computational and biological characteristics. However, in this research a linear approximation of the SRM is employed rather than the original SRM. That approximated SRM is linear to simplifies the mathematical analysis of the proposed model with no effect on the performance and biological concepts as proofed by the author [7].

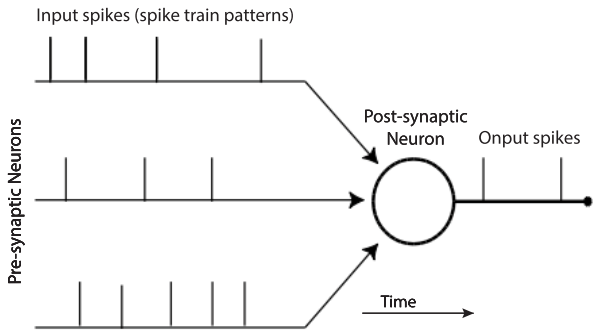

FIGURE 1. A spiking neuron excited by spike trains as inputs and fires output spikes. Each vertical line represents a spike at a specific time. All input spikes transmit from pre-synaptic neurons (are not shown in this figure) into the post-synaptic neuron.

As shown in Fig. 1, input spikes from pre-synaptic neurons are fed into a post-synaptic neuron. Each of those input spikes contains information in its time. Additionally, information is represented as the time between any two consecutive spikes is called the Inter-Spike Interval (ISI). Consequently, the post-synaptic neuron fires output spikes at different times when it reaches a predetermined threshold value from below [6]. However, real-world patterns, such as speech signals, images, and other analog signals, should be converted because they are not represented originally in the shape of spikes in the time domain. Thus, there exist many methods to convert values into suitable spikes input [6], [24].

III. RELATED WORKS

The learning of SNNs is done by changing one or more of its parameters as synaptic weights, synaptic delays, and the threshold level of the neuron. The diversity of learning parameters is one strength of SNNs over other types of neural networks. However, like in most of ANNs learning algorithms, SNNs learning methods employ synaptic weights as the principal learning parameter. Various supervised [6], [25], and unsupervised Spike-Timing-Dependent Plasticity (STDP) [26], [27] learning algorithms had been proposed which employed SNNs synaptic weights as the principal learning parameters. Delays of the synapses had been used as extra and supporting variables for the learning task [6], [28]–[31].

Like in this work, extensive research showed that generating of spikes in an SNN can be modeled by adapting the threshold of its neurons besides other parameters such as synaptic weight [22], [32]–[41]. Various research showed that the increase of variability of neuron threshold is proportional to the increasing in the input spikes firing rate [42]. Also, a few neuromorphic hardware was built to achieve the relationship between threshold and firing rate [36]. An experiment that had been performed in vivo showed that threshold variability plays a crucial role in the information transfer process in the brain of electric fish [43], [44].

Mainly, researchers use two approaches in modeling adapted threshold approaches: (a) adaptation of the incremental change of the threshold potential level according to the input spike on a particular time; and (b) adaptation in the neuron potential using changes in synaptic weight, during training, while assuming a constant spiking neuron threshold [42], [45], [46].

In this research, a novel model that exploits the adaptation of the spiking neuron threshold for real-world applications is proposed. The proposed model considered that the variability of the spiking neuron threshold has to include the history of the occurrence of input spikes times and their accumulated potentials.

Various research utilized the changes in the SNN parameters in various terminologies. One of these terminologies is neural Intrinsic Plasticity (IP). IP has gained growing concern from the computational perspective [47]–[51].

IP is a self-adaptive mechanism that can adapt the SNN response to input stimuli by employing parameters of the neurons. Thus, it plays an important role in temporal coding and improving the learning and convergence behavior for different SNN structures. IP is essential in achieving the dynamics of neural circuits [47] as it allows spiking neurons to transmit the maximum amount of information. The highest entropy may measure this amount of information to their outputs with a constrained degree of firing activity. It was observed in various types of biological neurons, as visual cortical neurons, that IP can change the response of neurons through regulating their intrinsic parameters such as voltage-gated channels [52], [53].

In [47], the authors proposed that SNN dynamics may depend on the interaction between IP parameters such as synaptic weights and time intervals. IP can change both synaptic weights and the intrinsic membrane properties of neurons. They showed that neurons with complex temporal dynamics could provide short-term memory approaches that rely solely on intrinsic neuronal properties. They represented that IP may aid in the long-term regulation of intrinsic neuronal properties. They proved that memory in the SNN results from the interaction between variations in synaptic weights and intrinsic membrane properties.

In [48], the computational behavior of a feed-forward SNN (SFNN-IP) was studied based on brain-inspired IP membrane potential adaptive technique to adjust the intrinsic excitation capability of each neuron. They presented that the learning rule can regulate the neural activity for different quantities of external input. The training started from a conventional ANN using backpropagation. Afterward, the rate-based neurons were converted into spiking neuron models with the IP learning. Results confirmed that both over-activation and under-activation of neuronal response could be avoided during the computations.

In [49], an SNN approach, based on STDP + IP learning rules, was developed for Liquid State Machine (LSM) with a biologically inspired self-organizing network. The STDP + IP was based on two neural plasticity learning rules. The model may optimize the computational performance, with the intrinsic features of dynamical memory and recurrent

synapses. The STDP learning rule adapted the connectivity among neurons; meanwhile, the neuronal excitability degrees were adapted to provide a moderate average activity level by an IP learning rule. The threshold for generating spikes was distinct for each neuron. Results showed that the neural network learned with the STDP + IP approach has better entropy and high signal propagation than the other networks.

In [50], an IP mechanism, depends on threshold level, was applied to a multi-layer SNN. The IP mechanism adapted the firing rates of neurons at a steady level while maximizing the information entropy. They showed that the IP rule affected the network classification accuracy at a high level and the convergence speed was greatly enhanced.

In [51], they proposed an approach, named SpiKL-IP, which focused on the limitations of other IP rules. They developed the SpiKL-IP to enhance the entropy of the output firing rate distribution of each spiking neuron by regulating the output firing rate into an optimal distribution. By employing IP tuning, SpiKL-IP adapted the intrinsic values of a spiking neuron while reducing the divergence between the desired distribution and the actual output firing rate distribution. They showed that SpiKL-IP could effectively operate in a real-time way under heterogeneous inputs and network situations. Simulations demonstrated that SpiKL-IP was applied to different neurons or as a portion of a larger SNN effectively produced the target distribution. Estimation of the approach for real-world speech and image recognition tasks proved its remarkable performance and recognition accuracy.

One of the main points in ANN algorithms is the time of convergence to the correct results with high accuracy. Moreover, feature extraction is vital for results accuracy. Recently, deep learning becomes the state-of-the-art of many audio classification tasks [54]. In deep learning algorithms, especially Convolutional Neural Network (CNN), feature extraction consumes much time because of the tremendous number of the learning parameters employed in the multiple convectional layers. Nonetheless, revolutionary developments in deep learning models show state-of-the-art performance in ASR starting from the significant milestone achievements in acoustic modeling [55]–[57] with the aid of DNNs [58]. Moreover, DNN significantly outperformed state-of-the-art methods in Automatic Speech Recognition (ASR) such as Hidden Markov Model (HMM) [59]. As an example, CNN was applied to audio modeling in a way that applied the convolution across time windows of audio frames to recognize the audio stream into classes [56]. They applied CNN to ASR where a limited-weight-sharing algorithm was employed to enhance speech features modeling. The proposed approaches have reduced the error by $6\% - 10\%$ compared with DNNs on the TIMIT dataset. However, the final CNNs were quite complex and employed over 4 million parameters. Therefore, compared to CNN, the proposed model consumes much fewer parameters for feature extraction and classification.

Furthermore, as a significant point, CNN is not a desirable model for ASR because of the nature of audio signals and their features. Audio signals depend on temporal features besides the spatial features. On the contrary, CNN offers excellent achievements in image recognition research because of the nature of images which mainly depend on spatial features. Thus, in most of the ASR research, speech signals inputs to CNNs are converted into spectrograms (spectrum images) to deal with speech signals as input images [56], [57].

If a model can represent any set of input features as images, then CNN can classify them. Therefore, the magnitude spectrograms obtained via Discrete Fourier Transformation (DFT) of audio signals are used as the input to CNNs when used for audio classification. Because spectrograms are 2D images, the audio classification problem may be treated similarly to a regular image classification problem using a CNN. However, [60], [61] showed that CNNs are better suited than DNN and Long Short-Term Memory (LSTM) networks for the ASR. Although it is not straightforward to compare state-of-the-art models in ASR because of the variety of datasets and evaluation methods, CNNs seems to perform better than other types of classifiers for ASR.

Recently, end-to-end models were considered in deep learning for ASR. However, these algorithms still are not suitable for real-time ASR applications because of their large model sizes and computation complexity [62]. On the contrary, SNNs models and algorithms offer challenging tools for temporal tasks such as ASR because they can directly deal with temporal features besides spatial features.

However, it much rather compares the ATM with recurrent neural networks (RNNs). RNNs capture the appropriate temporal patterns in consecutive data, such as audio signals, to enhance their recognition accuracy. RNNs architecture layers can hold the memory of past parts of an input sequence. Besides, the long short-term memory (LSTM) networks have special hidden units, called gates, that can regulate the amount of information to remember or forget in the input sequences [63]. Other ASR studies using the LSTM networks showed notable accuracy enhancement compared to state-of-the-art DNN models. In [64], researchers applied a deep LSTM architecture in the ASR over a large vocabulary set. They found that deep LSTM is superior to baseline DNN models. This LSTM is successful in an end-to-end speech learning method for large English and Mandarin Chinese datasets. In [65], researchers performed extensive experiments using various LSTM architectures for ASR and compared the performance to state-of-the-art models. The LSTM model is extended in [66] to a bidirectional LSTM (BLSTM) which was set on top of convolutional layers to enhance the ASR accuracy. Moreover, including attention enabled LSTM algorithms to outperform pure RNN models. An attention algorithm, called Listen, Attend, and Spell (LAS), was used to encode, attend, and decode, respectively. The LAS appraoch was used with LSTM to enhance speech recognition accuracy [67].

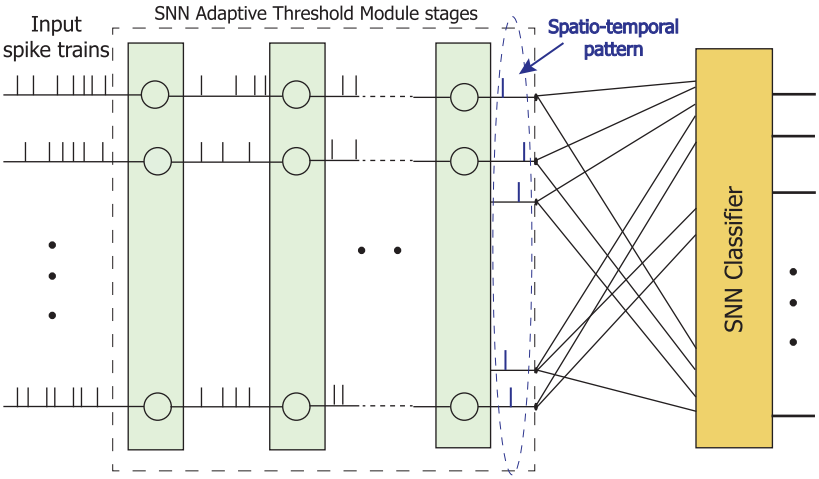

FIGURE 2. The proposed SNN model. The ATM is for extracting features of the input spike trains into spatiotemporal outputs. The classifier is for clustering these input spatiotemporal spikes.

IV. MODEL STRUCTURE

The main idea of the proposed ATM model depends on that the SNNs have many parameters that can be used for feature extraction and learning tasks. One of these parameters is the threshold level of a spiking neuron. The model exploits the adaptation of the spiking neuron threshold as the fundamental technique for feature extraction. The model employs the variability of the spiking neuron threshold to include the history of the input spike times occurrence in an input spike train.

The model is composed of two main stages: the ATMs and the classifier modules as shown in Fig. 2. A flowchart of the proposed model is shown in Fig. 3. First, the spike trains from a preprocessed input pattern are fed into the cascaded ATM stages. These stages and their spiking neurons are similar in structure and function. As shown in Fig. 2, the input pattern is a combination of spike trains that result from converting real-world input patterns into spike trains, like the example shown in Fig. 4. Therefore, the ATM stages extract the timing information of the input spike trains and convert them into a pattern composed of spatiotemporal spikes. Finally, those spatiotemporal patterns are applied into the classifier which is trained using a local semi-unsupervised learning algorithm for clustering the input patterns according to their clusters [6]. In the following subsections, I introduce a detailed description of both the ATM and the classifier modules. These two modules, shown in Fig. 2, are composed of spiking neurons with different characteristics.

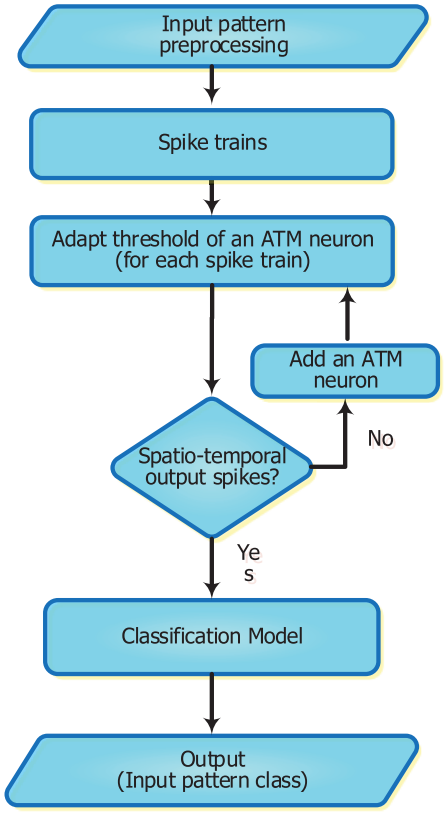

FIGURE 3. Flowchart of the proposed model.

A. THE ADAPTIVE THRESHOLD MODULE (ATM)

The main contribution of this research is the ATM and its adaptation algorithm. For every input spike train, the feature extraction process takes place in multiple stages of the ATM, as shown in Fig. 5.

In this research, the adaptation of the threshold of the ATM neurons in each stage occurred to achieve two main objectives. The first goal is to preserve the uniqueness between all input patterns (spike train times) information and the final spatiotemporal output during the feature extraction process. The second goal is to reduce the number of output spikes in each stage, which leads to reducing the number of the required stages for the feature extraction process. In this way, by stepping deep into ATM, the feature extraction is accomplished.

Mainly, in most of the SNNs research, a spiking neuron threshold level is kept constant according to simulation calculations or to some predetermined settings [3], [5], [6].

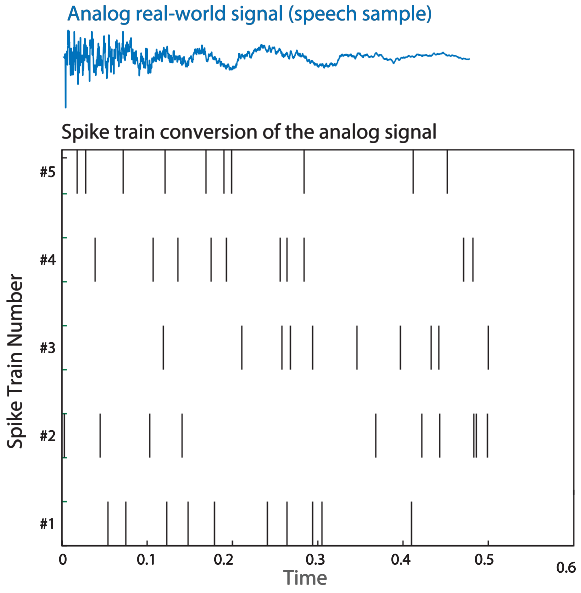

FIGURE 4. An example of a preprocessed real-world pattern (signal). The signal is converted into and represented by multiple spike trains.

Therefore, if the neuron threshold level is assigned to a low potential, then the firing rate of the output spikes is very high with no much information can be extracted, especially when input spikes are close in time or burst mode inputs. In contrast, if the neuron threshold level is assigned to a high potential, time information represented by the input spike times maybe not be detected, especially when input spikes are distant in time. In this case, information may be lost from the input spike train. In both cases, the output spikes are not accurately reflecting information in the input spike train pattern. Hence, the uniqueness between input and output is missed. Therefore, to overcome those problems, an adaptation of the threshold level of the neuron should be involved through the SNNs encoding process. Therefore, an adaptation of the threshold should exploit the input spike times, inter-spike intervals (ISIs) between input spikes, and the mode of input spikes. However, the mode of spikes is the average ISIs between neurons that may be low or distant in time. Thus, the adaptation of the neuron threshold level has to follow how the input spikes contain information.

As shown in Fig. 5, an input spike train is fed into the first neuron L_1 in the ATM. The spikes of the input spike train are represented by times t_{i1} to t_{iN}. The time of the earliest spike, t_{i1}, is used as a reference time for all next spikes in all stages. Consequently, the output of the ATM neurons is another spike train that contains fewer spikes compared to the input spike train. Undoubtedly, the output spike times are generated at later times to the input spike train times. Thus, as shown in Fig. 5, output spike times are $t_{oN} > \ldots > t_{o1} > t_{iN} > \ldots > t_{i1}$, and the same for all the later output spikes. The process of the ATM neurons is repeated in all stages until one output spike is achieved in one of the ATM stages. Thus, this single spike time is a unique representation of the input spike train times. However, the number of levels of the ATM stages should be adjusted in simulations to achieve this target. The reason behind this is that input spike trains are not similar in both the number of spikes and inter-spike interval distribution. Therefore, at a certain level of the ATM, there would be a single output spike. Thus, some of the ATM neurons may continue for the next few stages with just passing through.

To achieve the stated properties, the adaptation of the ATM employs a spiking neuron model with an excitatory potential, as in Eq. (1),

$$v_j(t) = \sum_{i=1}^{n} w_i \alpha(t - t_i)$$
$$\alpha\,(t) = \frac{t}{\tau} e^{(1-\frac{t}{\tau})} \tag{1}$$

where v_j represents the spiking neuron post-synaptic potential, w_i represents the input synaptic weights from the pre-synaptic neurons i, t_i represents the input spike arrival times. $\alpha(t)$ is the SRM potential function as shown in Fig. 6 as an exponential function and τ represents the time constant of the exponential function which is the decay time constant of the synapse.

Post synaptic potential, described in the literature, such as the exponential $\alpha(t)$ function can be reformulate as a bi-exponential function can be expressed as $(exp(-t/\tau_1) - exp(-t/\tau_2)$ [68]. However, the SRM function is simplified using the bi-exponential function concept for the reason of linear implementation [6] as in Eq. (2),

$$\alpha(t) \approx \begin{cases} \tau_1 t & \text{if } t_i \leq t < t_{top} \\ 1 & \text{if } t = t_{top} \\ -\tau_2 t & \text{if } t_{top} < t \leq t_{max} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where τ_1 and τ_2 represent the time constants of the rising and decaying parts, respectively, of the linear potential function shown in Fig. 6. The times t_{top} and t_{max} represent the time of the peak and vanish, respectively, of the potential which has been arisen due to the spike at time t_i. However, as shown in Fig. 6, both functions are almost the same by adjusting the parameters of Eq. (2).

As shown in Fig. 7, a preliminary demonstration for one of the ATM neurons. The threshold level is increased proportionally with the increase of input spike times. Thus, the adaptation of the threshold value of the neuron follows a proposed formula represented in Eq. 3,

$$\vartheta_{th}\,(t_i) = \frac{\delta.t_i}{t_i - t_{i-1}} + \vartheta_{th}\,(t_{i-1}) \tag{3}$$

where δ is a small constant which controls the rising of the threshold value and it is in the range $0 < \delta < 1$. $t_i - t_{i-1}$ is the ISI between the last two consecutive spikes in an input spike train. $\vartheta_{th}(t_{i-1})$ is the last known adapted threshold value that is applied from the spike at time t_{i-1} until a new spike come at time t_i then it is changed to $\vartheta_{th}(t_i)$ on time t_i. However, an initial $\vartheta_{th}(0)$ is applied at the time of first spike (t_1).

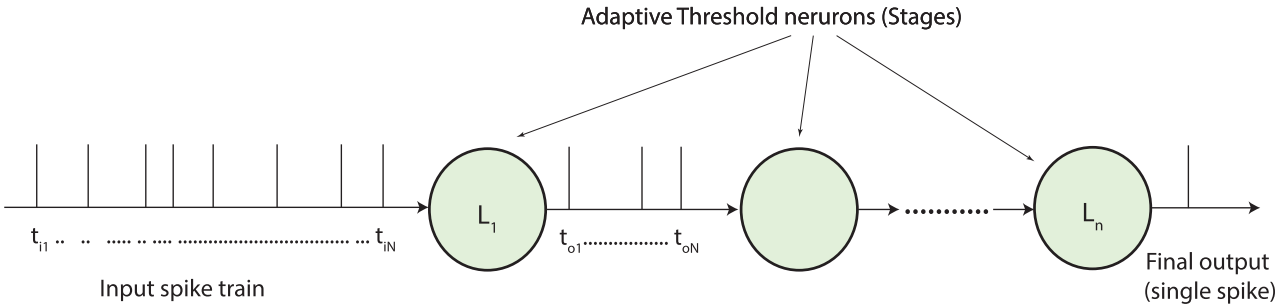

FIGURE 5. The ATM stages. Each ATM stage receives a spike train as an input and outputs another spike train that in turn is submitted into another ATM stage. In every stage, a neuron adapts its threshold level independently of other neurons. The number of spikes is reduced in every stage until generating one spike at the final stage.

Equation (3) implies that the increase of the threshold level is inversely proportional to the ISI between the last two consecutive spikes, $(t_i - t_{i-1})$. Thus, if the ISI between any two successive spikes is short in time, i.e. burst mode spikes, there is a need for much increasing of the threshold level to reduce the number of output spikes. In contrast, if the ISI between any two successive spikes is long in time, then there is a need for a small increase of the threshold level to transfer all possible information from input to output spike times. In that way, the main idea is achieved as the ISIs in an input spike train are short in time then there is a big chance for multiple output spikes. On the other hand, if the time between spikes is broad, there is less chance for output spikes. Moreover, Eq. (3) implies that when the ISI time between two consecutive spikes is too large, the increase in threshold value is almost zero. More adjustment of the threshold level is controlled using the δ parameter that has a great rule during the threshold adaptation process. δ may change from one neuron to another as shown in Fig. 5. More discussions are in the experimental results section. To preserve information, the adaptation of the threshold level is proportionally relative to the input spike time, t_i.

In that way, features of the input patterns are extracted during multiple stages of the ATMs. Thus, the final output spikes are almost in the shape of a spatiotemporal pattern instead of the complex input spike train patterns.

B. CLASSIFIER LEARNING ALGORITHM

Fig. 8 shows the classifier that is the last stage of the whole proposed model, as shown in Fig. 2. The SNN classifier consists of spiking neurons that are working differently compared with the ATM neurons. The main differences are that, during the learning process in the classifier, the input synapses weights are modified, and the threshold level of its neurons is set to a constant potential level. Its final outputs represent clusters of the input patterns. Each cluster is represented by more than one ISI unit according to the behavior of the input spike times [6]. As shown in Fig. 8, the input to the classifier is a spatiotemporal spike pattern, which is the output of the ATM. This pattern is applied to the classifier module that works as a local semi-unsupervised SNN for clustering input patterns [6], [7].

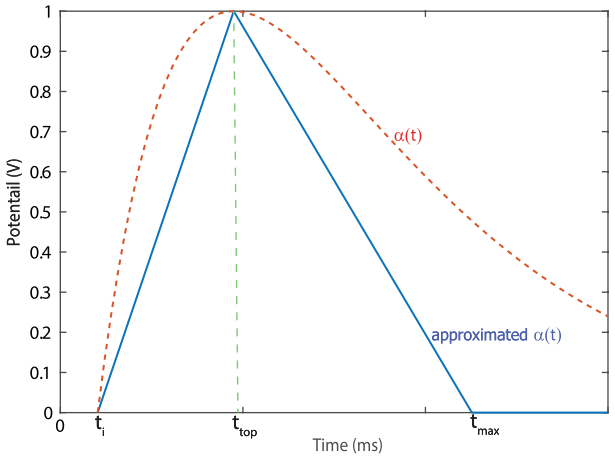

FIGURE 6. The original spike response function (the dashed line) and its approximation.

The learning in the classifier is done by changing the synaptic weight values using formulas as in Eq. (4) [7]:

$$w_i = \beta_1 \times t_i; \quad w_i = \beta_2 / t_i \tag{4}$$

where β_1 and β_2 are constants with values less than one. These constants are employed during the learning process for limiting the increase of the synaptic weight values that are adapted according to input spike times t_i. The main unit of the classifier is the ISI unit shown in Fig. 9. Each of the ISI units consists of two spiking neurons. However, both formulas in Eq. (4) are used for changing the weight values w_i of the two spiking neurons in the ISI unit. Moreover, the algorithm is considered a local learning algorithm that uses the changing the weight values according to the times of input spikes t_i [6]. As shown in Fig. 9, the use of two neurons with different weight changing techniques is to keep a unique relationship between inputs and outputs [7]. However, the two spikes, the output from each ISI unit, are allied with one spike for the sake of cluster output using a coincident detector [6].

The learning in the classifier is a local semi-unsupervised clustering algorithm where there is no prior knowledge of the number of clusters of patterns, while their inputs are labeled patterns according to their classes. Thus, the learning

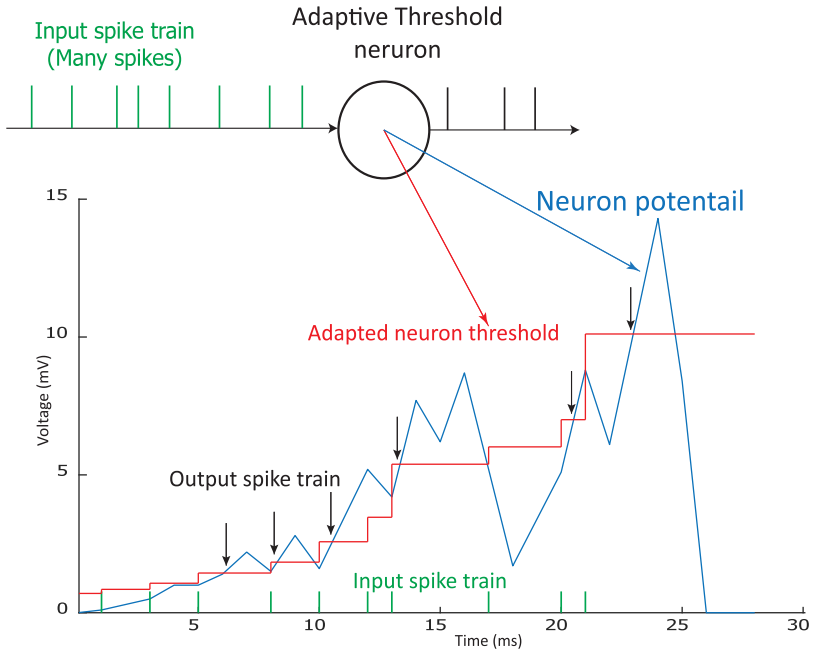

FIGURE 7. An example shows one of the ATM neurons. The upper part represents an ATM neuron with a spike train input (vertical green pulses) to that ATM neuron and outputs another spike train (vertical black pulses). The lower part represents the adaptation of the threshold of that ATM neuron (red stair line) with each of the input spikes. The changes in the ATM neuron potential are represented by blue lines. When the neuron potential passes the adapted threshold level, from blew, an output spike fires at this time (black arrows show these points).

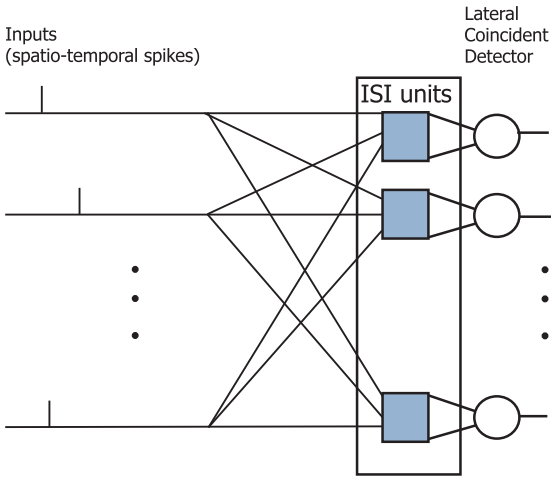

FIGURE 8. The classifier with spatiotemporal spikes inputs represented by vertical lines. Each ISI unit outputs two spikes due to the learning method. These final two spikes are collected using a coincident detector to generate only one output.

method operates by adding one neuron for each new cluster and approximate any other pattern to the closest cluster [6].

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this section, various experiments are conducted to demonstrate the feasibility of the proposed model. Moreover, the experiments are carried out to cover and discuss all needed adaptation of the model parameters. The proposed model and algorithm are demonstrated using codes written from scratch and simulated using the Matlab software.

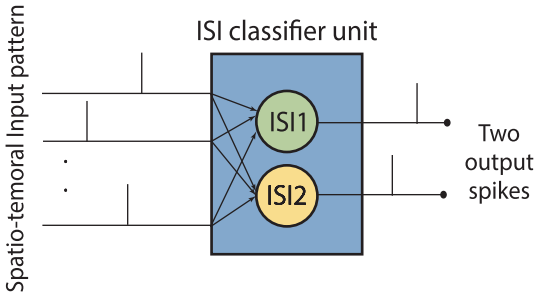

FIGURE 9. One of the classifier ISI units. Vertical lines represent input and output spikes. Each ISI unit contains two neurons called ISI1 and ISI2 that learned using (4).

In these simulations, the employed datasets are converted to various types of spike trains. These spike trains are applied as inputs to the ATM. Consequently, the spatiotemporal spikes outputs of the ATM are clustered by the classifier.

In the following subsections, the simulations are as follows: First, a preliminary example to describe the method and to discuss the ATM parameters. Second, complex Poisson spike train patterns with different rates are used as input patterns. Finally, real-world datasets of speech and sound signals are used to compare with other models' accuracy.

A. PRELIMINARY EXAMPLE AND ATM PARAMETERS ANALYSIS

Figure 10 illustrates the idea of the ATM using an example with an input spike train contains nine spikes shown by nine

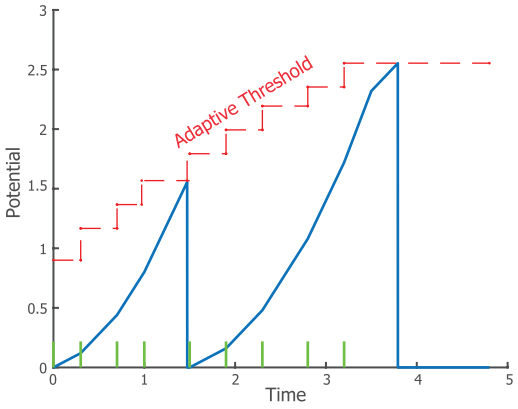

FIGURE 10. ATM input/output exa mple. Nine input spikes (vertical green spikes) are mapped into two output spikes.

green vertical lines on the time axis. In this example, the input spike train features are extracted by one neuron of the ATM into only two output spikes at times 1.48 and 3.80 Sec. represented by the dropped blue vertical lines. Generally, the output spikes are released when the neuron potential crosses the threshold level from below [6]. The adapted threshold, illustrated by a dashed stair red line, is adapted according to the input spikes and other parameters, as stated in Eq. (3). Consequently, in every neuron of the ATM stage, the same process is repeated until one spike is out.

It is recognized from Fig. 10 that the output spike times increase during extraction in the ATM, and the quantity of spikes decreases while there is an increase in the threshold level from one stage to another. This is predicted because the adapted threshold is proportional to the input spikes times which increase from one stage to another. Therefore, the parameters in Eq. (3) should be modified to assess that decrease in the number of spikes and the increase of the threshold level from an ATM stage to another.

Therefore, consider the synaptic weights are constant between the ATM stages, the main parameter that should be adapted is the δ parameter in Eq. (3). The δ value has to increase when going from one stage to another in the ATM module. Moreover, as shown in Fig. 12, the time constants of the rising (τ_1) and decaying (τ_2) of the potential function Eq. (2) has to decrease when going from one stage to another in the ATM. Thus, the potential function Eq. (2) has to be broader to cover the increase of the ISI from one stage to another due to the encoding process. That decrease is done because the inter-spike intervals between spikes grew wider due to the reduction of the number of spikes from one stage to another in the ATM.

To achieve these adjustments, a method is proposed to adjust the time constants (Eq. (2)) in each stage of the ATM. It is proposed that the slope of the time constants be inversely proportional to the maximum inter-spike intervals, ISI_max, of the input spike train for each of the ATM stage neurons. Thus, in each stage, the ISI_max is recalculated, then the time constants are changed according to it. However, to reduce the computations for real-time applications, ISI_max is calculated only in the first stage, then constant decreases may be applied in the following stages as shown in Fig. 11. It is recognized that the slopes do not noticeably change after some stages, because ISI grows broader in spike trains in the later stages of the ATM. However, as the frequency of input spikes increases, the ISI is smaller for a constant spike train length. Therefore, as shown in Fig. 11, the steady slope in the lower frequencies (100 Hz) starts in earlier stages than for higher frequency inputs (300 Hz).

On the other hand, in Eq. (2) δ increases linearly through the ATM stages. The initial value is chosen as a small value, $\delta = 0.1$ and it increases by a tiny constant value (around 0.05) in each following stage in the ATM.

To overcome the increase of the threshold values, from one stage to another, it is proposed to reset the time of the first spike for each stage in the ATM. In that way, the first spike time in each stage is initialized to 0. Analysis of this approach is discussed in the following subsection.

B. POISSON SPIKE TRAINS

Spike trains with various Poisson distributions are generated to demonstrate the proposed model using complex input spike trains distributions. The Poisson distribution spike trains with predetermined rates (frequencies) have been discussed and commonly employed in theoretical research [3], [69]–[71]. Moreover, the inner parts of the human ear receive the frequency and strength of sound sources and generate spike trains through an inhomogeneous Poisson encoding process [71].

In this experiment, the spike trains are generated using various rates and time lengths to emulate real-world problems such as speech, images, and human activities [72]. The generated spike trains have diverse points of challenge. For instance, the ATM outputs, which are spatiotemporal patterns fed into the classifier, are not similar even for spike trains with the same distribution and same time lengths. This variability of the spatiotemporal patterns shows one of the strengths of the SNN and the proposed ATM where some inputs may have no input spikes within a predetermined duration. However, in this simulation, no refractory time is used to preserve all information. As an example of the generated spike trains, Fig. 13 represents spike trains are generated using Poisson distribution at a frequency of 50 Hz and length of 300 mSec.

Generally, spike trains are represented as a set of pulses occur at specific times with an abstracted shape and duration. Therefore, in the numerical simulations, time is discretized to 1 msec bins, where each bin contains either pulse (spike) or nothing in the spike-driven simulations. Some other numerical simulations use zeros for no spike and ones for a spike in the suitable bin. In this research, the simulations depend on the latter method because it is a more biological spike-driven one.

To demonstrate the proposed model and its algorithm using one of the generated Poisson spike trains, Fig. 14 represents an example of full ATM stages from the input spike train until generating one spike at the final stage. The shown

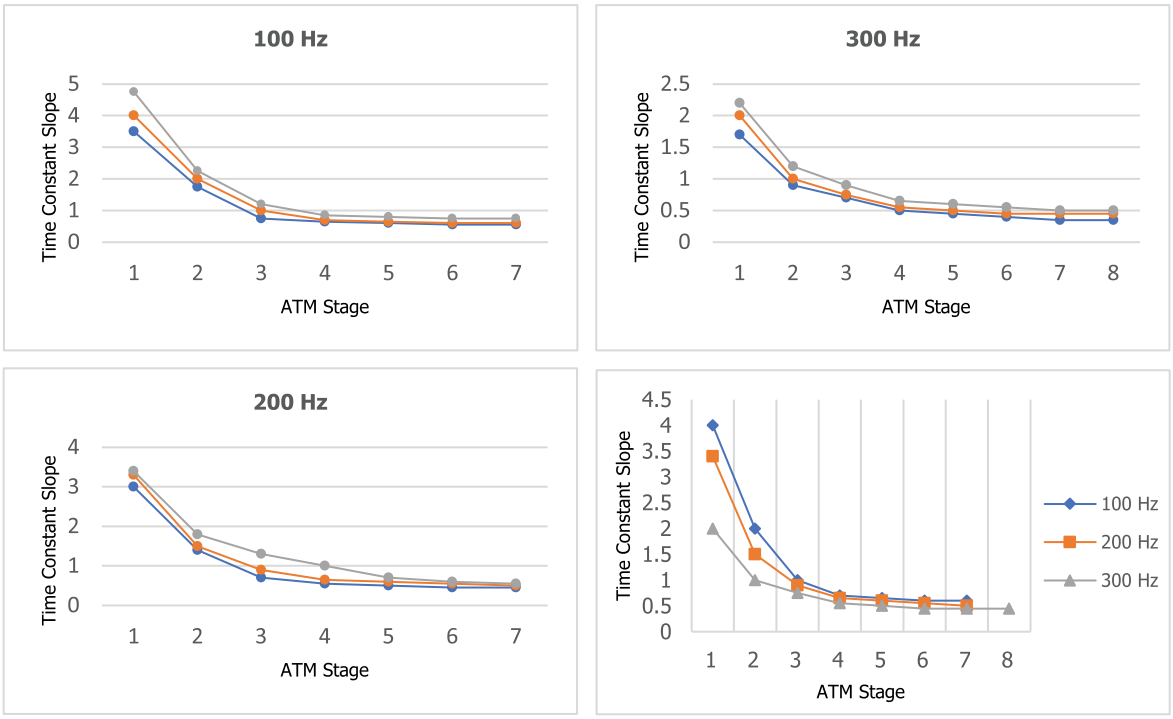

FIGURE 11. Variations of time constant (τ_1 and τ_2) slopes through the ATM stages for different input frequencies. When moving through the ATM stages, the slope decreases. The slope is calculated for random spike trains with various frequencies: 100 Hz, 200 Hz, and 300 Hz. In the right-bottom figure, the time constant slope for higher frequency 300 Hz is less than slopes for 200 Hz and 100 Hz in order.

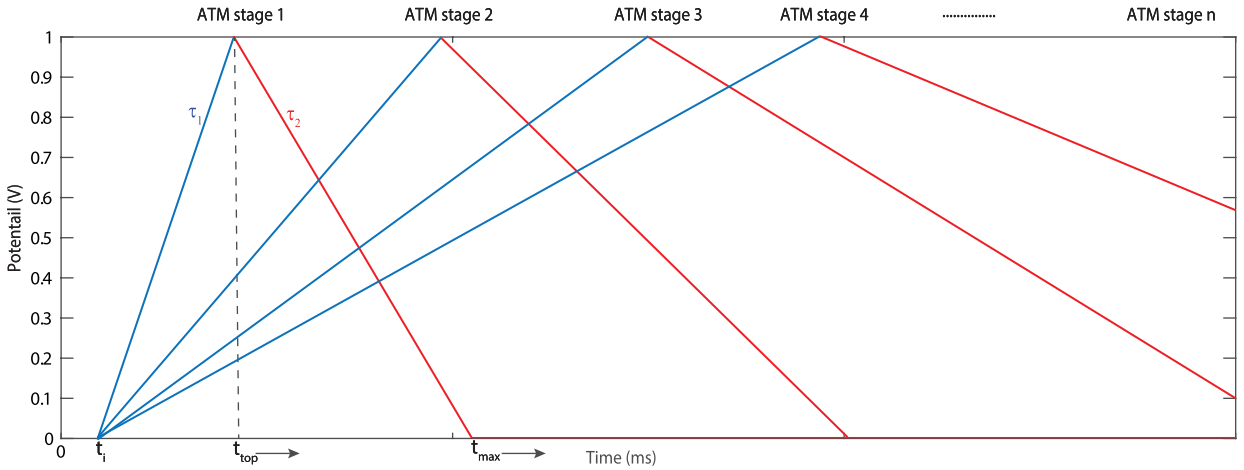

FIGURE 12. The variations in time constants τ_1 and τ_2 through the ATM stages. When going further in the ATM stages, the potential function gets broader to cover the increasing spacing in ISI of spikes trains.

example employs an input spike train which is generated randomly using 200 Hz Poisson distribution at times $t = [22\ 42\ 48\ 50\ 53\ 56\ 58\ 76\ 82]$ with a length of 100 mS and the initial threshold set to 0.7 V with an initial $\delta = 0.1$. Figures 14 (a) to (e), represent a sequence of ATM neural stages to extract features using threshold adaptation. As stated earlier, many parameters of Eq. (3) and Eq. (4) should be treated in each stage. As an instance, the rising τ_1 and decaying τ_2 time constants of the neuron potential decrease from one stage to another. That is predicted as the time spacing between output spikes from one stage to another is wider. Thus, first τ_1 and τ_2 are both with hight slope values to detect the burst in the input spikes (Fig. 14 (a)). Then, in the next stages, they must be broad in time (less slope) to detect multiple spikes at one output (Fig. 14 (d) and (e)). Moreover, for the same reason, to adjust the increasing of the threshold value, an increase of the ATM parameter δ is carried out. Figure 15 illustrates the same example as in Fig. 14 but with

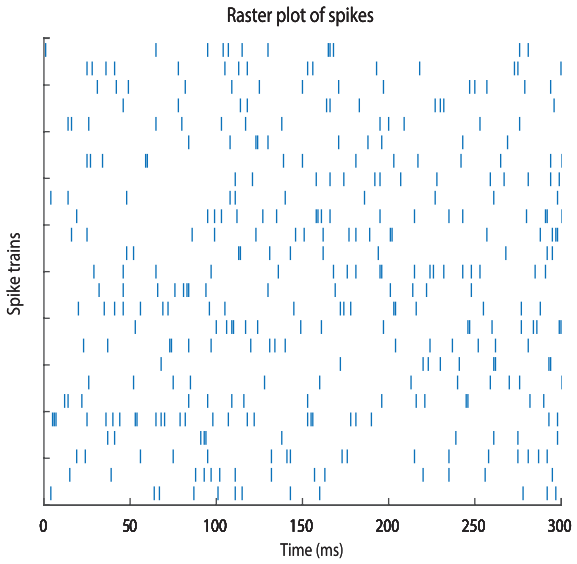

FIGURE 13. A set of spike trains generated by poisson distribution of a firing rate (frequency) 50 Hz and a 300 mS period.

resetting the input spike reference time for each stage to be zero. Thus, all spike times in each stage input train are referenced by its first spike. As shown in Fig. 15, in this way, there are a couple of improvements over the previous example. First, fewer stages may be employed in the ATM module to extract the final feature spike time. That is because new time values arise in each stage as it is counted each time from zero. Besides, the increase of threshold is less than the previous example, as stated in Eq. (3). In this way, the increase of the synaptic potential is less than the previous example due to fewer spike time values are used. However, the parameters change considerations for Eq. (3) from one stage to another still as in the previous example.

Table 1 represents simulation results of 4000 generated patterns per each Poisson distribution. These patterns are divided into 70% for training while the other 30% are for testing tasks. Each pattern consists of 20 spike trains with the same rate. Therefore, the ATM outputs 20 spikes as a spatiotemporal pattern to the classifier. The classifier receives those 20 spikes as one input pattern to cluster it properly. As shown in Table 1, the output for each rate is divided into four clusters according to the spike train lengths. From these results, it is shown that the model can deal with various complexities of patterns.

For a fair comparison with other models, I have compared to research using available online codes and resources with suitable changes to fit inputs types. As shown in Table 2, Poisson spike trains with the frequency (300 Hz) are used for this comparison. It is shown that the proposed ATM has accuracy comparable and challenging to other models. However, some of these methods accept input spike trains as it with no conversions [51]; while others such as [73] and [74] accept the inputs in the shape of spectrograms (images). Furthermore, models of [73] and [75] show better training accuracy and closer testing accuracy as shown in Table 2

TABLE 1. Training and testing results of the proposed algorithm.

Avg. Freq. (HZ)	Avg. Length (ms)	No. Samples	Training (%)	Testing (%)	# Classifier Neurons
50	100 250 400 700	1000 1000 1000 1000	99.7	97.2	512
100	100 250 400 700	1000 1000 1000 1000	99.3	96.8	654
200	100 250 400 700	1000 1000 1000 1000	99.0	96.3	729
300	100 250 400 700	1000 1000 1000 1000	98.8	96.1	774

TABLE 2. The comparisons for 300 Hz poisson distribution patterns (average accuracy for 4000 patterns).

Neural encoding-classifier model	Training accuracy (%)	Testing accuracy (%)
Proposed ATM+2-phases classifier	**98.8**	**96.1**
Spiking CNN-HMM [74]	98.6	95.6
AER silicon cochlea-SVM [76]	97.1	95.0
SuperSpike [75]	99.0	96.1
Deep SNN [73]	**99.1**	**96.15**
SpiKL-IP (With IP, Reservoir size = 540) [51]	96.7	93.3

while their time complexity and the number of employed parameters are much larger than the ATM due to the recurrent and deep layers included in these models.

C. AUDIO SIGNALS RECOGNITION

Auditory signals are good candidates for testing SNNs due to their continuous, dynamic, and time properties. Compared with static image datasets, speech datasets are more similar to real-world stimuli. However, employing the audio signals datasets in research is much less compared to image datasets. Moreover, static image datasets, such as the famous MNIST [77], have no implicit time information required for pattern classification using the SNNs. This provides motivations to employ audio datasets that implicitly contain spatiotemporal information. Moreover, datasets have to be general and complex enough to simulate real-world problems.

Most studies of audio recognition are optimized for specific objectives. Specifically, in the SNN applications for

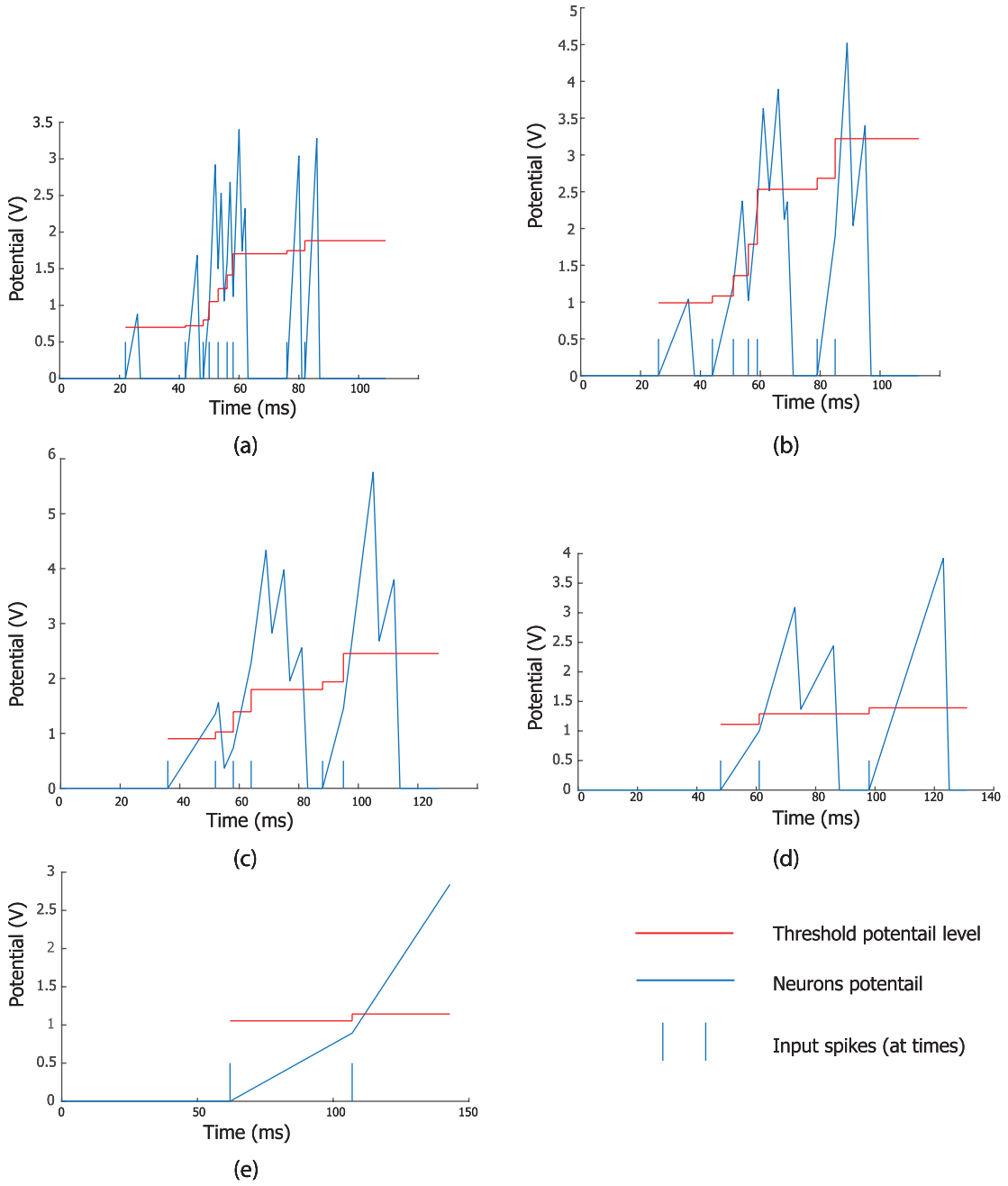

FIGURE 14. An example of the ATM stages and feature extraction of an input spike train (with original input times are used).

speech recognition [78], Mel-Frequency Cepstral Coefficients (MFCCs) are frequently used as the spectral representation for speech recognition. Other researchers tried to employ a biologically plausible cochlear filter bank, but they are either analog filters [79], or were studied in a spike-driven SNN system [80]. Moreover, [81] successfully implemented a silicon cochlear for event-driven audio sensing.

In this section, the experiments are based on two remarkable audio datasets: TIDIGITS [82] and RWCP [83].

The TIDIGITS [82] is a speech dataset of spoken words for speaker-independent speech recognition. The speakers are of both genders with different ages (adults and children) and various American English accents. Thus, the corpus provides sufficient speaker diversity and becomes the most common benchmark dataset in ASR research. It contains a vocabulary of 11 spoken words of isolated digits where each utterance has one individual spoken word, including '0'-'9' and 'oh'. Each word in the TIDIGITS is repeated 450 times, where 2464 (224 × 11) and 2486 (226 × 11) isolated words utterances for the training and testing set, respectively. However, in the study of neuromorphic computing, there are other speech datasets such as the N-TIDIGITS [84] that was designed for SNN benchmarking, but it is relatively small compared with the TIDIGITS. Thus, the TIDIGITS

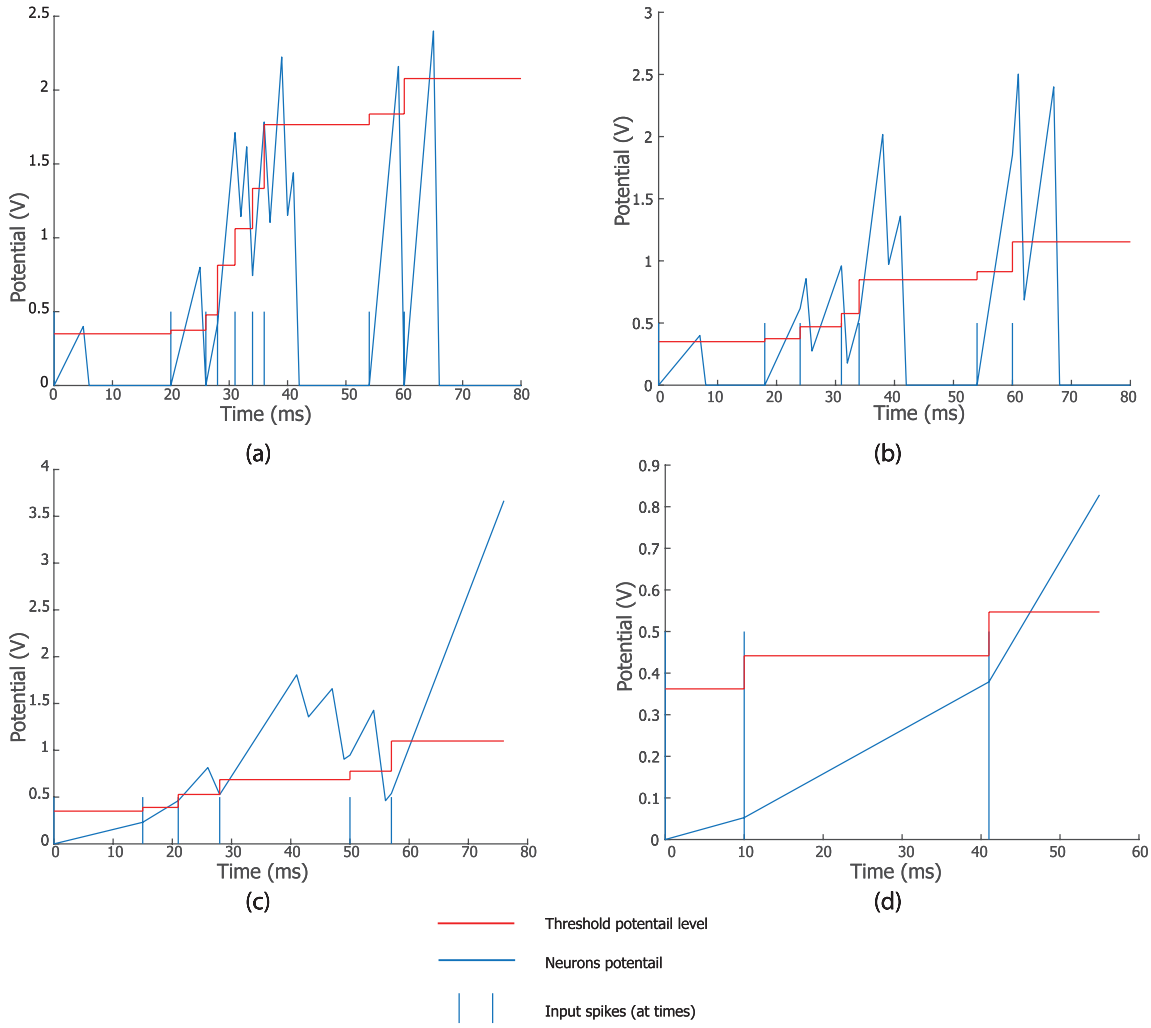

FIGURE 15. An example of ATM stages and feature extraction of an input spike train (with input times referenced in each stage to zero).

dataset is employed as one of the main benchmarks for ASR research [85]–[87].

The RWCP dataset [83] is a non-speech natural sound dataset recorded in an anechoic room composed of sound samples with rich and diverse frequency components. Therefore, there is a diversity of natural environmental sound categories in this dataset. The sound signals are available in RAW format (16 kHz and 48 kHz, 16 bit, Mono). Therefore, the RWCP dataset provides real-world schemes as sounds cover broader frequency ranges than speech. Each RWCP clip is encoded into several spike trains representing one pattern. Ten types of sounds were selected for the simulations from the RWCP dataset. Each type has 40 sound instances, where 20 samples are used for training and the other 20 samples for testing. Those instances are sounds of bells, bottle, buzzer, cymbals, horn, kara, metal, phone, ring, and whistle because they have a relatively longer time than other sounds in the RWCP dataset.

As shown in Fig. 16, the classification task of typical patterns, including audio patterns, consists of three main stages: the encoding of real-world signals, feature extraction, and pattern classification. Therefore, the encoding scheme may significantly simplify the duty of the following classification task. The method proposed in [88] is employed to encode the input audio stimulus into spike trains. That method uses a Mel-spaced tuned filter bank with special function filters. These filters respond to transient changes in the input signals. They can detect onsets, offsets, and peaks of transient changes in the input signals [88]. The output of these filters passes predetermined levels from below, above, and maximum critical value, respectively.

Comparisons between the testing result of the proposed model and other state-of-the-art methods for the RWCP dataset are shown in Table 4.

In this experiment, all onset, offset, and peak output times of each filter, belonging to a filter bank consisting of 20 band-pass filters, are used. Thus, the filter bank generates outputs of 20 spike trains. Consequently, These spike trains are used by the ATM to extract their features. The filter bank center frequencies are up to 8 kHz for speech signals

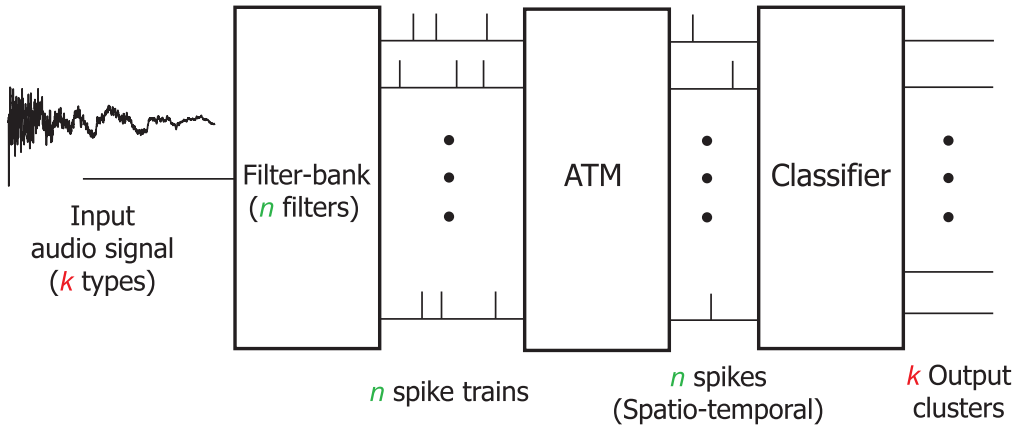

FIGURE 16. A schematic diagram represents encoding, feature extraction, and clustering of audio signals.

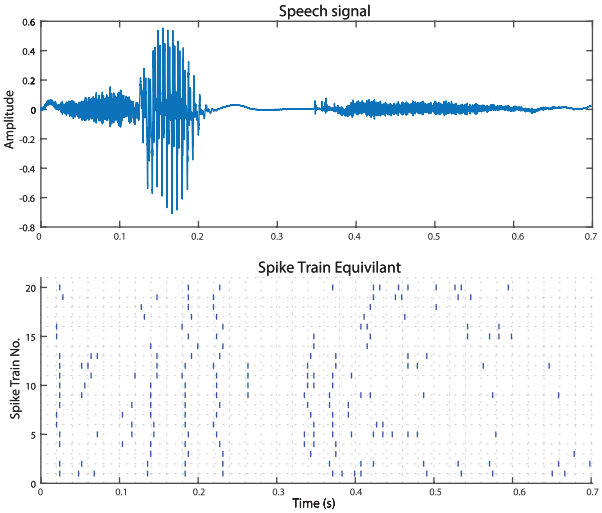

FIGURE 17. A word six from the TIDIGITS dataset recorded by an adult male and its conversion into 20 spike trains using 20 filter banks.

TABLE 3. Comparisons between the testing results of the proposed model and other methods for TIDIGITS dataset.

Neural encoding-classifier models	Testing accuracy (%)
Proposed ATM+2-phases classifier	**97.64**
BAE-MPDAL [85]	97.4
Liquid state machine [90]	92.3
SNN-SVM [91]	91.0
Spiking CNN-HMM [74]	96.0
AER silicon cochlea-SVM [76]	95.6
SOM-SNN [89]	97.4
MFCC and GRU RNN [84]	**97.75**
AER Silicon Cochlea & Deep RNN [92]	96.10
SpiKL-IP (With IP, Reservoir size 540) [51]	94.38
Spiking MLPs [50]	94.38
SFNN-IP (3 hidden layers) [48]	**97.7**

in the TIDIGITS dataset, with each filter having a bandwidth of 400 Hz. For the sound signals of the RWCP dataset, the filter bank center frequencies are up to 16 kHz, and each filter has a bandwidth of 800 Hz. Finally, the output of the ATM, in the shape of spatiotemporal patterns for the various audio utterances, is used as input patterns for the classifier.

Using the proposed encoding scheme for both datasets, each pattern in the datasets is finally converted into a spatiotemporal pattern that can be classified based on one spike per input. It indicates that when information is encoded in both temporal (spike time) and spatial (one spike per neuron) domain, the encoding scheme can project the inputs to another dimension, which takes some of the workloads of the subsequent classification stages. Thus, dimension reduction using the ATM stages is necessary for the classifier to cluster different audio categories. If the experiments are performed directly on the input spike trains of the audio signals, the SNN classifier is unable to classify such low-level spatiotemporal spike patterns, and only achieves very low classification accuracy for spike patterns [89].

An example of the TIDIGITS dataset is shown in Figure 17 for the speech signal and its equivalent 20 spike trains using the employed filter bank method. After extracting the features in all spike trains using the ATM, the spatiotemporal spike patterns are fed into the classifier. The classifier uses a temporal learning rule to fire spikes for patterns from the desired cluster while remaining silent for patterns of other clusters. In these experiments, the number of output clusters is 11 that equals the number of speech classes. It is found by simulations, each word needs an average of 4 ATM stages for converting 20 spike trains into a spatiotemporal pattern contains 20 spikes.

Comparisons between the testing results of the proposed model and other state-of-the-art methods for TIDIGITS dataset are stated in Table 3.

As shown in Table 3, it is encouraging to note that the proposed model achieves an accuracy of 97.64%, which is competitive with other bio-inspired and deep learning models for the TIDIGITS dataset. It is noticed that the traditional RNN based system offers a competitive accuracy of 97.90% [84]. However, the proposed model is fundamentally different from traditional RNN learning approaches in many aspects such as input method and training complexity. Moreover, the time complexity of the RNN is much higher than the proposed

TABLE 4. Comparisons between the testing result of the proposed model and other methods for the RWCP dataset.

Neural encodings-classifier model	Testing accuracy (%)
Proposed ATM+2-phases classifier	**99.50**
BAE-tempotron [85]	95.03
MFCC-HMM [94] (with noise inputs)	47.30
LSF-SNN [94]	98.50
LTF-SNN [95]	97.50
DKP-SNN [96]	99.10
SOM-SNN (Maximum-Margin Tempotron) [89]	**99.60**
SFNN-IP (3 hidden layers) [48]	99.50

ATM models due to the multiple recurrent stages in the RNN model. The inputs in the ATM are direct spike trains, while inputs in the traditional RNN are spectrogram images [84].

Although modern ASR learning techniques perform fairly fine under noise-free conditions, it is a challenging task for these techniques to recognize audio signals accurately in noisy surroundings. To discuss this problem, I investigate training of the proposed model using noisy audio signals. The reason for such a technique is that with training examples gathered from various noisy backgrounds, the model may be improved to extract unique features and become robust to noise. This procedure was proven to be efficient for different ANN models with a reasonable trade-off in performance for clean sound data [93]. Here, I investigate its generalizability to the proposed model under noisy environments.

To estimate the accuracy and robustness of the proposed model, different levels of Gaussian noise, measured by Signal-to-Noise Ratio (SNR) in dB, are added to the original speech signals. Figure 18 shows the classification accuracy under different noise levels and for noise-free conditions. Moreover, Fig. 18 represents a comparison between the different number of filters in the encoding filter bank. It is noticed that using fewer filters, 10 filters, leads to less accuracy compared to the employing of 20 filters. The reason is that when using a few filters, the bandwidth of each filter increases. In the case of 10 filters, the bandwidth for each filter is 800 Hz instead of 400 Hz in the case of 20 filters. Thus, the result shows lower accuracy when using 10 filters. On the other hand, when increasing the number of filters to 40, there is no remarkable accuracy increase. However, increasing the number of filters leads to an increase in computation complexity and hardware implementation costs.

Comparisons between the testing result of the proposed model and other state-of-the-art methods for the RWCP dataset are shown in Table 4.

As shown in Table 4, the proposed ATM model achieves a test accuracy of 99.50%, which is competitive with other deep learning and modern SNN-based models. However, the CNN models are trained using spectrogram images instead of directly modeling the temporal characteristics of speech signals. Despite the CNNs high accuracy on this dataset, it may be challenging to use them in the recognition of audio signals for a long period as the temporal structures will be changed unpredictably due to the mandatory rescaling of the spectrogram images [97]. On the other hand, RNN and LSTM models can extract the temporal characteristics directly. However, these models are hard to train for long-time audio signals due to the vanishing and exploding gradient problem [98]. The LSF-SNN and The LTF-SNN [95] classified the audio signals indirectly by recognizing the spectral characteristics in the spectrogram images, and then converting these characteristics into a spatiotemporal spike pattern for classification by an SNN classifier. Thus, comparing to the previous models, the ATM can extract the key features embedded in the audio signals directly in a more biologically plausible way.

D. COMPUTATIONAL COMPLEXITY OF THE ATM AND CLASSIFIER

In this section, I provide discussions and analysis for the results in terms of model performance and computation complexity.

The main concern is that the threshold values are calculated for each of ATM neurons while their membrane potentials are updated accordingly.

In each of the ATM neurons there are two different types of computations. The threshold is updated with respect to input spike times using Eq. (3), while the neuron potential is recalculated at every time step in simulation using Eq. (2). Therefore, according to Eq. (3), at each spike t_i in the input spike trains, in addition to some fixed parameters like τ_1, τ_2, and δ, the ATM neurons calculate the current update amount of the threshold $\vartheta_{th}(t_i)$ based on the spike input times t_i and t_{i-1}, and previous threshold $\vartheta_{th}(t_{i-1})$. Moreover, according to Eq. (2), at each time step it regulates the changes of the neuron membrane potential according to τ_1 and τ_2 to change the firing behavior of neurons. On the other hand, as stated previously, for the whole ATM neurons, τ_1, τ_2, and δ are updated from one neuron stage to another.

In general, the input dimension of the ATM is consistent with the dimension of the input spike trains. Therefore, the amount of computation brought by the ATM threshold in Eq. (3) and the potential update in Eq. (2) is proportional to the number of neurons in the ATM n_a, number of input spike trains T, and the average number of spikes in these input spike trains S_{avg}; then the computational complexity of the ATM is $O(2 \cdot n_a \cdot T \cdot S_{avg})$.

For the classifier, only the weights are changed locally for each neuron while the other parameters, including the threshold level, β_1, and β_2 are kept constant during the learning process. Thus, assuming the number of classifier neurons is n_c, the computational complexity of the classifier is $O(2n_c \cdot T)$. because the calculations are done twice according to Eq. (4).

Moreover, regarding the time of simulation, if the number of used time bins are b. All the ATM plus the classifier calculation are repeated for each bin time.

As a numerical example, assume $T = 20$, $S_{avg} = 100$, $n_a = 4$ stages $\times 20$ neurons in each stage, and $n_c = 250$,

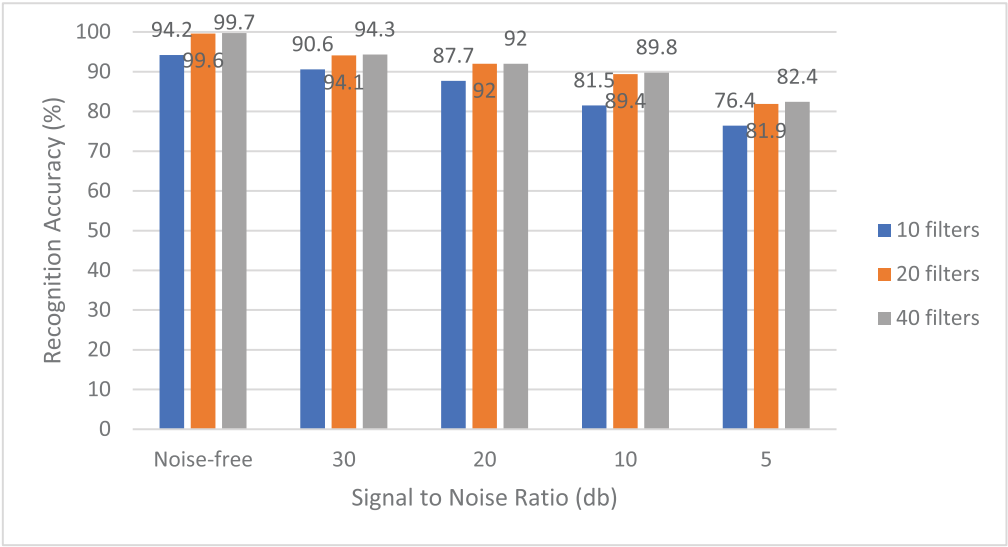

FIGURE 18. The training accuracy for the TIDIGITS dataset using different signal-to-noise ratio and different numbers of filter banks.

then the needed calculations are 320,000 for the ATM and 10000 for the classifier as a maximum.

Thus, in the case of ASR, the proposed ATM has two main points of advantages over CNNs: 1. The preprocessing of the input signals is done using fewer transformations, so more accurate information is transferred to the output especially for real-time applications. 2. The feature extraction is done more simply from the input signals to the final output, as shown in examples.

VI. CONCLUSION

In this research, a new SNN model is proposed using the powerful capabilities of spiking neurons. The two main targets are: model applicability for real-time applications, and using input spikes with no much conversion to keep the input information as possible.

In this model, the ATM is proposed to extract the main features of input patterns before being clustered by the classifier module. As the input patterns are in the shape of spike trains, the extraction is done according to a particular algorithm and parameters. This algorithm leads to unique output times representing time features in the input spike trains. The proposed ATM mainly depends on the adaptation of the neuron threshold to make the extraction follow biological rules. Demonstrations of the proposed method are done using multiple spike trains with different Poisson distributions and a real-world audio dataset. Moreover, the classification of those input patterns is done using an SNN classifier attached to the end of the proposed model. Promising results and discussions are shown for demonstrating the capabilities of the proposed model and algorithm.

Although the results proved quite illuminating and competing with other state-of-the-art models such as CNN and RNN, the model is in its first shape and needs more effort. However, future work has to be done to enhance the model. The model has to be modified to deal with more complex and long-time audio datasets such as the TIMIT dataset.

ACKNOWLEDGMENT

The author would like to thank Deanship of Scientific Research at Umm Al-Qura University (Project ID 43308002).

REFERENCES

[1] R. Mu and X. Zeng, "A review of deep learning research," *KSII Trans. Internet Inf. Syst.*, vol. 13, no. 4, pp. 1738–1764, 2019.
[2] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015.
[3] P. Dayan and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling*. Cambridge, MA, USA: MIT Press, 2001.
[4] W. Maass and T. Natschläger, "Networks of spiking neurons can emulate arbitrary hopfield nets in temporal coding," *Netw., Comput. Neural Syst.*, vol. 8, no. 4, pp. 355–371, Jan. 1997.
[5] W. Gerstner and W. M. Kistler, *Spiking Neuron Models*. Cambridge, U.K.: Cambridge Univ. Press, 2002. [Online]. Available: http://ebooks.cambridge.org/ref/id/CBO9780511815706
[6] H. H. Amin, *Spiking Neural Networks Learning, Applications, and Analysis*. Moldova: LAP LAMBERT Academic Publishing, 2011.
[7] H. H. Amin, "Spiking neural network inter-spike time based decoding scheme," *IEICE Trans. Inf. Syst.*, vol. E88-D, no. 8, pp. 1893–1902, Aug. 2005.
[8] A. Taherkhani, A. Belatreche, Y. Li, G. Cosma, L. P. Maguire, and T. M. McGinnity, "A review of learning in biologically plausible spiking neural networks," *Neural Netw.*, vol. 122, pp. 253–272, Feb. 2020, doi: 10.1016/j.neunet.2019.09.036.
[9] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Netw.*, vol. 111, pp. 47–63, Mar. 2019, doi: 10.1016/j.neunet.2018.12.002.
[10] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003.
[11] K. S. Fathima and R. J. A. Kumar, "FPGA implementation of optimized spiking neural network for efficient speak recognition system," *Int. J. Advance Res. Innov.*, vol. 3, no. 1, pp. 216–220, 2015.
[12] A. Vanarse, A. Osseiran, and A. Rassau, "An investigation into spike-based neuromorphic approaches for artificial olfactory systems," *Sensors (Switzerland)*, vol. 17, no. 11, 2017, doi: 10.3390/s17112591.
[13] N. Kasabov, N. M. Scott, E. Tu, S. Marks, N. Sengupta, E. Capecci, M. Othman, M. G. Doborjeh, N. Murli, R. Hartono, J. I. Espinosa-Ramos, L. Zhou, F. B. Alvi, G. Wang, D. Taylor, V. Feigin, S. Gulyaev, M. Mahmoud, Z.-G. Hou, and J. Yang, "Evolving spatio-temporal data machines based on the NeuCube neuromorphic framework: Design methodology and selected applications," *Neural Netw.*, vol. 78, pp. 1–14, Jun. 2016, doi: 10.1016/j.neunet.2015.09.011.
[14] S. Sheik, M. Pfeiffer, F. Stefanini, and G. Indiveri, "Spatio-temporal spike pattern classification in neuromorphic systems," in *Biomimetic and Biohybrid Systems*, N. F. Lepora, A. Mura, H. G. Krapp, P. F. M. J. Verschure, and T. J. Prescott, Eds. Berlin, Germany: Springer, 2013, pp. 262–273.

[15] M. A. Nuno-Maganda, M. Arias-Estrada, C. Torres-Huitzil, H. H. Aviles-Arriaga, Y. Hernández-Mier, and M. Morales-Sandoval, "A hardware architecture for image clustering using spiking neural networks," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, Aug. 2012, pp. 261–266.
[16] A. Basu, S. Shuo, H. Zhou, M. H. Lim, and G.-B. Huang, "Silicon spiking neurons for hardware implementation of extreme learning machines," *Neurocomputing*, vol. 102, pp. 125–134, Feb. 2013, doi: 10.1016/j.neucom.2012.01.042.
[17] C. Lee, P. Panda, G. Srinivasan, and K. Roy, "Training deep spiking convolutional neural networks with STDP-based unsupervised pre-training followed by supervised fine-tuning," *Frontiers Neurosci.*, vol. 12, p. 435, Aug. 2018.
[18] D. Liu and S. Yue, "Video-based disguise face recognition based on deep spiking neural network," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.
[19] J. Li, W. Hu, Y. Yuan, H. Huo, and T. Fang, "Bio-inspired deep spiking neural network for image classification," in *Neural Information Processing*, D. Liu, S. Xie, Y. Li, D. Zhao, and E.-S. M. El-Alfy, Eds. Berlin, Germany: Springer, 2017, pp. 294–304.
[20] 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 Neurosci.*, vol. 7, pp. 1–13, Oct. 2013.
[21] M. Fatahi, M. Ahmadi, A. Ahmadi, M. Shahsavari, and P. Devienne, "Towards an spiking deep belief network for face recognition application," in *Proc. 6th Int. Conf. Comput. Knowl. Eng. (ICCKE)*, Oct. 2016, pp. 153–158.
[22] Y. Xu, H. Tang, J. Xing, and H. Li, "Spike trains encoding and threshold rescaling method for deep spiking neural networks," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Nov. 2017, pp. 1–6.
[23] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–53065, 2019.
[24] B. Petro, N. Kasabov, and R. M. Kiss, "Selection and optimization of temporal spike encoding methods for spiking neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 2, pp. 358–370, Apr. 2019.
[25] H. Paugam-Moisy and S. Bohte, "Computing with spiking neuron networks," in *Handbook of Natural Computing*, G. Rozenberg, T. Bäck, and J. N. Kok, Eds. Berlin, Germany: Springer, 2012, pp. 335–376, doi: 10.1007/978-3-540-92910-9_10.
[26] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Competitive STDP-based spike pattern learning," *Neural Comput.*, vol. 21, no. 5, pp. 1259–1276, May 2009.
[27] A. Sboev, D. Vlasov, R. Rybka, and A. Serenko, "Solving a classification task by spiking neurons with STDP and temporal coding," *Procedia Comput. Sci.*, vol. 123, pp. 494–500, Jan. 2018, doi: 10.1016/j.procs.2018.01.075.
[28] X. Wang, X. Lin, and X. Dang, "A delay learning algorithm based on spike train kernels for spiking neurons," *Frontiers Neurosci.*, vol. 13, p. 252, Mar. 2019.
[29] F. Devalle, E. Montbrió, and D. Pazó, "Dynamics of a large system of spiking neurons with synaptic delay," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 98, no. 4, pp. 1–14, Oct. 2018.
[30] S. Luccioli, D. Angulo-Garcia, and A. Torcini, "Neural activity of heterogeneous inhibitory spiking networks with delay," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 99, no. 5, pp. 1–13, 2019. [Online]. Available: http://arxiv.org/abs/1902.03801
[31] A. Taherkhani, A. Belatreche, Y. Li, and L. P. Maguire, "DL-ReSuMe: A delay learning-based remote supervised method for spiking neurons," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 12, pp. 3137–3149, Dec. 2015.
[32] C. Huang, A. Resnik, T. Celikel, and B. Englitz, "Adaptive spike threshold enables robust and temporally precise neuronal encoding," *PLoS Comput. Biol.*, vol. 12, no. 6, pp. 1–25, 2016.
[33] J. Benda, L. Maler, and A. Longtin, "Linear versus nonlinear signal transmission in neuron models with adaptation currents or dynamic thresholds," *J. Neurophysio.*, vol. 104, no. 5, pp. 2806–2820, Nov. 2010.
[34] Y. Yang, A. Kamboh, and J. M. Andrew, "Adaptive threshold spike detection using stationary wavelet transform for neural recording implants," in *Proc. Biomed. Circuits Syst. Conf. (BioCAS)*, Nov. 2010, pp. 9–12.
[35] D. Bollé and R. Heylen, "Adaptive thresholds for neural networks with synaptic noise," *Int. J. Neural Syst.*, vol. 17, no. 4, pp. 241–252, Aug. 2007.
[36] C. Ganguly and S. Chakrabarti, "A leaky integrate and fire model for spike generation in a neuron with variable threshold and multiple-input-single-output configuration," *Trans. Emerg. Telecommun. Technol.*, vol. 30, no. 7, p. e3561, Jun. 2019.
[37] P. Falez, P. Tirilly, I. M. Bilasco, P. Devienne, and P. Boulet, "Multi-layered spiking neural network with target timestamp threshold adaptation and STDP," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2019, pp. 1–8. [Online]. Available: http://arxiv.org/abs/1904.01908
[38] H. Peng, J. Wang, M. J. Pérez-Jiménez, and A. Riscos-Núñez, "Dynamic threshold neural P systems," *Knowl.-Based Syst.*, vol. 163, pp. 875–884, Jan. 2019.
[39] R. Kobayashi, "Made-to-order spiking neuron model equipped with a multi-timescale adaptive threshold," *Frontiers Comput. Neurosci.*, vol. 3, pp. 1–11, Jul. 2009.
[40] Y. Yang, C. S. Boling, A. M. Kamboh, and A. J. Mason, "Adaptive threshold neural spike detector using stationary wavelet transform in CMOS," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 23, no. 6, pp. 946–955, Nov. 2015.
[41] J. Ladenbauer, M. Augustin, and K. Obermayer, "How adaptation currents change threshold, gain and variability of neuronal spiking," *BMC Neurosci.*, vol. 14, no. 1, p. P299, 2013. [Online]. Available: http://arxiv.org/abs/1307.5728
[42] M. J. Chacron, B. Lindner, and A. Longtin, "Threshold fatigue and information transfer," *J. Comput. Neurosci.*, vol. 23, no. 3, pp. 301–311, Dec. 2007.
[43] M. G. Metzen, R. Krahe, and M. J. Chacron, "Burst firing in the electrosensory system of gymnotiform weakly electric fish: Mechanisms and functional roles," *Frontiers Comput. Neurosci.*, vol. 10, p. 81, Aug. 2016.
[44] B. Fontaine, J. L. Peña, and R. Brette, "Spike-threshold adaptation predicted by membrane potential dynamics *in vivo*," *PLoS Comput. Biol.*, vol. 10, no. 4, pp. 1–11, 2014.
[45] J. Benda and A. V. M. Herz, "A universal model for spike-frequency adaptation," *Neural Comput.*, vol. 15, no. 11, pp. 2523–2564, Nov. 2003.
[46] Y.-H. Liu and X.-J. Wang, "Spike-frequency adaptation of a generalized leaky integrate-and-fire model neuron," *J. Comput. Neurosci.*, vol. 10, no. 1, pp. 25–45, 2001.
[47] E. Marder, L. F. Abbott, G. G. Turrigiano, Z. Liu, and J. Golowasch, "Memory from the dynamics of intrinsic membrane currents," *Proc. Nat. Acad. Sci. USA*, vol. 93, no. 24, pp. 13481–13486, Nov. 1996.
[48] A. Zhang, H. Zhou, X. Li, and W. Zhu, "Fast and robust learning in spiking feed-forward neural networks based on intrinsic plasticity mechanism," *Neurocomputing*, vol. 365, pp. 102–112, Nov. 2019.
[49] X. Li, W. Wang, F. Xue, and Y. Song, "Computational modeling of spiking neural network with learning rules from STDP and intrinsic plasticity," *Phys. A, Stat. Mech. Appl.*, vol. 491, pp. 716–728, Feb. 2018.
[50] S. Zhang, A. Zhang, Y. Ma, and W. Zhu, "Intrinsic plasticity based inference acceleration for spiking multi-layer perceptron," *IEEE Access*, vol. 7, pp. 73685–73693, 2019.
[51] W. Zhang and P. Li, "Information-theoretic intrinsic plasticity for online unsupervised learning in spiking neural networks," *Frontiers Neurosci.*, vol. 13, pp. 1–14, Feb. 2019.
[52] R. Baddeley, L. F. Abbott, M. C. A. Booth, F. Sengpiel, T. Freeman, E. A. Wakeman, and E. T. Rolls, "Responses of neurons in primary and inferior temporal visual cortices to natural scenes," *Proc. Roy. Soc. London B, Biol. Sci.*, vol. 264, no. 1389, pp. 1775–1783, Dec. 1997.
[53] N. S. Desai, L. C. Rutherford, and G. G. Turrigiano, "Plasticity in the intrinsic excitability of cortical pyramidal neurons," *Nature Neurosci.*, vol. 2, no. 6, pp. 515–520, Jun. 1999.
[54] J. Zhao, X. Mao, and L. Chen, "Speech emotion recognition using deep 1D & 2D CNN LSTM networks," *Biomed. Signal Process. Control*, vol. 47, pp. 312–323, Jan. 2019.
[55] T. N. Sainath, B. Kingsbury, A.-R. Mohamed, G. E. Dahl, G. Saon, H. Soltau, T. Beran, A. Y. Aravkin, and B. Ramabhadran, "Improvements to deep convolutional neural networks for LVCSR," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Dec. 2013, pp. 315–320.
[56] O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Trans. Audio, Speech Language Process.*, vol. 22, no. 10, pp. 1533–1545, Oct. 2015.
[57] C. Cooney, A. Korik, R. Folli, and D. Coyle, "Evaluation of hyperparameter optimization in machine and deep learning methods for decoding imagined speech EEG," *Sensors*, vol. 20, no. 16, p. 4629, Aug. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/16/4629
[58] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.
[59] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting deep bottleneck features using stacked auto-encoders," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 3377–3381.

[60] H. M. Fayek, M. Lech, and L. Cavedon, "Evaluating deep learning architectures for speech emotion recognition," *Neural Netw.*, vol. 92, pp. 60–68, Aug. 2017.
[61] A. M. Badshah, J. Ahmad, N. Rahim, and S. W. Baik, "Speech emotion recognition from spectrograms with deep convolutional neural network," in *Proc. Int. Conf. Platform Technol. Service (PlatCon)*, Feb. 2017, pp. 1–5.
[62] Z. Li, Y. Ming, L. Yang, and J.-H. Xue, "Mutual-learning sequence-level knowledge distillation for automatic speech recognition," *Neurocomputing*, vol. 428, pp. 259–267, Mar. 2021, doi: 10.1016/j.neucom.2020.11.025.
[63] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015, *arXiv:1506.00019*. [Online]. Available: http://arxiv.org/abs/1506.00019
[64] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," 2014, *arXiv:1402.1128*. [Online]. Available: http://arxiv.org/abs/1402.1128
[65] J.-T. Chien and A. Misbullah, "Deep long short-term memory networks for speech recognition," in *Proc. 10th Int. Symp. Chin. Spoken Lang. Process. (ISCSLP)*, Oct. 2016, pp. 1–5.
[66] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke, "The microsoft 2017 conversational speech recognition system," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 5934–5938.
[67] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, and M. Bacchiani, "State-of-the-art speech recognition with sequence-to-sequence models," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 4774–4778.
[68] R. Brette, M. Rudolph, T. Carnevale, M. Hines, D. Beeman, J. M. Bower, M. Diesmann, A. Morrison, P. H. Goodman, F. C. Harris, and M. Zirpe, "Simulation of networks of spiking neurons: A review of tools and strategies," *J. Comput. Neurosci.*, vol. 23, no. 3, pp. 98–349, Dec. 2007.
[69] S. R. Lehky, "Decoding Poisson spike trains by Gaussian filtering," *Neural Comput.*, vol. 22, no. 5, pp. 1245–1271, May 2010.
[70] S. R. Lehky, "Bayesian estimation of stimulus responses in Poisson spike trains," *Neural Comput.*, vol. 16, no. 7, pp. 1325–1343, Jul. 2004.
[71] J. A. Wall, L. J. McDaid, L. P. Maguire, and T. M. McGinnity, "Spiking neural network model of sound localization using the interaural intensity difference," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 574–586, Apr. 2012. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6145692
[72] H. H. Amin, W. Deabes, and K. Bouazza, "Hybrid spiking neural model for clustering smart environment activities," in *Proc. IEEE 15th Int. Conf. Ind. Informat. (INDIN)*, Jul. 2017, pp. 206–211.
[73] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Direct training for spiking neural networks: Faster, larger, better," 2018, *arXiv:1809.05793*. [Online]. Available: http://arxiv.org/abs/1809.05793
[74] A. Tavanaei and A. S. Maida, "A spiking network that learns to extract spike signatures from speech signals," *Neurocomputing*, vol. 240, pp. 191–199, May 2017.
[75] F. Zenke and S. Ganguli, "SuperSpike: Supervised learning in multilayer spiking neural networks," *Neural Comput.*, vol. 30, no. 6, pp. 1514–1541, Jun. 2018.
[76] M. Abdollahi and S.-C. Liu, "Speaker-independent isolated digit recognition using an AER silicon cochlea," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, Nov. 2011, pp. 269–272.
[77] Y. Lecun, C. Cortes, and C. Burges, "THE MNIST database of handwritten digits," Courant Inst. Math. Sci., New York, NY, USA, 1998, pp. 1–10. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[78] K. A. Darabkh, L. Haddad, S. Z. Sweidan, M. Hawa, R. Saifan, and S. H. Alnabelsi, "An efficient speech recognition system for arm-disabled students based on isolated words," *Comput. Appl. Eng. Educ.*, vol. 26, no. 2, pp. 285–301, Mar. 2018.
[79] S.-C. Liu and T. Delbruck, "Neuromorphic sensory systems," *Current Opinion Neurobiol.*, vol. 20, no. 3, pp. 288–295, Jun. 2010.
[80] S. Loiselle, J. Rouat, D. Pressnitzer, and S. Thorpe, "Exploration of rank order coding with spiking neural networks for speech recognition," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2005, pp. 2076–2080.
[81] M. Yang, C. H. Chien, T. Delbruck, and S. C. Liu, "A 0.5 V 55 μW 64 $\times$ 2 channel binaural silicon cochlea for event-driven stereo-audio sensing," *IEEE J. Solid-State Circuits*, vol. 51, no. 11, pp. 2554–2569, Sep. 2016.
[82] R. G. Leonard and G. Doddington, "Tidigits speech corpus," Linguistic Data Consortium, Philadelphia, PA, USA, Tech. Rep. TIDIGITS LDC93S10, 1993.
[83] S. Nakamura, K. Hiyane, F. Asano, T. Nishiura, and T. Yamada, "Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition," in *Proc. 2nd Int. Conf. Lang. Resour. Eval. (LREC)*. Athens, Greece: ELRA, May 2000, pp. 965–968.
[84] J. Anumula, D. Neil, T. Delbruck, and S.-C. Liu, "Feature representations for neuromorphic audio spike streams," *Frontiers Neurosci.*, vol. 12, p. 23, Feb. 2018.
[85] Z. Pan, Y. Chua, J. Wu, M. Zhang, H. Li, and E. Ambikairajah, "An efficient and perceptually motivated auditory neural encoding and decoding algorithm for spiking neural networks," *Frontiers Neurosci.*, vol. 13, pp. 1–18, Jan. 2020.
[86] M. Dong, X. Huang, and B. Xu, "Unsupervised speech recognition through spike-timing-dependent plasticity in a convolutional spiking neural network," *PLoS ONE*, vol. 13, no. 11, pp. 1–19, 2018.
[87] J. Wu, Y. Chua, and H. Li, "A biologically plausible speech recognition framework based on spiking neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.
[88] H. H. Amin and R. H. Fujii, "Sound classification and function approximation using spiking neural networks," in *Advances in Intelligent Computing*, D.-S. Huang, X.-P. Zhang, and G.-B. Huang, Eds. Berlin, Germany: Springer, 2005, pp. 621–630.
[89] J. Wu, Y. Chua, M. Zhang, H. Li, and K. C. Tan, "A spiking neural network framework for robust sound classification," *Frontiers Neurosci.*, vol. 12, pp. 1–17, Nov. 2018.
[90] Y. Zhang, P. Li, Y. Jin, and Y. Choe, "A digital liquid state machine with biologically inspired learning and its application to speech recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2635–2649, Nov. 2015.
[91] A. Tavanaei and A. Maida, "Bio-inspired multi-layer spiking neural network extracts discriminative features from speech signals," in *Neural Information Processing*, D. Liu, S. Xie, Y. Li, D. Zhao, and E.-S. M. El-Alfy, Eds. Berlin, Germany: Springer, 2017, pp. 899–908.
[92] D. Neil, M. Pfeiffer, and S.-C. Liu, "Learning to be efficient: Algorithms for training low-latency, low-compute deep spiking neural networks," in *Proc. 31st Annu. ACM Symp. Appl. Comput.*, Apr. 2016, pp. 293–298.
[93] I. McLoughlin, H. Zhang, Z. Xie, Y. Song, and W. Xiao, "Robust sound event classification using deep neural networks," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 3, pp. 540–552, Mar. 2015.
[94] J. Dennis, Q. Yu, H. Tang, H. D. Tran, and H. Li, "Temporal coding of local spectrogram features for robust sound recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 803–807.
[95] R. Xiao, R. Yan, H. Tang, and K. C. Tan, "A spiking neural network model for sound recognition," in *Cognitive Systems and Signal Processing*, F. Sun, H. Liu, and D. Hu, Eds. Singapore: Springer, Jul. 2017, pp. 584–594.
[96] Y. Yao, Q. Yu, L. Wang, and J. Dang, "A spiking neural network with distributed keypoint encoding for robust sound recognition," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2019, pp. 1–8.
[97] R. Gütig and H. Sompolinsky, "Time-warp–invariant neuronal processing," *PLoS Biol.*, vol. 7, no. 7, Jul. 2009, Art. no. e1000141.
[98] G. Klaus, S. K. Rupesh, K. Jan, S. R. Bas, and S. Jürgen, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Jul. 2016.
[99] R. Xiao, R. Yan, H. Tang, and K. C. Tan, "A spiking neural network model for sound recognition," *Commun. Comput. Inf. Sci.*, vol. 710, pp. 584–594, 2017.

HESHAM H. AMIN received the B.Sc. degree (Hons.) in electrical and computer engineering from the Faculty of Engineering, Assiut University, Assiut, Egypt, in 1997, and the M.Sc. and Ph.D. degrees in computer engineering and science from Aizu University, Fukushima, Japan, in 2002 and 2005, respectively. From 2007 to 2009, he was a Visiting Researcher with the RIKEN Center for Brain Science, Wakoshi, Aizu University. He is currently an Assistant Professor with the Department of Computer Science, University College, Umm Al-Qura University, and on a leave from the Electrical Engineering Department, Faculty of Engineering, Aswan University, Egypt. His research interests include biological neural networks, deep learning, pattern recognition, speech processing and recognition, machine learning, and the Internet of Things.