Utilizing the Neuronal Behavior of Spiking Neurons to Recognize Music Signals Based on Time Coding Features

DHVANI SHAH, AJIT NARAYANAN, AND JOSAFATH ISRAEL ESPINOSA-RAMOS
School of Engineering, Computer and Mathematical Sciences, Auckland University of Technology, Auckland 1142, New Zealand
Corresponding author: Dhvani Shah (dshah@aut.ac.nz)

ABSTRACT This paper presents a Spiking Neural Network (SNN) architecture to distinguish two musical instruments: piano and violin. The acoustic characteristics of music such as frequency and time convey a lot of information that help humans in distinguishing music instruments within few seconds. SNNs are neural networks that work effectively with temporal data. In this study, 2-layer SNN temporal based architecture is implemented for instrument (piano and violin) recognition. Further, this research investigates the behaviour of spiking neurons for piano and violin samples through different spike based statistics. Additionally, a Gamma metric that utilises spike time information and Root Mean Square Error (RMSE) from the membrane potential are used for classification and recognition. SNN achieved an overall classification accuracy of 92.38% and 93.19%, indicating the potential of SNNs in this inherently temporal recognition and classification domain. On the other hand, we implemented rate-coding techniques using machine learning (ML) techniques. Through this research, we demonstrated that SNN are more effective than conventional ML methods for capturing important the acoustic characteristics of music such as frequency and time. Overall, this research showed the potential capability of temporal coding over rate coding techniques while processing spatial and temporal data.

INDEX TERMS Classification, music, spiking neurons, spiking neural networks, STDP, temporal data, unsupervised learning.

I. INTRODUCTION
When listening to a piece of music we can usually recognize the musical instruments involved with some degree of accuracy. Recognizing a single instrument can be quicker than recognizing multiple instruments depending on the nature of the instruments. For instance, it may take longer to identify four instruments (two lead guitars, bass guitar and drums). Recognizing a difference in type of instrument (e.g., guitar from drum) can be quicker than distinguishing two instruments of the same type (e.g., two wind instruments of clarinet and recorder). Given that music is a form of ordered sound as opposed to noise, human musical instrument recognition (HMIR) can be hypothesized to be based on separating pitch and loudness of music into component frequencies, including fundamental (the first amplitude component) and dominant (highest amplitude component) frequencies, and their ratios [1]. Another important aspect of HMIR is timbre or tone quality, which conveys the identity of musical instruments playing with the same pitch and loudness [2]. Trying to provide a quantitative method for measuring or capturing the qualitative aspect of timbre is an open research question, with appeal to spectral envelopes that identify parameters such as decay, sustain and transients [3]. However, timbre is usually considered the most important musical quality that helps us distinguish two different instruments playing the same note at the same loudness. Humans are good at recognizing monophonic (single instrument) as well as some polyphonic (multi-instrument) sources, with increasing difficulty for non-experts as the number of instruments grows [4], [5]. Humans have a qualitative sense of the music being played through appreciation of tonality, such as the key in which the music is being played and the pitch, or identification of musical tones to a scale, based on auditory perception [6]. Research has been conducted to study the cognitive functions of brain using music as a stimulus. Also, music affects us at an
emotional level and has the potential to evoke different kinds of feelings. Music has been considered as a therapeutic tool to treat different mental disorders by regulating the emotions through multiple neurobiological pathways [7]–[10]. Further, music contributes a lot towards a rich and happy life by reducing stress and pain, elevating mood, increasing IQ and mental alertness, and improving sleep, to mention just a few benefits [11].

Automatic musical instrument recognition (AMIR) is a relatively under-explored area of pattern recognition and machine learning in comparison to voice and image recognition. Successful applications of AMIR to single instrument identification can also be expected to play a role in, for example, voice recognition (identifying a speaker), automated detection and classification of birds from birdsong, and identification of possible malfunctions in mechanical systems. Also, methods for distinguishing individual sources from polyphonic sources in noisy environments (e.g. one voice trying to activate Alexa in a noisy environment of multiple voices) are also in the early stage of research, and successful AMIR can be expected to play an important role in these applications. AMIR does not involve intensive feature extraction and representation steps and thus saves classification time making the process easy. The approach of analysing musical samples (clips) to extract features for use in classification algorithms was established early in the field [12], [13]. Feature-based approaches for monophonic recognition dominated initial attempts in AMIR (e.g. [14]–[18]) using a variety of techniques such as support vector machines, Gaussian mixture models and k-nearest neighbour approaches. In the absence of an universally agreed set of music features on which to base classification, researchers have chosen their own feature sets, including those dealing with frequency, pitch, loudness and other aspects of spectral content. The same features are also typically used for polyphonic recognition [19]. Feature-based approaches using fuzzy clustering [20] and rule-based approaches [21] have also been tried. Advanced feature representations based on mel scale (non-linear scaling of frequency to match what the human ear can detect) and mel-frequency cepstrum (linear cosine transform of log power spectrum of mel scale) [22] can aid efficiency of feature extraction for longer signal sources.

Other advanced features based on local energies and linear prediction coefficients have also been used for AMIR [5], as well as zero crossing rates, auto-correlation, temporal centroids and other spectral features such as moments and slopes for use in machine learning [23]. Neural network, in particular, have been shown to be capable of using advanced vector-represented features for effective recognition [24]. More recently, there has been growing interest in the application of deep learning neural networks (DLNNs) for AMIR. DLNNs have proved successful in many image recognition tasks [25], [26] and their application to AMIR is still relatively new. DLNN approaches to AMIR are distinguished by the use of spectograms (representations of amplitude at different frequencies over time) rather than features extracted through pre-processing. The use of mel-spectograms (frequencies converted to mel scale to represent improved human hearing at lower frequencies than upper frequencies) is particularly appropriate for image recognition convolutional neural networks (CNNs), which apply small filters or kernels (typically, 3 × 3 or 5 × 5) to spectrogram images (typically, around 50 × 100 at the input level) in a series of convolution layers followed by compression through max-pooling in the search for features to use for classification in a final, fully connected layer [27], [28]. Feature extraction for AMIR is an essential part of CNN processing rather than a pre-processing step, even if it is not always clear what features have been learned. CNN filters convolutes across the samples and hence the sequential information in the music samples is lost whereas music lovers recognise a music instrument within few seconds by continually listening to the music [29].

The main issue for all AMIR techniques is how to represent the music samples. Currently, the two general ML approaches are: preprocess music samples to extract features for input to classifiers, or minimize preprocessing and allow the AMIR technique to extract whatever features it needs during learning. Preprocessing happens off-line and prior to learning, and replaces musical samples with numeric vectors representing features ready for ML input. One advantage is that multiple samples can be preprocessed prior to input. However, features useful for one problem, such as monophony, may not be useful for other problems involving polyphony and identification of instruments belonging to the same or different families. It is not known whether a minimally redundant common set of ML features, or a common taxonomy, will or can be found for AMIR. Minimal preprocessing, such as the use of spectograms, while having the advantage of allowing AMIR techniques to work with input that is closer to musical samples, can make AMIR techniques computation-ally demanding. Music passages have to be sampled first, and different sampling rates can affect accuracy of representation. Since music can contain frequencies as high as 12kHz or 13kHz, sampling rates of at least double that may need to be used to capture all frequencies and prevent aliasing. Four seconds of music can generate over 100,000 data samples for one instrument before conversion to spectrograms through Fast Fourier Transform (FFT). For that reason, the number of samples that can be input to ML for AMIR may need to be restricted.

While AMIR has resulted in reasonable to good ML performance (between 55% to 95% accuracy depending on type of classifier), one aspect that has not been addressed so far is the desirability of using a ML method that uses temporal processing as a key component of AMIR. That is, AMIR ML techniques use input representations that contain temporal information implicitly encoded as part of their data structure (e.g. spectograms) or learning presentation order (e.g. presenting vectors in a specific order). Spiking neural networks (SNNs), on the other hand, are known to effectively
learn sequential inputs using inter-spike interval times (more details below). So far there has been no attempt to explore SNNs for AMIR, which this study now aims to address. One advantage of an SNN over other ML techniques is that it arguably is ‘more biologically plausible’ [30], since brain neurons are not simple on-off or continuous value output devices. Instead of threshold or sigmoid-based computations, SNNs use the original Hodgkin-Huxley model of action potentials (spikes or pulses) being initiated and propagated in biological neural networks, thereby representing more closely the flow of chemical neurotransmitters in the synaptic cleft onto neuron ion channels. Also, the use of SNNs may more closely model the way that humans distinguish musical instruments by identifying differences in the temporal characteristics of instruments as they are received neurally, rather than storing the signals first in memory for analysis of features. The aim of this study is demonstrate, for the first time, the application of an SNN to demonstrate the feasibility of a more biologically plausible approach to AMIR. The task is to distinguish monophonic piano from violin. On the other hand, music play vital role in our lives. Music can inspire us, make us feel our emotions, heal us, make us sleep. Psychologists and Neuroscientists use music therapy for patients suffering from Parkinson, Depression and other mental disorders. This research explores AMIR domain by implementing a biologically plausible NN which can be further extended to study the impact of music on the brains of these patients [31], [32].

The first objective of this research is explore the capability of SNN in recognising the music samples. The second objective is analysing the behaviour of spike train to detect the spiking patterns. The third objective is to compare the rate-coding and temporal coding mechanisms in order to tale classification decisions. We initiate this research with a hypothesis that temporal dimension in spiking activity and membrane potential contains information that could potentially distinguish the musical instruments. To test this hypothesis by setting the above objectives, evolutionary strategies are used for the optimisation process throughout this research.

As far as we are aware, this is the first time that an SNN has been tried in AMIR. The aim of this paper is to demonstrate that an SNN can identify and distinguish a single musical instrument (piano) from another musical instrument (violin) when played separately (monophonic). Before further work is undertaken using SNNs to identify and distinguish two instruments played together (polyphonic), the fundamental principles of how to represent and encode music instrument signals to an SNN, as well as appropriate SNN architectural parameters, have to be first established before SNN generalization to polyphonic signals.

The paper is organised as follows: Section 2 introduces the core components of SNN and the techniques implemented used in this research for the recognition and classification task. Section 3 describes the experimental framework of this research that includes the proposed architecture. Section 4 describes the methodology and Section 5 discusses the results obtained. Finally, Section 6 highlights the major contribution of this research and the future enhancements to this study. Finally, Section 7 provides technical details of the proposed architecture.

II. SNN FUNDAMENTALS

Neurons are the basic processing units in the brain. Neurons are interconnected through synapses, which are electrically conductive junctions made up of gap junction channels. They send and receive information through pre-synaptic action potentials causing changes in the membrane potential of neurons that lead to post-synaptic electrical signals in the form of spikes. Different pre-synaptic patterns cause changes in the strength of synaptic connections, leading to spike-timing-dependent behaviour that can be useful in neural networks, such as SNNs, for modelling learning [33].

The music recognition system presented in this study implements Leaky-Integrate-Fire (LIF) [34] neuron model and Hebbian learning based Spike-Time Dependent Plasticity (STDP) [33]. Standard or simple IF represents membrane potential as a function for integrating temporally stochastic synaptic inputs or currents until a threshold is exceeded, at which point a spike is produced and the membrane potential returns to its resting state for the next cycle. Membrane potential in IF is therefore a ‘passive’ integrative component and may fire continuously in the presence of signals. LIF, or ‘forgetful’ IF, allows for membrane potential to rise in response to input but also to reduce (leak). The amount of membrane potential leakage over time, in conjunction with refractory effects during which the neuron is inactive, allows for the emergence of spiking rates and interspike intervals. More details are provided below.

The Implementation Appendix provides further details of how the SNN model was constructed. Also, this is the first time Gamma and RMSE factors haven been used for music classification and recognition using SNN and further details on these factors are provided below. The next part of this section describes the core components of the SNN architecture for instrument recognition.

A. LEAKY-INTEGRATE-FIRE MODEL

The LIF neuron model is one dimensional spiking neuron model with low computational cost [35]. The dynamics of LIF neuron models are defined by the following equation:

\[
\tau \frac{dv}{dt} = -v(t) + RI(t)
\]

where \(\frac{dv}{dt}\) represents the change in membrane potential, \(v(t)\) is the neuronal membrane potential at time \(t\), \(\tau\) is the membrane time constant, \(R\) is the membrane resistance and \(I(t)\) is the sum of current supplied by the input synapses (connection from the input layer to the hidden layer). \(I(t)\) is computed
using the following equation:

\[ I(t) = W \cdot S(t) \]  \hspace{1cm} (2)

where \( W = w_1, w_2, \ldots, w_n \) is the weight vector, \( S(t) = [s_1(t); s_2(t); \ldots; s_N(t)] \) is the spatio-temporal input spike pattern containing \( N \) input spike trains, \( s_i(t) \) for \( i = 1, 2, \ldots, N \). If it either contains 1 (spike) or 0 (no spike). The input current to the neighboring neurons is calculated only when there is a spike as shown in (3):

\[ s_i(t) = \sum_j \delta (t - t_{ij}) \]  \hspace{1cm} (3)

where \( t_{ij} \) is the firing time of the \( f = 1, 2, \ldots \) spike in the \( j \)th input spike train \( s_i(t) \), \( s_i(t) \) is applied to the \( j \)th synapse, and \( \delta(t) \) is the Dirac function. A spike is recorded when the membrane potential \( v(t) \) crosses the threshold value \( v_f \). The membrane potential is then reset to its resting state for a certain time limit known as the refractory period. Thus, LIF spiking neurons represent a hidden state in the form of membrane potential. The membrane potential acts as an implicit recurrence since it integrates inputs and passes them on to the next time step. This implicit recurrent allows the SNN to handle sequential data better than non-sequential neural nets at the cost of increased time complexity.

B. SYNAPTIC PLASTICITY

Synaptic plasticity, i.e. change in strength of the synapses, is the biological basis of learning and memory. This study adopts the unsupervised learning inspired by the Hebb Rule [36], where weight changes during the learning process are dependent on the time dimension. STDP is a variant of Hebbian unsupervised learning algorithm where the weights are adapted based on the relative timing of pre- and post-synaptic spikes. For two interconnected neurons the neuron which sends the signals is called as pre-synaptic neuron, whereas the signal receiving neuron is the post-synaptic neuron. The dynamics of STDP algorithm can be represented by equations (4) and (5):

\[ W(x) = A_+ \exp \frac{-x}{\tau_+} \quad \text{for} \quad t_i < t_j \]  \hspace{1cm} (4)

\[ W(x) = -A_- \exp \frac{x}{\tau_-} \quad \text{for} \quad t_i > t_j \]  \hspace{1cm} (5)

The synaptic weight \( W(x) \) will be potentiated if the pre-synaptic neuron \( t_i \) fires before the post-synaptic neuron, \( t_j \). The potentiation is a function of \( t \) which decays exponentially with a time constant \( \tau_+ \) and can be calculated by \( A_+ \exp \frac{-t}{\tau_+} \), where \( A_+ \) is the maximum synaptic change. On the other hand, if the post-synaptic neuron fires before the pre-synaptic neuron the weight of the synapse between these two neurons is decreased by a magnitude of \( -A_- \exp \frac{t}{\tau_-} \), where \( A_- \) indicates the maximum negative change and \( \tau_- \) is the time constant. Time constant variables for potentiation and depression are set to 10ms. Only the last (recent) firing time of the pre-synaptic neuron is considered for this STDP implementation.

C. CLASSIFICATION

At the end of each training process, the firing times and membrane potential of the output neuron are extracted for classification. Using the spike time, firing rate and Gamma (coincidence) factor [37] are calculated.

The coincidence factor [38] defined between two spike trains is described as:

\[ \Gamma = \frac{N_{\text{coinc}} - \left( N_{\text{data}} + N_{\text{SRM}} \right) \times N}{\frac{1}{2} \times \left( N_{\text{data}} + N_{\text{SRM}} \right) \times N} \]

where \( N_{\text{data}} \) is the number of spikes in the actual (training sample) spike train, \( N_{\text{SRM}} \) is the number of spikes in the predicted (test/validation sample) spike train, \( N_{\text{coinc}} \) is the number of coincidences between the two spike trains with precision \( \Delta \) (time window), and \( \left( N_{\text{coinc}} \right) = 2N_{\text{data}} \) is the number of expected coincidences generated with same rate \( v \) as the predicted spike train. The coincidence factor can range between 0 and 1 (both inclusive), where 0 means the least coincident between the two spike trains and 1 denotes the maximum value when both spike trains are identical to each other.

Root-mean-square error (RMSE), sum of membrane potentials, and signal-to-noise ratio (SNR) are computed from the membrane potential of the output neuron for input to various classifiers (Decision Tree, Support Vector Machine) and the temporal based metrics, as will be explained in the Results section.

III. RATE AND TEMPORAL CODING TECHNIQUES

IV. EXPERIMENTAL FRAMEWORK

This section introduces the proposed SNN architecture and explains the various methods used. First, the data is prepared in the format that can be readily used by SNN i.e. music signals are encoded into 2D binary spike trains. Spike trains are propagated into the network of spiking neurons. STDP learning is applied between the inputs and spiking neurons in the hidden layer; from the hidden layer to the output layer learning is applied between the inputs and spiking neurons. The coincidence factor [37] are calculated.

Fig. 1 represents the experimental methodology for the SNN architecture for music instrument recognition. Fig. 2 describes the SNN aspects of the architecture. The methodology is as follows (see Algorithm Setup in Implementation Appendix for full details of each step).

- **Music signals**: To reduce ripple effects we applied Hamming windowing technique. This is the first input to our architecture.

- **Encoding**: For encoding we used spike encoding algorithms: Ben’s Spiker algorithm (BSA) [39] and step-forward [40]. Each music sample (frequency components) is a 2D matrix representing the frequency value at an instant of time. The output from encoding algorithm is also a 2D matrix of spike trains for each sample.
FIGURE 1. Experimental Setup: Music signals are converted into binary data using encoding algorithm. These spatio-temporal binary data are fed into spiking neurons. The spike trains from this hidden layer is propagated to the output layer. STDP is used across the input, hidden and output learn for weight adjustments. Different rate and temporal coding techniques are tested and analysed at the output layer. The various parameters are optimised where accuracy being the fitness function.

FIGURE 2. SNN based Temporal Architecture: Membrane potential and spike timings used as the classification criterion.

- Input layer and Network of spiking neurons. The number of nodes in the input layer is equal to the number of columns in the sample. The hidden layer consists of LIF spiking neurons. All the input nodes are connected to all the spiking neurons. Synaptic weights are initialized randomly includes positive (80%) and negative connections (20%). The maximum and minimum limits of these synaptic weights are determined using Differential evolution (DE) optimization technique (Implementation Appendix). The sum of weighted inputs to each spiking neuron is calculated (Eq.1.) to measure their membrane potential (Eq.2.). The spikes of neurons from this layer are propagated to the succeeding layer in the next times-tamp. Detection of spike in the hidden layer instantiates the STDP learning rule. Based on the timings of spikes of the input nodes (pre-synaptic neurons) and the spike timings of spiking neurons in the hidden layer (post-synaptic neurons), synaptic potentiation (Eq.4.) or depression (Eq.5.) takes place and the synaptic weights are updated accordingly.

- Output Layer: This layer consists of only one spiking neuron that performs the classification task. All the spiking neurons in hidden layer are connected to this neuron. The synaptic connections are learned using STDP rule. For each sample, the membrane potential (voltage at each timepoint) and the spike train (spike/no spike event at each timepoint) of the output neuron are extracted. From this information, five metrics are computed for the next stage of classification:
  1) Firing Rate: It is calculated by dividing the total number of spikes of the output neuron by total stimulation time.
  2) Total Membrane Potential: This is the sum of membrane potential of the output neuron.
  3) Signal-to-Noise ratio (SNR): SNR is calculated from the membrane potential signal as the mean of the signal divided by its standard deviation.
  4) Gamma Factor: This metric considers the number of coincidences of spikes with a given time window, further sliding the window for length of
the spike train. The gamma factor considers the temporal information (spike times). This factor is calculated between the spike train of the output neuron of the test sample with all the spike trains of the output neuron of training samples. In other words, gamma factor is calculated between each pair of the test sample and all the training samples. The label of the training sample which has the highest gamma factor among all the pairs is assigned as the label to the test sample and denotes the prediction.

5) Root Mean Square Error (RMSE): The RMSE value is calculated between the membrane potential signal of output neuron of the test sample and all the membrane potential of output neuron of the training samples. Euclidean distance is calculated between the test sample and the training samples. The label of the closest training sample among all the pairs is assigned as label to the test label.

- Classifier, Accuracy and Optimisation. The metrics from the output layer are used for classification, as described below. A feedback cycle checks the accuracy of test results, with optimisation then used to fine-tune the SNN parameters using differential evolution (see Optimization Process in Implementation Appendix).

V. METHODOLOGY
A 2-layer feed-forward SNN network described in the previous section and trained using STDP learning rule is implemented for identifying unique rhythmic spiking patterns in the piano and violin signals

A. DATASET DESCRIPTION
The piano and violin music files were downloaded from an online website [41]. Each group consists of 5 samples and the time duration is 61 seconds. The music signals were loaded using librosa python library at the sampling frequency of 22050Hz as illustrated in Fig. 3 and Fig. 4.

Short-time Fourier transform (STFT) [42] was applied to extract frequency components from the music signals. Fig. 5 and Fig. 6 shows the frequency spectrum of the respective signals. In a spectrum representation plot, x-axis represents time; y-axis represents frequencies and colours represent the magnitude (amplitude) of the observed frequency at a particular time. STFT converts signals in a manner that the amplitude of given frequency at a given time is known and, in this case, STFT function returns data in the time-frequency domain with dimension of 1025 rows and 2628 columns. For the experimental demonstration only 20 columns (centre) have been selected to explore the capability of SNN with less computational cost. For the proper brain circuity to be developed, the neurons produce negative and positive feedback signals during the generation and flow of neurotransmitters. [43]. To incorporate this concept, we have 20 positive and 20 negative connections from the input to hidden layer in the proposed SNN architecture.

B. DATA ENCODING
The BSA encoding converts the signals (real number) into spike train (binary data). The neurons in brain transmit information to each other in the form of spikes i.e. 0 (no spike) and 1 (spike). STDP has advantages over back-propagation in SNN unsupervised learning with limited data by manipulating weights based on temporal correlations between timings of pre-synaptic and post-synaptic spikes. BP-based
approaches on the other hand require large datasets and more computational effort [44], [45].

Fig. 7 and Fig. 8 represent a violin signal and its corresponding spike encoding. The x-axis shows time duration and y-axis represents the amplitude in Fig. 7. The blue color in Fig. 8 indicates the occurrence of spike at that time instant on crossing the threshold value.

![FIGURE 7. Violin sample: x-axis represent time and y-axis represents the amplitude.](image)

![FIGURE 8. Spike representation for violin sample: x-axis represent time and y-axis indicate spike where 0.5 is the spike threshold. Absence of blue lines indicate no spike.](image)

**C. CLASSIFICATION AND EVALUATION**

The dataset is split into train (80%) and test (20%) sets. The network is initialized with input nodes equal to the number of columns i.e. 40 in the encoded data and the simulation time 1000 is equal to the number of rows. The hidden layer and the output layer are implemented using LIF neurons. Within the 80% training set, we have applied leave-one-out cross validation technique for evaluating the performance of network. BSA encoded data is then propagated throughout the network of spiking neurons. As the neuronal membrane potential crosses the threshold, a spike is emitted and spike time is recorded. For each fired post synaptic neuron, the synapses are strengthened if the pre-synaptic neuron is fired before the post-synaptic neuron and vice versa. Thus, the spiking neurons along with STDP produces spatio-temporal patterns.

**VI. RESULTS**

This section discuss results and different spike train analysis techniques showing rhythmic behaviour of spiking neurons thus distinguishing the piano and violin signals. This nature of spiking neurons is evident because of their capability to process spatio-temporal information. The spike train analysis reflects aspects of neural functioning to effect the learning process.

Fig. 9 and Fig. 10 shows the firing rate of all the neurons in network. We have 40 input nodes, 60 hidden neurons and 1 output neuron. The firing rate of the hidden and output neuron in the piano class lies within the range of 0.02 and 0.03 whereas for the violin class it lies between 0.06 and 0.07. The initial exploration of rate coding scheme led further analysis considering the temporal dimension.

![FIGURE 9. Firing rate for piano class.](image)

![FIGURE 10. Firing rate for violin class.](image)

In Fig. 11 and Fig. 12, the x-axis represents time component and y-axis represent the neurons. Blue dot symbolizes a spiking event i.e. the neuron has emitted a spike at the given time. Violin samples excite the network much greater than the piano class. This firing activity reveals the temporal pattern hidden in spiking behaviour of neurons and proves to encode more rich information than the rate-encoding mechanisms.

Inter Spike Interval (ISI) is the time difference between each two successive spike arrival times at the output neuron as illustrated in Fig. 13 and Fig. 14. The x-axis shows number of times the neuron is fired, and y-axis shows the difference between spike arriving at time t, t + 1 and so on. There is a
noticeable difference in the pattern of ISI for both the classes. Extracting information from the temporal encoding shows the power to be useful for classification purposes.

Our main focus at every point in this research was the maximum utilization of all information obtained from the spiking neurons. Averaging the number of spikes over the entire time length i.e. FR, calculating the sum and SNR from the membrane potential resulted in loss of temporal data. Fig. 13, Fig. 14, Fig. 15, and Fig. 16 clearly illustrates the rhythmic behaviour hidden in the spike train. Thus, we realized the need of incorporating a metric that processes the temporal information and can aid in the classification performance. Hence, we decided to incorporate a coincidence factor called Gamma ($\Gamma$) factor [37] as the criteria to classify the samples.

The gamma factor considers the temporal information (spike times) which produced a classification accuracy of 98.33% with optimal parameter setting as shown below in Table 2. Further, the membrane potential signal contains the value of voltage at each time step. Hence, root mean square error was calculated at each time step from the signal.

As previous experiments, in the above-mentioned gamma and RMSE experiment setting, the network was trained using
TABLE 1. Performance evaluation of ML classifiers.

| Sr. No. | Classifier | Metric | Threshold | Number of neurons | Weights (min, max) | Mean Accuracy (%) |
|---------|------------|--------|-----------|-------------------|-------------------|-------------------|
| 2       | SVM        | FR     | 0.542374  | 74                | (0.01, 0.366542)  | 75.5              |
|         |            | SNR    | 0.031005  | 49                | (0.01, 0.109176)  | 59.34             |
| 3       | Decision Tree | MP_SUM | 0.32567   | 67                | (0.03, 0.6958)    | 72.87             |
|         |            | SNR    | 0.282274  | 75                | (−0.17485, 0.638234) | 78.55             |
|         |            | MP_SUM | 0.276836  | 77                | (−0.62101, 0.880977) | 78.87             |

TABLE 2. Performance evaluation of temporal metrics: gamma and RMSE.

| Sr. No. | Metric | Threshold | Number of neurons | Weights (min, max) | Mean Accuracy (%) |
|---------|--------|-----------|-------------------|-------------------|-------------------|
| 1       | Gamma  | 0.311189  | 54                | (−0.508710, 0.880842) | 92.38             |
| 2       | RMSE   | 0.3964    | 67                | (−0.288482, 0.9027)    | 93.19             |

TABLE 3. Gamma and RMSE performance.

| Sr. No. | Classification Criteria | Mean Rank |
|---------|-------------------------|-----------|
| 1       | RMSE                    | 8.69      |
| 2       | Gamma                   | 8.26      |
| 3       | FR                      | 7.01      |
| 5       | MP_SUM                  | 4.64      |
| 6       | SNR                     | 4.43      |

STDP and the firing times (spike train) and membrane potential were extracted from the output neuron. The coincidence factor and RMSE was calculated between the test/validation sample and all the training samples. The label of the closest training sample was assigned as the label to the test/validation sample and classification accuracy was recorded as shown in Table 2.

We also performed classification using rate coding mechanisms by computing following properties from the output neurons: firing rate (FR), sum of voltages (MP_SUM), signal-to-noise ratio computed using membrane potential (SNR). These features were used in different combinations to support vector machine and decision tree classifier. Top accuracies have been recorded as shown in Table 1.

Further, to test and rank all the implemented models we performed the Friedman test as shown in Table 3. After analysing the mean rankings, the model with RMSE as classification criterion achieved mean rank value of 8.69 which was the highest amongst all models followed by the model implemented with gamma factor. We could thereby infer that temporal coding mechanism carry more information relatively.

For post-hoc analysis we used the Wilcoxon signed-rank test which is a non-parametric statistical hypothesis test used to compare two related samples. Here, we compared the accuracies of gamma and RMSE factors with all the other criteria. We discovered that in all 70 runs, the score of gamma factor was higher than FR, MP_SUM and SNR. The performance of gamma and RMSE factors had 32 ties, 6 negative rankings (RMSE<gamma) and 32 positive rankings (RMSE>gamma). When analysing the performance of RMSE with other models, it was all win situation for RMSE. The conclusion after this analysis proves that temporal coding outperforms rate-coding mechanisms. Comparative analysis with traditional ML implies that SNN are capable of processing spatio-temporal data with acceptable performance.

Thus, the overall accuracy experimentally recorded in Table 1 and Table 2 along with the statistical results, concludes that the temporal dynamics of the neurons captured in its firing activity and membrane potential signal producing rhythmic patterns play an important role in the classification task.

The spike trains and the membrane potential thus carry useful information for discriminating samples from different groups. As we observed in the spike train analysis, gamma and RMSE criteria, this initial SNN architecture performs accurately when the classification criterion included temporal dimension of the spiking neuron for distinguishing violin and piano signals.

VII. CONCLUSION AND FUTURE WORK

The aim of this study was achieving higher performances and gaining insights into the inner workings of spiking neurons. The understanding of rhythmic nature of these neurons through analysing neural activity gave insights into the behaviour of two different musical instruments. In summary, the proposed deep SNN with Gamma and RMSE as the classification criteria proved to be successful in distinguishing two musical instruments: piano and violin. Finally, the attributes derived from voltage of the output neuron proved our that timely neuronal membrane potential is equally important along with event-based spike data. Hence, voltage information is important for recognition and classification tasks.

The findings of this study are a) highlight the importance of temporal-based spiking features for time-series classification, b) the power of SNN with less number of neurons and few training samples achieving higher accuracy, c) the application and relevance of coincidence factor and RMSE computed from spike timings and neuronal membrane potential respectively to estimate the category of the input d) demonstrate structurally optimised networks for producing distinguishable spiking activity using DE, a derivative free optimisation.
strategy, e) demonstrate the crucial role of analysing neural firing patterns which can potentially contribute in feature binding process, and f) open potential approaches to move forward in the area music recognition using SNN.

The future work of this research includes a) SNN model is yet to be tested on a benchmark dataset, b) the architecture is modeled on very few samples; irrespective of the fact that model performance was excellent, it is crucial to test it on large volume as well, c) evaluate the proposed deep SNN on polyphonic music signals, d) hybrid computation of weighted Gamma and RMSE as classification criteria. In addition, future work may be required to introduce other elements of tonality (relationships between notes, chords and keys) apart from pitch when the task is to model human experience of music at a neuronal level.

APPENDIX

This section describes the algorithmic steps involved in the implementation of this research. The other part explains the optimisation process to find optimal SNN parameters.

A. ALGORITHM SETUP

Z-Score Normalisation is used as part of the pre-processing method. No other pre-filtering techniques such as representing recording intervals and window size, or applying noise reduction techniques, were used. The aim is to test the capability of spiking neurons to distinguish musical instruments without such data cleansing steps, as would be usual with other techniques.

The following pseudo code explains the procedure used for the network setup. It represents the algorithmic step for the calculation of neuron’s membrane potential, propagation of spikes, and the computation of synaptic weights. The layout of network implementation can be described in following phases:

- Initialization. The data is divided into training set 80% and the testing set (20%). The number of spiking neurons in the hidden layer is decided based on the optimisation process described below and the output layer has one LIF neuron. Synaptic weights are randomly initialized and in the training process their range is optimized using Differential Evolution. k defines the number of split for cross-validation.

```
Initialization:
1: data = read(data)
2: initializeNetwork()
3: train, test= split(data)
4: k= 1,2...n
```

- Training. 80% of the training samples are further divided into trainData and validationData. Since we have used a small number of training samples (10), leave-one-out cross validation technique is used. For each validation sample, accuracy of the model is evaluated. During the SNN training spikes are propagated throughout, membrane potential is computed and synaptic weights are updated using the STDP rule. At this step, the training and testing sets are created.

```
Train:
5: validation_accuracy = list()
6: for i in k:
   trainData, validationData = cv_spilt(I, k, train)
   model = fit(trainData)
   accuracy_train = evaluate(model, validationData)
   validation_accuracy.append(accuracy_train)
end for
7: model = fit(train)
8: accuracy_test = evaluate(model, test)
```

- Propagation. The raw music signals are first converted into 2D binary data (spikes) through an encoding process to be described below. Post encoding stage, for each training set these pre-synaptic spikes are then propagated into the network of spiking neurons. Synaptic weights are updated when the neuron’s membrane potential crosses the threshold.

```
Propagate:
1: for each neuron
   computeMembranePotential()
   detectFiredNeurons()
end for
2: for each firedNeuron:
   propagateSpike()
end for
3: for each timestep:
   update_status_inputnodes()
   call Propagate
   update_Weights()
end for
```

B. OPTIMISATION PROCESS

DE algorithm is employed for optimizing various parameters in the encoding algorithm, LIF neuron model, STDP learning rule, number of neurons in the hidden layer, and range of synaptic weights. This approach analyses various parameters in the given search space and provides the best possible solution (the best set of parameters) through the processes of recombination, mutation and selection with an objective of maximising the overall classification accuracy.

In analogy with natural selection, DE generates child population by recombining the randomly chosen parent vectors from the initial population. After the recombination of parents, each of their children are mutated by adding a random deviation i.e. Gaussian white noise. All the children are evaluated on the fitness function and DE selects the best children to be the parents of next generation. These steps of recombination, mutation and selection continue in an iterative manner until convergence is met [terminating condition].
Inputs: Fitness function, lower bound ($l_b$) and upper bound ($u_b$) of each parameter (decision variables), the population size ($N_p$), termination criteria ($T$), the scaling factor ($0 < F \leq 2$) to be used in mutation and cross-over probability ($p_c = 0.1$).

1. Initialize a random population ($P$)
2. Evaluate fitness ($f$) of ($P$)
   for $i = 1$ to $T$
   for $i = 1$ to $N_p$
      Generate the donor vector ($V_i$) using mutation
      Perform crossover to generate offspring ($U_i$)
   end for
   for $i = 1$ to $N_p$
      Bound ($U_i$)
      Evaluate the fitness ($f(U_i)$) of ($U_i$)
      Perform selection using ($f(U_i)$) and ($f_i$) to update ($P$)
   end for
end for

From the various existing nature-inspired optimization techniques under evolutionary computation, we have implemented Differential Evolution (DE) [46]. DE is a stochastic, population-based optimization technique based on finding differences between candidate solutions in a population of solutions to guide the direction and length of search steps. Each solution is called as agent. Each agent undergoes mutation followed by the recombination operation. Target vector is the solution undergoing evolution [initial parameters], target vector is used in mutation to generate the donor vector, and finally the donor vector undergoes recombination to obtain the trial vector. The selection operation chooses the best solutions only after generating all the trial Vectors. Best solutions are chosen between the target and trial vectors as the next generation. DE strategies for optimizing SNN parameters have been previously shown to improve accuracy on classification and prediction problems [47]. The pseudo-code for the DE algorithm is explained in the block above:

We have used DE optimization during the cross-validation process to best the optimal SNN parameters. For the music recognition problem, the total number of evaluations is shown below:

Number of evaluations for optimizing various parameters in a 2-layer SNN:
- 7 SNN parameters: Threshold and refractory time of LIF neuron model, maximum potentiation ($A_+$) and depression ($A_-$) in STDP learning rule, the number of spiking neurons in the hidden layer, the range of synaptic weights (minimum and maximum value)
- Differential evolution: 70 candidate solutions (number of parameters *10), 100 iterations
- Every candidate solution: 30 models
- Total number of evaluations = 70 * 100 * 30 = 210,000

In BSA encoding which is used to transform the inputs into spikes, the number of filters, the size of filters and the threshold value were optimized using Differential Evolution algorithm. The optimisation is performed for each channel every sample. Since we have 20 channels in the input, every sample will have $19 \times 3 = 57$ parameters. Every channel will have the BSA threshold value, the size and the cut-off frequency of the FIR filter (threshold). The range of values for these parameter are: $th = [0.0, 2.0]$, $FIR size = [3, 50]$, $FIR cut-off = [0.0, 0.5]$.

REFERENCES

[1] R. Erickson, Sound Structure in Music. Berkeley, CA, USA: Univ. California Press, 1975.

[2] W. A. Sethares, Tuning, Timbre, Spectrum, Scale, 2nd ed. London, U.K.: Springer, 2005.
[3] B. Moore, *Psychoacoustics*. New York, NY, USA: Springer, 2007, pp. 459–501.

[4] T. Kitahara, M. Goto, K. Komatani, T. Ogata, and H. Okuno, “Instru-
ment identification in polyphonic music: Feature weighting with mixed
sounds, pitch-dependent timbre modeling, and use of musical con-
text,” in Proc. 6th Int. Conf. Music Inf. Retr. (ISMIR), pp. 558–563, 09
2005.

[5] F. Fuhrmann, “Automatic musical instrument recognition from polyphonic
music audio signals,” Ph.D. dissertation, Universitat Pompeu Fabra,
Barcelona, Spain, 2012.

[6] S. Aydın, Ç. Güdücü, F. Kutluk, A. Öniz, and M. Özgören, “The impact
of musical experience on neural sound encoding performance,” *Neurosci.
Lett.*, vol. 694, pp. 124–128, Feb. 2019.

[7] M. K. Weinberg and D. Joseph, “If you’re happy and you know it: Music
engagement and subjective wellbeing,” *Psychol. Music*, vol. 45, no. 2,
pp. 257–267, Mar. 2017.

[8] T. P. Theorell, A.-K. Lennartsson, M. A. Mosing, and F. Ullén, “Musical
activity and emotional competence—A twin study,” *Frontiers Psychol.*, vol. 5, p. 774, Jul. 2014.

[9] A. Maratos, C. Gold, X. Wang, and M. Crawford, “Music therapy for
depression,” *Cochrane Database Systematic Rev.*, to be published.

[10] K. Blum, T. Simpatico, M. Febo, C. Rodríguez, K. Dushaj, J. G. A. Barbedo and G. Tzanetakis, “Musical instrument classification with convolutional neural networks, source separation, and transfer learn-
ing,” in Proc. 19th Int. Soc. Music Inf. Retr. Conf., Paris, France, Sep. 2018, pp. 577–584.

[11] R. Jagiello, U. Pomper, M. Yoneya, S. Zhao, and M. Chait, “Rapid brain responses to familiar vs. unfamiliar music—An EEG and pupillometry study,” *Sci. Rep.*, vol. 9, no. 1, pp. 1–13, 2019.

[12] W. Maas, “Networks of spiking neurons: The third generation of neu-
network models,” *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, 1997.

[13] Z. Doborje, M. Doborje, T. Taylor, N. Kasabov, G. Y. Wang, R. Siegert, and A. Sumich, “Spiking neural network modelling approach reveals how mindfulness training rewire the brain,” *Sci. Rep.*, vol. 9, no. 1, pp. 1–15, Dec. 2019.

[14] D. Shah, G. Y. Wang, M. Doborje, Z. Doborje, and N. Kasabov, “Deep learning of eeg data in the neucube brain-inspired spiking neural
network architecture for a better understanding of depression,” in *Proc.
Int. Conf. Neural Inf. Process.*, Cham, Switzerland: Springer, 2019, pp. 195–206.

[15] N. Caporale and Y. Dan, “Spike timing-dependent plasticity: A Hebbian learning rule,” *Annu. Rev. Neurosci.*, vol. 31, pp. 25–46, Jul. 2008.

[16] W. Gerstner, “Integrate-and-fire neurons and networks,” in *The Handbook of Brain Theory and Neural Networks*, vol. 2, 2002, pp. 577–581.

[17] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge, UK, Cambridge Univ. Press, 2014.

[18] D. O. Hebb, *The Organization of Behavior: A Neuropsychological Theory*. Hoboken, NJ, USA: Wiley, 1949.

[19] R. Jolivet, A. Rauch, H.-R. Lüscher, and W. Gerstner, “Predicting spike timing of neocortical pyramidal neurons by simple threshold models,” *J. Comput. Neurosci.*, vol. 21, no. 1, pp. 35–49, Aug. 2006.

[20] W. M. Kistler, W. Gerstner, and J. L. van Hemmen, “Reduction of the hodgkin-Huxley equations to a single-variable threshold model,” *Neural Comp.*, vol. 9, no. 5, pp. 1015–1045, Jul. 1997.

[21] B. Schrauwen and J. Van Campenhout, “BSA, a fast and accurate spike train encoding scheme,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 4, Jul. 2004, pp. 2825–2830.

[22] 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.

[23] G. R. Home, “Free conservative Christian music,” *Tech. Rep.*, 2016.

[24] B. Kostek and R. Królkowski, “Application of artificial neural networks to the recognition of musical sounds,” *Arch. Acoust.*, vol. 22, no. 1, pp. 27–50, 2014.

[25] M. Pak and S. Kim, “A review of deep learning in image recognition,” in *Proc. 4th Int. Conf. Comput. Appl. Inf. Process. Technol. (CAIPT)*, Apr. 2017, pp. 1–3.

[26] J. Schnidhuber, “Deep learning in neural networks: An overview,” *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[27] Y. Han, J. Kim, and K. Lee, “Deep convolutional neural networks for predominant instrument recognition in polyphonic music,” *IEEE/ACM Trans. Audio Speech*, vol. 25, no. 1, pp. 208–221, Jan. 2017.

[28] S. J. Gómez, J. Arellán, and E. Cano. “Jazz solo instrument classification with convolutional neural networks, source separation, and transfer learn-
ing,” in Proc. 19th Int. Soc. Music Inf. Retr. Conf., Paris, France, Sep. 2018, pp. 577–584.

[29] S. Schliebs and N. Kasabov, “Evolving spiking neural networks: A sur-
vey,” in *Evolving Syst.* , vol. 4, p. 12, Jun. 2013.
DHVANI SHAH was born in India, in 1991. She received the Bachelor of Engineering (B.E.) degree in information technology (IT) from the University of Mumbai, India, in 2013, and the Master of Engineering (M.E.) degree (Hons.) in information technology (IT) from the University of Mumbai, India, in 2015. She is currently pursuing the Ph.D. degree with the Auckland University of Technology, New Zealand. Her study focuses on the development of new methodology for deep learning and deep knowledge representation of brain data, such as EEG and other time-based data like music signals. Her research interests include artificial intelligence, spiking neural networks, and brain data processing.

AJIT NARAYANAN received the B.Sc. degree (Hons.) in communication science and linguistics from the University of Aston, Birmingham, U.K., in 1973, and the Ph.D. degree in philosophy from the University of Exeter, Philosophy, Exeter, U.K., in 1976. He is currently a Professor with the School of Engineering, Computer Science and Mathematics, Auckland University of Technology. Before coming to New Zealand, in 2007, he was a Lecturer, a Professor, and the Dean of Universities in U.K. He has published over 100 articles in these areas. He has reviewed various journals/conferences in the field of AI and applications of AI in the field of medical. His research interests artificial intelligence, nature inspired computing, machine learning, computational statistics, and machine ethics.

Josafath Israel Espinosa-Ramos received the M.Sc. degree in cybernetics from La Salle University, Mexico, and the Ph.D. degree in computer science from the Centre for Computing Research, National Polytechnic Institute, Mexico. He was a Research Fellow with the Auckland University of Technology, New Zealand. His current research interests include modeling multisensory and multivariate streaming data and analyzing the spatial and temporal relationships among the variables that describe the dynamics of a sensor networks, computational neuroscience, evolutionary algorithms, and machine learning.