Terrain Classification with a Reservoir-Based Network of Spiking Neurons

Xinyun Zou[*], Tiffany Hwu[†‡], Jeffrey Krichmar[†*], and Emre Neftci[†*]
[*]*Department of Computer Science*, *University of California, Irvine*, Irvine, California, USA, 92697
[†]*Department of Cognitive Sciences*, *University of California, Irvine*, Irvine, California, USA, 92697
[‡]*HRL Laboratories, LLC*, Malibu, California, USA, 90265
Emails: xinyunz5@uci.edu, tjhwu@hrl.com, jkrichma@uci.edu, eneftci@uci.edu

***Abstract*—Terrain classification is important for outdoor path planning, mapping, and navigation. We developed a reservoir-based spiking neural network (r-SNN) to classify three terrain types (i.e. grass, dirt, and road) in a botanical garden. It included a recurrent layer and a supervised layer. The input spike trains to the recurrent layer were generated from linear accelerometer and gyroscope sensor signals as well as camera frames from an Android smartphone that controlled a ground robot. Compared to a Support Vector Machine (SVM) model and a 3-layer (3L) logistic regression model, our r-SNN method generated better prediction accuracy without reliance on a time window of data. Using both images and sensors as input, the test accuracy of the r-SNN was over 95%, which was significantly better than the SVM and the 3L logistic regression. Because the r-SNN is compatible with neuromorphic hardware, our proposed method could be part of a biologically-inspired power-efficient autonomous robot navigation system.**

I. Introduction

Outdoor robots face many dynamic challenges that are uncommon in indoor scenarios. In particular, uneven terrain and a wide variety of surfaces found outdoors can lead to unpredictability. Different terrain types have an effect on robot movement and power usage. For any outdoor autonomous navigation system, the robot should have long-term path planning strategies that consider trade-offs for traversing smooth surfaces, which may result in longer routes, versus direct routes that traverse over rough terrain, which may take more energy [1]. Moreover, field robots need to operate over long periods of time far from power sources. In these cases, accurate terrain classification may be beneficial for navigation.

Neuromorphic architectures have potential for controlling outdoor robotics under tight power constraints. Unlike the traditional Von Neumann architecture, a neuromorphic architecture consumes less power due to massive parallelism and event-driven processing [2], [3]. Spiking neural networks (SNN) can take advantage of neuromorphic hardware, because each neuron computes its state independently, making the SNN parallel, and spikes are asynchronous events.

Navigation requires the effective use of a map. SLAM algorithms [4] and GPS can provide solutions for navigation [5]. However, these maps do not include terrain information, which is critical for planning trajectories. Therefore, accurate terrain classification can be an important addition to generate cost maps and help with real-time localization [6]–[10].

To address these challenges, this paper introduces a reservoir-based spiking neural network (r-SNN) for terrain classification, which could be further integrated with other spiking navigation strategies to create a neuromorphic system for outdoor autonomous navigation.

II. Methods

A. Android Smartphone Solution

1) Android-Based Robotics Platform: Experiments were conducted using an Android-Based Robotics (ABR) Platform (see Figure 1). The GPS, accelerometer, gyroscope and visual information were directly obtained from an Android smartphone. A motor controller and IOIO-OTG microcontroller were mounted on the back of the platform. Communication between the phone and the robot platform was achieved through a Bluetooth connection with the IOIO-OTG. For robot specifications, see [11]. The testing environment was a 19-acre botanical garden which contained different terrain types (i.e., grass, dirt, and road), different inclinations, and different obstacles (e.g., trees, benches, pedestrians, etc.).

2) Terrain Data Collection: The ABR robot was programmed to run at a constant speed over grass, dirt, and road terrains, labeled as 0, 1, 2 respectively, in the botanical garden. The data collection process was conducted under different lighting conditions during the daytime for 42 trials. Each trial lasted between 1 and 5 minutes. The 3-dimensional gyroscope and linear accelerometer data were collected at 100 Hz via the smartphone (see Figure 2a), and camera frames were captured at 20 Hz with a resolution of 176×144 pixels (see Figure 2b).

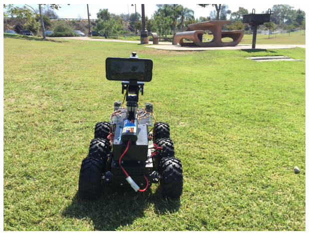

Fig. 1. A six-wheel Android-based ground robot (ABR) used for terrain classification experiments.

978-1-7281-3320-1/20/$31.00 ©2020 IEEE

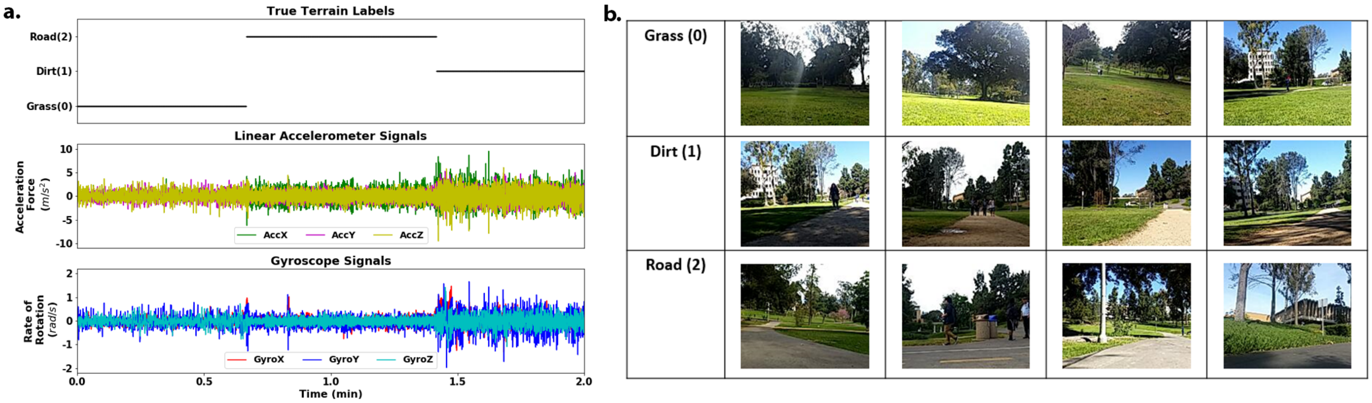

Fig. 2. a) A sample trial with the original 3D linear accelerometer and gyroscope signals. b) Sample camera frames from the smartphone during data collection, with a resolution of 176×144 pixels. Each frame was cropped to keep only the bottom-center 5×5 pixels as terrain visual information.

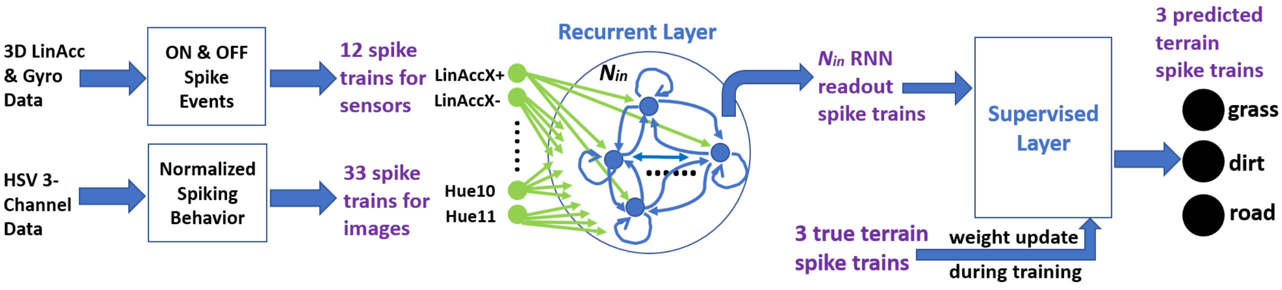
3D LinAcc & Gyro Data
ON & OFF Spike Events
12 spike trains for sensors
LinAccX+
LinAccX-
HSV 3-Channel Data
Normalized Spiking Behavior
33 spike trains for images
Hue10
Hue11
Recurrent Layer
Nin
Nin RNN readout spike trains
Supervised Layer
3 true terrain spike trains
weight update during training
3 predicted terrain spike trains
grass
dirt
road

Fig. 3. The terrain classification process with the r-SNN method.

B. Reservoir-based Spiking Neural Network

Since our terrain classification algorithm might include both image and sensor input data, sequence memory and feature selection for both data types could be challenging for a feed-forward network. However, a recurrent neural network (RNN), where connections between internal neurons form a directed cycle, could use its internal state as the memory to process arbitrary sequences of inputs. RNNs have been used for a variety of applications, such as motion prediction, health monitoring, speech recognition, and time series forecasting [12]–[15]. This recurrence can be tractably harnessed using a reservoir-based approach, such as Liquid State Machines (LSM) [16]. In the LSM, the recurrent weights in the RNN are randomly generated and only the RNN readout is trained.

We developed a reservoir-based SNN (abbreviated to "r-SNN") method for terrain classification (see Figure 3 for the flow diagram). The readout from the recurrent layer (referred to here as RNN) was trained using a surrogate gradient approach that can learn using precise spike times in the LSM [17]. The ability of the r-SNN to classify terrains is compared with two conventional approaches, the Support Vector Machine (SVM) and the 3-layer (3L) logistic regression.

1) Spiking Neuron Model: The spiking neuron model for the recurrent and supervised layers consisted of leaky integrate-and-fire (LIF) neurons with current-based synaptic input. For each postsynaptic neuron i at each time step t, if it was not within the refractory period, the postsynaptic membrane potential (U_i) was updated via the differential equation

$$\frac{dU_i}{dt} = \frac{U^{rest} - U_i}{\tau^{mem}} + I_i^{syn}(t), \tag{1}$$

where U^{rest} was the resting membrane potential, τ^{mem} was the membrane time constant, and $I_i^{syn}(t)$ was the synaptic input current. $I_i^{syn}(t)$ jumped by summation of the weight w_{ij} upon spike arrival from each presynaptic neuron j (i.e., when $S_j(t) = 1$), with the equation shown below

$$\frac{d}{dt} I_i^{syn}(t) = -\frac{I_i^{syn}(t)}{\tau^{syn}} + \sum_{j \in pre} w_{ij} S_j(t). \tag{2}$$

When U_i reached the threshold θ^{mem} and the neuron i was not in the refractory period, a spike was triggered (i.e., $S_i(t) = 1$). The neuron then remained refractory for n^{ref} time steps.

2) Supervised Learning Rule: Inspired by SuperSpike [18], during the supervised training process in which weight adaptation was requested (see Section II-E), the synaptic weight w_{ij} was updated at each time step according to a nonlinear Hebbian rule with individual presynaptic traces ϵ_j,

$$\Delta w_{ij} = \eta \cdot [\epsilon_j \otimes (\hat{S}_i - \sigma(U_i))] \cdot \sigma(U_i) \cdot (1 - \sigma(U_i)), \tag{3}$$

where η was the learning rate, $\hat{S}_i$ was the target postsynaptic spiking behavior, and ϵ was a linear filter on the presynaptic spike activities. The portion $(\hat{S}_i - \sigma(U_i))$ represented the error signal. The presynaptic traces ϵ evolved according to

$$\frac{d\epsilon_j}{dt} = -\frac{\epsilon_j}{\tau^{syn}} + S_j(t). \quad (4)$$

With a small synaptic time constant τ^{syn}, this first-order filter was sufficient to evaluate the temporal convolution with the error signal in the expression of the presynaptic traces.

C. Spike Generation of Input Data for the Reservoir

To convert gyroscope and linear accelerometer sensor signals from the smartphone into spike trains, we used the same spike train encoding as in the the Dynamic Vision Sensor (DVS) [19]. Six pairs of "plus" and "minus" spike trains for 3D signals of both sensors were converted into ON and OFF events, resulting in a total of 12 neurons for the sensor input data. For each axis of each sensor every time step, if the increase or decrease amount in the signal was above a threshold (i.e., 2 for the linear accelerometer and 0.5 for the gyroscope), an ON or OFF spike was generated, respectively.

Image data were converted from RGB (red, green, blue) to HSV (hue, saturation, value) and normalized between 0 and 1 for each channel. Each frame was cropped to keep only the bottom-center 5×5 pixels, which was enough to show the current terrain visual information without interference from distractors in the scene (e.g., other terrain types, trees, benches, pedestrians, or buildings). Each HSV channel was averaged across all 25 pixels in the image. There were 11 neurons for each HSV channel (i.e., a total of 33 neurons). Each neuron's activity was based on a Gaussian tuning curve. The means of the Gaussians were spread evenly 0 to 1 with $\sigma = 0.5$. The maximum activity for each tuning curve was $\alpha = 1/(\sigma\sqrt{2\pi})$. If activity was above 0.4α, the neuron spiked.

The frequencies of the sensor signals and camera frames were different (i.e., 100 Hz and 20 Hz respectively). Therefore when both sensor signals and image frame were fed into the recurrent layer, the same image frame would be repeated for each time step until the next frame was collected.

D. Recurrent Layer for Terrain Feature Extraction

The gyroscope signals, the linear accelerometer signals, and the cropped screenshots were encoded directly into spikes, as described above. These input spikes were fed into the recurrent layer. The sensor and image neurons were fully connected with the recurrent neurons. The recurrent neurons were also fully connected with one another. For this recurrent layer, the synaptic input current was a summation of both input weights and recurrent weights when spikes were received. The readout spikes from the recurrent layer were further fed into the supervised layer for terrain classification (see Section II-E).

There were $N_{in} = 70$ recurrent neurons, which received N_{ext} input spike trains from the sensors and/or images (i.e., $N_{ext} = 12$, 33, 45 respectively). Both input and recurrent weights were randomly drawn from a Gaussian distribution with zero mean. The standard deviation was $0.5/N_{ext}$ for input weights or $0.05/N_{in}$ for the recurrent weights, which was small enough to prevent bias on certain connections while assuring randomness. Therefore recurrent neurons were both excitatory and inhibitory. In Equation 1, U^{rest} and the initial U_i were both 0, whereas τ^{mem} was tuned to 66.7. In Equation 2, the initial $I_i^{syn}(t)$ was 0, whereas τ^{syn} was tuned to 1. Furthermore, θ^{mem} was $(1 - \gamma)$, with γ as the threshold Gaussian noise with mean at 0 and a standard deviation of 0.1. n^{ref} was zero, meaning there was no refractory period.

E. Supervised Layer for Terrain Classification

The supervised layer took as input the 70 readout spike trains from the recurrent neurons and generated spike activities for the three terrain prediction output neurons that represented grass, dirt, and road. The output weights were updated every time step (see Equation 3) for 100 training epochs and remained constant for testing. During the training process, the target postsynaptic spiking behavior was obtained from three spikes trains that represented actual terrain information at each time step. For the supervised layer, the synaptic input current evolved with summation of output weights upon spike arrival. The postsynaptic neurons were the three terrain prediction neurons. A terrain class was predicted by the output neuron with the highest activity at that time step.

Before the first training epoch, the output weights were all initialized as $0.001/N_{in} = 1\mathrm{e}{-5}$ so that all the readout spikes could excite the three terrain output neurons. In Equation 1, U^{rest} was 0 and τ^{mem} was tuned to 100, but U_i was initialized as -0.5 for each epoch. In Equations 2 and 4, τ^{syn} was tuned to 10, whereas $I_i^{syn}(t = 0)$ was initialized as zero at the beginning of each training epoch and of testing. θ^{mem} and n^{ref} were the same as in Section II-D. In Equation 3, η was tuned to $9\mathrm{e}{-9}$. The sigmoid function was tuned to $\sigma(x) = 1/(1 + \exp[-3.44 \cdot (x - 0.975)])$.

III. Results

A. Terrain Prediction Results for the r-SNN

The r-SNN method achieved over 90% of testing accuracy in predicting different terrain types, with either linear accelerometer and gyroscope sensors, or image inputs, or both sensor and image inputs (see Table I). Figures 4 and 5 show results using both images and sensors for an 8-minute testing period. From the 70 recurrent neurons (see Figure 4), the supervised layer generated output spikes for terrain classification. After 100 training epochs, the test prediction consistently matched the true terrain with little delay or noise (see Figure 5).

B. Optimal Settings for Two Conventional Approaches

For the SVM model and the 3L logistic regression model, we split the original sensor and image signals into data chunks with a time window of 500 milliseconds. The optimal performance on the SVM model was achieved by using the SVC package in the Scikit-learn library with the RBF kernel [20]. The SVM applied the nine features: (1) number of sign changes, (2) number of traverses over mean, (3) standard deviation, (4) autocorrelation at lag k=1, (5) maximum, (6) minimum, (7) Euclidean norm, (8) mean, and (9) median. Its best test accuracy was achieved after 430 training epochs.

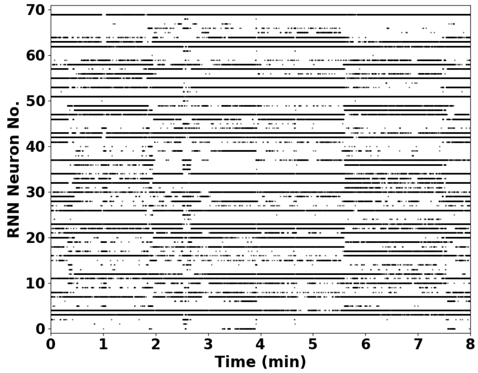

Fig. 4. Readout spikes from all the recurrent neurons using both image and sensor (the linear accelerometer and gyroscope) inputs. The horizontal axis labels partial testing period of 8 min, with sensor signals collected at 100 Hz and camera frames collected at 20 Hz. These 70 readout spike trains were further fed into the supervised testing part for terrain classification.

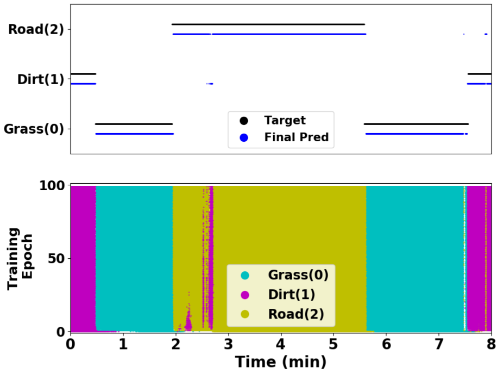

Fig. 5. Supervised layer output for both images and sensors (the linear accelerometer and gyroscope) for an 8-minute testing period. The upper subplot shows true terrain types and final test predictions using adapted output weights after 100 training epochs. The lower subplot shows the test prediction spikes using adapted output weights after each training epoch.

The optimal performance on the 3L logistic regression model required the following five features: (1) 20-percentile, (2) 50-percentile, (3) 80-percentile, (4) mean, (5) standard deviation. Its best test accuracy was achieved after 520 training epochs by using the mean squared error as the loss function, or after 820 training epochs with the cross-entropy loss function.

C. Model Performance Comparison

For comparison among three approaches on terrain classification, we applied the standard 80/20 rule for training and testing. The input data were generated from linear accelerometer and gyroscope sensor signals and/or cropped screenshots.

Table I shows the test error rates among three approaches under three input conditions. Using both image and sensor data instead of using one of them improved the accuracy and robustness of each model. The r-SNN method was more accurate than SVM and 3L logistic regression. The r-SNN may be the most efficient considering its usage of only 70 recurrent neurons, adaptation of only the output weights, and no need of splitting data into time chunks.

The r-SNN is compatible with low-power neuromorphic architectures, whose energy cost is often dominated by synaptic operations (SynOps), akin to multiply accumulate operations (MACs) in digital computers for artificial networks [21], [22]. For our entire training and test process with terrain classified every 500 milliseconds, the r-SNN would require roughly 10^9 SynOps on a neuromorphic hardware, which is equal to or smaller than the operations taken by the 3L logistic regression and the SVM (i.e., roughly $10^9 \sim 10^{10}$ MACs for each) on a standard computer. Based on the fact that a SynOp consumes many fold less energy than a MAC [21], [23], [24], the r-SNN is a promising approach to reduce power consumption.

TABLE I
TEST ERROR RATES ON THREE MODELS FOR TERRAIN CLASSIFICATION.

	r-SNN	**SVM**	**3L Logistic Regression**	
			mse	**xent**
Images only	**5.2%**	13.9%	11.5%	16.2%
Sensors only	**8.1%**	14.5%	13.7%	59.6%
Images + Sensors	**3.5%**	8.8%	10.2%	34.3%

IV. Conclusion

Unlike feed-forward networks, the recurrent layer processes both the sensor and image input data to extract abstract terrain features at each time step, with no need of remembering data chunks within a time window or carefully selecting feature components. The reservoir computing paradigm lowers the computational cost during supervised training, because only the output weights are plastic [16]. Moreover, having spiking neurons in the reservoir allows the model to be event-driven and highly parallel. Further performance gains can be achieved by implementing the present algorithm on neuromorphic hardware that utilizes spike-based strategies.

The r-SNN has several advantages for classification tasks such as discriminating terrains. First, it had the highest test prediction accuracy compared to the SVM and 3L logistic regression, regardless of whether images or sensors were the input (see Table I). Secondly, it had the lowest computational cost due to a small reservoir of spiking neurons and adaptation of only the output weights. Third, compared to the difficulty in selecting the terrain features and time window length for the two conventional approaches, the r-SNN reservoir can easily integrate the image and/or sensor data and generate an abstract representation of terrain features.

The trained r-SNN model is compatible with a ground robot for real-time terrain classification. The r-SNN can be used to augment a SLAM or GPS map with metadata pertaining to the cost of traversal. For example, the r-SNN can supplement a road following algorithm to signal when the robot veers off the road. The different terrains can be used as a cost function, based on terrain smoothness for path planning [5]. Finally, the r-SNN presented here can be used in to develop a complete neuromorphic robot navigation system capable of operating over long durations with minimal power consumption [1], [25].

References

[1] Tiffany Hwu, Jeffrey Krichmar, and Xinyun Zou. A complete neuromorphic solution to outdoor navigation and path planning. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4. IEEE, 2017.

[2] Carver Mead. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–1636, 1990.

[3] Giacomo Indiveri, Bernabé Linares-Barranco, Tara Julia Hamilton, André Van Schaik, Ralph Etienne-Cummings, Tobi Delbruck, Shih-Chii Liu, Piotr Dudek, Philipp Häfliger, Sylvie Renaud, et al. Neuromorphic silicon neuron circuits. *Frontiers in neuroscience*, 5:73, 2011.

[4] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[5] Tiffany Hwu, Alexander Y Wang, Nicolas Oros, and Jeffrey L Krichmar. Adaptive robot path planning using a spiking neuron algorithm with axonal delays. *IEEE Transactions on Cognitive and Developmental Systems*, 10(2):126–137, 2017.

[6] Joan S Weszka, Charles R Dyer, and Azriel Rosenfeld. A comparative study of texture measures for terrain classification. *IEEE transactions on Systems, Man, and Cybernetics*, SMC-6(4):269–285, 1976.

[7] Roberto Manduchi, Andres Castano, Ashit Talukder, and Larry Matthies. Obstacle detection and terrain classification for autonomous off-road navigation. *Autonomous robots*, 18(1):81–102, 2005.

[8] Jean-François Lalonde, Nicolas Vandapel, Daniel F Huber, and Martial Hebert. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of field robotics*, 23(10):839–861, 2006.

[9] Khairul Azmi Mahadhir, Shing Chiang Tan, Cheng Yee Low, Roman Dumitrescu, Adam Tan Mohd Amin, and Ahmed Jaffar. Terrain classification for track-driven agricultural robots. *Procedia Technology*, 15:775–782, 2014.

[10] Krzysztof Walas. Terrain classification and negotiation with a walking robot. *Journal of Intelligent & Robotic Systems*, 78(3-4):401–423, 2015.

[11] Nicolas Oros and Jeffrey L Krichmar. Smartphone based robotics: Powerful, flexible and inexpensive robots for hobbyists, educators, students and researchers. Technical Report 13-16, Center for Embedded Computer Systems, University of California, Irvine, Irvine, California, 2013.

[12] Hirak J Kashyap, Georgios Detorakis, Nikil Dutt, Jeffrey L Krichmar, and Emre Neftci. A recurrent neural network based model of predictive smooth pursuit eye movement in primates. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[13] Anup Das, Paruthi Pradhapan, Willemijn Groenendaal, Prathyusha Adiraju, Raj Thilak Rajan, Francky Catthoor, Siebren Schaafsma, Jeffrey L Krichmar, Nikil Dutt, and Chris Van Hoof. Unsupervised heart-rate estimation in wearables with liquid states and a probabilistic readout. *Neural Networks*, 99:134–147, 2018.

[14] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[15] Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. Recurrent neural networks for time series forecasting: Current status and future directions. *arXiv preprint arXiv:1909.00590*, 2019.

[16] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[17] EO Neftci, H Mostafa, and F Zenke. Surrogate gradient learning in spiking neural networks. *Signal Processing Magazine, IEEE*, Dec 2019. (accepted).

[18] Friedemann Zenke and Surya Ganguli. Superspike: Supervised learning in multilayer spiking neural networks. *Neural computation*, 30(6):1514–1541, 2018.

[19] Minhao Yang, Shih-Chii Liu, and Tobi Delbruck. A dynamic vision sensor with 1% temporal contrast sensitivity and in-pixel asynchronous delta modulator for event encoding. *IEEE Journal of Solid-State Circuits*, 50(9):2149–2160, 2015.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[21] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

[22] Emre O Neftci, Bruno U Pedroni, Siddharth Joshi, Maruan Al-Shedivat, and Gert Cauwenberghs. Stochastic synapses enable efficient brain-inspired learning machines. *Frontiers in neuroscience*, 10:241, 2016.

[23] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018.

[24] Emre O Neftci, Charles Augustine, Somnath Paul, and Georgios Detorakis. Event-driven random back-propagation: Enabling neuromorphic deep learning machines. *Frontiers in neuroscience*, 11:324, 2017.

[25] Raphaela Kreiser, Alpha Renner, Yulia Sandamirskaya, and Panin Pienroj. Pose estimation and map formation with spiking neural networks: Towards neuromorphic slam. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2159–2166. IEEE, 2018.