AD

Technical Report CCDCAC-BLTR-20016

Evaluation Framework for Input Layer Preprocessing in a Radial Basis Function Neural Network

Curtis W. Bradley

U.S. Army Combat Capabilities
Development Command Armaments Center
Benét Laboratories
Watervliet, NY 12189

18 May 2020

The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

The citation in this report of the names of commercial firms or commercially available products or services does not constitute official endorsement by or approval of the U.S. Government.

Destroy this report when no longer needed by any method that will prevent disclosure of its contents or reconstruction of the document. Do not return to the originator.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
05/18/2020	Technical Report	

4. TITLE AND SUBTITLE
Evaluation Framework for Input Layer Preprocessing in a Radial Basis Function Neural Network

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Curtis W. Bradley

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Combat Capabilities Development Command
Armaments Center, Benét Laboratories Directorate
1 Buffington Street
Watervliet, NY 12189

8. PERFORMING ORGANIZATION REPORT NUMBER
CCDCAC-BLTR-20016

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution A: Approved for public release; distribution is unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The need for immediate situational awareness updates in a military environment can be partially mitigated by employing machine learning (ML) at the edge of the network, where the warfighter operates. Technical challenges for edge computing, like limited power and data, require unique hardware and software implementations for viable solutions. Low power neuromorphic processors running radial basis function artificial neural networks (RBFNN) makes ML at the edge more practical but can introduce limitations in the data throughput. This power and data

15. SUBJECT TERMS
Low power neuromorphic processors, radial basis function artificial neural networks, RBFNN, RCE Learning, neural network preprocessing, spectrogram, scalogram

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Curtis W. Bradley
					19b. TELEPHONE NUMBER *(Include area code)*
U/L	SAR	SAR	SAR	12	518-266-4858

INSTRUCTIONS FOR COMPLETING SF 298

1. REPORT DATE. Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. REPORT TYPE. State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. DATE COVERED. Indicate the time during which the work was performed and the report was written, e.g., Jun 1997 - Jun 1998; 1-10 Jun 1996; May - Nov 1998; Nov 1998.

4. TITLE. Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. CONTRACT NUMBER. Enter all contract numbers as they appear in the report, e.g. F33315-86-C-5169.

5b. GRANT NUMBER. Enter all grant numbers as they appear in the report. e.g. AFOSR-82-1234.

5c. PROGRAM ELEMENT NUMBER. Enter all program element numbers as they appear in the report, e.g. 61101A.

5e. TASK NUMBER. Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. WORK UNIT NUMBER. Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. AUTHOR(S). Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES). Self-explanatory.

8. PERFORMING ORGANIZATION REPORT NUMBER. Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES). Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. SPONSOR/MONITOR'S ACRONYM(S). Enter, if available, e.g. BRL, ARDEC, NADC.

11. SPONSOR/MONITOR'S REPORT NUMBER(S). Enter report number as assigned by the sponsoring/ monitoring agency, if available, e.g. BRL-TR-829; -215.

12. DISTRIBUTION/AVAILABILITY STATEMENT. Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/FRD, PROPIN, ITAR, etc. Include copyright information.

13. SUPPLEMENTARY NOTES. Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. ABSTRACT. A brief (approximately 200 words) factual summary of the most significant information.

15. SUBJECT TERMS. Key words or phrases identifying major concepts in the report.

16. SECURITY CLASSIFICATION. Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. LIMITATION OF ABSTRACT. This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

ABSTRACT

The need for immediate situational awareness updates in a military environment can be partially mitigated by employing machine learning (ML) at the edge of the network, where the warfighter operates. Technical challenges for edge computing, like limited power and data, require unique hardware and software implementations for viable solutions. Low power neuromorphic processors running radial basis function artificial neural networks (RBFNN) makes ML at the edge more practical but can introduce limitations in the data throughput. This power and data limitation can be moderated using preprocessing of the input space to magnify the most pertinent data features. This paper presents a framework for evaluating different input space paradigms in a systematic manner. Using a representative small dataset for a pyroshock event, common in the military environment, several input preprocessing paradigms are evaluated.

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

The need for immediate situational awareness updates in a military environment can be partially mitigated by employing machine learning (ML) at the edge of the network, where the warfighter operates. Technical challenges for edge computing, like limited power and data, require unique hardware and software implementations for viable solutions [1]. Low power neuromorphic processors running radial basis function artificial neural networks (RBFNN) makes ML at the edge more practical but can introduce limitations in the data throughput [2, 3, 4, 5]. This power and data limitation can be moderated using pre-processing of the input space that can magnify the most pertinent features.

Architecture development for artificial neural networks (ANN) used for classification often assumes a raw input dataset to maximize the potential learning of the ANN [6, 7, 8]. This approach leads to an ever increasing need for network complexity from added neurons and layers. When employing these algorithms in an edge computing environment, this complexity and resulting data and power consumption can be prohibitive [9]. The RBFNN using a restricted Coulomb energy (RCE) learning method can classify input data in a low power environment by minimizing the number of active neurons [2, 5, 4].

To increase the practical implementation of an RBF for time or image based signals, they are often sent preprocessed data using fast Fourier transform (FFT), wavelets, or other data reduction methods [10, 2, 11, 12, 13, 14]. Time based signals generated from pyroshock events similar to those found in a mining or military environment tend to use frequency transforms like wavelets or the FFT [7]. Often the preprocessing algorithms are chosen through trial and error with the classification accuracy used as the performance metric [8]. The training process for an RCE involves adjusting neuron weights as well as sequentially activating additional neurons to recognize each class as the definition is refined by the training dataset [15, 16]. RCE neuron weights and activations depend on the order and composition of the input data training set [17].

While it is well known that an RBFNN using RCE learning can approximate a function smoothly, the number of committed neurons is not an in-line metric [15, 18, 5]. Generally, attention is focused on the RBFNN learning method [19, 20, 4, 21, 22] or on integration into a deeper network [14]. While preprocessing methods are often critical to the research, it is not viewed in the context of committed neurons [10, 2, 12, 11]. Considering the number of committed neurons relative to the preprocessing method for an RBFNN using RCE learning has been absent from the literature.

This paper proposes a framework for evaluating preprocessing methods to an RBFNN input layer based on the committed neuron count using a simulation of the NM500 neuromorphic chip [1]. The correlation coefficient between the number of committed neurons and the inference accuracy has a highly correlated p-value across the entire dataset of 1×10^{-7}. Using representative data for a pyroshock event common in the military environment, several input preprocessing methods were evaluated. By using the ANN structure in evaluating the preprocessing method, the potential computational cost and associated power may be mitigated. This research helps to establish a framework whereby preprocessing methods can be evaluated in real-time, which could serve as a precursor for a broader RBFNN design method.

The remainder of this paper is structured as follows. Section 2 details the data description and ANN structure. Section 3 describes the dataset and the manner in which it was used. Section 4 outlines the specifics of the cross-validation and RBFNN chip simulation. Finally, section 5 presents the results, while section 6 outlines the specific conclusions that can be drawn from

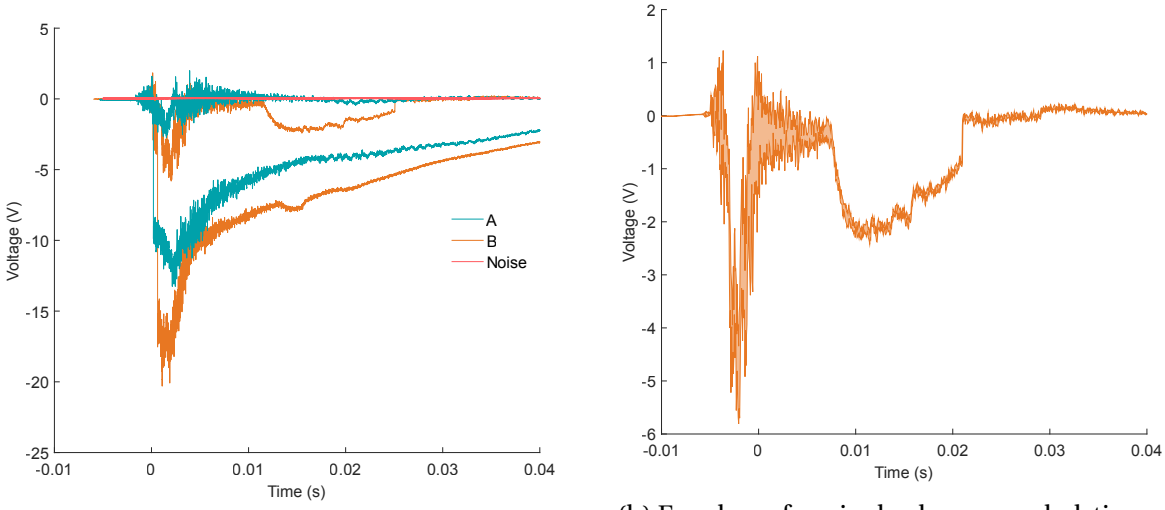
this study.

2 METHOD

The source data is from pyroshock testing on a military system. Given the typical high operational costs, the dataset is small. Furthermore, the dataset contains collection artifacts as is also common when dealing with military systems in an unpredictable operating environment. The utilization of the data and structure of the ANN must both be tailored to operate under these limitations. Processing this class of data has broad applicability in both military and commercial systems.

2.1 DATASET DESCRIPTION

Pyroshock events generate time based signals in various severe environments found in mining, military, and other systems. These events cause a decaying, oscillatory response due to high amplitude and frequency excitation [23]. There are two classes of labeled data with a third noise class as a counter-example to prevent over-fitting. Typical examples for each class are shown in Figure 2.1a. Traditional rule-based classification methods would be difficult with this amount of overlap between classes shown in Figure 2.1a.



(a) Typical high and low signals for each class

(b) Envelope for single downsampled time series signal for class "B"

Figure 2.1: Time series data

The noise data files were randomly generated based on the magnitude and frequency content prior to the pyroshock events. Half of the noise examples have a constant 60 Hz sinusoid component. The data rate of the dataset is much higher at 1 MHz than is typically feasibly for edge deployed applications. The data was downsampled at a 50:1 ratio and a phase shift of one sample to essentially split each data example into several of the same class and increase the data points for the limited dataset. The envelope defined by a single downsampled example of class "B" is shown in Figure 2.1b, demonstrating the variability introduced by downsampling, especially in the high frequency content portion early in the pyroshock event.

Each set of downsampled examples are treated as a group when placed in either the training or validation sets. All data is the voltage from a single sensor on the same test asset over the course of several days of testing. The allocation for a single training epoch is outlined in Table 2.1.

Table 2.1: Dataset example allocation breakdown for a single epoch

class	Original	50:1 Downsampled	Training	Validation
A	9	450	400	50
B	49	2450	2200	250
Noise	20	1000	900	100
Total	78	3900	3500	400

2.2 RBFNN WITH RCE LEARNING

An RBFNN is an artificial neural network with a fully connected input layer, partially connected hidden layer where each hidden layer neuron is connected to only one output layer neuron [24]. Output layer neurons are typically connected to several output hidden layer neurons, as is seen in Figure 2.2. The RBFNN considered in this work uses an RCE learning method to assign the

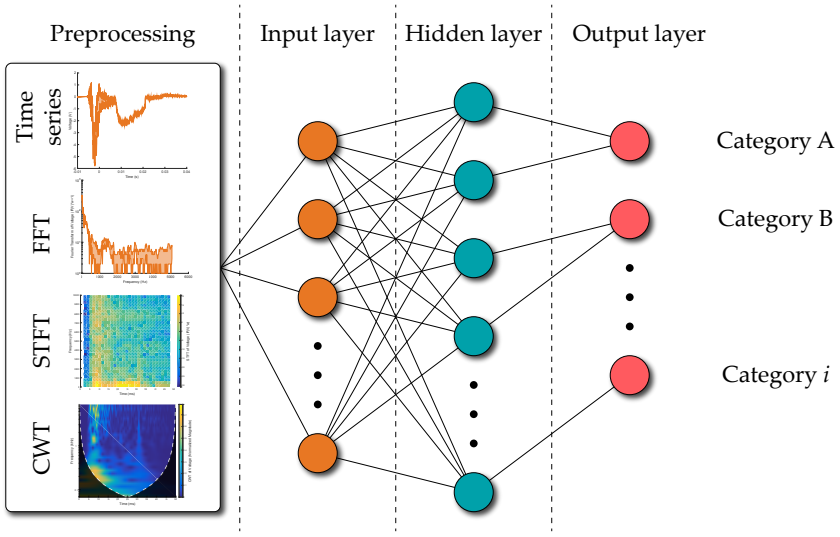


Figure 2.2: RBFNN diagram with representative preprocessing methods

center of each radial basis function referred to as a prototype because it is equal to the scaled input values of the first input vector defining a new area of influence for that class.

The other aspect of the RCE training process is to define the activation distance from a prototype referred to as the active influence field (AIF) [25, 26]. An RBFNN using RCE learning starts with one hidden layer neuron and only adds one neuron at a time during training. The

NM500 architecture has a fixed number of neurons, so when a neuron is added to the RBFNN it is merely committed to using that latest input as the prototype for distance measurements to subsequent inputs. Hidden layer radial basis function neurons are added if the input is outside all AIF radii for the labeled class. If this new neuron conflicts with the AIF radius of another class neuron, that other AIF radius is reduced to deconflict the output layer. The noise class of counter-examples also reduces AIF radii to prevent over generalization.

The RBFNN simulation for the NM500 allows for an input layer of at most 256 neurons for a single ANN context. While the NM500 can use both the L_1 or L_∞ distance for the distance calculation, this work uses L_1 for training and inference. The distance for a given input, i, from each committed neuron is,

$$d_i = \sum \left| w\mathbf{x} - \mathbf{p}_i \right|, \tag{2.1}$$

where w scales the input vector $\mathbf{x}$ and the prototype, $\mathbf{p}_i$, is the center of the radial basis function of the i^{th} hidden layer neuron. The scalar w is constant and specific to an RBFNN with a preprocessing method.

While there are many possible activation functions like PRelu and Gaussian this RBFNN implementation uses a Step function [27],

$$Activation(d_i) = \begin{cases} 1 & : d_i \leq AIF_i \\ 0 & : d_i > AIF_i \end{cases}. \tag{2.2}$$

The number of neurons that can be committed is limited to the physical number of on-board neuromorphic memory cells in the design. The absolute minimum is the number of labeled example classes input in training and the maximum the total number of examples input during training.

The RBFNN architecture using RCE learning creates a potentially functional inference model in real-time on an objective system without the need for high-power, high-bandwidth off-line training making the RBFNN attractive for edge computing applications. Intrinsic to this architecture is an input dependent increasing number of neurons during training. The number of neurons are determined by both the separation between input classes as well as the neighbor distance within a class. The dependence of neuron growth based on input vector distances is non-linear and order dependent, but still dependent and therefore insightful [27, 5].

3 NEURAL NETWORK PREPROCESSING

Often RBFNN architecture definitions focus on methods for AIF determination and activation functions, whereas this work focuses on the input preprocessing [19, 27, 15, 21, 4, 14]. While much attention has been paid to preprocessing methods, it is still worthwhile to consider preprocessing with respect to the number of committed neurons required to classify the input space [10, 2, 12, 14]. Preprocessing the input vector essentially changes the representation and thereby the features within that representation. The purpose is to have features unique to each class be separated by a larger distance while features within a class have a smaller separation.

3.1 TIME SERIES

The time series is raw input voltage with the minimum amount of preprocessing. The data was converted to fixed point to facilitate future hardware implementation. An additional scalar, w,

multiplies the signal as part of preprocessing to control the magnitude of the L_1 distance also related to hardware implementation. The envelope for every example in the dataset arranged by class is shown in Figure 3.1. The first 256 data points after a threshold trigger are used as

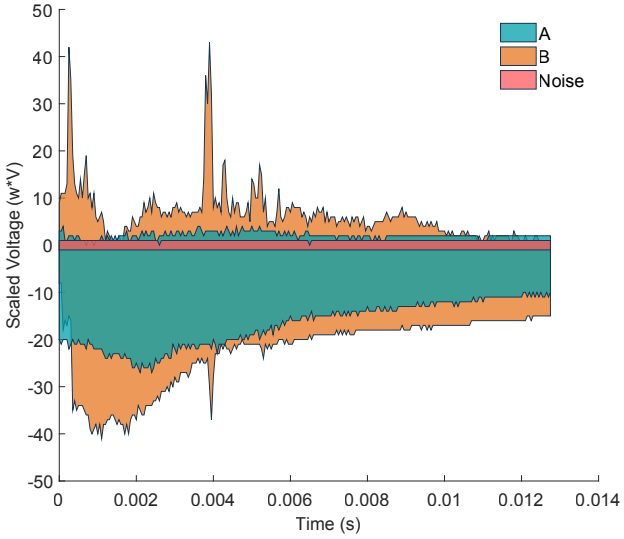


Figure 3.1: Envelope for each class across the entire dataset

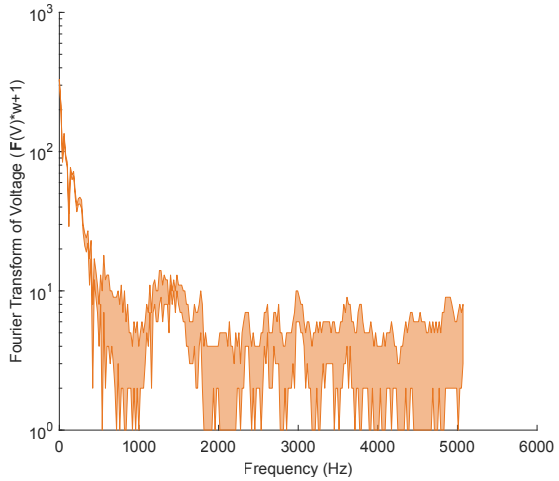
the input to the RBFNN. There is a significant amount of overlapping between the envelope of each class. It would be difficult to infer these classes using traditional methods directly based on these time series.

3.2 Fourier Transform

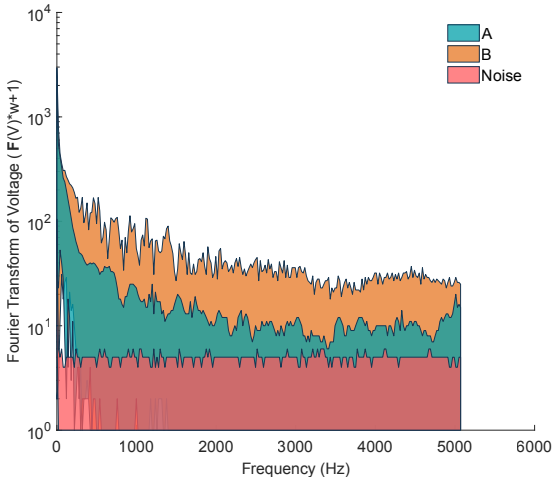
The FFT represents the input in the frequency domain [28]. This removes all time based information from input and highlights the various frequency components of the pyroshock event. The FFT for the downsampled example in Figure 2.1b is shown in Figure 3.2a. The FFT for a single example in Figure 3.2a shows discretization from the fixed point conversion. To better visually express the fixed point, FFT data in Figure 3.2 is offset by 1 and plotted on a logarithmic scale. While the figure is on the log scale, the RBFNN input is not. As with all the input preprocessing methods, the scalar w multiplies the input. The envelope for the FFT of every example in the dataset arranged by class is shown in Figure 3.2b. The FFT signals have significant overlap making traditional rules based classification difficult. The first 256 FFT points are used for the RBFNN input.

3.3 Spectrogram

The short-time Fourier transform (STFT) combines time and frequency features of the data to produce a spectrogram [29]. This makes any time-separated frequency based data features more prominent. The spectrogram for the downsampled example in Figure 2.1b is shown in Figure 3.3a. Only a subset of the logarithm of the spectrogram data was used for the RBFNN input at one time. The spectrogram input was a set of 31×8 contiguous points offset from the



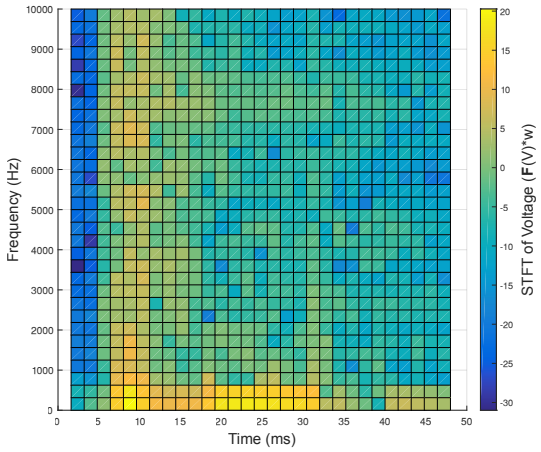
(a) FFT Envelope for a single downsampled example



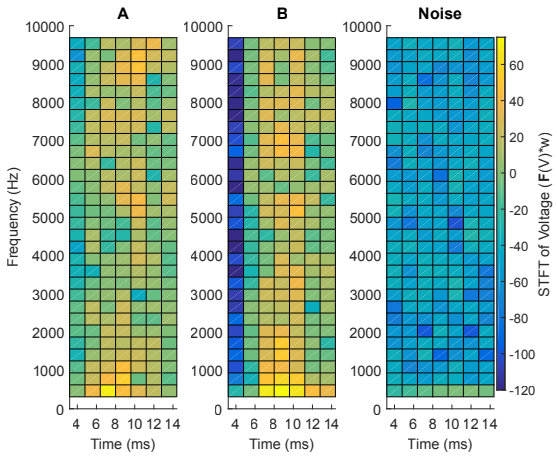
(b) FFT complete signal envelope

Figure 3.2: FFT data

origin and scaled by w. The subset of a typical spectrogram used for each class example is shown in Figure 3.3b. For the RBFNN input layer the spectrogram was arranged into a vector.



(a) Spectrogram for a single downsampled example



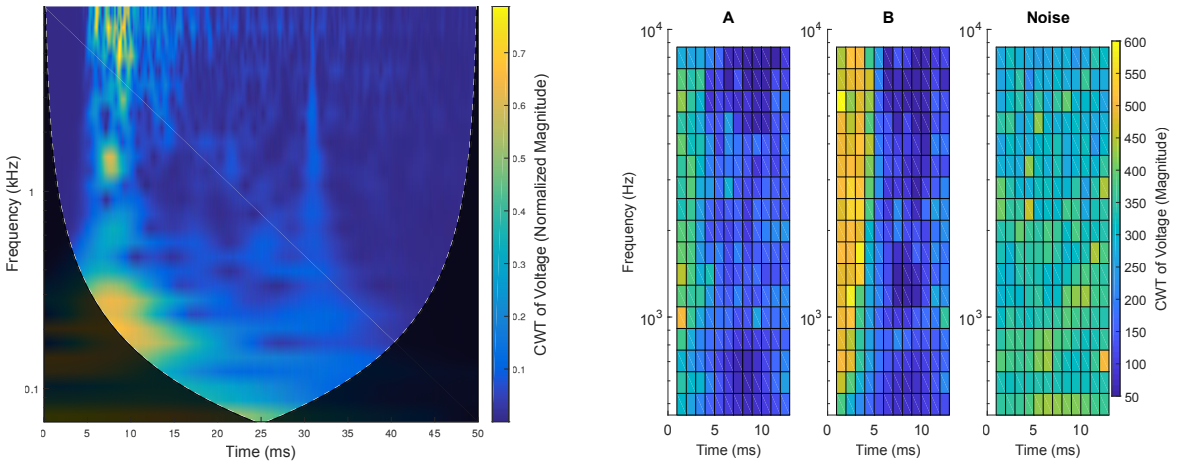
(b) Typical spectrogram subset for each class

Figure 3.3: Short-time Fourier transform data

3.4 SCALOGRAM

The continuous wavelet transform (CWT) in this work used the Morse wavelet with 4 voices per octave to produce a scalogram of the voltage signal. The scalogram for the downsampled example in Figure 2.1b is shown in Figure 3.4a. The CWT uses L_1 normalization by dilating the wavelet so all frequency amplitudes are normalized to the same value [30]. The CWT has a logarithmic frequency distribution. There are many methods to select points from the scalogram to map to the input layer. Input vectors in this work are uniformly distributed and offset from

the scalogram origin to produce an 18×13 set of points scaled by w. The subset of a typical scalogram used for each class example is shown in Figure 3.4b.



(a) Scalogram for a single downsampled signal

(b) Typical scalogram subset for each class

Figure 3.4: Continuous wavelet transform data

4 EVALUATION METHODOLOGY

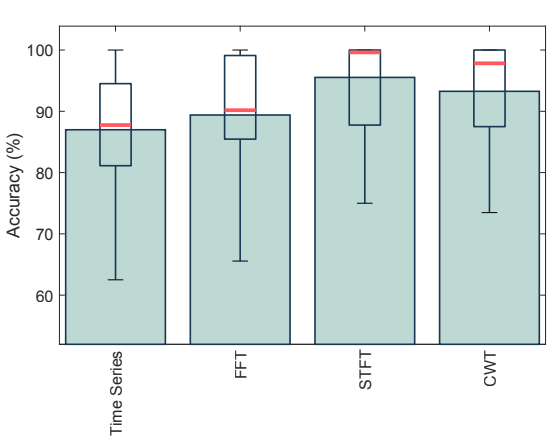
Training and validation was conducted using a simulation of the NM500 neuromorphic chip. Validation data encompassed at least 10% of the data in each class as previously shown in Table 2.1. RCE learning is implemented on the NM500 with a global initial setting for the maximum and minimum AIF radii that determines to what extent the RBFNN will generalize in inference. Each AIF is then reduced as needed during training. Scalar w was used to maximize the classification accuracy on a subset of the dataset for each preprocessing method. Once found, the scalar w was constant for a given preprocessing method. This allowed all preprocessing methods to use the same maximum initial neuron AIF.

A non-exhaustive Monte-Carlo cross-validation was used to compare the classification performance of each preprocessing method. The order of the Monte-Carlo training sets was the same for each epoch for every preprocessing method. Since relative performance between methods is of most interest, all four preprocessing methods used the same example ordering during training to allow a reasonable comparison. Each preprocessing method was trained for one epoch before validation. This was to maintain a training methodology that could be implemented in real-time with low memory and bandwidth requirements. Each preprocessing method was used to train and validate 100 RBFNN models.

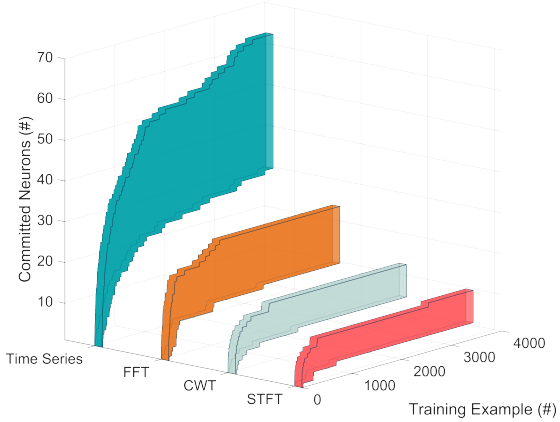
5 RESULTS

The summary statistics are in Figure 5.1a. Bar height shows the mean accuracy across all validation runs. The boxplot center shows the median, while the boxplot top and bottom show the interquartile range (IQR). Boxplot wiskers are 1.5 times the IQR above and below the boxplot. The STFT had both the highest accuracy and lowest skewness. The growth of committed

neurons was recorded for each training set, the envelope of neuron growth over the course of training for each preprocessing method is shown in Figure 5.1b. In this RCE implementation, the number of committed neurons can only increase where the methods are arranged in order from least to most committed neurons in Figure 5.1b.



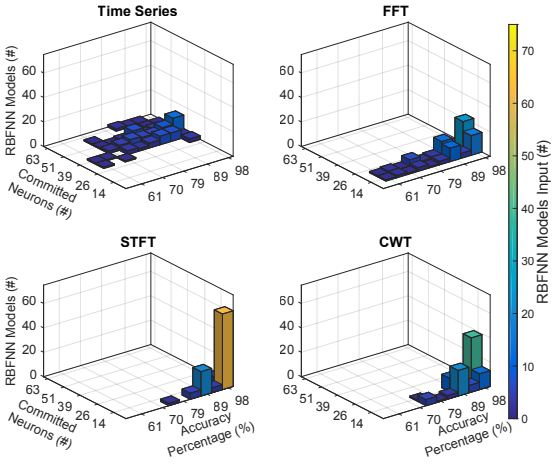
(a) Mean accuracy bar chart with boxplot



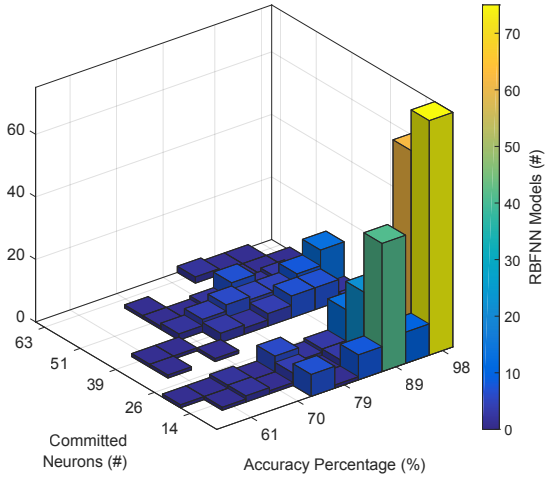
(b) Envelope of neuron growth by preprocessing method

Figure 5.1: Overall results

A Bivariate histogram for each method is shown in Figure 5.2a with the final number of neurons after completing one Monte-Carlo training epoch versus the accuracy of the validation for that model. For a dataset given two different preprocessing methods performing at the same classification accuracy level, the method that uses fewer neurons can be said to more efficiently processing the features of the input space. The time series method had the lowest mean accu-



(a) Bivariate histogram for each preprocessing method



(b) Total bivariate histogram sum of all methods

Figure 5.2: Bivariate histogram of accuracy and committed neurons

racy and most neurons committed, while the STFT had the highest mean accuracy and least neurons committed. Similarly, the FFT had the second lowest mean accuracy and second most

neurons committed and the CWT had the second highest mean accuracy and second least neurons committed.

The Pearson bivariate correlation coefficient calculated as the product of the standard scores shows the correlation between the number of committed neurons and accuracy [31],

$$\rho(acc, nn) = \frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{acc_i - \bar{acc}}{\sigma_{acc}}\right)\left(\frac{nn_i - \bar{nn}}{\sigma_{nn}}\right), \tag{5.1}$$

where acc is the mean accuracy and nn is the final number of neurons committed during training, σ is the standard deviation, and N is the number of trained RBFNN models. Of interest is that the p-value for both the time series and FFT are not significant, while both the STFT and CWT methods have a $p < .0001$. Neuron growth is a feature of the RBFNN and so can be evaluated in the aggregate across all input preprocessing methods. The composite of all methods in Figure 5.2b shows an acute trend where the more neurons are committed during training the lower the accuracy in validation. The composite across all the input preprocessing methods has a $p < 1 \times 10^{-7}$ demonstrating high correlation between the number of neurons and accuracy. While this correlation is useful, it is non-linear and non-deterministic, making it a useful heuristic for assessing relative accuracy between input preprocessing methods. The NM500 implementation of the RBFNN allows for up to 127 contexts, allowing independent RBFNN models to address different classification tasks sequentially on the same processing hardware. Practically speaking, this would allow all the preprocessing methods from this study to run essentially in parallel, where their relative performance could be assessed in real-time using the neuron growth heuristic as a framework for a higher level decision method. This could allow a priori higher accuracy classification because neuron growth precedes validation.

6 CONCLUSION

In this study, the committed neuron count is a useful heuristic for an RBFNN using an RCE neuron learning method that correlates well with the inference accuracy during validation. Since neuron count is intrinsic to the training process before any validation, it can be used to optimize the input preprocessing during training in real-time. It was shown, using the simulation of the NM500 neuromorphic chip, that a combination of neuron count and multiple RBFNN contexts provides a framework for evaluating input layer preprocessing methods. Conceivably, a higher level decision method based on this framework could be developed for real-time optimization of an RBFNN using RCE learning. Future work is planned to implement this work in hardware to investigate power and bandwidth performance.

ACKNOWLEDGMENTS

This research is funded by the U.S. Army.

REFERENCES

[1] nepes corp. *NM500 User's Hardware Manual*, 2019.

[2] Santu Sardar and K Ananda Babu. Hardware implementation of real-time, high performance, rce-nn based face recognition system. In *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*, pages 174–179. IEEE, 2014.

[3] Sam Wenke. *Application and Simulation of Neuromorphic Devices for use in Neural Networks*. PhD thesis, University of Cincinnati, 2018.

[4] Jaechan Cho, Yongchul Jung, Seongjoo Lee, and Yunho Jung. Vlsi implementation of restricted coulomb energy neural network with improved learning scheme. *Electronics*, 8(5):563, 2019.

[5] Seok-Hun Jeon, Jae-Hack Lee, Ji-Su Han, and Byung-Soo Kim. A study of unified framework with light weight artificial intelligence hardware for broad range of applications. *The Journal of the Korea institute of electronic communication sciences*, 14(5):969–976, 2019.

[6] Yann LeCun, LD Jackel, Léon Bottou, Corinna Cortes, John S Denker, Harris Drucker, Isabelle Guyon, Urs A Muller, Eduard Sackinger, Patrice Simard, et al. Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural networks: the statistical mechanics perspective*, 261:276, 1995.

[7] Ragam Prashanth and DS Nimaje. Estimation of ambiguous blast-induced ground vibration using intelligent models: A case study. *Noise & Vibration Worldwide*, 49(4):147–157, 2018.

[8] Shuai Zhang, Wing WY Ng, Jianjun Zhang, Chris D Nugent, Naomi Irvine, and Ting Wang. Evaluation of radial basis function neural network minimizing l-gem for sensor-based activity recognition. *Journal of Ambient Intelligence and Humanized Computing*, 1:1–11, 2019.

[9] Xiaofei Wang, Yiwen Han, Victor CM Leung, Dusit Niyato, Xueqiang Yan, and Xu Chen. Convergence of edge computing and deep learning: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 1, 2020.

[10] Tuba Kurban and Erkan Beşdok. A comparison of rbf neural network training algorithms for inertial sensor based terrain classification. *Sensors*, 9(8):6312–6329, 2009.

[11] Ahmed Hamza Osman and Ahmad A Alzahrani. New approach for automated epileptic disease diagnosis using an integrated self-organization map and radial basis function neural network algorithm. *IEEE Access*, 7:4741–4747, 2018.

[12] Sung-Kwun Oh, Wook-Dong Kim, and Witold Pedrycz. Design of radial basis function neural network classifier realized with the aid of data preprocessing techniques: design and analysis. *International Journal of General Systems*, 45(4):434–454, 2016.

[13] Paola Picco, Maria Elisabetta Schiano, Silvio Incardone, Luca Repetti, Maurizio Demarte, Sara Pensieri, and Roberto Bozzano. Detection and characterization of meteotsunamis in the gulf of genoa. *Journal of Marine Science and Engineering*, 7(8):275, 2019.

[14] George Sideratos, Andreas Ikonomopoulos, and Nikos D Hatziargyriou. A novel fuzzy-based ensemble model for load forecasting using hybrid deep neural networks. *Electric Power Systems Research*, 178:106025, 2020.

[15] N. Kouda and N. Matsui. On the function approximation in restricted coulomb energy neural network with gaussian radial basis function. In *2010 World Automation Congress*, pages 1–5. IEEE, Sep. 2010.

[16] Sibo Yang, TO Ting, Ka Lok Man, and Sheng-Uei Guan. Investigation of neural networks for function approximation. In *ITQM*, pages 586–594, 2013.

[17] Michael J Hudak. Rce classifiers: Theory and practice. *Cybernetics and System*, 23(5):483–515, 1992.

[18] Viet-Anh Le, Hai-Xuan Le, Linh Nguyen, and Minh-Xuan Phan. An efficient adaptive hierarchical sliding mode control strategy using neural networks for 3d overhead cranes. *International Journal of Automation and Computing*, 16(5):614–627, 2019.

[19] Michael R Berthold and Jay Diamond. Boosting the performance of rbf networks with dynamic decay adjustment. In *Advances in neural information processing systems*, pages 521–528, 1995.

[20] Guo Dong and Ming Xie. Color clustering and learning for image segmentation based on neural networks. *IEEE transactions on neural networks*, 16(4):925–936, 2005.

[21] Alison Jenkins, Vinika Gupta, and Mary Lenoir. General regression neural networks, radial basis function neural networks, support vector machines, and feedforward neural networks. *arXiv preprint arXiv:1911.07115*, 1, 2019.

[22] Sai Shyamsunder Kamat. Analyzing radial basis function neural networks for predicting anomalies in intrusion detection systems, 2019.

[23] Anthony Agnello, Jeff Dosch, Robert Sill, Strether Smith, and Patrick Walter. Causes of zero offset in acceleration data acquired while measuring severe shock. 1, 2019. (Accessed: 29 March 2020).

[24] David S Broomhead and David Lowe. Radial basis functions, multi-variable functional interpolation and adaptive networks. Technical Report ADA196234, Royal Signals and Radar Establishment Malvern (United Kingdom), 1988.

[25] Douglas L Reilly, Leon N Cooper, and Charles Elbaum. A neural model for category learning. *Biological cybernetics*, 45(1):35–41, 1982.

[26] Christopher L Scofield, Douglas L Reilly, Charles Elbaum, and Leon N Cooper. Pattern class degeneracy in an unrestricted storage density memory. In *Neural Information Processing Systems*, pages 674–682, 1988.

[27] Saman K Halgamuge, Werner Poechmueller, and Manfred Glesner. An alternative approach for generation of membership functions and fuzzy rules based on radial and cubic basis function networks. *International Journal of Approximate Reasoning*, 12(3-4):279–298, 1995.

[28] Matteo Frigo and Steven G Johnson. The fastest fourier transform in the west. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE, 1997.

[29] A Michael Noll. Short-time spectrum and cepstrum techniques for vocal-pitch detection. *The Journal of the Acoustical Society of America*, 36(2):296–302, 1964.

[30] Jonathan M Lilly and Sofia C Olhede. Generalized morse wavelets as a superfamily of analytic wavelets. *IEEE Transactions on Signal Processing*, 60(11):6036–6041, 2012.

[31] Ronald A Fisher, Frank Yates, et al. Statistical tables for biological, agricultural and medical research. *Statistical tables for biological, agricultural and medical research.*, 1, 1934.