Design and Optimization of Edge Computing Distributed Neural Processor for Biomedical Rehabilitation with Sensor Fusion

Kofi Otseidu
Department of EECS
Northwestern University
Evanston, IL, USA
kofiotseidu2022@u.northwetsern.edu

Tianyu Jia
Department of EECS
Northwestern University
Evanston, IL, USA
TianyuJia2015@u.northwetsern.edu

Joshua Bryne
Department of EECS
Northwestern University
Evanston, IL, USA
JoshuaBryne2018@u.northwetsern.edu

Levi Hargrove
Shirley Ryan Ability Lab
Chicago, IL, USA
l-hargrove@northwestern.edu

Jie Gu
Department of EECS
Northwestern University
Evanston, IL, USA
jgu@northwetsern.edu

ABSTRACT

Modern biomedical devices use sensor fusion techniques to improve the classification accuracy of motion intent of users for rehabilitation application. The design of motion classifier observes significant challenges due to the large number of channels and stringent communication latency requirement. This paper proposes an edge-computing distributed neural processor to effectively reduce the data traffic and physical wiring congestion. A special local and global networking architecture is introduced to significantly reduce traffic among multi-chips in edge computing. To optimize the design space of the features selected, a systematic design methodology is proposed. A novel mixed-signal feature extraction approach with assistance of neural network distortion recovery is also provided to significantly reduce the silicon area. A 12-channel 55nm CMOS test chip was implemented to demonstrate the proposed systematic design methodology. The measurement shows the test chip consumes only 20uW power, more than 10,000X less power than the current clinically used microprocessor and can perform edge-computing networking operation within 5ms time.

Keywords

Neural network; Low power edge processing; Mixed signal feature extraction; Inter-chip communication; Biomedical devices.

1 INTRODUCTION

The fast growth of miniaturized and efficient electronics enables the development of unobtrusive and portable personal health care systems. Within the application space of biomedical sensors and processors, the rehabilitation assistive device, e.g. cybergloves, prosthetic limbs, is one of the fastest-growing fields that heavily reply on wearable high performance low power compute device to enable stringent real-time control of robotic assistive devices [1-2]. It is reported that over 156,000 patients in the U.S. suffer from the loss of lower or upper-limbs, which provides constant need of highly reliable, low power, and autonomous device for their rehabilitation treatment [3]. The robotic devices used for rehabilitation represent one of the most complex biomedical system for assistance of human activities. A major bottleneck in the building of a robust assistive prosthetic device is the development of energy efficient electronic system with an accurate signal processing method for sensing and classifying the intention of the users. Machine learning algorithms have been at the center of many amount of studies to improve the accuracy of classification [4-5]. To continuously improve the accuracy of motion detection, sensor fusion techniques which deploy heterogeneous sensors at a wide spread of body locations are used to increase the dimensionality of biological data which in turn produce a rich volume of information for high-fidelity classification [3, 6].

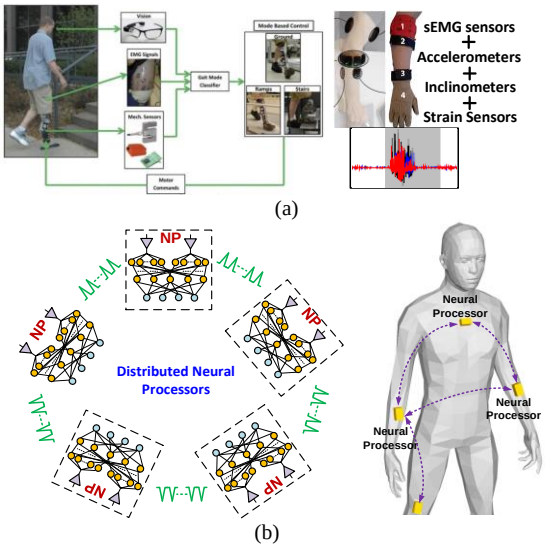

Figure 1: Overview of the biomedical rehabilitation sensor network and processing flow. (a) Sensor fusion used in existing rehabilitation [7-8]. (b) Proposed distributed neural processors.

In sensor fusion technique, heterogeneous sensors (e.g. surface electromyography (sEMG) sensors, strain sensors, accelerometers, inclinometers) across a wide body range are fused to provide highly accurate classification on patients' motion intent

ICCAD'18, November, 2018, San Diego, California USA

[7]. However, the large numbers of channels and heavy computing load lead to bottleneck at processor node. As shown in Fig. 1(a), the sensors for the motion recognition are distributed across human body, which creates physical wiring jam and data communication bottleneck at the microprocessor node. With sensor fusion, 10~100 channels with 80~800 input features need to be classified within 10~20ms to not delay the response from personal assistive device [7-8]. As a result, the heavy computation load poses significant challenges to modern wearable devices. The existing clinically used embedded microprocessor like the TI's OMAP4 processor, consumes six hundred milliwatts power. This results in heavy battery weight and routing congestion. More importantly, such a high-power consumption prevents the use of distributed architecture where computation is placed near the sensor nodes as the edge computing architecture proposed in this work.

In this work, we proposed a scalable distributed neural network processor, which brings the benefits of edge computing and reduce the data traffic for networking as well as silicon cost and memory space. A large neural network is effectively split into distributed smaller ones leading to significant reduced cost and communication latency. Fig. 1(b) shows the proposed configuration. Each neural processor is located near sensor node and performs local neural network classification. Only low dimension data is transferred through the network for final classification. The main contributions of this work are summarized as below:

- A novel edge computing neural processor for biomedical motion classification is proposed with a special distributed neural network (NN) architecture and communication protocol.
- A systemic design methodology and optimization strategy for the distributed NN architecture is provided in detail.
- A novel feature selection approach for sensor fusion is proposed to effectively reduce the computation requirement and memory space.
- A novel mixed-signal feature extraction approach assisted by NN is also introduced to significantly reduce silicon area.
- A test chip demonstration is provided to support the proposed edge computing method and systematic optimization scheme.

To the best of our knowledge, this is the first time, a complete edge computing technique is demonstrated for biomedical application through a machine learning capable neural processor.

2 EDGE COMPUTING USING DISTRIBUTED NEURAL NETWORK

2.1 Motivation

The conventional centralized computing strategy has been recently challenged by distributed computing such cloud computing or edge computing in the application space of smart health care system. For instance, cloud computing has been proposed for disease diagnosis where expensive computation jobs are uploaded to the cloud to relieve the computing burden on local devices [9-10]. However, cloud computing incurs large latency due to data exchange between cloud servers and local devices and hence does not meet the stringent computing time requirement in our targeted rehabilitation application. Fog or edge computing is also proposed where sensor data are pre-processed in distributed local devices rather than the central cloud servers [11-12]. The use of edge computing method reduces congestions on data movement and data computation leading to quicker response, reduced communication bandwidth requirement and reduced computing power needed to the servers. Existing edge computing techniques are mostly proposed for a large eco-system such as travel control, cellular network [12-13]. Body area network was also proposed to achieve shorter response time and better reliability as well as reduced communication bandwidth [14]. Unfortunately, all existing work used conventional Von Neumann architecture as computing unit. As machine learning accelerator, e.g. neural network processor, becomes more popular in existing compute unit, it is not clear how the emerging machine learning capable processing unit can be used to facilitate the benefit of edge computing.

This work, for the first time, proposes the incorporation of edge computing into the design of neural network (NN) for body area network used in biomedical application. Specially, we propose a distributed neural network design which combines both machine learning accelerator and edge computing techniques for energy efficient computing.

2.2 Proposed Distributed Neural Network

Fig. 2 shows the proposed architecture of the edge computing neural network processors. The sensed raw data by the biomedical sensors are processed by several local NN layers first, and then sent to a global output layers for classification. Multiple chips can be jointly combined to process a larger neural network. As a result, only low dimensional data needs to be communicated across chips significantly reducing both physical wiring connections and data traffic around the body area network. In addition, the multi-chip solution brings economic benefits of scalability as single chip does not need to be designed to cover the worst scenario when large number of channels are to be supported. The scalability of neural network leads to significant saving of silicon costs. Comparing with the conventional fully-connected multilayer perceptron (MLP) architecture, the proposed distributed NN architecture split the hidden layers at local node. As shown in Fig. 2, the implemented distributed neural network from multi-chips can collaboratively complete classification on large numbers of input nodes. One important benefit obtained from the distributed neural network is that the number of neuron connections are significantly reduced with slight reducing accuracy.

In the proposed NN design, each chip consists of both local and global NN layers. Only global NN layer is communicated externally. Multiple chips can be connected into a larger network. Compared with conventional single chip solution where all input channels, e.g. 72 channels in our example, need to be included in a single chip, the distributed design allows smaller units to form a larger network. The distribution of neural processor also brings the compute units closer to the sensor nodes leading to reduction of the traffic as well as physical wiring around the body. The parameter optimization and design strategy will be further discussed in Section 3.

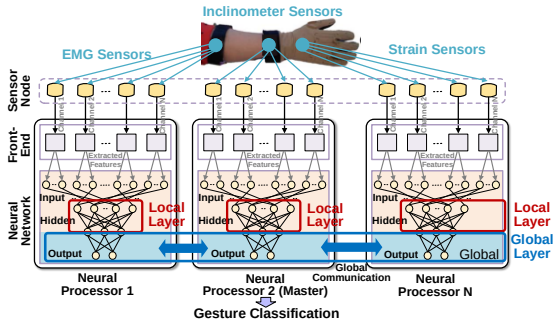

Figure 2: Overall distributed neural network architecture.

2.3 Communication Protocol

Fig. 3 shows the networking protocol of the proposed distributed neural network. Each chip will be given a chip ID and has all knowledges of how many chips, neuron nodes exist in the

network. A master chip will be responsible for starting communication as well as providing a global clock to sync up the remaining chips. Each chip sequentially sends its hidden layer neuron output to the global data bus. While one chip is sending data, all chips would be reading data from the single-bit data line.

The global clock signal works to synchronize individual chip clocks that would contain slight clock frequency mismatch and may be out of phase. The sender chip sends data to the data line at rising edge of global clock. This new data will not be read by the rest of chips until a falling edge from the global clock occurs. The period of the global clock, T_{global} is a few times larger than the period of the local clock on chip clock, T_{local}. Since the global clock is a few times slower than the local clock, the mismatch in phase and frequency of the local clocks in different chips would not result in errors in data transmission. To keep track of what data has been sent and received, each chip keeps counters of the current state of which bits have been send, what neurons have been sent and which chips have sent data. Fig. 3 shows the communication protocol diagram for the distributed neural network design.

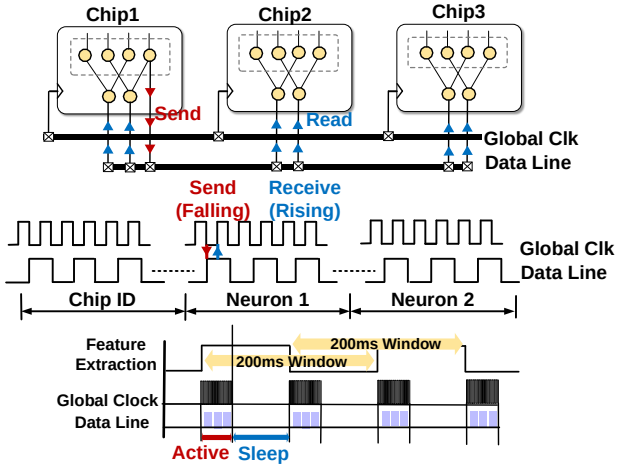

Figure 3: Communication protocol and networking between chips.

3 OPTIMIZATIONS FOR DISTRIBUTED NEURAL NETWORK

3.1 Distributed Processor Number

As explained before, the conventional fully-connected NN architecture is split into distributed processors to achieve edge processing. The number of distributed processors, i.e. parameter *P* in the following discussion, is one of the key parameters during the systematic design, as shown in Fig. 4.

The use of local and global neural network allows significant reduction of networking latency compared with fully connected neural network. The latency for the fully-connected MLP architecture can be expressed as equation (1).

$$t_{FC,latency} = I_t \cdot B \cdot T_{global} \quad (1)$$

in which I_t represents the total number of neurons inputs, B is the number of bits for each neuron. Meanwhile the latency for the proposed distributed NN architecture is modeled as equation (2).

$$t_{dist,latency} = \frac{I_t}{P} \cdot T_{local} \quad (2)$$

where *P* is the number of the distributed processors. Fig. 5 shows the simulated communication latency improvement with the scaling of the input neural nodes. Compared with fully connected network, in a three-chip distribution configuration, a 48X~240X reduction in networking latency is observed by the proposed distributed NN scheme.

Besides the latency, the proposed distributed network also leads significant memory storage space reduction. The required memory for storing the NN weights in unit of bit can be expressed by equation (3).

$$S_{MEM} = \frac{I_t \cdot N_i + \sum_{i=2}^{h} N_i \cdot N_{i-1}}{P} \cdot B \quad (3)$$

The neuron numbers within each layer are represented by N_i. As the simulated result in Fig. 5, there is about 3~5X reduction of on-chip memory storage space.

While significant saving in latency, area, and power is observed in the proposed networking scheme, classification accuracy is slightly reduced compared with fully-connected network leading a tradeoff of power and cost with accuracy. As shown in Fig. 6, with distributed processors, the inference accuracy is slight dropped by about 1~3% for one to three connected processors. As the completion time is critical for rehabilitation application, latency holds highest priority while low power is also important requirement for edge computing. Hence the accuracy is slightly traded off in this design to improve the overall performance, e.g. latency and power.

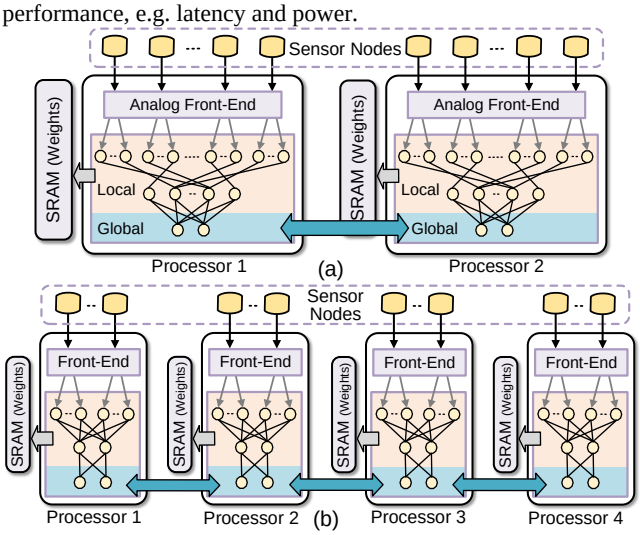

Figure 4: Dividing distributed neural distributed neural network (a) case of 2 processors (b) case of 4 processors.

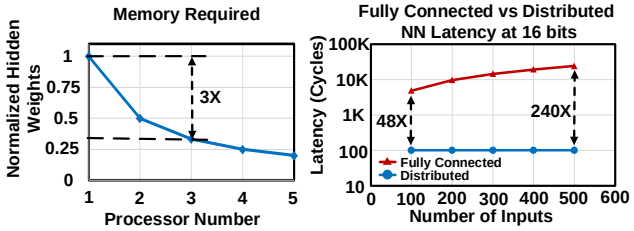

Figure 5: Benefit of using a distributed neural network in terms of latency and memory space.

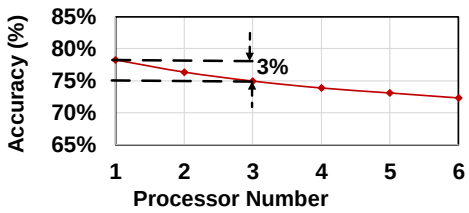

Figure 6: The accuracy impact of NN distribution numbers.

3.2 NN Architecture Optimization

The optimizations for the NN hidden layer number and neuron numbers are discussed in this section. Same as the conventional NN design, the tradeoff between accuracy and area overhead dictates the design choices.

For our target application, i.e. rehabilitation with sensor fusion, the total channels of the input sensing signals and associated features determine the number of input layer neurons, in the order of 80~800 input neurons as in our test cases. Accordingly, we performed simulation on the choices of the hidden layers and neuron numbers. As shown in Fig. 7, with more hidden layers, the NN accuracy can be improved by 1.5%. Meanwhile, the space required from memory increases by 70%, which lead to a 2.25X increase in latency as well as 3.4X increase in area. As a result, given the priority for latency and chip power, a single hidden layer is chosen in the final design.

Fig 7 shows the effect of neuron number on accuracy, communication latency and memory power. As the number of neurons increase, the prediction accuracy does increase, the rate of increase quickly saturates from the 24 neuron case. At the same time, increasing the number of neurons in the hidden layer neuron number would increase the communication latency since more neurons would need to send data. The amount of memory space needed also increases proportionally with the number of neurons added. As a result, 24 neurons per chip for a total of 72 neurons across 3 chips was decided under the tradeoff of accuracy, memory space and latency.

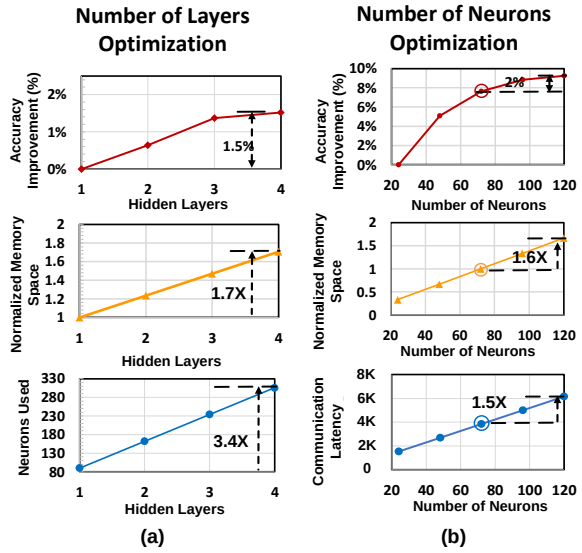

Figure 7: (a) Effect of neural network architecture (b) effect of neuron number on the hidden layer for three-chip configuration.

4 OPTIMIZATION FOR SENSOR FUSION

4.1 Characteristics of Heterogeneous Sensors

We evaluate our work using the published Ninapro database which contains 40 subjects with 72 channels and totally 10 hours of movement [7]. Three types of sensor data are included in the database for motion detection in upper limbs: surface EMG (sEMG) sensors, accelerometer sensors and glove strain sensors. The sEMG signals are gathered by 12 active double–differential wireless electrodes from a Delsys Trigno Wireless EMG system. The sEMG signal which is sampled at 2kHz is then filtered by a 50Hz filter to reduce the noise present in body area. Accelerometers are used to detect the acceleration change in motion within the gesture movement. 3-axis acceleration measurement is provided in the Delsys Trigno Wireless System. In total, 12 accelerometer sensors with 3 axes per sensor are used to generate 36 channels of acceleration data. In addition, the CyberGlove II is used for strain measurement at the joints of the arms [7]. Totally 22 channels are provided for strain measurements.

The use of sensor fusion techniques creates high accuracy classification on users' motion intent but also introduce large amount of data to be processed. Different from image processing, the physiological data possesses highly stochastic biological signals. As a result, features are extracted prior to classification. In this work, we extract the most commonly used features of the signals including mean, variance, the number of slope sign changes and histogram. The number of input neurons for the neural network equals to the multiplication of numbers of input channels and features used for each channel. As a result, the choices of features are important to achieve the best energy efficiency of the hardware design.

4.2 Feature Ranking with Sensor Characteristics

As different sensors, e.g. EMG, accelerometers, contain different characteristics of the signals, it is important to develop a methodology to analyze the significance of each feature for each sensor channel. In this work, we propose a novel statistical evaluation method which formally rank the sensor features according to its contribution to the final accuracy. To achieve the goal, we propose using the two-sample Kolmogorv-Smirnov statistical test, where we compare a distribution of data points to another distribution of data points belonging to another label in order to create a matrix of comparison of how different the data from each label is from each other. This procedure is given in Algorithm 1 below.

Algorithm 1 Feature Rank

Procedure *Feature Rank* (*sensors, label_list, channel_list, feature_list, data*)
1. **foreach** *k* ∈ sensors **do**
   //finding the similarity for each feature
2. **foreach** *feature* ∈ feature_list **do**
3. **foreach** *channel* ∈ channel_list **do**
4. *data_s*←*get_feature(data, channel, feature, sensor)*
5. **foreach** *i* ∈ label_list **do**
6. **foreach** *j* ∈ label_list && *j* > *i* **do**
7. *dist1*←*extract_distribution*(*data_s, i*)
8. *dist2*←*extract_distribution*(*data_s, j*)
9. *score_m(i,j)*←*two_sample_ttest (dist1, dist2)*
10. **end for**
11. **end for**
12. *channel_s(channel)*←*mean*(*score_m*)
13. **end for**
14. *feature_scores(sensor,feaure)*←*mean* (*channel_s*)
15. **end for**
16. **end for**
17. return sort(*feature_scores*) //return the order of all the scores

In this algorithm *data* represents the full dataset used. *Sensors* is the list of the types of sensors such as EMG, accelerometers and strain glove. The *label_list* is the list of all possible labels. *channel_list* is the channels associated with each sensor. *feature_list* contains all types of features being analyzed. This code would loop through every feature for every channel for every sensor and calculate a ranking score for that channel. To do this,

the data from a feature for a channel is divided into sections. These sections are grouped with examples with matching labels. To calculate the score, a two-sample t test is run on each of the distributions to determine how different labels affect the distribution. Every combination is averaged together to create one score for this channel's feature. Features of channels that shows low differentiation among different labels would provide data that is more ambiguous than features of that with high scores leading to confusion and difficulty for classification. Note that such a result varies from channel to channel. Once this is done for all channels feature combinations, scores are aggregated by sensor type and feature type to create a score for each combination of feature and sensor.

Fig. 8 shows the normalized scores given to features based on the feature rank method. For the sEMG channels, variance is the most important signal. For accelerometers, the mean feature is more important than the variance as well as some of the higher range histogram bins. The strain sensors from cyberglove values all the features although the mean, variance is more significant.

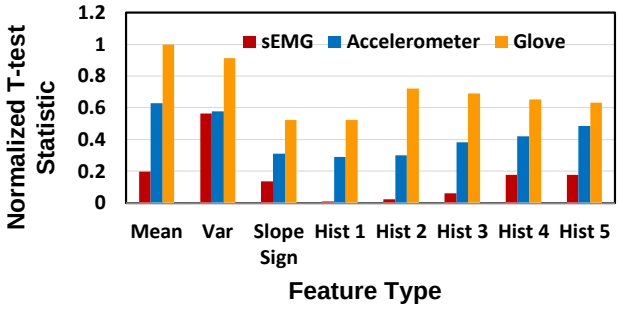

Figure 8: Feature rank score chart for each sensor and feature combination.

4.3 Feature Space Reduction

The benefit of removing various features is the reduction in weights required for the neural network as well as power saving from feature extraction. By choosing the right features for certain sensors, we minimize the impact on accuracy. The optimization problem proposed is to remove as many features from various sensors as possible while maintaining an accuracy loss within 1%. Using the proposed feature ranking method proposed in Algorithm 1, the search space of the optimization problem can be simplified. Fig. 9 shows the results of these simulations. In this test we used a neural network divided into three sections with different sensors for each section. Algorithm 2 shows the pseudo codes for feature selection.

Algorithm 2 Optimizing Features Selection

Procedure *neural_network_prunning* (*ranked_feature*, *max_accuracy*)
1. *performance*←*mac_accuracy*
2. **while** *max_accuracy-performance < 1%* **do**
3. *HiddenWeights*←*remove_feature*(*rank_feature*(*i*))
4. *performance*←*nn_classification (HiddenWeights)*
5. *i*←*i+1*
6. **end while**
7. return i-1 //return how many features were removed

The *ranked_feature* is a list of ranked features determined by the rank feature procedure described in Algorithm 1. *max_accuracy* is accuracy attained without removing any features. The algorithm loops through the list of the worst ranked features and removes the links to that feature within the hidden weights. After this is done, we run the training and testing procedure of the neural network without that feature and obtain a prediction accuracy. The procedure is repeated to the next lowest feature until significant performance loss, e.g. 1% is observed. Fig. 9 plots the recorded accuracy of this test. In total there are 24 sensor feature combinations. The ranked feature algorithm allows 8 different sensor feature combinations to be removed while keeping the accuracy reduction within 1%. If 4 features are chosen at random, the accuracy loss can exceed 1%. Tolerating a loss of 1% can reduce the amount of memory required by an additional 20% when using the feature ranking method. This would also result in a reduction of computing power.

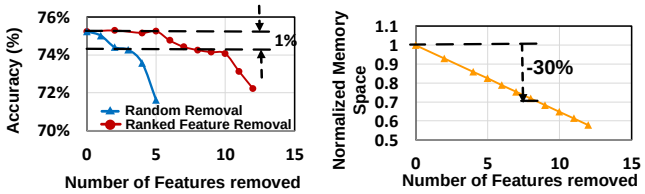

Figure 9: (a) Ranked feature removal vs random feature removal (b) Memory Reduction vs. number of features removed.

5 ANALOG MIXED-SIGNAL SENSING AND DISTORTION RECOVERY

To reduce the overhead of the design, we also propose a novel mixed signal feature extraction design which directly convert the analog signals into digital for the neural network classification. Conventional design uses high precision analog-digital converter (ADC) to process the signal from analog to the digital domain. In addition, a digital block for digital feature extractions (DFE) would be required to convert the digital signal into time-domain features, e.g. mean, variance, histogram, slope sign change, etc. [7]. The proposed architecture removes this two-step process and combines the front-end ADC and DFE into a simple direct mixed-signal feature extraction unit leading to 28X saving in area.

5.1 Mixed-signal Feature Extraction

The proposed mixed signal feature extraction unit calculates the four features discussed in section 4.1, i.e. mean, variance, slope-sign change, and five histogram bins. Fig. 10 presents the proposed mixed-signal feature extraction techniques where eight features of analog signals are extracted using only simple VCO, comparator, counters, etc.

With incoming signal bandwidth of a few kHz, the VCOs run at sub-threshold region between 10-300kHz speed and deliver pulses to subsequent counters for feature extraction. To calculate the mean feature, we send VCO's output to a counter. Since mean is proportional to the sum of all the events, we used this as the mean feature eliminating expensive digital calculation on mean feature and analog-to digital conversion. The ideal mean calculation can be represented by equation (4). The VCO based mean calculation can be represented by equation (5).

$$Mean_{ideal} = \sum_{i=1}^{N} \frac{Vin(i)}{N} \quad (4)$$

$$Mean_{vco} = \int_{0}^{N} VCO(Vin(i)) \quad (5)$$

in which N represents the total number of examples in a window and Vin represents the voltage. The VCO function converts the voltage at time *i* into a count value that would be accumulated. The

ideal variance calculation can be represented by equation (6). The VCO based variance can be represented by equation (7).

$$Var_{ideal} = \sum_{i=1}^{N} \frac{(Vin(i) - \mu)^2}{N} \quad (6)$$

$$Var_{vco} = \int_{0}^{N} VCO(Vin(i) - \mu) \quad (7)$$

in which μ is the average value of this channel. Like the mean VCO function, the variance VCO function converts the voltage at time *i* into a count value that would be accumulated. The overall design structure is similar to the mean as well. The VCO however is modified to take in a differential signal. The incoming analog signal is sent through a differential amplifier to modulate VCO speed according to signal's deviation from its average input. Since we are calculating the distance from the average value, this operation approximates the ideal variance operation.

The calculation for the slope sign change can be represented by equation (8).

$$SSC = \sum \left[sign\left(\frac{dVin(i)}{dt}\right) \neq sign\left(\frac{dVin(i-1)}{dt}\right) \right] \quad (8)$$

The slope sign change feature uses a bi-directional counter with the mean VCO. For one millisecond, this counter will count followed by one millisecond where the counter will count down. The most significant bit of the counter results is then compared with that from the previous 2-millisecond cycle. If this bit has changed, we determine that the slope sign has changed and will increment the output counts.

The calculation for histogram is shown in equation (9).

$$Hist = \sum_{bin\ n}^{B} \sum (Vin(i) > Vth(n)_l \ \&\ Vin(i) < Vth(n)_h) \quad (9)$$

where B is the total number of bins, $Vth(n)_l$ is the lower bound of bin *n* and $Vth(n)_h$ is the lower bound of bin *n*. To calculate the histogram of the inputs, the channel voltage is sent to a series a clocked comparator with various levels of reference voltages to determine what bin range the voltage fell into. The clocked comparators a triggered once every millisecond and produce a clock like pulse which is sent to the counter. Each bin range would have a separate counter.

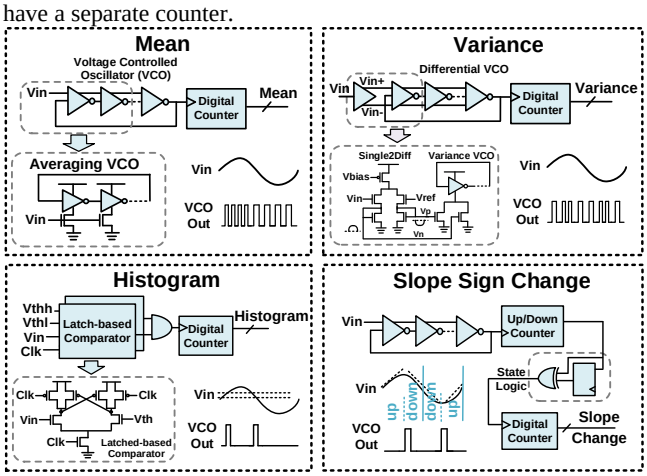

Figure 10: Circuit implementation of feature extraction.

5.2 Distortion Recovery from Neural Network

Despite of the dramatic saving from the proposed scheme by removal of ADC, such a VCO based conversion method leads to strong distortion in feature obtained [15]. Fig. 11 shows the non-linear relationship between input voltage and count generated. At the top end of the distribution, the count shows a decrease in linearity while the bottom end also loses some of the linearity as well. For the mean feature, this distorted curve can be modelled as equation 11.

$$Mean = -1.5x^4 + 0.5x^3 + 2.3x^2 - 0.1x \quad (10)$$

x represents the normalized signal value coming from a sensor.

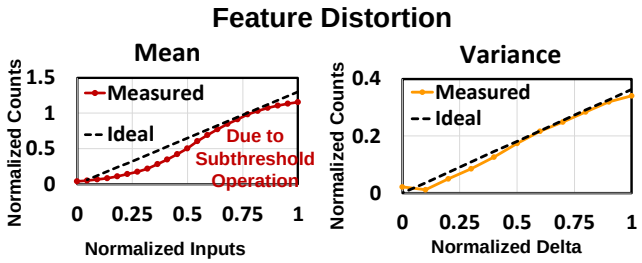

Figure 11: Ideal vs measured feature results for mean and variance.

The features mean, and variance show distortion from VCOs because the speed of the VCO is not linear with respect to the voltage input due to the operation in the near/subthreshold region of the transistors in VCO. Fig. 12(a) shows the loss of functional mapping between the ideal floating-point feature value and the VCO circuit implementation-based design.

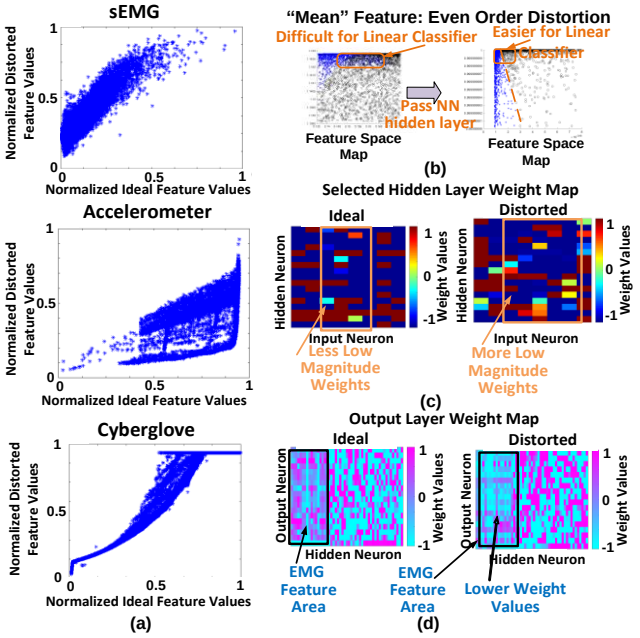

Figure 12: (a) Ideal feature values vs. distorted feature values (b) NN correction (c) hidden weight distortion filtering (d) output weight distortion filtering.

As seen in equation (10), the near-threshold operation of VCO produces strong 2nd and 4th order distortion leading to collapse of feature spaces and degradation from linear classifiers. Such a distortion leads to significant degradation, 6% from commonly used classifier, e.g. simple linear SVM. However, the degradation from neural network (NN) is only 1%, thanks to the strong nonlinear operation of neural network. The training of NN using the distorted feature characteristics leads to a recovery of the accuracy loss from the low-cost feature extraction circuits in this work.

Fig. 12(c) and Fig. 12(d) show how the weights are filtered by the neural network to combat distorted data. Given that the feature data is of a similar magnitude data will tend to have much smaller weights after training. This reduces the focus of the results on the distorted data and in turn moves it to less distorted features. This is seen within some individual weights associated with features as well as entire neurons if the results fed to the neuron are quite distorted. The error for each weight can be calculated using equation (11).

$$Error = \left(L_2Norm\left(\sigma\left(O_W \cdot \sigma(H_W \cdot I)\right)\right) - t\right) \quad (11)$$

in which O_W represents output weights, H_W represents hidden weights, and σ represents the activation function. I is the input vector and t the target vector for the example in question. The change in weights are calculated by equation (12).

$$\Delta O_W = \left(d\sigma(O_W \cdot H_V)\right)(O_V - t) \quad (12)$$

in which O_V represents the output of the output layer and H_V represents the output of the hidden layer. If the data is distorted, the delta weight values would remain large over time. Features that contain inconsistent results within the neural network would have a much tougher time creating a consistent impact on the for the backpropagation weights causing these values to go back and forth. The neural network will filter out these inconsistent features through the backpropagation algorithm. Overall, the use of neural network allows elimination of expensive analog front-end, e.g. ADC, leading to significant saving of silicon area. The proposed mixed-signal architecture highlights another contribution from machine learning technique to modern electronic design.

6 TEST CHIP MEASURMENT

6.1 Design Overview

To verify the proposed scheme, a 55nm CMOS test chip is built. Fig. 13 shows the top-level design of the system with 3 neural network processor chips. Each chip was built with 12 channels. The mixed signal feature extraction unit provides each channel with 8 extracted features. On chip memory was provided to store the values of the weights used.

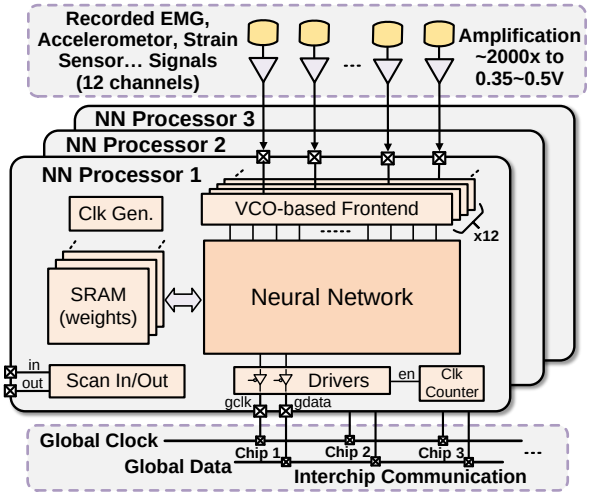

Figure 13: Neural network processor diagram.

For the testing setup, multiple chips are mounted on the test PCB board. A FPGA was used as interface for programming the chip and scanning in and out the data. The classification results are gathered through the scan out signal. The recorded analog signals from patients in Ninapro database are replayed using the USB-DA12-8A digital to analog converters (DAC) by ACCES. The EMG signals were amplified by ~2000X.

6.2 Classification Accuracy and Networking

Fig. 14 shows the measured classification results compared with ideal PC-based floating-point operation using Ninapro database [7]. When using the sensor fusion technique, accuracy improves with more channels due to the sensor fusion technique. Using floating point as opposed to 16-bit fixed point only degrades performance by 1%-2%. The degradation is limited within 2% due to the integer point operation and distortion from mixed-signal feature extraction. The confusion matrix in Fig. 14 shows where the errors with respect to each gesture is coming from with respect to each gesture. This shows that there are some similar gestures such as the wrist movements have a much higher chance of being confused with each other.

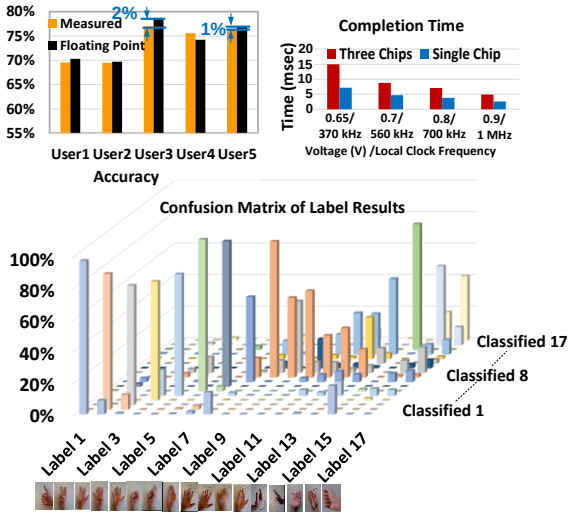

Figure 14: Measured motion classification accuracy and completion time.

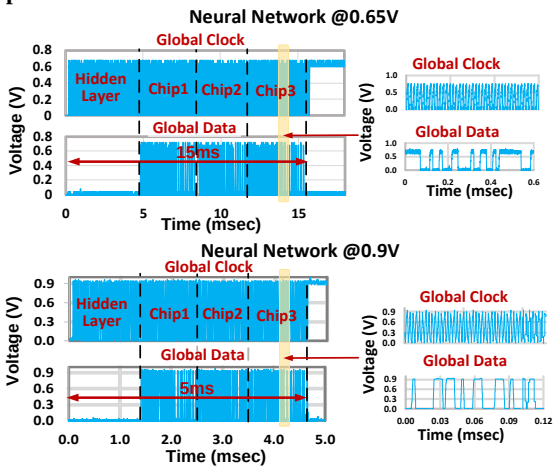

Figure 15: Measured communication waveforms across three chips.

Fig. 15 shows the measured communication waveforms when connecting three chips together. Communication is completed within 5~15ms scalable with supply voltages. When the supply voltage is at 0.65V for both the neural network and SRAM, the classification time would take 15msec. This is still under the 20msec tolerance time for notice by the users [3]. The same operation can be completed within a third of the time at a 0.9V supply voltage for SRAM and neural network. The communication

measurement confirms the effectiveness of the proposed architecture using edge computing distributed neural network. Up to six chips can be connected in the current implementation.

6.3 Area and Power Consumption

Fig. 16 shows the benefits of mixed-signal feature extraction compared with digital implementation of feature extraction using traditional ASIC design flow. For each channel, area was reduced by a factor of 4. Another benefit of the mixed signal extraction is the reduction in the use of the ADC. Compared with prior report, a 24X reduction in area is observed using the proposed mixed signal feature extraction [16,17]. Overall, a 28X reduction in area is achieved including digital feature extraction. Fig. 15 contains a power breakdown at various supply voltages. The proposed design is dominated by the feature extraction circuit since this block must always be turned on to gather continuous data. The neural network is clock gated at ~5% activation rate reducing the required power of this section. At a voltage of 0.65V, a 2.1µW/channel or total 26µW is observed. Due to the lack of existing ASIC design for motion classification, we compare with existing ECG processor and slower EMG applications [18,19]. This design consumes smaller power per channel with the addition of networking capability compared with that used in ECG application and complete the tasks in millisecond instead of second's operation in ECG application [18]. Compared with the existing clinically used microprocessor which consumes 600mW and requires 15ms for the same classification job, more than 10,000X power reduction is achieved.

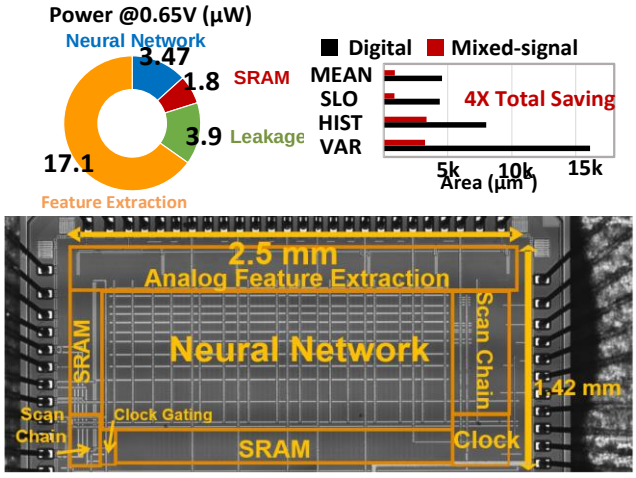

Figure 16: Measured single chip power, area savings and chip micrograph.

7 CONCLUSION

This work proposes an architecture and optimization method for a novel edge computing distributed neural processor for motion intent recognition used in rehabilitation. The distributed neural network processors produce a scalable, expandable neural network and effectively reduce the communication latency and memory space. Systematic design and optimization approaches including a novel feature ranking approach are proposed in this work to further improve the efficiency of the design. Edge computing can produce a 48X speedup in communication latency within a given layer and can reduce the number of weights required by 3X when using three processor units. Mixed-signal feature extraction design is also proposed to reduce the area by 24X. A 12-channel 55nm CMOS test chip is implemented consuming 2.1µW/channel with millisecond recognition time and networking capability fully demonstrating the proposed architecture and optimization method.

REFERENCES

[1] D. Farina, *et al.*, "The extraction of neural information from the surface EMG for the control of upper-limb prostheses: emerging avenues and challenges," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 797–809, 2014.

[2] N. Helleputte, *et al.*, "A 345 µW multi-sensor biomedical SoC with bio-impedance, 3-channel ECG, motion artifact reduction, and integrated DSP", *IEEE Journal of Solid-State Circuits*, vol. 50, no. 1, pp. 230-244, Jan. 2015.

[3] A. Young, *et al*., "Analysis of using EMG and mechanical sensors to enhance intent recognition in powered lower limb prostheses," *Journal of Neuural Engineering*, vol. 11, no. 5, Sep. 2014.

[4] S. Wurth, *et al*., "A real-time comparison between direct control, sequential pattern recognition control and simultaneous pattern recognition control using a fitts' law style assessment procedure", *Journal of NeuroEngineering and Rehabilitation*, vol. 11, no. 1, 2014.

[5] A. Adewuyi, *et al*., "An analysis of intrinsic and extrinsic hand muscle EMG for improved pattern recognition control", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 4, pp. 485-494, 2016.

[6] N. Krausz, *et al*., "Depth sensing for improved control of lower limb prostheses", *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 11, pp. 2576-2587, 2015.

[7] M. Atzori, *et al*., "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," in *Scientific Data*, 1:140053, Dec. 2014.

[8] N. Krausz, L. Hargrove, "Recognition of ascending stairs from 2D images for control of powered lower limb prostheses", *IEEE Inter. Conf. in Medicine and Biology Society (EMBC)*, 2015.

[9] A. Jamthe, *et al.*, "Harnessing big data for wireless body area network applications", *International Conf. on Computational Intelligence and Communication Networks (INFOCOM)*, 2015.

[10] G. Almashaqbeh, *et al.*, "A cloud-based interference-aware remote health monitoring system for non-hospitalized patients", *Symposium on Selected Areas in Communications*, 2014.

[11] H. Dubey, *et al.*, "Fog Computing in Medical Internet-of-Things: Architecture, Implementation, and Applications", *arXiv:* 1706.08012, 2017.

[12] W. Shi, *et al.*, "Edge computing: vision and challenges", *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637-646, Oct. 2016.

[13] M. Satyanarayanan, "The emergence of edge computing", *Computer*, vol. 50, no. 1, pp. 30-39, Jan. 2017.

[14] B. Calhoun, *et al.*, "Body sensor networks: a holistic approach from silicon to users", *Proceedings of the IEEE*, vol. 100, no. 1, pp. 91-106, Jan. 2012.

[15] K. AL-Tamimi, et al., "Preweighted Linearized VCO Analog-to Digitaal Converter," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol 435, pp. 1983-1987, June 2017.

[16] N. Desai, *et al.*, "A scalable, 2.9 mW, 1 Mb/s e-textiles body area network transceiver with remotely-powered nodes and bi-directional data communication", *IEEE Journal of Solid-State Circuits*, vol. 49, no. 9, pp. 1995-2004, Sep. 2014.

[17] J. Yoo, *et al.*, "An 8-channel scalable EEG acquisition SoC with fully integrated patient-specific seizure classification and recording processor," *ISSCC*, pp. 292-294, Feb. 2014.

[18] S. Yin, *et al.*, "A 1.06 µW smart ECG processor in 65 nm CMOS for real-time biometric authentication and personal cardiac monitoring," *Symposium on VLSI Circuits*, Jun. 2017.

[19] S. Benatti, *et al.*, "A sub-10mW real-time implementation for EMG hand gesture recognition based on a multi-core biomedical SoC," *IWASI*, pp. 139-144, Jul. 2017.