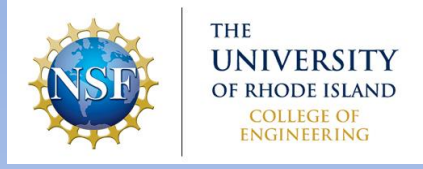


A Reconfigurable Multiclass Support Vector Machine Architecture for Real-Time Embedded Systems Classification

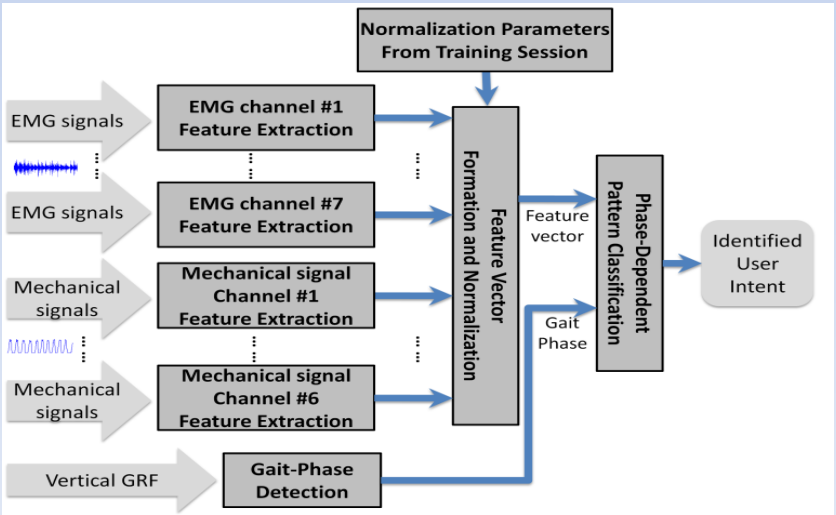
Jason Kane, Robert Hernandez, and Qing Yang
University of Rhode Island

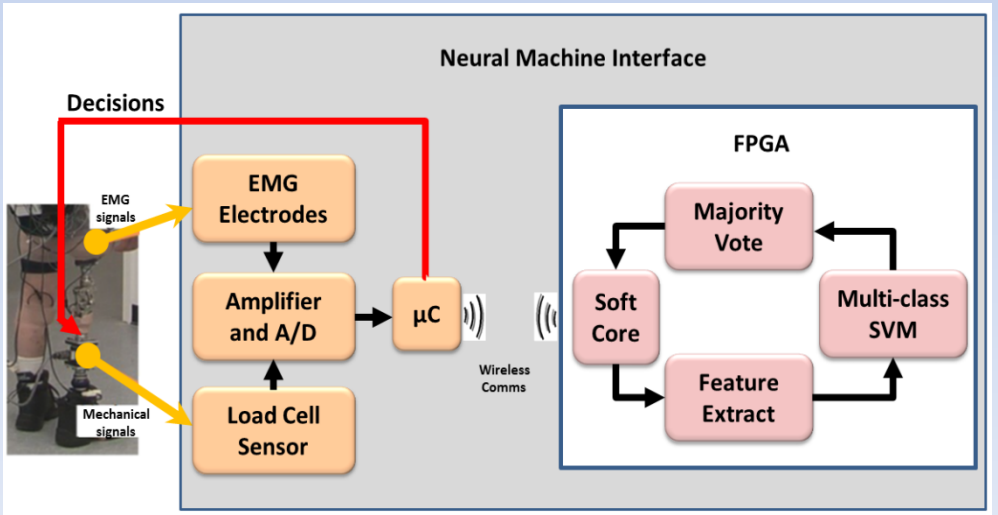
Overview

- Background
- Support Vector Machines Introduction
- Existing Classification Implementations
- R^2SVM Design
- Performance Results
- Future Work
- Questions

Background

- URI Electrical/Biomed Dept. looked into lower limb prosthesis control with EMG
  - Intel i7 tower implementation prototyped in lab with high accuracy. Had RT Issues.
  - FPGA prototype created to resolve RT problem.





Design Goal

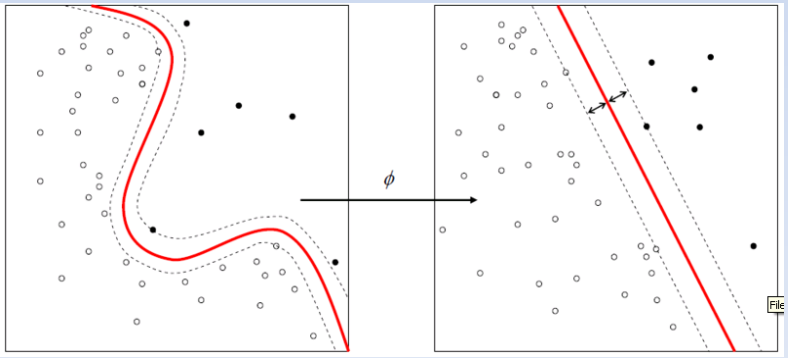
- What if we want to perform multiple, accurate classifications in Real-Time?
  - Ex1: Run multiple models and compare outputs.
  - Ex2: Run multiple models for control of a number of subsystem items: Ankle, Knee, Arm, Etc.

Objectives

- Design a real-time energy efficient, accurate, general purpose run-time reconfigurable accelerator for SVM
  - Optimize data paths for pipelining/parallelization
  - Work with 4 common kernels
  - Support up to a maximum #Classes/Features, specified at compilation
  - Allow for targeting of diverse workloads
- Develop a prototype compatible with libsvm to provide direct performance comparisons with existing architectures.
  - Operate with FP in Real-Time
- Evaluate performance using publically available machine learning datasets.

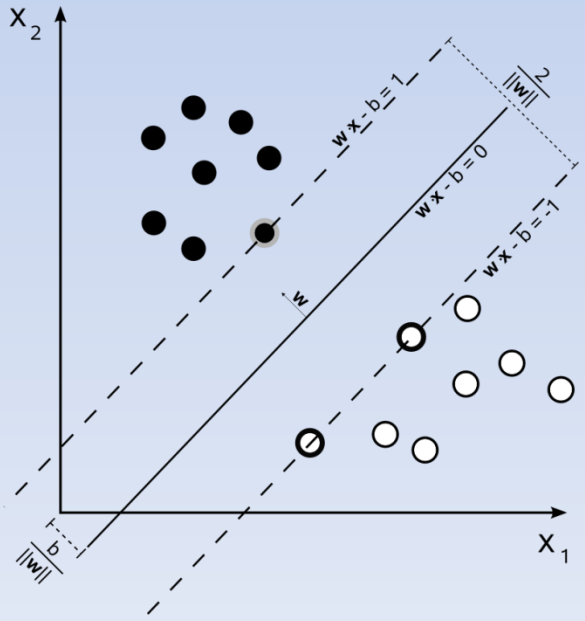
SVM Introduction

- Support Vector Machines (Binary Classification SVM)
  - Classification technique using machine supervised learning
    - 1995 Vapnik & Cortes [1]
  - Training Data is supplied in vector form.
  - Data is mapped into hyperplanes in a high dimension space either directly (Linear SVM) or using a “kernel” function to remap the data into a transformed feature space (Non-Linear SVM).



SVM Introduction

- Support Vector Machines (Binary Classification)
  - Hyperplanes constructed based on Largest Margin between data of one class and another.
    - Points lying on this margin are termed “Support Vectors”.
    - Hyperplanes used to classify data



SVM Test Phase

Decision Function:

$$f(x) = sign(\sum_{i=1}^{l} \alpha_i y_i k(x, x_i) + b)$$

Common SVM Kernels:

Linear Kernel

$$k(x_i, x_j) = ({x_i}^T x_j)$$

Polynomial Kernel

$$k(x_i, x_j) = (a{x_i}^T x_j + r)^d$$

Gaussian Kernel

$$k(x_i, x_j) = e^{\left(-\gamma \|x_i - x_j\|^2\right)}$$

Sigmoid Kernel

$$k(x_i, x_j) = \tanh(a{x_i}^T x_j + r)$$

Multi-Class SVM

- Multi-Class SVM
  - Extension of 1-Class SVM
  - Common Implementations
    - One Versus All - Classifier with highest output function wins.
    - One Versus One - Perform Multiple 1-Class SVM, Class with the most votes wins

Existing SVM Classifier Implementations

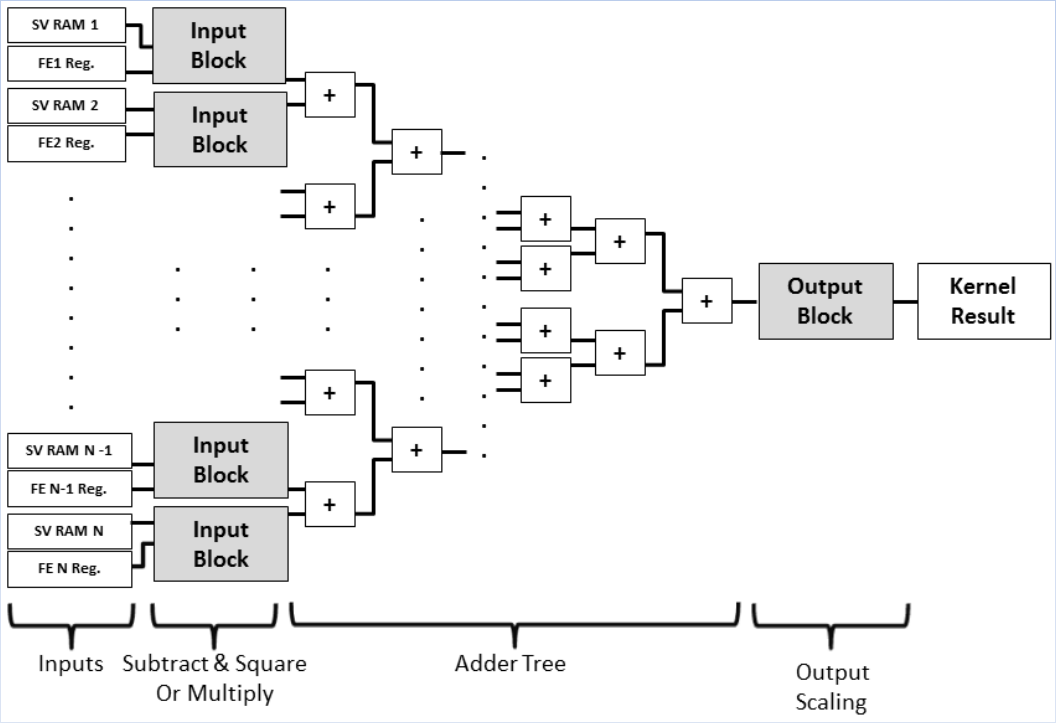
- CPU Software Based
  - Matlab libraries, libSVM, svmLite
  - No multithreaded versions
- GPU Based
  - Few public CUDA implementations: KMLib [2], GPUSVM. Neither support more than 2 classes.
  - We use a private library provided by the author of [3].
- FPGA Based
  - Mostly implementation specific designs.
  - None that support multi-class.

R²SVM: High-Level System Architecture

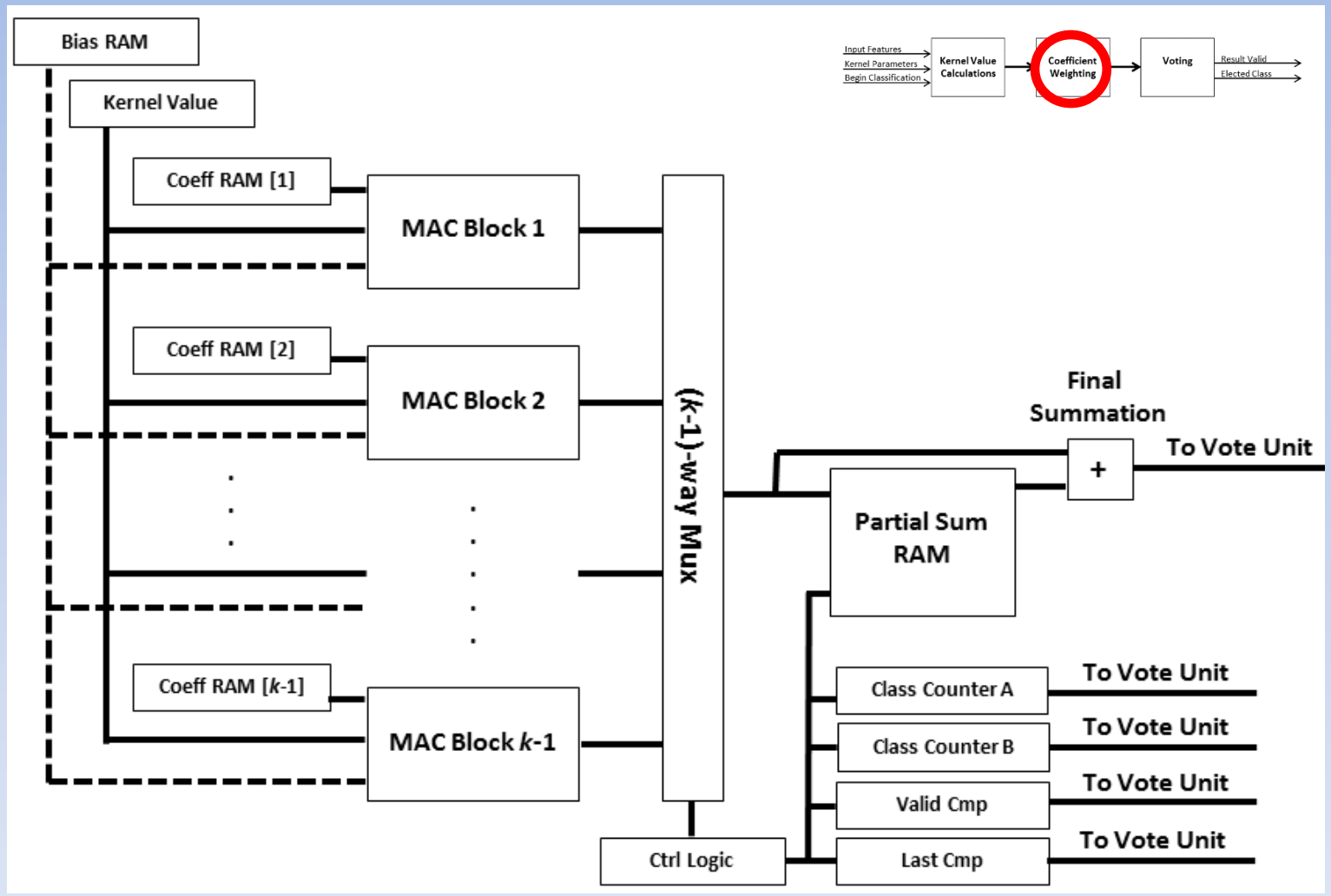
$$f(x) = sign(\sum_{i=1}^{l} \alpha_i y_i k(x, x_i) + b)$$

Kernel Calculations

- Goal:
  - Support 4 Kernels
  - Simplify Routing
    - Faster Clock
  - Pipeline
- Design:
  - Input Block
  - Common Block
  - Output Block

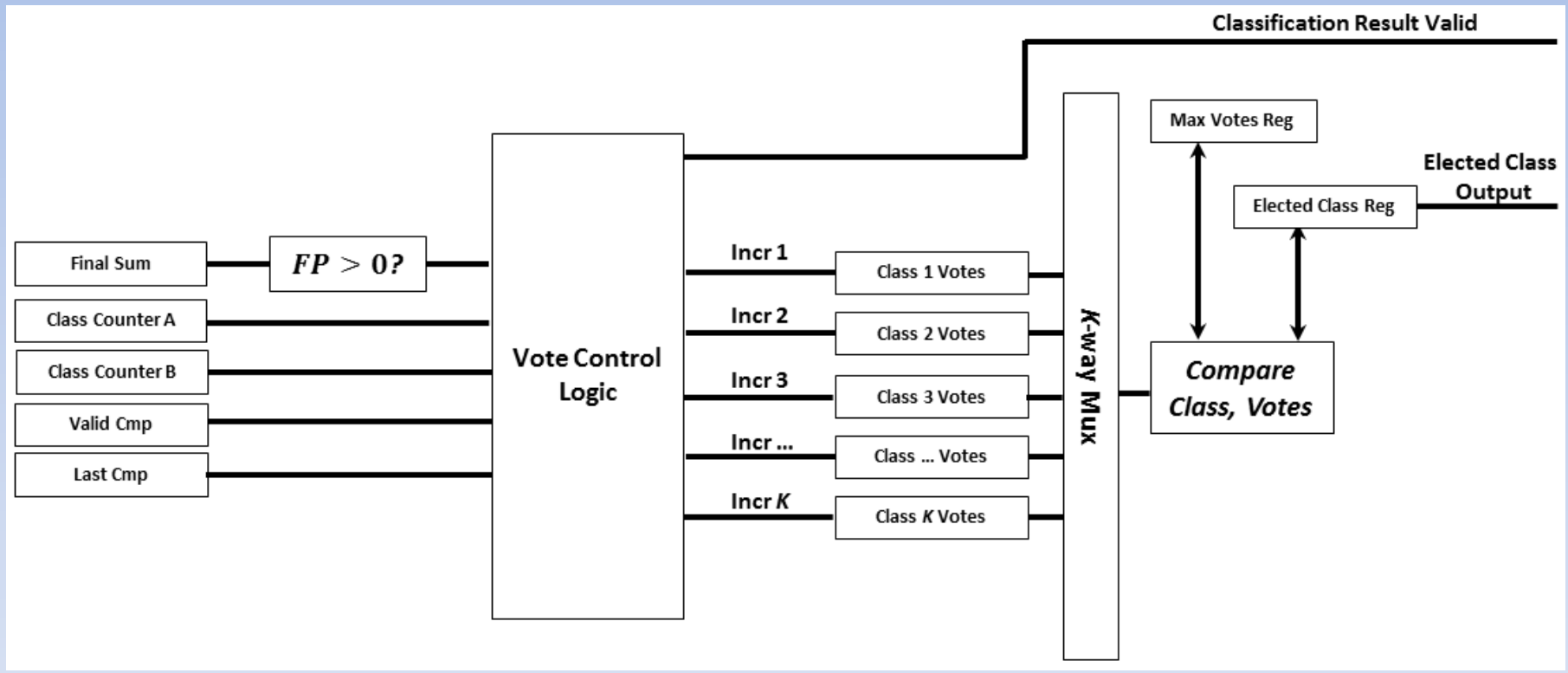


Coefficient Weighting

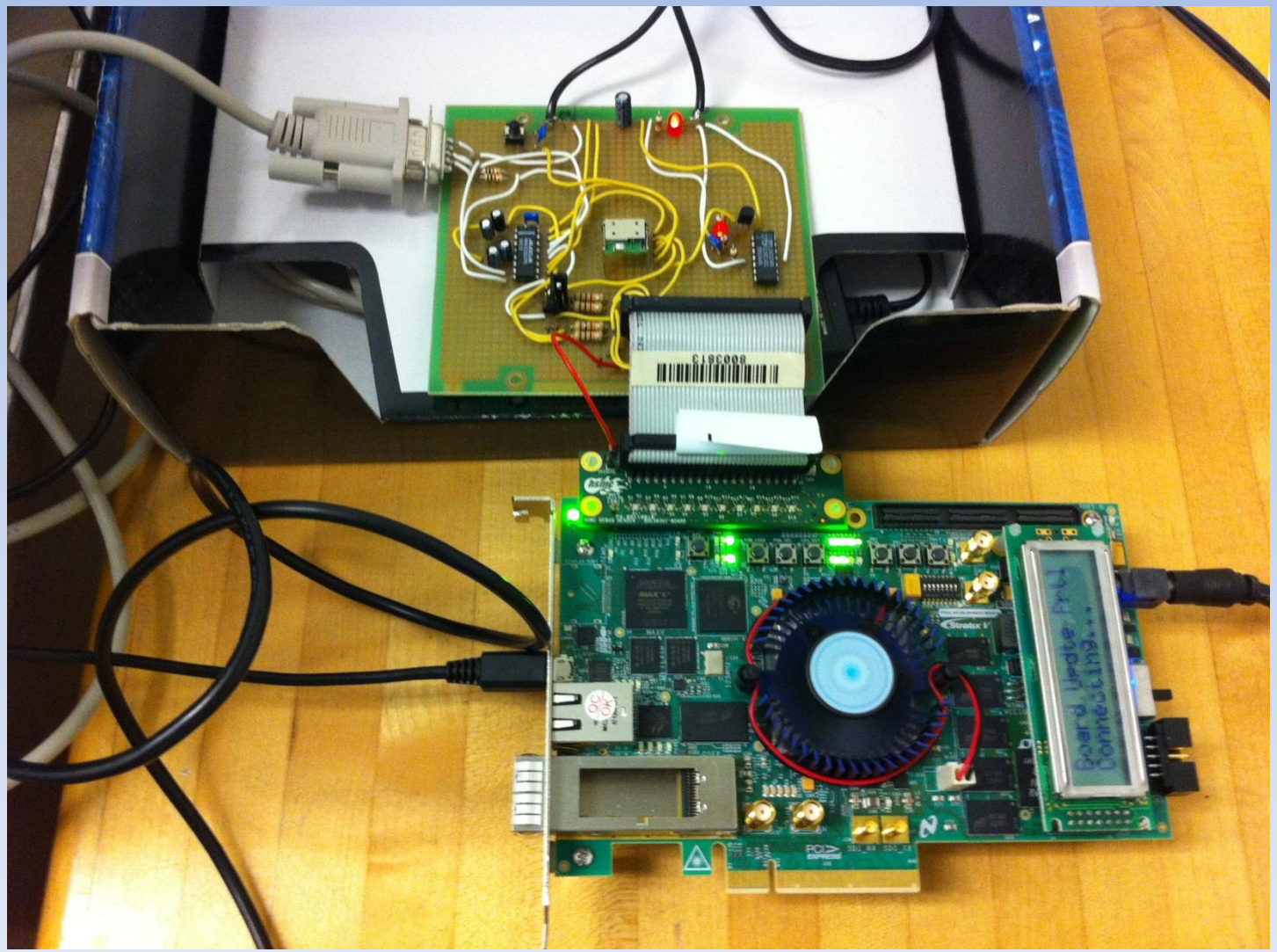


Voting Unit


Input Features
Kernel Parameters
Begin Classification
Kernel Value Calculations
Coefficient Weighting
Voting
Result Valid
Elected Class


Classification Result Valid
Max Votes Reg
Elected Class Output
Elected Class Reg
Final Sum
FP > 0?
Class Counter A
Class Counter B
Valid Cmp
Last Cmp
Vote Control Logic
Incr 1
Incr 2
Incr 3
Incr ...
Incr K
Class 1 Votes
Class 2 Votes
Class 3 Votes
Class ... Votes
Class K Votes
K-way Mux
Compare Class, Votes

Stratix V Development Board



Performance Testing

- Single compilation could have been achieved for timing
  - Multiple were made to show performance under different class/feature settings.
- 6 Standard Datasets used for Testing from Statlog/UCI
  - Case Study with data from URI Human-Computer Control Interface
- Compare timing/accuracy with i7-2600 (3.4GHz) and GeForce GT 750M GPU
  - SVM Models loaded prior to test for all architecture
  - Report Prediction Time

Prediction Results

Trial (#Class, #Feat) [Source]	Sub-trial	#SV	FPGA Time (µS)	Average FPGA Speedup		# Test Trials	Acc. FPGA (%)
				Over CPU	Over GPU		
Adult (2,123) libSVM, UCI	Lin	600	13.4	37.3x	14.7x	1000	100
	Poly	705	14.0	39.5x	23.3x	1000	100
	RBF	785	17.2	39.6x	14.0x	1000	100
	Sig	687	13.5	42.8x	17.2x	1000	100
DNA (3,180) Statlog	Lin	402	10.0	39.7x	17.7x	1187	100
	Poly	1053	21.8	39.1x	18.3x	1187	100
	RBF	696	13.5	53.7x	21.1x	1187	100
	Sig	444	9.4	48.7x	19.7x	1187	100
Letter (26,16) Statlog	Lin	6060	125.0	9.0x	6.6x	5000	99.97
	Poly	4408	100.0	10.0x	13.5x	5000	100
	RBF	5701	120.0	9.6x	7.2x	5000	100
	Sig	14602	266.0	14.4x	13.7x	5000	100
Shuttle (7,9) Statlog	Lin	3994	72.7	5.96x	6.9x	14500	99.97
	Poly	878	21.7	7.4x	14.9x	14500	100
	RBF	2208	43.6	7.1x	8.1x	14500	100
	Sig	4290	78.4	10.2x	8.6x	14500	100
Satimage (6,36) Statlog	Lin	1216	25.6	11.5x	9.5x	2000	99.95
	Poly	1051	21.7	13.8x	18.1x	2000	99.95
	RBF	1165	23.3	13.5x	11.0x	2000	100
	Sig	2468	47.3	15.2x	10.4x	2000	100
Vowel (11,10) UCI	Lin	268	13.7	4.0x	12.1x	463	100
	Poly	299	13.0	5.1x	6.72x	463	100
	RBF	290	56.8	4.0x	11.8x	463	100
	Sig	427	17.1	6.9x	11.4x	463	100
Arm1 (7,20) C. Study	RBF	443	14.1	9.1x	11.3x	1818	100
Arm2 (7,20) C. Study	RBF	572	14.1	8.9x	12.0x	1818	100
Arm3 (7,20) C. Study	RBF	435	14.0	8.6x	11.4x	1818	100

Prediction Results

- Very few Prediction Mismatches
  - Examined data, determined Single/Double FP Precision issue
  - Double Precision could be implemented
    - Worthwhile?
- FPGA Timing More Consistent than CPU

Conclusions and Future Work

- R^2SVM FPGA Prototype Demonstrated Benefits of Architecture
  - Scalability
  - Compatibility with libsvm models
  - Presented performance against CPU/GPU for several real-world datasets
    - Up to 53x faster than CPU
    - Up to 23x faster than GPU
- Future Work
  - Kernel Hardware Alternatives?
  - ASIC optimization?

Questions?

References

[1] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[2] K. Sopyła, P. Drozda, P. Górecki, “SVM with CUDA Accelerated Kernels for Big Sparse Problems,” in *Proc. of the 11th Int. Conf. on Artificial Intelligence and Soft Computing*, 2012, pp. 439-437.

[3] J. Faella, “On Performance of GPU and DSP Architectures for Computationally Intensive Applications,” M.S. thesis, Dept. Elec. Eng., Univ. of RI, Kingston, RI, 2013.