Accelerating Neuromorphic Vision Algorithms for Recognition

Ahmed Al Maashri* Michael DeBole† Matthew Cotter* Nandhini Chandramoorthy*
Yang Xiao* Vijaykrishnan Narayanan* Chaitali Chakrabarti‡
*Microsystems Design Lab, The Pennsylvania State University
{maashri, mjcotter, nic5090, yux106, vijay}@cse.psu.edu
†IBM System and Technology Group
mvdebole@us.ibm.com
‡School of Electrical, Computer and Energy Engineering, Arizona State University
chaitali@asu.edu

ABSTRACT
Video analytics introduce new levels of intelligence to automated scene understanding. Neuromorphic algorithms, such as HMAX, are proposed as robust and accurate algorithms that mimic the processing in the visual cortex of the brain. HMAX, for instance, is a versatile algorithm that can be repurposed to target several visual recognition applications. This paper presents the design and evaluation of hardware accelerators for extracting visual features for universal recognition. The recognition applications include object recognition, face identification, facial expression recognition, and action recognition. These accelerators were validated on a multi-FPGA platform and significant performance enhancement and power efficiencies were demonstrated when compared to CMP and GPU platforms. Results demonstrate as much as 7.6X speedup and 12.8X more power-efficient performance when compared to those platforms.

Categories and Subject Descriptors
C.3 [**SPECIAL-PURPOSE AND APPLICATION-BASED SYSTEMS**]: Signal processing systems

General Terms
Design, Experimentation, Performance

Keywords
Recognition, Domain-Specific Acceleration, Heterogeneous System, Power Efficiency

1. INTRODUCTION
The visual cortex of the mammalian brain is remarkable in its processing and general recognition capabilities providing inspiration for complex, power-efficient systems and architectures. Researchers [1,2,3] have made advances in understanding the processing that occurs in the visual cortex. These advances have a profound impact on a range of application domains used for image recognition tasks such as surveillance, business analytics, and the study of cell migration.

DAC 2012, June 3-7, 2012, San Francisco, California, USA.

As a step towards exploring how the brain efficiently processes visual information, a brain-inspired feed-forward hierarchical model (HMAX) [2] has become a widely accepted abstract representation of the visual cortex. HMAX models are mainly implemented on general-purpose processors (CPUs) and graphics processing units (GPUs), which do not attain the power and computational efficiency that can be achieved by custom hardware implementations [3,4,5,6]. To achieve the performance, power, and flexibility to support computations used in neuromorphic applications, the ideal platform is the heterogeneous integration of domain-specific accelerators with general-purpose processor architectures.

This paper proposes a neuromorphic system, based on HMAX, for universal recognition. The system is composed of customized hardware accelerators that target four applications; namely, object recognition, face identification, facial expression recognition, and action recognition. This neuromorphic system is evaluated on a prototype heterogeneous CMP system composed of multi-FPGA system interfaced to quad-core Intel Xeon processor. The results indicate that the proposed architecture achieves recognition accuracies ranging from 70-90% across the four recognition algorithms. A detailed comparison of the power and performance with respect to HMAX variants executed on GPUs, multi-core CPUs, and FPGAs is performed. The results reveal that the proposed FPGA prototype provides frames-per-second(fps)-per-watt improvement as much as 12.8X over CPUs, and 9.7X over GPUs.

The rest of this paper is organized as follows; Section 2 provides an overview of HMAX model. Section 3 describes the micro-architecture of the accelerators developed for HMAX, while Section 4 discusses the evaluation platforms and presents results from the FPGA-based emulation system. Section 5 presents Related Work. Finally, Section 6 concludes the paper.

2. HMAX COMPUTATIONAL MODEL
Figure 1 shows a computational template of HMAX [2,3,5]. The model primarily consists of two distinct types of computations: convolution and pooling (non-linear subsampling), corresponding to the Simple, S, and Complex, C, cell types, respectively. The first S-layer (S_1) is comprised of fixed, simple-tuning cells, represented by oriented Gabor filters. Following the S_1 layer, the remaining layers alternate between max-pooling layers and template-matching layers tuned by a dictionary encompassing patterns representative of the categorization task.

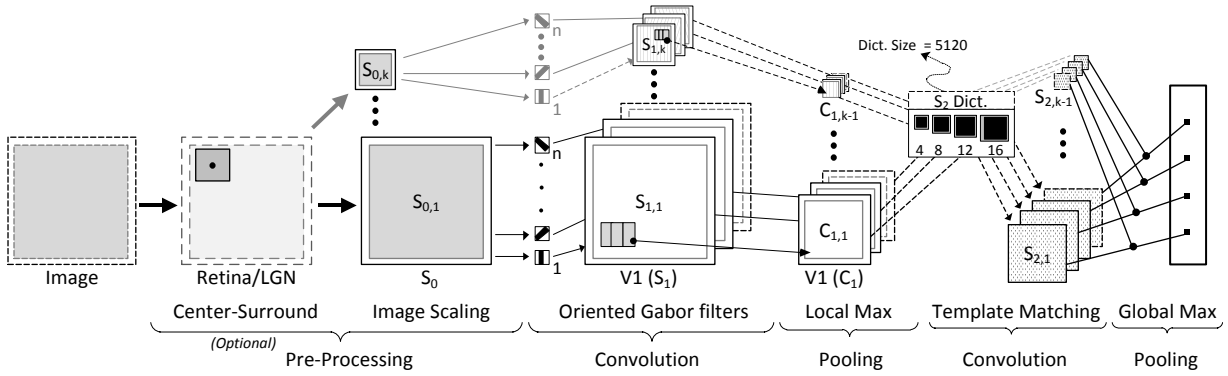

Figure 1. Typical Structure of HMAX Model: Center-Surround can be optionally applied to gray-scale input image. Image Scaling (S_0) produces *k* scales, forming a pyramid out of downsampled input image. A bank of Gabor filters ($S_{1,1}$ $S_{1,2}$ ... $S_{1,k}$) are used to detect edges at *n* orientations, while local maximum pooling ($C_{1,1}$ $C_{1,2}$ ... $C_{1,k-1}$) finds the maximum response over the S_1 output. Template matching is performed in S_2. Finally, C_2 performs global max operation producing a feature vector.

2.1 HMAX for Object Recognition

Our work uses a specific implementation for the object recognition task developed by [7], as it represents the current understanding of the ventral stream and produces good results when used for classification. This model is represented by a total of five layers: an image layer and four layers corresponding to the alternating S and C units.

Image layer: In this layer, the image is converted to grayscale and then downsampled to create an image pyramid of 12 scales, with the largest scale being 256*x*256.

S_1 (Gabor filter) layer: The S_1 layer corresponds to the V1 [2] simple cells in the ventral stream and is computed by performing a convolution with the full range of orientations at each position and scale. The number of orientations used in this model is 12, producing 12 outputs per scale for a total of 144 outputs.

C_1 (Local invariance) layer: This layer provides a model for the V1 complex cells and pools over nearby S_1 units with the same orientation. Within a scale, each orientation is convolved with a 3D max filter of size 10*x*10*x*2 (10*x*10 units in position and 2 in scale). This layer provides scale invariance over large local regions.

S_2 (Tuned features) layer: This layer models the V4 [2] by matching a set of prototypes against C_1 output. These prototypes have been randomly sampled from a set of representative images.

C_2 (Global invariance) layer: This layer provides global invariance by taking the maximum response from each of the templates across the scales. The layer removes all position and scale information, leaving only a complex feature set which can then be used later for classification.

2.2 Extensions of the HMAX Model

Neuroscientists have observed that the primates' visual system often shares a general, early-level processing structure, which eventually branches off into more specific higher-level representations. This serves as a motivation to configure the HMAX model to implement a variety of recognition problems beyond object classification.

2.2.1 HMAX for Face Processing

In order to support face identification and facial expression recognition, Meyers et al. [4] add a center-surround stage of processing to model the 'center-on surround-off' processing that is present in the retinal and LGN of the thalamus. In addition, the model does not perform S_2 and C_2 stages in order to maintain visual features localized to a particular region in space.

LGN/Retinal (Center-Surround) Layer: The center-surround is computed prior to pyramid generation and helps to eliminate intensity gradients due to shadows. The processing is done by placing a 2D window at each position in the input image that is identical in size to the filter used for S_1. The output is then computed by dividing the current pixel's intensity by the mean of the pixel intensities within the window.

2.2.2 HMAX for Action Recognition

While HMAX was originally limited to model the ventral stream, a model of the dorsal stream is useful for analyzing motion information. Jhuang et al. [6] have proposed augmenting the original HMAX model with the dorsal path as it can then be applicable to motion-recognition tasks, such as identifying actions in a video sequence. Computationally, this is done by integrating spatio-temporal detectors into S_1, while adding two additional layers, S_3 and C_3, which track features over time, providing time-invariance to the structure.

Space-Time Oriented S_1 (Gabor filter) layer: S_1 units for motion are extended by adding a third temporal dimension to the receptive fields. Computationally, this layer becomes an $n \times 2D$ convolution across a sliding window of past, present, and future frames, where *n* is total number of frames.

Space-Time Oriented S_3/C_3 (Tuning/Pooling) layers: S_3 unit responses are obtained by temporally matching the output of C_2 features to a dictionary, similar to S_2, where each patch represents a sampled sequence of frames. C_3 unit responses are the maximum response over the duration of a video sequence.

Based on these observations, the design of an accelerator based on the HMAX model should also contain additional hardware accelerators designed for spatio-temporal detection, retinal (center-surround) processing, and time-invariance operations, with an option for preserving localized spatial features.

3. HMAX SYSTEM ARCHITECTURE

The proposed HMAX accelerators are interconnected using a communication infrastructure that provides a number of features, including: high-bandwidth communication, run-time re-configurability and inter-accelerator message-passing for synchronization. The infrastructure accepts two types of

accelerators; namely, stream-based and compute-based. Stream-based accelerators are more suited for on-the-fly processing of streaming data, while the compute-based accelerators are more suitable for iterative processing on non-contiguous blocks of data. Due to space limitation, this paper does not discuss the details of this infrastructure. A more in-depth treatment of the infrastructure can be found in [8].

3.1 HMAX Accelerators

Table 1 lists the HMAX accelerators and their functions. In this paper, we focus on the S_2/C_2 accelerator since this is the most time consuming stage in the HMAX model.

The S_2/C_2 accelerator combines the S_2 and C_2 stages into a single accelerator, which allows pooling to occur immediately following the computation of a current S_2 feature. Also, this can effectively decrease the amount of data required to be sent across the network by $\sum_{s=0}^{S-1}(N_{prototypes}[X_s, Y_s]) \cdot (X_s \cdot Y_s)/N_{prototypes} * 2$. Here, S is the number of scales at the S_2 stage, X_S (Y_S) is the dimension of scale S in the x-direction (y-direction). Practically, with 5120 prototypes using 12 scales and 12 orientations, this reduces the data transferred by 4,135X.

The accelerator consists of one or more instances of systolic 2D filters, which are the most computationally demanding components of the S_2 unit. These filters are designed to enable data reuse and take advantage of available parallel computational resources. Additionally, the accelerator makes efficient use of available memory hierarchies to improve performance. For instance, the per-scale outputs of the 2D filters in the convolver must be accumulated across all orientations. In order to avoid the network traffic associated with buffering these results in an external network-attached memory, the accelerator uses a scratchpad memory to store and accumulate these outputs immediately as they are produced. This results in increased performance of up to nX, where n is the number of orientations. However, scratchpad memories are not suitable for large volumes of data. As an example, in this implementation of HMAX for object recognition, the S_2 uses 5120 prototypes which require approximately 24 MB of storage. A more suitable storage solution is the use of off-chip memory. While the communication infrastructure supports network-attached memories, the large increase in network traffic will degrade performance. Instead, the S_2/C_2 accelerator integrates an optimized memory controller that interfaces directly to an off-chip memory.

Table 1. HMAX accelerators and their functions

Stream-based	Function(s)
Downsampling (DS)	Generates multiple scales by subsampling input
Normalization (Norm)	Computes windowed average for normalizing S_2 output
	Computes center-surround
S_1	Streaming 2D convolution
S_3	Streaming 1D convolution
Compute-based	**Function (s)**
C_1	Windowed pooling
S_2/C_2	Prototype correlation and global max
C_3	Global max operation

3.2 HMAX Data Processing Flow

The HMAX accelerators, listed in Table 1, are interconnected using a communication infrastructure [8]. This acceleration system can be loosely coupled to a CMP within a heterogeneous system, as illustrated in Figure 2. In this heterogeneous system, the processor, executing the main application, offloads the entire HMAX computation to the accelerators. However, prior to offloading the computation, the CMPs will configure these accelerators to reflect the current structure of the model (e.g. number of orientations, scales, pooling window sizes, etc…). Figure 2 also demonstrates an example application, action recognition, executed on the accelerators. First, the processor copies the data to the image buffer memory, *Img_Mem*, and a notification message to C_1 through the interface. C_1 performs several reads from the image buffer for every pooled output through *flow0*, calculating the proper scale and S_1 tuned cells each time, on the fly. As each pooled output is computed, it is written to the S_2/C_2 accelerator unit through the Normalization, Norm, accelerator using *flow1*. C_1 then messages S_2/C_2 notifying the latter that all scales have been written and initiating the computation of the S_2/C_2 tuned and pooled cells. Once that is completed, the S_2/C_2 writes the outputs to the C_3 unit, which calculates the S_3 tuned output during data-movement using *flow2*. Finally, when C_3 has completed the computations; the output is returned via the interface to the invoking processor. Similarly, other recognition applications are executed according to their computational structure.

4. EXPERIMENTAL EVALUATION

To evaluate the neuromorphic accelerators, we use a Multi-FPGA system as an emulation platform. Also, the four vision application domains (i.e. object recognition, face identification, facial expression recognition and action recognition) were implemented on multi-core CPU and GPU platforms for performance and accuracy comparisons. The following subsections discuss the datasets used for testing the evaluated platforms and provide a comparative analysis of the performance of each platform.

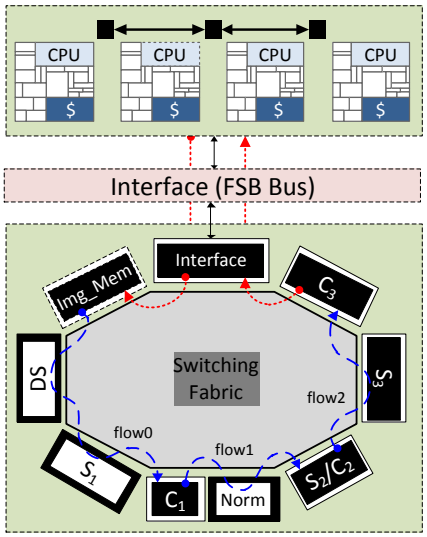

Figure 2: A heterogeneous system showing the interactions between the CMP and neuromorphic accelerators. The neuromorphic accelerators require access to three memory banks used to store the feature dictionaries used for S_2 and S_3 (not shown) and intermediate frames required for action recognition (*Img_Mem*).

4.1 Datasets for Evaluation

Table 2 shows the datasets used for evaluating the recognition accuracy of the neuromorphic accelerators. The Caltech 101 dataset [9] is used to test the accuracy of object classification using all 102 different categories. The *vehicles* dataset, prepared in-house, consists of 16 categories with a variety of objects including vehicles, aircrafts, military equipment and background scenery. The ORL dataset [10], used for testing face identification accuracy, contains a collection of close-up images of the faces of 40 different individuals from varying viewing angles. The FERET dataset [11] includes 1208 individuals from which a random subset of 10 individuals was chosen for evaluation. The JAFFE dataset [12] is used for testing the accuracy of facial expression recognition. Six different expressions were tested—anger, disgust, fear, happiness, sadness, and surprise. Finally, the Weizmann dataset [13] is used for testing human action recognition. This dataset includes 10 different categories of actions: bending, jumping jacks, vertical jumping, horizontal jumping, skipping, running, side-stepping, walking, one-hand-waving, and two-hand-waving.

4.2 Experimental Setup

The accelerators were prototyped on a Multi-FPGA platform that mimics a heterogeneous multi-core system. The platform contains a quad-core Xeon processor running at 1.6 GHz interfaced to an FPGA acceleration system through a Front-Side Bus, FSB. The FPGA system contains four Virtex-5 SX-240T FPGAs [14], all operating at 100 MHz. The quad-core processor is mainly used to transfer data to the accelerators through the FSB and to retrieve the output.

The reference CPU platforms contained 4- and 12-Core Xeon CPU systems with the total number of threads executed on each shown in Figure 3. The 4-Core CPU is clocked at 1.6 GHz, while the 12-Core CPU is clocked at 2.4 GHz. All CPU platforms utilized SSE instruction set extension. The 12-core Xeon processor configuration with 12 threads serves as the CPU reference when comparing implementations as it provides the best performance across CPU platforms and configurations. The GPU platform consists of an Nvidia Tesla M2090 board [15], which houses three 1.3 GHz Tesla T20A GPUs, and using CUDA as the programming language [5].

4.3 Performance

Accuracy: The fifth column in Table 2 shows the classification accuracy across all datasets using the feature vector produced by the accelerated HMAX. The recognition accuracy across all the platforms was similar; however, since accelerators use 32-bit fixed-point representation (1 bit for sign, 7 for integer and 24 for fraction), a slight degradation in accuracy was observed (i.e. ≤ 2%). This degradation is due to the truncation of the fixed-point representation during the multiply-accumulate operation.

Speed: We use frames (segments) processed per second (fps) as a metric to compare the speedup gained by each platform. In this paper, we use the term "segment" to refer to a group of 20 frames extracted from a video sequence for action recognition application. **Figure 4** shows a speedup comparison between the three platforms in terms of fps for the four recognition applications. The FPGA prototyping platform demonstrates a speedup ranging from 3.5X to 7.6X (1.5X to 4.3X) when compared to the CPU (GPU) platform. The FPGA platform exhibits increased performance improvement in the action recognition application. This is due to the cumulative effect of per-frame performance of the S_1 stage. Since each video segment consists of 20 frames, the FPGA accelerator sees a linear increase in performance with the number of frames.

Power Efficiency: **Figure 5** compares the power efficiency (fps/Watt) across the three platforms. For the GPU, the command tool 'nvidia-smi -q' is used to probe the power consumption from a power sensor found on the GPU board. For the CPU and FPGA platforms, power consumption was measured using a power meter. The meter provides continuous and instantaneous reading of power drawn by the platform with 99.8% accuracy.

The power consumption for all platforms is measured only after the platform reaches steady-state to obtain the baseline power measurement. Then, the workload is executed and peak power is measured throughout the duration of the workload execution. For example, the power measurements show that when running HMAX for object recognition, the GPU, CPU and FPGA platforms consume 144, 116 and 69 Watts, respectively. Using these measurements, the power efficiency of each platform is computed as shown in **Figure 5**. The results show that the HMAX accelerators demonstrate a significant performance-per-watt benefit, ranging from 5.3X to 12.8X (3.1X to 9.7X) when compared to CPU (GPU) platform.

Configurability/Tradeoffs: It is often desirable to trade off accuracy for higher performance. We performed further experimentation with the accelerated HMAX to analyze impact of reduced overall accuracy on the execution time. For example, we experimented with changing the number of orientations processed by the HMAX model from 12 to 4. Reducing the number of orientations improved speed by 2.2X, while producing only a 1.1% difference in accuracy for the *vehicles* dataset. In another experiment, the numbers of input scales was varied, while observing its influence on accuracy and speedup using the *vehicles* dataset. **Figure 6** (left) shows that as the number of scales decreases the classification accuracy decreases until it reaches ~70% when using 5 input scales. On the other hand, **Figure 6** (center, right), shows a consistent improvement in speedup and power efficiency as number of scales is decreased, effectively reaching 15.4X better speedup and power efficiency when using only 5 scales compared to 12-scale configuration. Permitting such trade-off analysis makes the proposed accelerator very suitable for studies in modeling refinements and vision algorithm tuning.

4.4 Discussion of Results

There are a number of factors that contribute to the increased performance of the accelerator-based system. First, the underlying framework provides up to 1.6 GB/s (3.2 GB/s) bandwidth when

Table 2: Datasets used for evaluation. Note that there is no overlap in training and testing samples.

Application Domain	Dataset	# Classes	# Test samples	Accuracy (%)
Object recognition	Caltech 101	102	4543	70
	vehicles	16	1382	83
Face ID	ORL	40	200	85
	FERET	10	60	70
Facial expr. recognition	JAFFE	6	60	86.7
Action recog.	Weizmann	10	40	77.7

clocked at 100 MHz (200 MHz), supporting high transfer rates across the network. Second, the parallelism exploited by the architecture is enabled by the large number of resources (multipliers, registers, etc) available on the FPGA. For example, this allows up to 256 multiply-and-add operations to be performed simultaneously providing a 256X increase in performance over sequential operation. Third, the accelerators implement customized processing pipelines, taking advantage of data reuse. Finally, the ability to instantiate multiple processing units of the same type (e.g. S_2/C_2 units), leverages task-level parallelism to the user.

Similarly, the power efficiency benefits are the result of customized, application-specific architectures that are able to process incoming data in fewer cycles (compared to CPU/GPU) at a lower frequency. The use of custom numerical representations also contributes to the performance gain. It should be noted that our FPGA was fabricated with an older 65nm technology, compared to 45nm and 40nm technologies used with CPU and GPU platforms, respectively. It is expected that implementing the neuromorphic accelerators in silicon rather than on an FPGA platform will accentuate such benefits. For instance, Kuon et al. [16] show that at 90nm fabrication process, moving from SRAM-based FPGA to CMOS ASIC architectures improves critical path delay by 3X – 4.8X, and dynamic power by 7.1X – 14X.

5. RELATED WORK

The effort demonstrated in this work is synergistic with recent efforts aimed at domain-specific computing with configurable accelerators [17,18,19,20,21,22]. In [17] the authors detail the implementation of a multi-object recognition processor on SoC. They present a biologically inspired neural perception engine that exploits analog-based mixed-mode circuits to reduce area and power. However, except for the visual attention engine and the vector matching processors, all other algorithm acceleration is performed on multiple SIMD processors executing software kernels. Tsai et al. [18] propose a neocortical computing processor interconnected with high-bandwidth NoC. The processor consists of 36 cores; each contains multiple processing elements for performing the actual computations. Unlike the accelerators proposed in this paper, these processing elements are generic and not customized for any specific stage in the HMAX model. Other works [19,22] have proposed an architecture for image processing using Convolutional Neural Networks, CNN. These architectures can configure and train the neural network to support a variety of recognition algorithms. The convolution engine forms the critical component of these accelerators. While the authors in [19,22] indicate that mapping HMAX models to CNN structures is straightforward, the authors do not describe modifications necessary to implement large convolution windows or the *n*-dimensional convolutions that are required in the S_2 layer.

6. CONCLUSIONS

This work proposed reconfigurable neuromorphic accelerators for universal recognition that can be fabricated within a heterogeneous system. An FPGA prototyping platform is implemented as an emulation of the heterogeneous system. The prototyping platform exhibits a remarkable performance gain of up to 7.6X (4.3X) compared to the CPU (GPU). Moreover, this prototyping platform shows a superior power efficiency of 12.8X (9.7X) compared to the CPU (GPU) platform.

7. ACKNOWLEDGMENTS

This work was funded in part by an award from the Intel Science and Technology Center on Embedded Computing, NSF Awards 1147388, 0916887, 0903432, 0829607.

8. REFERENCES

[1] L. Itti, C. Koch, and E. Niebur, "A Model of Saliency-Based Visual Attention for Rapid Scene Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254-1259, Nov 1998.

[2] M. Riesenhuber and T. Poggio, "Hierarchical Models of Object Recognition in Cortex," *Nature Neuroscience*, vol. 2, no. 11, pp. 1019-1025, November 1999.

[3] T. Serre et al., "Robust Object Recognition With Cortex-Like Mechanisms," *IEEE PAMI*, vol. 29, no. 3, March 2007.

[4] E. Meyers and L. Wolf, "Using Biologically Inspired Features for Face Processing," *International Journal of Computer Vision*, vol. 76, no. 1, pp. 93-104, January 2008.

[5] J Mutch, U Knoblich, and T Poggio, "CNS: A GPU-Based Framework for Simulating Cortically-Organized Networks," Massachusetts Institute of Technology, Cambridge, MA, MIT-CSAIL-TR-2010-013 / CBCL-286 2010.

[6] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A Biologically Inspired System for Action Recognition," in *International Conference on Computer Vision (ICCV)*, 2007, pp. 1-8.

[7] J. Mutch and D. G. Lowe, "Object Class Recognition and Localization Using Sparse Features with Limited Receptive Fields," *International Journal of Computer Vision (IJCV)*, vol. 80, no. 1, October 2008.

[8] S. Park et al., "System-On-Chip for Biologically Inspired Vision Applications," *Information Processing Society of Japan*, 2012, [In Press].

[9] L. Fei-Fei et al., "Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Tested on 101 Object Categories," in *IEEE CVPR 2004, Workshop on Generative-Model Based Vision*, 2004.

[10] F. Samaria and A. Harter, "Parameterisation of a Stochastic Model for Human Face Identification," in *2nd IEEE Workshop on Applications of Computer Vision*, 1994.

[11] P. J. Phillips et al., "The FERET Evaluation Methodology for Face Recognition Algorithms," *Trans. of Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, October 2000.

[12] M. Lyons et al., "Coding Facial Expressions with Gabor Wavelets," in *Third IEEE International Conference on Autmatomatic Face and Gesture Recognition*, 1998.

[13] M. Blank et al., "Actions as Space-Time Shapes," in *International Conference on Computer Vision*, 2005.

[14] Xilinx, "Virtex-5 Family Overview," DS100(v5.0) 2009.

[15] Nvidia. (2011) Tesla M2090 Board Specification. [Online]. http://www.nvidia.com/docs/IO/43395/Tesla-M2090-Board-Specification.pdf

[16] I. Kuon and J. Rose, "Measuring the Gap Between FPGAs and ASICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 203-

215, February 2007.

[17] J.-Y. Kim et al., "A 201.4 GOPS 496 mW Real-Time Multi-Object Recognition Processor With Bio-Inspired Neural Perception Engine," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 1, pp. 32-45, Jan 2010.

[18] C.-Y. Tsai et al., "A 1.0TOPS/W 36-Core Neocortical Computing Processor with 2.3Tb/s Kautz NoC for Universal Visual Recognition," in *IEEE Int. Conference Digest of Technical Papers*, San Francisco, 2012, pp. 480-482.

[19] C. Farabet et al., "Hardware Accelerated Convolutional Neural Networks for Synthetic Vision Systems," in *ISCAS*, Paris, 2010, pp. 257-260.

[20] R. Iyer et al., "CogniServe: Heterogeneous Server Architecture for Large-Scale Recognition," *IEEE Micro*, vol. 31, no. 3, pp. 20-31, May-June 2011.

[21] J. Clemons et al., "EFFEX: An Embedded Processor for Computer Vision Based Feature Extraction," in *The 48th ACM/EDAC/IEEE DAC*, San Diego, 2011, pp. 1020-1024.

[22] S. Chakradhar et al., "A Dynamically Configurable Coprocessor for Convolutional Neural Networks," in *ISCA*, 2010, pp. 247-257.

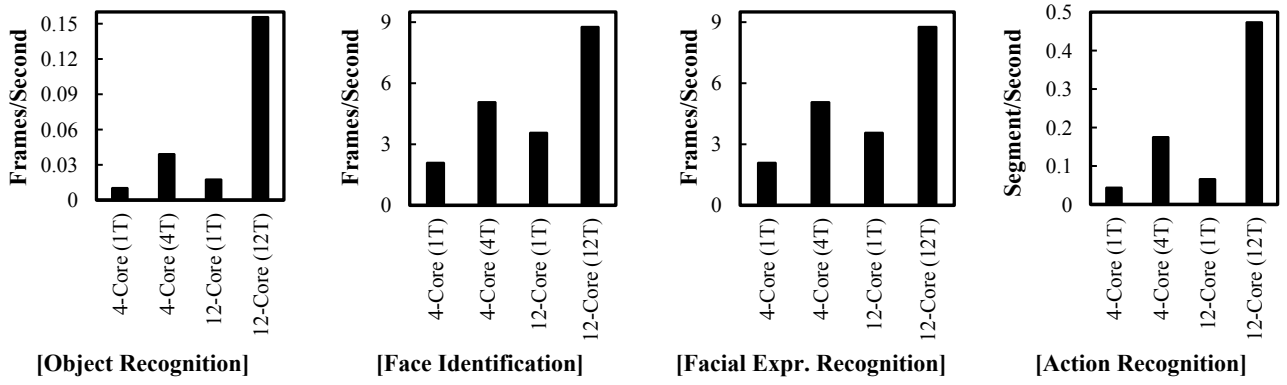

Figure 3: Performance of reference CPU platforms across four application domains. Number of threads is indicated within brackets on the x-axis. The metric used to measure performance is frames/second (segment/second) for the first three applications (action recognition application). Segment in this context refers to a group of twenty frames.

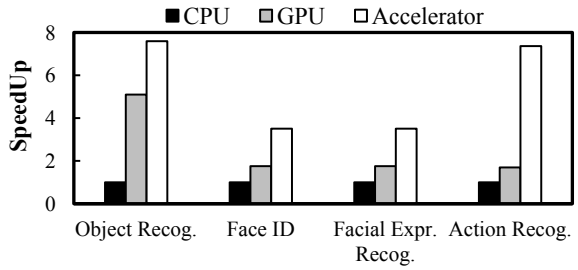

Figure 4: Speedup (FPS): A comparison across the three platforms for each application domain. Figures are normalized to the CPU platform

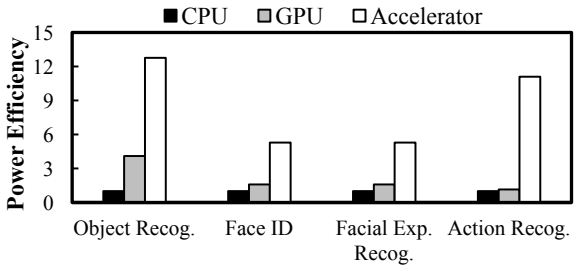

Figure 5: Power Efficiency (fps/Watt): A comparison across the three platforms for each application domain. Figures are normalized to the CPU platform

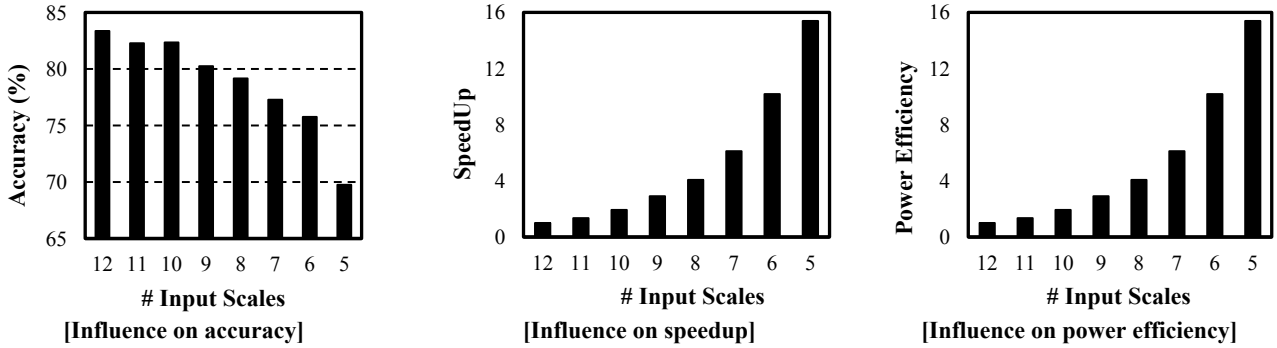

Figure 6: The influence of number of scales on classification accuracy and performance. As the number of input scales decreases, the classification accuracy decreases and power efficiency increases. Figures in "Speedup" & "Power Efficiency" are normalized to the 12-input-scale configuration