Visual Cortex on the GPU: Biologically Inspired Classifier and Feature Descriptor for Rapid Recognition

Kris Woodbeck, Gerhard Roth
School of Information Technology and Engineering
University of Ottawa
Ottawa, Ontario, Canada
kris.woodbeck@alumni.uottawa.ca, gerhardroth@rogers.com

Huiqiong Chen
Faculty of Computer Science
Dalhousie University
Halifax, Nova Scotia, Canada
huiqiong@cs.dal.ca

Abstract

We present a biologically motivated classifier and feature descriptors that are designed for execution on Single Instruction Multi Data hardware and are applied to high speed multiclass object recognition. Our feature extractor uses a cellular tuning approach to select the optimal Gabor filters to process a given input, followed by the computation of scale and rotation-invariant features that are sparsified with a lateral inhibition mechanism. Neighboring features are pooled into feature hierarchies whose resonant properties are used to select the most representative hierarchies for each object class. The feature hierarchies are used to train a novel form of Adaptive Resonance Theory classifier for multiclass object recognition. Our model has unprecedented biologically plausibility at all stages and uses the programmable Graphics Processing Unit (GPU) for high speed feature extraction and object classification. We demonstrate the speedup achieved with the use of the GPU and test our model on the Caltech 101 and 15 Scene datasets, where our system achieves state-of-the-art performance.

1. Introduction

Computer vision systems have become quite adept at recognizing objects that they have already seen, but the more general task of recognizing a class from a limited number of instances has proven somewhat elusive. Since the primate visual system outperforms all existing computer vision systems at recognition, it is reasonable to look towards biology for inspiration. Biologically inspired work [21, 17, 26] has produced a number of models that compete with the top computer vision systems at object recognition. These models aim to replicate the lowest levels of the visual cortex and generally place a great deal of focus on the computation of visual features. Much less focus has been placed on developing a biologically plausible classifier. Existing models [19, 21, 17] have been implemented on Single Instruction Single Data (SISD) processors, but given the highly parallel nature of the brain, a more parallel processor is required to accurately model the processing in the visual cortex. In recent years, the graphics industry has developed general-purpose, highly parallel programmable Graphics Processing Units (GPU). In our work, we show that the GPU is better suited to the task of modeling the visual cortex and clearly outperforms its SISD counterpart, the CPU, at biologically motivated processing.

The lowest areas of the visual cortex are retinotopic [13], meaning that adjacent cells process adjacent regions of the visual field with identical operations in a Single Instruction Multi Data (SIMD) manner. At the lowest areas, the cells perform a convolution operation, which continues onto various forms of cellular response pooling at higher levels. Most object classes are too complex to be recognized using only small, isolated regions of an image. Our model mimics the visual cortex by gradually pooling simpler features at lower layers into more complex features at higher layers [19]. The features used in our model take heavy cues from visual features indicated by cells in the cortex [14, 3].

Our model uses a novel filter parameter tuning strategy to select the optimal cells with which to process each visual field. This is followed by feature extraction which uses three novel feature descriptors and comparison metrics of gradually increasing complexity. We present a biologically motivated classifier based on Adaptive Resonance Theory [5] for object recognition that is implemented as a series of SIMD processing states on the GPU.

Our model has rapid recognition capabilities; it is able to perform feature extraction and object classification in 270 milliseconds on current hardware using over 250,000 feature descriptors in the classifier. It is tested on the Caltech 101 [8] and 15 Scene [16] datasets; the results are comparable to the state-of-the-art and surpass all other biologically motivated models. We give timing results which show the speedup obtained by our system. Our results make a strong case both for using the GPU and biologically inspired methods for recognition.

Related Work Due to the inherent incompatibility between most SISD and SIMD algorithms, our model could not be built from any existing models. Our model uses a similar methodology to existing models of the visual cortex, including the HMAX or 'Standard Model' [19], the 'Standard Model 2.0' [21] and later derived work [17]. In turn, these models have similarities with previous work, such as the Neocognitron [9]. All models start with V1 simple and complex cells, as discovered by Hubel and Wiesel [15]. Simple cells perform convolution operations with Gabor filters. Complex cells pool simple cell responses at a given orientation using a MAX operator over all scales and positions in the cell's receptive field. Simple and complex cells are alternated and their receptive fields gradually grow in size. Recent biological models [21, 17, 26] have used (non-biologically motivated) Support Vector Machines (SVM) for classification. Non-biological approaches comprised of hierarchies of Gabor filters have also yielded some success at object recognition [18]. Our model distinguishes itself from previous models [19, 21, 17] in the following ways:

- our feature descriptors have rotation, scale and translation invariant properties,
- we use a filter tuning approach is used instead of static filterbanks,
- an inhibition mechanism is employed to reduce feature redundancy, as in other work [17], and
- a novel multiclass classifier, based on Adaptive Resonance Theory, is used for recognition.

2. Physiological Feature Descriptors

Our model consists of several classes of feature descriptors which progressively increase in complexity and are used in combination with a biologically-motivated classifier. Figure 1 gives an overview of the process used to isolate features and classify an image. Our descriptors are built around key anatomical structures, the first of which are simple and complex cells, as used in other models [19, 21, 17]. We introduce the physiological cortical column structure [10]. This structure serves to group simple cells that vary only by orientation; it is used both to regulate our cellular tuning process and to aid in feature normalization. The hypercomplex cell [15] is introduced, which serves to pool complex cell features into higher order features that are both rotation and scale-invariant. We introduce a novel type of cell that pools hypercomplex cell features and links them in a hierarchical manner. We define comparison metrics for each descriptor and give a method of isolating the pertinent features for a given object class. Classification is done on the GPU using a biologically motivated classifier, derived from Adaptive Resonance Theory (ART) [4]. Note that all cellular data, meaning all system input and output, is stored in texture format on the GPU during all stages of processing.

V1 Simple Cells (V1S Layer) V1 simple cells perform a convolution on the visual field using Gabor filters over a range of orientation. We use four orientations, similar to other models [21, 17]. A Gabor filter is described by:

$$C_s(x,y) = \exp\left(-\frac{x_1^2+\gamma^2 y_1^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x_1}{\lambda}+\psi\right) \quad (1)$$

where $x_1 = x\cos\theta + y\sin\theta$ and $y_1 = y\cos\theta - x\sin\theta$. σ and λ are related by the bandwidth parameter b from Equation 2.

$$\frac{\sigma}{\lambda} = \frac{1}{\pi}\sqrt{\frac{\ln 2}{2}}\frac{2^b+1}{2^b-1} \quad (2)$$

The system uses parameters within the ranges shown in Table 1. The initial values of the parameters are: $\psi = 0.0, \sigma = 0.7, \gamma = 0.6, b = 1.1$, but the final filter values are obtained with the use of a tuning operation which dynamically selects the optimal simple cell parameters for each image. This procedure is explained below.

V1 Complex Cells (V1C Layer) V1 complex cells perform a pooling operation over simple cells within the complex cell's receptive field, σ_{RF}. The pooling is modeled with a MAX operation [19, 21, 17, 26]. Physiologically, most complex cells pool the responses of simple cells with a similar preferred orientation θ. Let $C_s(s_i,\theta,x,y)$ be a simple cell's response at (x,y) with orientation θ and scale s_i. The complex cell pooling operation is defined as:

$$\begin{aligned} C_c(\theta,x,y) = \max\{C_s(s_i,\theta,x+x',y+y') \ : \\ \forall(x',y')\in\sigma_{RF}, \forall s_i \in [5,31]\} \end{aligned} \quad (3)$$

Complex cells effectively perform a sub-sampling operation: a complex cell with a given receptive field σ_{RF} reduces a group of simple cells within its receptive field to a single cell. Note that the feature derived from a complex cell, C_c^i, has an orientation value α_i which comes from the orientation of the object in the visual field activating the underlying simple cell(s). α_i is closely related to the simple cell θ parameter. The α_i value is used to achieve rotation and scale invariance, it is further detailed in the V1HC layer.

Parameter	$scale$	θ	ψ	b	γ	λ
Value Range	$[5, 31]$	$0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}$	$[-\frac{\pi}{2}, \frac{\pi}{2}]$	$[0.2, 3.0]$	$[0.0, 3.0]$	$[0.0, 6.0]$

Table 1: Gabor parameter ranges used during tuning; all parameters begin at a default value and are tuned within this range.

Cellular Tuning Selecting the best simple cells with which to process a given visual field is not trivial: the visual cortex has a wide range of simple cells available for visual processing. Other models [19, 21, 17] use static sets of Gabor filters for simple cells, but our model does away with this assumption. Simple cell parameters are selected using an iterative tuning model that serves to mimic intra-columnar communication in the cortex [10]. The results of this tuning process are an optimized set of simple cells with which to process the visual field. The tuning process is summarized as follows:

1. The cortex is initialized with default simple cell values
2. Tuning takes place over the following V1 simple cell parameters: γ, ψ, b, λ
3. There are N tuning steps, with M simple cell settings tested at each step
4. At each tuning step, a new series of simple cells are generated by altering the current simple cell values in a biologically plausible manner
5. All M newly generated simple cells are applied to the visual field
6. The results from the M simple cells are evaluated based on intra-columnar (corner) and inter-columnar (edge) activations
7. The parameters that create the optimal ratio of corners to edges while maximizing corners are selected as the winning parameters
8. The tuning process is repeated N times

The full details of the tuning process are covered in [26]. A number of physiological properties are taken into account when generating parameters during the tuning process, such as an increase of cellular receptive field size and, generally speaking, similar cellular setting ranges as in [21]. Due to the fact that $4NM$ total convolutions are performed per visual field, the tuning is only plausible if done using hardware capable of SIMD computation, such as the GPU.

V2 Hypercomplex Cells (V2HC Layer) V2 has received much less focus from the neuroscientific research community than V1; electrophysiological studies are nearly 10 times more common for V1 than V2 [3]. V2 is known to activate in response to illusory contours of objects [22], it has a complex contrast ratio response and activates for relatively complex shapes and textures [14].

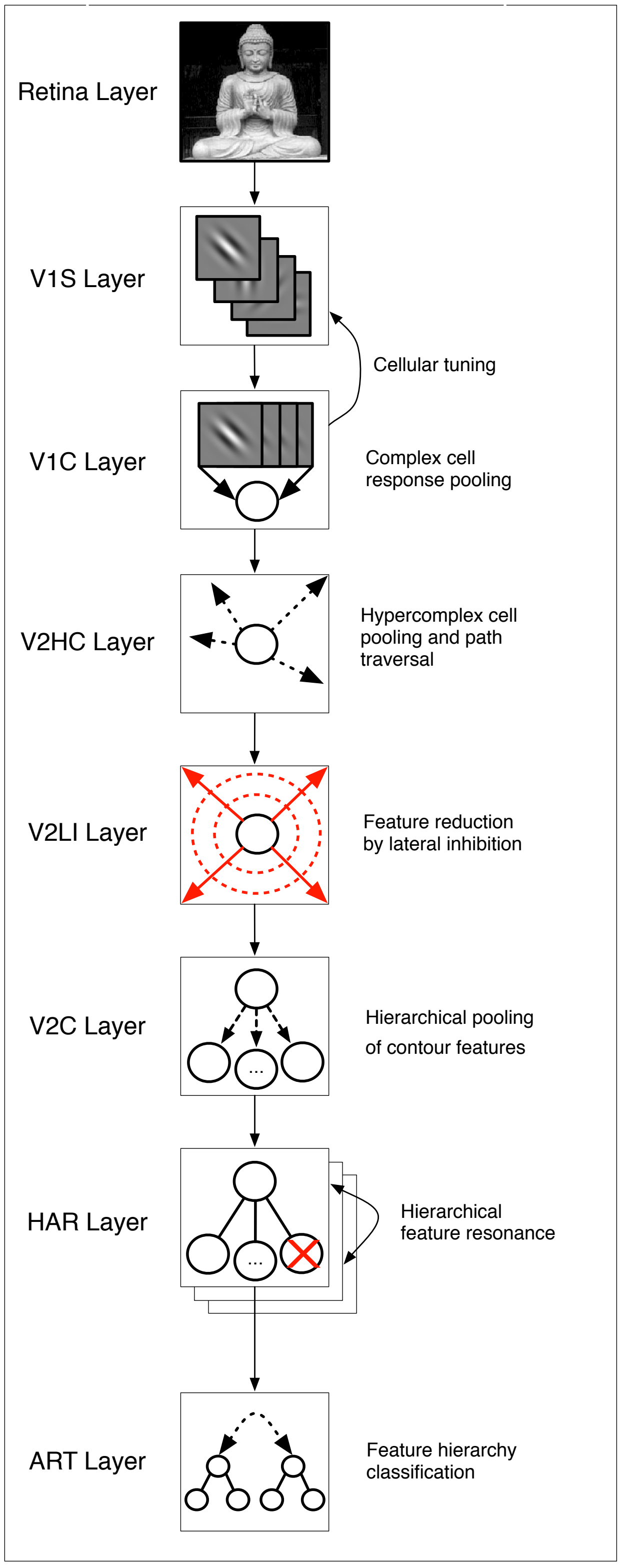
While a complex cell will pool simple cells at a common orientation, cells in V2 are known to pool cells at distinct orientations [1, 3]. To model this, we introduce the hypercomplex cell [15]. Hypercomplex cells are known to pool over complex cell orientation, they respond to end-stopped input (bars of a specific length) and also respond to stimuli with either direction-specific or general motion. Since our model's recognition component does not require motion, we use only end-stopped and orientation pooling hypercomplex cells. These properties allow hypercomplex cells to activate for simple shapes, such as that of a chevron or star. An orientation pooling hypercomplex cell, comprised of all active complex cells in its receptive field, is defined as follows:

$$C_h(x, y) = \{C_c(\theta_i, x, y) : \\ \theta_i \in 2\pi, C_c(\theta_i, x, y) > t_{act}\} \quad (4)$$

Here, t_{act} is an activation threshold and is set to 10% of the maximum response value. A hypercomplex cell feature C_h^a is comprised of all active complex cell features $C_{c0}^a \dots C_{cN}^a$, which are arranged in a clockwise manner using the cortical column structure. Hypercomplex cell features have properties which allow them to be normalized in both rotation and scale-invariant manners.

Let C_{ci}^a be an active complex cell with a receptive field σ_a. Rotation invariance is achieved by traversing all previously pooled simple cells in σ_a to detect both end-stopped and non-end-stopped regions of activation. The orientation of the traversal path is α_i and reflects the orientation of the underlying object that is activating the complex cells. The orientation difference between any two complex cell features is $\alpha_{ij} = abs(\alpha_i - \alpha_j)$; examples of this can be seen in Figure 2. Rotation invariance is achieved by determining the largest α_{ij} value and setting C_{ci}^a as the primary complex cell. The remainder of the cells are arranged according to their order in the cortical column, nameley in a clockwise manner.

Scale invariance is achieved by normalized each feature with respect to the length of the longest complex cell feature within the cortical column. Eeach complex cell C_c has a length associated with its activation, dictated by σ_a, s_i and the traversal path. Once rotation and scale normaliza-



Retina Layer
V1S Layer
Cellular tuning
V1C Layer
Complex cell response pooling
V2HC Layer
Hypercomplex cell pooling and path traversal
V2LI Layer
Feature reduction by lateral inhibition
V2C Layer
Hierarchical pooling of contour features
...
HAR Layer
Hierarchical feature resonance
...
ART Layer
Feature hierarchy classification

Figure 1: An overview of the model; each state is a GPU-resident processing step. Results are sent to the CPU only after the HAR Layer.

tion have taken place, C_h^a is comprised of complex cells $C_{c0}^a \dots C_{cN}^a$ and has the following properties:

$$\alpha_{01} = \max_{i<N} abs(\alpha_i - \alpha_{i+1}) \tag{5}$$

$$\forall C_{ci} \in C_h(\alpha_i - \alpha_{i+1} < 0) \tag{6}$$

$$\forall C_{ci} \in C_h(len(C_{ci}) = \frac{len(C_{ci})}{\max len(C_{cj})}) \tag{7}$$

A metric for comparing two complex cell features (C_{ci} and C_{cj}) is given in Equation 8, and a similar metric for comparing two hypercomplex cells (C_h^i and C_h^j) is given in Equation 9.

$$||C_{ci} - C_{cj}|| = \sqrt{(\alpha_{ij})^2 + (len(C_{ci}) - len(C_{cj}))^2} \tag{8}$$

$$||C_h^i - C_h^j|| = \sqrt{\frac{\sum_{k=0}^{N} ||C_{ck}^i - C_{ck}^j||^2}{N}} \tag{9}$$

These metrics allow the comparison of both complex cell features and hypercomplex features. The complex cell metric is chosen for its simplicity, but we believe that a more biologically motivated Mahalanobis metric could be derived in the future. These metrics raise the important issue of how to separate foreground from background, since hypercomplex cell features will inevitably contain complex cell features with both foreground and background information. Physiologically, having two eyes aids us greatly in determining object boundaries: depth is a very obvious cue. But in most computer vision problems, we do not have two views of a given scene, so other techniques must be employed to isolate an object of interest from its background. This problem is the focus of the HAR layer.

V2 Lateral Inhibition (V2LI Layer) The visual field now consists of a set of hypercomplex features whose cells' receptive fields overlap one another, leaving redundant features in the visual field. Lateral inhibition is a neural mechanism whereby a given cell inhibits neighboring cells with lower activations. In our model, this inhibitory mechanism activates over the previously defined paths of complex cells. As in previous models [17, 18], lateral inhibition or feature sparsification has been shown to help to effectively isolate the important features within the visual field. The inhibitory mechanism in our model uses the receptive fields of hypercomplex cells to inhibit the response of neighboring hypercomplex cells. The inhibition mechanism acts as a MAX operator and is similar in nature to the V1 complex cell.

V2 Hierarchy Building (V2C Layer) Once lateral inhibition has taken place, the visual field consists of a collection of hypercomplex cells pooled from complex cells. The next step is to introduce higher order features by pooling the hypercomplex cells one step further. We define another type of cell that has not directly been observed within the visual cortex, but its properties are quite similar to cells in the lower levels: it is yet another type of pooling operation. This pooling is geared towards activation by object contours.

This new type of cell pools the response of hypercomplex cells to form a hierarchy of hypercomplex features. Let (C_h^a, C_h^b) be two hypercomplex cells that are linked together by a shared complex cell. Each hypercomplex cell has its constituent complex cells' receptive fields traversed to determine its hypercomplex cell neighbors. The end result is the resonant cell feature C_r^a:

$$\begin{aligned} C_r^a \;=\; & \{(C_h^a, C_h^k) \;:\; k \in [0, N] \wedge \\ & (\exists i, j, C_{ci}^a \in C_h^a, C_{cj}^k \in C_h^k) : \|C_{ci}^a - C_{cj}^k\| = 0\} \end{aligned} \tag{10}$$

A sample resonant feature can be seen in Figure 2. The pooling operation links 2-tuples of hypercomplex cell features together into a resonant feature. Resonant features are designed to activate in response to illusory contours of objects, but one problem remains: features contain a good deal of background information. The ideal resonant feature will activate solely to an object's contour, whose elements are generally linked with a similar luminance value, as in the visual cortex [22]. The isolation of resonant features for an object class is the focus of the HAR layer. In practise, a given feature descriptor does not typically acquire more than 10 child feature descriptors.

3. Feature Isolation and Classification

Hierarchical Adaptive Resonance (HAR Layer) The visual field now comprises a series of hierarchical resonant features; however, every feature can consist either partially or completely of background information. In order to isolate background information, the HAR layer creates and adjusts a series of weight vectors for each resonant feature. The HAR layer allows two complete visual fields to be compared to one another in a resonant comparison operation. The HAR layer performs this resonance comparison operation over a collection of positive and negative (background) visual fields for a given object class. The result of this layer is a set of weights which rank the resonant features with an image-to-class metric, instead of the typical image-to-image metric used in SVMs [19, 21, 17, 26, 18].

Let C_r^a be a resonant feature composed of hypercomplex features $C_{h0}^a \ldots C_{hN}^a$. Let C_r^b be a resonant feature from a separate visual field. The following code shows an algorithm for comparing two resonant features to one another. The $discardChildren$ function serves to discard the smallest hypercomplex child features until both features have the

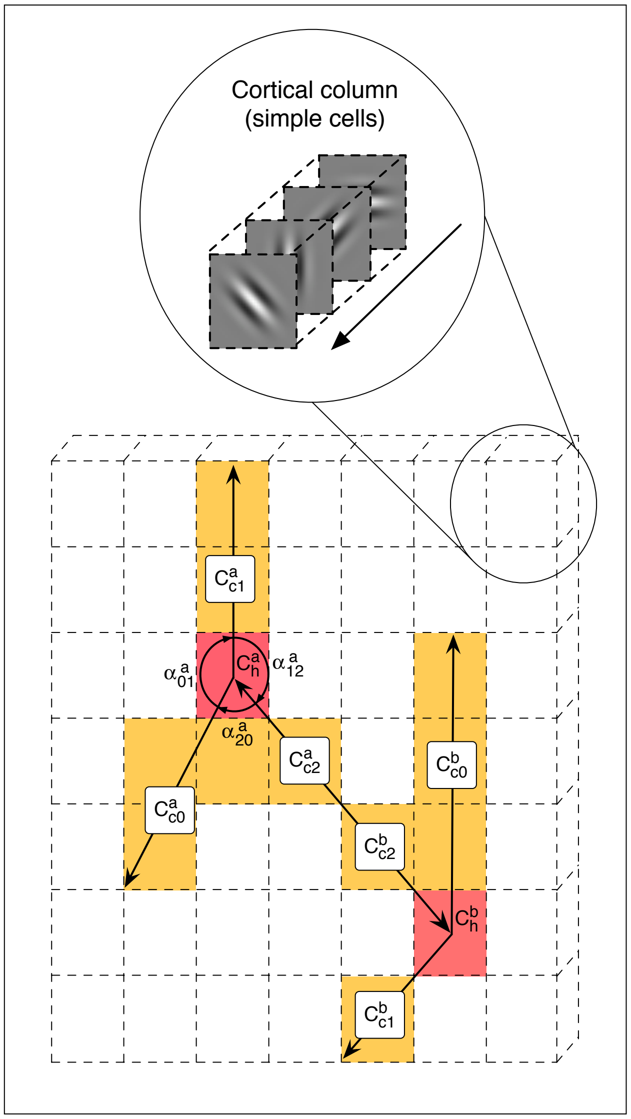


Figure 2: A sample of a single resonant feature, C_r^a, comprised of 2 hypercomplex features (C_h^a and C_h^b) and 6 complex features ($C_{c0}^a \ldots C_{c2}^a$ and $C_{c0}^b \ldots C_{c2}^b$, where $C_{c2}^a = C_{c2}^b$). The grid represents a simple cell field arranged in cortical columns, on top of which complex cell pooling occurs. Red areas represent active hypercomplex cells and orange areas represent active simple cells that have been pooled by complex cells (represented by lines). The primary complex cells, C_{c0}^a and C_{c0}^b, are associated with the largest angles in their respective hypercomplex cells. The cortical column is used to arrange complex cells in a clockwise manner for rotation invariance and to normalize complex cell length for scale invariance. The grid is represented as a texture on the GPU.

same child count.

```
if $childCount(C_r^a) < childCount(C_r^b)$ then
  $C_r^b \leftarrow discardChildren(C_r^b)$
end if
if $childCount(C_r^a) == childCount(C_r^b)$ then
  $rootAct \leftarrow ||C_{h0}^a - C_{h0}^b||$
  if $rootAct > \rho$ then
    for $k = 1$ to childCount($C_r^a$) do
      $weight[i] \leftarrow rootAct \times ||C_{hk}^a - C_{hk}^b||$
    end for
  end if
end if
```

The ρ parameter is identical to the one used in the classifier. The matching process is run on every feature combination in two visual fields. For each C_r^a in the source visual field, the best matching feature is selected (via the max operator) in the destination visual field. The process is repeated for a series of positive examples, and a set of background examples. When a feature activates for a negative class, $weight[i]$ is subtracted from the total, otherwise it is added to the total. Once complete, all final weights and features are extracted and stored in a database for use by the classifier.

Classifier (GPU ART Layer) Our model uses a classifier based on the Adaptive Resonance Theory (ART) [5] family of classifiers. The majority of biologically inspired visual cortex models use classifiers that are not biologically plausible, such as Support Vector Machines (SVM) [21, 17, 11, 26]. While SVMs are popular, they are a binary classifier which makes them inherently ill-suited for multiclass object recognition. Adaptive Resonance Theory (ART) is a form of biologically motivated classifier with several unique and highly desirable properties for object recognition. ART classifiers do not suffer from the train / test dichotomy that plagues most classifiers; they can switch between training and testing modes at any point. ART classifiers are also not limited to binary classification, they are well adapted to multiclass problems and have shown promise in visual classification tasks [24].

There are several ART based classifiers available, the most popular of which is Fuzzy ARTMAP [6]. This uses a combination of an ART-1 layer and ART-2 layer with an associative map field in between. The ART-1 layer receives a stream of input vectors and ART-2 receives a stream of correct predictions. When the two layers activate in resonance, the map field pairs their activation. The map field includes a feedback control mechanism which can trigger the search for another recognition category, in a process called match tracking. We have made the following adjustments to ARTMAP for our classifier, which we term GPU-ARTMAP:

- the classifier uses an ‘image-to-class’ metric, instead of the typical ‘image-to-image’ metrics,
- the classifier does not require bounded length input samples,
- the feature complement coding step is omitted,
- the classifier is implemented on the graphics processor; all comparisons are done in a single render pass with $O(MN)$ comparisons of M known features to N input features.

The final feature-set used for Caltech 101 includes over 250,000 features selected from the HAR Layer, all of which are stored on a series of 1024x1024 textures used during training and classification. Each feature’s class identifier is discarded when loaded into the classifier (prior to training). Since all feature data is stored on a relatively small number of textures, training or classification can be done in a single render pass.

The Programmable Graphics Processing Unit With current programmable graphics processors reaching well over 500 GFLOPS (billion FLoating point Operations Per Second), they are fast becoming the ideal platform for high performance computing [7]. We show that the GPU performs significantly better than the CPU at various SIMD processing tasks. Since the primate visual cortex operates in a retinotopic/topographic fashion [13], which is inherently SIMD in nature, the GPU is an ideal platform for modeling the visual cortex. All cellular processing in our model, from feature descriptor isolation and comparison to object classification, is done using OpenGL GLSL fragment shaders [20]. While development on the GPU has generally been quite challenging due to lack of any high level method of debugging, hardware-aware debuggers have recently advanced significantly [23], helping to reduce GPU development time.

Our model represents each layer of the cortex as one or more fragment shaders. A fragment shader a program that is executed (in parallel) on every fragment (pixel) of a texture, in what is known as a render pass. When developing algorithms for the GPU, coprocessor bandwidth can become a significant bottleneck: it is therefore crucial to minimize data transfer between the CPU and GPU. In our model, feature descriptors are read from the GPU only after the HAR layer has been applied. At this point, the features for a given object class have been isolated and a subset of these features are selected for use by the classifier.

Method	[17]	[11]	[25]	[16]	[27]	[12]	[2]	Our method
101 Accuracy	56.0	58.2	63.0	64.6	66.0	67.6	77.8	64.3
15 Scene Accuracy	-	-	-	81.4	-	-	-	76.3

Table 2: Published results for the Caltech 101 & 15 Scene datasets. Results for our model are the average of 10 independent runs.

4. Results

4.1. Multiclass Classification

The Caltech 101 [8] and 15 Scene [16] datasets are a collection of 9197 and 4485 images, divided into 101 object categories and 15 scene categories respectively. Caltech 101 includes a background category, which is also used for the HAR layer in the 15 scene dataset. Our model was run on grayscale versions of the images which have been resized such that neither the width nor height exceeded 256 pixels. We combine the "faces" and "faces easy" categories in Caltech 101, due to a high mismatch rate. Each result is an average of 10 runs. For each run we do the following:

1. For each class, we perform HAR tuning on all images within the class:
    (a) Run all steps in the model up to HAR layer for all images in the target class
    (b) Run the HAR layer on all images within the class
    (c) Run the HAR layer on 50 random images from the background class
2. Select all features from HAR tuning whose accuracy is above 50%
3. Randomly select N image training examples from each class, placing all others in the test set
4. Train ART classifier with N training examples from each class
5. Perform classification on all remaining test images

Our results for the 101-dataset is 64.3% +/- 1.3% using 30 training images. Our results for the 15 scene dataset is 76.3% +/- 1.8% using 100 training images. Our results are comparable to the state-of-the-art for both of these datasets.

4.2. SIMD versus SISD Timing

We demonstrate the speedup obtained by our GPU implementation with two benchmarks. First, we compare the execution time for all cellular operations in feature extraction, including simple, complex and hypercomplex cells on both the CPU and GPU. This operation is performed on a number of input texture sizes and the results are show in Figure 3. Second, we take a feature comparison operator, effectively the HAR Layer and compare its execution on the CPU and GPU.

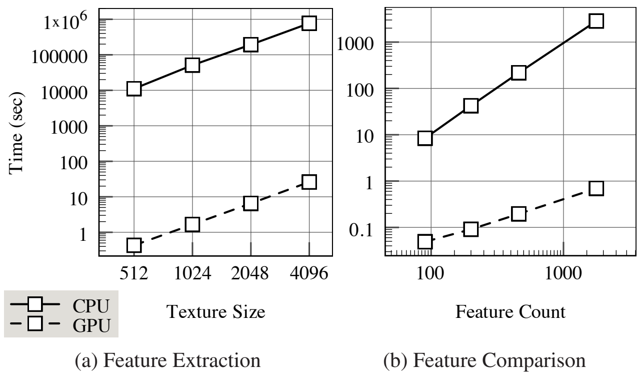


Figure 3: Execution time of SIMD versus SISD algorithms. 3a) Execution of area V1 and parts of V2 in processing 10,000 images with layer V1S to layer V2C. 3b) Execution of HAR layer feature field comparisons; an $O(N^2)$ comparison operation is performed that consist of all features in two identical images. We vary the feature count in the images by adjusting lateral inhibition. A GeForce 8800 GTX Ultra GPU and an AMD64 processor at 2.2 GHz CPU were used for tests.

The GPU achieves a speedup of approximately four orders of magnitude for the first test and approximately three orders of magnitude for the second test. This is achieved due largely to the design of the GPU: it is composed of 128 processors at 1.5 GHz each, with memory at 2.2 GHz and a memory bandwidth of 100 GB/s. The processor tested was clocked at 2.2 GHz, with memory of 100 MHz and a memory bandwidth of 3.2 GB/s. This GPU is capable of approximately 500 GFLOPS, while the CPU is approximately 6 GFLOPS. The increased processor count, memory speed and bandwidth easily accounts for the speedup observed; the GPU is clearly much better suited for the operations used in our model than the CPU. With the recent trend of CPUs becoming more multi-core, they will inevitably improve in performance at these tasks, but are unlikely to surpass the GPU in parallelism.

5. Discussion and Future Work

We have shown that a biologically motivated model can compete with state-of-the-art computer vision systems in object recognition. We have expanded the biological basis of existing cortex models to create both feature descriptors and a classifier that are closely aligned with the properties of the visual cortex. Using the graphics processor, our model is able to classify images quite rapidly compared to other models, which have been indicated to require several seconds to classify an image [17]. Our setup involves only a single GPU per machine, we could obtain a further speedup with the use of Scalable Link Interface (SLI), which allows up to four GPUs to work in parallel with one another.

It is important to consider the influence of processor architecture on a system's design: one often cannot solve a problem in the same manner on SISD and SIMD processors. Throughout this model, we have detaied operations which are much too costly to perform in an SISD environment, but are in fact are reasonable for the SIMD framework of the GPU. While there is certainly an overhead and a number of limitations that must be taken into account when developing on any new hardware, the speedup is noteworthy.

Future work involves exploring other Mahalanobis distance metric of a more biological basis for complex and hypercomplex feature comparison. It is also clear that biology uses a form of hierarchical classification to handle such a large variety of object classes; exploring methods of building hierarchies of classifiers could prove useful for scaling this model to handle much larger datasets.

References

[1] A. Anzai, X. Peng, and D. C. V. Essen. Neurons in monkey visual area v2 encode combinations of orientations. *Nature Neuroscience*, 10(10):1313–1321, 2007.

[2] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *CVPR*, 2007.

[3] G. M. Boynton and J. Hegdé. Visual cortex: The continuing puzzle of area v2. *Current Biology*, 14:523–524, 2004.

[4] G. Carpenter and S. Grossberg. ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks*, 3:129–152, 1990.

[5] G. Carpenter and S. Grossberg. *Adaptive resonance theory*. MIT Press, second edition edition, 2003.

[6] G. Carpenter, S. Grossberg, N. Markuzon, J. Reynolds, and D. Rosen. Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Trans. on Neural Networks*, 3:698–713, 1992.

[7] Z. Fan, F. Qiu, A. Kaufman, and S. Yoakum-Stover. Gpu cluster for high performance computing. In *ACM/IEEE SC Conference*, 2004.

[8] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *CVPR*, 2004.

[9] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.

[10] G. Goodhill and M. Carreira-Perpinán. Cortical columns. *Encyclopedia of Cog. Science*, 1:845–851, 2002.

[11] K. Grauman and T. Darrell. Pyramid match kernels: Discriminative classification with sets of image features. Technical Report MIT-CSAIL-TR-2006-020, MIT, 2006.

[12] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[13] K. Grill-Spector and R. Malach. The human visual cortex. *Ann. Rev. of Neuroscience*, 27:649–77, 2004.

[14] J. Hedgé and D. C. V. Essen. Selectivity for complex shapes in primate visual area v2. *J. of Neuroscience*, 20:RC61:1–6, 2000.

[15] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. *J. of Neurophysiology*, 28:229–289, 1965.

[16] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[17] J. Mutch and D. Lowe. Multiclass object recognition with sparse, localized features. In *CVPR*, 2006.

[18] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with application to object recognition. In *CVPR*, 2007.

[19] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2:1019–1025, 1999.

[20] M. Segal and K. Akeley. *The OpenGL Graphics System: A Specification (Version 2.1)*, 2006.

[21] T. Serre, L. Wolf, and T. Poggio. Object recognition with features inspired by visual cortex. In *CVPR*, 2005.

[22] M. Soriano, L. Spillman, and M. Bach. The abutting grating illusion. *Vision Research*, 36:109116, 1996.

[23] M. Strengert, T. Klein, and T. Ertl. A hardware-aware debugger for the opengl shading language. In *ACM Siggraph Workshop on Graphics Hardware*, 2007.

[24] M. Uysal, E. Akbas, and F. Yarman-Vural. A hierarchical classification system based on adaptive resonance theory. In *ICIP*, 2006.

[25] G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *CVPR*, 2006.

[26] K. Woodbeck. On neural processing in the ventral and dorsal visual pathways using the programmable graphics processing unit. Master's thesis, University of Ottawa, 2007.

[27] H. Zhang, A. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *CVPR*, 2006.