# Acceleration of ensemble machine learning methods using many-core devices

THE UNIVERSITY of EDINBURGH

## Introduction

We present a case study into the acceleration of ensemble machine learning methods using many-core devices in collaboration with Toshiba Medical Visualisation Systems Europe (TMVSE). The adoption of GPUs to execute a key algorithm in the classification of medical images was shown to significantly reduce overall processing time. The same GPU-based algorithm developed for TMVSE could also be directly applied to a suitably formed dataset to benefit supervised learning techniques applied in High Energy Physics.

## Medical Context

TMVSE provide software to process three-dimensional medical imaging data - such as computerised tomography (CT) scans - using automatic detection of anatomical landmarks defined on the skeleton, vasculature and major organs. Landmark detection underpins a semantic understanding of the medical data and thus has many diverse applications. This method facilitates rapid navigation to a named organ thereby assisting radiologists performing post-scan analysis.

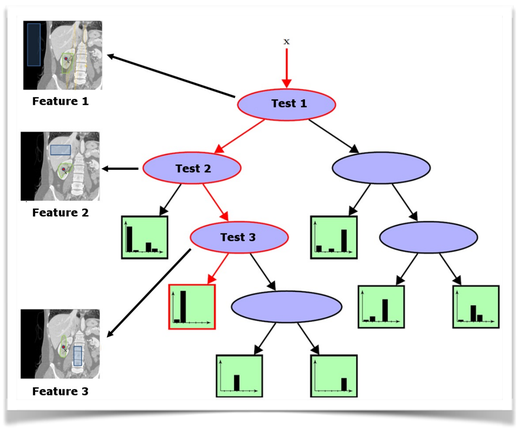



Figure 1 Landmark detection using the Decision Tree technique

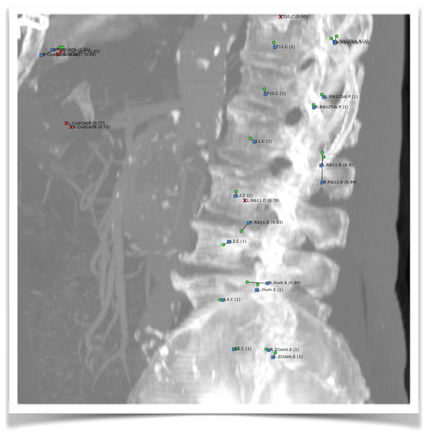


Figure 2 Visual illustration of automatically detected landmarks positioned on novel CT scan

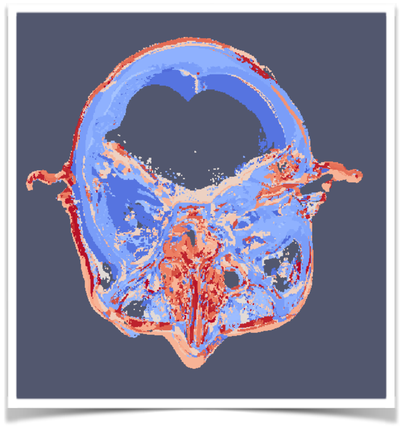


Figure 3 Visualisation of a slice from the volume coloured by identified landmarks

## GPU-based classification

It is important that the classification step runs quickly to facilitate user interaction with a sub-second processing time being desirable. A GPU-based decision tree traversal algorithm was developed to determine if many-core devices could offer a significant performance advantage. Testing and validation was possible by the processing of an anonymised image volume and training output independent of the TMVSE framework. A CPU-based implementation of the same traversal algorithm was developed in-step with the GPU version for performance comparison studies. The timing results for the GPU-based version demonstrates a clear improvement (Figure 4) over both single-threaded CPU and multithreaded (OpenMP) CPU implementations.

| threads | traversal time (s) | traversal speedup | execution time (s) | speedup |
|---|---|---|---|---|
| 1 tree | | | | |
| 1 | 21.16 | 1 | 21.32 | 1 |
| 8 | 2.70 | 7.8 | 2.86 | 7.5 |
| 16 | 1.72 | 12.3 | 1.89 | 11.3 |
| GPU | 0.1 | 211.6 | 5.72 | 3.72 |
| 80 trees | | | | |
| 1 | 1687.38 | 1 | 1687.56 | 1 |
| 8 | 213.24 | 7.9 | 213.31 | 7.9 |
| 16 | 134.76 | 12.5 | 143.92 | 11.7 |
| GPU | 7.96 | 212.0 | 12.18 | 138.6 |

Figure 4 Overview of timing performance for image classification

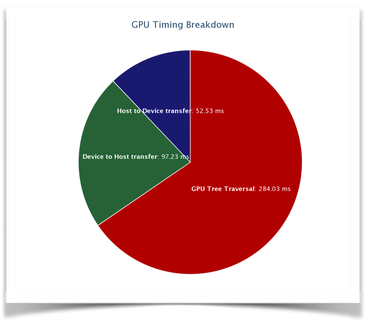
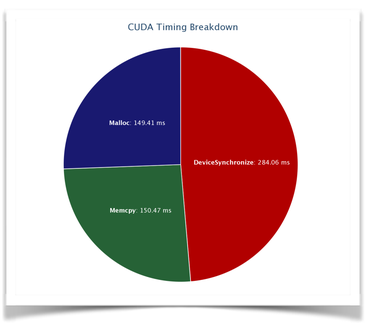
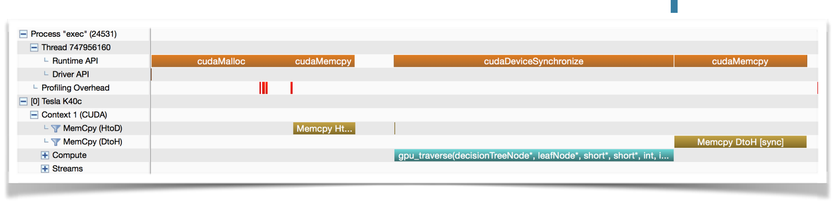



Figure 5 Breakdown of timing results from GPU profiling studies




## Classification Technique

Supervised ensemble machine learning methods are applied to imaging data to determine whether a given image voxel (a volumetric pixel) can be classified as belonging to one of 127 registered landmark sites marked by anatomists. *Decision forests* are favoured over similar supervised learning methods (such as Support Vector Machines) due to their straight-forward application to multi-class problems and classification transparency.

Decision forests composed of 80 decision trees are each trained with 40 (out of possible 369) reference datasets with each dataset combining training voxels in the neighbourhood of a landmark with randomly sampled background. Every voxel in a new image is passed down each tree in the forest to calculate a set of normalised likelihoods. For each landmark, the voxel with the greatest normalised likelihood for that landmark is selected as the potential detection point.

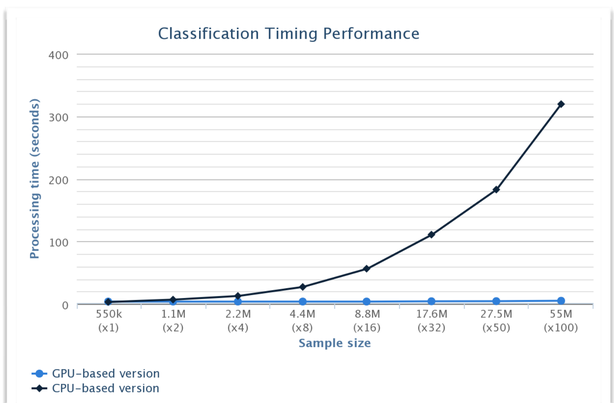
## Optimisation Approaches

Optimisations for GPU execution were made iteratively using guidance from CUDA profiling results (Figure 5). These included:

- Maximising device occupancy by aligning thread block and grid sizes to match image volume dimensions
- Defining data structures to access consecutive GPU memory addresses and using constant memory on the GPU for faster per-thread read access to decision trees
- Reducing host to device transfer latency overhead by the efficient packing of data structures and bulk copying of image volumes and training data
- Enabling CUDA streams to overlap data transfer with classification execution
- Coalescing global memory writes across threads when storing voxel classification results

## Applications for High Energy Physics

The GPU-based decision forest classification technique developed for the processing of medical images was directly applied to the classification of HEP collision event data without loss of functionality. The *Scikit learn* toolkit was used to construct and train a decision forest and converted into the same input format developed for medical image processing. The main difference between the two datasets was the classification of data into only two classes (signal and background) for HEP data compared with 127 classes in the TMVSE example. Processing time independent of sample size was observed up to 55 million events (100 times the original sample size) before GPU memory threshold was reached compared to the superlinear scaling found in the CPU-based reference version (Figure 6). Further optimisations have been identified in the existing code to enable improved performance for processing of HEP data.



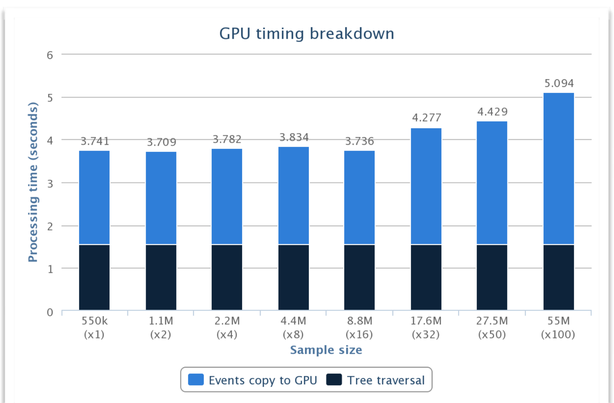




Figure 6 Classification timing performance using HEP dataset


Arjen Tamerus[1], Andrew Washbrook[1], Daniel Wyeth[2]

[1]University of Edinburgh [2]Toshiba Medical Visualisation Systems Europe