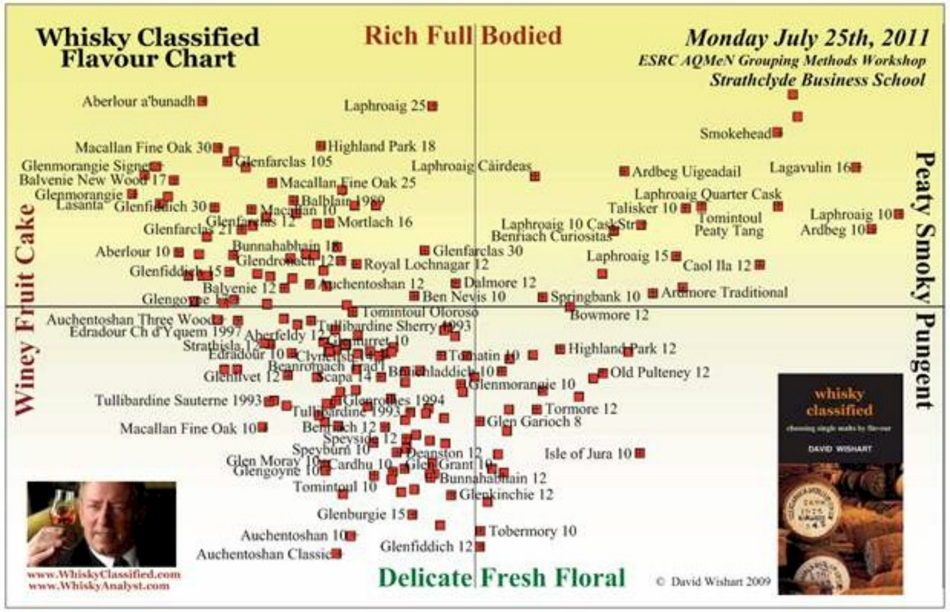


TMVA Method Optimisation Feasibility Study

Hazel McKendrick and Andrew Washbrook
Future Computing Workshop, 16th June 2011, University of Edinburgh

Toolkit for Multivariate Analysis (TMVA)

- The Toolkit for Multivariate Analysis (TMVA) provides a ROOT-integrated environment for the processing, evaluation and application of multivariate classification (and regression) techniques.
- The software package consists of abstract, object-oriented implementations in C++/ROOT for each of the MVA techniques, as well as auxiliary tools such as parameter fitting and transformations.
- Their training and testing is performed with the use of user-supplied data sets in form of ROOT trees or text files.
- The TMVA training job runs as a ROOT script, as a standalone executable, or as a python script via the PyROOT interface.

(from the TMVA users guide)

TMVA Methods

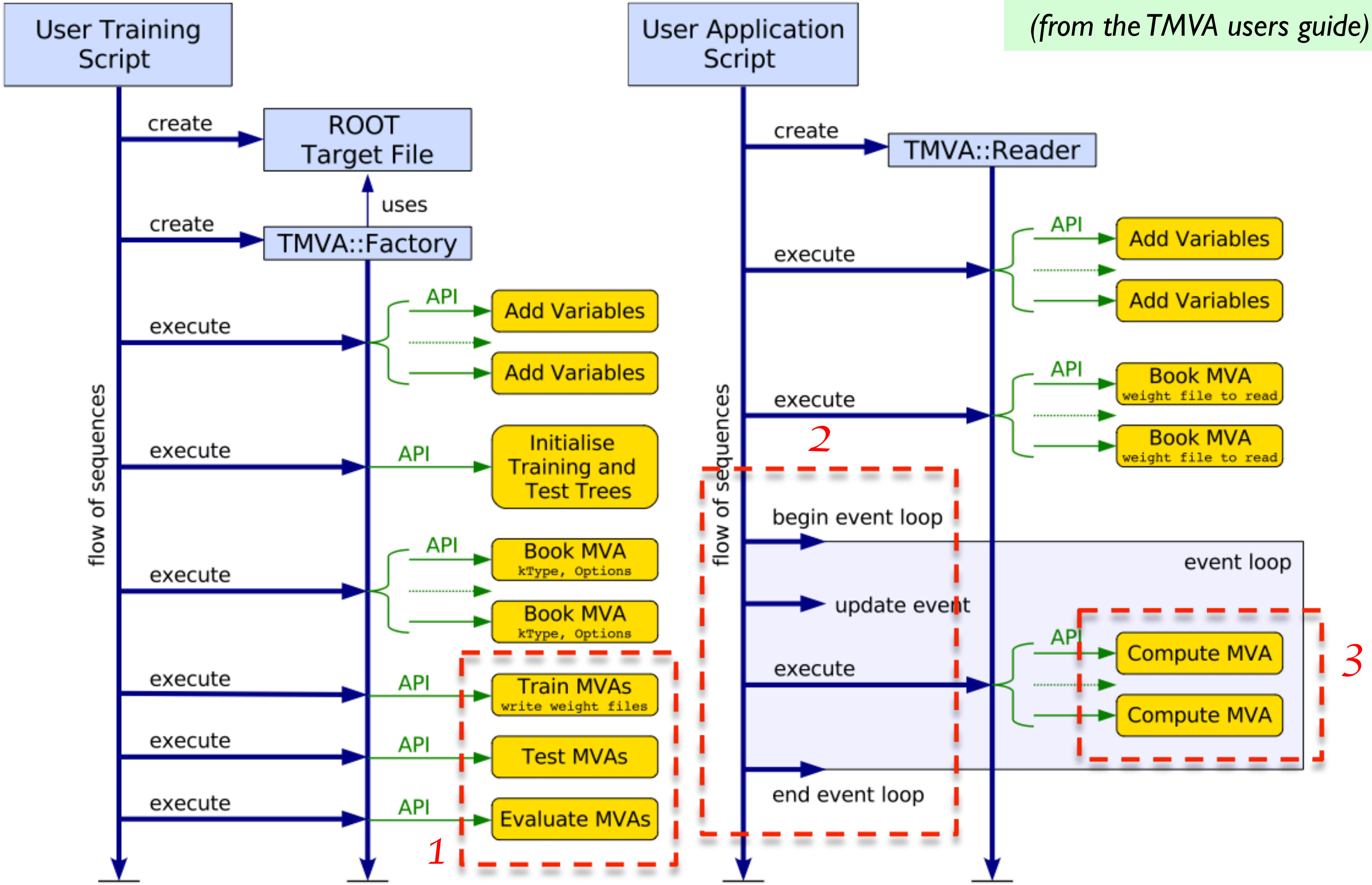
- A whole host of multivariate techniques are available:

Rectangular cut optimisation	Projective likelihood estimator (PDE)
Multi-dimensional likelihood estimator (PDE range search)	Likelihood estimator using self-adapting phase-space binning (PDE Foam)
k-Nearest Neighbour Classifier	H-Matrix discriminant
Linear Discriminant Analysis	**Artificial Neural Networks**
Support Vector Machines	Boosted Decision Trees

Parallelisation effort elsewhere	Non-linear approximations

TMVA Application Flow

(from the TMVA users guide)



TMVA Technique Performance

- Investigate where MVA technique performance gaps are found:

		CLASSIFIERS									
CRITERIA		Cuts	Likeli-hood	PDE-RS	k-NN	H-Matrix	Fisher	ANN	BDT	Rule-Fit	SVM
Perfor-mance	No or linear correlations	★	★★	★	★	★	★★	★★	★	★★	★
	Nonlinear correlations	○	○	★★	★★	○	○	★★	★★	★★	★★
Speed 1	Training	○	★★	★★	★★	★★	★★	★	○	★	○
	Response	★★	★★	○	★	★★	★★	★★	★	★★	★
Robust-ness	Overtraining	★★	★	★	★	★★	★★	★	○	★	★★
	Weak variables	★★	★	○	○	★★	★★	★	★★	★	★
Curse of dimensionality 2		○	★★	○	○	★★	★★	★	★	★	
Transparency		★★	★★	★	★	★★	★★	○	○	○	○

Fair

Good

Bad

TMVA and GPUs

- Feasibility studies will be performed on GPUs using Nvidia CUDA (for now)

Pros	Cons
Potential for large speed gains	Challenge of (re-)developing applications
Greater increases in performance when compared with CPUs	Not well suited to all tasks
Power consumption, price to performance	Constantly evolving hardware and APIs

CUDA 4.0 just released

C++ Support
• Dynamic memory allocation (new/delete)
• virtual function support
GPU Device Memory Addressing
• No-copy pinning of system memory
Multi-GPUs
• GPUDirect v2.0 support for Peer-to-Peer Communication
• Use all GPUs in the system concurrently from a single host thread

Previous GPU Multivariate effort

- Simulated Annealing (Rectangular cut optimisation)
  - [Parallelizing Simulated Annealing-Based Placement using GPGPUs]
  - An average speedup of about 10x was achieved
- Genetic Algorithms (Rectangular cut optimisation)
  - [Parallel Genetic Algorithms on Programmable Graphics Hardware]
  - Fitness functions must be evaluated entirely on GPU
  - Challenge of generating pseudo random numbers on GPU
- Artificial Neural Networks
  - [Artificial Neural Network Computation on Graphic Process Units]
  - GPU based computation is about 200 times faster than CPU
- Support Vector Machines
  - [Fast Support Vector Machine Training and Classification on Graphics Processors]
  - Training time is reduced by 5—32×, and classification time is reduced by 120—150×

- Significant speed up reported for several techniques, but can this be easily ported into the TMVA framework?

Feasibility Study Approach

- Optimise Individual TMVA techniques
  - Rectangular Cut Optimisation, Neural Network and Support Vector Machines are early candidates
- Go for a general approach
  - Data structure analysis
  - "Accelerator" method
- Consider algorithm patterns for parallelisation
  - e.g. Map-Reduce in SVM
- Start with Bottleneck studies – look for hotspots.
- Cross platform performance analysis

Feasibility Study Approach

- Listen to the developers!

"We stress however that, to solve a concrete problem, all methods require at least some specific tuning to deploy their maximum classification or regression capabilities"

- The training (and evaluation) phase is far more time consuming than the application phase.
- Would like to introduce automatic parameter optimisation to the training procedure to avoid sub-optimal training.

- Possible approach:
  - Run training cycles - in parallel - with differing parameters
  - Decide the best parameter choice with rudimentary fit.

Comments and suggestions welcome!