An exemplar of Accurate Approximation Computation ---- a GC7 subtopic

A Bayesian computer

D.Partridge,T.C.Bailey,R.M.Everson,A.Hernandes,W.J.Krzanowski,J.E.Fieldsend and V.Schetinin, The Critical Systems Group, School of Engineering, Computer Science and Mathematics,University of Exeter

Real-world classification tasks are a rich source of problems for which traditional approaches to the development of a computational solution do not easily apply; the main difficulties are:

1. The task is defined by a set of data (input-output pairs) rather than by an abstract specification --- they are data-defined problems.
2. It is usually not possible, nor expected, nor required that an implementation be `correct' in the traditional sense, i.e. that every valid input will be correctly classified. Indeed, it may be that certain valid inputs do not have an unequivocally correct classification.

Such problems vary from classification of persons in a database as likely to purchase a given product, in which case very small success rates can be commercially valuable, and misclassifications amount to no more than a reduction in profit. At the other extreme, an air-traffic control collision avoidance problem may demand 100% success (to within a demanding probabilistic bound) for potential collision inputs and involve major loss of life when dangerous situations are misclassified.

We propose one manifestation of accurate approximation computation (AAC) that might offer an optimal computational solution to a wide variety of such data-defined classification tasks.

AAC fundamentals[1]

basic quantities not discrete values but probability densities

coherent strategy for combining probabilistic components

results not *correct/incorrect* but accurately approximate

statistics rather than formal logic

In this AAC strategy an unknown input generates not a discrete classification (or even a set of discrete classifications such as a multiclassifier system, MCS, might produce) but a probability distribution over each of the target classes. This set of distributions constitutes a precise approximation to the classification of the input data which may be further interpreted to yield a (possibly probabilistic) discrete classification outcome --- i.e. a classical classification of the input.

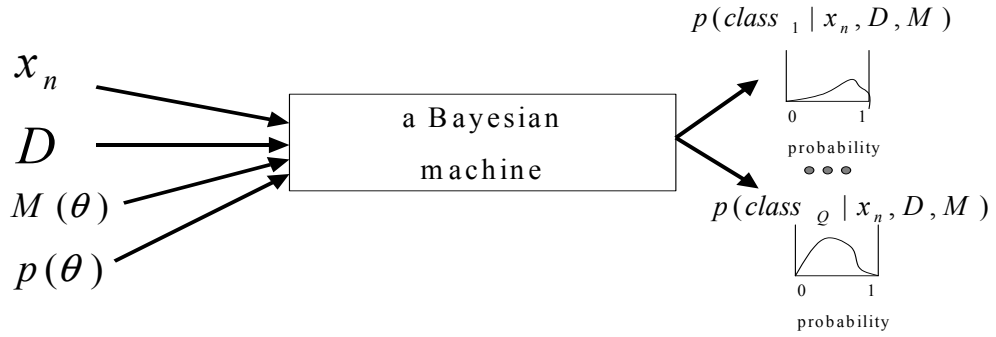
One way to view classification problems is that they involve the fitting of a mathematical model to the data, and different models give rise to different classifier systems (e.g. k nearest neighbours, knn, or multilayer perceptrons, MLP). Each model contains adjustable parameters (e.g. k, the number of nearest neighbours, in the knn model, and the number of hidden units and hidden layers in an MLP). Thus the fitting of a model involves techniques for optimizing the particular model parameter values from the information contained in (sub)set of the available data --- the training data. Having set the parameter values, we have a classifier for this data such that for any valid input we obtain a predicted classification.

The AAC strategy to be described uses Bayes' rule. Thus the output of our computation is $p(x \mid y, D, M)$, the probability that the input data ($x_n = (X_1, \cdots, X_n)$ a vector of n features) can be

[1] Abstracted from "A Science of Approximate Computation", appendix to GC7 "Journeys in Non-Classical Computation"

classified as target class, *y* given a set of training data, *D*, and a classifier model *M(θ)* parameterized by the vector of parameters,*θ*. Bayes' rule gives the posterior density over the model parameters *θ*, $p(\theta \mid D, M)$ as $p(\theta \mid D, M) = \frac{p(D \mid \theta, M)p(\theta \mid M)}{p(D \mid M)}$ for which we need to define prior probabilities over the parameters, $p(\theta \mid M)$. With the posterior density over *θ* on hand we can integrate out the dependence upon the model parameters $p(y \mid x, D, M) = \int p(y \mid x, \theta, \mathrm{M})\mathrm{p}(\theta \mid \mathrm{D}, \mathrm{M})\mathrm{d}\theta$, and so obtain a probability distribution for the input vector (conditioned on both the training data and the classifier model type) for each of the target classes, the *y*s.

For this we require a Bayesian machine which can be envisaged as a classifier in terms of the following inputs and outputs, when *y* is composed of *Q* classes.



A number of limited-scope (virtual) Bayesian computers do exist (e.g.BUGS, see http://www.mrc-bsu.cam.ac.uk/bugs/references/bugs-core-papers.shtml), but in general many of the desired integrals are analytically insoluble (not to mention the necessity for real computational continua, such as the real numbers). Markov Chain Monte Carlo (MCMC) methods are one classical technique used to sample distributions in a way that focuses the sampling in areas of high probability thus providing a means efficient approximation to the desired integrals. The well-founded bases of both Bayes' theorem and MCMC methods underwrites the validity of the classification probability distributions generated, and the distributions provide a basis for the `confidence' associated with each classification result. The outcome is that MCMC methods permit us to draw samples $\theta^{(i)}$ from $p(\theta \mid D, M)$ so that $p(x \mid y, D, M)$ is approximated as $p(y \mid x, D, M) \approx \frac{1}{N}\sum_{i=1}^{N} p(y \mid x, \theta^{(i)}, M)$.

This classical simulation of a generalized Bayesian computer is computationally expensive (it is only recent hardware advances that have moved such strategies into the realms of practicality). The sampling is massive, as it must be to generate accurate continuous distributions from a set of data points, but the reward is the extra information contained in such soft solutions as opposed to the poverty of information and brittleness of a single categorical result (or indeed a set of such results when generated by an ad hoc MCS).

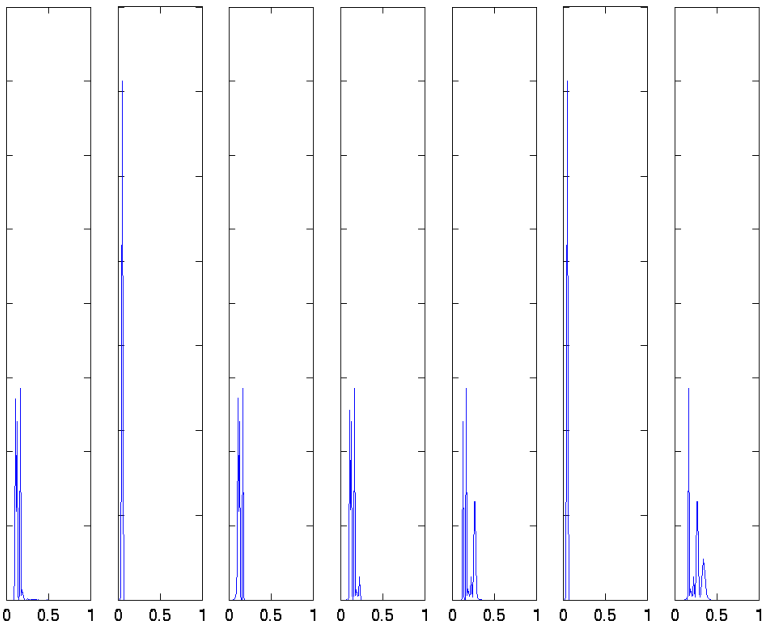
It is true that this distribution is constructed from a large number of discrete classification results, but this interchange from discrete to continuous and vice versa occurs repeatedly throughout computational

hierarchies (e.g. the continuous output of a sum-and-squash function that is the output of an MLP may be thresholded to yield a categorical classification; the fundamental continuity of the real numbers is typically discretized when they enter a computation). Discrete or continuous is more a matter of interpretational convenience, or level of interpretation, than it is an inherent property of a representation.

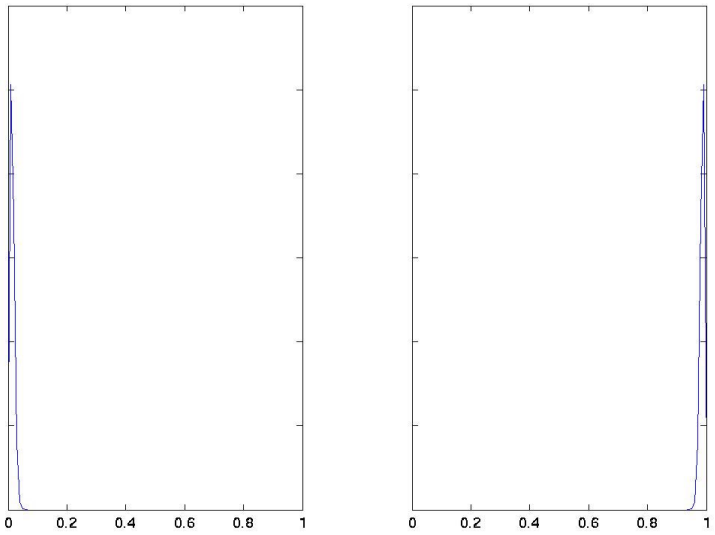
A recent paper (Wegner and Eberbach, 2004) outlines three "superTuring" models of computation --- Interaction Machines, the π-calculus for evolutionary computation and the $-calculus for robotics --- as examples of areas that cannot properly be described using Turing Machines (TM) and algorithms. The essential weakness of the TM with respect to the proposed superTuring models is that the TM is a closed model. Our proposed departure from the TM model is required because of the discrete exactness that is a foundation of the TM model.

So, although our output distributions are composed of thousands of discrete results and our computational base is a standard Turing machine running classical algorithms, we view the computational system as a virtual Bayesian computer that integrates over continua and computes probability distributions. This discrete classical basis is no more than a convenient way to simulate the (currently) necessary MCMC approximations to the desired Bayesian machine that would support a direct implementation of this particular approach to AAC. Even in this Bayesian-MCMC virtual machine the fine-grained discrete computations lack meaning as individuals (e.g. a point probability plucked from a Markov chain); it is only the totality, the shape of the continuous distribution that a large number of such points represent, that is a meaningful/useful outcome.

Some examples: the following outputs have been selected from results of applying the above computational scheme to a variety of the UCI Machine Learning database files. Further details may be found elsewhere (Partridge et al., 2003). For these illustrations, $M(\theta)=Pknn(k,\beta)$ and $p(k)=U\{1,\ldots,kmax\}$ and $p(\beta)=U(0,\infty)$; full technical details can be found in Denison et al.. 2002. The first illustration shows distributions of the 7 classes for the Image data. As can be seen none of the distributions appear to strongly favour any of the classes and this output was classified as UNSURE using our uncertainty envelope approach (Fieldsend et al. 2003).



Illustrated below is the output for the 2 classes of the Wisconsin data. As can be seen, these particular distributions clearly favour the classification given on the right-hand side (note: for a two class problem only one distribution is necessary because the two distributions will be mirror images, as illustrated). Using our uncertainty envelop idea to further interpret the Wisconsin data, we obtained 88.6% SURE correct and 11.4% UNSURE which can be compared to 98.3% correct using best classifier (Partridge et al., 2003).



References

D.G.T. Denison, C.C. Holmes, B.K. Mallick and A.F.M. Smith, 2002, "Bayesian Methods for Nonlinear Classification and Regression," Wiley:NY.

J.E. Fieldsend, T.C. Bailey, R.M. Everson, W.J. Krzanowski, D. Partridge and V. Schetinin, 2003, "Bayesian Inductively Learned Modules for Safety Critical Systems," Proc. Of the 35th Symp. On the Interface: Computing Science and Statistics, March 12-15, Salt Lake City.

D. Partridge, J.E. Fieldsend, W.J. Krzanowski, T.C. Bailey, R.M. Everson and V. Schetinin, 2003, "MCS Diversity and Classifier Confidence: a Bayesian approach," Dept. Computer Science, Res. Rep. No. 417, University of Exeter.

P.Wegner and E.Eberbach, 2004,New Models of Computation, *The Computer Journal* 47(1), 4-9.