DFUB-92-13

DFUB-92/13

13

Neural Network Discrimination of Heavy Flavor Jets: a Survey

P. Mazzanti* and R. Odorico**

University of Bologna, Department of Physics Bologna U.
and
Istituto Nazionale di Fisica Nucleare, Sezione di Bologna INFN, Bologna
Via Irnerio 46, 40126 Bologna, Italy

ABSTRACT

A short survey of the use of neural networks and statistical discriminants in high energy physics for recognition of heavy flavor jets is presented. After illustrating the various neural and statistical classifiers currently used, some assessment of their comparative performance for top and bottom jets is made.

Presented at Second Workshop on Neural Networks: from Biology to High Energy Physics,
June 18-26, 1992, Marciana Marina,Isola d'Elba, Italy

Research supported in part by a computing grant from CINECA, Bologna

* e-mail: mazzanti@bologna.infn.it (bitnet), 39948::mazzanti (decnet)

** e-mail: odorico@bologna.infn.it (bitnet), 39948::odorico (decnet)

Neural Networks (NN) are receiving increasing attention for recognition problems in High Energy Physics (HEP), like t-quark [1-3] and b-quark [4] tagging with t, b → jets. Also of interest are other applications like e/π discrimination for improved lepton tagging. Especially used are NN based on Learning Vector Quantization [5] to which Training Count can be profitably added (LVQTC) [3,6] and Back-Propagation (BP) [7-9]. One should not forget more conventional statistical methods like the Fisher linear discriminant [10,11] and the Gaussian classifier [11].

To get some orientation, let me start with the simplest classifier: Fisher's linear discriminant [10,11]. As for all classifiers, one must first encode the event (or whatever objects one is classifying, e.g. jets) into a number of feature variables $s_1, s_2, ..., s_n$, which can be arranged into a vector, the pattern vector s. Thus, each event corresponds to a point s in the multidimensional pattern space. Let us consider a schematic 2-dimensional example: Fig. 1. When projecting the events onto the s_1 axis, one gets two overlapping distributions in s_1. When trying to discriminate the two classes by a cut in s_1 one gets penalized in efficiency and purity. The same holds true when projecting onto the s_2 axis. But if one projects onto the Fisher axis shown in the figure, the two distributions get separated, and thus discrimination with 100% purity and efficiency is achieved. That is a dream situation, of course, but in a real case one can reduce the overlap between the two distributions to some minimal value in this way. The Fisher variable F, associated with the Fisher axis, is a linear combination of the feature variables $s_1, s_2, ..., s_n$, which can be determined by simple mathematics involving the the correlation (or covariance) matrices for the two distributions. The physical ingredient one is exploiting is the **correlation** between the variables within each class distribution. In the example considered, for each s_2 bin s_1 ranges over two distinct intervals for the two distributions. As the s_2 bin moves, the populated s_1 intervals change, which means that s_1 is correlated to s_2 within each distribution. The s_1-s_2 correlation turns out to be different within each distribution, and that is what is exploited in the discrimination.

The Fisher classifier cannot be outperformed if the two distributions are gaussian and have the same correlation matrix, i.e. if they have the same shapes and just distinct centroids. In this case, its purity and efficiency of classification are only limited by the amount of overlap between the two distributions. If the two distributions have different correlation matrices, one can introduce a more general Gaussian classifier, that cannot be outperformed if the two distributions are gaussian [11]. For that, one considers the probability density functions corresponding to the gaussian approximations for the two distributions:

$$p(s) = A \exp\{-\frac{1}{2}(s-<s>)M^{-1}(s-<s>)\}$$

where M is the correlation matrix for the distribution and <s> is its centroid. The Gaussian classifier for a pattern s is given by the variable $G = \ln(p_A(s)/p_B(s))$. If G is positive, s is classified as A, otherwise as B. The absolute magnitude of G gauges the reliability of the classification. If the two correlation matrices are equal, G identifies with the Fisher variable F apart from a constant term: $G = F + \ln(A_A/A_B)$.

If the distributions are not gaussian, in general the Gaussian classifier does not yield the best purity and efficiency which are in principle obtainable given the overlap between the two distributions (i.e. it no longer reaches the Bayesian limit). That is where Neural Networks come to help.

As an illustration, let us consider the simple 2-dimensional example of Fig. 2, with the two distributions being uniform within the regions they cover. The outer distribution (B) is quite far from gaussian. The Gaussian classifier gives a bad performance in this case.

Let us see how a NN like LVQTC [5,3,6] handles the problem. With this NN architecture neurons can be associated with vectors, or points, in pattern space. Their positions are fixed by a training procedure in which a sequence of patterns of known class s(t), t = 1, 2,3... is presented. For each pattern s(t), one corrects the position of the neuron closest to it, m_c, by moving it closer to the pattern if the two belong to the same class, or moving it away from it if they belong to different classes:

$$m_c(t+1) = m_c(t) + \alpha_+(t)\,[s(t)-m_c(t)] \qquad \text{if } m_c \text{ and s belong to the same class}$$
$$m_c(t+1) = m_c(t) - \alpha_-(t)\,[s(t)-m_c(t)] \qquad \text{if } m_c \text{ and s belong to different classes}$$

$\alpha_+(t)$ and $\alpha_-(t)$ are positive learning parameters decreasing with t. During training the number of times each neuron is corrected by patterns of the various classes is counted. From that the neuron purity can be calculated, i.e. the fraction of times the neuron is corrected by patterns of its own class. The classification of a pattern s of unknown class is made by simply assigning the pattern to the class of its closest neuron. The purity of the classification can be estimated by the purity of the neuron providing the classification. For the problem of Fig. 2, 100% purity and efficiency of classification can be achieved with 1 neuron of class A and 16 neurons of class B. Their positions in Fig. 2 are those resulting from training.

In a BP net [7] the relevant neurons are not associated with points but rather with hyperplanes in pattern space. For the 2-dimensional problem of Fig. 2, the hyperplanes become just straight lines. In such a net, neurons are arranged in successive layers, the excitations of neurons in a layer being determined by the excitations of neurons in the previous layer, Fig. 3 (hence the name of Multi-Layered Perceptron more correctly used for such a net, the term Back-Propagation referring more properly to the type of training algorithm used). The excitations of neurons in the input layer (L=0), one for each pattern component, are directly given by the values of the corresponding pattern components. The excitation $x_i(L=1)$ of a neuron in the next hidden layer is determined by feeding the value of the linear expression $a_i = \Sigma_k\, \omega_{ik}\, s_k + \theta_i$ into a saturating transfer function: $x_i(L=1) = g(a_i)$, e.g. $g(a_i) = \tanh(a_i)$. a_i can be visualized as the distance of the pattern from a straight line (a hyperplane in the general case), whose orientation and position are determined by the "weight" parameters ω_{ik} and the "bias" term θ_i associated with the neuron. Several hidden layers may be included, but one has been found to be enough in most HEP applications. The last, output, layer may consist of several neurons, but one is enough if the classes are just 2. Its excitation is determined by iterating the procedure used to calculate the excitations of the hidden neurons. The value of the net is due to the existence of a training algorithm (BP) which starting from the discrepancies of the output excitations with respect to the desired (target) results for each training pattern (of known class) corrects the net parameters (weights and bias terms) so as to achieve minimum output discrepancies at the end of training. There is no guarantee, though, that the minimum obtained is an "absolute" minimum. For our example, 100% purity and efficiency of classification are achieved on an independent test set of patterns by using 3 hidden neurons, represented by the 3 straight lines in Fig. 2, and an output excitation given by $x(L=2) = \tanh(1.5\ \Sigma_i\ x_i(L=1) + 4.5)$ (the distance metric from the hidden neurons lines is 4.5 times the Euclidean metric). A pattern is classified as A if $x(L=2) < 0.5$, otherwise as B.

The example helps to illustrate some important differences in the usage of LVQTC and BP nets.

BP requires a relatively limited number of parameters and thus training statistics can be kept small. The cpu time required for training is typically long, since when correcting positions of hyperplanes describing hidden neurons to better accommodate patterns in a given region of pattern space, far away patterns can easily get penalized. Optimization must thus be handled globally, going through the whole training set. The purity/efficiency trade-off in classification can be controlled by cuts on the output excitation.

LVQTC requires comparatively many more parameters, and therefore the training statistics must be much larger than in BP. The hoped for reward is a higher degree of purity in classification. The cpu training time is typically short, since in order to better accommodate patterns in a region one must correct only neurons in that region, without affecting classification performance for far away patterns. The purity/efficiency trade-off can be controlled by cuts in the neuron purity.

Dedicated hardware implementations of BP and LVQTC (e.g. for triggers) also have distinct requirements. BP needs vectorization and a realization of the non-linear transfer function. LVQTC can profit of massive parallelism of vector processors, the winner-takes-all step being implemented by a few cycles of a neural net with lateral inhibitions.

The relative performances of BP and LVQTC largely depend on the problem at hand. The two-spirals problem of Fig. 4 provides a classification example that, while trivial for LVQTC, has found no solution to date with standard BP [12]. Only a solution with a modified feed-forward net has been found, in which each unit receives incoming connections from every unit in every earlier layer, not just from the immediately preceding layer [13].

A typical field of application of statistical and NN classifiers is provided by the discrimination of top and bottom jets, originated by t and b quarks decaying into anything (i.e. without a rate-reducing lepton tagging). In [2] it has been shown that for $t\bar{t}$ events produced at the Fermilab Tevatron collider one can get a ratio signal/background ≈ 1.5 with a residual $\sigma(t\bar{t}) \approx 2$ pb, for $m_t = 100$ GeV, by using Fisher's discrimination after some preliminary cuts. The utilization of LVQTC [3] or of BP does not improve on this result.

During the last year a substantial number of contributions have appeared on the utilization of NN's for discriminating b jets at LEP [4]. In Fig. 5 the results for the purity versus efficiency curve are compared. Efficiency is defined as the fraction of actual b jets recognized as b, purity as the fraction of jets recognized as b which are actually b. All calculations in the figure have been done with BP and with the JETSET [14] event generator (entries containing the ALEPH and DELPHI tags include the simulation of the corresponding experimental apparatuses, see [4,16] for references). Since the calculations differ only for the input variables used, one can appreciate the dependence on their choice from this comparison. In all cases presented, no information on leptons and on the impact parameters of b secondaries is included. Apart from some unlucky options, in the relevant high purity (> 0.6) region there is an essential coincidence of results obtained using quite different variables, which suggests that there is little room for improvement within the domain of jet or event shape variables. Needless to say, the selection of variables is essentially empirical, with some guidance obtainable from statistical tests like, e.g., the magnitudes of Fisher vector components with respect to their statistical errors. Besides the discriminating power, there are other considerations entering the selection, like: i) simplicity of calculation from the detector output for on-line triggering applications, ii) stability with respect to apparatus effects, to make event simulation easier, iii) portability to other processes and/or energies which, e.g., makes jet variables preferable to event variables.

It is of interest to study the dependence of the results on the discrimination technique used and, most important of all, on the event generator used for the simulation of jets.

We consider two event generators for this sake: i) COJETS, in which fragmentation is handled by an independent jet fragmentation model, with jet-mass phase-space effects taken into account, and parton-coherence left out; ii) JETSET, with its elaborate string model for fragmentation and with parton coherence effects included. Both event generators give acceptable fits to the relevant e^+e^- data [16]. They mainly differ by the degree of dynamical correlations, which is higher in JETSET. For the comparison, the 17 jet calorimetric variables of [17] are used, whose results for JETSET and BP are reported in Fig. 5. Fig. 6 shows the purity versus efficiency results obtained with the various classifiers and using the same event generator for training and testing. Fig. 7 does the same, but using different event generators for training and testing. From Fig. 6 it appears that, independently of the differences among the various classifiers, according to JETSET b jets are easier to recognize than what one expects with COJETS. That remains true when the event generator used for training is changed, Fig. 7. Comparing Fig. 6 and 7, it also appears that COJETS b jets are better recognized after JETSET training, apart from the case of the Gaussian classifier (maybe because of its critical use of the correlation matrix). To appreciate this point more conveniently, Fig. 8 compares the Fisher's discriminant results for all possible event generator combinations for training and testing. In their globality, these results can be rationalized by concluding that COJETS and JETSET contain the same type of correlations for b jets, but that the latter are more pronounced in JETSET. In this connection, Fig. 9 is meant to help us realize that, e.g., it is easier to discriminate geometric figures (both ideal and fuzzy ones) by training our eye on ideal examples of them than by doing the training on fuzzy ones. SA larger difference between b and non-b jets in JETSET is also supported by estimates of the volumes V and linear dimensions L of the corresponding distributions in pattern space, which one can get from the square roots of the determinants and traces of the associated correlation matrices. One has $V_b / V_{non\text{-}b} = 7.20$ for COJETS and = 12.56 for JETSET, $L_b / L_{non\text{-}b} = 1.18$ for COJETS and = 1.26 for JETSET. I.e., the b distribution is more spread out than the non-b distribution, the difference being larger in JETSET than in COJETS. From LVQTC one can also estimate the overlap between the two distributions using the neuron training counters. Defining the overlap as $W = 2\, V_b \cap V_{non\text{-}b} / (V_b + V_{non\text{-}b})$, one gets W = 0.432 for COJETS and W = 0.412 for JETSET, i.e. less overlap for JETSET.

As to the relative performances of the various classifiers, one can observe that there are not dramatical differences among them for the problem at hand. When looking at the various combinations of event generators for training and testing, relative performances fluctuate with no clear trend favoring one classifier over the other. The simplest of them, the Fisher discriminant, appears to do an adequate job in all cases, without need to use neural nets.

In conclusion, for b jets:

i) Jet shape variables can be usefully exploited for b jet tagging. Used alone or in combination with lepton and/or impact parameter information they substantially improve statistics for b jet recognition (among other possibilities, one can loosen lepton isolation criteria and impact parameter cuts). One does not need sophisticated choices for them: simple jet calorimetric variables, which are stable towards apparatus effects, do the job.

ii) It is not necessary to use a complex discrimination technique to handle b jet tagging. A conventional Fisher's discriminant is adequate, with all the advantages offered by its simple to reproduce parametrization and its computing speed, which is especially interesting for possible on-line applications.

iii) More attention should be paid to the systematic errors associated with the event generators uses for the simulation. One should remember that existing generators have been largely tuned to data for single variable distributions, and are left essentially untested for the correlations they contain, which represent a crucial ingredient in jet classification. Most of the results for b jet tagging by neural nets presented up to now have ignored this point by using just one event generator. Because of the model dependence of the way jet fragmentation is handled, event generators with as different as possible fragmentation schemes should be used (compatibly with an acceptable reproduction of existing data).

References

1) R. Odorico, Phys. Lett. 120B (1983) 219; G. Ballocchi and R. Odorico, Nucl. Phys. B229 (1983) 1

2) A. Cherubini and R. Odorico, Z. Physik C 47 (1990) 547

3) A. Cherubini and R. Odorico, Z. Physik C 53 (1992) 139; A. Cherubini and R. Odorico, Proc. of the Workshop on "Neural Networks: from Biology to High Energy Physics", June 5-14, 1991, Marciana Marina, Isola d'Elba (Italy), ETS Editrice, Pisa (1991), p. 515

4) L. Lönnblad et al., Nucl. Phys. B349 (1991) 675; C. Bortolotto et al., Nucl. Instr. Meth. A306 (1991) 459, Udine preprint 91/04/AA (1991); L. Bellantoni et al. (Wisconsin), Nucl. Instr. Met. A310, 618 (1991); J. Proriol et al. (Clermont-Ferrand), Proc. Elba NN Workshop (1991), p. 419; J. Jousset et al. (Clermont-Ferrand), L'Agelonde Workshop (January 1992); B. Brandl (Heidelberg), preprint HD IHEP 92-01 (1992); M. Los and N. DeGroot (NIKHEF-H), Proc. Elba NN Workshop (1991), p. 459, CERN NN Workshop (December 1991); P. Branchini et al. (INFN Sanità, Rome), L'Agelonde Workshop (January 1992), preprint INFN-ISS 92-1 (1992); F. Block (CERN), CERN NN Workshop (December 1991)

5) T. Kohonen, "Self Organization and Associative Memory", 2nd ed., Springer, Berlin (1988)

6) A. Cherubini and R. Odorico, LVQNET vers. 1.10, Bologna preprint DFUB 91/13 (1991), Comp. Phys. Comm., in press; R. Odorico, program NEURAL, in preparation

7) D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning Internal Representations by Error Propagation", in D.E. Rumelhart and J.L. McClelland (Eds.), "Parallel Distributed Processing: Explorations in the Microstructure of Cognition" (Vol. 1), MIT Press (1986), p. 318

8) L. Lönnblad et al., JETNET vers. 2.00, Lund preprint LU TP 91-18 (1991)

9) J. Proriol, MPL vers. 1.00. Clermont-Ferrand preprint (1991)

10) R.A. Fisher, Annals Eugenics 7 (1936) 179

11) M. Kendall, A. Stuart and J.K. Ord, "The Advanced Theory of Statistics", Vol. 3, 4th ed., C. Griffin & Co. Ltd., London

12) S.E. Fahlman and C. Lebiere, Carnegie Mellon University report CMU-CS-90-100 (1990)

13) K.J. Lang and M.J. Witbrock, "Learning to Tell Two Spirals Apart", in Proc. 1988 Connectionist Models Summer School, Morgan Kaufmann

14) T. Sjöstrand, Comp. Phys. Comm. 27, (1982) 243, 28 (1983) 229; JETNET vers. 7.3

15) R. Odorico, Comp. Phys. Comm. 32 (1984) 173; COJETS version 6.23, University of Bologna preprint DFUB 91/13 (1991), Comp. Phys. Comm., in press

16) P. Mazzanti and R. Odorico, University of Bologna report DFUB 92/1 (1992)

17) P. Mazzanti and R. Odorico, University of Bologna report DFUB 92/15 (1992)

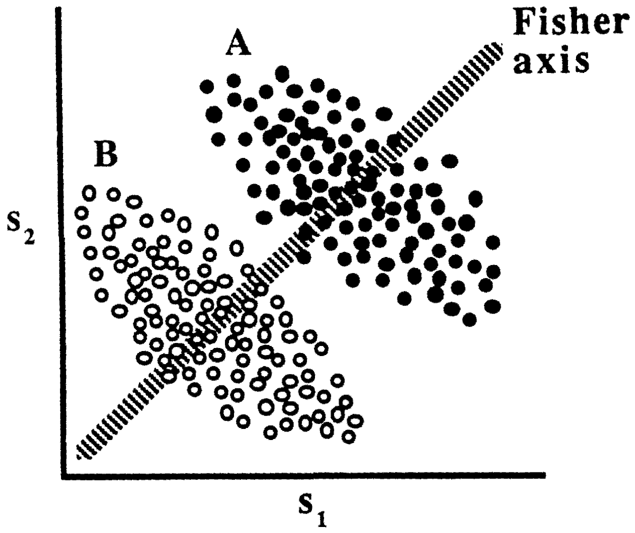

Fig. 1 - Classification example illustrating how Fisher's linear discrimination works.

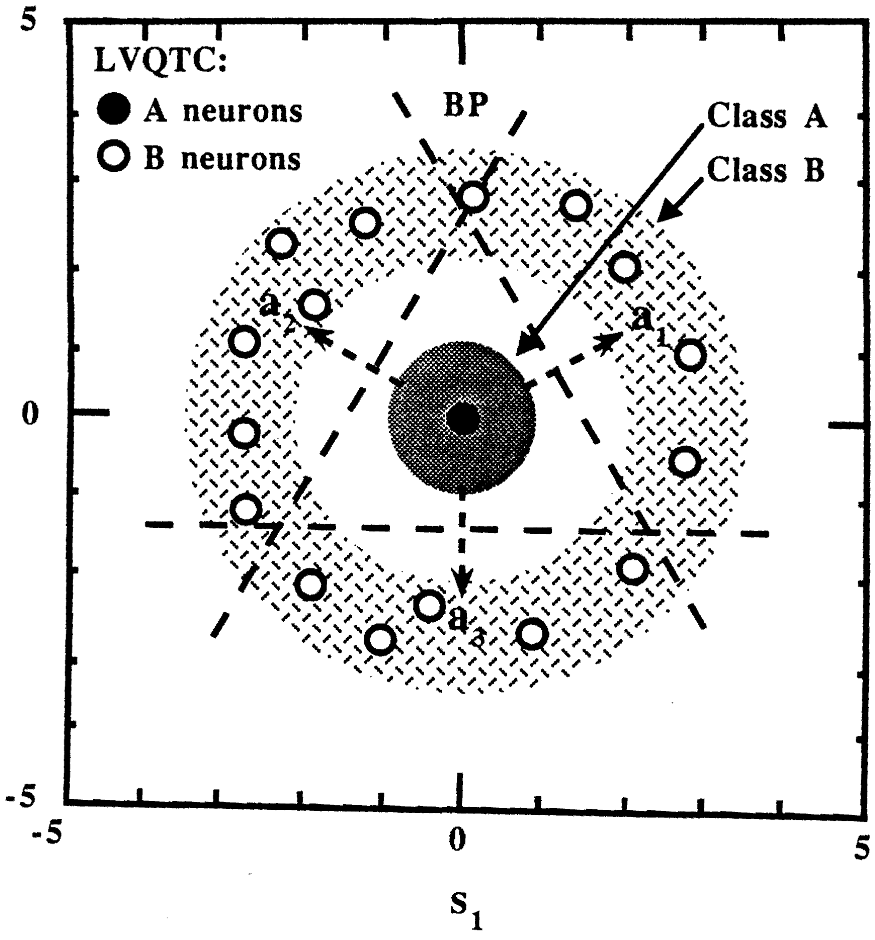

Fig. 2 - Classification example illustrating the way LVQTC and BP neural nets work. Circles represent LVQTC neurons. Straight lines represent BP hidden neurons.

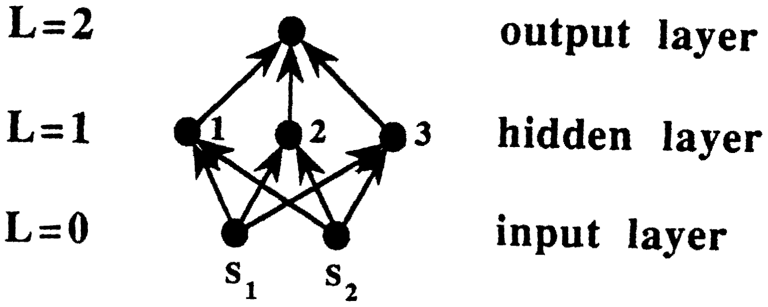

Fig. 3 - Architecture of the BP net used for the example of Fig. 2.

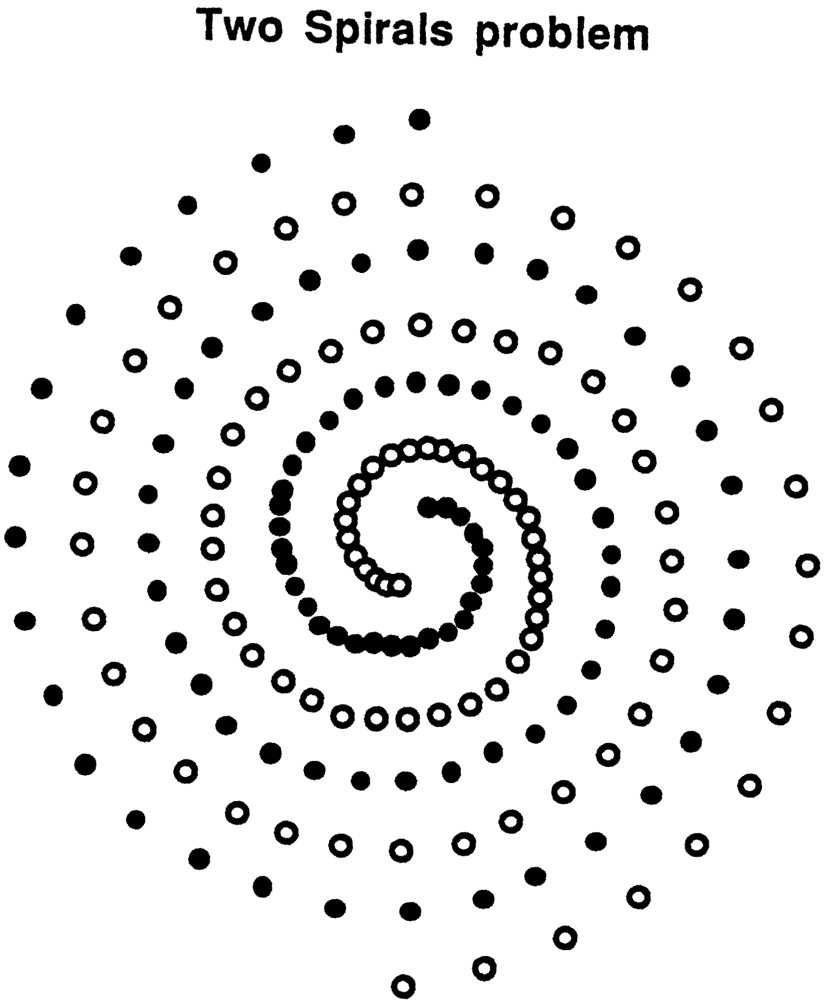

Fig. 4 - Two spirals problem, see text.

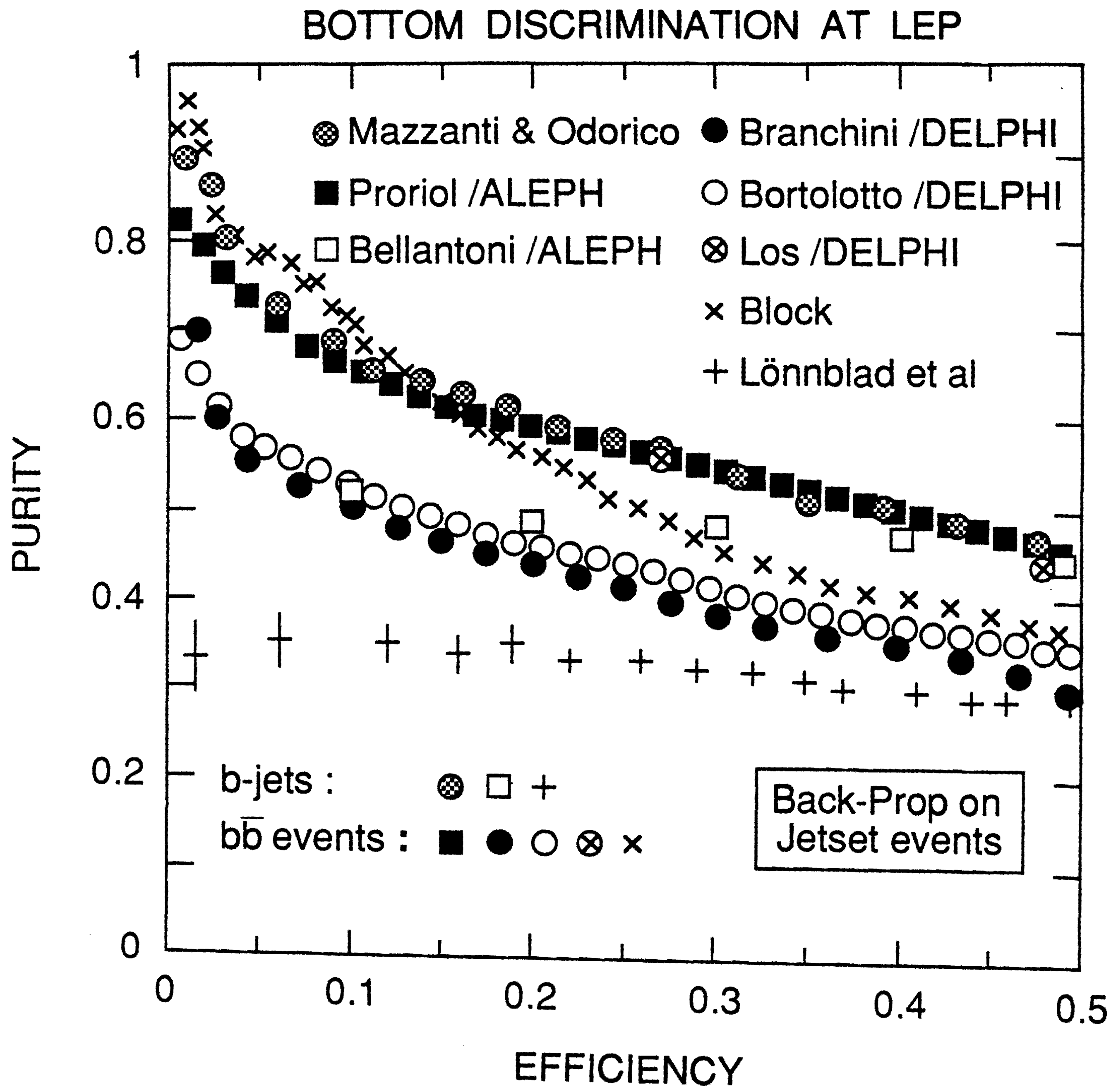

Fig. 5 - Comparison of purity vs efficiency results for b/non-b jets discrimination at LEP obtained using BP neural networks and JETSET event generator with various choices of shape variables.

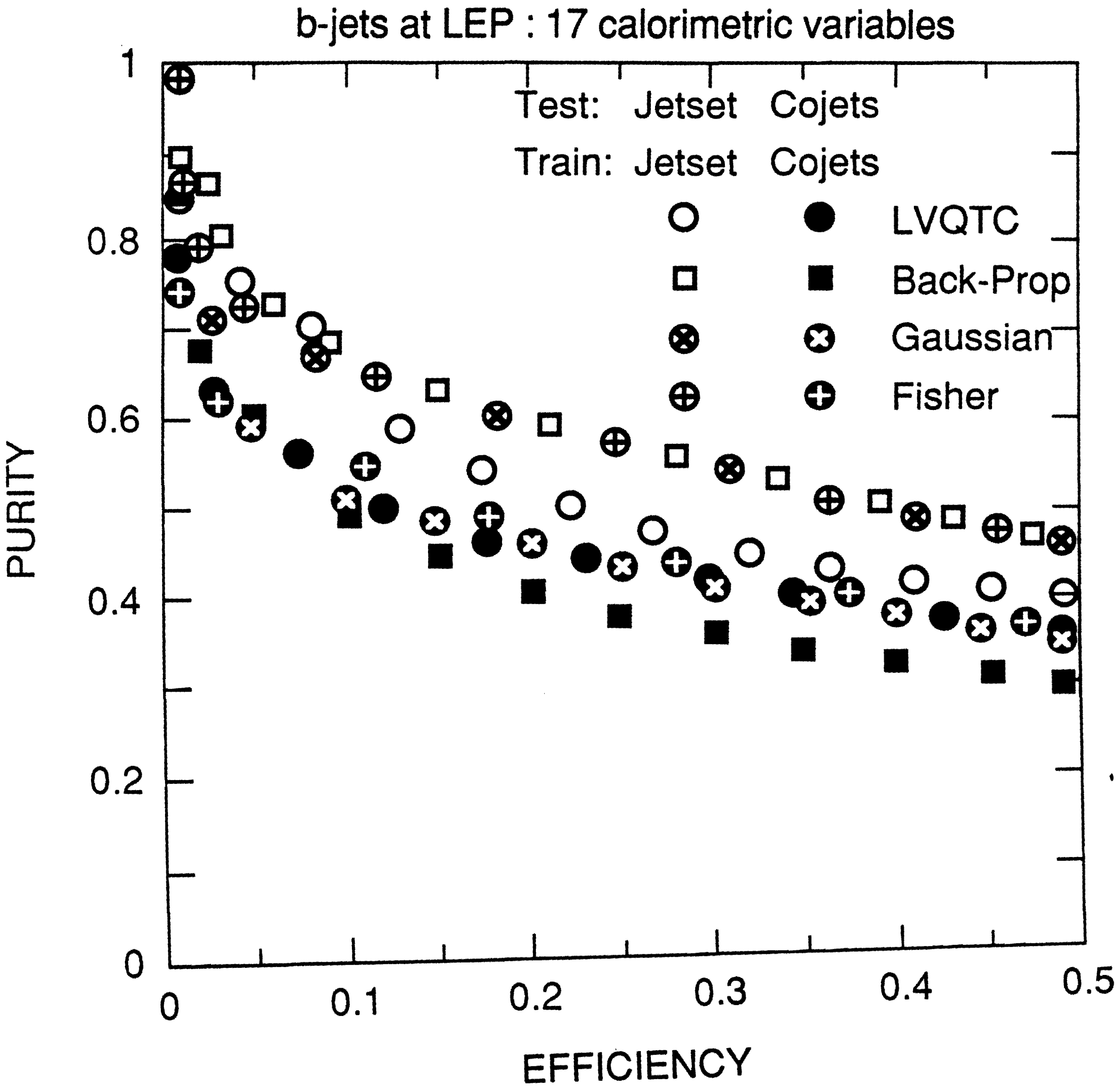

Fig. 6 - Purity versus efficiecy results for b jets at LEP using the same event generator, COJETS or JETSET, for training and testing.

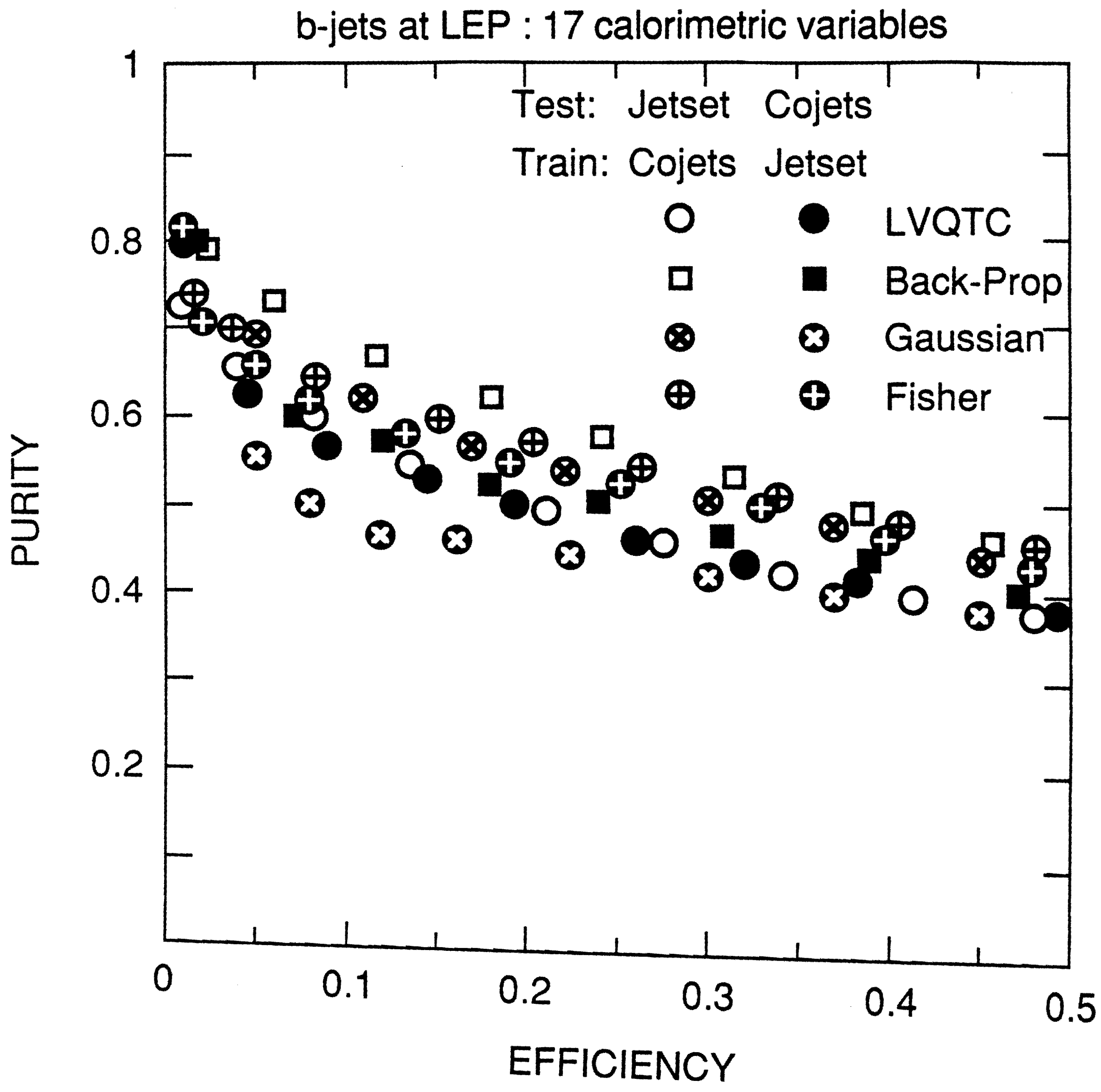

Fig. 7 - Purity versus efficiency results for b jets at LEP using different event generators (COJETS, JETSET) for training and testing.

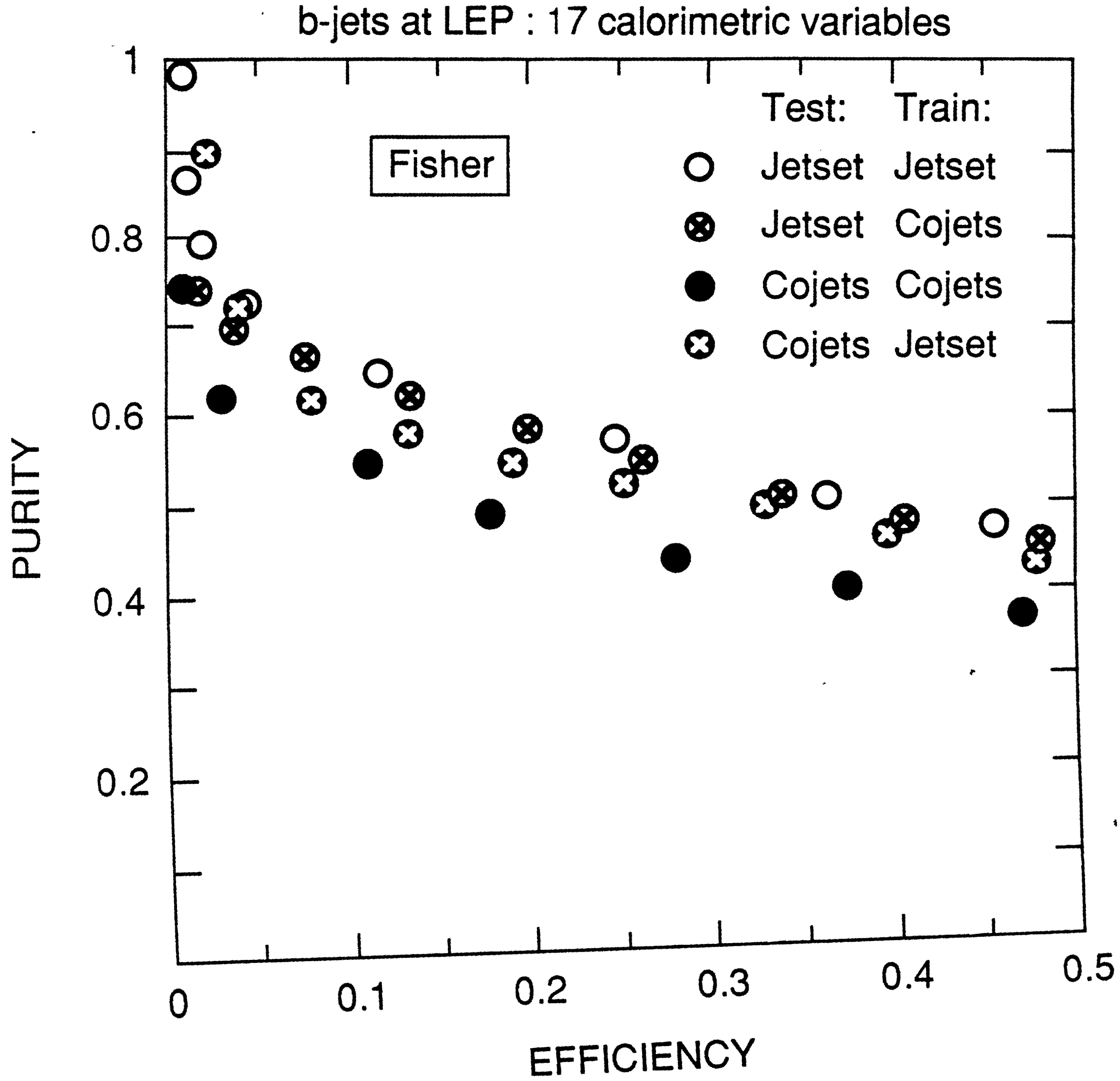

Fig. 8 - Purity versus efficiency results for b jets at LEP using the Fisher discriminant and all possible combinations of the event generators COJETS and JETSET for training and testing.

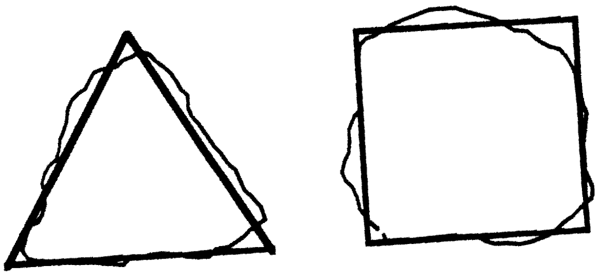

Fig. 9 - Simple example meant to illustrate that by training on ideal cases one may also get a better recognition of fuzzy ones.