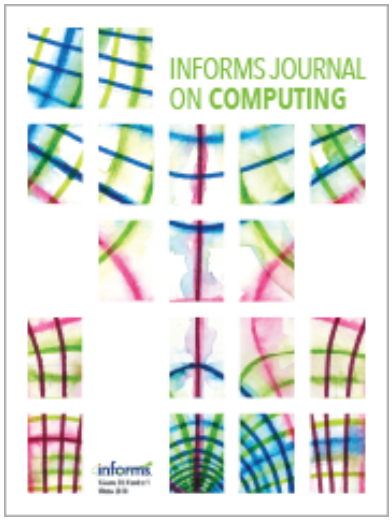

INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

The Supervised Normalized Cut Method for Detecting, Classifying, and Identifying Special Nuclear Materials

Yan T. Yang, Barak Fishbain, Dorit S. Hochbaum, Eric B. Norman, Erik Swanberg

To cite this article:
Yan T. Yang, Barak Fishbain, Dorit S. Hochbaum, Eric B. Norman, Erik Swanberg (2014) The Supervised Normalized Cut Method for Detecting, Classifying, and Identifying Special Nuclear Materials. INFORMS Journal on Computing 26(1):45-58. https://doi.org/10.1287/ijoc.1120.0546

INFORMS Journal on Computing
Vol. 26, No. 1, Winter 2014, pp. 45–58
ISSN 1091-9856 (print) | ISSN 1526-5528 (online)

http://dx.doi.org/10.1287/ijoc.1120.0546

The Supervised Normalized Cut Method for Detecting, Classifying, and Identifying Special Nuclear Materials

Yan T. Yang
Department of Industrial Engineering and Operations Research, University of California, Berkeley, Berkeley, California 94720, yxy128@berkeley.edu

Barak Fishbain
Faculty of Civil and Environmental Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel, fishbain@technion.ac.il

Dorit S. Hochbaum
Department of Industrial Engineering and Operations Research, University of California, Berkeley, Berkeley, California 94720, hochbaum@ieor.berkeley.edu

Eric B. Norman, Erik Swanberg
Department of Nuclear Engineering, University of California, Berkeley, Berkeley, California 94720
{ebnorman@lbl.gov, swany@nuc.berkeley.edu}

The detection of illicit nuclear materials is a major tool in preventing and deterring nuclear terrorism. The detection task is extremely difficult because of physical limitations of nuclear radiation detectors, shielding by intervening cargo materials, and the presence of background noise. We aim at enhancing the capabilities of detectors with algorithmic methods specifically tailored for nuclear data. This paper describes a novel graph-theory-based methodology for this task. This research considers for the first time the utilization of *supervised normalized cut* (SNC) for data mining and classification of measurements obtained from plastic scintillation detectors that are of particularly low resolution. Specifically, the situation considered here is for when both energy spectra and the time dependence of such data are acquired.

We present here a computational study, comparing the supervised normalized cut method with alternative classification methods based on support vector machine (SVM), specialized feature-reducing SVMs (i.e., 1-norm SVM, recursive feature elimination SVM, and Newton linear program SVM), and linear discriminant analysis (LDA). The study evaluates the performance of the suggested method in binary and multiple classification problems of nuclear data. The results demonstrate that the new approach is on par or superior in terms of accuracy and much better in computational complexity to SVM (with or without dimension or feature reduction) and LDA with principal components analysis as preprocessing. For binary and multiple classifications, the SNC method is more accurate, more robust, and is computationally more efficient by a factor of 2–80 than the SVM-based and LDA methods.

Key words: nuclear material detection; supervised learning; normalized cut; support vector machine; multiclassification

History: Accepted by Karen Aardal, Area Editor for Design and Analysis of Algorithms; received March 2011; revised July 2011, March 2012; accepted June 2012. Published online in *Articles in Advance* April 11, 2013.

1. Introduction

The detection of illicit nuclear materials is of great interest in the efforts to deter and prevent nuclear terrorism. Today's typical approaches to special nuclear material (SNM) detection primarily employ fixed inspection portals, installed at national borders, sea-ports, and traffic and railway checkpoints within the national interior. Although one can detect the presence of radioactive material using simple gamma-ray counting equipment, such as a Geiger counter, this creates a great deal of false-positive errors as some legitimate cargoes such as bananas, fertilizers, cat litter, tiles and ceramics (containing potassium, ^{40}K), smoke detectors (with americium, ^{241}Am), and colored glass (containing natural uranium) may also generate high radioactivity levels. Therefore, it is important to identify not only the presence of a radioactive material, but also its identity. One way of identifying the source is by examining the radiation's spectrum, the number of gamma rays detected at each energy interval, and finding the best match for that spectrum in a set of spectra

obtained from several known SNMs. With low-resolution detectors this task is challenging, even for human experts. It is therefore important to enhance the capabilities for identifying the nuclear material based on the radiation spectrum.

In comparing a given spectrum with that of a set of known SNMs the latter is used as a so-called training set. As such, the illicit SNM detection problem can be cast as a machine learning classification problem. The goal is to classify the target material examined by its acquired spectrum, or a set of spectra gathered by various sensors or in different time intervals, so as to generate information about the material and discern whether it poses a threat in a relatively quick manner.

In real-life scenarios of shipping cargo screening, training sets are usually used in several screening methods based on passive radiation counting. In these cases, the training set consists of spectra acquired by detectors when the content of the examined cargo is known. Thus, different containers known to contain a specific SNM, as well as containers with benign substances (such as bananas, colored glass, or without any radioactive material) are placed in front of the detector. For each of these materials a set of spectra is acquired and labeled accordingly. Upon an arrival of a new container with unknown content, a new set of spectra (with unknown labels) is acquired. The purpose of the classification is to group these unknown spectra with the best matched samples from the training set.

There are two methods of detection—passive and active interrogation. Passive interrogation measures a material's emitted radiation. As such, passive interrogation is limited by the rate and energy of natural radioactivities and their attenuation through shielding. Because of these shortcomings, the active interrogation alternative was proposed for the nuclear material detection task (Bertozzi and Ledoux 2005, Norman et al. 2004), especially for use on cargo at ports of entry. In active interrogation, the target is irradiated by bremsstrahlung X-rays (Bertozzi and Ledoux 2005) or highly penetrating neutrons (Norman et al. 2004) to produce spectra that are characteristic to each SNM. Still, even with active interrogation the identification of nuclear materials by its acquired spectra is difficult because of physical limitations of nuclear radiation detectors, the presence of background noise, and intervening shielding materials.

Different types of detectors deliver spectra with different merits. High purity germanium (HPGe) gamma detectors have excellent energy resolution. However, they are expensive and require cryogenic cooling, making field use cumbersome. Sodium iodide (NaI) detectors are less expensive and do not require cooling, and the quality of the delivered spectra is less. Plastic scintillators are detectors that do not require cooling nor high maintenance and as such are more practical for nuclear field detection applications. The trade-off is that these detectors produce low-resolution spectra that are very challenging to analyze. Even for human experts the differentiation between the spectra produced by plutonium and those produced by uranium is subtle. Data mining and pattern recognition techniques tailored for nuclear data have the potential of enhancing the ability to differentiate between different SNMs and make up for the hardware shortcomings. Several such methods reported in the literature include artificial neural networks (Kangas et al. 2008), naive Bayesian framework classification (Carpenter et al. 2010), support vector machine (Gentile 2010), and graph-theory-based techniques (Mihailescu et al. 2010). However, all these tools were used on measurements recorded by high-resolution HPGe detectors. Other than the aforementioned references, we find no systematic efforts in the literature to construct a robust automated technique to identify nuclear threats. In addition to the problems such as the effect of intervening cargo on signal distortion, one reason for this is the lack of a comprehensive data set of SNM spectral signatures.

Swanberg et al. (2009) have recently acquired spectral data from a plastic scintillator detector by active interrogation. They produced the only data set currently available that presents spectra of SNMs acquired by a low-resolution plastic detector. This data set consists of spectra of plutonium, uranium, latite (rock material), and blank. The challenging task within the scope of this work is to distinguish between plutonium and uranium. Recent studies (Marrs et al. 2008) demonstrated that a classification of plutonium's versus uranium's spectra can be accomplished when high-resolution HPGe detectors are employed. Here we show that this classification task can be accomplished by employing appropriate data-mining techniques on low-resolution spectra acquired by plastic detectors.

The data sets obtained by Swanberg et al. (2009) are the basis for our computational study, which presents, for the first time, a graph-theory-based method for classifying low-resolution spectra. This provides preliminary evidence that the use of the inexpensive and low-maintenance plastic detectors with data-mining techniques for the purpose of detecting illicit nuclear material is practical. Furthermore, our results appear to be promising enough to encourage the testing of the technique we propose for this task, the *supervised normalized cut*, in other areas of data mining and classification contexts as well.

1.1. Taxonomy of Machine Learning Problems

Machine learning techniques can be categorized either as supervised (inductive) learning, i.e., inferring a

function from supervised training data (Caruana and Niculescu-Mizil 2006), or as *un*supervised learning, thus the learner is given only unlabeled examples (Duda et al. 2001).

In supervised learning, one tries to infer a functional relation $y = f(x)$ from a training set $(\vec{x}_1, y_1), \ldots, (\vec{x}_m, y_m)$, where $\vec{x}$'s are feature vectors of data points, and the y's are the known labels assigned to the vectors of the training set. In this study, the feature vectors represent the spectra from the dynamic decay of fission products of ^{239}Pu (plutonium) and ^{235}U (uranium), the radioactivities induced in latite, and the radioactivities induced in background materials; the y's are plutonium, uranium, latite, and blank. We seek a classifier f that can accurately predict the labels $y_{m+1}, y_{m+2}, \ldots$ for future input vectors $\vec{x}_{m+1}, \vec{x}_{m+2}, \ldots$ This paradigm is common to many other problems that appear in different areas, such as computer vision (Carneiro et al. 2007), geostatistics (Dowd and Pardo-Iguzquiza 2005), credit scoring (West 2000), and biometric identification (Zhang et al. 2008).

Unsupervised learning groups data points into different clusters, based on some measure of sameness, *without using prior training data*. The general procedure is to map all the data points into different clusters such that some criteria are optimized. The supervised classification method proposed here relies on the conversion of an unsupervised technique to a supervised context.

Because the number of illicit radioactive substances is finite and well defined (International Atomic Energy Agency 2007), the respective multiclassification problem is to classify into K classes, where K is known in advance. Here we solve the multiclassification problem by repeated calls to a binary classification subroutine. This is a common practice in many multiclassification techniques (Crammer and Singer 2002, Goldschmidt and Hochbaum 1994, Kotsiantis 2007).

Examples of criteria for binary clustering, also referred to as bipartitioning, are (i) minimum cut (Ford and Fulkerson 1956), (ii) ratio regions (presented by Cox et al. 1996), (iii) the normalized cut (suggested by Shi and Malik 2000), and (iv) a variant of normalized cut NC′ (studied by Hochbaum 2010b).

We choose to implement the NC′ criterion because it provides better quality solutions for image clustering and pattern recognition applications than other, commonly used techniques (Hochbaum 2010b, Sharon et al. 2006). Furthermore, NC′ was shown to be efficiently solvable (Hochbaum 2010b) and as such it is a good candidate for solving clustering problems in short running times.

Because NC′ does not require any prior information on the data, it is an unsupervised technique. The method described here, called the *supervised normalized cut* (SNC), utilizes the NC′ criterion in a supervised context. The method incorporates training data and is used for the purpose of detecting, classifying, and identifying spectra acquired by low-resolution radiation detectors.

The SNC method is compared here with two traditional data mining techniques—three variants of support vector machine (SVM) and linear discriminant analysis (LDA). The results of this study suggest that SNC is preferred to SVM (with or without feature reduction) for the task of nuclear material detection and might be better suited than these techniques for other classification problems.

The paper is organized as follows: §2 describes the SNC method. Section 3 describes how the data were generated and explores different ways to present the acquired data. Section 4 presents the classification results, both in terms of accuracy and running times. Section 5 concludes the paper.

2. The Variant of Normalized Cut and Supervised Normalized Cut

2.1. Binary Classification

The binary classification problem is formalized as a graph bipartitioning problem. An undirected (complete) graph $G = (V, E)$ is constructed, where each node $v \in V$ corresponds to a data point—in our case, a feature vector (see §3.2) associated with a set of spectra acquired from a material sample. There is an edge in the graph for each pair of nodes i and j associated with a weight w_{ij} that corresponds to the similarity between the feature vectors associated with nodes i and j. Higher similarity is associated with higher weights.

The following notation facilitates the presentation of the bipartitioning optimization problem: A bipartition of the graph is called a *cut*, $(S, \bar{S}) = \{[i, j] \mid i \in S, j \in \bar{S}\}$, where $\bar{S} = V \backslash S$. We denote the *capacity of a cut* $(S, \bar{S})$ as

$$C(S, \bar{S}) = \sum_{i \in S,\, j \in \bar{S},\, [i,j] \in E} w_{ij}. \tag{1a}$$

The *capacity of a set*, $S \subset V$, is denoted by

$$C(S, S) = \sum_{i,\, j \in S,\, [i,j] \in E} w_{ij}. \tag{1b}$$

The goal of NC′ is to find a nonempty set $S^* \subset V$, so that the capacity of the cut, or the similarity between S^* and its complement, divided by the capacity of the set S^*, which is the similarity of the feature vectors within the set, is minimized. The set S^* is called a *source* set, and its complement is called a *sink* set.

The mathematical objective of this goal can be written as (Hochbaum 2010b, Sharon et al. 2006)

$$NC'(S^*) = \min_{S \subset V} \frac{C(S, \bar{S})}{C(S, S)}. \tag{2}$$

As noted before, optimization problem (2) was shown to be solvable in polynomial time (Hochbaum 2010b) with a minimum cut procedure on an associated graph.

The NC′ solution procedure requires to assign, in advance, a single node that will be included in the source S (or sink $\bar{S}$) set (see Hochbaum 2010b for details). This node is referred to as a *seed node*. Here, we exploit the seed node mechanism to force a priori the training data to be in either the source S or in the sink $\bar{S}$, based on the material from which they were acquired. Specifically, the input consists of three sets: two sets of nodes, A and B, which are associated with feature vectors acquired from two different known materials, M^1 and M^2, and a third set I corresponding to feature vectors acquired from an unknown material or materials. The goal of the binary classification problem is to associate each feature vector in I with either M^1 or M^2.

The input to the classification problem is the complete graph $G = (V, E)$ defined on the set of objects $V = A \cup B \cup I$ and the similarity weights associated with each pair of nodes (edge) $[i, j] \in E$. Two nodes s and t are added to the graph with an arc of infinite weight from s to each node in A and from each node in B to t. On this graph we seek a partition that minimizes the NC′ criterion so that $s \in S$, $t \in \bar{S}$. The nodes in I that end up in S are classified as A, i.e., acquired from material M^1; and nodes in I that end in $\bar{S}$ are classified as B, thus acquired from M^2. This process is illustrated in Figure 1.

The adjustment of NC′ to a supervised context, as described, is a new *supervised* classification methodology that takes advantage of the solvability of NC′ and broadens the application of NC′ to a wider class of problems.

As previously mentioned, the efficiency of NC′ algorithm was established in Hochbaum (2010b), where it was shown that NC′ is solvable in the running time of a minimum s, t-cut problem, which is strongly polynomial and combinatorial. In the context of SNC, the only additional step before performing NC′ solution procedure is to separate and assign training data to belong to the source or the sink set, which does not affect the time complexity. Thus SNC is efficient, running in polynomial time.

2.2. Multiclassification

The binary classification with NC′ is used here as a subroutine for solving multiclassification problems involving three or more different classes. Multiclassification is more realistic in the context of nuclear threat detection, as it is necessary to identify, e.g., the contents of cargo, as one of an array of possible materials.

For multiclassification we utilize a scheme generally referred to as one-versus-all decomposition (e.g., Duan and Keerthi 2005, Rifkin and Klautau 2004). For a problem with K different classes, we create K different binary problems. The kth binary problem is to classify the unknown nodes I into two classes—material M^k, or not-M^k, E (stands for *Else*). Each node is classified by K binary classifiers. The label of the node is determined to be class k, if it was classified as M^k. If the node was classified as material more than once, all possible materials are reported. If the node is classified as E for all k, then the label is undecided.

In the case where all the nodes in I are acquired from the same container, a voting can be used to determine the final grouping of all nodes in I. In this case, the label of a node is determined to be the class k that has the highest score. If there is more than one class with the highest score, the tie is broken arbitrarily or both materials are reported as possible classification. If the highest score is zero, then the label of the node is undecided.

Figure 2 demonstrates multiclassification to four possible materials or classes A, B, C, and D. Three unclassified data points need to be classified. Training data are provided for each class as shown in Figure 2(i). Figure 2(a)–(d) are the binary classifications for each class A–D. Figure 2(r) shows the combined

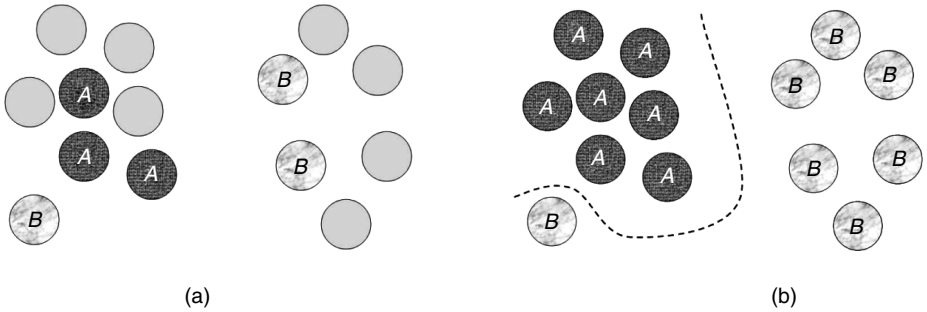

Figure 1 (a) The Input with the Training Sets A (Black) and B (Light Gray) and the Unclassified Nodes C (Gray); (b) The Solution: The Two Sets Are Separated by a Minimum Cut

Note. The set on the left consists of the nodes, classified as A nodes, forms the set S; and the set on the right of the B nodes is $\bar{S}$, where the similarity within S and the dissimilarity between the two sets are high.

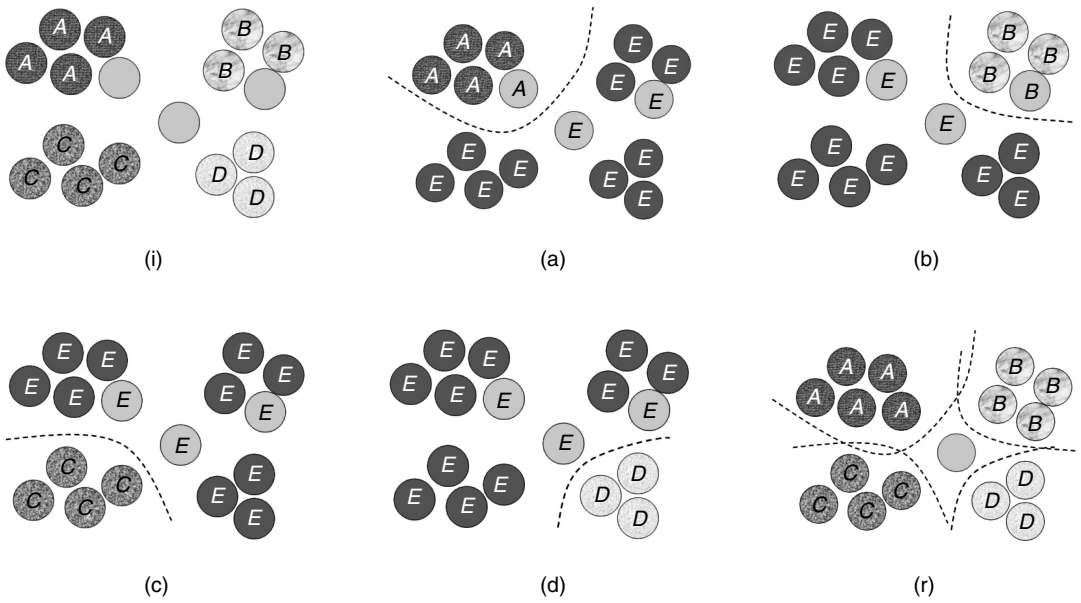

Figure 2 **Multiclassification: (i) A, B, C, and D are Four Distinct Materials or Classes. (a)–(d) are A–D Decomposed Binary Classifications. Each Classification Gives a Label to the Unknown Points. (r) The Final Multiclassification Combines All the Labels for a Given Unknown Point**

results of all subproblems; two of the unknown points are successfully classified to A and B. The middle unclassified node is undecided, because it has E for all its labels.

3. Data and Experimental Setup

The measurement data for the nuclear classification problem were acquired in a controlled environment with plastic detectors. We use here a data set of active interrogation of plutonium and uranium made available by Swanberg et al. (2009). In this experiment, a sample of either blank, 0.19 grams of ^{235}U, 0.568 grams of ^{239}Pu, or 3 grams of latite, an igneous rock material was placed in a cave and irradiated for 30 seconds with neutrons generated by the 88-inch cyclotron at Lawrence Berkeley Laboratory (Kelly 1962). When irradiated with neutrons, materials may become radioactive or undergo nuclear fission. Activation products and fission products from different materials have different characteristic gamma rays and decay times. These are the characteristics that we use as the pattern distinguishing a specific nuclear material from others.

The target was exposed to the detectors for a total of 25 seconds and each 2.5-second interval yields a cumulative energy spectrum measurement for that interval. The detector system measured energies in the range from approximately 100 keV to 14 MeV using 1,024 channels.

The data set used in our experiments consists of the measurements reported in Swanberg et al. (2009) and additional measurements that were made available by the same authors. The additional data include 10 measurements for each run. In total, 275 runs were conducted: 20 with blank; 22 with latite; 92 with uranium ^{235}U; and 140 with plutonium ^{239}Pu; resulting in a total of 2,750 acquired spectra.

3.1. The Detector Live Time

During the data acquisition, the operator set the detector to a nominal run time of 2.5 seconds. The actual acquisition time of the detector, however, depends on the particularities of each run. Specifically, when the gamma rays arrive at high frequency, the detector does not process all of them because of hardware limitations. Therefore the length of the actual run time, or the so-called *live time*, is shorter than the nominal run time or real time. To correct this inherent hardware bias, we adjust for the live times of the detector in each run by rescaling each gamma-ray count by the instrument's live time produced with the data. This means that the gamma-ray counts in each spectrum are scaled (divided) by the live time associated with that spectrum measurement. The results presented here use such scaled data.

3.2. Feature Vectors and Data Analysis

Each target was placed in front of the detector for a run of 25 seconds. The spectrum—the energy histogram of the gamma ray received by the detector—was recorded every 2.5 seconds. Each entry in the histogram corresponds to a different energy band (channel). Hence, 10 spectral measurements taken at consecutive periods of time were produced in every run.

The obtained data can be regarded as a two-dimensional array composed of the gamma-ray counts for

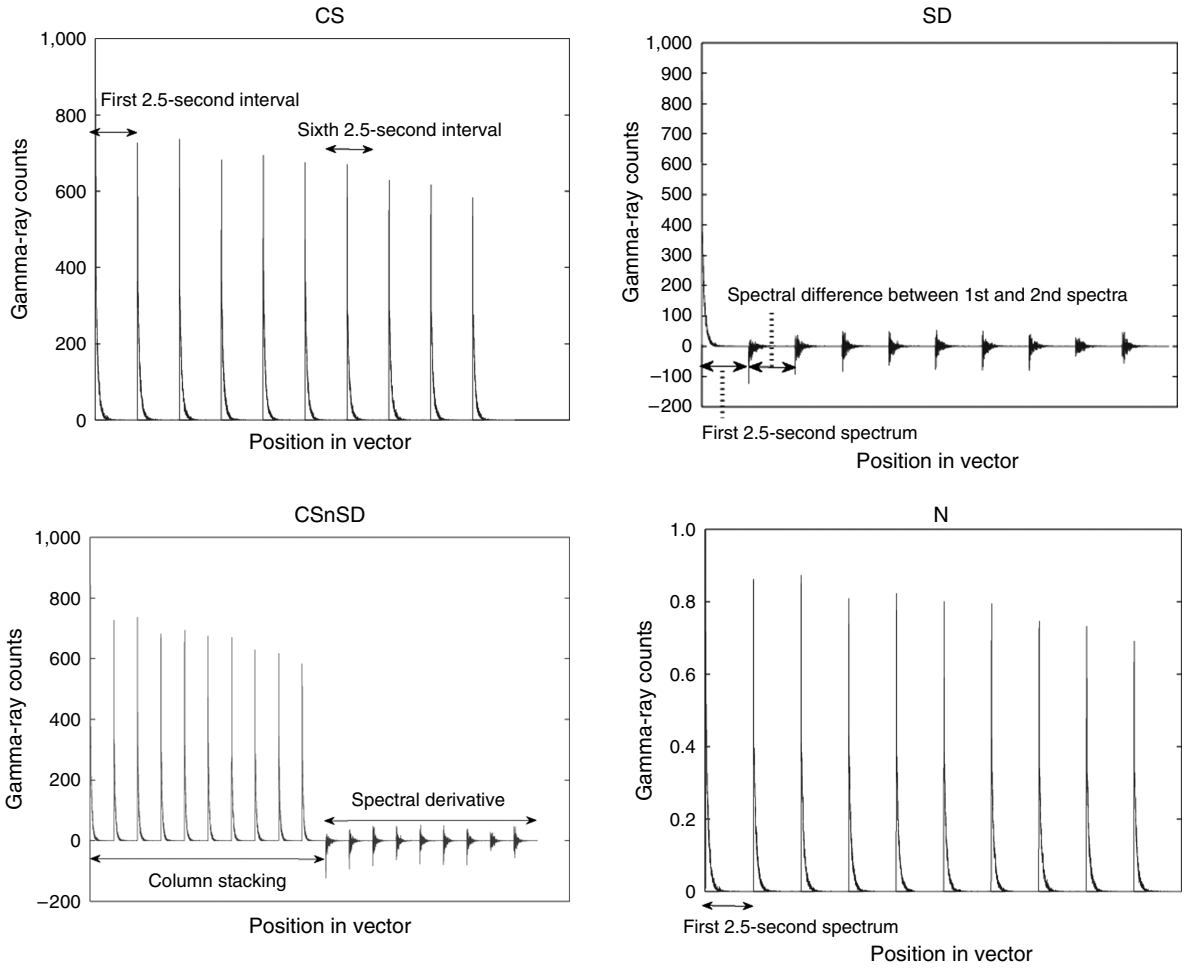

Figure 3 **Feature Vectors Produced by Column Stacking (CS), Spectral Difference (SD), Column Stacking and Spectral Difference (CSnSD), and Normalization of the Data (N). For CS, SD, and N, There are 10,240 Indices in the Vectors; for CSnSD, There are 19,456 Indices**

each energy channel in each of the 10 given measurements. For 1,024 energy channels and 10 consecutive measurements, each such sample is an array with 1,024 columns and 10 rows. Each row vector is denoted by $\vec{s}_i$, for $i = [1, \ldots, 10]$, where $\vec{s}_1$ corresponds to the first 2.5-second interval and $\vec{s}_i$ corresponds to the ith 2.5-second interval. To convert this $2D$ array into a feature vector, four different methods are considered:

1. Column stacking (CS). The vectors $\vec{s}_1, \ldots, \vec{s}_{10}$ are concatenated to a vector of length 10,240. The feature vector produced is $(\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_{10})$ as in Figure 3 (top left).

2. Spectral difference (SD). The purpose of this method is to emphasize the radioactive decay captured in the temporal domain. To do that, we concatenate the first spectrum vector, followed by the difference between the second spectrum vector and the first spectrum vector, and in general the ith entry is the difference between the the ith vector and the $(i-1)$th vector. The feature vector produced is then $(\vec{s}_1, \vec{s}_2 - \vec{s}_1, \vec{s}_3 - \vec{s}_2, \ldots, \vec{s}_{10} - \vec{s}_9)$ as in Figure 3 (top right).

3. Column stacking and spectral difference (CSnSD). Here we join (concatenate) the feature vectors resulting from CS and SD into a single feature vector. The feature vector produced is $(\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_{10}, \vec{s}_2 - \vec{s}_1, \vec{s}_3 - \vec{s}_2, \ldots, \vec{s}_{10} - \vec{s}_9)$ as in Figure 3 (bottom left).

4. Normalization (N). To remove the dependence on absolute counts of the spectra, which grow with the sample quantity and weight, while preserving the general patterns, we *normalize* the CS feature vector. This is done by dividing the entries of the feature vector by the largest entry of that vector. Let $s_{\max} = \max_{i=1,\ldots,10;\, j=1,\ldots,1,024} (\vec{s}_i)_j$, then the N feature vector is $(\vec{s}_1/s_{\max}, \vec{s}_2/s_{\max}, \ldots, \vec{s}_{10}/s_{\max})$ as in Figure 3 (bottom right).

The intuition behind using the previous four methods is that they are simple and the first step attempt to incorporate both spatial and temporal dimensions of our data set. Using these feature vectors serves two goals: (i) converting the $2D$ data to $1D$, and (ii) capturing the local temporal changes of spectra. To achieve these two goals, one could apply more sophisticated signal processing methods, such

as pyramid transform, discrete wavelets transform, or discrete cosine transform. A possible advantage of using these signal processing transformations is that they may result in better posed covariance matrices of the data, which can potentially improve the classification accuracy of principal components analysis (PCA) and LDA. However, these signal transformations require more computation time than the simple methods utilized here, and the trade-off between the added computational complexity and better accuracy should be investigated. Within the scope of this paper we have utilized only the aforementioned feature-vectors construction methods, which are simple and computationally efficient.

4. Results

In this section, results concerning different aspects of our method are presented: Section 4.1 establishes standards for measuring the quality of a classification technique to compare across different methods. Section 4.2 describes the classification methods used here to compare to SNC and addresses practical aspects of classification methods such as feature selection. In §4.3, classification methods, including various versions of SVM, LDA, and SNC are presented for the Swanberg et al. (2009) nuclear data. Section 4.4 gives a more detailed account on the results of SNC. Section 4.5 compares the methods in terms of running times. Multiclassification results are shown in §4.6. Finally, §4.7 presents the influence of different constructions of feature vectors on the different algorithms running time.

As described in §1.1, a classification procedure consists of two stages: "training" stage, where one tries to infer a functional relation (i.e., $y = f(x)$) between a training set $(\vec{x}_1, y_1), \ldots, (\vec{x}_m, y_m)$ and its known labels y's; and "testing" phase in which the labels $y_{m+1}, y_{m+2}, \ldots$ of unlabeled input vectors $\vec{x}_{m+1}, \vec{x}_{m+2}, \ldots$ are estimated. The outcome of the training phase is a classifier that is used for labeling new data points in the testing phase. For SVM and PCA the testing is very quick. We produce an analogous testing phase for SNC. The output of the training phase of SNC is a bipartition of the training and other data points. When a new data point becomes available the testing phase assigns that point to the side of the bipartition that increases the least objective value of NC′ criterion ($C(S, \bar{S})/C(S, S)$, the objective in Equation (2)). This process involves a comparison of few values and it is at least as fast as the testing phase for PCA and SVM.

Our study establishes that SNC is faster than SVM and PCA in the training phase, and all three methods are almost instantaneous, and thus on par with each other, in the testing phase. Because SNC is also at least as accurate as the other methods, it should be the preferred technique. The speed of the training phase with SNC makes the recalibration of the algorithm with changing conditions easy and fast, and therefore can be done more frequently than if one is to use the other techniques. Therefore with SNC it is possible to retain a more accurate and updated classification model.

4.1. Quality of Classification

In the machine learning community, the quality of a classification method is generally measured by simulation applying the method to a known data set. Here, we use an extended version of the data reported in Swanberg et al. (2009) consisting of 275 data points in the form of feature vectors, each labeled with its underlying material: blank (20 samples), latite (22 samples), ^{235}U (92 samples), or ^{239}Pu (140 samples).

To study the performance of the method we apply *random subsampling*, which divides the data set into two subsets: training and testing. The training data are used to construct a classifier, and the labels of the testing data are hidden, i.e., we pretend that the labels are unknown. The classifier provides *predicted* labels that are then compared to the true labels. The accuracy of a classifier is the fraction of the correct predictions across all testing data. For statistical significance purposes, this subsampling and classification are repeated 100 times. Each time the training and testing sets are resampled. The *accuracy of a classification method on a certain data set* can then be defined as the mean accuracy of these 100 runs. In addition, standard deviation and 95% confidence interval of these runs can be calculated. These measure the *consistency of a classification method on a certain data set*. The lower the standard deviation and confidence interval, the more consistent the method.

Random subsampling can involve different training-testing ratios, e.g., 40%–60%. A 40%–60% ratio means that 40% of the total data are used for training and the other 60% are used for testing. In our experiments we used the following ratios: 50%–50%, 40%–60%, 30%–70%, 20%–80%, and 10%–90%. As the size of the training data decreases, less information is provided to construct the corresponding classifier. Thus the accuracy of the classifier decreases. A more *robust* classification method is one that is less affected by the decreasing size of the training data.

4.2. Classification Methods

4.2.1. Features Selection and Data Dimension Reduction. Feature selection approaches try to find a subset of the original variables (also called features or attributes) that can best explain the data. The underlying idea is that other variables do not contribute to the understanding of the data and introduce noise. Therefore classification done in the reduced space may be more accurate than in the original space.

PCA is a routinely used tool for reducing the data space's dimension. The underlying paradigm of PCA is that an orthogonal linear transformation is performed on the data to a new coordinate system with the properties that the first coordinate, the so-called first principal component, contains the greatest variance by any projection of the data; the second coordinate contains the second greatest variance and so on. PCA essentially rotates the data points around its mean and moves variance into the first few dimensions as much as possible. The remaining dimensions contain negligible amounts of variance and are omitted, with a relatively small loss of information. Thus PCA is often used as dimensionality reduction.

To evaluate the principal components, one has to compute the covariance matrix for all acquired spectra. Because the feature vectors used here contain more than 11,200 coefficients each (see §3.2), finding this covariance matrix and its eigenvectors is computationally intractable. When evaluating smaller feature vectors (with 5,000 coefficients) the SNC method is 150 times faster than PCA. In addition, the gain in accuracy results from applying PCA before applying SVM is less than 1%. Therefore, for the task at hand, the use of PCA in this context is unlikely to improve the overall quality of the detection while significantly slowing down the detection speed.

4.2.2. Support Vector Machine. SVM is a widely accepted tool for classification in many fields, including machine learning (Fung and Mangasarian 2005), communication and mobile computing (Wang et al. 2010), biology (Ding et al. 2007), economics (Hua et al. 2007), nuclear applications (Carpenter et al. 2010), and X-ray spectrometry (Luo 2006).

For using SVM optimally one has to choose the type of kernel and its associated parameters. To this end we compare SNC to SVM with polynomial and radial basis function (RBF) kernels. The polynomial kernel's parameter is the degree of the polynomial d, and RBF uses a derivative parameter σ. Another parameter that applies for both kernels is the soft margin penalty C (see Burges 1998, Cristianni and Shawe-Taylor 2000, for details).

The selection of the SVM's parameters follows an exhaustive search. The work of Hastie et al. (2004) provides some guidance on how to search for optimal parameters of SVM. It has been shown that the range of SVM solutions with respect to the parameters does not grow beyond a certain value and degenerates quite poorly for very large and very small values of C. We thus tune the parameters of SVM by setting them in a grid $\{2^{-7}, 2^{-6}, 2^{-5}, \ldots, 2^{6}, 2^{7}\}$, the searching range used in Mangasarian and Wild (2007, 2008). Note that the SVM parameters tuning times are *not* included in the computation times that are used to compare the different methods.

The SVM classification is then performed for all possible parameters' combinations. The parameters' set that produces the best classification results is the one used for the evaluation. This tuning procedure is different than that of using a separate validation set or cross validation, and gives the best possible accuracy of a particular SVM classifier (Burges 1998, Cristianni and Shawe-Taylor 2000). The implementation package used is LIBSVM (Chang and Lin 2001). It is important to note that although these tuning times are not included in our run-time comparison, the tuning process is time consuming.

SVM-based methods that incorporate the feature selection process as an inherent part rather than running a generic PCA as a preprocessing stage can be found in the literature. These SVM procedures not only produce the classifier (as the regular SVM procedure does), but also the subset of features that are used for constructing this classifier. These include (1) a feature-reducing linear kernel 1-norm SVM (SVM-1) (Zhu et al. 2003); (2) a recursive feature elimination SVM (SVM-RFE) (Guyon et al. 2002); and (3) a feature-reducing Newton method for LPSVM (SVM-NLP) (Fung and Mangasarian 2004). These methods are shown to improve SVM's prediction power by removing features that are of the least relevance. SVM-1 is implemented by using MATLAB function `svmtrain` with 1-norm option. SVM-RFE is run with modified MATLAB code from Resson et al. (2009) (the original code has extra functionalities that are not used here). LPSVM was downloaded from Fung and Mangasarian (2002). All SVM-related parameters were tuned according to the suggestion in Fung and Mangasarian (2005, §4.2.2). The SVM-RFE method requires an additional parameter—the number of remaining features. We tune this parameter by exhaustive search in the space $\{2^{1}, 2^{2}, 2^{3}, \ldots, 2^{13}\}$.

4.2.3. Linear Discriminant Analysis. LDA, like PCA, aims at finding a spanning space for the data. PCA constructs the spanning space (i.e., the principal components) so the first principal component has the largest possible variance (i.e., accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it be orthogonal to (i.e., uncorrelated with) the preceding components. LDA receives as input a data set and a label for each data point in the set. The goal of LDA is to find a spanning space that accounts for as much of the variability between classes (or labels) rather than between all data points as PCA (Martinez and Kak 2001, McLachlan 2004). The resulting linear combination can also be used as a linear classifier. However, because the dimensionality of our data (the combined number of energy channels) is far larger than the number of data points, Geisinger (2010) suggests, in

cases like this, to perform PCA first before using LDA for classification—we take this approach and combine PCA with LDA.

To compare PCA-LDA, SVMs, and SNC, we use the four types of feature vectors CS, SD, CSnSD, N described in §3.2. For PCA-LDA, the reduced version of PCA—mentioned earlier for the reason of computational cost—is used: the PCA is performed first by centering data points and adding first few important principal components until the total variance accounted for is just above 80%. Then the standard LDA is used for the training classifier and classifying the test data. One note here is that the data is not scaled before PCA is applied; this is because by scaling data, in many instances, the resulting covariance matrix becomes singular for the training data from our data set—thus LDA could not proceed. Therefore, the decision is made to use only centered data while performing PCA before LDA.

4.3. Binary Classification

The performance of the binary classification method, SNC, is tested here. It is compared to two common classification methods: SVM and LDA, as well as the three specialized feature-reducing SVMs: 1-norm SVM (SVM-1), recursive feature elimination SVM (SVM-RFE), and Newton method LPSVM (SVM-NLP).

To solve SNC, the graph construction is written in MATLAB. The resulting minimum cut problem is solved with Hochbaum's PseudoFlow algorithm (HPF) the implementation of which is downloaded from Hochbaum (2010a). The similarity between two feature vectors v_i and v_j is quantified by

$$w_{ij} = \frac{1}{||v_i - v_j||_2 + \epsilon},$$

for $0 < \epsilon \ll 1$.

Table 1 displays the accuracy and precision of the supervised normalized cut for varying training-testing ratios with different types of feature vectors. For 50%–50% ratio, all four types of feature vectors produce similar results. These results for the different feature vectors are statistically identical as confirmed by ANOVA test failing to reject null hypothesis at 95% significant level for every pair of vectors. However, as the training proportion decreases, the CS feature vector gives the highest accuracy. This behavior also characterized the standard deviation and the 95% confidence interval, which are also the best for CS.

Table 2 details the corresponding results for SVM and specialized SVMs when using either RBF or polynomial kernels. Each of these kernels takes user defined parameters including a parameter for soft margin penalty C. In addition, RBF uses a derivative parameter σ. For SVM-RFE, the optimal number of features is also displayed (NumFeat) and for SVM-NLP, another parameter ν is included. Table 2 lists the best accuracy results for various SVM methods for different training-testing ratios, and the corresponding parameters that provide the best accuracy for those ratios.

Table 1 SNC Runs for the Feature Vectors {CS, SD, CSnSD, N} with Five Training-Testing Ratios

	CS (%)	SD (%)	CSnSD (%)	N (%)
		50%–50%		
Mean	99.62	99.61	99.52	99.17
Std. dev.	0.43	0.43	0.43	2.83
95% CI	0.08	0.08	0.08	0.55
		40%–60%		
Mean	99.25	92.02	99.62	95.73
Std. dev.	0.34	16.80	0.36	6.76
95% CI	0.07	3.29	0.07	1.33
		30%–70%		
Mean	99.62	46.48	99.56	92.38
Std. dev.	0.30	6.89	0.28	8.27
95% CI	0.06	0.08	0.08	0.55
		20%–80%		
Mean	99.59	42.02	98.98	80.08
Std. dev.	0.23	1.22	3.31	6.43
95% CI	0.05	0.24	0.65	1.26
		10%–90%		
Mean	98.61	41.14	85.91	50.37
Std. dev.	4.24	1.37	11.55	16.77
95% CI	0.83	0.27	2.26	3.29

Notes. Mean is the average accuracy of a prediction based on 100 runs; std. dev. and 95% CI are the standard deviation and the 95% confidence interval of the prediction. A higher average indicates a more accurate prediction, and a lower standard deviation and a lower confidence interval indicate higher consistency in the prediction.

Similar to the results of SNC, CS gives high accuracy in most cases. Unlike the results for SNC, for 50%–50% ratio, CSnSD gives better accuracy when run with SVM-RFE. Still, using CS feature vectors gives highly accurate results for both the SNC and the SVM methods. We conclude that CS is the best suitable of the feature vectors to use. Furthermore, for all methods involved that take the RBF kernel (SVM and SVM-1), RBF consistently presents better results than polynomial kernels.

Among SVM and specialized SVMs, SVM-1 appears to improve the results of SVM, whereas SVM-RFE improves the results only in some cases and SVM-NLP does not appear to improve SVM on this set of data.

Table 3 displays the detailed results for PCA-LDA. We observe that the results indicate that SD is a more appropriate feature construction for PCA-LDA. This

Table 2 SVM, SVM-1, SVM-RFE, and SVM-NLP Runs with Five Training-Testing Ratios

	SVM	SVM-1	SVM-RFE	SVM-NLP
50%–50%	99.59% (CS, C = 32, RBF, $\sigma = 128$)	99.54% (CS, C = 32, RBF, $\sigma = 128$)	99.38% (CSnSD, C = 1, Feats = 256)	98.27 (N, C = 100, $\nu = 256$)
40%–60%	99.60% (CS, C = 16, RBF, $\sigma = 128$)	99.68% (CS, C = 16, RBF, $\sigma = 128$)	99.10% (CS, C = 2, Feats = 128)	94.75% (CS, C = 100, $\nu = 2^{-11}$)
30%–70%	99.46% (CS, C = 128, RBF, $\sigma = 128$)	99.58% (CS, C = 64, RBF, $\sigma = 128$)	98.92% (CS, C = 32, Feats = 128)	94.56% (CSnSD, C = 100, $\nu = 1$)
20%–80%	99.43% (CS, C = 64, RBF, $\sigma = 128$)	99.56% (CS, C = 64, RBF, $\sigma = 128$)	98.88% (CS, C = 16, Feats = 512)	94.92% (CS, C = 100, $\nu = 0.031$)
10%–90%	98.13% (CS, C = 128, RBF, $\sigma = 128$)	98.80% (CS, C = 64, RBF, $\sigma = 128$)	96.74% (CS, C = 0.008, Feats = 1,024)	93.18% (CS, C = 100, $\nu = 0.004$)

Notes. The best accuracy result for each method and each ratio is listed along with the optimal parameters. Feats is the optimal number of features for SVM-RFE.

is in contrast to CS as the best and most accurate feature construction for both SNC and SVM. The only case for which SD is not best for PCA-LDA is when the training-testing percentage is 10%–90%. In this instance, N gives better accuracy, but SD still gives the most consistent results.

Figure 4 summarizes the results of Tables 1–3. In Figure 4, the highest accuracy is presented for the different training-testing ratios. Examining the graph in Figure 4 shows that SNC, in terms of accuracy, is on par or superior both in accuracy and robustness to the methods compared, except SVM-1. SVM-1 for 10%–90% ratio has (slightly) higher accuracy which comes at a price of a substantial increase in running time (see §4.5). In terms of robustness, which is measured by the decrease in accuracy as the sample size decreases, SNC presents better results than most methods. For example, in the case of 10%–90% training-testing ratio, SNC has a more than 1% accuracy lead over SVM, 2% lead over SVM-RFE, and 4% lead over PCA-LDA.

Table 3 PCA Plus Linear Discriminant Analysis Runs for the Feature-Vectors {CS, SD, CSnSD, N} with Five Training-Testing Ratios

	CS (%)	SD (%)	CSnSD (%)	N (%)
		50%–50%		
Mean	76.91	97.04	77.56	93.00
Std. dev.	3.39	1.26	4.46	3.72
95% CI	0.66	0.25	0.87	0.73
		40%–60%		
Mean	76.21	96.96	76.03	93.42
Std. dev.	4.02	1.46	4.10	3.86
95% CI	0.79	0.29	0.80	0.76
		30%–70%		
Mean	76.83	96.61	75.55	93.28
Std. dev.	4.33	1.77	5.10	4.25
95% CI	0.85	0.35	1.00	0.83
		20%–80%		
Mean	74.68	95.46	74.55	93.43
Std. dev.	5.45	2.25	5.15	5.66
95% CI	1.07	0.44	1.01	1.11
		10%–90%		
Mean	74.13	91.34	74.76	94.00
Std. dev.	5.90	4.86	5.39	5.05
95% CI	1.16	0.95	1.06	0.99

Notes. "Mean" is the average accuracy of a prediction based on 100 runs; "std. dev." and "95% CI" are the standard deviation and the 95% confidence interval of the prediction, respectively. A higher average indicates a more accurate prediction, and a lower standard deviation and a lower confidence interval indicate higher consistency in the prediction.

4.4. SNC Misclassifications Analysis

A *confusion matrix* is a presentation of the data that helps to identify the source of prediction errors. In a confusion matrix, the rows are the true labels of the data points and the columns are the predictions. For example, an entry at position (Pu, U) corresponds to the average number of data points, over 100 runs, that are incorrectly labeled as uranium when their true identity is plutonium. According to this definition, the

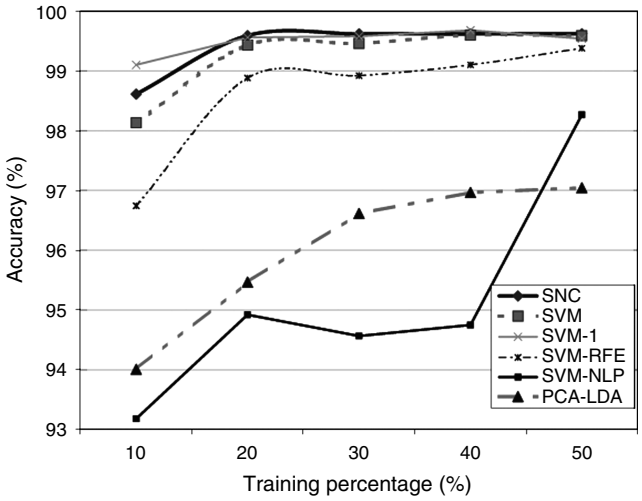

Figure 4 The Best Classification Accuracy for SNC, SVM, Specialized SVMs, and PCA-LDA with Different Training Sizes

Table 4 Confusion Matrices for Different Training Sizes

	50% training		40% training		30% training		20% training		10% training	
	Pu	U	Pu	U	Pu	U	Pu	U	Pu	U
Pu	69.56	0.44	83.88	0.67	97.38	0.62	111.24	0.76	123.1	2.9
U	0	46	0	55	0	65	0	74	0	83

sum of diagonal entries is the number of correctly predicted points. Table 4 is the confusion matrices of the results of SNC with CS feature vectors for the different training-testing ratios. Table 4 shows that all samples acquired in the presence of uranium were labeled with 100% accuracy. It is interesting to observe that the only source of error for all matrices is the misclassification of plutonium samples as uranium samples.

4.5. Run Times

We report on the run times of SVM and specialized SVMs, that *exclude* the time required to find the best tuned parameters, including only run times for training and classification. Figure 5 graphs the running times of SNC, SVM, and PCA-LDA. Because the complexity of SVMs depends on the number of training data points, the smaller the training data, the shorter SVMs' running times. SVM-RFE and SVM-NLP have the longest running times—about 80 times more than that of SNC. SVM-1, whose accuracy result is a bit better at 10%–90% ratio than SNC, is four times slower than SNC at that ratio. The running time of SVM-RFE is influenced not only by the ratio, but also by the number of features used. For SNC, the run time of the algorithm is dominated by the graph construction, and therefore appears constant regardless of the size of the training set. PCA-LDA has the running time of more than 40 times than that of SNC—this is primarily because of using PCA, which is the reduced version; the full version of PCA takes even longer. Figure 5 clearly shows that SNC is significantly more efficient than SVMs (factor of 2–80) and PCA-LDA (factor of 40) under the same hardware setup (1.3 GHz Intel SU7300 Core 2 Duo ULV Processor with 1 GB 1,066 MHz RAM).

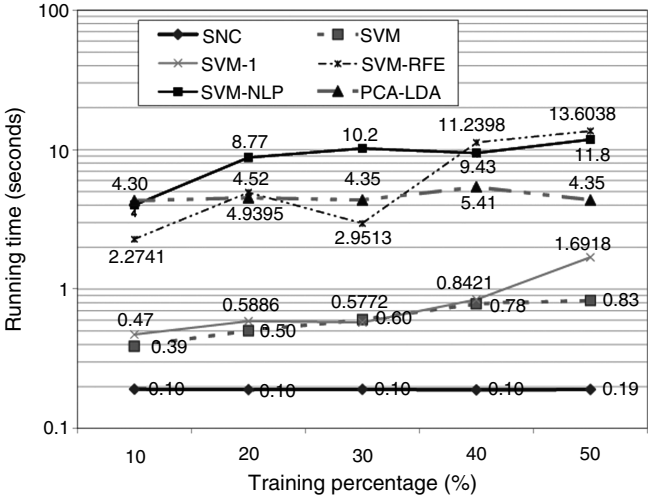

Figure 5 Computation Times of SNC, SVM, Specialized SVMs, and PCA-LDA for Different Training Sizes

Note. The graph is drawn in log scale and the running times are labeled next to the curves.

4.6. Multiclassification

In this section we evaluate the performance of SNC with respect to SVM and PCA-LDA for solving multiclassification problems. As both SNC and SVM use binary classification as a subroutine for solving the multiclassification problem, we apply the voting mechanism described in §2.2 for both methods. Specialized SVMs have less established extension from binary to multiclassifications and are left for future investigations. For the evaluation process we use three subsets of the Swanberg et al. (2009) data set: (1) blank, plutonium, and uranium; (2) latite, plutonium, and uranium; and (3) blank, latite, plutonium, and uranium. Note that the latter consists of the entire data set.

The results for SNC of 50%–50% training-testing case (Table 5) show that SNC-based multiclassification gives highly accurate and consistent predictions for several sets of data. When all four materials are used, the best prediction accuracy is 98.65% under CSnSD feature vectors. In fact across all permutations, CSnSD is the best feature vector in terms of both accuracy and consistency. We note that the presence of latite affects the quality of the classification more adversely than blank or background noise, as observed in the difference between the first row and the second row of Table 5. When all materials are present, the prediction improves from that with latite, plutonium, and uranium.

Figure 6 displays the comparison among SNC, SVM, and PCA-LDA, for three different classification problems. The results presented in Figure 6, for each training portion, are the best results achieved across all processing methods (CS, SD, CSnSD, and N). As can be seen in Figure 6 for all classification setups and at each training portion, SNC produces better accuracy than SVM and PCA-LDA. Furthermore, in terms of robustness, SNC is superior to SVM and PCA-LDA.

Table 5 Multiclassification for Different Permutations of the Data Set (B: Blank; L: Latite; Pu: Plutonium; and U: Uranium) and the Four Different Kinds of Feature Vectors {CS, SD, CSnSD, N}

	CS (%)		SD (%)		CSnSD (%)		N (%)	
	Mean	Std. dev.	Mean	Std. dev.	Mean	Std. dev.	Mean	Std. dev.
L, Pu, and U	94.85	3.40	98.34	0.98	98.91	0.78	84.61	0.27
B, Pu, and U	100.00	0.00	99.88	1.21	100.00	0.00	87.37	0.54
B, L, Pu, and U	98.65	1.25	98.32	1.75	98.65	1.12	86.22	0.62

Note. The subsampling ratio is 50%–50%.

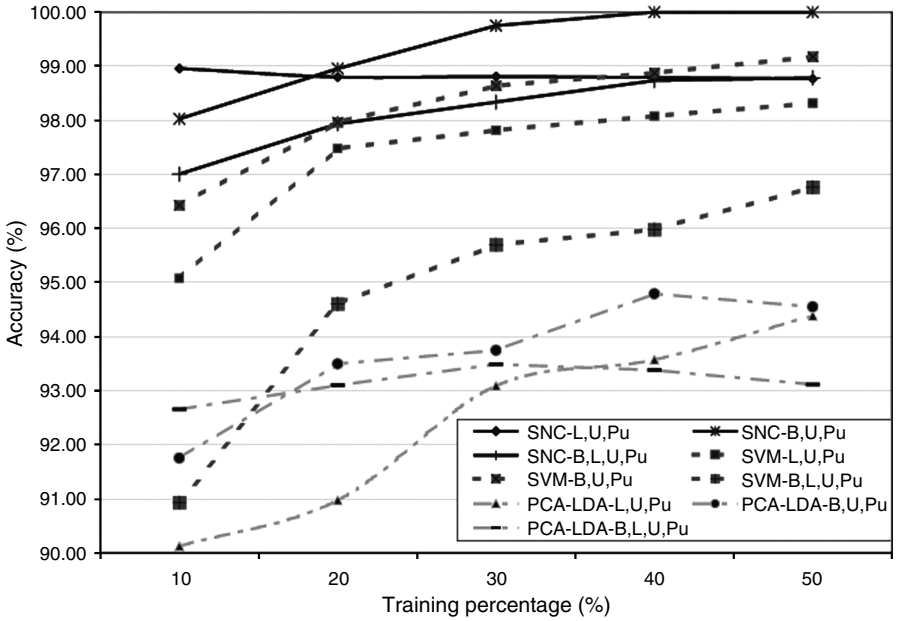

Figure 6 The Best Multiclassification Accuracy for SNC, SVM, and PCA-LDA with Different Training Sizes

These results are in agreement with the results of binary classification, which are reported in §4.2.1.

Figure 7 displays the run times of SNC, SVM, and PCA-LDA. It is noted that the SVM method requires extensive tuning of parameters for each data set and each feature-vector representation method (§3.2). The running times of SVM, reported in Figure 7, do not include the time it takes for tuning of the various parameters. Were these times to be included, the superiority of SNC's efficiency would have been even more pronounced.

4.7. Constructions of Feature Vectors and Running Time

We conclude this section with the presentation of running time results for the different feature vectors presented in §3.2. The major reason for choosing these methods is their simplicity: despite the fact that more complex derived features such as those from signal processing may produce better results, the four methods proposed here are simple and incorporate both spatial and temporal dimensions of our data set.

Figure 8 displays the running times of various feature-vector constructions with different algorithms—SVM, SNC, and PCA-LDA for the 50%–50% training-testing ratio of binary classification problems. Overall, Figure 8 shows, as concluded in the previous

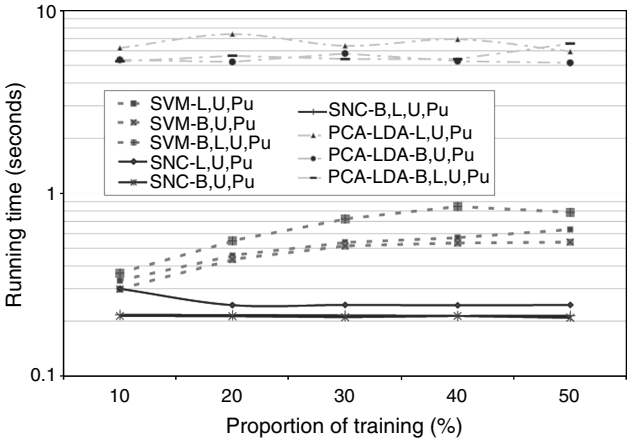

Figure 7 Computation Times of SNC, SVM, and PCA-LDA for Different Training Sizes

Note. The running times of B, U, Pu and L, U, Pu overlap for SNC.

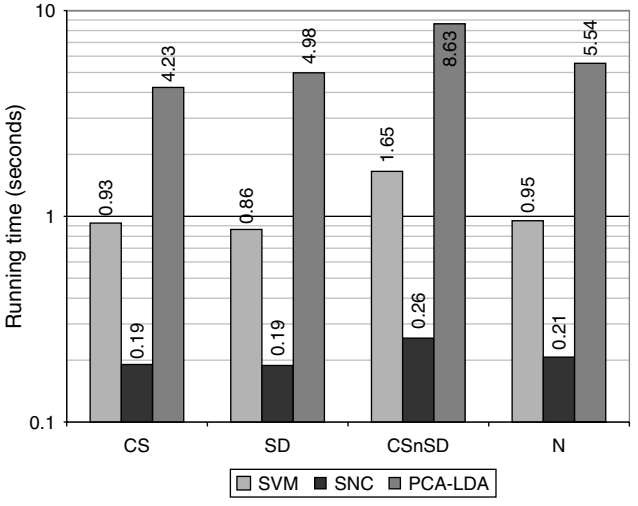

Figure 8 Computation Times of SNC, SVM, and PCA-LDA for Different Feature-Vector Constructions {CS, SD, CSnSD, N} for the 50%–50% Training-Testing Ratio of Binary Classification Problems

Note. The plot is drawn in log scale and individual running times are marked above their respective bars.

sections, that SNC is the fastest algorithm involved, SVM is the next fastest, and PCA-LDA is the slowest. We can also observe how sensitive the running times depend on different feature-vector constructions: CSnSD appears to require the most computation than other methods—this is confirmed by the fact that CSnSD gives the longest feature vectors among all methods. The next method that constructs feature vectors and requires long running times for all methods is N, followed by CS and SD, which are the fastest for PCA-LDA and SVM, respectively, and tied for SNC. One should note that the SNC algorithm takes the edge weights as input and thus is not affected by the length of the feature vectors. The computation times of the weights, however, are proportional to the feature-vector lengths. SVM and PCA-LDA have different running times when applied on the different feature vectors. This observation strongly suggests that if one considers to replace the method for generating the feature vectors, he should consider not only the method's running times for creating the vectors, but also the total run times with the new vectors.

These observations, combined with the accuracy results from §4.2.1, conclude that CS is the most appropriate feature-vector construction method—using feature vectors constructed by CS allows algorithms to have fast and accurate classification results.

5. Conclusions

We present here a new technique for classification and clustering devised for the purpose of enhancing the identification of nuclear threats with low-resolution detectors. The technique builds on a bipartitioning procedure called normalized cut prime (NC′). The solution method proposed here incorporates training data and as such it is a supervised classification method, *supervised normalized cut* (SNC). We test SNC and compare it with SVM, specialized SVMs, and LDA on data of low-resolution nuclear spectra. The results demonstrate that SNC is comparable or superior to SVM methods in terms of accuracy and much more superior in terms of efficiency and robustness for either the binary or the multiclassification problem of the nuclear data set. The analysis presented in the paper provides a proof of concept that the SNC approach is worth investigating further in this context and with more detailed and advanced data sets. It should also be pursued as a complementary approach for identifying isotopes in spectra obtained even with higher resolution detectors, complementing the analysis of each energy line and their combinations. Future research will test supervised normalized cut on additional nuclear data sets on a vaster variety of SNMs as they become available. It is planned also to extend the application of SNC beyond the nuclear detection problems to more general classification and data mining problems.

Acknowledgments

This research has been supported in part by CBET-0736232 and by the U.S. Department of Homeland Security. The research of the third author was also supported in part by the National Science Foundation [Award DMI-0620677].

References

Bertozzi W, Ledoux RJ (2005) Nuclear resonance fluorescence imaging in non-intrusive cargo inspection. *Nuclear Instruments and Methods Phys. Res. Sect. B: Beam Interactions with Materials and Atoms* 241(1–4):820–825.

Burges CJC (1998) A tutorial on support vector machines for pattern recognition. *Data Mining Knowledge Discovery* 2(2):121–167.

Carneiro G, Chan AB, Moreno PJ, Vasconcelos N (2007) Supervised learning of semantic classes for image annotation and retrieval. *IEEE Trans. Pattern Anal. Machine Intelligence* 29:394–410.

Carpenter T, Cheng J, Roberts F, Xie M (2010) Sensor management problems of nuclear detection. Pham H, ed. *Safety and Risk Modeling and Its Applications* (Springer, London), 299–323.

Caruana R, Niculescu-Mizil A (2006) An empirical comparison of supervised learning algorithms. Cohen W, Moore A, eds. *Proc. 23rd Internat. Conf. Machine Learn., ICML '06* (ACM, New York), 161–168.

Chang C-C, Lin C-J (2001) LIBSVM: A Library For Support Vector Machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Cox IJ, Rao SB, Zhong Y (1996) Ratio regions: A technique for image segmentation. Kropatsch WG, ed. *Proc. Int. Conf. Pattern Recognition*, Vol. B (IEEE Computer Society Press, Los Alamitos, CA), 557–564.

Crammer K, Singer Y (2002) On the learnability and design of output codes for multiclass problems. *Machine Learn.* 47(2–3): 201–233.

Cristianni N, Shawe-Taylor J (2000) *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods* (Cambridge University Press, Cambridge, UK).

Ding YS, Zhang TL, Chou KC (2007) Prediction of protein structure classes with pseudo amino acid composition and fuzzy support vector machine network. *Protein Peptide Lett.* 14(8):811–815.

Dowd PA, Pardo-Iguzquiza E (2005) Estimating the boundary surface between geologic formations from 3D seismic data using neural networks and geostatistics. *Geophysics* 70(1):1–11.

Duan K-B, Keerthi SS (2005) Which is the best multiclass SVM method? An empirical study. Oza NC, Robi P, Josef K, Fabio R, eds. *Multiple Classifier Systems*, Lecture Notes in Computer Science, Vol. 3541 (Springer, Berlin), 732–760.

Duda RO, Hart PE, Stork DG (2001) Unsupervised learning and clustering. *Pattern Classification*, Chap. 10, 2nd ed. (Wiley, New York).

Ford LR, Fulkerson DR (1956) Maximal flow through a network. *Canadian J. Math.* 8(3):399–404.

Fung G, Mangasarian OL (2002) A feature selection Newton method for support vector machine classification. Technical report, University of Wisconsin, Madison.

Fung G, Mangasarian OL (2004) A feature selection Newton method for support vector machine classification. *Comput. Optim. Appl.* 28(2):185–202.

Fung GM, Mangasarian OL (2005) Multicategory proximal support vector machine classifiers. *Machine Learn.* 59(1–2):77–97.

Geisinger NP (2010) Classification of digital modulation schemes using linear and nonlinear classifiers. Ph.D. thesis, Naval Postgraduate School, Monterey, CA.

Gentile CA (2010) US Patent 7,711,661 b2, filed May 2, 2007, issued May 4, 2010.

Goldschmidt O, Hochbaum DS (1994) A polynomial algorithm for the k-cut problem for fixed k. *Math. Oper. Res.* 19(1):24–37.

Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. *Machine Learn.* 46(1–3):389–422.

Hastie T, Rosset S, Tibshirani R, Zhu J (2004) The entire regularization path for the support vector machine. *J. Machine Learn. Res.* 5(1):1391–1415.

Hochbaum DS (2010a) HPF: Hochbaum's Pseudo-Flow Algorithm Implementation. Software available at http://riot.ieor.berkeley.edu/riot/Applications/Pseudoflow/maxflow.html.

Hochbaum DS (2010b) Polynomial time algorithms for ratio regions and a variant of normalized cut. *IEEE Trans. Pattern Anal. Machine Intelligence* 32(5):889–898.

Hua Z, Wang Y, Xu X, Zhang B, Liang L (2007) Predicting corporate financial distress based on integration of support vector machine and logistic regression. *Expert Systems Appl.* 33(2): 434–440.

International Atomic Energy Agency (2007) Combating illicit trafficking in nuclear and other radioactive material. Technical report/reference manual, IAEA Nuclear Security Series 6, International Atomic Energy Agency, Vienna, Austria.

Kangas LJ, Keller PE, Siciliano ER, Kouzes RT, Ely JH (2008) The use of artificial neural networks in PVT-based radiation portal monitors. *Nuclear Instruments Methods Phys. Res. Sect. A, Accelerators, Spectrometers, Detectors Associated Equipment* 587(2–3): 398–412.

Kelly EL (1962) General description and operating characteristics of the Berkeley 88-inch cyclotron. *Nuclear Instruments Methods* 18–19(1):33–40.

Kotsiantis S (2007) Supervised learning: A review of classification techniques. *Informatica* 31(1):249–268.

Luo L (2006) Chemometrics and its applications to X-ray spectrometry. *X-ray Spectrometry* 35(4):215–225.

Mangasarian OL, Wild EW (2007) Nonlinear knowledge in kernel approximation. *IEEE Trans. Neural Networks* 18(1):300–306.

Mangasarian OL, Wild EW (2008) Nonlinear knowledge-based classification. *IEEE Trans. Neural Networks* 19(10):1826–1832.

Marrs RE, Norman EB, Burke JT, Macri RA, Shugart HA, Browne E, Smith AR (2008) Fission-product gamma-ray line pairs sensitive to fissile material and neutron energy. *Nuclear Instruments Methods Phys. Res. Sect. A: Accelerators, Spectrometers, Detectors Associated Equipment* 592(3):463–471.

Martinez AM, Kak AC (2001) PCA versus LDA. *IEEE Trans. Pattern Anal. Machine Intelligence* 23(2):228–233.

McLachlan G (2004) *Discriminant Analysis and Statistical Pattern Recognition* (Wiley Interscience, New York).

Mihailescu L, Fishbain B, Hochbaum DS, Maltz J, Moore CJ, Rohel J, Supic L, Vetter K (2010) Dynamic stand-off 3D gamma-ray imaging. Wehe DK, ed. *12th Sympos. Radiation Measurements Appl. (SORMA)* (National Nuclear Security Administration, Washington, DC), 106.

Norman EB, Prussin SG, Larimer R-M, Shugart H, Browne E, Smith AR, McDonald RJ, et al. (2004) Signatures of fissile materials: High-energy [gamma] rays following fission. *Nuclear Instruments Methods Phys. Res. Sect. A: Accelerators, Spectrometers, Detectors Associated Equipment* 521(2–3):608–610.

Resson HW, Varghese RS, Goldman R (2009) Computational methods for analysis of MALDI-TOF spectra to discover peptide serum biomarkers. *The Protein Protocols Handbook* IV: 1175–1183.

Rifkin R, Klautau A (2004) In defense of one-vs-all classication. *J. Machine Learn. Res.* 5(1):101–141.

Sharon E, Galun M, Sharon D, Basri R, Brandt A (2006) Hierarchy and adaptivity in segmenting visual scenes. *Nature* 442(7104): 810–813.

Shi J, Malik J (2000) Normalized cut and image segmentation. *IEEE Trans. Pattern Anal. Machine Intelligence* 22(8):888–905.

Swanberg E, Norman E, Shugart H, Prussin S, Browne E (2009) Using low resolution gamma detectors to detect and differentiate ^{239}Pu and ^{235}U fissions. *J. Radioanalytical Nuclear Chemistry* 282(3):901–904.

Wang F, Qian Y, Dai Y, Wang Z (2010) A model based on hybrid support vector machine and self-organizing map for anomaly detection. Wang C-X, Fan P, Shen X, He Y, eds. *Comm. Mobile Comput., Internat. Conf.* Vol. 1 (IEEE Computer Society, Los Alamitos, CA), 97–101.

West D (2000) Neural network credit scoring models. *Comput. Oper. Res.* 27(11–12):1131–1152.

Zhang J, Zhang XD, Ha S-W (2008) A novel approach using PCA and SVM for face detection. Guo M, Zhao L, Wang L, eds. *Fourth Internat. Conf. Nat. Comput.*, (*ICNC '08*) Vol. 3 (IEEE Computer Society, Los Alamitos, CA), 29–33.

Zhu J, Rosset S, Hastie T, Tibshirani R (2003) 1-norm support vector machines. Thrun S, Saul L, Schölkopf B, eds. *Neural Information Processing Systems* (MIT Press, Cambridge, MA), 16–23.