Computer Aided Detection via Asymmetric Cascade of Sparse Hyperplane Classifiers

[Extended Abstract]

Jinbo Bi, Senthil Periaswamy, Kazunori Okada, Toshiro Kubota,
Glenn Fung, Marcos Salganicoff, Bharat Rao
Computer Aided Diagnosis and Therapy Group
Siemens Medical Solutions, Inc.
51 Valley Stream Parkway, Malvern, PA 19355
jinbo.bi, senthil.periaswamy, kazunori.okada, toshiro.kubota,
glenn.fung, marcos.salganicoff, bharat.rao@siemens.com

ABSTRACT

This paper describes a novel classification method for computer aided detection (CAD) that identifies structures of interest from medical images. CAD problems are challenging largely due to the following three characteristics. A CAD system has to satisfy a real-time requirement. Typical CAD training data sets are large and extremely unbalanced between positive and negative classes. When searching for descriptive features, researchers often deploy a large set of experimental features, which consequently introduces irrelevant and redundant features.

This work is distinguished by three key contributions. The first is a cascade classification approach which is able to tackle all the above difficulties in a unified framework by employing an asymmetric cascade of sparse classifiers each trained to achieve high detection sensitivity and satisfactory false positive rates. The second is the incorporation of feature computational costs in a linear program formulation that allows the feature selection process to take into account different evaluation costs of various features. The third is a boosting algorithm derived from column generation optimization to effectively solve the proposed cascade linear programs.

We apply the proposed approach to the problem of detecting lung nodules from helical multi-slice CT images. Our approach demonstrates superior performance in comparison against support vector machines, linear discriminant analysis and cascade AdaBoost. Especially, the resulting detection system is significantly sped up with our approach.

Categories and Subject Descriptors

I.5.m [**Pattern Recognition**]: Miscellaneous

General Terms

Algorithms

Keywords

Computer aided detection, Cascading classification, Mathematical programming, Support vector machines, Sparse solutions

1. PROBLEM SPECIFICATION

Over the last decade, Computer-Aided Detection (CAD) systems have moved from the sole realm of academic publications, to robust commercial systems that are used by physicians in their clinical practice to help detect early cancer from medical images. The growth has been fueled by the Food and Drug Administrations (FDA) decision to grant approval in 1998 for a CAD system that detected breast cancer lesions from mammograms (scanned x-ray images) [19]. Since then a number of CAD systems have received FDA approval. Virtually all these commercial CAD systems focus on detection (or more recently diagnosis [7]) of breast cancer lesions for mammography. The CAD concept can be generalized to many other detection tasks in medical image analysis, such as lung nodule detection and colon polyp detection.

The typical workflow for a CAD system when used to identify structures in a new patient image is:

1. *Identify candidate structures in the image*: Most medical images, particularly image volumes generated by high-resolution computed tomography (CT), are very large. Typically, an efficient image processing algorithm considers each pixel (or voxel) in the image as a potential candidate "seed", and selects a fraction of the seeds as candidates.

2. *Extract features for each candidate*: A large number of image features are usually calculated to describe the target structure. Some of the features can be irrelevant, or redundant, or computationally expensive. Thus, sparse feature selection is necessary in order to ensure a relatively small

number of relevant features in the deployed CAD system.

3. *Classify candidates as positive or negative*: A previously-trained classifier is used to label each candidate.

4. *Display positive candidates*: Commonly, the digitized image is displayed with marks for inspection by physicians.

In the candidate identification stage, even a small fraction of the seeds is necessarily very large in order to maintain high sensitivity. High sensitivity (ideally close to 100%) is essential, because any cancers missed at this stage can never be found by the CAD system, which otherwise may be detected later in the classification stage by exploring effective features. Hence, a lot false positives are generated in this stage (less than 1% of the candidates are positive), which makes the classification problem highly unbalanced. Moreover, CAD systems are required to be fast enough for physicians to use in the middle of their diagnostic analysis.

2. OVERVIEW OF OUR APPROACH

Our major contribution in this paper lies in a new cascaded classification approach that solves a sequence of linear programs, each constructing a hyperplane classifier of the form $\text{sign}(\mathbf{w}'\mathbf{x} + b)$ where $\mathbf{x}$ is the feature vector and $(\mathbf{w}, b)$ are model parameters to be determined. The linear programs are derived through piece-wise linear cost functions and the ℓ_1-norm regularization condition. The resulting linear program works in the same principle as for the 1-norm SVM. The ℓ_1-norm regularization inherently performs feature selection since penalizing on the 1-norm regularization of $\mathbf{w}$ drives the resulting optimal $\mathbf{w}$ to be sparse, meaning only a few features receive a non-zero weight w. To incorporate the feature computational complexity into the selection of features, a weighted ℓ_1-norm is employed where weights are determined by the computational cost of each feature. Each linear program employs an asymmetric error measure that penalizes with different weights on false negatives and false positives. An extreme case is that the penalty for a false negative is infinity, which is used in the early stage of the cascade design to alleviate the skewed class distribution and preserve high detection rates.

Previous cascade classification approaches are mostly based on AdaBoost [10, 20, 11]. Cascade AdaBoost serves as a great tool for building real-time robust applications [22, 24], especially for object detection systems. However, cascade AdaBoost works with two implicit assumptions: 1. a significant amount of representative data is available for training the cascade classifier; 2. all features can be equally evaluated with a relatively low computational cost. These assumptions, unfortunately, often do not hold in CAD systems. Collecting patient data is very expensive and time-consuming. Available data can be noisy and hardly represent all aspects of the target characteristics. One of the major concern about cascade classification approaches is if a classifier within the cascade does not generalize well and hence screens out more true positives than necessary, then these true positives will never be recovered at later stages. The more stages in the cascade, the riskier the system becomes unstable. This observation motivates us to design a cascade that consists of significantly few stages. Furthermore, simple and low-cost image features are often not sufficient for detecting target structures in a CAD system. Advanced features are indispensable for performance enhancement, but they require great computation time. If these features need to be calculated for a large portion of the candidates at the early stage of the cascade, the system may become prohibitively slow. Cascade AdaBoost treats all features equally when selecting features for each individual stage classifier, which leads to a computation inefficiency.

Unlike cascade AdaBoost, the proposed approach incorporates the computational complexity of features into the cascade design. Our cascading strategy brings advantages of multiple folds: 1. Easy classification: the detection problem becomes much more balanced at later stages, facilitating advanced classification algorithms to be applied and perform well at these stages when overall accuracy becomes more demanding at later stages. 2. High computational efficiency: early stages weed out many non-target patterns, so most stages are not evaluated for a typical negative candidate. Computationally expensive features are only calculated for a small portion of the candidates at later stages. 3. Robust system: the linear program with a ℓ_1-norm regularization at each stage is a robust system. Although no theoretical justification is derived, a cascade of very few stages unlikely harms the robustness of linear classifiers, opposed to a cascade of over 20 stages as AdaBoost cascade often obtains.

3. AN EXAMPLE OF CAD: AUTOMATIC NODULE DETECTION

In this article, we particularly discuss an automatic lung nodule detection system. Lung cancer is the leading cause of cancer-related death in western countries with a mean 5 year survival rate for all stages of only 14%. The prognosis of stage I cancer is more optimistic with a mean 5 year survival rate of about 49%. Although multi-slice CT scanners allow acquisition of the entire chest, only 15% of lung cancers are diagnosed at the early stage. The problem is that a single CT examination may acquire up to 700 axial images whose interpretation is tedious and perceptually demanding. CAD is considered to be a helpful diagnostic tool to handle this increasing amount of radiological data. It is well recognized that the use of CAD not only offers the potential to decrease detection and recognition errors as a second reader, but also to reduce mistakes related to misinterpretation [16][1]. Recently a variety of research has been dedicated to improvement of automatic nodule detection performance using state-of-the-art machine learning techniques, such as convolution neural networks [13], support vector machines [15] or a combined system which uses simple rules to reduce the number of nodule candidates followed by linear discriminant analysis [2].

Our data was collected from multiple sites. The CT volumes are typically of size $512 \times 512 \times 350$ (approximately), representing a slice thickness of about 1mm. We conduct a pre-processing step. The region of interest, which is the lung in our problem, is first extracted using segmentation techniques, so that all candidates generated will be guaranteed inside the lung. This also facilitates the detection of wall-attached nodules (an example is shown in Figure 1, top-left). The candidate generation (CG) algorithm employs a robust blob detection strategy that identifies all the blob-like structures. The size of the blob-like structures vary in

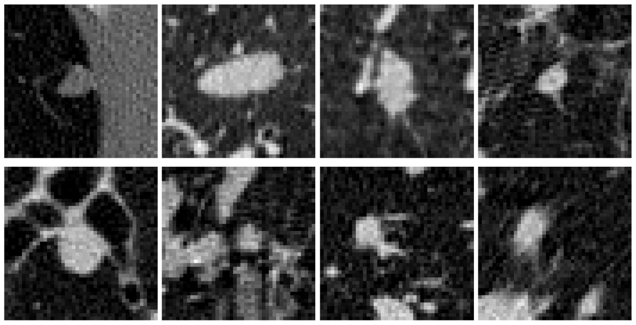
Figure 1: Shown in this figure are four sample nodules (top row) and four sample false positives (bottom row) from the CG step.

diameter starting from 3mm, which is the minimum size of interest for radiologists. The output of the CG step is the locations of candidates, along with some internal features that are generated as intermediate results of the candidate generator. These features include simple gray scale statistics and shape based information of the blob. There are a total of 10 such features output by the candidate generator. The CG algorithm identifies most of the nodules from the CT scans, and some non-obvious nodule examples, that are successfully detected by our CG, are illustrated in Figure 1 (first row, from left to right, wall-attached, elliptical, vessel-attached, very small nodules). However, it also generates a lot false positive candidates as shown in Figure 1 (second row, these false positives, from left to right, are due to lymph tissue, tissue scarring, unknown structure and motion).

After the CG step, a large number of image features are computed for each of the candidates to describe the shape (both 2D and 3D), size, intensity statistics (that describe texture), and template matching. The complexity of these features is around linear with respect to the size of the sub-volume where the feature computation takes place. This can be as large as $21 \times 21 \times 21$ voxels, and the amount of computation may become prohibitively large if we need to compute them on every candidate.

Another set of more advanced and computationally demanding features are calculated as follows. For each detected candidate region, the target is segmented by using a nodule segmentation technique developed in [18]. Twenty seven statistical features, computed in a remapped coordinate system, are derived from the segmented shape of the candidate. The feature set includes local intensity statistics and geometrical features, such as size and anisotropy of the gray values at multiple scales. Furthermore, the intensity homogeneity is represented in a set of entropic measures by using the generalized Jensen-Shannon divergence [14]. Jensen-Shannon divergence extends the well-known Kullback-Liebler divergence between a pair of probability distributions in order to describe overall similarity of a *set* of distributions. Intensity homogeneity can be represented with the Jensen-Shannon divergence by computing overall similarity of a set of intensity-based histograms derived from different sub-partitions. This approach allows a flexible extension of the entropy-based intensity homogeneity index as a function of arbitrary data partition and/or sampling. The computation time of different sets of features are summarized in Table 1.

Feature Sets	Features	Mean	STD
Feature set 1 (Simple)	CG features	0	0
Feature set 2 (Complex)	Shape & size	17.3	1.8
	Intensity statistics	26.4	7.2
	Template	1.5	0.26
Feature set 3 (Most Complex)	Multi-scale statistics	2010	351

Table 1: Statistics of the computation time of various features in milliseconds per candidate.

4. CASCADE OF HYPERPLANE CLASSIFIERS

Cascade classifiers were previously investigated in such works as [27, 12, 9]. Especially, in the face detection field, many classification cascading strategies [25, 21, 23] were discussed and shown good performance. Among those methods, the AdaBoost boosting algorithm provides a simple yet effective stagewise learning approach for the cascade design. However, as discussed in Section 2, it has some disadvantages.

In this section, we investigate a linear programming framework for constructing a cascade of sparse linear classifiers $\mathbf{w}'\mathbf{x} + b$. Each stage of the cascade solves a linear program which is formulated through the hinge loss

$$\xi = \max\{0, 1 - y(\mathbf{w}'\mathbf{x} + b)\}$$

and the ℓ_1-norm penalty or weighted ℓ_1-norm penalty

$$\|\mathbf{w}\|_1^\gamma = \sum \gamma_i |w_i|$$

where γ_i is a weighting factor related to the computational cost of the i-th feature assuming the cost information is available or otherwise it becomes the regular ℓ_1-norm by setting all $\gamma = 1$. Although the linear programs at each stage can be solved using any general-purpose linear program solver, we show in the next section that the column generation technique for linear programs, is equally suitable for optimizing each linear program in an incremental fashion as AdaBoost does. Moreover, the column generation boosting derivation can be applied to any linear program regardless of the choice of the trade-off factor between the detection rate and the false positive rate whereas AdaBoost needs to be revised for an asymmetric re-weighting scheme [25, 23].

The proposed cascade classification strategy provides a general framework for building a cascade, but a concrete cascade design is problem-specific. Prior knowledge or domain insights may help identify good features to be included in various stages for a specific problem. Users can give high priority to more meaningful and interpretable features to use in early stages. If no such priori information exists, one philosophy is to have a preference for less complex and computationally cheaper features.

The cascade hierarchy for our nodule detection system has

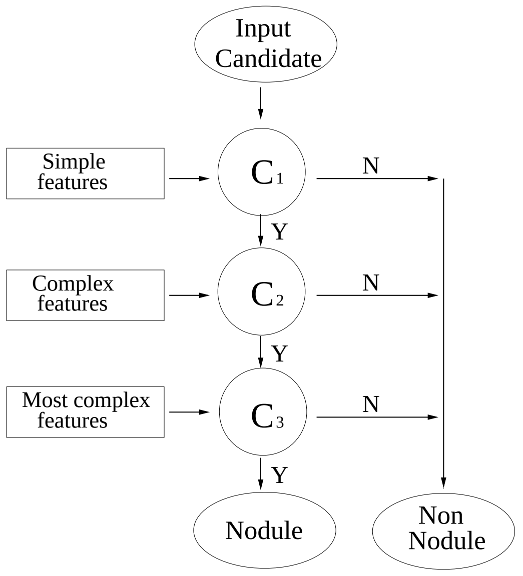

Figure 2: Cascade of classifiers for nodule detection

3 stages as depicted in Figure 2. The first stage (C_1) considers the internal features of the CG algorithm as discussed in Section 3 because these features come together with candidates and do not require any extra computational cost. As observed in our experiments, a significant amount of non-nodule candidates are eliminated at this stage. This stage in the classification cascade can also be viewed as an effort to optimize the CG algorithm itself. After the CG step, more advanced features are calculated to improve the overall accuracy for both false negative and false positive rates. With these features, the second classifier (C_2) in the cascade often achieves an acceptable performance. However, to further improve from the acceptable performance to an excellent performance, it requires the third set of features which are computationally demanding. Our cascade evaluate these features in the last stage (C_3) for the candidates that have survived from the second stage. Bear in mind that each stage can take a single set of features or treat computationally expensive features as an addition to the feature set which is already being used. Our system uses the accumulative set of features.

4.1 Utilizing asymmetric loss function at early stages

In a cascade, computation time and detection rate of the early stages are critically important to overall performance of the final system. As emphasized already in previous sections, any nodules missed at the first stage can not be recovered later by the system. Detection sensitivity needs to be extremely high, and often requires to be 100%. In most cascade classification methods, a reasonable classifier is trained at this stage and then the decision threshold is adjusted (manually or in a greedy fashion) to minimize the false negatives. We propose a more principled formulation that guarantees a 100% detection rate as well as optimizes the best possible false positive rate at 100% detection rate.

Denote $\{\mathbf{x}_i, y_i\}, i = 1, \cdots, \ell$ as our candidate set generated by the CG algorithm. We use $\mathbf{X}$ to denote the feature matrix where each row represents a candidate feature vector $\mathbf{x}$ and each column specifies a feature, and use i to index the rows (or candidates) and j to index the various features. Notice that the feature vector $\mathbf{x}$ realizes different image features at different stages. Without loss of generality, we assume that the classification stage receives ℓ^+ positive candidates and ℓ^- non-nodule candidates, $\mathbf{X}$ contains d features, and C^+ and C^- contain, respectively, the sets of indices of positive sample and negative sample.

Our linear program formulation at each stage seeks an optimal hyperplane classifier by minimizing a weighted sum of the empirical error measure and the regularization factor. The classification error measure approximated via the hinge loss is generally defined by $\frac{\sum \xi_i}{\ell}$. To deal with the unbalanced problem, we define the error measure as a convex combination of the false negative rate and false positive rate, i.e.,

$$\frac{\mu}{\ell^+} \sum_{i \in C^+} \xi_i + \frac{1-\mu}{\ell^-} \sum_{i \in C^-} \xi_i \tag{1}$$

where $0 \leq \mu \leq 1$ is a tuning parameter. The asymmetric cascade classifier can be achieved by choosing an appropriate value of μ close enough to 1. However, this does not guarantee a 100% detection rate at the first stage which is desired for our design. An extreme case of the asymmetric error measure is to give a penalty of infinity to a false negative so that the resulting classifier preserves all nodule candidates detected. This asymmetric error cannot be formulated as a convex combination of false positive and false negative rates, and it corresponds to imposing the constraint $\sum_j \mathbf{X}_{ij} w_j + b \geq 0, \ \forall i \in C^+$.

To form a linear program, we rewrite $w_j = u_j - v_j$ and require $u_j, \ v_j \geq 0$. The linear program is written as the following optimization problem with a regular ℓ_1-norm regularization:

$$\begin{array}{ll} \min_{\mathbf{u},\mathbf{v},\boldsymbol{\xi}} & \lambda \sum_{j=1}^{d} (u_j + v_j) + \frac{1}{\ell^-} \sum_{i \in C^-} \xi_i \\ \text{s.t.} & \sum_j \mathbf{X}_{ij}(u_j - v_j) + b \geq 0, \quad i \in C^+ \\ & -\sum_j \mathbf{X}_{ij}(u_j - v_j) - b + \xi_i \geq 1, \quad i \in C^-, \\ & \xi_i \geq 0, \quad i \in C^-, \\ & u_j, \ v_j \geq 0, \quad j = 1, \cdots, d. \end{array} \tag{2}$$

where $\lambda > 0$ is the regularization parameter. Note that $|w_j| = u_j + v_j$ if either u_j or v_j has to be 0 so $\sum_j |w_j|$ is replaced by $\sum_j (u_j + v_j)$. Solving the above linear program yields optimal solutions to the formulation directly with $\sum_j |w_j|$ since at optimality, at least one of the two variables u_j and v_j will be zero for all $j = 1, \cdots, d$.

We now derive the dual problem for the above linear program since the dual will play a key role in the column generation derivation for the boosting algorithm which we shall discuss in the next section. There are two variables u_j, v_j corresponding to a feature $\mathbf{X}_{\cdot j}$ in problem (2). Correspondingly the Lagrangian dual problem has two constraints for the feature $\mathbf{X}_{\cdot j}$, i.e., $\sum_{i=1}^{\ell} \beta_i y_i \mathbf{X}_{ij} \leq \lambda$ and $-\sum_{i=1}^{\ell} \beta_i y_i \mathbf{X}_{ij} \leq \lambda$. Combining both constraints, we have

$-\lambda \le \sum_{i=1}^{\ell} \beta_i y_i \mathbf{X}_{ij} \le \lambda$. Hence the dual problem is written as:

$$\begin{aligned} \max_{\boldsymbol{\beta}} \quad & \sum_{i \in C^-} \beta_i \\ \text{s.t.} \quad & -\lambda \le \sum_{i=1}^{\ell} \beta_i y_i \mathbf{X}_{ij} \le \lambda, \quad j = 1, \cdots, d, \\ & \sum_{i=1}^{\ell} \beta_i y_i = 0, \\ & 0 \le \beta_i \le \tfrac{1}{\ell^-}, \ i \in C^-. \end{aligned} \tag{3}$$

The two linear programs (2) and (3) guarantee that all true positives remain to the next cascade stage. With all nodule candidates preserved, they reduce the most possible amount of false positives by minimizing the error $\frac{1}{\ell^-} \sum_{i \in C^-} \xi_i$.

4.2 Incorporating computational complexity of features

In later stages of a cascade, 100% detection rate may not be realistic to maintain in order to attain a reasonably good false positive rate. The convex combination error measure (1) is thus used in the linear programs to allow the presence of false negatives. In later stages, more and more computationally demanding features are included in the training of classifiers. One philosophy we hold is to allow cheaper features to do their best before resorting to expensive features, thus leading to a computational efficiency. The weighted ℓ_1-norm regularization is employed to form linear programs where weighting factors γ are each determined by the computational cost of the corresponding feature. Consequently, expensive features will be selected with greater penalty in the objective function of linear programs. The optimization problem can be formulated as the following linear program similarly by rewriting each $w_j = u_j - v_j$:

$$\begin{aligned} \min_{\mathbf{u},\mathbf{v},\boldsymbol{\xi}} \quad & \lambda \sum_{j=1}^{d} \gamma_j (u_j + v_j) + \tfrac{\mu}{\ell^+} \sum_{i \in C^+} \xi_i + \tfrac{1-\mu}{\ell^-} \sum_{i \in C^-} \xi_i \\ \text{s.t.} \quad & y_i \left(\sum_j \mathbf{X}_{ij} (u_j - v_j) + b \right) + \xi_i \ge 1, \\ & \xi_i \ge 0, \quad i = 1, \cdots, \ell, \\ & u_j, \ v_j \ge 0, \quad j = 1, \cdots, d. \end{aligned} \tag{4}$$

Analogous to the duality analysis for problem (2), the dual to problem (4) can be derived and written as follows:

$$\begin{aligned} \max_{\boldsymbol{\beta}} \quad & \sum_{i=1}^{\ell} \beta_i \\ \text{s.t.} \quad & -\lambda\gamma_j \le \sum_{i=1}^{\ell} \beta_i y_i \mathbf{X}_{ij} \le \lambda\gamma_j, \quad j = 1, \cdots, d, \\ & \sum_{i=1}^{\ell} \beta_i y_i = 0, \\ & 0 \le \beta_i \le \tfrac{\mu}{\ell^+}, \ i \in C^+, \\ & 0 \le \beta_i \le \tfrac{1-\mu}{\ell^-}, \ i \in C^-. \end{aligned} \tag{5}$$

Determining an appropriate weight vector γ based on the feature computational complexity can be problem specific, and can be an interesting topic for further research. In our implementation, we simply normalized the computation time in milliseconds by the Sigmoid function, so γ ranges from 0.5 to 1.

5. FEATURE SELECTION VIA COLUMN GENERATION BOOSTING

We describe a column generation technique in this section. The column generation techniques have been widely used for solving large-scale LPs or difficult integer programs since 1950s [17], and have been introduced to the machine learning community, i.e., the so-called LPBoost [5, 6]. But the LPBoost procedure derived in [5, 6] does not directly solve our formulations. Although our formulations (2) and (4) can be optimized by any linear program solvers. The to-be-derived boosting scheme offers an on-line and incremental fashion solution, and provides as well insights into which features play roles during the training phase, facilitating feature selection.

In the context of column generation, a feature $\mathbf{X}_{\cdot j}$ is viewed as a column. During the process, features are continuously selected and classifiers are optimized based on the selected features corresponding to the columns generated. In the primal space, the column generation method solves linear programs on a subset of variables $\mathbf{w}$, which means not all columns of the matrix $\mathbf{X}$ are generated at once and used to construct the classifier. Columns are generated iteratively and added to the problem to achieve optimality. In the dual space, a column in the primal problem corresponds to a constraint in the dual problem. When a column is not included in the primal, the corresponding constraint does not appear in the dual. If a constraint absent from the dual problem is violated by the solution to the restricted problem, this constraint (a cutting plane) needs to be included in the dual problem to further restrict the dual feasible region. Thus these techniques are also referred to as cutting plane methods [3].

The variables w_j are then partitioned into two sets, the working set W used to build the model and the remaining set denoted as N that is eliminated from the model as the corresponding columns are not generated. Each generation step optimizes a subproblem over the working set W of variables and then selects a column from N to add to W. At each iteration, w_j (i.e., u_j, v_j) in N can be interpreted as $w_j = 0$, or accordingly, $u_j, \ v_j = 0$. Hence once a solution $\boldsymbol{\alpha}^W = \mathbf{u}^W - \mathbf{v}^W$ to the restricted problem is obtained, $\hat{\boldsymbol{\alpha}} = (\boldsymbol{\alpha}^W \ \boldsymbol{\alpha}^N = 0)$ is feasible to the master linear program (4). The following statement examines when an optimal solution for the master problem is obtained in the column generation procedure.

Remark 1 (Optimality of Column Generation). *Let $(\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\beta}})$ be the primal-dual solution to the restricted version of problem (4) with variable b always included in W. The solution is optimal to (4) if and only if for all $j \in N$, $\left| \sum_i \hat{\beta}_i y_i \mathbf{X}_{ij} \right| \le \lambda\gamma_j$.*

To show the optimality is achieved, we need to confirm primal feasibility, dual feasibility and the equality of primal and dual objectives. Recall how we define $\hat{\mathbf{u}} = (\mathbf{u}^W \ \mathbf{u}^N = 0)$ and $\hat{\mathbf{v}} = (\mathbf{v}^W \ \mathbf{v}^N = 0)$, so $(\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\boldsymbol{\xi}})$ is feasible for LP (4). Since the solution is optimal to the restricted problems, the primal objective is equal to the dual objective. Now the key issue to evaluate is the dual feasibility. Since $\hat{\boldsymbol{\beta}}$ is optimal for the restricted problem, it satisfies all constraints of the restricted dual. Hence the dual feasibility is validated if $\left|\sum_i \hat{\beta}_i y_i \mathbf{X}_{ij}\right| \leq \lambda\gamma_j,\ j \in N$. Following the same line of proof, we can show a similar optimality condition for linear program (2).

Any column that violates dual feasibility can be added. A common heuristic is to choose the column $\mathbf{X}_{\cdot j}$ that maximizes $\frac{1}{\gamma_j}\left|\sum_i \hat{\beta}_i y_i \mathbf{X}_{ij}\right|$ over all $j \in N$. In other words, the column $\mathbf{X}_{\cdot j}$ that solves

$$\tau = \max_{j \in N} \frac{1}{\gamma_j}\left|\sum_i \hat{\beta}_i y_i \mathbf{X}_{ij}\right| \tag{6}$$

will be included in the restricted problem. Compared with original LPBoost in [5], our method encloses negations of weak models $\mathbf{X}_{\cdot j}$ in the hypothesis set. We summarize the column generation steps in Algorithm 1 where $\mathbf{e}$ is a vector of ones of appropriate dimension corresponding to the bias term b.

Algorithm 1. **Column generation for LP (4)**

```
1. Initialize the first column $\mathbf{X}_0 = \mathbf{e}$,
   specify the tolerance tol
2. For $t = 1$ to $T$, do
3.     Solve problem (4) with $\mathbf{X}_{t-1}$,
       obtain solution $(\mathbf{u}^t, \mathbf{v}^t, \boldsymbol{\xi}^t, \boldsymbol{\beta}^t)$
4.     Solve problem (6) to obtain $\tau$,
       and let $\mathbf{z}$ be the solution
5.     If $\tau \leq \lambda + tol$, optimal, break from loop,
       otherwise, $\mathbf{X}_t = [\mathbf{X}_{t-1}\ \ \mathbf{z}]$, continue
6. End of loop
7. $\hat{\mathbf{w}} = \mathbf{u}^t - \mathbf{v}^t$.
```

Similarly, column generation for linear program (2) can be derived and is depicted in Algorithm 2.

Algorithm 2. **Column generation for LP (2)**

```
1. Initialize the first column $\mathbf{X}_0 = \mathbf{e}$,
   specify the tolerance tol
2. For $t = 1$ to $T$, do
3.     Solve problem (2) with $\mathbf{X}_{t-1}$,
       obtain solution $(\mathbf{u}^t, \mathbf{v}^t, \boldsymbol{\xi}^t, \boldsymbol{\beta}^t)$
4.     Solve problem
           $\tau = \max_{j \in N} \left|\sum_i \hat{\beta}_i y_i \mathbf{X}_{ij}\right|$
   to obtain $\tau$,
       and let $\mathbf{z}$ be the solution
5.     If $\tau \leq \lambda + tol$, optimal, break from loop,
       otherwise, $\mathbf{X}_t = [\mathbf{X}_{t-1}\ \ \mathbf{z}]$, continue
6. End of loop
7. $\hat{\mathbf{w}} = \mathbf{u}^t - \mathbf{v}^t$.
```

6. COMPUTATIONAL RESULTS

We validate the cascade linear program (LP) classification algorithm with respect to its generalization performance and computational efficiency. We compared our cascade LP strategy to a single stage 1-norm SVM model constructed using all features, and the commonly-used cascade AdaBoost. We also compared our approach to a greedy search algorithm [26] based on linear discriminant analysis (LDA) in the most recent lung nodule detection system.

6.1 Data and experimental settings

A prototype version of our system (not commercially available) was applied on a proprietary de-identified patient data set. The dataset consisted of 176 high-resolution CT images that were randomly partitioned into two groups: a training set of 90 volumes and a test set of 86 volumes. In total, 129 nodules were identified and labeled by radiologists, among which 81 appeared in the training set and 48 in the test set. The training set was then used to optimize the classification parameters, and construct the final classifier which was then tested on the independent test set of 86 volumes.

The candidate generation algorithm was independently applied to the training and test sets, achieving 98.8% detection rate on the training set at 121 FPs per volume and 93.6% detection rate on the test set at 161 FPs per volume, resulting in totally 11056 and 13985 candidates in the respective training and test sets. There can exist multiple candidates pointing to one nodule, so 131 and 81 candidates were labeled as positive in the training set and test set, respectively. Numerical image features of totally 86 were designed, 10 of which came from the CG step with no extra computational burden, and were used to train the first classifier. The feature set 2 contained size, shape, intensity, template matching features which required on average 15.6 millisec. cpu time per feature per candidate, and was used together with the CG features to learn the second classifier. The multi-scale statistical features depicting sophisticated higher-order properties of nodules comprised the feature set 3 and were used in the final classifier construction together with all other features. These features each on average need 2010 millisec. cpu time for a candidate.

6.2 Generalization performance

During the training phase, the tuning parameters in the greedy LDA approach and the parameters (λ, μ) in our cascade LP approach as well as 1-norm SVM were optimized according to the leave-one-patient-out (LOPO) cross validation performance [8]. The LOPO procedure is, in spirit, similar to leave-one-out. The parameter λ was chosen from choices of $\{0.01,\ 0.1,\ 1,\ 10,\ 100,\ 1000\}$, and μ was chosen from a range $[0.6,\ 0.98]$ with a stepsize 0.02. Parameters at each stage of our cascade LP were tuned to achieve the best LOPO performance. Notice that the first stage of cascade LP does not require μ. The single 1-norm SVM was tuned to attain the best overall LOPO performance. The choice of $\lambda = 1$ turned out to be the best option for both cascade LP and 1-norm SVM on the training data, and μ was selected as 0.96 for 1-norm SVM, and 0.98 for the second and third stages in our cascade. Figure 3 depicts the 3 Receiver Operator Characteristic (ROC) curves of the LOPO performance for classifiers, respectively, obtained by the 3 algorithms. Cascade LP outpermed the single 1-norm SVM, and dominated greedy LDA at the lower end of false positive rates.

In the experiments with cascade AdaBoost, we largely fol-

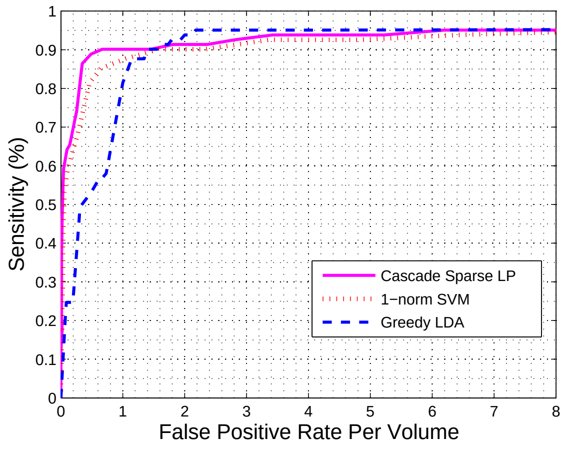

Figure 3: ROC curves show the leave-one-patient-out performance for 3 classifiers on the training data.

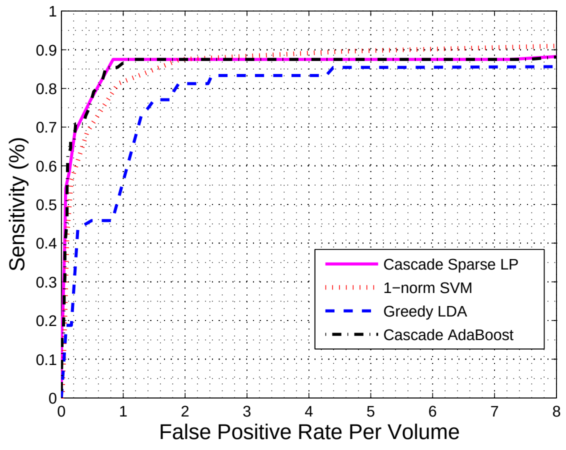

Figure 4: ROC curves of 4 classifiers on the test data.

lowed the procedure described in [24][1]. Each stage classifier was learned using all available features and after a classifier was obtained, the decision threshold was adjusted to minimize false negatives since AdaBoost itself aimed to optimize overall classification accuracy whereas the cascade design requires high detection rates. Cascade AdaBoost does not have hyper-parameters to tune. Instead it requires a validation set. The performance on the validation set is used to determine when to terminate the boosting steps at each stage. Hence other than using LOPO process as used by other approaches, we randomly sampled 30% of the training patient cases and used them as a validation set. Cascade AdaBoost also requires a pre-specified target accuracy which we chose as the minimum acceptable detection rate of 88% at 4 false positives per volume. In our experiment, it reached the target accuracy on the validation set at the 4th stage, so 4 classifiers were constructed in the cascade. The overall training and validation performance is reported in Figure 5. We also tried to only include CG features in the first stage of AdaBoost cascade, but it failed reducing a reasonable amount of non-nodule candidates with a full detection rate (i.e., the CG sensitivity).

The four classifiers obtained respectively by cascade LP, 1-norm SVM, greedy LDA and cascade AdaBoost were evaluated on the unseen test set. The performance is summarized in Figure 4. We see cascade LP and cascade AdaBoost generalize equally well. Greedy LDA seems to be a bit overfiting as LOPO and test ROC curves have a large gap. The two cascade approaches outpermed the other two methods significantly with a t-test p-value close to 0.

6.3 Selected features and speed

The numbers of features selected from different feature sets are listed in Table 2 together with the numbers of candidates

[1]The AdaBoost approach [11] used in [24] has been implemented by other sources. A MatLab version of the implementation was downloaded from MatLab Statistics web page *http://www.mathtools.net/MATLAB/Statistics/*

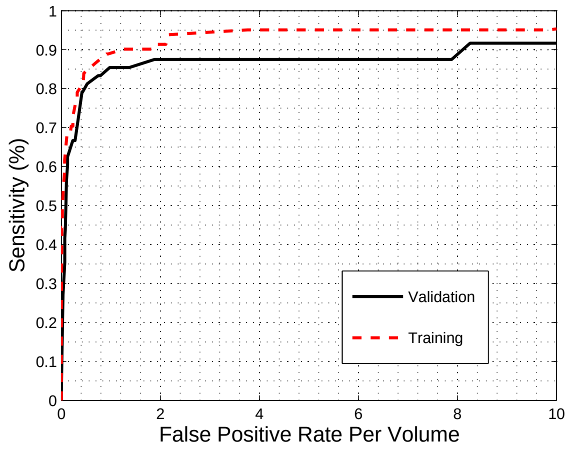

Figure 5: ROC curves of cascade AdaBoost classifier on the validation set and training set.

remained after corresponding stages or classifiers. Features listed at the columns corresponding to later classifiers are the features selected different from those in previous stages. Notice that the computational cost mainly came from feature evaluation since all the four algorithms adopted linear classifiers which cost ignorable time in comparison with the time for feature calculation.

Clearly, cascade LP demonstrates computational efficiency. Its 3 stages together selected 18 features, and only 2 of them were from feature set 3 which were evaluated only for 393 candidates in the last stage. The final system achieved 87.5% versus 0.7 FP rate. The first stage significantly reduced the false positive rate from 161 to 34.1. Only 3011 candidates left for further evaluation of image features. The 1-norm SVM single model and greedy LDA approach both selected more features from set 3 and they computed these features for all the candidates, resulting in momentously

	Cascade Sparse LP			ℓ_1-norm SVM	Greedy LDA	Cascade AdaBoost			
	C_1	C_2	C_3			C_1	C_2	C_3	C_4
Feature set 1	2	1	0	7	3	3	0	2	2
Feature set 2	–	8	5	12	7	3	2	3	4
Feature set 3	–	–	2	14	6	2	4	1	2
Number of candidates	3011	393	102	231	298	4384	1452	754	159
Detection percentage	93.6	89.1	87.5	87.5	83.3	93.6	89.6	88.0	87.5
False positives per volume	34.1	3.5	0.7	2.2	3.0	50	16	7.9	1

Table 2: Selected features and performance summary of different classifiers

longer running time ($14 * 2010 + 12 * 15.6 = 28327$ millisec. per candidate and totally 3.9×10^5 sec. on all candidates for "SVM", and $6 * 2010 + 7 * 15.6 = 12169$ millisec. per candidate, and totally 1.7×10^5 sec. for "LDA") in comparison with a total execution time of $3011*8*15.6+393*(5*15.6+2*2010)$ millisec., and approximately 1986 sec. that the cascaded classifier achieved. Although cascade AdaBoost achieved similar generalization performance, it required a significant greater execution time of 9.7×10^4 sec.

7. CONCLUSIONS

We have proposed a novel cascade classification approach based on sparse linear programs for computer aided detection systems. Our approach can handle very large training sets and produce an excellent generalization. In addition, it offers the advantage of producing highly sparse hyperplane classifiers. The proposed cascade algorithm is relatively easy to implement since at each stage of the algorithm only a linear program has to be solved. In general, any linear program solver can be used to optimize the related linear programs. We particularly presented an incremental solver via column generation optimization. The proposed approach prioritizes features with low computational cost to be at the top of the cascade and incorporates the feature computational complexity into the selection of features, resulting into fast CAD systems. Comparisons to other existing lung CAD algorithms on a real dataset consisting of 176 high-resolution CT images illustrate the superiority of the new approach.

Our current approach optimizes hyper-parameters (λ, μ) at each stage to achieve the best performance of that stage. A possible extension of this work is to develop automatic optimization of hyper parameters of individual stages towards the final system performance. Theoretical examination of the system robustness is also an important extension for further research.

8. REFERENCES

[1] S. G. Armato-III, M. L. Giger, and H. MacMahon. Automated detection of lung nodules in CT scans: preliminary results. *Medical Physics*, 28(8):1552 – 1561, 2001.

[2] S.G. Armato-III, M.B. Altman, J. Wilkie, S. Sone, F. Li, K. Doi and A.S. Roy. Automated lung nodule classification following automated nodule detection on CT: a serial approach. *Medical Physics*, 30(6):1188 – 1197, 2003.

[3] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, Inc., New York, NY, 1993.

[4] K. P. Bennett. Combining support vector and mathematical programming methods for classification. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Machines*, pages 307–326. MIT Press, Cambridge, MA, 1999.

[5] K. P. Bennett, A. Demiriz, and J. Shawe-Taylor. A column generation algorithm for boosting. In *Proceedings of the 17th International Conference on Machine Learning*, pages 65–72. Morgan Kaufmann, San Francisco, CA, 2000.

[6] J. Bi, T. Zhang, and K. P. Bennett. Column-generation boosting methods for mixture of kernels. In R. Kohavi, J. Gehrke, W. DuMouchel, and J. Ghosh, editors, *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 521–526, 2004.

[7] S. Buchbinder, I. Leichter, R. Lederman, B. Novak, P. Bamberger, M. Sklair-Levy, G. Yarmish, and S. Fields. Computer-aided classification of BI-RADS category 3 breast lesions. *Radiology*, 230:820 – 823, 2004.

[8] M. Dundar, G. Fung, L. Bogoni, M. Macari, A. Megibow, and B. Rao. A methodology for training and validating a CAD system and potential pitfalls. In *Proceedings of CARS 2004 Computer Assisted Radiology and Surgery*, 2004.

[9] A. Ferencz, E. Learned-Miller, and J. Malik. Building a classification cascade for visual identification from one example. In *Proceedings of International Conference of Computer Vision*, 2005.

[10] Y. Freund and R.E. Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, pages 325-332, 1996.

[11] J. Friedman, T. Hastie and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 38(2):337-374, 2000.

[12] X. Huang, S. Z. Li, and T. Wang. Learning with cascade for classification of non-convex manifolds. *Conference on Computer Vision and Pattern Recognition Workshop*, 5:66 – 72, 2004.

[13] J.S. Lin, S.C. Lo, A. Hasegawa, M.T. Freedman, and S.K. Mun. Reduction of false positives in lung nodule detection using a two-level neural classification. *IEEE Trans. Med. Imag.*, 15(2):216-227, 1996.

[14] J. Lin. Divergence measures based on the Shannon entropy. *IEEE Trans. Info. Theory*, 37(1):145-151, 1991.

[15] X. Lu, G. Wei, J. Qian and A. K. Jain. Learning-based pulmonary nodule detection from multislice CT data. In *Proceedings of CARS 2004 Computer Assisted Radiology and Surgery*, 2004.

[16] D. P. Naidich, J. P. Ko, and J. Stoechek. Computer aided diagnosis: Impact on nodule detection amongst community level radiologist. A multi-reader study. In *Proceedings of*

CARS 2004 Computer Assisted Radiology and Surgery, pages 902 – 907, 2004.

[17] S. G. Nash and A. Sofer. *Linear and Nonlinear Programming.* McGraw-Hill, New York, NY, 1996.

[18] K. Okada, D. Comaniciu and A. Krishnan. Robust anisotropic Gaussian fitting for volumetric characterization of pulmonary nodules in multislice CT. *IEEE Trans. Med. Imag*, 24(3):409-423, 2005.

[19] J. Roehrig. The promise of CAD in digital mamography. *European Journal of Radiology*, 31:35 – 39, 1999.

[20] R.E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297-336, 1999.

[21] G. Shakhnarovich, P. Viola, and B. Moghaddam. A unified learning framework for real time face detection and classification, 2002.

[22] X. Tang, Z. Ou, T. Su and P. Zhao. Cascade AdaBoost classifiers with stage features optimization for cellular phone embedded face detection. In *Lecture Notes in Computer Science (Advances in Natural Computation)*, volume 3612, pages 688–695, 2005.

[23] R. Verschae and J. R. del Solar. A hybrid face detector based on an asymmetrical adaboost cascade detector and a wavelet-bayesian-detector. In *Lecture Notes in Computer Science IWANN (1)*, volume 2686, pages 742–749, 2003.

[24] P.A. Viola and M.J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137-154, 2004.

[25] P.A. Viola and M.J. Jones. Fast and robust classification using asymmetric AdaBoost and a detector cascade. In *Advances in Neural Information Processing Systems*, volume 14, Cambridge, MA, 2002. MIT Press.

[26] M. Wolf, A. Krishnan, M. Salganicoff, J. Bi, M. Dundar, G. Fung, J. Stoeckel, S. Periaswamy, H. Shen, P. Herzog, and D. P. Baidich. CAD performance analysis for pulmonary nodule detection on thin-slice MDCT scans. In H. Lemke, K. Inamura, K. Doi, M. Vannier, and A. Farman, editors, *Proceedings of CARS 2005 Computer Assisted Radiology and Surgery*, pages 1104–1108, 2005.

[27] H. Zhao and S. Ram. Constrained cascade generalization of decision trees. *IEEE Transactions on Knowledge and Data Engineering*, 16(6):727 – 738, 2004.