Sparse Representation Classification via Sequential Lasso Screening

Yun Wang, Xu Chen and Peter J. Ramadge
Department of Electrical Engineering, Princeton University
Princeton, NJ, USA

Abstract—The sparse representation of signals with respect to an over-complete dictionary has been of recent interest in a broad range of applications. One of the most used methods for obtaining sparse codes, the Lasso problem, becomes computationally costly for large dictionaries and this hinders the use of this approach to large-scale decision tasks. Recently, dictionary screening has been used to address this computational issue. In this spirit, we show how sequential Lasso screening can also facilitate faster completion of sparse representation decision tasks, such as classification, without affecting statistical accuracy. Moreover, the sequential screening process allows us to employ an early decision mechanism that can further accelerate classification, possibly at the cost of small decrease in accuracy. We demonstrate empirically for several classification tasks. In particular, for clip-level music genre classification, using scattering features and a new voting scheme, we show that the proposed method yields improved clip classification accuracy and considerable computational speedup.

Index Terms—sparse representations, classification, dictionary screening, sequential decision rules

I. INTRODUCTION

A sparse representation with respect to an over-complete dictionary is an effective means of data representation for an array of applications in computer vision, signal processing, machine learning and statistics. A common way to obtain such representations is by solving the Lasso problem [1]:

$$\min_{w \in \mathbb{R}^p} \frac{1}{2}||y - Bw||_2^2 + \lambda ||w||_1,$$

(1)

where $B$ is the dictionary, with its columns $\{b_i\}_{i=1}^p$ called codewords or features, and $\lambda > 0$ is a regularization parameter.

In particular, sparse representations have been very successful in classification applications. For example, the Sparse Representation-based Classifier (SRC), [2], yields excellent classification accuracy in problems ranging from face recognition [2] to music genre classification [3]. SRC first computes a sparse coding for a test sample using a dictionary of labelled training samples. This can be done by solving (1). Points $y_i$, $i = 1, \ldots, c$, are then constructed using the components of the sparse code corresponding to codewords of class $i$, and the test sample is assigned the label of the closest $y_i$. The regularization parameter in (1) is usually selected using cross validation to maximize accuracy on a validation dataset.

Of the steps in the SRC algorithm, finding the solution of (1) incurs the greatest cost - both in terms of time and memory usage. Though efficient algorithms for solving the Lasso problem exist, for example [4], scalability to ever larger dictionaries is an issue. This is a potential bottleneck to the application of this approach to large-scale decision problems.

Dictionary screening for the Lasso problem, recently proposed and developed in [5]–[6], is one way to address the above computational issue. Basically, before solving an instance of the Lasso problem (1), dictionary screening is used to remove a subset of codewords which are guaranteed to have zero weights in the solution $w$. The problem can then be solved with a smaller dictionary using less time and memory.

Sequential screening, [7], [8], [9], [10], operates as follows. Given a test sample $y$ and a value $\lambda_t$ for $\lambda$, one screens and solves a sequence of Lasso problems for a decreasing sequence $\{\lambda_k\}_{k=1}^N$ until $\lambda_N = \lambda_t$. This results in a sequence of solutions $\{\hat{w}_i\}_{i=1}^N$, where $\hat{w}_i$ is a solution of (1) with $\lambda = \lambda_i$.

Here we investigate the following idea. The structure of sequential screening permits classification decisions at all points in the sequence $\{\hat{w}_i\}_{i=1}^N$. This gives a form of sequential SRC decision process that can terminate and classify before reaching the terminal problem at $\lambda_t$. In particular, we investigate how the SRC classifier can be integrated with the data-adaptive sequential Lasso screening method (DASS) in [10] to yield both fast and accurate classification.

Other work has also exploited methods involving solving Lasso problems for a sequence of regularization parameter values. For instance, homotopy [11] generates the entire Lasso solution path as the regularization parameter decreases towards zero. The sequential screening scheme proposed in [12], first uses homotopy to solve a sequence of lasso problems as $\lambda$ decreases, then in a final step, applies one-shot screening before solving the final desired lasso problem. Sequential Lasso [13], is a variant on homotopy in that it solves a sequence of partially L-1 penalized least squares problem where features with non-zero weights in earlier steps are no longer penalized. In contrast to these methods, the data-adaptive sequential Lasso screening method (DASS) [10] uses feedback from the previous solutions to select the next value of the regularization parameter.

II. BACKGROUND

A. Sparse representation classification

For a classification problem with $c$ classes, SRC works as follows. Fix a labelled dictionary $B = [B_1, B_2, \ldots, B_c] \in \mathbb{R}^{n \times p}$ with the columns of $B_k$ composed of codewords labelled as class $k$, $k = 1, \ldots, c$. For simplicity, assume the columns have unit norm. Given a test sample $y \in \mathbb{R}^n$ (also of unit
norm), we then solve (1) with $\lambda = \lambda_o$ to obtain a sparse coding $\hat{w}_o$ of $y$. The coding $\hat{w}_o$ is a nonlinear function of $y$. The value $\lambda_o$ is usually selected via cross validation to maximize accuracy on a training dataset. Typically, $\lambda_o$ is in the range $[0.1,0.3]\lambda_{max}$, where $\lambda_{max} = \max_i |b_i^Ty|$,[14],[3].

The sparse coding $\hat{w}_o$ is used to produce vectors $\hat{y}_i$, $i = 1,\ldots,c$, via $c$ linear mappings of $\hat{w}_o$ back into the original data space. This is done by forming $\hat{y}_i = B\delta_i(\hat{w}_o)$, $i = 1,\ldots,c$, where $\delta_i: \mathbb{R}^p \rightarrow \mathbb{R}^p$ zeros the coefficients of $\hat{w}$ not associated with the $i$-th class. SRC then selects the class with the least residual: $r_i(y) = \| y - \hat{y}_i \|_2$, $i = 1,\ldots,c$. Thus the SRC decision rule is

$$l(y,\lambda_o) = \arg\min_i \| y - \hat{y}_i \|_2$$

(2)

### B. Data adaptive sequential screening

Given a Lasso problem $(B,y,\lambda)$, dictionary screening uses a predesigned test to determine a subset of codewords $b_i$ with $\hat{w}_i = 0$, for $i = 1,2,\ldots,p$. For example, in a basic form of screening called a Sphere Test, if $|y^TB_i| < \lambda - 1 + \lambda/\lambda_{max}$, then $\hat{w}_i = 0$,[14].

In sequential screening, one screens and solves (1) along a sequence of instances $\{\lambda_k\}_{k=1}^N$ with $\lambda_1 < \lambda_{max} \text{ and } \lambda_N = \lambda_i$. The solution $\hat{w}_{i-1}$ corresponding to $\lambda_{i-1}$ helps screen the next instance using $\lambda_i$, for $i = 2,\ldots,N$. The design of $\{\lambda_k\}_{k=1}^N$ has practical implications for screening performance,[9],[10]. In open-loop sequential screening, the sequence is designed given $\lambda_i$, but is independent of $B$ and $y$. In contrast, the feedback-controlled Data Adaptive Sequential Screening (DASS) scheme proposed in [10] selects $\lambda_i$ after solving the instance with $\lambda_{i-1}$, using feedback from previous solutions in the sequence. This ensures robust and efficient screening along the sequence. See please [10] for complete details of DASS. Using DASS has some key advantages when the sparse representations obtained are used for classification. This is because the feedback mechanism yields consistently strong screening of large dictionaries over a broad range of values of the parameter $\lambda$. As pointed out earlier, in practical classification problems, selecting $\lambda$ via cross-validation typically results in values around $[0.1,0.3]\lambda_{max}$, with $\lambda_{max}$ depending on the specific test sample $y$.

### III. SEQUENTIAL CLASSIFICATION: SRC AND DASS

In a regular application of SRC, we can use DASS to speed up the solution of (1) and hence make SRC more efficient. To do so, we run DASS until $\lambda = \lambda_o$. This produces sequences: $\{\lambda_i\}_{i=1}^N$ and $\{\hat{w}_i\}_{i=1}^N$, with $\hat{w}_N = \hat{w}$ the solution of (1) with $\lambda = \lambda_o$. Then we perform SRC classification using $\hat{w}_o$.

However, combining SRC with DASS offers new opportunities and potential advantages. The sequential nature of DASS means that at each point $i$ in the sequence, we can perform an SRC classification and accumulate the evidence from these classifications as we move along the sequence. At any point, we can terminate the sequence and classify based on the trajectory of accumulated evidence seen to date.

Assume we have upper and lower bounds on $\lambda_o$: $\lambda_u > \lambda_o > \lambda_l$, with $\lambda_u \leq \lambda_l$ and $\lambda_l > 0$. For $\lambda_i < \lambda_u$, we use $\hat{w}_i$ and the SRC classifier to compute the residual $r_{ij}$ for class $j$. The evidence supporting decision $j$ at step $i$ is $e_i(j) = f(r_{ij})$, where $f$ is some predetermined function. For example, we could use: $f(r_{ij}) = (1 - r_{ij})^2$, $f(r_{ij}) = (1 - r_{ij})$, or $f(r_{ij}) = e^{-r_{ij}/\sigma^2}$, for some constant $\sigma^2 > 0$. The accumulated evidence for class $j$ at step $i$ is then

$$s_i(j) = \sum_{\lambda_k \leq \lambda_u} e_k(j).$$

Set a threshold $T > 0$ and terminate the process at step $i$ if $s_i(j) \geq T$ for any $j = 1,\ldots,c$. If the sequence reaches $\lambda_l$ without any $s_i(j)$ hitting the threshold, then select the class with the maximum accumulated evidence at step $N$. This gives the following integrated (SRC, DASS) classifier.

**Sequential (SRC, DASS):**

1. Using DASS, screen and solve (1) using a decreasing sequence $\{\lambda_k\}_{k=1}^N$ generated as the process proceeds.
2. If $\lambda_i < \lambda_u$, compute the residuals $r_{ij}$ and accumulate the evidence $s_i(j) = s_{i-1}(j) + f(r_{ij}))$, for each class.
3. Stop the algorithm and make a final decision if:
   - $\lambda_i = \lambda_l$. In this event, the final decision is:
     $$l(y) = \arg\max_j s_N(j).$$
   - for any step $i$ there is a class $j$ with $s_i(j) \geq T$. In this event, the classification is:
     $$l(y) = \arg\max_{j: s_i(j) \geq T} \{s_i(j)\}.$$
IV. APPLICATION TO MUSIC GENRE CLASSIFICATION

The proposed sequential (SRC, DASS) classification method is quite general and hence applicable to many classification problems. Of particular interest is the problem of music genre classification in which a short music clip must be classified into one of a number of predetermined genres.

In the standard approach to this problem, feature extraction and classification are performed using short music segments called texture windows (TW). A music clip is divided into many partially overlapping texture windows, each TW is classified into a genre, and clip classification is done by voting over the TW classifications.

The reliability of TW classification will naturally vary over the TW′s. Some TW′s can be classified with high confidence while others can′t. As a result, in situations where the number of confident TW classifications is small, a traditional equal-weight majority vote is prone to an incorrect classification.

As an illustration, consider using SRC to classify clips into two classes $i,j$. Let $d_{ij} = r_i(y) - r_j(y)$ be the margin between the residuals used by SRC for TW classification. Fig. 2 shows histograms of $d_{ij}$ for a set of four example music clips. In these examples, $i$ is the correct class. So positive margins correspond to correct classifications. The area under the histogram for negative values is larger than that for positive values. This results in a misclassification of the music clip under equal-weight voting. The TW′s yielding a correct classification generally have a greater confidence in the prediction. This suggests that combining class-evidence across TW′s would yield a more accurate indicator of the class label.

Motivated by the above, we propose to use the evidence provided by each TW to inform clip-level classification. Let TW$_k$ denote the $k$-th texture window. When SRC (or (SRC, DASS)) classifies TW$_k$, an evidence vector $\epsilon_k$ over all classes is available. This can be accumulated over the TW′s to yield:

$$\epsilon = \sum_{k=1}^{m} \epsilon_k,$$

where $m$ is the total number of TW′s in a clip. We then select the class with the highest evidence score as the predicted label of the clip: $l(\text{clip}) = \arg \max_x \epsilon(j)$.

This approach is in the same spirit as the proposed integration (SRC,DASS). The sequential (SRC,DASS) method accumulates evidence as we vary $\lambda$ for each TW. At (SRC,DASS) termination we have an evidence vector $\epsilon_k$ for TW$_k$. We then accumulate this evidence across time (TW′s) and make a final classification based on the combined evidence.

V. EXPERIMENTS

We empirically evaluate the sequential (SRC, DASS) classification scheme in terms of average classification accuracy and average time to classify a test example. We use the following datasets and training/testing procedures.

(1) MNIST: $28 \times 28$ hand-written digit images (60,000 in the training set and 10,000 in the testing set [15]). From the training set we randomly sampled balanced sets of training images (40,000) and validation images (10,000). For training, we randomly select 100 validation images and 5000 training images as the dictionary. We do this 100 times to obtain $\lambda_0$.

For testing, we randomly select 100 testing images and 5000 dictionary images from the training set. We report average results over 100 random selections. We use the evidence function $f(r_{ij}) = (1 - r_{ij})$.

(2) GTZAN: This dataset consists of 100 music clips (30 sec, sampled at 22,050 Hz) for each of ten genres of music [16]. Each clip is divided into 3-second texture windows with 50% overlap between adjacent windows. Each texture window is then represented using either Musical Surface Features (MSF) [17], or a 1st-order scattering vector [18]. The feature dimensions are respectively, 35 and 199. We randomly select a dictionary of 12,000 features and generate test vectors $y$ from the remaining 8,000 features. We use $f(r_{ij}) = (1 - r_{ij})$.

### Discussion of Results

For MNIST, SRC without screening achieves an average accuracy of 0.96 in 2.06 secs/sample. As indicated in Fig. 3, SRC (using DASS to find the sparse coding) gives an average accuracy of 0.96 in an average time of 0.53 seconds. This is about four times faster. Since the integrated (SRC, DASS) method allows early decisions, we expect additional speed improvement. We use thresholds $T$ from 2 to 12. These result in the points plotted in Fig. 3 giving the corresponding accuracy and classification time. Compared with SRC (using DASS), the integrated (SRC, DASS) method further halves the average classification time at a cost of less than 0.002 reduction of average accuracy. See Fig. 3.

For the GTZAN dataset, SRC (using DASS to find the sparse coding) takes an average of 0.3s (resp. 0.91s) per TW, with accuracies of 48%, (resp. 67%) for the MSF data (resp. 1st order scatting data). Per clip it takes 6s (resp. 18.5s), with accuracies of 61% (resp. 70.5%). In comparison, applying the integrated (SRC, DASS) method with MSF, we can improve TW-based accuracy by more than 1% and classify in about half
Fig. 3. Using (SRC,DASS) on MNIST. Top row: Time and accuracy performance of (SRC,DASS). Bottom row: Examples of accumulated evidence under (SRC,DASS), for a test sample in class “1” and in class “3”. The trajectories of competitive labels are also shown. For $T=6$, the test sample “1” is misclassified as “5” and the test sample “3” is misclassified as “8”. The test samples are shown as inserts.

Fig. 4. Using (SRC,DASS) with TW evidence accumulation on GTZAN. Top: Texture window-based and clip-based performance using Musical Surface Features (MSF). Bottom: Texture window-based and clip-based performance using 1st-order scattering.

the time, and improve clip accuracy by 5% in 2/3 the time. Using (SRC,DASS) with first-order scattering, TW-accuracy improves by 6% in half the time and clip accuracy improves by 9% in half the time. See Fig. 4.

VI. CONCLUSIONS

By integrating SRC with Data-Adaptive Sequential Screening (DASS), we were able to speed up the completion of SRC classification beyond what can be achieved using SRC and DASS independently. This incurred a negligible decrease in average classification accuracy on MNIST. When applied to the music genre classification problem, using the same idea of accumulated evidence significantly improved both classification accuracy and computation speed.

Acknowledgement: Research partially supported by NSF grant CIF:1116208

REFERENCES