Blind Signal Classification via Sparse Coding†

Youngjune Gwon*
MIT Lincoln Laboratory
gyj@ll.mit.edu

Siamak Dastangoo
MIT Lincoln Laboratory
sia@ll.mit.edu

H. T. Kung
Harvard University
kung@harvard.edu

Carl Fossa
MIT Lincoln Laboratory
cfossa@ll.mit.edu

***Abstract*—We propose a novel RF signal classification method based on sparse coding, an unsupervised learning method popular in computer vision. In particular, we employ a convolutional sparse coder that can extract high-level features of an unknown received signal by maximal similarity matching against an overcomplete dictionary of filter patterns. Such dictionary can be either generated or learned in an unsupervised fashion from measured signal examples conveying no ground-truth labels. The computed sparse code is then applied to train SVM classifiers for discriminating RF signals. As a result, the proposed approach can achieve blind signal classification that requires no prior knowledge (*e.g.*, MCS, pulse shaping) about the signals present in an arbitrary RF channel. Since modulated RF signals undergo pulse shaping to aid the matched filter detection, our method exploits variability in relative similarity against the dictionary atoms as the key discriminating factor for classification. Our experimental results indicate that we can blindly separate different classes of digitally modulated signals with a 0.703 recall and 0.246 false alarm at 20 dB SNR. Provided a small labeled dataset for supervised classifier training, we could improve the classification performance to a 0.878 recall and 0.141 false alarm.**

I. Introduction

Cognitive radios have emerged as a new means to share radio spectrum, the most expensive resource to build a wireless network. For commercial applications, Dynamic Spectrum Access (DSA) [1] presents a compelling opportunity to improve the utility of radio spectrum resources. Much of contemporary research has viewed cognitive radios as the secondary user of a licensed channel and focused on developing the mechanism to opportunistically access the channel to its maximal spectral efficiency.

While commercial opportunities are promising, the applicability of cognitive radios for tactical networking seems even more adequate. The primary advocate for tactical cognitive radio systems is intelligent decision making that can enhance resiliency against a hostile, fiercely competing radio environment. There has been significant amount of previous work devoted to algorithmic approaches for a cognitive strategy layer, including game-theoretic frameworks [2]–[5] to sequential decision making [6]–[8].

These approaches have provided a strong foundation for cognitive tactical radio systems, yet their performance highly depends on the lower layer capability such as sensing, detection, and inference of radio signals. In order to operate the cognitive strategy layer, our claim is that we require intelligent sensing mechanisms enabled by learning. In this paper, we focus on the development of such mechanisms. Particularly, we use sparse coding [9], a feature learning technique widely used in machine learning, to perform blind and semi-supervised signal classification for cognitive radios.

Our methods are new and unconventional to the field of signal detection and estimation. Our methods can learn over time after bootstrapping with no prior knowledge about RF signals of interest and achieve a 72.6% recall for blind signal classification under a reasonably good SNR. If a labeled dataset were available for semi-supervised training, our classifiers would have achieved a 87.8% recall with 14.1% false alarms, all without any protocol-specific knowledge about modulation of radio signals.

The rest of the paper is organized as follows. In Section II, we provide a comprehensive background on sparse coding. In Section III, we describe a discriminative framework that employs sparse coding as the primary means to extract features from raw data in a powerful classification pipeline. Section IV presents our RF signal classification methods. We propose a method for blind signal classification before presenting a semi-supervised approach under the availability of a labeled dataset. We evaluate the proposed classification methods in Section V. In Section VI, we discuss related work, and Section VII concludes the paper.

II. Sparse Coding Background

This section presents a background on sparse coding and dictionary learning.

A. Sparse Coding

Sparse coding [9] is an unsupervised method to learn a dictionary of overcomplete basis vectors that can represent data efficiently. Each basis vector in the dictionary is also known as an atom. The mathematical objective of sparse coding is to describe an input vector as a *sparse* linear combination of the dictionary atoms.

Fig. 1 explains the sparse coding problem. Given an N-dimensional input $\mathbf{x} \in \mathbb{R}^N$ and dictionary $D \in \mathbb{R}^{N\times K}$, sparse coding seeks for a sparse representation $\mathbf{y} \in \mathbb{R}^K$ that minimizes the loss function

$$J(\mathbf{x}, D) = \min_{\mathbf{y}\in\mathbb{R}^K} \frac{1}{2}\|\mathbf{x} - D\mathbf{y}\|_2^2 + \lambda\psi(\mathbf{y}), \qquad (1)$$

where the first term optimizes the reconstructive error, and the second term is due to regularization to control sparsity of $\mathbf{y}$. The regularization parameter λ weighs in the sparsity penalty

*The author was a Ph.D. student at Harvard University when he wrote this paper.

†This work is sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

for the optimization. The sparse code $\mathbf{y}$ can be thought as a high dimensional (feature) representation. Its dimension K is generally larger than the dimension N of the raw data $\mathbf{x}$.

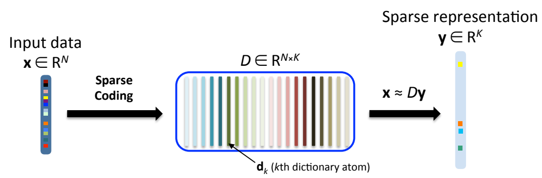

Fig. 1. Sparse coding

Sparse coding considers two regularization strategies for $\psi(.)$. First of all, we can adopt the ℓ_0 pseudo-norm, $\|\mathbf{y}\|_0$, to strictly regulate the total number of nonzero elements in $\mathbf{y}$. Although the ℓ_0 penalty gives a precise control over sparsity of $\mathbf{y}$, it is known to be NP-hard [10].

The second approach of regularization resorts to convex relaxation of the first. Instead of the computationally hard ℓ_0-minimization, we can use the ℓ_1 penalty term on $\mathbf{y}$ instead. There are numerous ways to solve this ℓ_1-regularized optimization problem. In this paper, we consider *basis pursuit* [11], which will be explained shortly. The least absolute shrinkage selection operator (LASSO) [12] or least angle regression (LARS) [13] are other popular methods. It is a well-known result that the ℓ_1-minimization leads to a sparse solution exacting its ℓ_0 counterpart [14].

1) Orthogonal Matching Pursuit: While exact determination via the ℓ_0-minimization is hard, approximate solutions for optimizing ℓ_0-norm are possible. Especially, fast greedy algorithms are possible by selecting the dictionary atoms sequentially from specifically enforcing sparsity requirement such as S-sparse $\mathbf{y}$:

$$\hat{\mathbf{y}} = \arg\min_{\mathbf{y}} \|\mathbf{x} - D\mathbf{y}\|_2^2 \quad \text{s.t.} \;\; \|\mathbf{y}\|_0 \leq S. \tag{2}$$

Orthogonal Matching Pursuit (OMP) [15] selects the best dictionary atom by evaluating the inner product between the input and a dictionary atom and uses least squares to accurately settle the coefficients inside $\mathbf{y}$ iteratively for each round.

2) Basis Pursuit: The ℓ_1-minimization for the sparse coding problem can be written as

$$\hat{\mathbf{y}} = \arg\min_{\mathbf{y}} \|\mathbf{y}\|_1 \quad \text{s.t.} \; D\mathbf{y} = \mathbf{x}. \tag{3}$$

One approach for this optimization is linear programming [16]. Eq. (3), however, is not in the standard dual of a linear program

$$\min \mathbf{c}^\top \mathbf{y} \quad \text{s.t.} \; D\mathbf{y} = \mathbf{x}, \;\; \mathbf{y} \geq 0.$$

Chen, Donoho & Saunders [11] recommend make the following translations

$$\mathbf{y} \Leftrightarrow (\mathbf{u}, \mathbf{v}), \;\; \mathbf{c}^\top \Leftrightarrow (\mathbf{1}^\top, \mathbf{1}^\top), \;\; D \Leftrightarrow (D, -D).$$

Subsequently, solving

$$\min \mathbf{u} + \mathbf{v} \quad \text{s.t.} \; D\mathbf{u} - D\mathbf{v} = \mathbf{x}, \;\; \mathbf{u},\mathbf{v} \geq 0$$

gives the ℓ_1-minimization solution via linear programming.

B. Dictionary Learning

How can we learn a dictionary for sparse coding? A dictionary is trained by an unsupervised learning algorithm such as K-means clustering. A classical approach [17] examines the projected first-order stochastic gradient descent in a sequence of updates for D

$$D_t = \Pi_{\mathcal{C}} \left[D_{t-1} - \frac{\rho}{t} \nabla J(\mathbf{x}_t, D_{t-1}) \right], \tag{4}$$

where ρ is the gradient step, $\Pi_{\mathcal{C}}$ is the orthogonal projector on $\mathcal{C}$, and unlabeled training examples $\{\mathbf{x}_k\}_{k=1}^{T}$.

In principal component analysis (PCA), we learn a complete set of basis vectors—*i.e.*, a square matrix of eigenvectors. Dictionary learning for sparse coding aims to learn an overcomplete set of basis vectors such that the column dimension of D is larger than its row dimension. (Recall $D \in \mathbb{R}^{N \times K}$, so $K > N$.) The advantage of having an overcomplete bases is that we can better capture structures and patterns inherent in the input data more conveniently.

K-SVD [18] is a fast iterative algorithm for PCA-like basis learning. The inner loop of K-SVD has two phases. First, it performs batch sparse coding with current dictionary. Using the notation $\mathbf{X} = [\mathbf{x}_1 \ldots \mathbf{x}_T]$, the batch sparse coding yield the corresponding matrix of sparse codes $\mathbf{Y} = [\mathbf{y}_1 \ldots \mathbf{y}_T]$ such that $\mathbf{X} \approx D\mathbf{Y}$. In the next phase, K-SVD updates each dictionary atom in D by rank-1 update via singular value decomposition of residual matrix for the atom. K-SVD also updates each sparse code in $\mathbf{Y}$ accordingly. The K-SVD optimization is given by

$$\min_{D,\mathbf{Y}} \|\mathbf{X} - D\mathbf{Y}\|_F^2 \qquad \text{s.t.} \;\; \|\mathbf{y}_k\|_0 \leq S \;\; \forall k. \tag{5}$$

Because of the batch sparse coding phase, K-SVD requires a sparse coder. We can use OMP, ℓ_1-minimization via linear programming or LASSO.

III. Discriminative Sparse Coding Framework

In this section, we present a sparse coding framework for discriminative machine learning tasks (*e.g.*, classification). We explain unsupervised feature learning method based on sparse coding and dictionary training. Given the learned feature mapping, we describe how we can compute feature vectors of an input, train classifiers, and predict a class label.

A. Unsupervised Feature Learning via Sparse Coding

Typically, an unsupervised method is used to learn a feature representation of raw data. Since the feature mapping should be generally applicable and descriptive of all classes of data, feature learning takes in randomly mixed, unlabeled training examples. Sparse coding and dictionary training provide an unsupervised feature learning algorithm that consists of the following steps as illustrated in Fig. 2:

1) Form input patches $\mathbf{x}$ from measured/received signal data that are unlabeled of their classes;
2) (Optionally) apply preprocessing such as normalization and whitening;

3) Learn a feature-mapping via joint sparse coding (compute $\mathbf{y}$) and dictionary (D) training.

In summary, unsupervised feature learning takes the unlabeled dataset $\mathbf{X} = [\mathbf{x}_1 \ldots \mathbf{x}_T]$ of random input patches (each $\mathbf{x}_i$ with dimension N), undergoes sparse coding and dictionary learning, and yield a function $f_{\text{ext}} : \mathbb{R}^N \mapsto \mathbb{R}^K$. The transformation via f_{ext} converts the raw data input $\mathbf{x}$ to sparse code $\mathbf{y}$ in the feature space learned by sparse coding and dictionary training. For classification, we use the sparse code $\mathbf{y}$ as a feature vector whose K elements are features of the input $\mathbf{x}$ according to dictionary D.

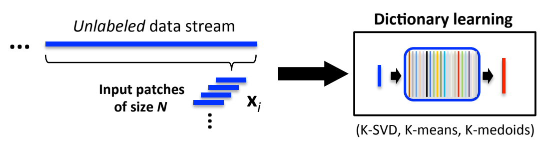

Fig. 2. Unsupervised learning via sparse coding and dictionary training

B. Supervised Classifier Training

A representational feature mapping learned from the unsupervised method plays a crucial role for classification tasks. Having the feature mapping alone, however, is usually insufficient to classify data. Classifiers take a feature vector as the input, and they should be instructed with the ground truth class (*i.e.*, supervision) about the feature inputted. Therefore, classifier training is typically done by a supervised method such as logistic regression [19] and support vector machine (SVM) [20]. Supervised classifier training is depicted in Fig. 3. Note the labeled input $\{\mathbf{x}_i, l_i\}$, where l_i designates the class label for an input $\mathbf{x}_i$.

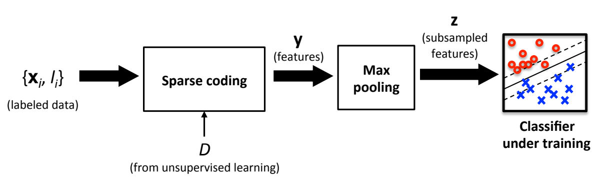

Fig. 3. Supervised classifier training

Under the context of tactical networking scenarios, it may be too optimistic to assume the availability of labeled dataset for supervised classifier training. This is because of the null *a priori* assumption for an adversarial radio network. Signal examples of the adversary could be hard to acquire for pre-analysis before a field operation. However, we can assume a plenty of signal examples for the friendly network. With only friendly network signal examples, one can employ one-class classifier [21] instead.

C. Subsampling Features with Max Pooling

In Fig. 3, feature vectors (*i.e.*, sparse code $\mathbf{y}$) go through one more processing step known as *max pooling* before being inputted to a classifier under training. If all feature vectors resulted from a stream of input vectors were used straightforwardly for classification, we could overwhelm the classifier training. The dimensionality of feature vectors is highly correlated with the complexity of classifiers. Usually, a complex classification model leads to classifier overfit, which is the discrepancy in the classification results between the training and test datasets. It is therefore customary to reduce the number of extracted features by subsampling.

Pooling, popular in convolutional neural networks [22], operates over multiple (sparse) feature representations and aggregates to a higher level of features in reduced dimension. Pooling is by no means to discard any useful information. An important property of the pooled feature representation is translation invariance. Max pooling [23] takes the maximum value for the elements in the same position over a group of feature vectors. For example, consider max pooling of L sparse codes $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_L\}$ that yields the pooled feature vector $\mathbf{z}$ as in Fig. 4. Noting $\mathbf{y}_k = [y_{k,1} \ \ldots \ y_{k,K}]$ and $\mathbf{z} = [z_1 \ \ldots \ z_K]$, max pooling operation is given by $z_j = \max(y_{1,j}, y_{2,j}, \ldots, y_{L,j})$.

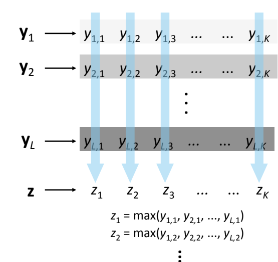

Fig. 4. Max pooling of L sparse codes

IV. RF Signal Classification with Convolutional Sparse Coder

This section introduces a new method for RF signal classification based on feature extraction via sparse coding. We consider two case scenarios. In the first scenario, we consider that there is no labeled dataset for supervised classifier training. Here, we completely rely on unsupervised learning by sparse coding and dictionary training. The first scenario can be considered as blind source separation in the feature domain. In the second scenario, our approach is based on the semi-supervised learning framework.

A. Sparse Coding Stage

Sparse coding is a customizable feature extractor. The learned features represent a high-level structure of the raw signal examples. The sparse coding setup in Fig. 1 is a realization based on matching or basis pursuits that emphasize reconstructive representation with the regularization on sparsity. For discriminative purposes, an OMP sparse coder evaluates the membership of a given input $\mathbf{x}$ to each dictionary atom with the inner product. Conceptually, this is equivalent to the way that K-means clustering evaluates the Euclidean distance between data and a cluster centroid, or that the Gaussian mixture model computes the posterior probabilities given $\mathbf{x}$.

Since our eventual goal is classification, we want to optimally configure the sparse coding framework with the most suitable metric that examines the correlation of a received RF measurement to our dictionary atoms. We propose a convolutional sparse coder that maps an input vector $\mathbf{x} \in \mathbb{C}^N$ (samples of received signal) to the feature $\mathbf{y} \in \mathbb{R}^K$ with respect to the matched filter templates in dictionary D. The ith element in $\mathbf{y}$ is chosen by

$$y_i = \max |\mathbf{x} * \mathbf{d}_k|, \quad (6)$$

where $*$ denotes the convolution operator, and $\mathbf{d}_k$ the kth dictionary atom with $k \in \{1, \ldots, K\}$. We impose a sparse regularization on $\mathbf{y}$ that forces a small number $S \ll N$ of nonzero elements.

The underlying principle behind our setup is matched filtering. Mathematically, the nonzero element in the convolutional sparse code $\mathbf{y}$ reflects the maximum correlation between the input $\mathbf{x}$ and the corresponding dictionary atom, which is some matched filter.

What are the matched filter templates that constitute the dictionary? Radio protocols employ specific pulse shaping functions to aid effective detection of known signals for the receiver. According to detection and estimation theory, an optimal matched filter design leverages *a priori* knowledge about the pulse shaping function to maximize SNR.

B. Signal Classification Stage

Essentially, our approach is a semi-supervised pipeline. We have performed unsupervised feature learning via sparse coding and dictionary training. Partly, we generate dictionary atoms by taking variations of a well-known pulse shape filter such as rectangular, Gaussian, and raised cosine. The rest of the dictionary atoms come from training with example signals. These examples are received from various RF channels used to transmit modulated wireless signals. We use K-SVD [18], a generalization of the K-means clustering algorithm, to train the sparse coding dictionary. We note that this unsupervised dictionary learning step essentially performs blind (signal) source separation in the signal's feature (*i.e.*, sparse code) domain.

Given the learned dictionary, we train linear 1-vs-all SVM classifiers. Assuming a multiclass classification problem with M classes, each SVM is trained to classify signal class j against class $k \neq j \; \forall j, k = 1, \ldots, M$. During the runtime test, we take sample measurements, perform sparse coding and subsampling of sparse codes by max pooling, and predict the signal class label using the pooled sparse code with the trained SVMs.

V. Evaluation

In this section, we evaluate our blind and semi-supervised signal classification methods using MATLAB.

A. Signals and Scenarios

We consider target signals S1 and S2, and non-target signals S3 and S4, mixed in radio channels of a given bandwidth. Blind sampling at each channel is triggered by energy detection above a certain threshold. We sample according to the channel bandwidth and store the measured signals for further processing. We assume sufficient amount of received *target* signal examples with labels S1 or S2. The exact specification (*e.g.*, modulation scheme, pulse shaping) about S1 and S2 is unnecessary. If the exact knowledge about pulse shaping were given, we would have performed matched filter detection. The target signals are defined below.

- (S1) Single-carrier: QPSK with rectangular pulse;
- (S2) OFDM: QPSK modulated on-carriers with raised cosine pulse.

We define the non-target signals S3 and S4 as follows. Similarly, we have no knowledge about the non-target signals. Moreover, we do not assume sufficient number of S3 an S4 examples available to us.

- (S3) Single-carrier: QPSK with unknown custom pulse $p(t) = \frac{1}{2}[1 - \cos(\frac{2\pi t}{T_s})]$;
- (S4) OFDM: BPSK, QPSK, 16-QAM modulated on-carriers with $p(t)$.

We consider the following evaluation scenarios.

1) Blind classification with K-means clustering
2) Blind classification with one-class SVM
3) Semi-supervised learning with 1-vs-all SVM

The two blind classification scenarios do not require any labeled non-target examples. This assumption is reasonable in tactical networking where friendly entities typically communicate using known waveforms. We note that K-means clustering is a completely blind scenario requiring no labeled examples from both target and non-target signals.

B. Experimental Methodology

1) Generation and transmission of signals: To generate signals, we have first generated random data bit stream b_k. The baseband signals are generated according to the following digital (I-Q) modulation schemes.

- BPSK: $d_{\mathrm{BPSK}}(t) = \sum_k b_k p(tkT_b)$
- QPSK: $d_{\mathrm{QPSK}}(t) = \sum_k b_{2k} p(tkT_s) + \sum_k b_{2k+1} p(t - kT_s)$
- 16-QAM: $d_{\mathrm{QAM}}(t) = \sum_k i_k p(tkT_s) + \sum_k q_k p(tkT_s)$
- OFDM: generated by `comm.OFDMModulator` method in MATLAB

For 16-QAM, i_k, q_k are the in-phase and quadrature amplitudes, taking values ± 1, ± 3.

As mentioned earlier, we use rectangular, raised cosine, square-root raised cosine, and custom pulse functions for baseband pulse shaping of the baseband modulated waveforms. The final carrier-modulated signal is given by

$$s(t) = A_c[d_I(t)\cos(2\pi f_c t) + d_Q(t)\sin(2\pi f_c t)],$$

where f_c is the carrier frequency, and A_c the carrier amplitude gain. The in-phase and quadrature components $d_I(t)$, $d_Q(t)$ are generated according to one of the I-Q modulation schemes above.

We transmit $s(t)$ through the AWGN channel at 20 dB and 0 dB SNR. Hence, the measurement at a receiver constitutes noisy samples. For each signal class, we generate two datasets `train` and `eval`. We use `train` for sparse coding and SVM training, and `eval` to test classification performances. There are 1,000 training examples in `train`, whereas `eval` contains 10,000 test examples for each signal class. Thus, we are evaluating a scenario where the size of training dataset is substantially smaller than the test set.

2) Data processing and classification pipeline: The data processing and classification pipeline is depicted in Fig. 5. The measured signal samples are vectorized in a size $N = 64$. Note that an I-Q modulated signal is complex-valued, hence the received samples are also complex, *i.e.*, $\mathbf{x} \in \mathbb{C}^{64}$. We can train the dictionary using the received samples via unsupervised K-SVD algorithm (without knowing what ground-truth classes are). We summarize how we partition the sparse coding dictionary D, which has generated and learned dictionary atoms.

1) D has $K = 256$ dictionary atoms (*i.e.*, $D \in \mathbb{C}^{64\times 256}$)
2) Each atom $\mathbf{d}_i$ has matching size $N = 64$ and is complex-valued
3) D is divided to 4 regions—first 30 atoms belong to the family of rectangular pulses, second 30 atoms to raised cosine family, third 30 atoms to square-root raised cosine; the last 166 atoms are trained

We use a convolutional sparse coder whose operation is described by Eq. (6) given a patch $\mathbf{x}$ of the received signal samples. Sparse code has a dimension K and is real-valued, *i.e.*, $\mathbf{y} \in \mathbb{R}^{256}$. We use the max pooling factor $M = 10$. Note that the pooled feature vector $\mathbf{z}$ has the same dimension as $\mathbf{y}$ and is also real.

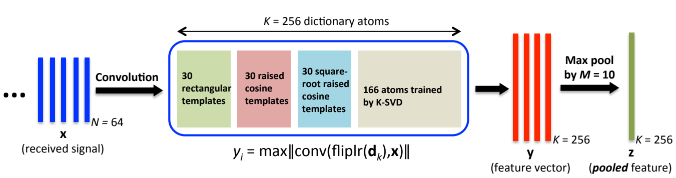

Fig. 5. Sparse coding setup with convolution for RF signal classification

In summary, the feature transformation $\mathbf{x} \mapsto \mathbf{y} \mapsto \mathbf{z}$ takes place by sparse coding of sequentially-fed raw input patches followed by max pooling. The pooled feature vector $\mathbf{z}$ is used for classification.

3) SVM classifier training: One-class SVM is trained by a dataset containing examples from only one signal class. Fig. 6 explains 1-vs-all SVM training. For target signals, we can train two linear SVM classifiers using the pooled feature vectors $\mathbf{z}$. The first SVM classifies the signal class S1 against all others, using labeled datasets $\{\mathbf{z}^{(j)}_{S1}, +1\}^{T}_{j=1}$ and $\{\mathbf{z}^{(j)}_{S2\cup S3\cup S4}, -1\}^{T'}_{j=1}$. (Note that T and T' are the number of examples for target and non-target signals, respectively, for the SVM.) Similarly, the second SVM that classifies S2 against all others are trained with labeled datasets $\{\mathbf{z}^{(j)}_{S2}, +1\}^{T}_{j=1}$ and $\{\mathbf{z}^{(j)}_{S1\cup S3\cup S4}, -1\}^{T'}_{j=1}$.

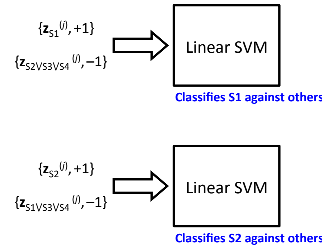

Fig. 6. 1-vs-all SVM training for semi-supervised approach

4) Evaluation metric: We compute confusion matrices with all four signal classes for the three scenarios. We also evaluate recall and false alarm rate for the target vs. non-target classification performance.

C. Results and Discussion

Using K-means, we try to observe separability in the pooled sparse-coded ($\mathbf{z}$) domain. Setting the number of clusters $K = 4$ for K-means (not to be confused with the number of atoms K in dictionary D), we have been able to find reasonably separable clusters that we seek for. We compute recall and false alarm based on majority decoding rule for each cluster. In Fig. 7, we show confusion matrices for the K-means blind classification under SNR = 0 dB and 20 dB. Similarly, we present confusion matrices for blind classification with one-class SVM and the semi-supervised 1-vs-all SVM in Figs. 8 and 9, respectively.

With no labeled examples, sparse-coded feature vectors seem effective for discriminative clustering as we have been able to achieve the average recall of 0.703 with 0.246 false alarm at SNR = 20 dB. Availability of labeled target examples allows us to train one-class SVM, which improves the average recall and false alarm to 0.768 and 0.213. If a labeled dataset for non-target signals were available, our classifiers would have achieved a 0.878 average recall with 0.141 false alarm, all without any protocol-specific knowledge about modulation of radio signals. Table I summarizes the classification accuracy of our approaches. In any case, all of our classification scenarios should be viable for cognitive spectrum sensing.

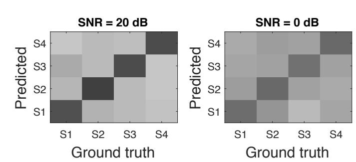

Fig. 7. Confusion matrices for K-means (darkest box: 0.75, lightest: 0.20)

VI. Related Work

Our signal classification methods are inspired by the way that sparse representations of raw image, audio, and text data are used in computer vision and pattern recognition. Wright *et al.* [24] have developed a recognition system that can classify an image of human face using sparse representations of image segments, which is a similar idea to ours. Pooling

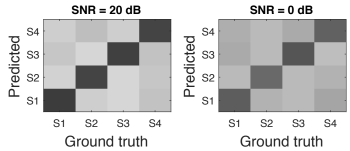

Fig. 8. Confusion matrices for one-class SVM (darkest box: 0.78, lightest: 0.15)

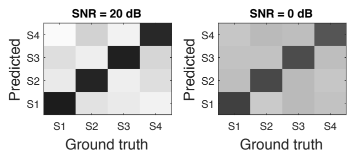

Fig. 9. Confusion matrices for 1-vs-all SVM (darkest box: 0.89, lightest: 0.06)

multiple sparse features to make an aggregate representation is widely studied in computer vision. The original idea of spatial pooling techniques dates back to Riesenhuber and Poggio [25]. Heisele, Ho, and Poggio [26] explain useful techniques of applying SVM for multi-class classification such as training a 1-vs-all classifier in our semi-supervised approach.

TABLE I
CLASSIFICATION ACCURACY
(VALUES IN PARENTHESES ARE FOR SNR = 0 dB)

Scenarios	Recall	False alarm
Blind classification (K-means clustering)	0.703 (0.582)	0.246 (0.367)
Blind classification (One-class SVM)	0.768 (0.634)	0.213 (0.307)
Semi-supervised (1-vs-all SVM)	0.878 (0.726)	0.141 (0.262)

VII. CONCLUSION

We have introduced a blind signal classification method based on sparse coding. Our method is motivated by an active area of research in sparse representation learning [27]. With no prior knowledge or assumptions on an unknown received signal, we take advantage of correlating it to an overcomplete dictionary of (matched) filter patterns, which can be generated offline or learned by an unsupervised learning algorithm. This coding process yields a discriminative feature that captures the variability of correlations measured by convolving the signal with respect to each dictionary atom.

For further improvement of our blind classification tasks, we have regularized the convolved outputs with a sparsity constraint that keeps only the largest several elements. In a simulated experiment similar to blind source separation for modulated RF signals, we have found that our method can achieve up to a 0.703 recall at 0.246 false alarm rate under a reasonably good SNR of 20 dB without any protocol-specific knowledge about simulated radio signals. If a small labeled dataset were available for supervised training, our classifiers would have achieved a 0.878 recall with 0.141 false alarm.

REFERENCES

[1] Q. Zhao and B. Sadler, "A Survey of Dynamic Spectrum Access," *IEEE Signal Processing Magazine*, May 2007.
[2] B. Wang, Y. Wu, K. Liu, and T. Clancy, "An Anti-jamming Stochastic Game for Cognitive Radio Networks," *IEEE JSAC*, vol. 29, no. 4, 2011.
[3] H. Li and Z. Han, "Dogfight in Spectrum: Combating Primary User Emulation Attacks in Cognitive Radio Systems, Part I," *IEEE Trans. on Wireless Communications*, vol. 9, no. 11, pp. 3566–3577, 2010.
[4] S. Sodagari and T. Clancy, "An Anti-jamming Strategy for Channel Access in Cognitive Radio Networks," in *Decision and Game Theory for Security*. Springer LNCS, 2011, vol. 7037, pp. 34–43.
[5] Y. Gwon, S. Dastangoo, C. Fossa, and H. Kung, "Competing Mobile Network Game: Embracing Antijamming and Jamming Strategies with Reinforcement Learning," in *IEEE Conference on Communications and Network Security (CNS)*, 2013.
[6] K. Liu and Q. Zhao, "Distributed Learning in Multi-Armed Bandit With Multiple Players," *IEEE Trans. on Signal Processing*, vol. 58, no. 11, pp. 5667–5681, Nov 2010.
[7] L. Lai, H. Jiang, and H. Poor, "Medium Access in Cognitive Radio Networks: A Competitive Multi-armed Bandit Framework," in *Asilomar Conference on Signals, Systems and Computers*, 2008.
[8] Y. Gwon, S. Dastangoo, and H. Kung, "Optimizing Media Access Strategy for Competing Cognitive Radio Networks," in *IEEE Global Communications Conference (GLOBECOM)*, 2013.
[9] B. Olshausen and D. Field, "Emergence of Simple-cell Receptive Field Properties by Learning a Sparse Code for Natural Images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.
[10] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *Constructive Approximation*, vol. 13, no. 1, pp. 57–98, 1997.
[11] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic Decomposition by Basis Pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.
[12] R. Tibshirani, "Regression Shrinkage and Selection Via the Lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, pp. 267–288, 1994.
[13] B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani, "Least Angle Regression," *Annals of Statistics*, 2004.
[14] E. J. Candes and T. Tao, "Decoding by Linear Programming," *IEEE Trans. on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.
[15] S. Chen, S. A. Billings, and W. Luo, "Orthogonal Least Squares Methods and Their Application to Nonlinear System Identification," *International Journal of Control*, vol. 50, pp. 1873–1896, 1989.
[16] A. Schrijver, *Theory of Linear and Integer Programming*. John Wiley & Sons, 1986.
[17] L. Bottou, "Online Learning and Stochastic Approximations," in *Online Learning in Neural Networks*, 1998, pp. 9–42.
[18] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," *IEEE Trans. on Sig. Proc.*, vol. 54, no. 11, 2006.
[19] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.
[20] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.
[21] L. M. Manevitz, M. Yousef, N. Cristianini, J. Shawe-taylor, and B. Williamson, "One-Class SVMs for Document Classification," *Journal of Machine Learning Research*, vol. 2, pp. 139–154, 2001.
[22] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face Recognition: A Convolutional Neural Network Approach," *IEEE Trans. on Neural Networks*, vol. 8, no. 1, pp. 98–113, 1997.
[23] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A Theoretical Analysis of Feature Pooling in Visual Recognition," in *ICML*, 2010.
[24] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. M., "Robust Face Recognition via Sparse Representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.
[25] M. Riesenhuber and T. Poggio, "Hierarchical Models of Object Recognition in Cortex," *Nature*, vol. 2, no. 11, pp. 1019–1025, 1999.
[26] B. Heisele, P. Ho, and T. Poggio, "Face Recognition with Support Vector Machines: Global versus Component-based Approach," in *ICCV*, 2001.
[27] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.