APPROVAL SHEET

Title of Thesis: COGNITIVE RADIO SPECTRUM SENSING USING ONLINE DICTIONARY LEARNING AND DEEP LAYERED ARCHITECTURES

Name of Candidate: Dan Abid
Master of Science, Electrical Engineering, 2017

Thesis and Abstract Approved: _____
(Seung-Jun Kim)
(Assistant Professor)
(Department of Computer Science and Electrical Engineering)

Date Approved: 12/8/17

NOTE: *The Approval Sheet with the original signature must accompany the thesis or dissertation. No terminal punctuation is to be used.

ABSTRACT

Title of Thesis: COGNITIVE RADIO SPECTRUM SENSING USING ONLINE DICTIONARY LEARNING AND DEEP LAYERED ARCHITECTURES

Dan Abid, CSEE, 2017

Thesis directed by: Professor Seung-Jun Kim
Department of Computer Science and Electrical Engineering

Dictionary learning based on sparse coding has exhibited excellent performance for various tasks such as denoising, prediction and classification with diverse applications. However, sparse coding-based dictionary learning does not capture potential clusters of subspaces in the data. In this work, dictionary learning based on both sparsity and low rank properties is formulated and efficient solution methods are derived in both batch and online implementations. The algorithms are applied to a spectrum sensing problem for cognitive radios. The numerical experiments illustrate the merit of the novel approach. Furthermore, the algorithm is extended to the spectrum prediction problem, where the future interference levels are forecasted. Finally, a RF signal classficiation problem is tackled using a deep layered architecture combining the scattering transform and the convolutional neural network.

COGNITIVE RADIO SPECTRUM SENSING USING ONLINE DICTIONARY LEARNING AND DEEP LAYERED ARCHITECTURES

by

Dan Abid

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County, in partial fulfillment
of the requirements for the degree of
Masters in Electrical Engineering
2017

Advisory Committee:
Dr. Seung-Jun Kim, Chair/Advisor
Dr. E.F. Charles Laberge
Dr. Mohamed Younis

Acknowledgments

I would like to express my appreciation to my supervising professor Dr. Seung-Jun Kim for his help and support during my thesis process. His experience and input has been instrumental in my success and without his help I would still be working on this project. I would like to thank Hao Chen for his input and his hard work in obtaining the scattering network output I used to perform the experiments in Chapter 5.4. I would also like to thank Byron Mcmullen for helping me understand how to obtain the real data used in Chapter 4.3. Finally I would like to especially thank my girlfriend Emily, my parents Ramsey and Christine, my girlfriends parents Jessie and Debbie, my sisters Cindy and Sarah, brother-in-law Kevin, and all my family and friends for pushing me to complete my masters degree.

Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Cognitive Radio Networks

As the need for wireless communication has increased, the limited spectral resources have become a problem. These limited spectral resources have been allocated and licensed, assuming that these frequency bands are constantly used. However, this is not the case in practice [1], [2]. The observation that the bands are used only during a small percentage of the time gave rise to a solution called Cognitive Radios (CRs). CR techniques aim at mitigating the inefficiency of the rigid allocation of spectral resources to wireless services, by monitoring the RF environment in real time and opportunistically re-using under-utilized frequency bands also called "spectrum holes" or "white spaces" [3]. A critical task for CR operation is spectrum sensing(see Fig. 1.1). CRs sense the RF environment in which they operate and learn to make decisions for dynamic spectrum access [4](see Fig. 1.2). A basic objective of spectrum sensing is to detect the on-going transmissions in the frequency band of interest exploiting various features such as interference energy, cyclostationarity, or known pilot sequences. To combat the impediments for reliable detection in a wireless environment, cooperative spectrum sensing is often adopted, where the measurements from a network of CRs are combined together.

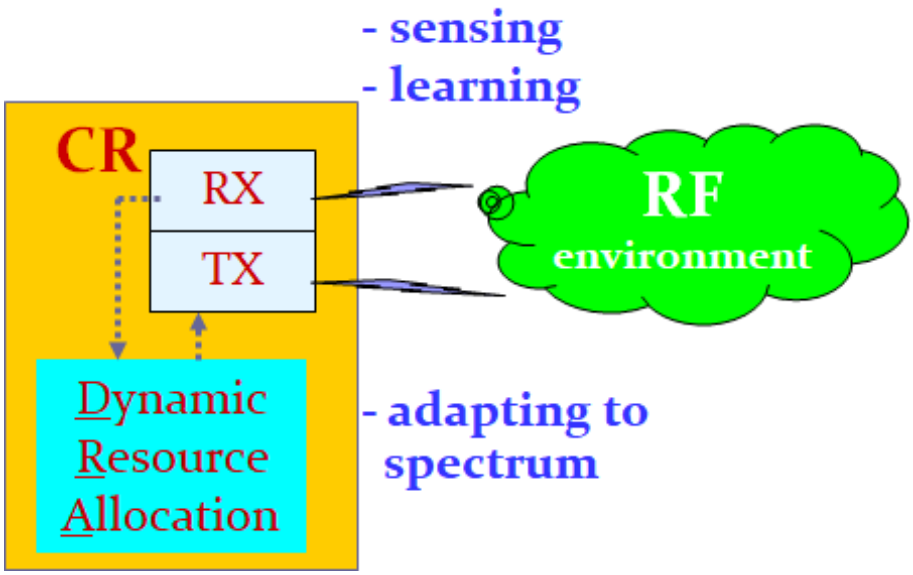

Figure 1.1: Cognitive radios task.

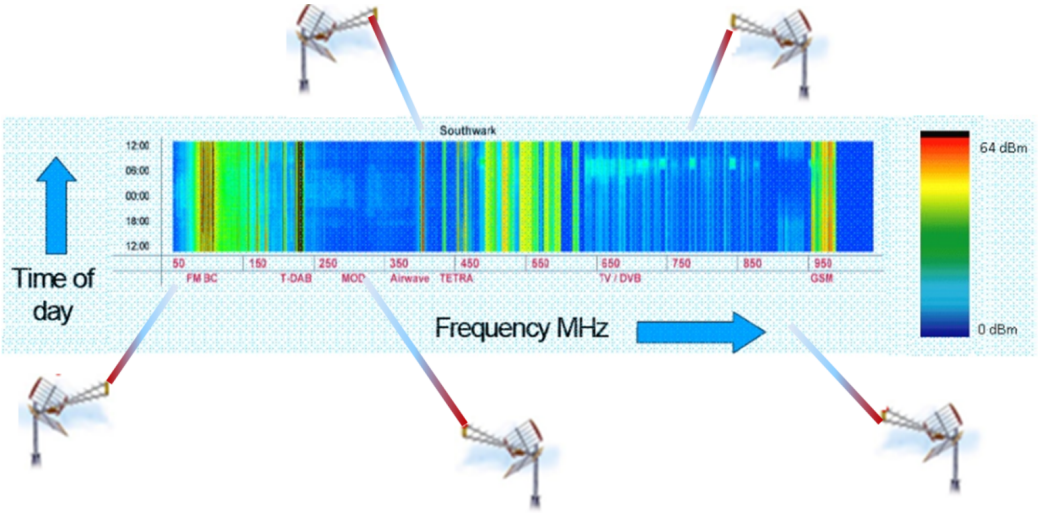

Figure 1.2: Identifying the idle bands in the spectrum.

1.2 Spectrum Sensing

Spectrum sensing is essentially a hypothesis test, whose goal is to test whether the primary user is inactive (the null or noise-only hypothesis), or not (the alternate or signal-present hypothesis). The simple methods originally used to perform spectrum sensing were energy detectors and matched filter detectors [4]. Energy detectors can easily detect if the signal is present by measuring the energy received during a finite time interval and comparing it to a predetermined threshold. The matched filter detector works by convolving the incoming signal with a matched filter. Based on the peaks formed, the detector determines if the received signal is the expected or protected user of the spectrum. Thus, one must know what the characteristics of the expected protected signal. However, due to interference, multi-path effects, or other channel effects, the signal will not always perfectly match the template.

According to the information theory, communication signals that are statistically independent and Gaussian have maximum information content, or entropy. This means that the signals used in communication systems are expected to be nearly white Gaussian. If all signals were like this, then no spectrum sensing algorithm would do better than the energy detector [4]. Luckily, each signal used in communication systems contains distinctive features that can be exploited for detection and enable us to achieve a detection performance better than an energy detector. With feature detection, one can exploit known statistical properties of the signal that occur in both space and time. The features of the transmitted signals are caused by the redundancy added due to the modulation and coding scheme used to transmit the

signal. One of the popular feature detection methods is to estimate the second-order statistics of the received signals and make decisions based on these estimates [4]. For example, based on the fact that most man-made signals show periodic patterns related to the symbol rate, chip rate, channel code, or cyclic prefix (CP), the signals can be modeled as second-order cyclostationary random processes [5]. Knowing some of the cyclic characteristics of a signal, one can construct the detectors that exploit the cyclostationarity and benefit from the spectral correlation [6], [7].

Cognitive radios must also be able to sense a wide band of the spectrum. This requires, high sample rates and high power consumption in the constituent A/D converters. To fix this, multiple methods have been developed to address these design constraints. One solution is to divide the wideband channel into multiple parallel narrowband channels and to jointly sense transmission opportunities on those channels. This technique is called multi-band sensing. Fig. 1.3 illustrates multi-band sensing.

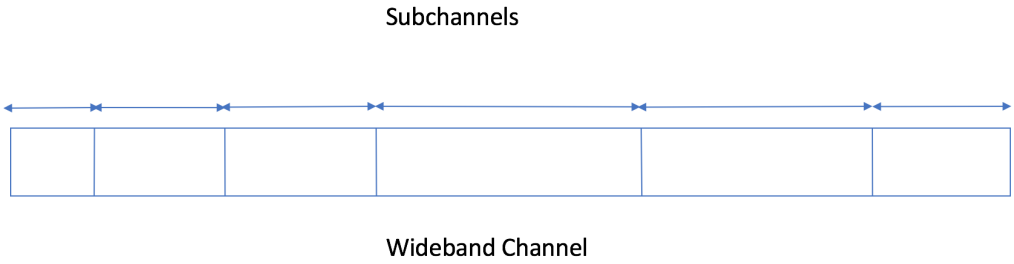

Figure 1.3: Example of a wideband channel divided into multiple subchannels.

The problem of multi-band sensing is deciding which of the subchannels are occupied and which are available. A simple assumption often made for tractability is that each subchannel is independent. By making this assumption, the problem becomes simple binary tests. Unfortunately, in practice, the subchannels are not independent.

Many researchers have also considered joint spectrum sensing and efficient resource utilization. Some of these methods include maximizing the communication rate or the resource allocation methods based on constrained detection probability rates [8], [9].

Recent methods have efficiently utilized their resources by performing compressive sensing. The idea behind compressive spectrum sensing is to exploit the fact that the original passband signal can often be sampled below Nyquist rate without losing information, if the frequency occupancy is sparse [4]. Cooperative versions of compressive wideband sensing have also been developed [10], [11]. By performing these cooperative versions, radios can determine the presence of a signal from a Primary User (PU), and these results can then be fused in a centralized or decentralized manner. However, a better performance can be achieved by fusing the raw measurements.

Using a single CR to perform spectrum sensing faces a number of limitations. One of the problems stems from the fact that a single sensing device might be limited by energy restrictions. For example, the CR might be located in an area where it might miss the detection due to the fading effects. Even if the CR is blocked from the PU's transmitter, it might not be blocked from the PU as receiver (see Fig. 1.4).

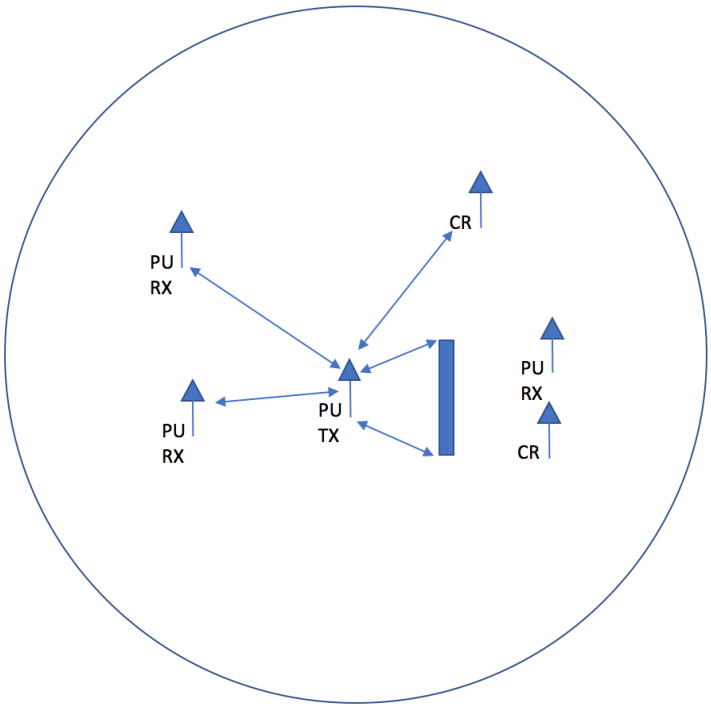

Figure 1.4: Some of the radios cannot see a PU because of the obstacles.

This could cause the CR transmissions to still significantly interfere with the PU receiver. In order to avoid this issue and allow for better performance of the CR, cooperative sensing can be used. The concept of cooperative sensing is to use multiple CR sensors and combine their measurements into one common decision.

There are two major problems cooperative sensing helps overcome. The signal-to-noise-power ratio (SNR) of the signal from the PU at the CR sensor can be extremely small. This is because the propagation path between the PU and CR sensor could be blocked by obstacles. Furthermore, the channel may be faded at a given time and location [12]. In order to combat these problems, the idea of using a network of multiple secondary users (SUs) was adopted to determine the opportune time for the CR to transmit [13], [14]. The authors devised a solution in which a network of SUs would individually make local decisions as to whether or not a primary signal was present in the frequency band they were monitoring. They would then relay their decision to a fusion center to make a final decision. This fusion center would know the geographic location of all PUs and would be responsible for scheduling and assigning channels for all SUs in the network.

This idea was extended based on the fact that even if a frequency band was occupied at one CR location, the power at another location could be much lower. This allows for the reuse of the frequency without interfering with the PU systems. Using kriged Kalman Filtering [15] to help capture channel gains from any location and time to each CR has allowed CRs to interpolate the inference power spatially in real time based on the measurements collected by a network of CRs [16].

1.3 Aim and Scope

Departing from the initial assumption that the spectrum occupancy is constant in the footprint of the CR network [14], recent works advocate estimating spatially varying interference distribution and propagation characteristics [17], [16], which can lead to more aggressive spatial re-use of the scarce spectral resources [18]. In this context, the goal of this work is to interpolate the inference power spatially in real time based on the measurements collected by a network of CRs. The challenges are that the prior knowledge on the number of PUs, their transmit-powers and channel gains are unavailable, and all of these factors are typically needed for coherent combination of the multiple measurements. Furthermore, the temporal correlations of the spectrum occupancy are not known either. The PUs usually do not directly support the estimation of such parameters by the CR network. Therefore, such information must be learned in a blind fashion by the CRs. Moreover, CRs might not be able to report their measurements every time, due to energy-saving sleep modes or congested signaling channels. Thus, the network controller must also account for missing observations.

In this work, dictionary learning based on sparsity and low rank properties is proposed and applied to CR sensing. Conventional dictionary learning postulates that the data can be represented by a linear combination of few template vectors (called atoms), which constitute the dictionary [19]. This means that the coefficients for the linear combination are sparse. However, this does not capture potential *clusters* of subspaces inherent in the data as the correlations across the data vectors are ignored.

Low rank representation-based subspace clustering models that the coefficients, put in a matrix as columns, form a *low-rank* matrix [20]. Thus, dictionary learning based on the low-rank property can extract correlations across the data vectors (or across time in the CR sensing context). In fact, even better performance may be obtained by combining both the sparsity and low rank properties [21]. Various algorithms have been developed for sparsity-based dictionary learning [22], [23]. In this paper, a batch algorithm for the proposed dictionary learning is derived, as well as an online version, which is more suitable for (streaming) big data in general, and addresses the real-time sensing requirement of the spectrum sensing task. Using the online algorithm, predicting the future interference levels is also attempted.

Recently, researchers have been able to use deep layered architectures to classify various signal types such as images or audio. Deep scattering transforms can decompose the signal using layered wavelet transforms without explicit training. We employed the deep scattering transform output as the input to a deep neural network classifier to identify 9 different classes of RF signals at different SNRs.

1.4 Thesis Outline

The present thesis is organized as follows. In Chapter 2, the dictionary learning method is reviewed. In Chapter 3, spectrum sensing using dictionary learning is formulated and the algorithms are derived. Next, an evaluation is done using both synthesized and real data to show that this new approach has significant benefits. In Chapter 4, spectrum prediction is tackled using a modified version of the derived

algorithm. Finally, in Chapter 5, a deep layered architecture is used for signal classification.

Chapter 2: Dictionary Learning and Subspace Models

When sensors collect data, the important information that helps us understand the underlying process is often of a lower dimensionality than that of the recorded data set. This redundancy comes from the fact that the data collected from sensors are often highly correlated. Knowing that the observations can be represented in a lower dimension, one can create a dictionary of atoms to represent the data using sparse combination of the atoms. This chapter reviews dictionary learning and low-rank subspace models.

2.1 Sparse Representation

Before introducing dictionary learning, it is necessary first to understand sparse representation. The sparse representation method [24] has proven to be an extraordinarily powerful solution to a wide range of application fields, especially in signal processing [25], image processing [26], [27], machine learning, and computer vision [28], such as image denoising, de-bluring, inpainting, image restoration, super-resolution, visual tracking, image classification and image segmentation. Using the idea of compressed sensing, one can reconstruct the original signal by exploiting a few measured values if a signal admits a sparse representation over some basis set. Given

a signal vector $\mathbf{x}$ and the collection of spasifying basis $\mathbf{D}$, the sparse representation postulates that $\mathbf{x} \approx \mathbf{Dz}$ where $\mathbf{z}$ is sparse. Mathematically, such $\mathbf{z}$ can be found by solving equation (2.1), where $||\mathbf{z}||_0$ is the number of non-zero entries of $\mathbf{z}$.

$$\min_{\mathbf{z}} ||\mathbf{z}||_0 \quad s.t. \quad \mathbf{x} = \mathbf{Dz}. \tag{2.1}$$

However, since solving this problem requires exponential complexity, a simplified formulation called matching pursuit (MP) is often used [23].

$$\min_{\mathbf{z}} ||\mathbf{z}||_1 \quad s.t. \quad \mathbf{x} = \mathbf{Dz}. \tag{2.2}$$

where ℓ_1-norm (sum of absolute values of the entries). A greedy solution approach called orthogonal matching pursuit (OMP) is often used [29]. Another popular method uses the formulation (2.3) below where λ is a positive parameter controlling the sparsity of $\mathbf{z}$. This algorithm is called least absolute shrinkage selection operator (LASSO) [30].

$$\min_{\mathbf{Z}} \|\mathbf{X} - \mathbf{DZ}\|_2^2 + \lambda \|\mathbf{Z}\|_1. \tag{2.3}$$

2.2 Dictionary Learning

One can learn the spasifying basis $\mathbf{D}$ from a set of data samples, which is called dictionary learning. In order to overcome the issue of high dimensionality and complex statistics of images, researchers created an algorithm to solve for an

over-complete dictionary for image represenation. Olshausen and Field [31], in 1997, created a maximum likelihood algorithm with the intent to demonstrate that the coding in the primary visual area V1 in the human cortex probably follows a sparse coding model. The results found by Olshausen and Field proved that dictionary learning can identify the most important elements in natural images, meaning it could approximate the data in a spare representation.

Collecting the sample vectors $\mathbf{x}_n, n = 1, ..., N$, in a matrix $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]$, the dictionary learning problem can be stated as

$$\min_{\mathbf{D},\mathbf{Z}} ||\mathbf{z}||_0 \quad s.t. \quad \mathbf{x} = \mathbf{DZ} \tag{2.4}$$

where $||\mathbf{z}||_0$ is the number of non-zero entries of $\mathbf{z}$. Vector quantization (VQ) using K-means clustering can be used to perform dictionary learning. First proposed by Schmid-Saugeon and Zakhor [32], this algorithm optimizes the dictionary by grouping patterns so that the distance to any atom is minimal, then updating the atom so that the overall distance in the group of patterns is minimal. Here the assumption is that each image patch represented by an atom has a coefficient equal to one. The K-SVD algorithm tackles this formulation efficiently [22]. By doing this K-means clustering [22] can be used to perform the learning. Using Orthogonal Matching Pursuit (OMP) to perform the sparse approximation step and the dictionary update step is done using singular value decomposition (SVD). While the algorithm is not guaranteed to converge, this algorithm has been used for image denoising.

Another approach is to use

$$\min_{\mathbf{D},\mathbf{Z}} \|\mathbf{X} - \mathbf{D}\mathbf{Z}\|_F^2 + \lambda\|\mathbf{Z}\|_1 \tag{2.5}$$

where $\|\cdot\|_F$ represents the Forbenius norm [23]. Fast online algorithms have been created to quickly perform dictionary learning when the dataset is large [23]. These algorithms overcome this issue by using a subset of the data which is then used with the new training sample. After the new training sample is computed, the optimization is then run on the new sample. Eventually, when all samples have been used, the result converges to the batch solution while drastically lowering the computational complexity.

2.3 Low-Rank Subspace Models

The data often contains a low-dimensional subspace structure which allows for intelligent representation and processing. Based on the low-dimensional subspace structure, one can achieve dimensionality reduction using methods like principle component analysis (PCA), and reconstruct missing entries using matrix completion. However, such methods can model only a single subspace and cannot effectively capture structures that consist of a mixture of multiple subspaces (see Fig. 2.1). Subspace clustering attempts to cluster the data into segments that correspond to separate subspaces. This subspace clustering can be used in problems such as system identification, image processing, and computer vision. While it is easy to solve this

problem when data is clean, one of the main challenges is to handle noise and corrupt data. To this end, the goal of [20] was to create a subspace clustering algorithm, which mitigates corrupted data.

Consider robust PCA as in [33]. Robust PCA attempts to recover a low-rank matrix from the observed matrix and handle the outliers by solving

$$\min_{\mathbf{D},\mathbf{E}} rank(\mathbf{D}) + \lambda||\mathbf{E}||_1 \quad s.t. \quad \mathbf{X} = \mathbf{D} + \mathbf{E} \tag{2.6}$$

In order to handle the union of subspaces, the authors of [20] suggested a more general rank minimization problem defined as

$$\min_{\mathbf{Z},\mathbf{E}} rank(\mathbf{Z}) + \lambda||\mathbf{E}||_{2,1} \quad s.t. \quad \mathbf{X} = \mathbf{A}\mathbf{Z} + \mathbf{E} \tag{2.7}$$

where $\mathbf{A}$ is a dictionary and $\mathbf{Z}$ represents the low-rank coefficient matrix, $||\mathbf{E}||_{2,1}$ is defined as $||\mathbf{E}||_{2,1} = \sum_{n=1}^{N} ||\mathbf{e}_n||_2$ where $\mathbf{e}_n$ is the n-th column of $\mathbf{E}$. Using this norm encourages group sparsity in $\mathbf{E}$, meaning that the entire column $\mathbf{e}_n$ is zero, or the entire column is non-zero. Having $\mathbf{e}_n$ non-zero implies that $\mathbf{x}_n$ is a corrupt data vector. Directly minimizing the rank of $\mathbf{Z}$ is difficult. An efficient algorithm can be derived by replacing the rank function with the nuclear norm.

$$\min_{\mathbf{Z},\mathbf{E}} ||\mathbf{Z}||_* + \lambda||\mathbf{E}||_{2,1} \quad s.t. \quad \mathbf{X} = \mathbf{A}\mathbf{Z} + \mathbf{E} \tag{2.8}$$

where $\|\cdot\|_*$ denotes the nuclear norm, which is the sum of singular values of the matrix. By using the nuclear norm, one can perform the low-rank recovery by

solving a convex optimization problem. This problem can then be solved by using Augmented Lagrange Multiplier (ALM) method.

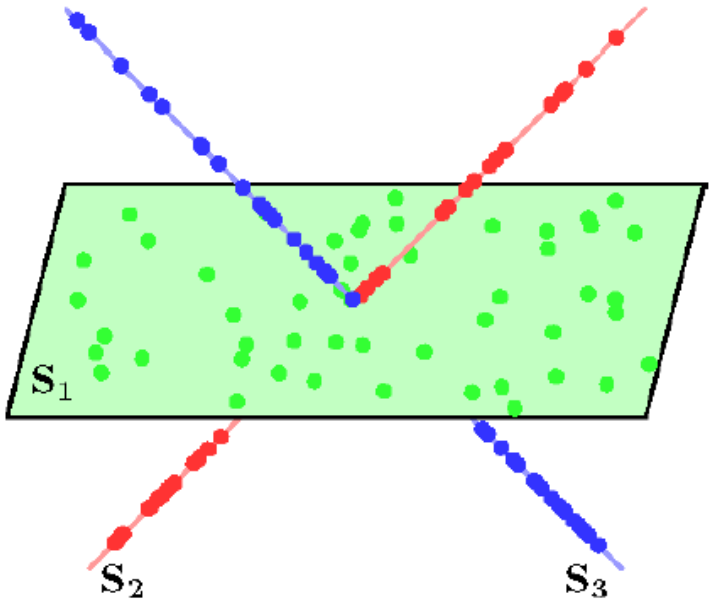

Figure 2.1: Union of subspaces.

Chapter 3: Spectrum Sensing using Dictionary Learning

In this chapter, the spectrum sensing method based on dictionary learning is first reviewed. Then, a novel dictionary learning formulation using both sparsity and low-rank is developed for spectrum sensing. After this, the batch and online algorithms are derived and the performance of these algorithms are investigated. Finally, it is tested how this algorithm performs in the real world using data collected from sensor nodes.

3.1 Prior Approaches

For CRs to transmit without interfering with the PUs, it is useful to estimate the power spectral density (PSD) at any location in space in order to know where the interference is low. Knowing the PSD at any location allows the CRs to dynamically reuse the idle bands and to adapt their transmit-power so that there is minimal interference with the PUs. One of the ways to determine the PSD in space and frequency is to use sparsity, as the number of locations with active transmitters is typically much smaller than the number of potential locations of such transmitters, resulting in spatial sparsity. That is, the active transmitters are distributed in a scarce manner. Moreover, in the under-used spectrum being sensed, only a few

PU transmissions are transmitting at any point in time. This allows one to exploit spatio-temporal sparsity to improve the estimation performance [17].

Another novel idea is to develop channel gain maps to capture the up-to-date channel gains from an arbitrary point in space to the individual CR receiver as a function of frequency [16]. This concept is an improvement over the initial methods of obtaining the PSD because this method could cope with the time-varying environments using kriged Kalman filtering [34], also known as space-time Kalman filtering [15], which exploits the spatio-temporal channel correlation structure. By doing this they were able to obtain an estimate of the entire map by using a finite set of spatio-temporal samples. Once this map is created, it can be used to perform spectrum sensing by sparse regression.

While the kriged Kalman filtering method interpolates channel gains over the entire area, this algorithm had to swap training sequences to collect channel gain measurements. Thus, this method could not learn the channel gains blindly while at the same time detect the ongoing transmissions of the PUs. To fix this issue an online and a distributed algorithm that could track slowly varying channels by local message passing was created [35]. In order to achieve the blind estimation the algorithm took a dictionary learning approach.

3.2 Motivation and Problem Statement

For the operation of CR networks, it is critical to predict spatial distributions of interference power, so that appropriate resource allocation decisions can be made

for the purpose of opportunistically reusing the precious spectral resources. Our focus is to interpolate and predict (Chapter 4) the interference power based on the measurements made by the CRs. Consider a CR network of M nodes, deployed over an area, in which the CRs transmission could disrupt the use of the spectrum by the PU network. The CRs create a network by identifying neighbors and work together in order to obtain the interference levels at each CR node. The goal is to estimate the interference levels at each CR node. Missing information is caused by limitations such as errors and congestion in the control channel, or the fact that radios are in the sleep mode to save battery. Let x_{mn} be the interference power measured by the CR m at time n. Let x_{mn} be the (m, n)-th entry of $\mathbf{X}$, Ω be the set of indices of the *observed* entries in $\mathbf{X}$ and $\mathcal{P}_\Omega$ be the operator that sets the indices in Ω^c to zero; that is, the (m, n)-entry of $\mathcal{P}_\Omega(\mathbf{X})$ is equal to zero if $(m, n) \notin \Omega$ and equal to x_{mn} otherwise. Then the problem to be solved is to estimate $P_{\Omega^c}(\mathbf{x})$ given $P_\Omega(\mathbf{x})$.

3.3 Problem Formulation

In this work, dictionary learning based on sparsity and low rank properties is proposed and applied to CR sensing. Conventional dictionary learning postulates that the data can be represented by a linear combination of few template vectors (called atoms), which constitute the dictionary. This means that the coefficients for the linear combination are sparse. However, this does not capture potential *clusters* of subspaces inherent in the data as the correlations across the data vectors are ignored. Low rank representation-based subspace clustering models that the

coefficients, put in a matrix as columns, form a *low-rank* matrix. Thus, dictionary learning based on the low-rank property can extract correlations across the data vectors (or across time in the CR sensing context). In fact, even better performance may be obtained by combining both the sparsity and low rank properties.

To put this into mathematical terms, let us denote the data matrix as $\mathbf{X} \in \mathbb{R}^{M\times N}$, the dictionary $\mathbf{D} \in \mathbb{R}^{M\times K}$, and coefficient matrix $\mathbf{Z} \in \mathbb{R}^{K\times N}$. The dictionary learning model postulates that $\mathbf{X} \approx \mathbf{DZ}$. Denote the k-th column of $\mathbf{D}$ as $\mathbf{d}_k$. To identify the dictionary, sparsity is often imposed on $\mathbf{Z}$ to solve

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{Z}} \|\mathbf{X} - \mathbf{DZ}\|_F^2 + \lambda\|\mathbf{Z}\|_1 \tag{3.1}$$

where $\mathcal{D} := \{[\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_K] : \|\mathbf{d}_k\|_2 \leq 1\}$ normalizes individual columns (called *atoms*) of the dictionary, $\|\cdot\|_F$ is the Frobenious norm, and $\|\cdot\|_1$ is the ℓ_1-norm (sum of absolute values of the entries), which promotes sparsity. The normalization of the atoms in $\mathbf{D}$ occurs because as the coefficients of $\mathbf{Z}$ become smaller due to the norm penalty, the entries of $\mathbf{D}$ must compensate and become very large (in order to match the difference between $\mathbf{X}$ and $\mathbf{DZ}$). This method is effective in estimating the dictionary and sparse coefficients so that when multiplied together, $\mathbf{x} \approx \mathbf{DZ}$ closely represents the given data.

This method, however, is unable to capture potential clusters of subspaces inherent in the data and ignores the correlations across the data vectors in time. Recently, subspace clustering and low-rank representation models were introduced [20],

[36], which suggest a closely related formulation

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{Z}} \|\mathbf{X}-\mathbf{D}\mathbf{Z}\|_F^2 + \mu\|\mathbf{Z}\|_* \tag{3.2}$$

where $\|\cdot\|_*$ denotes the nuclear norm, which is the sum of singular values of the matrix. Employing the low rank representation allows us to identify subspace clusters.

To achieve a better performance one can create a robust algorithm where the advantages of the both approaches are pursued via

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{Z}} \|\mathbf{X}-\mathbf{D}\mathbf{Z}\|_F^2 + \lambda\|\mathbf{Z}\|_1 + \mu\|\mathbf{Z}\|_*. \tag{3.3}$$

As one of the possible applications of the dictionary learning model, the imputation of missing entries is considered. In the presence of missing entries, dictionary learning can be done via

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{Z}} \|\mathcal{P}_\Omega(\mathbf{X}-\mathbf{D}\mathbf{Z})\|_F^2 + \lambda\|\mathbf{Z}\|_1 + \mu\|\mathbf{Z}\|_*. \tag{3.4}$$

Once $\hat{\mathbf{D}}$ and $\hat{\mathbf{Z}}$ are obtained by solving (3.4), the entries of $\mathbf{X}$ (including the missing ones) can be reconstructed by $\hat{\mathbf{X}} = \hat{\mathbf{D}}\hat{\mathbf{Z}}$.

3.4 Batch Algorithm

To solve (3.4) in batch mode, the alternating direction method of multipliers (ADMM) can be employed [37]. The method is summarized in the Appendix.

Although the convergence of the method for a nonconvex problem as (3.4) is not guaranteed theoretically, the convergence is often observed in practice. Specifically, consider the reformulation of (3.4), where a copy of $\mathbf{Z}$ is introduced as $\mathbf{W}$.

$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{Z}} \|\mathcal{P}_\Omega(\mathbf{X}-\mathbf{D}\mathbf{Z})\|_F^2 + \lambda\|\mathbf{Z}\|_1 + \mu\|\mathbf{W}\|_* \quad s.t. \quad \mathbf{Z}=\mathbf{W}. \tag{3.5}$$

The corresponding augmented Lagrangian is built as

$$\begin{aligned} L(\mathbf{Z},\mathbf{W},\mathbf{D};\mathbf{Y}) &= \|\mathcal{P}_\Omega(\mathbf{X}-\mathbf{D}\mathbf{Z})\|_F^2 + \lambda\|\mathbf{Z}\|_1 + \mu\|\mathbf{W}\|_* \\ &+ \mathrm{tr}\{\mathbf{Y}^T(\mathbf{Z}-\mathbf{W})\} + \alpha\|\mathbf{Z}-\mathbf{W}\|_F^2 \end{aligned} \tag{3.6}$$

where $\alpha > 0$. The ADMM alternately minimizes the augmented Lagrangian over two blocks of variables as well as the Lagrange multiplier. Let us use $\mathbf{Z}$ as the first variable block, and $(\mathbf{D},\mathbf{W})$ as the second block. Then the algorithm is given as follows. At iteration $t = 1, 2, \ldots,$

$$\begin{aligned} \mathbf{Z}^{t+1} &= \arg\min_{\mathbf{Z}} L(\mathbf{Z},\mathbf{W}^t,\mathbf{D}^t;\mathbf{Y}^t) \\ &= \arg\min_{\mathbf{Z}} \left[\|\mathcal{P}_\Omega(\mathbf{X}-\mathbf{D}^t\mathbf{Z})\|_F^2 + \lambda\|\mathbf{Z}\|_1 + \mathrm{tr}\{(\mathbf{Y}^t)^T\mathbf{Z}\} + \alpha\|\mathbf{Z}-\mathbf{W}^t\|_F^2\right]. \end{aligned} \tag{3.7}$$

It can be noted that (3.7) can be decomposed into individual columns. Also it can be noted in (3.7) that the tr $\{\}$ comes from the fact that this algorithm deals with matrices instead of scalars. Let $\Omega_n := \{m : (m,n) \in \Omega\}$ and $\mathbf{O}_n$ be the $|\Omega_n| \times M$ matrix obtained by eliminating from the $M \times M$ identity matrix all rows with row

indices $m \notin \Omega_n$. Upon denoting the n-th column of $\mathbf{Z}$ as $\mathbf{z}_n$ for $n = 1, 2, \ldots, N$, and likewise denoting $\mathbf{X} := [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$ and $\mathbf{Y} := [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N]$, (3.7) is equivalent to

$$\begin{aligned}\mathbf{z}_n^{t+1} &= \arg\min_{\mathbf{z}_n} \left[\|\mathbf{O}_n(\mathbf{x}_n - \mathbf{D}^t\mathbf{z}_n)\|_2^2 + \lambda\|\mathbf{z}_n\|_1 + (\mathbf{y}_n^t)^T\mathbf{z}_n + \alpha\|\mathbf{z}_n - \mathbf{w}_n^t\|_2^2\right] \\ &= \arg\min_{\mathbf{z}_n} \mathbf{z}_n^T\mathbf{Q}_n\mathbf{z}_n - 2\mathbf{q}_n^T\mathbf{z}_n + \lambda\|\mathbf{z}_n\|_1 \end{aligned} \tag{3.8}$$

where

$$\mathbf{Q}_n := (\mathbf{D}^t)^T\mathbf{O}_n^T\mathbf{O}_n\mathbf{D}^t + \alpha\mathbf{I} \tag{3.9}$$

$$\mathbf{q}_n := (\mathbf{D}^t)^T\mathbf{O}_n^T\mathbf{O}_n\mathbf{x}_n - \frac{1}{2}\mathbf{y}_n + \alpha\mathbf{w}_n^t. \tag{3.10}$$

Equation (3.8) results from expanding the ℓ_2-norm and then simplifying the equation. Problem (3.8) can be efficiently solved via a variety of methods [38].

The update for $\mathbf{D}$ and $\mathbf{W}$ are then given by

$$(\mathbf{D}^{t+1}, \mathbf{W}^{t+1}) = \arg\min_{\mathbf{D}\in\mathcal{D},\mathbf{W}} L(\mathbf{Z}^{t+1}, \mathbf{W}, \mathbf{D}; \mathbf{Y}^t). \tag{3.11}$$

It can be easily verified that this optimization with respect to (w.r.t.) $\mathbf{D}$ and $\mathbf{W}$ can be decoupled. That is,

$$\mathbf{D}^{t+1} = \arg\min_{\mathbf{D}\in\mathcal{D}} \|\mathcal{P}_\Omega(\mathbf{X} - \mathbf{D}\mathbf{Z}^{t+1})\|_F^2 \tag{3.12}$$

$$\mathbf{W}^{t+1} = \arg\min_{\mathbf{W}} \mu\|\mathbf{W}\|_* + \alpha\|\mathbf{W} - \mathbf{Z}^{t+1} - \frac{1}{2\alpha}\mathbf{Y}^t\|_F^2 \tag{3.13}$$

Problem (3.12) can be solved by block coordinate descent (BCD) [39]. Problem (3.13) has a closed form solution [40]. Upon denoting the singular value decomposition (SVD) of $\mathbf{Z}^{t+1} + \frac{1}{2\alpha}\mathbf{Y}^t := \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, and the function $S_\tau^+(x) := \max\{0, x - \tau\}$,

$$\mathbf{W}^{t+1} = \mathbf{U}S_{\mu/2\alpha}^+(\mathbf{\Sigma})\mathbf{V}^T \tag{3.14}$$

where $S_{\mu/2\alpha}^+(\cdot)$ is applied entry-wise.

The update rule for $\mathbf{Y}$ is given by

$$\mathbf{Y}^{t+1} = \mathbf{Y}^t + \alpha(\mathbf{Z}^{t+1} - \mathbf{W}^{t+1}). \tag{3.15}$$

The overall algorithm is listed in Table 3.1.

1: For $t = 0, 1, \ldots$

2: Compute $\mathbf{Z}^{t+1}$ via (3.8)

3: Set $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_K] = \mathbf{D}^t$

4: Repeat

5: For $k = 1, 2, \ldots, K$

6: $\bar{\mathbf{d}}_k = \mathbf{d}_k + \left(\sum_{n=1}^{N} \mathbf{O}_n^T \mathbf{O}_n \left(z_{kn}^{t+1}\right)^2\right)^{-1} \cdot \left[\sum_{n=1}^{N} z_{kn}^{t+1} \mathbf{O}_n^T \mathbf{O}_n \left(\mathbf{x}_n - \mathbf{D}\mathbf{z}_n^{t+1}\right)\right]$

7: $\mathbf{d}_k = \bar{\mathbf{d}}_k / \max\{\|\bar{\mathbf{d}}_k\|_2, 1\}$

8: Next k

9: Until convergence

10: Set $\mathbf{D}^{t+1} = \mathbf{D}$

11: Compute $\mathbf{W}^{t+1}$ via (3.14)

12: Compute $\mathbf{Y}^{t+1}$ via (3.15)

12: Repeat until convergence

Table 3.1: Batch algorithm.

3.5 Online Algorithm

To derive an online algorithm, we start again from (3.4). The nuclear norm of a matrix $\mathbf{Z}$ with the rank no larger than R can be written as

$$\|\mathbf{Z}\|_* = \min_{\mathbf{A}\in\mathbb{R}^{K\times R},\mathbf{B}\in\mathbb{R}^{N\times R}} \frac{1}{2}(\|\mathbf{A}\|_F^2 + \|\mathbf{B}\|_F^2)$$
$$\text{subject to } \mathbf{Z} = \mathbf{A}\mathbf{B}^T. \tag{3.16}$$

Using this, the next reformulate (3.4) is considered.

$$\min_{\mathbf{D},\mathbf{A},\mathbf{B}} \|\mathcal{P}_\Omega(\mathbf{X} - \mathbf{D}\mathbf{A}\mathbf{B}^T)\|_F^2 + \lambda\|\mathbf{A}\mathbf{B}^T\|_1$$
$$+ \frac{\mu}{2}(\|\mathbf{A}\|_F^2 + \|\mathbf{B}\|_F^2) + \frac{\nu}{2}\|\mathbf{D}\|_F^2. \tag{3.17}$$

The last term in (3.17) with $\nu > 0$ relaxes the constraint $\mathbf{D} \in \mathcal{D}$ into a form that is more suitable for fast implementation, as will be discussed shortly. Denote the n-th row of $\mathbf{B}$ as $\mathbf{b}_n$, that is, $\mathbf{B}^T = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_N]$. Then, (3.17) can be equivalently written as

$$\min_{\mathbf{D},\mathbf{A},\{\mathbf{b}_n\}} \frac{1}{N}\sum_{n=1}^{N}\left[\|\mathbf{O}_n(\mathbf{x}_n - \mathbf{D}\mathbf{A}\mathbf{b}_n)\|_2^2 + \lambda\|\mathbf{A}\mathbf{b}_n\|_1 + \frac{\mu}{2}\|\mathbf{b}_n\|_2^2\right]$$
$$+ \frac{1}{2N}\left(\mu\|\mathbf{A}\|_F^2 + \nu\|\mathbf{D}\|_F^2\right). \tag{3.18}$$

Invoking the law of large numbers, one can recognize that the sum in (3.18) approaches the expected value. Thus, an online algorithm can be derived based on stochastic

approximation.

First, given the current iterates for $\mathbf{D}$ and $\mathbf{A}$ at time slot n, denoted as $\mathbf{D}_n$ and $\mathbf{A}_n$, respectively, $\mathbf{b}_n$ is updated as

$$\mathbf{b}_n := \arg\min_{\mathbf{b}} \|\mathbf{O}_n(\mathbf{x}_n - \mathbf{D}_n\mathbf{A}_n\mathbf{b})\|_2^2 + \lambda\|\mathbf{A}_n\mathbf{b}\|_1 + \frac{\mu}{2}\|\mathbf{b}\|_2^2. \tag{3.19}$$

To solve (3.19) efficiently, ADMM can again be adopted for the following equivalent problem, where an auxiliary variable $\mathbf{w}$ is introduced.

$$\begin{aligned} &\min_{\mathbf{b},\mathbf{w}} \|\mathbf{O}_n(\mathbf{x}_n - \mathbf{D}_n\mathbf{A}_n\mathbf{b})\|_2^2 + \lambda\|\mathbf{w}\|_1 + \frac{\mu}{2}\|\mathbf{b}\|_2^2 \\ &\text{subject to } \mathbf{A}_n\mathbf{b} = \mathbf{w}. \end{aligned} \tag{3.20}$$

Then, upon defining the soft-thresholding function $S_\tau(x) := \text{sign}(x)\max\{0, |x| - \tau\}$, the resulting ADMM update rule for iterations $t = 1, 2, \ldots,$ and $\alpha > 0$ is given for the iterates $\mathbf{b}^t$ and $\mathbf{w}^t$ as well as the Lagrange multiplier vector $\mathbf{y}^t$ as

$$\mathbf{b}^{t+1} = \left(\mathbf{A}_n^T\mathbf{D}_n^T\mathbf{O}_n^T\mathbf{O}_n\mathbf{D}_n\mathbf{A}_n + \alpha\mathbf{A}_n^T\mathbf{A}_n + \mu\mathbf{I}\right)^{-1}\mathbf{A}_n^T \cdot \left(\mathbf{D}_n^T\mathbf{O}_n^T\mathbf{O}_n\mathbf{x}_n - \mathbf{y}^t + \alpha\mathbf{w}^t\right) \tag{3.21}$$

$$\begin{aligned} \mathbf{w}^{t+1} &= \arg\min_{\mathbf{w}} \frac{1}{2}\left\|\mathbf{w} - \mathbf{A}_n\mathbf{b}^{t+1} - \frac{1}{\alpha}\mathbf{y}^t\right\|_2^2 + \frac{\lambda}{\alpha}\|\mathbf{w}\|_1 \\ &= S_{\frac{\lambda}{\alpha}}\left(\mathbf{A}_n\mathbf{b}^{t+1} + \frac{1}{\alpha}\mathbf{y}^t\right) \end{aligned} \tag{3.22}$$

$$\mathbf{y}^{t+1} = \mathbf{y}^t + \alpha(\mathbf{A}_n\mathbf{b}^{t+1} - \mathbf{w}^{t+1}). \tag{3.23}$$

The update for $\mathbf{D}_n$ and $\mathbf{A}_n$ are based on the stochastic gradient descent method.

For a step size $\rho > 0$, the update equations are given by

$$\mathbf{D}_{n+1} = \mathbf{D}_n + 2\rho \mathbf{O}_n^T \mathbf{O}_n (\mathbf{x}_n - \mathbf{D}_n \mathbf{A}_n \mathbf{b}_n) \mathbf{b}_n^T \mathbf{A}_n^T - \frac{\rho\nu}{N} \mathbf{D}_n \tag{3.24}$$

$$\mathbf{A}_{n+1} = \mathbf{A}_n + 2\rho \mathbf{D}_{n+1}^T \mathbf{O}_n^T \mathbf{O}_n (\mathbf{x}_n - \mathbf{D}_{n+1} \mathbf{A}_n \mathbf{b}_n) \mathbf{b}_n^T - \rho\lambda \text{sign}(\mathbf{A}_n \mathbf{b}_n) \mathbf{b}_n^T - \frac{\rho\mu}{N} \mathbf{A}_n. \tag{3.25}$$

The overall algorithm is given in Table 3.2.

1: For $n = 1, 2, \ldots, N$

2: Receive measurement $\mathbf{x}_n$

3: For $t = 0, 1, \ldots$

4: Update $\mathbf{b}^t$, $\mathbf{w}^t$ and $\mathbf{y}^t$ via (3.21)–(3.23)

5: Until convergence

6: Set $\mathbf{b}_n = \mathbf{b}^t$

7: Compute $\mathbf{D}_{n+1}$ via (3.24)

8: Compute $\mathbf{A}_{n+1}$ via (3.25)

9: Next n

Table 3.2: Online algorithm.

3.6 Spectrum Prediction Algorithm

The algorithms previously mentioned are designed to interpolate missing interference levels of spatial interference distributions, given the incomplete measurements of the current and the past time instants. While these algorithms are important for assessing the proper times to transmit, it is also important to plan ahead and predict future interference levels. For this, it is important to incorporate temporal correlation structures into the model, or learn such structures from the data. This section's goal is to explore how using spatio-temporal prediction can help predict future interference levels. Finally, it will be shown how using the combined approach outperforms both the sparse and low-rank approaches on their own, in the next chapter.

The idea is to leverage the data-driven dictionary learning framework to learn temporal dynamics from the data even in an online fashion. To do this, concatenate the observations over T consecutive intervals into a super-vector, i.e., define,

$$\overline{\mathbf{x}}(t) := [\mathbf{x}(t-T+1), ..., \mathbf{x}(t)]^T. \tag{3.26}$$

Then to perform dictionary learning using $\overline{\mathbf{x}}(t)$ as measurements to obtain dictionary $\overline{\mathbf{D}}(t)$. The algorithm in Table 3.2 can be used. To perform the prediction of the next set of interference levels, one must compute sparse coefficients for a fictitious observation. The fictitious observations are created by $\overline{\mathbf{x}}(t+1) := [\mathbf{x}(t-T+2), ..., \mathbf{x}(t+1)]^T$ assuming $\mathbf{x}(t+1)$ is missing. Then we can find $\hat{\mathbf{x}}(t+1)$

via

$$\hat{\mathbf{x}}(t+1) = \overline{\mathbf{D}}(t)[(T-1)*M+1 : M*T, :]\hat{\mathbf{A}}(t+1)\hat{\mathbf{b}}(t+1) \tag{3.27}$$

where $\overline{\mathbf{D}}(t)[(T-1)*M+1 : M*T, :]$ denotes the last M rows of $\overline{\mathbf{D}}(t)$.

Chapter 4: Numerical Tests

4.1 Batch Algorithm Performance

Consider a CR network consisting of $M = 20$ nodes, whose locations are indicated by the circles in Fig. 4.1. The interference power distribution due to 3 PUs is also depicted in Fig. 4.1, where the emitter locations are clearly revealed as the red spots. The pathloss was set to $(d/d_0)^\alpha$, where d was the distance, $d_0 = 0.01$ and $\alpha = 2.5$. The interference power measurements by the CR nodes in time slot n were collected in the column vector $\mathbf{x}_n$. The measurement noise was assumed to be Gaussian with mean zero and variance 10^{-2}. The missing entries were chosen at random with probability P_{miss}. The number of atoms in the dictionaries was set to $K = 10$. The Rayleigh channel coefficients were randomly generated and fixed. Temporal correlations in the interference distribution were simulated by assuming certain patterns in the PU traffic. Namely, in each time slot n, PU 1 transmits with probability 0.1. If PU 1 does transmit, then PU 2 will transmit in the next time slot. Likewise, per time t, PU 3 transmits with probability 0.15, followed by PU 2 in the next time slot, and PU 1 in the 3rd time slot. As the performance metric, the average normalized mean-square error (MSE) of predicting the missing entries was used, defined as $\|\mathcal{P}_{\Omega^c}(\mathbf{X} - \hat{\mathbf{X}})\|_F^2 / \|\mathcal{P}_{\Omega^c}(\mathbf{X})\|_F^2$.

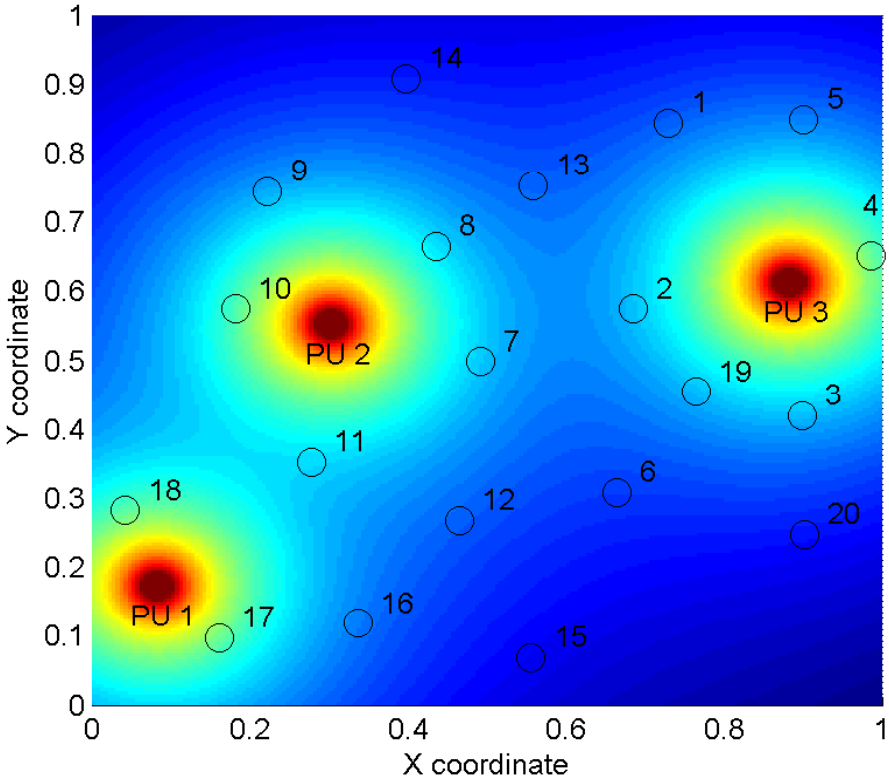

Figure 4.1: CR configuration and interference distribution.

Fig. 4.2 shows the average normalized MSE when sparsity-based dictionary learning is used, that is, when μ is set to 0. The batch algorithm was tested with $N = 6,000$ under different values of P_{miss}. The averages were obtained from 20 independent trials. Parameter λ was varied to find the value that yields the best performance. It can be seen that the algorithm can reconstruct the interference power quite accurately. In particular, when 10% of the data is missing, $P_{miss} = 10\%$, an average MSE of 0.55 was obtained. The MSE performance deteriorates as the missing probability increases, which is reasonable.

Fig. 4.3 depicts the average normalized MSE performance when the low-rank-based dictionary learning was used. Again, the averages were obtained from 20 independent trials. As μ is varied to find the optimal parameter value, it can be observed that the low-rank-based algorithm achieves MSE performances that are better than those from the sparsity-based counterpart. For instance, with $P_{miss} = 10\%$, a normalized MSE of 0.5 was obtained. This suggests that capturing temporal correlation via the low rank property may be more effective in modeling the data than relying on sparsity.

The performance of the batch algorithm is shown in Fig. 4.4 under various P_{miss} values, where $N = 6,000$ samples were processed jointly. The curve with the circle markers represents the best normalized MSE when λ was tuned with $\mu = 0$ found by using the results in Fig. 4.2, the curve with the triangles corresponds to the case of tuning μ with $\lambda = 0$, found by using the results in Fig. 4.3, and the one with the squares was obtained from a 2-dimensional search over (λ, μ)-pair for the best performance. It can be seen that the low-rank property provides a significant

advantage in performance, and the best performance is achieved when both the sparsity and low rank properties are exploited.

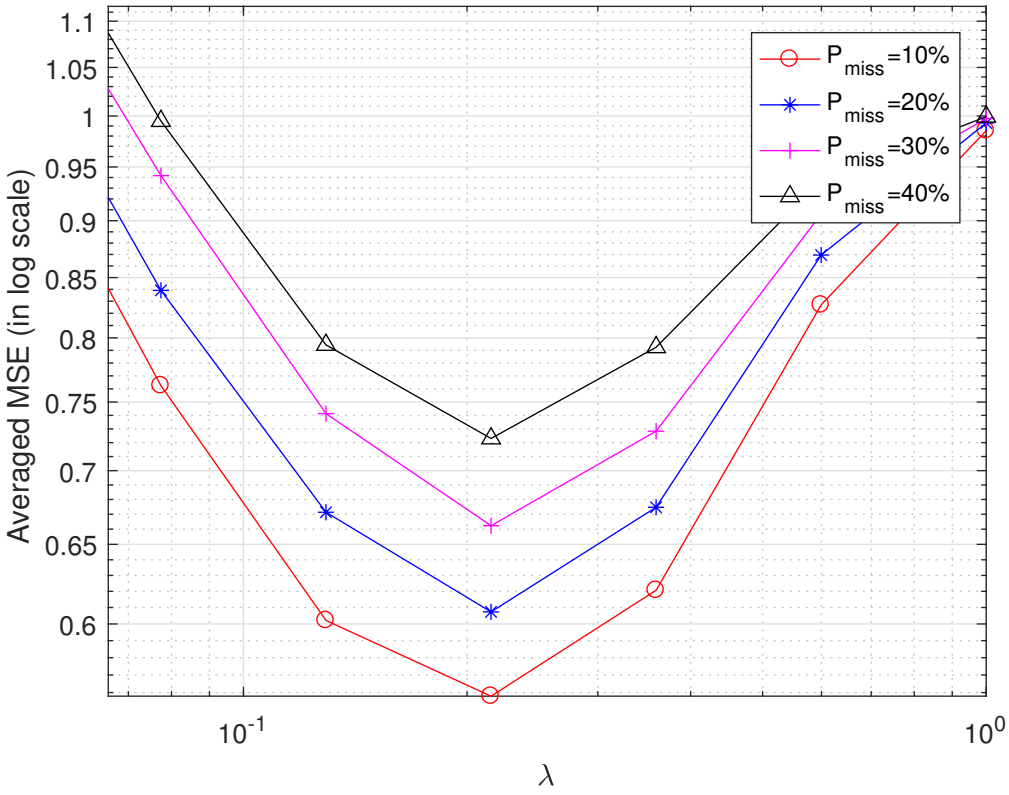

Figure 4.2: Batch prediction using sparsity-based dictionary learning.

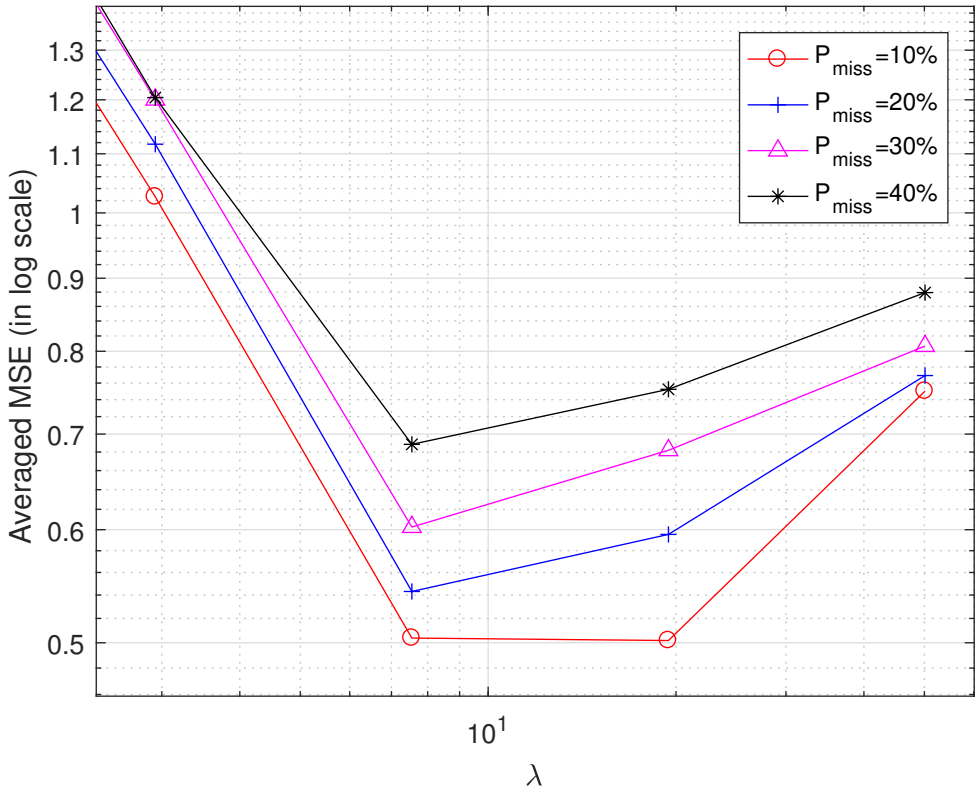

Figure 4.3: Batch prediction using low-rank-based dictionary learning.

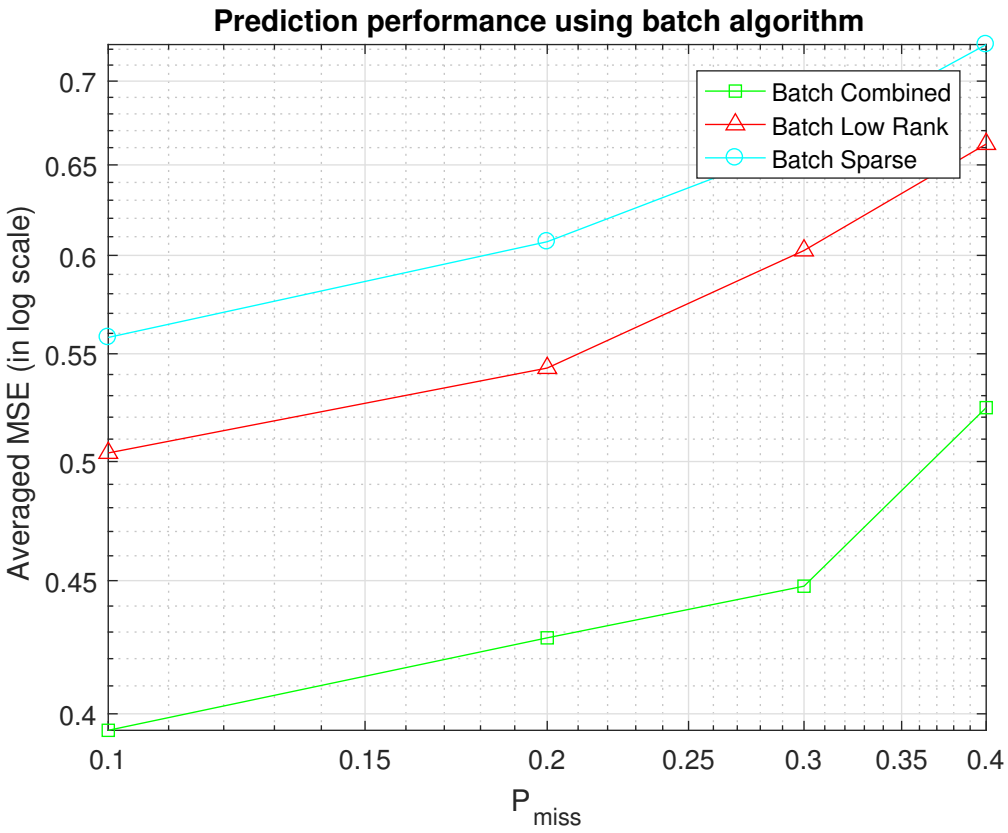

Figure 4.4: Prediction performance using the batch algorithm.

4.2 Online Algorithm Performance

Using the same system model, same data obtained to display the performance of the online algorithm, same number of runs, and same P_{miss}, the next step is to show the performance of the online algorithm. Fig. 4.5 show the best normalized MSE for when λ was tuned with $\mu = 0$, the case of tuning μ with $\lambda = 0$, and the 2-dimensional search over (λ, μ)-pair for the best performance. For the sparse-only case, the blue curve, the parameter λ was varied to find the value that yields the best performance. It can be seen that the algorithm can reconstruct the interference power quite accurately. In particular, when $P_{miss} = 10\%$, an average MSE of 0.59 was obtained. The MSE performance deteriorates as the missing probability increases, which is reasonable. For the low-rank only case, the red curve, the parameter μ was varied to find the optimal parameter value, it can be observed that the low-rank-based algorithm achieves MSE performances that are better than those from the sparsity-based counterpart. For instance, with $P_{miss} = 10\%$, a normalized MSE of 0.45 was obtained. This suggests that capturing temporal correlation via the low rank property may be more effective in modeling the data than relying on sparsity. Finally in the combined case, the green curve was obtained from a 2-dimensional search over (λ, μ)-pair for the best performance. It can be seen that the low-rank property provides a significant advantage in performance, and the best performance is achieved when both the sparsity and low rank properties are exploited.

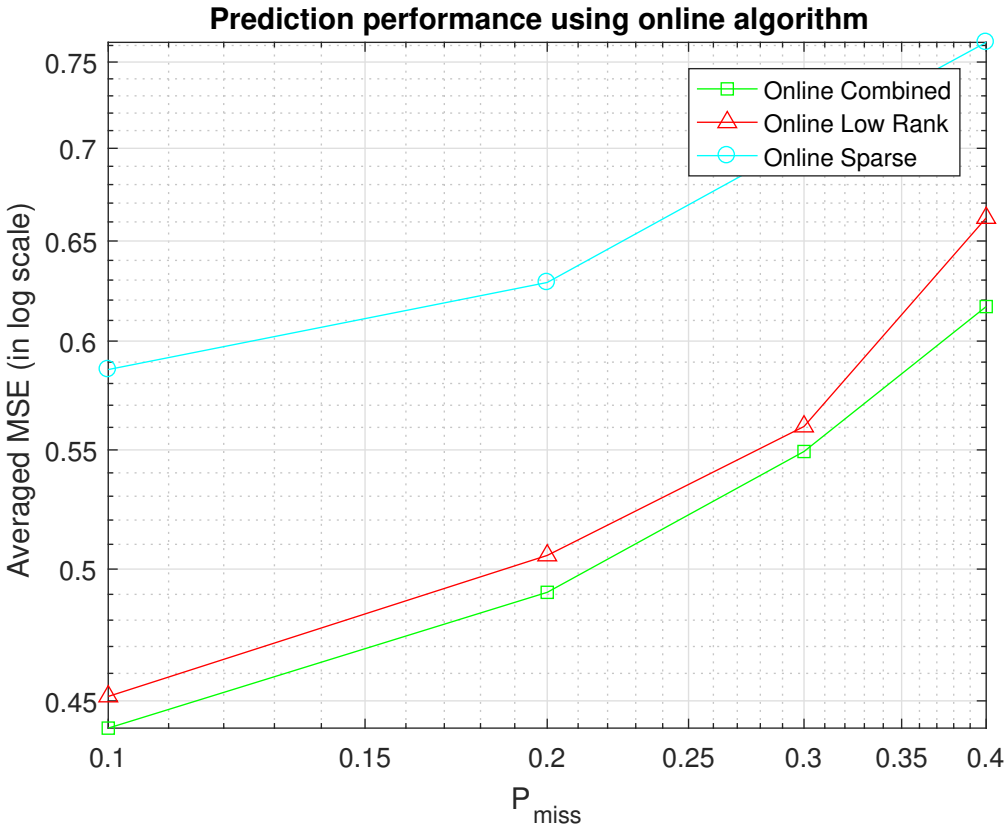

Figure 4.5: Prediction performance using the online algorithm.

4.3 Experiment with Real Data

This new algorithm, exploiting the low rank property and the combining properties, improves the performance on synthetic data over just sparsity alone. It is important to see how the algorithm is affected when data collected from the real world is used. The data used was collected using 7 TelosB sensors, which are IEEE 802.15.4-compliant wireless sensor nodes which collect ambient noise [41]. To see if this data would be viable, one test setup was performed, in which the Wi-Fi was turned off during the evening, in a residential apartment, similar to the one seen in Figs 4.6 and 4.7. Then in two different configurations shown in Figs 4.6 and 4.7, where the numbers represent where each node was located and xxx represents where the Wi-Fi router was located. Collecting measurements with the Wi-Fi router ON and OFF in the both Configurations 1 and 2, giving rise to four different configurations. 1000 time samples were first collected in the test setup. Then 25 sets of 1024 time samples in each of the four configurations were collected. The RSSI measurements were reported by the sensors in dBm and converted to mW based on [41]. The missing entries were simulated by removing random entries in the measurements with probability P_{miss}.

The performance of the online algorithm with the test setup is shown in Fig. 4.8 uses different K values ranging from 5 to 30 and for each K value, also plotted are their different P_{miss} values. This figure was obtained from a 2-dimensional search over (λ, μ)-pair for the best performance for each (K, P_{miss}) combination. Just like the matlab synthesized data, it is expected the MSE performance to deteriorate as

the missing probability increases, here best performance is when K is set to 10 with 10% of the data missing.

The test setup, just mentioned, proved that these sensors were indeed a good next step to take in proving the viability of this algorithm. Therefore it was decided to increase the time samples and wanted to see the effect. The performance of the online algorithm is shown in Fig. 4.9, using configuration 1 with the Wi-Fi OFF found in Fig. 4.6 under various K values and a fixed P_{miss} value, where $N = 27000$ samples were processed. The curve with the circle markers represents the best normalized MSE when λ was tuned with $\mu = 0$, the curve with the triangles corresponds to the case of tuning μ with $\lambda = 0$, and the one with the star was obtained from a 2-dimensional search over (λ, μ)-pair for the best performance. It can be seen that the low-rank property provides an advantage in performance when both the sparsity and low rank properties are exploited. One of the possible reasons the low rank property did not do as well as the sparse case alone is due to how the data was collected. In the matlab synthesized data the 6000 time samples were continuously collected, which means that the correlations between data columns was preserved. However between each of the 25 sets of 1024 time samples we collected there was a time delay. This time delay allows there to be less correlation between data columns especially at the beginning and end of the sets.

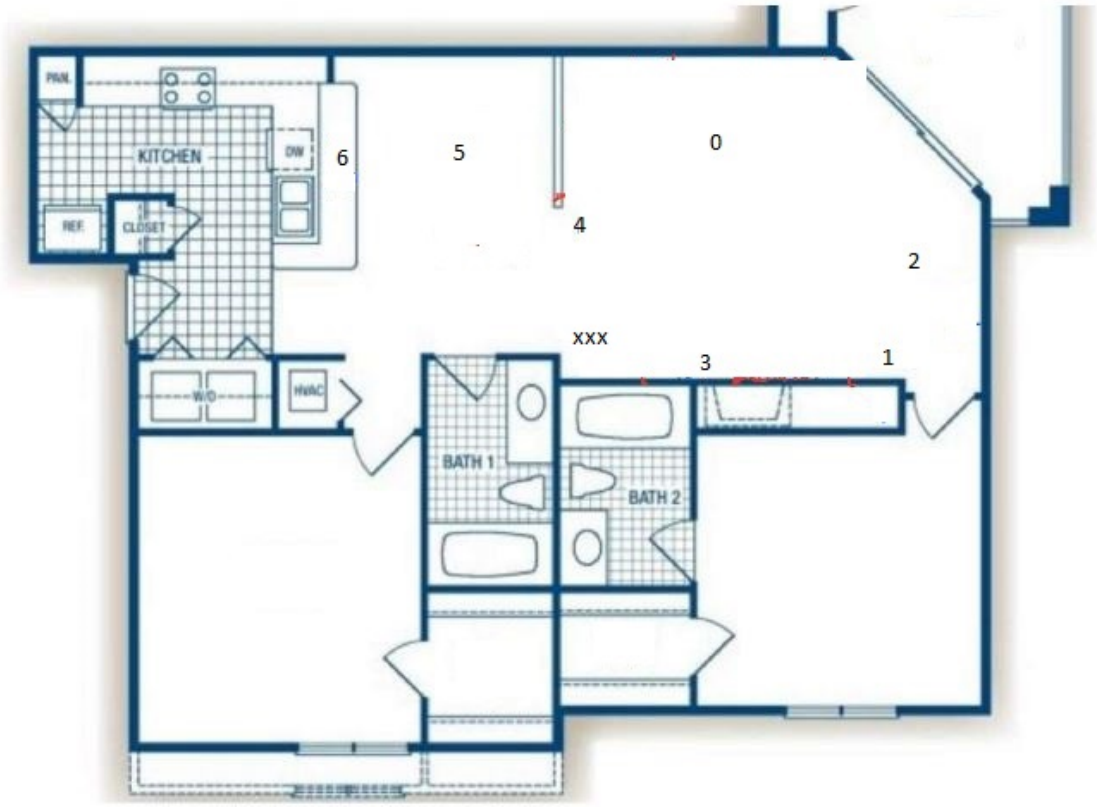

Figure 4.6: Mote configuration 1.

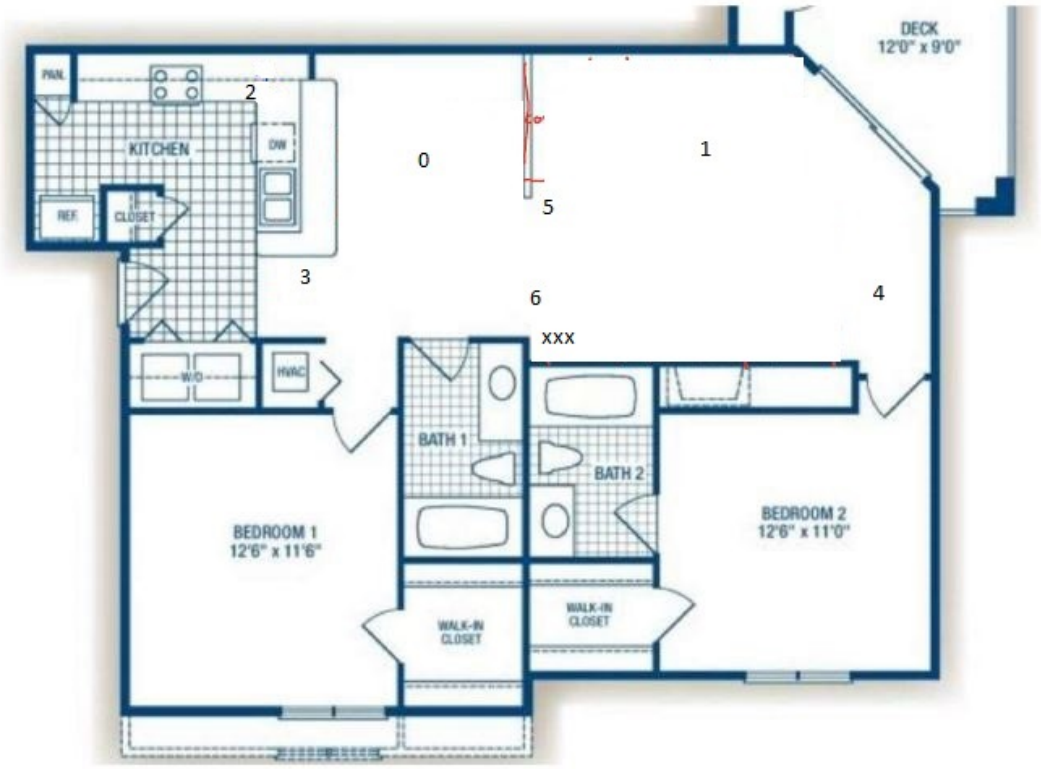

Figure 4.7: Mote configuration 2.

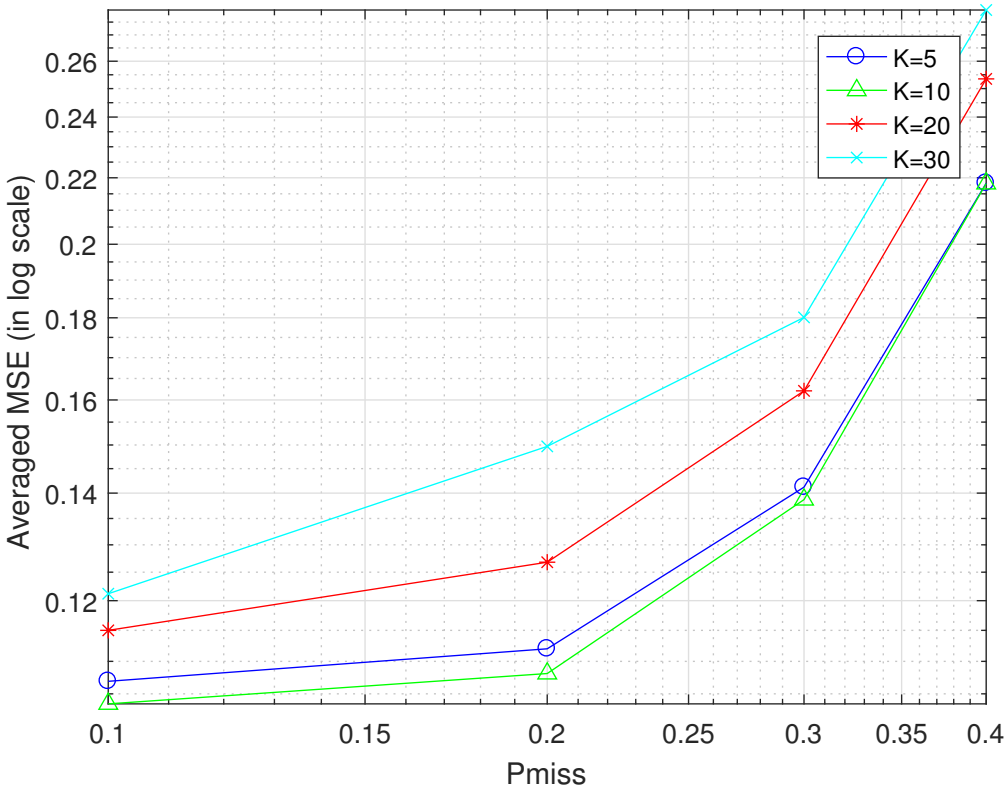

Figure 4.8: Test setup combined approach varying number of atoms and P_{miss}

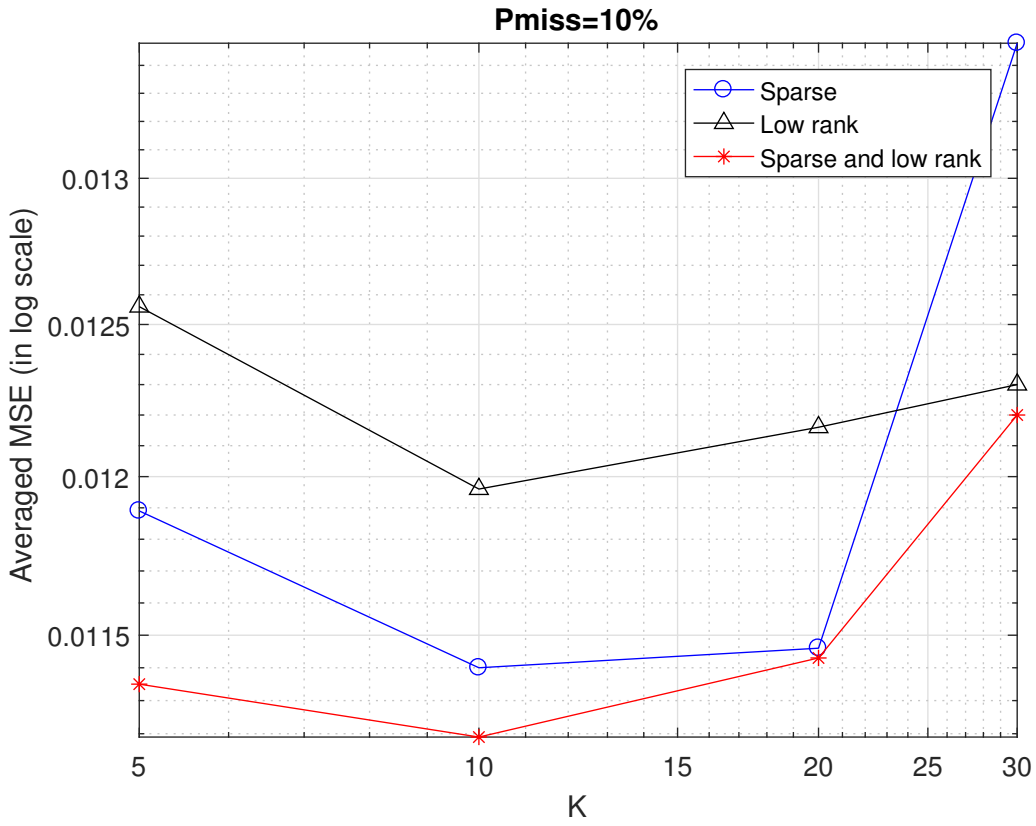

Figure 4.9: Configuration 1 Wi-Fi off combined approach varying K while keeping P_{miss} set to 10%

4.4 Results of Numerical Tests for Spectrum Prediction

In order to test the temporal prediction, we used the same simulation setup as matlab setup in sections 4.1 and 4.2. Fig. 4.10 was obtained from a 2-dimensional search over (λ, μ)-pair for the best performance for each T consecutive intervals (T_{span}) while varying P_{miss}. In this we expect that the larger the span the better the performance compared to the previous T_{span}. Here we wanted to show how the optimal (λ, μ)-pair was found for a particular (P_{miss}, T_{span})-pair. In Fig. 4.11 the best (λ, μ)-pair was found when λ was set to 0.05 and μ was set to 184.8. Fig. 4.12 was obtained by tracking a particular (λ, μ)-pair for a certain P_{miss} while varying T_{span}. It can be seen that increasing the T_{span} for the most part improves the performance, as one increase the λ, the performance improves as one increase the T_{span} and the reverse occurs when one lowers the λ and increase the T_{span}.

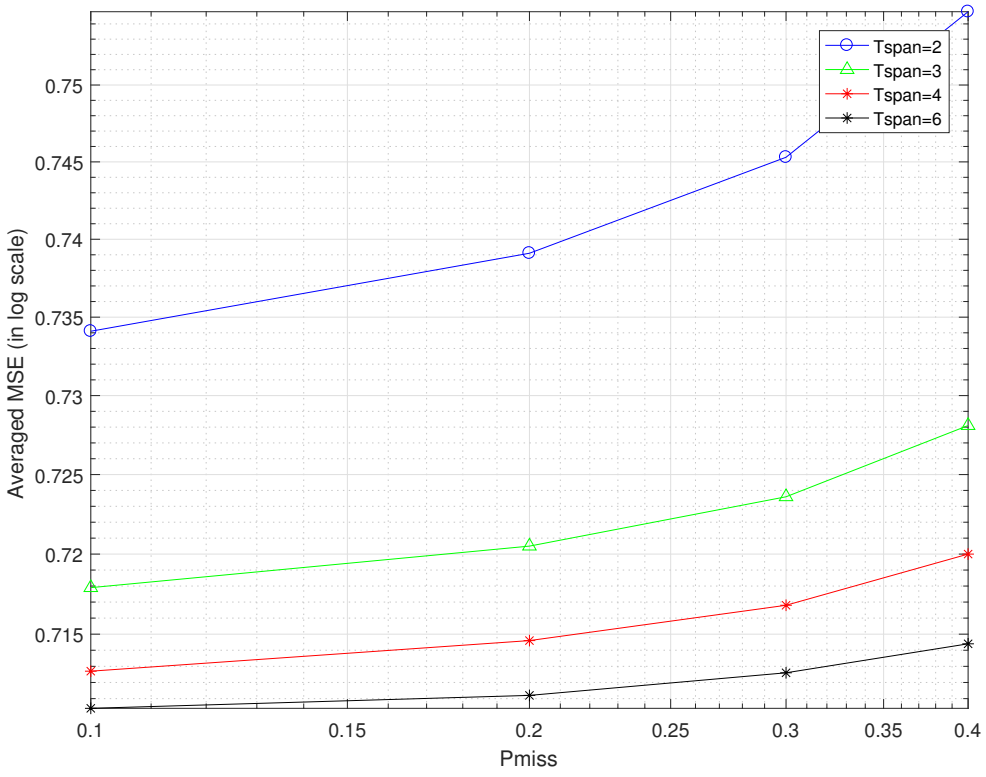

Figure 4.10: Performance of interference prediction for the optimal (λ, μ) pair.

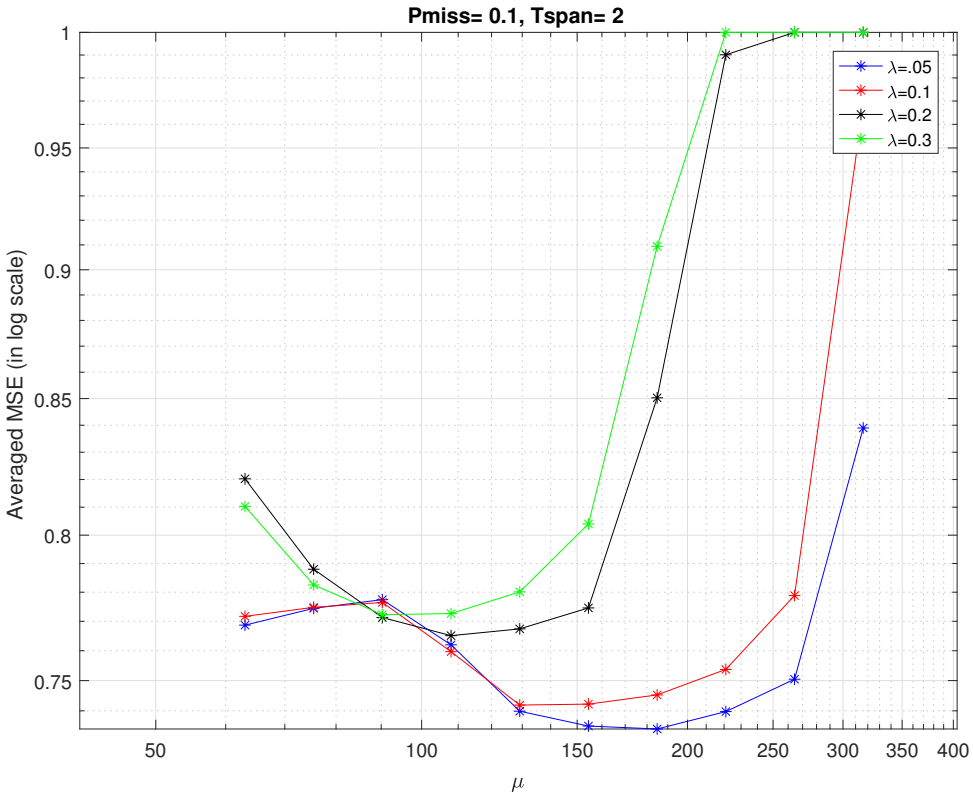

Figure 4.11: Prediction performance for the various (λ, μ) values.

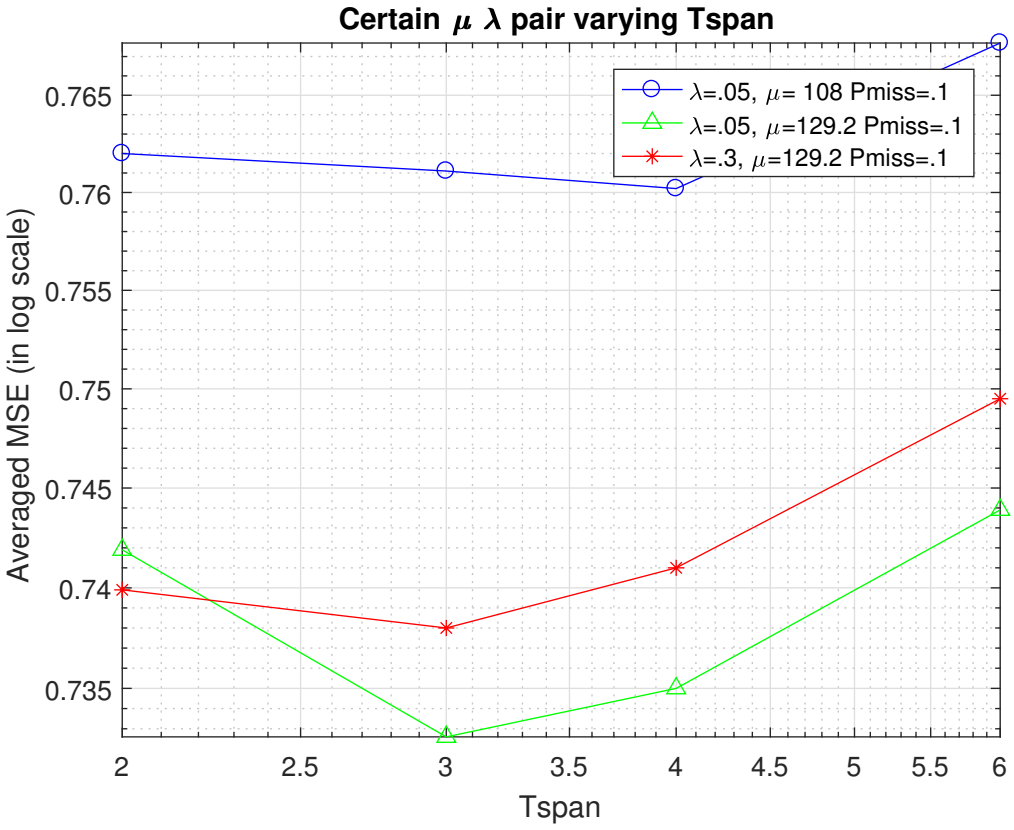

Figure 4.12: Prediction performance with P_{miss}=10%

Chapter 5: RF Signal Classification Using Deep Architectures

In this chapter, we tackle the problem of classifying the received RF signal, whether it is from a Wi-Fi transmitter, or a Bluetooth transmitter, and so forth. We will first introduce the problem and review recent methods that have been employed to tackle the problem. In Section 5.2, a deep convolutional neural network will be reviewed. Then, we will review a particular deep layered architecture called deep scattering network, which will be used for feature extraction. In Section 5.4, the classification results using two architectures will be presented and compared.

5.1 Problem Statement

An important problem in CR systems is to identify the type of received RF interference. For example, modulation recognition is vital in that CRs must classify the different schemes in order to better cope with the interference. The cyclicmoment based features were used to perform modulation recognition and form a decision tree to sort modulations into different schemes [42].

Successful algorithms using machine learning to classify images [43] and perform voice recognition [44] have given researchers the ability to create new methods that

can be applied to spectrum sensing. These new methods can learn features across a wide range of tasks and have shown to improve the classification accuracy compared to the existing methods. One idea is to used a deep layered architecture called the Convolutional Neural Network (CNN). In [45], a CNN used in the radio signal domain was studied. Using a CNN, they could learn the features on a large and densely encoded time series, which improved the performance in the low-SNR situations.

When a signal is transmitted, the signal does not always come to the receiver the same way it left the transmitter. These are called channel effects and they are not deterministic and not completely reversible. They result in scaling, temporal shifting, and complex rotation of the received signal. Using CNNs, this algorithm can learn invariant features to roughly create matched filters, which will allow it to operate in lower SNRs and help robust classification.

5.2 Deep Convolutional Neural Network

A CNN is a type of deep, feed-forward, neural network that has been used successfully in examining images. CNNs, inspired by biological process [46], require minimal preprocessing because of the fact that they use multilayer perceptrons. By using very little pre-processing, this network is able to learn the filters, required to perform classification, as opposed to pre-designed filters, which gives this method an extreme advantage. CNNs have applications in image and video recognition, recommender systems [47] and natural language processing [48].

A CNN architecture called ImageNet was proposed in [43] to perform image

classification. Using eight layers with weights, the first five are convolutional layers while the final three are fully connected layers. The first layer filters an image with 96 kernels of size 11 x 11 x 3 with a stride of 4 pixels. A stride is the distance between the receptive field centers of neighboring neurons in a kernel map. Next the output of the first layer is fed into the second layer which filters this input with 256 kernels of size 5 x 5 x 48. The next 3 layers are connected to one another without any intervening pooling or normalization layers. After the image has been passed through the 5 layers the output is then fed to a 1000-way softmax which produces a distribution over the 1000 class labels to classify the test images.

While the ImageNet architecture performed very well for NMIST image classification, recent improvements to the architecture were made by means of utilizing deeper and wider networks. One of the most recent advances comes from building off the architecture of the ImageNet to create a architecture called Inception Architecture, or Inception-v3 [49]. The authors were able to improve the ImageNet network by utilizing deeper and wider networks. In doing this, the authors wanted to utilize the added computation as efficiently as possible by suitably factorized convolutions and aggressive regularization.

The Inception-V3 architecture made four design choices to improve the performance. The first decision was to avoid representational bottlenecks [50]. They avoided this by using feed-forward networks that would be represented by an acyclic graph from the input layer(s) to the classifier. They made this decision because they note that the representation should decrease steadily before reaching the final stage to improve classification. The next principle is that the data should be in a

higher dimension to help show more disentangled features which will allow for faster training. The third principle is the idea that the data represented in a sparse manner will not result in a loss of representational power. This is because there is strong correlation between neighboring elements, which means there is smaller loss of the information. Finally, balancing the number of filters per stage and the depth of the network will increase the quality of the network.

5.3 Deep Scattering Network

Spectrograms can capture a signal's energy over frequency and time. They are heavily used in fields like music, sonar, radar, speech processing, and many others. While it can be helpful for signal classification, it is not stable against time-warping deformations that occur in many cases. This lack of stability, or the ability to force small signal deformations to result in small modification to the representation of the signal, is a major issue for the classifier design. On the other hand, being able to detect the variabilities are crucial for classifying different classes of signals.

Deep scattering networks [51] can cope with these variations by employing scattering transforms [52], which are a locally translation-invariant representation, and stable to time-warping deformation. This algorithm extends Mel-frequency cepstral coefficients (MFCC), which is widely used in audio signal processing and can capture the patterns in longer time durations, through the use of the wavelet transforms and modulus operators. MFCC is an efficient descriptor that can scale up to 25 ms. One can obtain mel-frequency spectrograms by averaging spectro-

grams over mel-frequency bands. While this improves stability to time-warping, this method removes informations, and as the averaging intervals grow over 25 ms, the information loss is too great, degrading the classification performance. Recovery of the lost information can be achieved by calculating multiple layers of wavelet coefficients, which is the idea behind the deep scattering networks.

Deep scattering spectrum remains stable (Lipschitz continuous) to deformations, while capturing high-order statistics and scale interactions. This contractive representation can be modeled mathematically by defining $\psi_\lambda(t)_{\lambda\in\Lambda}$ as a set of analytic wavelet filters, with a bandpass filter $\psi_\lambda(t)$ with a center frequency λ and a bandwidth of λ/Q. These filters have a Fourier transform of zero for negative frequencies. The wavelet transform of x(t) is then convolved with a low pass filter $\phi(t)$ and a bandwidth of 1/T to produce

$$Wx = (x * \phi(t), x * \psi_\lambda(t))_{\lambda\in\Lambda}. \tag{5.1}$$

Taking the moduli of the wavelet filter bank output, features that are locally translation-invariant and stable are gathered then passed through a low pass filter: $S_1x(t,\lambda) := |x * \psi_\lambda|\phi(t)$. These first-order scattering coefficients can be shown to be equivalent to the mel-frequency spectrograms. One can see below that by setting λ to λ_1 this is part of the wavelet transform $x * \psi_{\lambda_1}$

$$W|x * \psi_{\lambda 1}(t)| = (S_1x(t,\lambda_1), |x * \psi_{\lambda 1}(t)| * \psi_{\lambda_2}(t))_{\lambda_2\in\Lambda} \tag{5.2}$$

However, rich discriminative information may get lost as averaging occurs. One can use deep scattering transforms to retain the lost information. One can capture the co-occurrence of patterns by obtaining second-order scattering coefficients by $S_2x(t, \lambda_1, \lambda_2) := ||x * \psi_{\lambda 1}| * \psi_{\lambda 2}(t)| * \phi(t)$. To obtain the deep scattering spectrum, one must repeatedly use the modulus and wavelet transform operations. This means that the m-th order scattering coefficients can be obtained by

$$S_m x(t, \lambda_1, ..., \lambda_m) = U_m x(t, \lambda_1, ..., \lambda_m) * \phi(t) \tag{5.3}$$

where $U_m x(t, \lambda_1, ..., \lambda_m) = ||....|x * \phi_{\lambda_1}| * ....| * \phi_{\lambda_m}|$. The collection of scattering coefficients then defines the deep scattering spectrum. The figure below show a tree architecture for the scattering network. The number of layers and the order M are not equivalent and the number of layers in the equivalent deep convolutional neural network can be significantly larger.

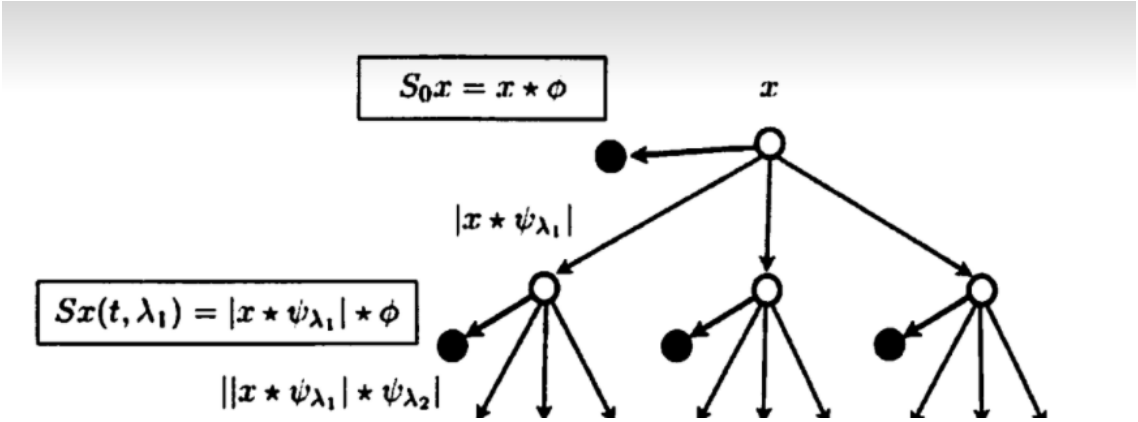

Figure 5.1: A scattering transform iterates on wavelet modulus.

5.4 Results of Numerical Experiments

The goal of the numerical experiments is to compare the RF signal classification performances using 2 different signal features, namely, the spectrogram and the deep scattering spectrum. These features are then fed to a CNN for classification. The CNN is fine-tuned to adapt to the different features. The dataset contained 9 classes of signals, each of which had 390 samples. The classes were Ambient, Bluetooth Low Energy (BLE), Bluetooth, FHSS1, FHSS2, Wi-Fi, Wi-Fi1, Wi-Fi2, and Zigbee signals. By introducing varying levels of Gaussian noise, different SNR levels were simulated. From the dataset, spectrogram and deep scattering spectrum were generated and saved as image files (see Fig 5.2 and Fig 5.3). Then, they were split into a training set, which contained 80% of the data, and the testing set, containing 20% of the data.

Using the original code found at https://github.com/tensorflow/models/tree /master/research/inception, the first step was to determine if the original parameters defined in the code needed to be changed. Also, the correct number of steps required to produce the best accuracy must be determined. Eventually it was seen that the best changes to make were to change the original dropout rate of .8 to .7 and changing the initial learning rate from .001 to .01. Using the same input and using these new parameters, the classification accuracy of 100% was achieved for the deep scattering spectrum input.

In order to see the robustness of the classifier for different SNR levels, the CNN was trained using the data at 7 different SNR levels, and tested with the test sets

of 7 different SNR levels. Thus, it was possible to have mismatch of the training SNR and test SNR. Table 5.1 shows the classification accuracy results for all 49 combinations when the spectrogram input was used. This was compared to the case where the deep scattering spectrum was used as the input to the CNN classifier. The accuracy of this case is listed in table 5.2. Figs. 5.4 and 5.5 show the accuracy curves using deep scattering spectrum and spectrogram as input respectively. It can be seen that using the deep scattering spectrum improves the accuracy of classification across SNR levels tested.

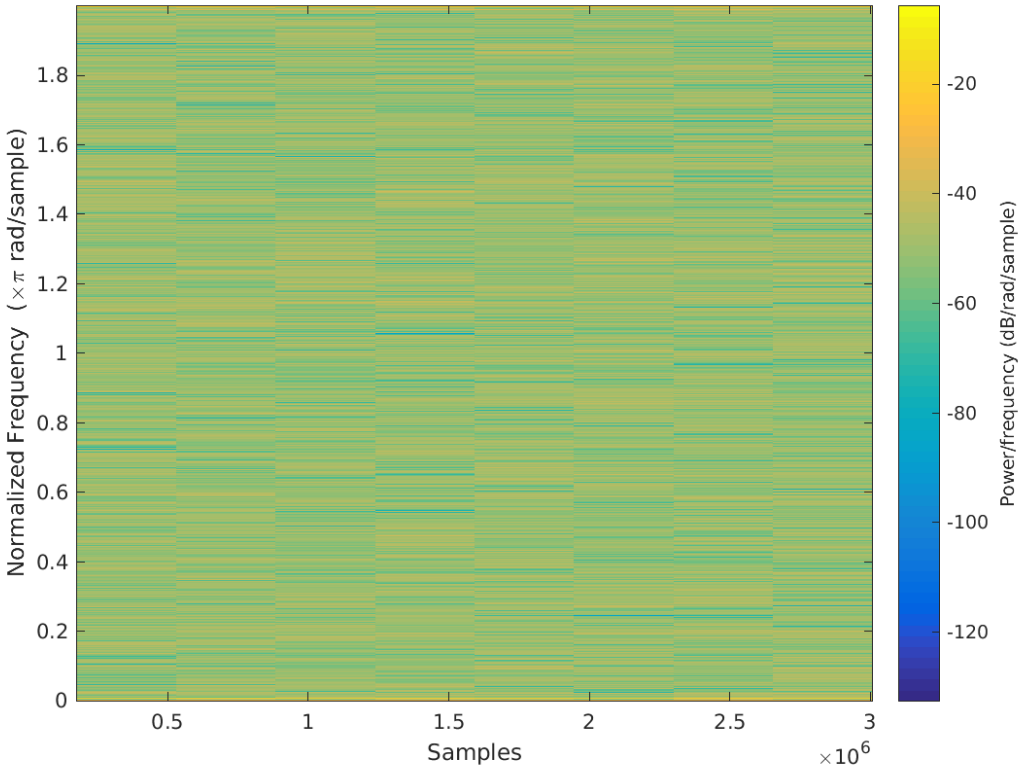

Figure 5.2: Spectrogram output.

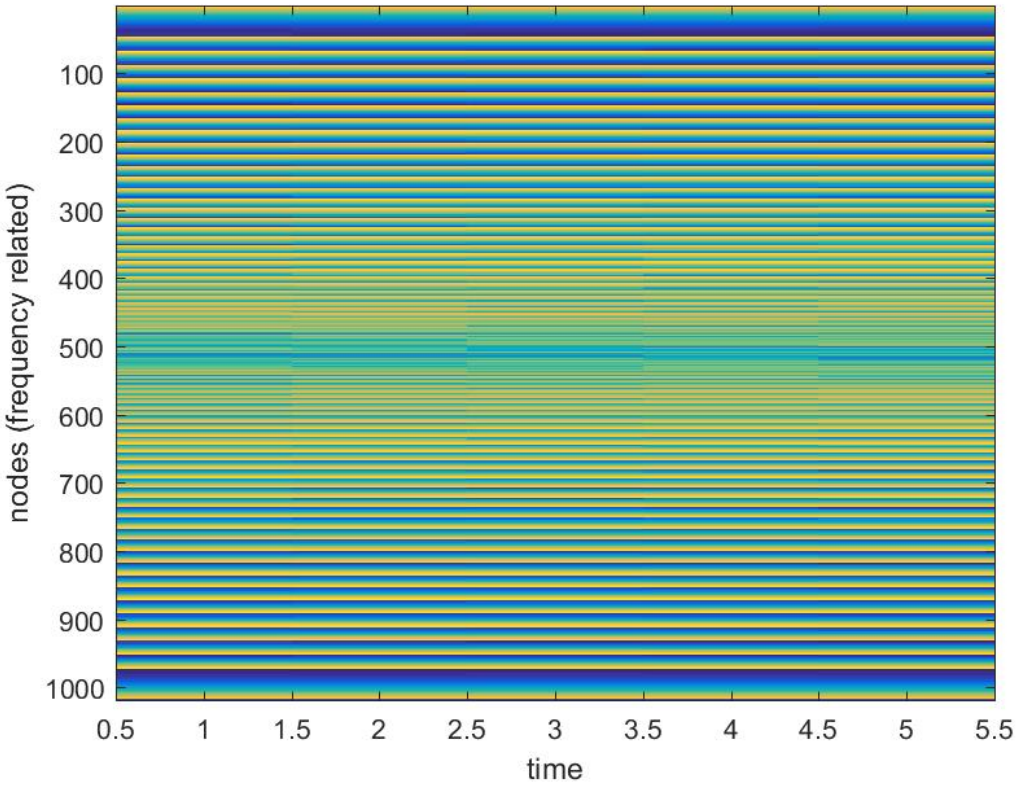

Figure 5.3: Scattering network output.

Training SNR \ Test SNR	-20	-10	-5	0	5	10	20
-20	0.5298	0.1122	0.1122	0.1122	0.1108	0.1136	0.0341
-10	0.1094	0.8722	0.7244	0.4730	0.4290	0.4077	0.2770
-5	0.1108	0.5540	0.9560	0.8693	0.7415	0.5753	0.5114
0	0.1108	0.1250	0.7188	0.9276	0.8580	0.7386	0.5909
5	0.1108	0.1193	0.4901	0.7855	0.9517	0.1179	0.7102
10	0.1108	0.1222	0.2443	0.7457	0.8523	0.9318	0.6193
20	0.1108	0.1534	0.1648	0.3125	0.5469	0.6293	0.9375

Table 5.1: Spectrogram input performance

Training SNR \ Test SNR	-20	-10	-5	0	5	10	20
-20	0.8395	0.4517	0.1534	0.1108	0.1634	0.1747	0.1108
-10	0.1236	0.9901	0.8920	0.5824	0.4205	0.4503	0.3082
-5	0.1634	0.8679	0.9957	0.9815	0.6108	0.6122	0.5611
0	0.2216	0.6776	0.8679	1.0000	0.9616	0.9247	0.7827
5	0.1619	0.2344	0.5256	0.9560	0.9730	0.9403	0.8139
10	0.1733	0.1705	0.3210	0.9517	0.8608	0.9560	0.5994
20	0.2216	0.2145	0.5099	0.7798	0.7386	0.7713	1.0000

Table 5.2: Scattering transform input performance

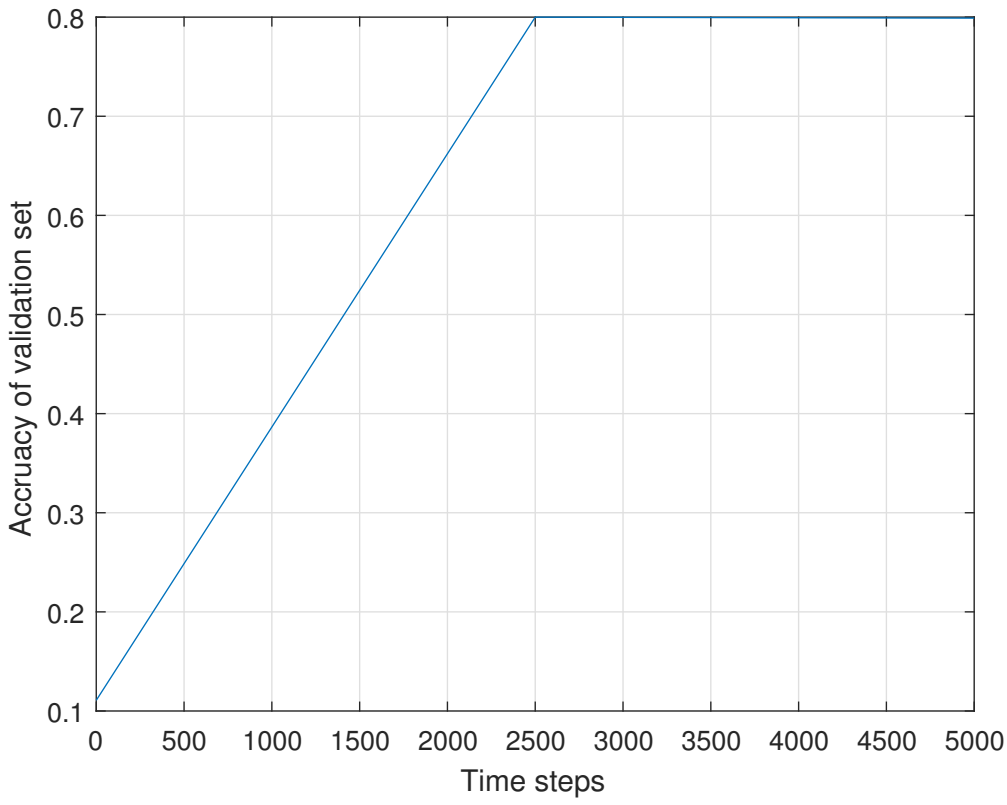

Figure 5.4: Accuracy using scattering network with traning SNR 20 dB and testing SNR 0 dB.

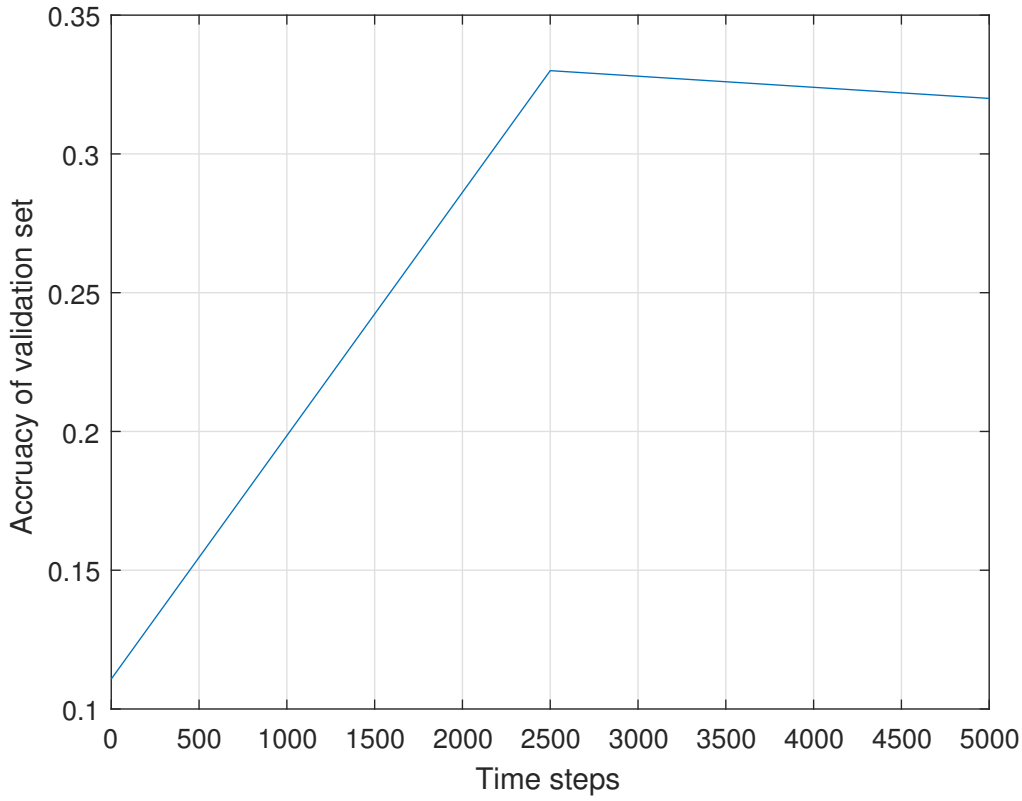

Figure 5.5: Accuracy using spectrogram with traning SNR 20 dB and testing SNR 0 dB.

Chapter 6: Conclusion

In this thesis, machine learning techniques were employed to tackle some of the challenging CR sensing problems. We used dictionary learning techniques that involve both sparsity and low-rank of coefficients to capture the salient structures in the spatio-temporal interference measurement data. Unlike the conventional dictionary learning that only capitalizes on sparsity, our scheme leveraging the low-rank property could exploit temporal correlations across different samples. Using the learned structures, the missing interference measurements could be inferred effectively. Also, by extending the formulation to temporally concatenated measurement vectors, prediction of future interference levels was also shown to be feasible. Both batch and online algorithms were derived and tested using simulated and real measurement data collected using TelosB sensors. We also used contemporary deep layered architecture to classify the type of RF interference. Using the spectrogram and more sophisticated deep scattering spectrum features, and employing state-of-the-art CNN, pre-trained for image data, the classification of 9 different RF transmitter types could be done very reliably. In particular, using deep scattering spectrum as the feature yielded significantly more robust performance to varying SNR conditions.

Appendix: Review of Alternating Direction Method of Multipliers (ADMM)

While it has been seen that using multiple CRs in a network is the one of the most viable approaches to perform spectrum sensing, setting the network just right makes all the difference in how well it can detect the transmissions. The problem with a fusion based architecture is that it is not robust and is prone to isolated point of failure. Also as the geographical area increases there is high transmitting power required to transmit and depending on where the nodes are located there is a timing issue because information collected at the same time at different nodes might not reach the center at the same time due to distance from the center. Creating an in-network where each node communicates with the other nodes whether it be directly or through other nodes has been shown to solve the problems a fusion center based network cannot handle.

Because the information collected is in a distributed manner, it is important to create an algorithm that is well suited to distributed convex optimization, can handle a large dataset in a parallelized or fully decentralized fashion, and can tackle this problem through data analysis particularly through the use of statistical and machine learning algorithms. Alternating direction method of multipliers (ADMM) fits this bill [37]. First introduced in the mid- 70's, ADMM solves smaller local sub-problems to solve the larger global problem. ADMM is seen as a combination of dual decomposition and augmented Lagrangian methods for constrained optimization.

Dual ascent, which can lead to a decentralized algorithm, considers the constrained convex optimization problem

$$minimize \quad f(x) \quad \text{subject to } Ax = b. \tag{1}$$

where x$\in \mathbf{R}^n$, A $\in \mathbf{R}^{mxn}$, and f : $\mathbf{R}^n \to \mathbf{R}$ is convex. Here the Lagrangian and dual function respectively is

$$L(x, y) = f(x) + y^T(Ax - b) \tag{2}$$

$$g(y) = \inf_x L(x, y) = -f^*(-A^T x) - b^T y \tag{3}$$

The dual problem is maximize g(y) and to recover a primal optimal point x^* from a dual optimal point y^*

$$x^* = arg\min_x L(x, y*) \tag{4}$$

In this method one solves the dual problem by using gradient ascent where this method consists of iterating the updates.

$$x^{k+1} := arg\min_x L(x, y^k) \tag{5}$$

$$y^{k+1} := y^k + \alpha^k(Ax^{k+1} - b) \tag{6}$$

Here α is a step size and this algorithm is called dual ascent because with an appropriate α^k the function increases with each step. Unfortunately the assumptions required for this method to find the optimal x and y points doesn't always hold in many applications so dual ascent can't often be used.

For the dual ascent method to be robust and yield convergence without assumptions like strict convexity of f the Augmented Lagrangian method was developed. This yields the following Augmented Lagrangian for (1) where ρ is the penalty parameter.

$$L_\rho(x, y) = f(x) + y^T(Ax - b) + (\rho/2)||Ax - b||_2^2 \tag{7}$$

The solution to this problem is then solved by minimizing x, and then evaluating the resulting equality constraint residual. While this method is similar to the dual ascent method it is changed by the fact the x-minimization step uses the augmented Lagrangian, and the penalty parameter ρ is used as the step size α^k and this method converges under far more general conditions than dual ascent.

$$x^{k+1} := arg\min_x L_\rho(x, y^k) \tag{8}$$

$$y^{k+1} := y^k + \rho(Ax^{k+1} - b) \tag{9}$$

ADMM blends the decomposability of dual ascent and has a superior convergence property of the method of multipliers. Using The equation below it will be show how ADMM is used to find the optimal solution of each local subproblem that will solve the overall global problem.

$$\min_{x,z} f(x) + g(z) \quad \text{subject to } Ax + Bz = c \tag{10}$$

This is then formed into an augmented Lagrangian and following minimization steps. In ADMM updates x and z in an alternating form as opposed to the augment Lagrangian method which jointly minimizes the two primal variables.

$$L_\rho(x, y, z) = f(x) + g(z) + y^T(Ax + Bz - c) + (\rho/2)||Ax + Bz - c||_2^2 \tag{11}$$

$$x^{k+1} := arg\min_x L_\rho(x, z^k, y^k) \tag{12}$$

$$z^{k+1} := arg\min_x L_\rho(x^{k+1}, z, y^{k+1}) \tag{13}$$

$$y^{k+1} := y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \tag{14}$$

Solving (20)-(22) then simply requires grabbing all like-terms corresponding to (20)-(22) and then solving their local problems by fixing two of the three variable

while updating the third and repeating this till all three variables have been updated. This algorithm will come to completion once all three variables converge to there local minimum solving the overall global problem.

Bibliography

[1] Federal Communications Commission, "Spectrum policy task force report." *Tech. Rep. 02-135*, 2002.

[2] M. McHenry, "NSF spectrum occupancy measurements project summary," *Shared Spectrum Co., Tech. Rep*, 2005.

[3] Q. Zhao and B. M. Sadler, "A survey of dynamic spectrum access," *IEEE Sig. Process. Mag.*, vol. 24, no. 3, pp. 79–89, May 2007.

[4] E. Axell, G. Leus, E. G. Larsson, and H. V. Poor, "Spectrum sensing for cognitive radio: State-of-the-art and recent advances," *IEEE Sig. Process. Mag.*, vol. 29, no. 3, pp. 101–116, May 2012.

[5] W. A. Gardner, "Cyclostationarity: Half a century of research," *Sig. Process.*, vol. 86, pp. 693–697, Apr. 2006.

[6] A. V. Dandawat and G. B. Giannakis, "Statistical tests for presence of cyclostationarity," *IEEE Trans. Sig. Process*, vol. 42, pp. 2355–2369.

[7] J. Lundn, V. Koivunen, A. Huttunen, and H. V. Poor, "Collaborative cyclostationary spectrum sensing for cognitive radio systems," *IEEE Trans. Sig. Process*, vol. 57, pp. 4182–4195, Nov. 2009.

[8] Z. Quan, S. Cui, A. Sayed, and H. V. Poor, "Optimal multiband joint detection for spectrum sensing in cognitive radio networks," *IEEE Trans. Sig Process*, vol. 57, no. 3, pp. 1128–1140, Mar. 2009.

[9] R. Fan, H. Jiang, Q. Guo, and Z. Zhang, "Joint optimal cooperative sensing and resource allocation in multi-channel cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 60, no. 2, pp. 722–729, Feb, 2011.

[10] Y. Wang, A. Pandharipande, Y. Polo, and G. Leus, "Distributed compressive wide-band spectrum sensing," *Proc. Information Theory and Applications Workshop*, pp. 178–183, Feb. 2009.

[11] Z. Tian, "Compressed wideband sensing in cooperative cognitive radio networks," *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, pp. 1–5, Nov.–Dec. 2008.

[12] A. Sahai, N. Hoven, and R. Tandra, "Some fundamental limits on cognitive radio," *Proc. 42nd Allerton Conf. Communication, Control, and Computing*, pp. 1662–1671, Oct. 2004.

[13] R. W. Broderson, A. Wolisz, D. Cabric, S. Mishra, and D. Willkomm, "Corvus: A cognitive radio approach for usage of virtual unlicensed spectrum," *Berkeley, CA: Univ. California Berkeley Whitepaper*, 2004.

[14] J. Unnikrishnan and V. V. Veeravalli, "Cooperative sensing for primary detection in cognitive radio," *IEEE J. Sel. Topics Sig. Proc.*, vol. 2, no. 1, pp. 18–27, Feb. 2008.

[15] C. K. Wikle and N. Cressie, "A dimension-reduced approach to space-time Kalman filtering," *Biometrika*, vol. 86, no. 4, pp. 815–829, 1999.

[16] S.-J. Kim, E. Dall'Anese, and G. B. Giannakis, "Cooperative spectrum sensing for cognitive radios using kriged Kalman filtering," *IEEE J. Sel. Topics Sig. Proc.*, vol. 5, no. 1, pp. 24–36, Feb. 2011.

[17] J.-A. Bazerque and G. B. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," *IEEE Trans. Sig. Process.*, vol. 58, no. 3, pp. 1847–1862, Mar. 2010.

[18] E. Dall'Anese, S.-J. Kim, G. B. Giannakis, and S. Pupolin, "Power control for cognitive radio networks under channel uncertainty," *IEEE Trans. Wireless Commun.*, vol. 10, no. 10, pp. 3541–3551, Oct. 2011.

[19] I. Tošić and P. Frossard, "Dictionary learning," *IEEE Sig. Process. Mag.*, vol. 28, no. 2, pp. 27–38, Mar. 2011.

[20] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 35, no. 1, pp. 171–184, Jan. 2013.

[21] Y. Zhang, Z. Jiang, and L. S. Davis, "Learning structured low-rank representations for image classification," in *Proc. of the IEEE Conf. Comput. Vis. Pattern Recognition*, Oct. 2013, pp. 676–683.

[22] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Sig. Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[23] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *J. Mach. Learn. Res.*, vol. 11, pp. 19–60, 2010.

[24] Z. Zhang, Y. X. J. Yang, X. Li, and D. Zhang., "A survey of sparse representation: Algorithms and applications," *arXiv:1602.07017v1*, 2016.

[25] S. Mallat, *A wavelet tour of signal processing: The sparse way*, 2008.

[26] X. Lu and X. Li, "Group sparse reconstruction for image segmentation," *Neurocomputing*, vol. 136, pp. 41–48, 2014.

[27] M. Elad, M. Figueiredo, and Y. Ma, "On the role of sparse and redundant representations in image processing," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 972–982, 2010.

[28] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031–1044, 2010.

[29] J. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inform. Theory*, vol. 50, no. 10, pp. 2231–2242, Sept. 2004.

[30] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. R. Stat. Soc. Ser. B (Method.)*, vol. 58, no. 1, 1996.

[31] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1," *Vis. Res.*, vol. 37, no. 23, 1997.

[32] P. Schmid-Saugeon and A. Zakhor, "Dictionary design for matching pursuit and application to motion-compensated video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 6, 2004.

[33] E. Cande's, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis," *J. ACM*, 2009.

[34] K. V. Mardia, C. Goodall, E. J. Redfern, and F. J. Alonso, "The kriged kalman filter," *Test*, vol. 7, no. 2, pp. 217–285, 1998.

[35] S.-J. Kim, N. Jain, G. B. Giannakis, and P. Forero, "Joint link learning and cognitive radio sensing," in *Proc. Asilomar Conf. Sig., Syst., Comput.*, Pacific Grove, CA, Nov. 2011, pp. 1415–1419.

[36] R. Vidal and P. Favaro, "Low rank subspace clustering (LRSC)," *Pattern Recognition Letters*, vol. 43, pp. 47–61, Jul. 2014.

[37] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[38] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least-angle regression," *Ann. Stat.*, vol. 32, no. 2, pp. 407–499, 2004.

[39] P. Tseng, "Convergence of block coordinate descent method for nondifferentiable minimization," *Journal on Optimization Theory and Applications*, vol. 109, pp. 475–494, Jun. 2001.

[40] J.-F. Cai, E. J. Candes, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, pp. 1956–1982, Mar. 2010.

[41] Texas Instruments, "2.4 GHz IEEE 802.15.4 ZigBee-ready RF transceiver."

[42] W. A. Gardner and C. M. Spooner, "Signal interception: performance advantages of cyclic-feature detectors," *IEEE Trans., Commun.*, vol. 40, no. 1, pp. 149–159, 1992.

[43] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems 25*, 2014.

[44] T. N. Sainath and et al, "Learning the speech front-end with raw waveform cldnns," *Proc. Interspeech*, 2015.

[45] T. OShea, J.Corgan, and T. Clancy, "Convolutional radio modulation recognition networks," *arXiv:1602.04105v3*, Jun. 2016.

[46] M. Matsugu, K. Mori, Y. Mitari, and Y. Kandeda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Networks.*, vol. 16, no. 5, pp. 555–559, June 2003.

[47] A. van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," pp. 2643–2651, 2013.

[48] R. Collobert and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," pp. 160–167, July 2008.

[49] C. Szegedy, V. Vanhoucke, S. Ioffe, and J. Shlens, "Rethinking the inception architecture for computer vision," *arXiv:1512.00567v3 [cs.CV]*, 2015.

[50] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Proc. the 32nd International Conference on Machine Learning*, pp. 448–456, Feb. 2015.

[51] J. Anden and S. Mallat, "Deep scattering spectrum," *IEEE Trans. on Sig. Process.*, vol. 62, 2014.

[52] S. Mallat, "Group invariant scattering," *Commun. Pure Appl. Math.*, vol. 65, no. 10, pp. 1331–1398, 2012.