Exploring Latent Structure in Data: Algorithms and Implementations

by

Prasanna S. Sattigeri

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Doctor of Philosophy

Approved November 2014 by the
Graduate Supervisory Committee:

ARIZONA STATE UNIVERSITY

December 2014

ABSTRACT

Feature representations for raw data is one of the most important component in a machine learning system. Traditionally, features are *hand crafted* by domain experts which can often be a time consuming process. Furthermore, they do not generalize well to unseen data and novel tasks. Recently, there have been many efforts to generate data-driven representations using clustering and sparse models. This dissertation focuses on building data-driven unsupervised models for analyzing raw data and developing efficient feature representations.

Simultaneous segmentation and feature extraction approaches for silicon-pores sensor data are considered. Aggregating data into a matrix and performing low rank and sparse matrix decompositions with additional smoothness constraints are proposed to solve this problem. Comparison of several variants of the approaches and results for signal de-noising and translocation/trapping event extraction are presented. Algorithms to improve transform-domain features for ion-channel time-series signals based on matrix completion are presented. The improved features achieve better performance in classification tasks and in reducing the false alarm rates when applied to analyte detection.

Developing representations for multimedia is an important and challenging problem with applications ranging from scene recognition, multi-media retrieval and personal life-logging systems to field robot navigation. In this dissertation, we present a new framework for feature extraction for challenging natural environment sounds. Proposed features outperform traditional spectral features on challenging environmental sound datasets. Several algorithms are proposed that perform supervised tasks such as recognition and tag annotation. Ensemble methods are proposed to improve the tag annotation process.

To facilitate the use of large datasets, fast implementations are developed for sparse coding, the key component in our algorithms. Several strategies to speed-up Orthogonal Matching Pursuit algorithm using CUDA kernel on a GPU are proposed. Implementations are also developed for a large scale image retrieval system. Image-based exact search and visually similar search using the image patch sparse codes are performed. Results demonstrate large speed-up over CPU implementations and good retrieval performance is also achieved.

To my family and friends ...

ACKNOWLEDGEMENTS

This dissertation would not have seen the light of the day without the unending support and guidance from my mentors, family and friends.

I am grateful to my advisor Dr. Andreas Spanias for providing inspiration and assistance throughout my PhD. Along with excellent guidance he provided me a great atmosphere for pursuing my research interests. I am also thankful to all the members of my thesis committee Dr. Trevor Thornton, Dr. Michael Goryll and Dr. Konstantinos Tsakalis for providing inputs and their valuable feedback.

I would like to extend my sincere thanks to Jayaraman Thiagarajan and Karthikeyan Ramamurthy. I have had several successful collaborations with them. Our collaborations have taught me a great deal. My sincere thanks to all the other collaborators I have worked with. I would like to express my deep sense of gratitude towards Dr. Jieping Ye for teaching the wonderful Machine Learning course at ASU. I would like to thank Dr. Gil Speyer for inspiring through his teaching at ASU and his guidance during my internship.

I would also like to thank Sherin Muckatira for always cheering me up and providing me unending support. The study sessions in Nobel library and Biodesign were a great source of motivation and certainly accelerated my progress.

I would like to extend my gratitude to Mohit Shah for being a great friend, roommate, lab-mate and always ready to discuss my ideas. Special thanks to my friends Ramesh Thulisraman, Hitesh Khunti, Vijay Sarvepalli, and Pushkar Kulkarni for their unconditional support. This acknowledgement would not be complete without mentioning Dutch Bros and Altitude Coffee Lab for serving wonderful coffee and an atmosphere to recharge my batteries.

Finally, I would like to thank my parents and my brother for always wishing me well and encouraging me in my endeavours.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

1.1 Importance of Features in ML Systems

A typical machine learning (ML) system takes low level sensor data as input and produces decisions ranging from label prediction to estimation of continuous value variables. Internally, this is a two step process, firstly, the low-level sensor data are transformed into efficient representations called features, and secondly, the built features are passed through a prediction algorithm to produce decisions. Prediction algorithms have been a subject of great interest to the research community and several efficient algorithms with strong theoretical guarantees have been developed [7]. These algorithms can be broadly described as classifiers for predicting categorical variables and regression functions for continuous value prediction.

Linear classifiers are one of the earliest pattern recognition algorithm. Input data $\mathbf{x}$ is first transformed to $\mathbf{f} = \Phi(\mathbf{x})$. This process is called feature extraction and $\mathbf{f}$ is called feature representation of the data $\mathbf{x}$. The sign of the linear discriminant function $y(\mathbf{x}) = \mathbf{w}^T\mathbf{f} + b$ is used as the prediction for the class. The learning process involves finding $\mathbf{w}$ and b which are optimal for the training data and also have the generalization power to perform well on unseen test data. These parameters define the optimal hyperplane which divides the classes. Support Vector Machines (SVM) with their maximum margin properties and the ability to use kernel tricks have enjoyed great success and have become the de-facto choice for a classification algorithm. Support Vector Regression (SVR) are their continuous value prediction counterparts.

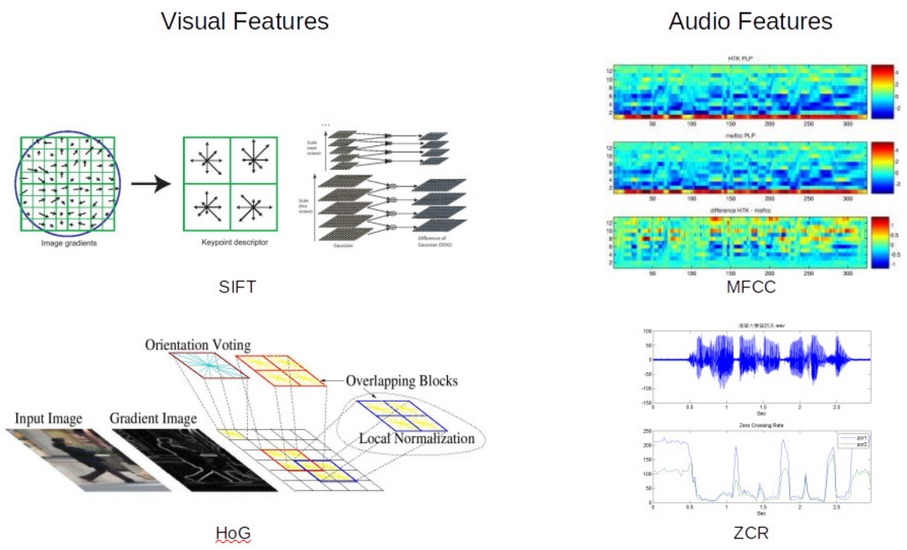


Figure 1.1: A Few Examples Showing Hand Engineered Features for Image and Audio [2].

1.2 Learning Feature Transformation Functions

The performance of an ML system often hinges on the quality of the feature representations. Thus, most efforts are usually directed towards this step. The transformation function $\Phi(\mathbf{x})$ to obtain the features are usually designed using years of domain-specific knowledge. This process is expensive and time consuming. Another, downside of *hand-crafted* feature transformations are that they are not scalable, while we know that, more the data we can utilize for training, we can do better at our task. Figure 1.1 and 1.2 show the different sets of feature fund to work well for images and audio. These are highly task specific and do not generalize well to unseen data and thus are not reusable in novel but related tasks.

In light of these problems with hand-designed features, the focus is shifting to data-driven approaches to learn the transformation function $\Phi(.)$ directly from the data. This has been made possible by availability of large scale databases and powerful hardware to process them. Features learned from large-scale data can effectively capture the multitudes of variation in the data and can provide large improvements

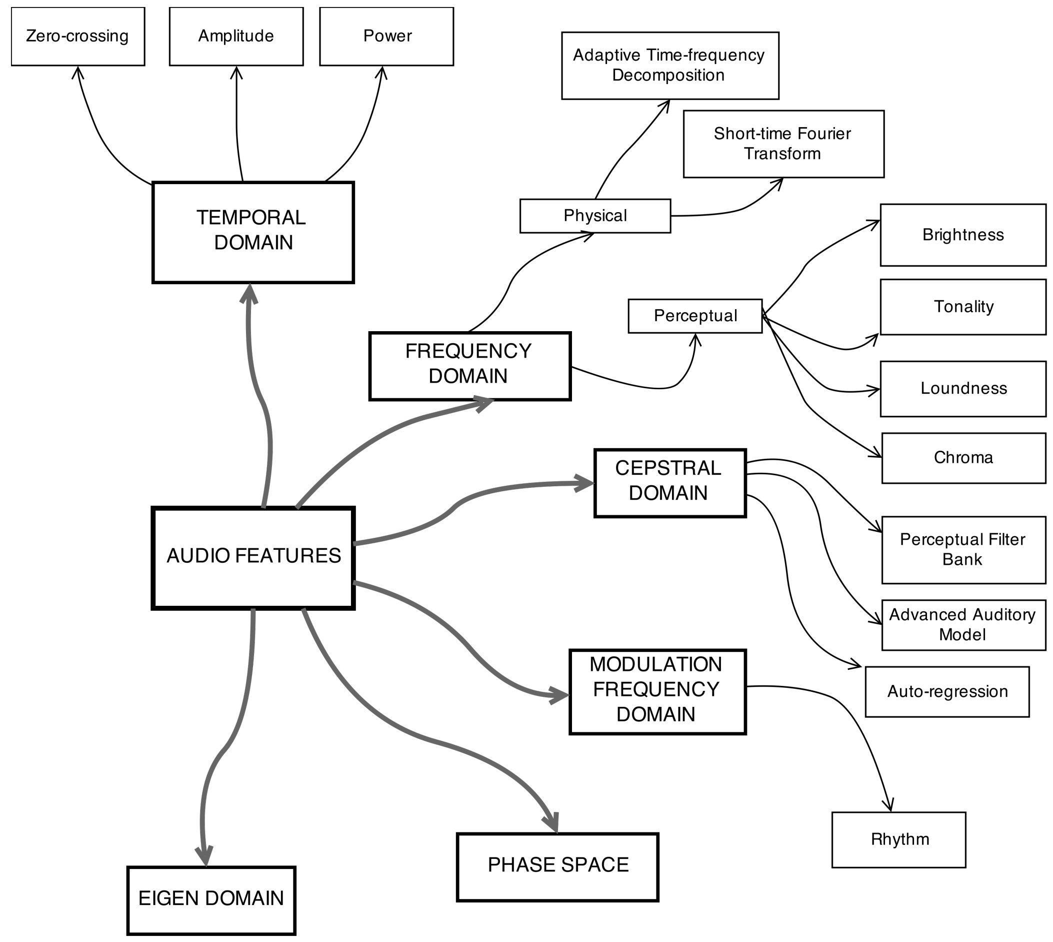


Figure 1.2: Categorization of the Hand-Designed Features for Audio [3].

in performance. These approaches have lead to breakthroughs in speech and vision domain and provide state of the art in several object detection tasks [8]. Winners in competitions such as ImageNet Large Scale Visual Recognition Challenge and several Kaggle competitions have used feature learning approaches. They are now part of many commercial systems and are deployed in Speech recognition and image analysis systems used by millions of people on daily basis.

In this dissertation, the focus will be on strategies to handle prior information to learn efficient features representations. Latent structure is explored using sparsity and low rank constraints. Deep architecture are proposed to exploit the hierarchical structure in data. Efficient algorithm designs to use parallelization and to tackle hardware/ software limitations are proposed.

1.3 Problem Statement

1.3.1 Improving Transform Domain Features for Ion-Channel Signals

Ion-channel sensors which mimic naturally occurring pore-forming proteins can be used to detect small metal ions and organic molecules. A chamber with a lipid bilayer hosting ion-channels produced by protein insertion constitutes such a sensor. Each analyte produces a characteristic signal pattern during its migration from one section of the chamber to another through the ion-channels. We propose the use of power distribution information in the transform domain as discriminatory features for the signal. Several subspace based algorithms are proposed to improve the discriminative power of the features.

1.3.2 Representations for Environment Sound Analysis

The growing interest in wearable computing, automatic life logging, and predictive inferences in robotics presents a huge potential for algorithms that characterize environmental sounds. A commonly adopted pipeline for processing such data involves extracting features to succinctly describe them, modeling the statistics of the features, and deriving predictors that reveal the underlying semantics. The quality of the extracted features is intimately tied to the subsequent stages for obtaining inferences. In this dissertation, we propose a new approach for extracting hierarchical/deep features that can be very effective in characterizing natural environmental sounds. Furthermore, we present an approach based on sparse representations to predict tags for novel test samples using the proposed features.

1.3.3 Feature Learning at Scale

Sparse coding is a key component in feature learning but computationally expensive method. Processing data segments in parallel can result in tremendous speed-ups. Performing fine-grain parallelization of the algorithms can also reduce computation times. The inexpensive Graphical Processing Units (GPUs) are a popular option for parallel processing. We propose GPU friendly strategies to save memory and to achieve maximum speed-up and show their performance in large-scale sparse-coding and image-retrieval tasks.

1.3.4 Subspace Based Methods for Silicon-Pore Data

Silicon pores with diameters in the range of micro/nano-meters can be used to detect a range of analytes. Silica beads are used as carriers of biomolecules through the pores. Passage of beads through the pores are termed as translocation events. In the presence of certain pairs of biomolecules, the pores exhibit trapping behaviour where the pores gets partially blocked. Such behaviour is termed as a trapping event. In this dissertation, procedures for simultaneously de-noising the signal and extracting the sparse translocation/trapping events using subspace methods have been proposed.

1.4 Contributions

In Chapter 3, building and improving transform domain features for ion-channel time-series signals are considered. Power distribution features, extracted from the frequency/sequency domains, are proposed that can effectively discriminate different ion-channel signals. Improvement of power spectral density (PSD) features for ion-channel signals is posed and solved as matrix completion problem. Improved features achieve better performance in classification and in reducing the false alarm rates when applied to analyte detection. Algorithms are proposed to decompose the features into low-rank and sparse components to capture the group behaviour and also give importance to intra-group variation. The work discussed in this chapter has been reported in [9] [10].

In Chapter 4, deep learning based models are considered to learn richer representations for challenging tasks in environment sound analysis. A new framework is developed for feature extraction and obtaining semantic inferences from such data. In particular, a new pooling strategy for deep architectures is proposed, that can effectively preserve the temporal dynamics in the resulting representation. Furthermore, algorithms are presented to perform partitioning of the feature space based on the temporal dynamics of the corresponding filters. In addition to making filter learning computationally tractable in the subsequent layers, the partitioning provides a multi-view representation of the data. By constructing an ensemble of semantic embeddings using the multiple partitions, an $l1$-reconstruction based prediction algorithm for estimating the relevant tags is proposed. Proposed features outperform traditional spectral features on challenging environmental sound recognition datasets. Architecture, algorithms and results provided in this chapter have been published in [11].

In Chapter 5, scalability issues with sparse coding as a feature learning algorithm are considered. Several strategies to speed-up Orthogonal Matching Pursuit algorithm using CUDA kernel on a GPU are proposed. Implementations are developed as part of a large scale image retrieval system. Image-based exact search and visually similar search using the image patch sparse codes are performed. Results demonstrate large speed-up over CPU implementations and good retrieval performance is also achieved. The work discussed in this chapter has been reported in [12].

In Chapter 6, simultaneous segmentation and feature extraction approaches for silicon-pores sensor data are considered. This problem is posed as a low rank and sparse matrix decomposition in the presence of dense noise with additional smoothness constraints on the low rank and sparse components. Several variants of RPCA and GoDec suitable for this data are proposed. Comparison of both approaches and results for signal de-noising and translocation/trapping event extraction are presented. Analysis of the shape and duration of these events enables us to estimate the properties of analytes. The proposed approach uses spectral clustering to cluster the events in the feature domain as the shape of the clusters is unknown. Spectral clustering accurately finds the natural clusters and the outlying cluster. The outlying cluster is further clustered to find the categories of the events within it. Methods and results discussed in this chapter has been reported in [13].

Chapter 2

BACKGROUND

2.1 Transform Domain Features

Transform-domain approaches suited for characterizing signals have been developed. In this approach, the signals are transformed to new co-ordinate system using hand-designed basis functions, with the hope that the data dimensions are uncorrelated in the new space. Feature vectors are then generated either by picking largest coefficients from the transformed representation of the signals or alternatively various statistical operations can be performed to generate the feature representations. As a generative process, the data is assumed to be obtained as the linear combination of the fixed basis functions or vectors. For unitary transformations, the forward operation is given as

$$\mathbf{a} = \mathbf{\Psi}^{*}\mathbf{x} \tag{2.1}$$

where the columns of $\mathbf{\Psi}$ are the basis vectors. $\mathbf{x}$ is the data vector and $\mathbf{a}$ is the representation of the data in the transformed domain. The inverse transform reduces to

$$\mathbf{x} = (\mathbf{\Psi}^{*})^{-1}\mathbf{a} = \mathbf{\Psi}\mathbf{a} \tag{2.2}$$

after using the property of the unitary matrices. These transformations preserve the vector length or energies in the new space and can be understood as rotation of the co-ordinate system around the origin with possible sign flips.

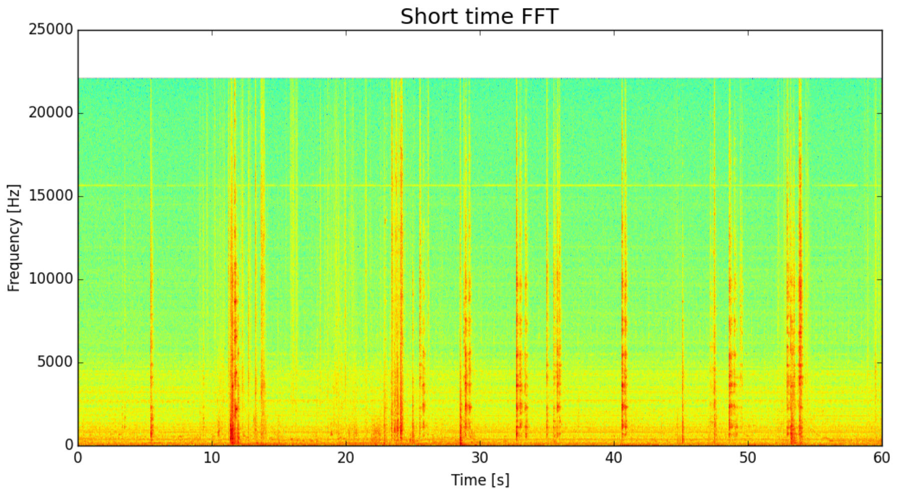


Figure 2.1: Short-time Fourier Transform. The Plot Shows the Variation in Spectral Content Over Time of an Audio Clip Recorded in a Kitchen Environment.

The Discrete Fourier Transform (DFT) is one of the most well known fixed basis transformation method. In this transform, the basis are set as complex exponentials with $\Psi_{k,n} = \mathrm{e}^{j2\pi kn/N}$. The forward and inverse transforms are given as below.

$$\mathbf{a}[k] = \sum_{n=0}^{N-1} \mathbf{x}[n]\mathrm{e}^{-j2\pi kn/N} \quad , \quad 0 \leq k \leq N-1 \tag{2.3}$$

$$\mathbf{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1} \mathbf{a}[k]\mathrm{e}^{j2\pi kn/N} \quad , \quad 0 \leq n \leq N-1 \tag{2.4}$$

$$\mathbf{x}[n] = p_0 + \sum_{k=0}^{N/2-1} p[k]cos\left(\frac{2\pi kt[n]}{Ndt}\right) + q[k]sin\left(\frac{2\pi kt[n]}{Ndt}\right) \quad , \quad 0 \leq n \leq N-1 \tag{2.5}$$

$$\begin{aligned} p_0 &= \mathbf{a}[0]/N \\ p[k] &= 2 \cdot real(\mathbf{a}[k+1])/N \\ q[k] &= -imag(\mathbf{a}[k+1])/N \end{aligned} \tag{2.6}$$

The Discrete Cosine Transform (DCT) uses cosines and real numbers only to reconstruct the signals and is valid only for real signals. This representation leads to better compression compared to DFT. The coefficients for DFT are computed as

$$\mathbf{a}[k] = \sum_{n=1}^{N-1} \mathbf{x}[n] \, cos\left(\frac{\pi}{N}(n+\frac{1}{2})k\right) \tag{2.7}$$

DCT and DFT use infinite length sinusoids as basis and lack localization property. This short-coming is highlighted when the signal is non-stationary in nature. One of the remedy is to perform DFT over short windows. Figure 2.1 shows the energy of the DFT transform when computed over small windows. The Discrete Wavelet Transform (DWT) is a linear transformation that can be used to analyze the temporal and spectral properties of non-stationary signals. The DWT of a sequence $\mathbf{x}[n]$ is defined by the following equation:

$$\mathbf{a}[k] = \sum_{j} \sum_{k} \mathbf{x}[n] 2^{-j/2} \psi(2^{-j}n - k) \tag{2.8}$$

where $\psi(.)$ is the transforming function called the mother wavelet. DWT decomposes the signal into coarse and fine information coefficients providing a multi-resolution representation. DWT is computed by successively applying pairs of analysis filters to the input signal. The advantage of the wavelet representation is that it can provide both time and frequency parameters for specific dynamic signal events, i.e. time-frequency localization [14]. In contrast, the Fourier transform based filtering methods assume that the signal is stationary and thus cannot provide any information on the variations in the spectrum with respect to time.

The distribution of power over frequency bins for Fourier and Wavelet transforms contain key statistical characteristics of the signals, although the coefficients of the representations themselves are random. A similar argument can be applied to the distribution of power over sequency bins for the Walsh-Hadamard transforms as well. This is because the Walsh functions are similar to the Haar wavelet, albeit lacking the localization property. Consider a signal represented by $\mathbf{x}$. The problem is to extract relevant features from the Fourier, Wavelet and Walsh-Hadamard domain representations of $\mathbf{x}$, that are sufficiently close to each other for signals within a single class. In addition, the features across multiple classes should have sufficient discriminatory information. We can use the distribution of power over a number of pre-specified bins in the frequency or sequency domain as the feature vector. The power in each bin specifies the relative dominance of the frequencies in the bin and is a robust statistical representative of the signal than the coefficients of the representation, which are themselves random in nature.

2.1.1 Fourier Domain Features

The Fourier domain features are computed by taking the PSD of the signal $\mathbf{x}$ using the Welch procedure given in [14]. Estimating the PSD involves windowing of the signal and averaging the modified periodograms computed over the windows to reduce the variance of the estimate. We denote f_s as the sampling frequency and f_c as the frequency where the flat and sloping portions of the PSD intersect.

The feature to be used is derived from the PSD. The DC value of the PSD is neglected and the PSD is divided into bins spaced in powers of two. The PSD values in each bin are summed and finally normalized by the total signal power, which results in the feature vector. The lower frequencies are thus represented by more points in the feature vector and the higher frequencies are represented by lesser number of points. This is expected to better capture the signal power concentrated in the lower frequencies. Each bin represents the frequency range from $f_s/2^{l+1}$ to $f_s/2^l$ and the center frequency of the bin is given by $3f_s/2^{l+2}$, where $l = \{1, ..., L\}$ is the index of the bin

2.1.2 Wavelet Domain Features

For computing the wavelet domain features we use the PDF described in [15]. We use the Haar wavelet, as it has a shape that correlates well with the general switching state structure of the ion-channel signal. The signal $\mathbf{x}$ is divided into M non-overlapping windows, and the PDF is averaged across the windows to reduce the variance, similar to the case of estimating PSD. Denoting $c_{l,k,m}$ as the k^{th} wavelet coefficient in scale l for the m^{th} frame, the average PDF at scale l is

$$\mathbf{x}^W(l) = \frac{\sum_{m=1}^{M} \sum_{k=1}^{K_l} c_{l,k,m}^2}{\|\mathbf{x} - \overline{\mathbf{x}}\|_2^2}, \tag{2.9}$$

where K_l is the number of coefficients at scale l, $\overline{\mathbf{x}}$ is the vector containing the average of the elements in $\mathbf{x}$.

As in the case of PDF, the center frequency of the scale l is given by $3f_s/2^{l+2}$ and this feature also gives more importance to the lower frequencies (coarser scales) of the wavelet decomposition, which predominantly captures the signal characteristics.

2.1.3 Walsh-Hadamard Domain Features

The Walsh-Hadamard transform uses the Walsh basis functions to decompose a signal. The representation does not have an interpretation in terms of frequency, but the variation in the basis function is defined in terms of sequency. Similar to the case of Fourier and Wavelets, the the signal $\mathbf{x}$ is divided into M non-overlapping windows. The Walsh-Hadamard transform is computed for each windowed signal and the power distribution is calculated over L linear bins of sequencies. The average power distribution is calculated by averaging the power distributions of the M windowed signals and they are the representative features for the signal.

As an example, these transform domain methods can be used to develop representations for ion-channels. Ion-channels are the pores across engineered membranes and

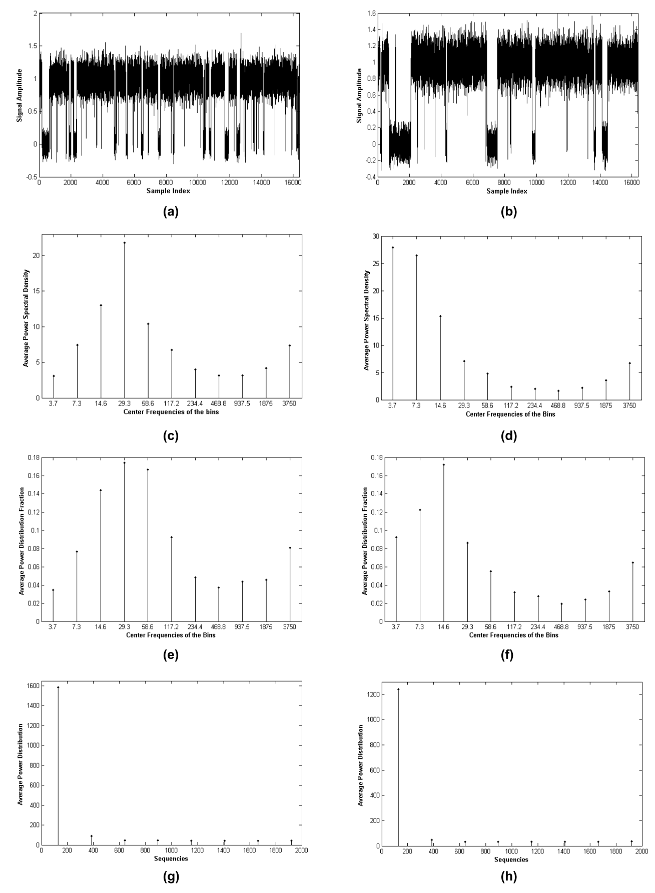


Figure 2.2: Ion-channel Signal From (a) Analyte 1 (b) Analyte 2. (c) and (d) - Corresponding Fourier Domain Features. (e) and (f) - Wavelet Domain Features. (g) and (h) - Walsh-Hadamard Domain Features.

the opening and closing of pores can be characterized by stochastic process [16]. In the simplest situation, the ion-channels produce a fluctuating current characterized by binary states and the identity of the analyte can be inferred from the magnitude and duration of the fluctuations. The frequency of the fluctuations often reveal the concentration of the analytes [17]. Simulated ion-channel signals for two different analytes are shown in Figures 2.2 (a) and (b) and their corresponding Fourier domain feature vectors are shown in Figures 2.2 (c) and (d) respectively. Figures 2.2 (e) and (f) show the feature vectors from the wavelet representation corresponding to the signals in Figures 2.2 (a) and (b).Figures 2.2 (g) and (h) show the features derived from the Walsh-Hadamard representation for the ion-channel signals shown in Figures 2.2 (a) and (b). The Walsh functions are similar to the Haar wavelet except that they lack the time localization properties of Haar wavelet. Hence, this feature also captures the key statistical properties of the ion-channel signals though it could be slightly inferior to the wavelet domain features.

Under zero-noise conditions, a state-switching signal is the realization of a continuous time Markov random process, with states denoted by $i \in S$, where S is the state space. The continuous time random process at any instant τ is denoted by the random variable $\tilde{x}_\tau$ which can take the values $g(i)$, where $i \in S$ and $g(.)$ is an invertible map. The continuous time Markov process is defined by the rate transition matrix $\mathbf{Q}$, whose rows sum to 0 [18]. The total number of states is given by $|S|$.

Sampling the process at time intervals Δ_τ gives rise to a discrete time Markov process denoted by the random variable $\tilde{x}_t$ such that $\tau = \Delta_\tau t$. The state transition matrix of the discrete time Markov process is obtained as $\mathbf{A} = \exp(\mathbf{Q}\Delta_\tau)$. This implies that $\mathbf{A} = \sum_{k=1}^{|S|} \mathbf{A}_k \exp(\lambda_k \Delta_\tau)$, where $\mathbf{A}_k$ is the product of the k^{th} right and left eigenvectors of $\mathbf{Q}$ [18]. It can be observed that the eigenvectors of $\mathbf{A}$ are the same as the eigenvectors of $\mathbf{Q}$ and the eigenvalues of $\mathbf{A}$ are always positive. $\mathbf{A}$ always has 1 as the maximum eigenvalue. The state transition probabilities of the Markov chain are given by, $p_{ij} = Pr(\tilde{x}_{t+1} = g(j)|\tilde{x}_t = g(i))$, where $i \in S$ and $j \in S$. p_{ij} is the $(i,j)^{\text{th}}$ element of $\mathbf{A}$. The stationary distribution of the Markov chain is given by $\pi_i = Pr(\tilde{x} = g(i))$. The mean of the random process denoted by $\tilde{x}_t$ is $\mu = E[\tilde{x}_t] = \sum_{i \in S} \pi_i g(i)$ and the autocorrelation function is given by,

$$R(n) = E[\tilde{x}_t \tilde{x}_{t+n}] = \sum_{i \in S} \sum_{j \in S} g(i) g(j) p_{ij}^{n,0}, \tag{2.10}$$

where $p_{ij}^{n,0} = Pr(\tilde{x}_{t+n} = g(j)|\tilde{x}_t = g(i))$.

The one-sided Power Spectral Density (PSD) of the Markov chain can be computed using the z-transform as $F^+(z) = \sum_{n=0}^{\infty} R(n)z^{-n}$ [19]. Therefore,

$$F^+(z) = \sum_{n=0}^{\infty}\sum_{i\in S}\sum_{j\in S} g(i)g(j)p_{ij}^{n,0}z^{-n},$$

$$= \sum_{i\in S}\sum_{j\in S} g(i)g(j)\pi_i P_{ij}^+(z), \tag{2.11}$$

where $z = re^{j\omega}$, $p_{ij}^{n|0} = Pr(\tilde{x}_{t+n} = j|\tilde{x}_t = i)$ and $P_{ij}^+(z) = \sum_{n=0}^{\infty} p_{ij}^{n|0}z^{-n}$. Using the Chapman-Kolmogorov equation,

$$p_{ij}^{n+1|0} = \sum_{r\in S} p_{ir}p_{rj}^{n|0}, \tag{2.12}$$

$$P_{ij}^+(z) = z^{-1}\sum_{r\in S} p_{ir}P_{rj}^+(z) + p_{ij}^{0|0}, \tag{2.13}$$

where $p_{ij}^{0|0} = 1$ when $i = j$, $p_{ij}^{0|0} = 0$ when $i \neq j$. Define $\mathbf{P}^+(z)$ as the z-transform matrix with the (i, j)th entry being $P_{ij}^+(z)$. Now $\mathbf{P}^+(z) = z^{-1}\mathbf{A}\mathbf{P}^+(z) + \mathbf{I}$, and hence $\mathbf{P}^+(z) = (\mathbf{I} - z^{-1}\mathbf{A})^{-1}$. Therefore, from (2.11), $F^+(z) = \mathbf{s}^T\mathbf{P}_\pi\mathbf{P}^+(z)\mathbf{s}$, where $\mathbf{s}$ is a vector with elements $g(i)$ and $\mathbf{P}_\pi$ is a diagonal matrix with π_i as the diagonal elements. The two sided PSD can be obtained as $F(z) = F^+(z) + F^+(z^{-1}) - F^+(\infty)$. Since $R(0) = F^+(\infty)$,

$$F(z) = \mathbf{s}^T\mathbf{P}_\pi\mathbf{P}^+(z)\mathbf{s} + \mathbf{s}^T\mathbf{P}_\pi\mathbf{P}^+(z^{-1})\mathbf{s} - \mathbf{s}^T\mathbf{P}_\pi\mathbf{s},$$

$$= \mathbf{s}^T\mathbf{P}_\pi\left(\mathbf{I} - \mathbf{A}^2\right)\left(\mathbf{I} - 2r\mathbf{A}\cos\omega + \mathbf{A}^2\right)\mathbf{s}. \tag{2.14}$$

Indicating the eigen decomposition of $\mathbf{A}$ as $\mathbf{U\Gamma U}^{-1}$, such that the eigenvalues are $\{\gamma_k\}_{k=1}^{|S|}$, we obtain,

$$F(z) = \mathbf{s}^T\mathbf{P}_\pi\mathbf{U}(\mathbf{I} - \mathbf{\Gamma}^2)(z\mathbf{I} - \mathbf{\Gamma})^{-1}(z^{-1}\mathbf{I} - \mathbf{\Gamma})^{-1}\mathbf{U}^{-1}\mathbf{s},$$

$$= \mathbf{s}^T\mathbf{P}_\pi\mathbf{U}\hat{\mathbf{\Gamma}}(z)\mathbf{U}^{-1}\mathbf{s}, \tag{2.15}$$

where $\hat{\mathbf{\Gamma}}(z)$ is a diagonal matrix with the (i, i)th entry as $(1 - \gamma_i^2)/[(z - \gamma_i)(z^{-1} - \gamma_i)]$. This means that the poles of the PSD lie on the positive real axis in the z-plane because $0 < \gamma_i \leq 1$. Therefore, the PSD of the Markov chain, which characterizes an ion-channel signal will always have low-pass characteristics.

The wavelet power spectrum at scale l and location m of the Markov random process $\tilde{x}_t$ is given by [20],

$$W_{lm} = \sum_{n=-\infty}^{\infty} R(n)\Psi_{lm}(n), \tag{2.16}$$

where $l \in \mathbb{Z}$, $m \in \mathbb{Z}$ and $\mathbb{Z}$ is the set of integers. Note that in this case, we do not assume the random process to be zero mean, but it is easy to remove the contribution of mean from $R(n)$. $\Psi_{lm}(n)$ is the two-sided autocorrelation function of the wavelet function $\psi_{lm}(\xi)$ at scale l and location m given by,

$$\Psi_{lm}(n) = \sum_{\xi=0}^{\infty} \psi_{lm}(\xi)\psi_{lm}(\xi + |n|). \tag{2.17}$$

When the parameters of the random process are not available and we only have the realization $\{x_t\}_{t=0}^{T-1}$, we can use the Discrete Wavelet Transform (DWT) to compute the distribution of energy across multiple scales. The scalogram at scale l is defined as,

$$W^s(l) = \sum_m c_{lm}^2, \tag{2.18}$$

where c_{lm} is the wavelet coefficient of $\{x_t\}_{t=0}^{T-1}$ at scale l and location m. It has been shown that under certain conditions [20], $E[W^s(l)] = \sum_m W_{lm} + O(T^{-1})$. We use the scalogram in order to classify ion-channel signals.

Similar to the Fourier power spectrum, the Walsh power spectrum of a random process is defined to be the Walsh-Hadamard transform of the *logical* auto-covariance function. When the realization $\{x_t\}_{t=0}^{T-1}$ of the random process is available and $T = 2^n$, for some positive integer n, the Walsh transform is defined as $H(m) = T^{-1}\sum_{t=0}^{T-1} H_{mt}x_t$ for $m = 0, 1, \cdots, T-1$ [21]. Half the number of zero crossings of the Walsh function H_{mt} is referred to as *sequency*. The expected value

$E[H(m)]$ has been proven to be equal to the Walsh power spectrum $H^p(m)$ and this is known as the logical Wiener-Khintchine theorem [21, 22]. Therefore, for practical purposes, the Walsh power spectrum can be computed by dividing the realization of the random process into equal length windows, calculating and squaring the Walsh transform coefficients point by point and averaging all the coefficients to find the Walsh power spectral estimate.

2.2 Learning Subspace Based Features

Recently there has been great interest in learning basis from the data directly instead of using fixed transformations. This is advantageous in terms of the efficiently and the quality of representations obtained. Intuitively, we can say that the span of the basis vectors learned helps us discover the natural structure in the data. The criterion used to learn the basis lead to different methods. Minimum reconstruction error in representations lead to Karhunen Love Transform (KLT). Statistically independent coefficients requirement leads to the method called Independent Component Analysis (ICA). And one of the most well algorithm Principal Component Analysis (PCA) retains coefficients whch preserve the maximum variance in the data. PCA algorithm can be summarized as follows

1. Let the data be represented as $\mathbf{X}$. Subtract the mean to centralize the data.

2. Compute the covariance matrix $C = \frac{1}{N-1}\mathbf{X}^T\mathbf{X}$.

3. Compute the eigenvectors of the covariance matrix.

4. The learned basis $\mathbf{V}$ are the S eigenvectors that correspond to the largest S eigenvalues.

5. Also using SVD, the right singular vectors of $\mathbf{X}$ can form $\mathbf{V}$.

6. Features are the projection $\mathbf{F} = \mathbf{V}\mathbf{X}$ of the data onto the space spanned by $\mathbf{V}$.

2.2.1 Sparse Models

Sparse models assume that instead of a global subspace the data lies on the union of multiple subspaces. A sparse linear combination of elementary features [23] can capture statistical structure in data and allows for their efficient representation. The linear model used for sparse coding is given by

$$\mathbf{y} = \mathbf{\Psi}\mathbf{a} + \mathbf{n}, \tag{2.19}$$

where $\mathbf{y} \in \mathbb{R}^M$ is the data vector, $\mathbf{\Psi} = [\boldsymbol{\psi}_1\boldsymbol{\psi}_2 \dots \boldsymbol{\psi}_K] \in \mathbb{R}^{M \times K}$ is the dictionary, that contains the set of representative patterns, whose columns are normalized to unit ℓ_2 norm. $\mathbf{a} \in \mathbb{R}^K$ is the coefficient vector and $\mathbf{n}$ is a noise vector whose elements are independent realizations from $\mathcal{N}(0, \sigma^2)$. Sparse representation problems require a good approximation with a constraint on coefficient sparsity. Adapting dictionaries to the data allows the extraction of key patterns specific to the data, thereby providing a better representation than using predefined dictionaries. Learned dictionaries are useful in many signal/image processing applications [24], [25].

Sparse Coding Algorithms

Some of the widely used methods for computing sparse representations [26] include the Matching Pursuit (MP), the Orthogonal Matching Pursuit (OMP), the Basis Pursuit (BP) and iterated shrinkage approaches. When structured as an overcomplete set of vectors adapted to the training data [27], dictionaries achieve improved performances in several applications.

For an orthonormal dictionary, $\mathbf{\Psi}$, we can compute an $S-$sparse representation by choosing S atoms that provide the largest inner products with the target signal. This

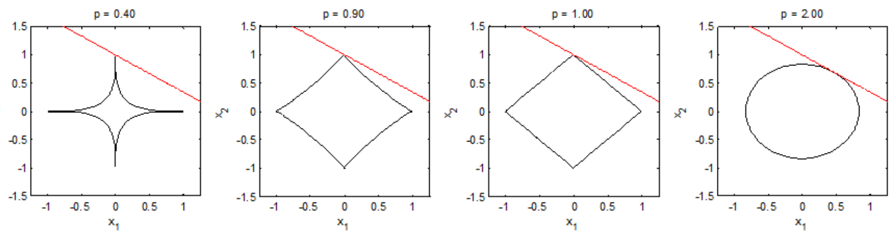


Figure 2.3: l^p norms. The Behaviour of the Norm Functions With Different p Values are Shown. $p \leq 1$ Lead to Sparse Solutions Among Which $p = 1$ is Convex. $p = 2$ Leads to Standard Convex Squared Euclidean Norm.

can be performed greedily by choosing the most strongly correlated atom, removing its contribution from the signal and iterating. The greedy methods such as MP and OMP generalize this idea to the case of any arbitrary dictionary. Hence, greedy methods make a sequence of locally optimal choices in an effort to obtain a global optimal solution [28]. The OMP algorithm performs a least squares minimization in each step to ensure that the approximation is obtained over all the atoms that have been chosen until that step. This implies that the greedy selection always picks an atom that is linearly independent from the atoms already chosen.

Dictionary Learning Algorithms

Dictionary learning problem can be written mathematically as the minimization of

$$g(\mathbf{\Psi}) = \mathbf{E}_{\mathbf{x}}[l(\mathbf{x}, \mathbf{\Psi})] \tag{2.20}$$

where $l(\mathbf{x}, \mathbf{\Psi})$ denotes the representation error associated with $\mathbf{x}$ when using $\mathbf{\Psi}$ as the dictionary. The optimal dictionary $\mathbf{\Psi}$ is obtained by minimizing the expected representation error for all data points. This is hard problem as we usually do not know the underlying data distribution. We resort to empirical minimization using the available training data points and assigning equal probabilities to each of them.

$$g(\mathbf{\Psi}) = \frac{1}{T}\sum_{i=1}^{T} l(\mathbf{x}_i, \mathbf{\Psi}) \tag{2.21}$$

When we know the sparse codes, squared $l2-$ norm of the approximation error $\mathbf{e}_i = \mathbf{x}_i - \mathbf{\Psi}\mathbf{a}_i$ is used as the loss function $l(.)$.

$$\begin{aligned} &\text{minimize}_{\mathbf{\Psi}} \sum_{i=1}^{T} \|\mathbf{x}_i - \mathbf{\Psi}\mathbf{a}_i\|_2^2 \\ &\text{subject to } \|\Psi_j\|_2 \leq 1 \text{ for all } j = 1, .., K \end{aligned} \tag{2.22}$$

In the method of optimal directions, alternating minimization scheme is used. Using $l0$-minimization constraint the sparse codes are obtained in one iteration.

$$\begin{aligned} &\text{minimize}_{\mathbf{a}_i} \sum_{i=1}^{T} \|\mathbf{x}_i - \mathbf{\Psi}\mathbf{a}_i\|_2^2 \\ &\text{subject to } \|\mathbf{a}_i\|_0 \leq s \text{ for all } i = 1, .., T \end{aligned} \tag{2.23}$$

In the next iteration, all dictionary is updated such it minimizes

$$MSE = \frac{1}{T}\|\mathbf{X} - \mathbf{\Psi}\mathbf{A}\|_F^F \tag{2.24}$$

where $\mathbf{X} = [\mathbf{x}_1\mathbf{x}_2 \ldots \mathbf{x}_T]$ is the data matrix and $\mathbf{A} = [\mathbf{a}_1\mathbf{a}_2 \ldots \mathbf{a}_T]$ is the matrix containing the sparse coefficients of all the data samples.

The minimization leads to a closed form solution and the dictionary update at time $t+1$ is given as

$$\mathbf{\Psi}_{t+1} = \mathbf{X}\mathbf{A}_t^T(\mathbf{A}_t^T\mathbf{A}_t)^{-1} \tag{2.25}$$

The columns of the dictionary are normalized after each update.

To speed up the process the K-SVD algorithm proposes to update one dictionary column or *atom* at a time and recompute the sparse codes for all the samples. The dictionary update step can therefore be simplified as

$$\|\mathbf{X} - \mathbf{\Psi}\mathbf{A}\|_F^F = \|\mathbf{X} - \sum_{j=1}^{K} \Psi_j \mathbf{a}_j^T\|_F^F = \|\mathbf{X} - \sum_{j \neq k}^{K} \Psi_j \mathbf{a}_j^T - \Psi_k \mathbf{a}_k^T\|_F^F = \|\mathbf{E}_k - \Psi_k \mathbf{a}_k^T\|_F^F \tag{2.26}$$

This reduces to computing rank-1 SVD of $\mathbf{E}_k$ and using it to update the k^{th} atom. This can lead to dense representations and is therefore modified by using only those samples which use the k^{th} atom to construct $\mathbf{E}_k^{'}$ and $\mathbf{a}_k^{'}$.

K-means algorithm is a special case of the joint optimization problem of learning the dictionary and the sparse codes. The sparse codes are restricted to have one non-zero component out of the possible K and also that value has to be 1. This essentially leads to clustering of the data and the centroids of each cluster can be used as dictionary atoms. This has enjoyed great success in general ML systems due to its low computational complexity compared to other algorithms. After simplifying for this special case, the joint optimization problem reduces to

$$\text{minimize}_{\mathbf{S},\mathbf{\Psi}} \sum_{j=1}^{K} \sum_{i \in S_j} \|\mathbf{x}_i - \Psi_j\|_2^2$$

where S_j is a set containing the indices of data samples that pick the atom j. This is solved by iterating over the following two steps. In the assignment step, the 1-sparse codes are obtained for the data samples using the current dictionary atoms (centroids) and $\mathbf{S}$ is updated after solving for all data samples.

$$\text{minimize}_j \|\mathbf{x}_i - \Psi_j\|_2^2$$

In the update step, each dictionary atom is updated as the average of the data samples which picked it.

$$\Psi_j = \frac{1}{|S_j|} \sum_{i \in S_j} \mathbf{x}_i$$

Recently, this has been used as fast dictionary learning algorithm in sparse code based systems with state-of-the-art results in several recognition tasks.

K-lines based dictionaries are based on k-hyperline clustering of the data. Unlike k-means, which minimizes intra-cluster distances, k-lines tries to fit 1-D subspaces (hypelines) to the data. The sparse codes are still 1-sparse but now can also attain values other than 1. The assignment step changes to

$$j = \arg\min_{j} \|\mathbf{x}_i - \Psi_j(\mathbf{x}_i^T \Psi_j)\|_2^2$$

In the update step, the atoms are updated by solving the following using SVD algorithm.

$$\Psi_j = \arg\min_{\Psi} \sum_{i \in S_j} \|\mathbf{x}_i - \Psi(\mathbf{x}_i^T \Psi)\|_2^2$$

All the data samples which picked j^{th} atom are accumulated in a matrix $\mathbf{X}_j$ and the top-most left singular vector of that matrix is used as update the atom. Compared to k-means, this algorithm has better generalization properties and is invariant to scaling of the data samples. The multilevel dictionary (MLD) learning algorithm is a hierarchical procedure where the dictionary atoms in each level are obtained using this k-lines clustering. Multilevel dictionaries have been shown to generalize well to novel test data, and have resulted in high performance in compressive recovery.

Instead of learning dictionaries using sophisticated learning algorithms, it is possible to use the training examples themselves as the dictionary. This gives tremendous speed advantage when a small number of random data samples are used as dictionary atoms as the learning step is completely eliminated. This approach has been applied in image recovery problems such as super-resolution and compressive sensing.

This approach has also been used to built robust graphs called $l1$-graph by finding a sparse linear combination for a data sample using the rest of the data samples. Low dimensional representations for the data using this graph have been shown to work well in image recognition tasks. In a classification framework based on examples as dictionaries, the training data from all the classes are stacked into a matrix $\mathbf{\Psi} = [\mathbf{\Psi}_1 ... \mathbf{\Psi}_C]$, where C is the number of classes. For a test data $\mathbf{y}$, a sparse code $\mathbf{a}$ is obtained using $\mathbf{\Psi}$ as the dictionary. Let Ω_c be the operator to select the indices of the class c examples. The predicted class label p is obtained by solving,

$$p = \arg\min_c \|\mathbf{y} - \mathbf{\Psi}_c \Omega_c \mathbf{a}\|_2^2 \tag{2.27}$$

This harnesses the subspace structure in the data and is robust to noise in the data.

Completely opposite approach to the joint learning of dictionary and sparse codes in the use of random vectors as dictionary atoms. This can work well in scenarios when are interested in learning discriminatory models as compared to recovery models. The structure in the data is encodes through the sparse coding step only. This approach can be really fast and is used in the context of hierarchical and deep representations, where structure and coding help us encode structure in the data.

2.2.2 Matrix Decomposition Based Methods

Learning a data model can be be posed as matrix completion problem under low rank conditions, where unreliable entries are eliminated. This allows to exploit the structure in data at the global scale.

Consider a matrix $\mathbf{M} \in \mathbb{R}^{n_1 x n_2}$ with missing entries. The indices of the observed entries $(i, j) \in \Omega$ where Ω is a subset of the cross-product set $\{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$. The sampling operator P_Ω applied to a matrix $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2}$ is given by

$$[P_\Omega(\mathbf{Y})]_{i,j} = \left\{ \begin{array}{rl} Y_{i,j} & (i,j) \in \Omega \\ 0 & \text{otherwise} \end{array} \right\} \tag{2.28}$$

A unique low rank matrix $\mathbf{Y}$ consistent with the observed entries of $\mathbf{M}$ exists when the singular vectors of the latter matrix obeys certain conditions. Such a matrix $\mathbf{Y}$

can be obtained by solving the following optimization problem.

$$\begin{aligned}&\text{minimize } \operatorname{rank}(\mathbf{Y})\\&\text{subject to } P_\Omega(\mathbf{Y}) = P_\Omega(\mathbf{M})\end{aligned} \tag{2.29}$$

The conditions on the singular vectors of M are expressed as

$$\|u_k\|_{l_\infty} \leq \sqrt{\mu_B/n_1}, \|v_k\|_{l_\infty} \leq \sqrt{\mu_B/n_2} \tag{2.30}$$

where $k \in [r]$, r is the rank of the matrix $\mathbf{M}$, u_k and v_k are singular vectors of matrix $\mathbf{M}$ obtained using singular value decomposition (SVD). When μ_B is small the singular values are well spread and are not spiky.

The rank minimization problem in (2.29) is non-convex and NP-hard. The rank can be replaced by the *nuclear norm* defined as the sum of the singular values of the matrix. It has been shown that this is the tightest convex relaxation to the rank minimization problem [29]. The relaxed problem is given by

$$\begin{aligned}&\text{minimize } \|\mathbf{X}\|_*\\&\text{subject to } P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{M})\end{aligned} \tag{2.31}$$

where $\|\mathbf{X}\|_* = \sum_k \sigma_k$ is the *nuclear norm* of the matrix $\mathbf{X}$.

We can also decompose the matrix $\mathbf{D}$ directly as $\mathbf{D} = \mathbf{L} + \mathbf{S} + \mathbf{G}$, where $\mathbf{S}$ is a non-positive sparse error matrix and $\mathbf{G}$ is a dense noise matrix. In order to extract the baseline components in the signals, we impose a low rank constraint on $\mathbf{L}$. As a result, we obtain the following joint optimization problem:

$$\begin{aligned}&\arg\min_{\mathbf{L},\mathbf{S}} \|\mathbf{D} - \mathbf{L} - \mathbf{S}\|_F^2,\\&\text{subject to } \operatorname{rank}(\mathbf{L}) < r, \operatorname{card}(\mathbf{S}) < s.\end{aligned} \tag{2.32}$$

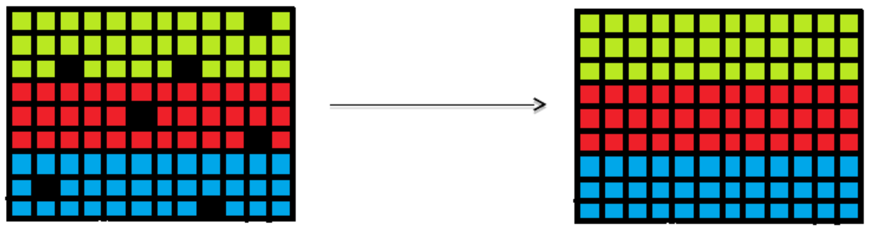


Figure 2.4: Matrix Completion. The Problem Involves Filling the Missing Entries Under Low Rank Assumption.

2.2.3 Topic Models

Topic models can be viewed as discrete analog of the PCA for dimensionality reduction of the data. In these models, a particular data vector is interpreted as a document made of multiple words. The inference step involves associating the document with a topic or a mixture of topics in the probabilistic setting. In the later case, each topic provides a distribution over the discrete words. Consider a corpus of D documents $d_1, ..., d_D$. Each document d_i contains N words $v_1, ..., v_N$. Each word comes from vocabulary comprising of K words. The training process involves inferring J latent topics $h_1, ...h_J$ by maximizing the joint probability $p(v, h)$.

In Latent Dirichlet Allocation (LDA) approach of topic modeling, each word is assumed to be generated from a single topic. The topic mixture for each document is drawn from a Dirichlet distribution. Exact Inference is intractable and therefore variational methods are employed.

RSM topic models belongs to the family of undirected, energy-based models known as restricted Boltzmann machines (RBM). Compared to LDA RSM allows a distributed representation for the topics and alternatively, each words can be assumed to be generated from multiple topics. Boltzmann machines (BM) are Energy Based Models (EBMs) where each configuration of variables is mapped to a scalar energy function. BMs are a special form of log linear Markov Random Fields (MRFs) whose energy function is linear in its free parameters. The modelling capacity of a BM is improved by incorporating variables which are never observed. The energy function of a BM is denoted as $E(\mathbf{v}, \mathbf{h})$, where $\mathbf{v}$ denotes the visible or observable variables and $\mathbf{v}$ are hidden or unobserved variables. Each configuration of a visible variable $\mathbf{v}$ is assigned the probability based on the energy value.

RBMs are a special form of BMs with no connections among the visible variables or among the hidden variables. Connections exist only between a pair consisting of one visible variable and one hidden variable. In the case of RBM, the energy of a configuration is given by, $E(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^T\mathbf{v} - \mathbf{c}^T\mathbf{h} - \mathbf{h}^T\mathbf{W}\mathbf{v}$ where, $\mathbf{b}$, $\mathbf{c}$ are the biases and $\mathbf{W}$ is the matrix of weights between the visible and hidden units. The hidden variables are conditionally independent given the visible variables and vice-versa. We use with RBM with binary units in which inputs and outputs attain value between 0 and 1. The Contrastive Divergence (CD) method [30] is used to learn the parameters. The gradients are updated as $\Delta\mathbf{W} = \langle\mathbf{v}'\mathbf{h}\rangle_{data} - \langle\mathbf{v}'\mathbf{h}\rangle_{recon}$, $\Delta\mathbf{b} = \langle\mathbf{v}\rangle_{data} - \langle\mathbf{v}\rangle_{recon}$ and $\Delta\mathbf{c} = \langle\mathbf{h}\rangle_{data} - \langle\mathbf{h}\rangle_{recon}$ where, $\langle.\rangle$ is the expectation operator computed with respect to the distribution of the data or the reconstructed data. The output of the hidden variables, called activations, can be used as features in any learning framework.

2.2.4 Manifold Based Methods

Manifold learning refers to discovering low-dimensional surfaces in high dimensional spaces which can be used to describe the data efficiently. Unlike sparse coding, where we assume data lies on the union of subspaces, here we are interested in low-dimensional surfaces which can be triangulated. PCA algorithm described above assumes a global linear model and finds subspaces which maximize the variance in the resulting representation. In this section, we will describe some of the more general approaches which help us perform non-linear dimensionality reduction.

Construction of Neighborhood Graph

For a given a set of data points, the similarity matrix $\mathbf{S}$ represents a measure of similarity between all the points. There are several constructions to transform a given set $\{\mathbf{x}_1, ...\mathbf{x}_N\}$ of data points with pairwise similarities s_{ij} into a graph [31]. The local neighborhood relationships between the data points should be preserved while constructing similarity graphs. Some of the well-known constructions are: a) ϵ-neighborhood graph, b) k-neighborhood graph and c) fully connected graph. In our framework, we want to construct a fully connected graph. In this case, we use the Gaussian similarity function $s_{ij} = \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2})$. Here the parameter σ controls the width of the neighborhoods, similarly to the parameter ϵ in case of the ϵ-neighborhood graph.

Let $G = (V, E)$ be an undirected graph with vertex set $V = \{v_1, ..., v_N\}$. The vertices v_i in the graph represent the data $\mathbf{x}_i$. Two vertices are connected if the similarity s_{ij} is larger than a certain threshold. We assume that the graph G is weighted, that is each edge between two vertices v_i and v_j carries a non-negative weight $w_{ij} = s_{ij}$. If the vertices v_i and v_j are not connected w_{ij} is set equal to 0. As G is undirected we require $w_{ij} = w_{ji}$. The weighted adjacency matrix of the graph is the matrix $\mathbf{W} = \{w_{ij}\}_{i,j=1,...,n}$. The degree of a vertex v_i is defined as

$$d_i = \sum_{j=1}^{n} w_{ij} \tag{2.33}$$

Note that this sum only runs over all vertices adjacent to v_i, as for all other vertices v_j the weight w_{ij} is 0. The degree matrix $\mathbf{D}$ is defined as the diagonal matrix with the degrees $d_1, ..., d_N$ on the diagonal.

The unnormalized Laplacian matrix is defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$. There are two matrices which are called normalized graph Laplacians in the literature. Both matrices are closely related to each other and are defined as

$$\mathbf{L}_{sym} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2} \tag{2.34}$$

$$\mathbf{L}_{rw} = \mathbf{D}^{-1}\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W} \tag{2.35}$$

We denote the first matrix by $\mathbf{L}_{sym}$ as it is a symmetric matrix, and the second one by $\mathbf{L}_{rw}$ as it is closely connected to a random walk [32]. It is important to note that both these matrices are positive semi-definite (i.e.) they have N non-negative real-valued eigenvalues $0 = \lambda_1 \leq ... \leq \lambda_N$.

Locality Preserving Projections

Low-dimensional projections can be obtained from the sparse coding graphs using the locality preserving projections (LPP) approach. Given the affinity matrix $\mathbf{W} \in \mathbb{R}^{T \times T}$, we compute the degree matrix $\mathbf{D}$ with each diagonal element containing the sum of the corresponding row or column of $\mathbf{W}$. The d projection directions for LPP are then computed by optimizing

$$\min_{\mathbf{V}} \sum_{i,j=1}^{T} \|\mathbf{V}^T\mathbf{x}_i - \mathbf{V}^T\mathbf{x}_j\|_2^2 w_{ij} \text{ s.t. } \sum_{i=1}^{T} \|\mathbf{V}^T\mathbf{x}_i\|_2^2 \delta_{ii} = 1. \tag{2.36}$$

where w_{ij} and d_{ii} are the corresponding elements of the affinity and the degree matrices. This optimization ensures that the embedding preserves the structure defined by the sparse coding based graph. Defining the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$, (2.36) can be rewritten as

$$\min_{\text{trace}(\mathbf{V}^T\mathbf{X}\mathbf{D}\mathbf{X}^T\mathbf{V})=\mathbf{I}} \text{trace}(\mathbf{V}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{V}). \tag{2.37}$$

and this can be solved using eigen decomposition.

Spectral Clustering

Spectral clustering methods make use of the eigen decomposition of the similarity matrix, also referred to the spectrum of the matrix, in order to perform clustering. Given a set of data points $\mathbf{x}_1, ..., \mathbf{x}_N$ and the similarity $s_{ij} \geq 0$ between the pairs of data points $\mathbf{x}_i$ and $\mathbf{x}_j$, the goal of clustering is to divide the data points into several groups such that points in the same group are similar and points in different groups are dissimilar to each other.

Spectral clustering requires computation of the graph Laplacian matrices. We achieve clustering by finding a partition of the graph such that the edges between different classes have a very low weight and the edges within a class have high weight. In our algorithm, we use a normalized spectral clustering approach proposed in [33]. The algorithm is presented in Table 2.1.

2.3 Deep Architectures

Along with sparsity depth is also an important paradigm that can be used to develop efficient and informative representations for the data. This can be used to uncover the latent hierarchical structure found naturally in data such as image and audio and may other data modalities. Human brain is believed to process sensor inputs in a multi-level hierarchical fashion. Visual information is processed in the visual cortex where the V1 cells behave as Gabor functions and perform edge detection. The information from these cells are passed on to complex cells and further to sophisticated regions built for higher cognition [34] [35]. These layers build on abstractions and lead to efficiency in representation. Similar observations were made in the case of auditory input where the auditory cortex consists of simple to complex cells and form a hierarchical information flow path [36]. This has been the inspiration

Table 2.1: Normalized Spectral Clustering Algorithm.

```
Input
- Similarity matrix, $\mathbf{S} \in \mathbf{R}^{N \times N}$
- Number of clusters, K

Algorithm
    - Construct a fully connected similarity graph using the input
    similarity matrix $\mathbf{S}$.
    - Compute the weighted adjacency matrix, $\mathbf{W}$.
    - Compute the degree matrix, $\mathbf{D}$.
    - Construct the normalized Laplacian matrix, $\mathbf{L}_{sym}$ as given in (2.34)
    - Find the $K$ largest eigenvectors $\mathbf{v}_1, ..., \mathbf{v}_K$ of $\mathbf{L}_{sym}$ and
    construct the matrix $\mathbf{V} \in \mathbf{R}^{N \times K}$ with the eigenvectors
    as the columns.
    - Form the matrix $\mathbf{U}$ from $\mathbf{V}$ by renormalizing each of $\mathbf{V}$'s
    rows to have unit length.
    - Treating each row of $\mathbf{U}$ as a point in $\mathbf{R}^K$, cluster them into $K$
    clusters using any algorithm that minimizes distortion (k-means).
    - Finally assign the point $\mathbf{x}_i$ to cluster $k$ if the row $k$ of the
    matrix $\mathbf{U}$ was assigned to cluster $k$.
```

of many ML models especially for vision and audio recognition systems. Experiments have also shown that after rewiring the auditory nerves to the visual cortex, the later can process audio information leading to the conclusion that after the initial transformation of the sensor data, similar learning process unfolds irrespective of the data type [37].

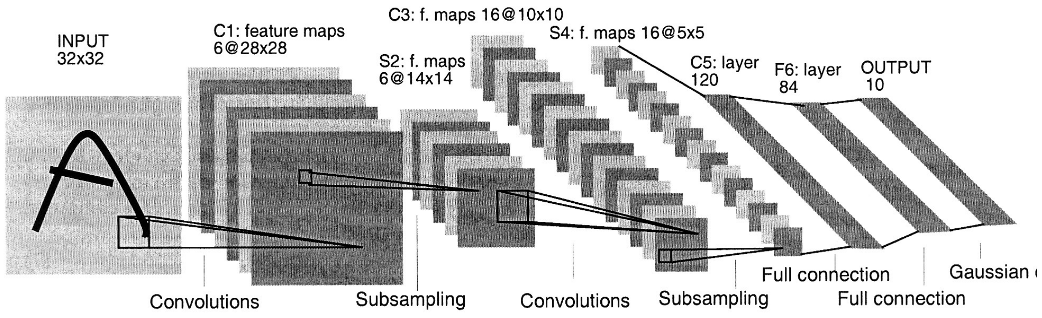


Figure 2.5: Lenet Architecture Used for Hand-writing Recognition [4].

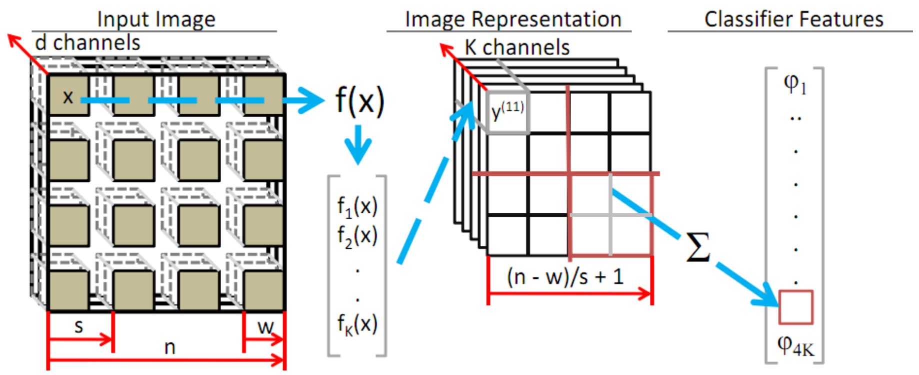


Figure 2.6: Coates Deep Architecture for Image Classification Task [5].

Artifical Neural Networks (ANNs) with layers of nodes called neurons are loosely modeled on the brain neurons. Layers with hidden nodes are used in supervised learning problems. Issues with training these models slowed down the progress in this area albeit Convolutional Neural Networks (CNN) have enjoyed early success [4]. With the availability of vast amount of data coupled with better training algorithms, the field is seeing lot of success. Advent of affordable computing resources such as Graphical Processing Unit (GPU) and cloud infrastructure such as Amazon Web Services (AWS), it is now possible to train the traditional NN models much faster and obtain state-of-the-art results.

2.3.1 Receptive Fields

CNNs were able to mitigate the problem of vanishing gradients during the back-propagation by having sparse and local connection between two layers. A neuron is thus sensitive only few neurons in the previous layer. Similar behavior has been found in the brain where a particular neuron has a fixed region of influence called the receptive field. Computationally, this reduces the number of parameters and keeps the learning tractable.

The receptive field in a CNN is purely in terms of spatial locality and is a fixed parameter. In [38], authors propose auto-pooling for image sequences to automatically group the filters to determine the pooling region. This region is found while balancing the invariance desired and the information loss due to pooling. The main idea guiding their approach is that consecutive images should have similar pooled features.

In [5], the authors propose a scalable approach to cluster the filters and hence obtain the receptive fields. The methods uses the correlations between the filter responses to compute the relationship between them. The input feature responses were pre-processed using ZCA whitening to obtain uncorrelated responses. The similarity between each pair of features j and k was then computed using all the feature responses $\mathbf{X} = [\mathbf{x}^1 ... \mathbf{x}^N]$ as

$$S_{j,k} = \frac{\sum_i (x_j^i)^2 (x_k^i)^2 - 1}{\sqrt{(\sum_i (x_j^i)^4 - 1)(\sum_i (x_k^i)^4 - 1)}} \tag{2.38}$$

2.3.2 Pooling Operations

Once the receptive field or the pooling region is obtained, the next operation is to generate a aggregated response in that region. In the case of binary features it can be shown [39] that, the pooling operations specifically max-pooling and square pool-

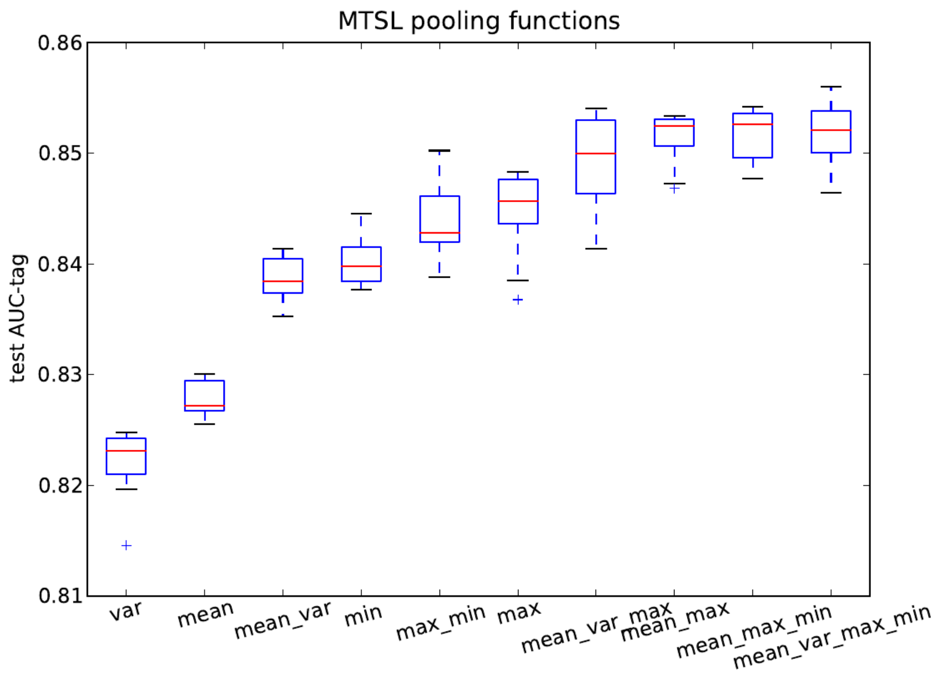


Figure 2.7: Combination of Multiple Pooling Functions Improves Performance as Shown in [6].

ing increase the separability of the resulting features class conditional distribution. Assuming K independent filters and P locations in the pooling region, the un-pooled data is in the form of a PxK matrix $\mathbf{V}$. Let the pooled features be represented by f and the two classes be represented as C_1 and C_2. The class conditional distribution of the features can then be denoted as $p(f|C_1)$ and $p(f|C_2)$. Response to a filter j (column i in the matrix $\mathbf{V}$) at the locations can be represented as $\mathbf{v}_j$. The average pooling operation is given as

$$f_a(\mathbf{v}) = \frac{1}{P}\sum_{i=1}^{P}\mathbf{v}_i \tag{2.39}$$

This is in fact the sum over i.i.d Bernoulli variables of mean α resulting into a binomial distribution. Thus $f_a \sim \text{Binomial}(\mu_a, \sigma_a^2)$ with

$$\mu_a = \alpha \tag{2.40}$$

$$\sigma_a^2 = \frac{\alpha(1-\alpha)}{P} \tag{2.41}$$

The separation can increase when there is shift in the means of the class conditional distribution of the pooled features or the variance decreases. In the case of square pooling or average pooling the variance decreases as P increases leading to good performance when the α is reasonably high. Max-pool operation f_m turns out to be better when the feature activations are sparse and thus $alpha$ is low. This operation is given as

$$f_m(\mathbf{v}) = \max_i \mathbf{v}_i \tag{2.42}$$

$$f_m \sim \text{Bernoulli}(\mu_m, \sigma_m^2) \tag{2.43}$$

$$\mu_a = 1 - (1-\alpha)^P \tag{2.44}$$

$$\sigma_a^2 = (1-(1-\alpha)^P)(1-\alpha)^P \tag{2.45}$$

Assuming that the mean activation rate for C_1 and C_2 are α_1 and α_2, respectively, the distance between the means of the class conditional distribution is given as

$$\Phi = |(1-\alpha_2)^P - (1-\alpha_1)^P| \tag{2.46}$$

Φ increases between $(0, P_M)$ and decreases between (P_M, inf), where P_M is given as

$$P_M = |\frac{log(\frac{log(1-\alpha_2)}{log(1-\alpha_1)})}{log(\frac{(1-\alpha_1)}{(1-\alpha_2)})}| \tag{2.47}$$

Also the variance attains its peak value 0.5 when P is

$$P = \frac{log2}{|log(1-\alpha)|} \tag{2.48}$$

The above results tell us that there is an optimal pooling cardinality value P which gives us best performance, deviating from it can decrease the separability and hence decrease the performance of the resultant pooled features. The optimal pooling cardinality increases with number of filters used.

In [40], the authors show that the square pooling operation on convolutional filter responses is inherently frequency selective. The near optimal input for a pooling unit was found to be a sinusoid at the maximal frequency present in the corresponding filter. More specifically, in the case of circular convolutions, they show that near optimal input $x^{opt}[m, s]$ can be given as

$$x^{opt}[m,s] = \frac{\sqrt{2}}{n} cos(\frac{2\pi mv}{n} + \frac{2\pi sh}{n} + \Phi) \tag{2.49}$$

where v and h and the maximum frequency present in the filter. This hold true even when the filters are randomly chosen. Thus, a well chosen architecture and encoding process can make a considerable difference in the performance of a feature learning or recognition system.

Chapter 3

FEATURES FOR ION-CHANNEL SIGNALS

The interest in analyte classification based on a deterministic change in the stochastic switching behavior of ion-channels was raised by the results published in [17]. Different classes of analytes can be distinguished by ion-channel proteins that have been modified to allow for a significant change in gating behavior upon presence of a respective analyte [41]. Such modifications can be accomplished using genetic engineering. Signal processing techniques for analyzing the ion-channel signals have been proposed in [42].

Dwell time analysis of ion-channel signals involve statistical characterization of the times of open and closed states. The traditional way of analyzing the single channel kinetics is fitting lifetime histograms to the dwell times [43]. Continuous-time Markov processes are used as more sophisticated models for the dwell times [18] and an approach has been proposed to estimate the kinetic parameters from data containing missing events [44]. For extracting the idealized data from the noisy patch clamp record, an approach based on Hidden Markov Models (HMMs) and segmental k-means has been proposed [45]. Since idealizing the data to obtain dwell times produces unreliable results during low signal-to-noise ratio conditions, approaches based on direct modeling of the raw data using HMMs have gained prominence. Metastate or vector HMMs have been proposed to deal with the colored noise in the raw ion-channel data. Alternative transform domain approaches suited for characterizing the ion-channel signals have also been developed. A method for classifying ion-channel signals using neural networks has been developed in [46]. The stationarity of the ion-channel signals and the changes in the kinetics of ion-channels have been analyzed using the wavelet transform [47]. Chen *et. al.* have used the Power Distribution Fraction (PDF) in the wavelet domain and the Power Spectral Density (PSD) in the Fourier domain to infer the average opening time of the ion-channel [15].

We demonstrate mathematically that the Fourier power spectrum of an ion-channel signal captures the key statistical characteristics of the channel. Results obtained with the Support Vector Machine (SVM) classifier, using the proposed features, show high classification, specificity and sensitivity rates for both simulated and real ion-channel data. We observe that the classifier achieves improved performance when highly noisy signals are denoised prior to feature extraction. Furthermore, we compare the classifier's performance to that obtained using an *ideal* HMM classifier with known parameters. Although we have not used an analyte in our experiments, the proposed setup is suited for stability analysis and baseline calibration of ion-channels.

3.1 Analyte Sensing Using Stochastic Ion-channel Signal Modulation

For the purpose of analyte detection, the ion-channel $\alpha-$Haemolysin pore of *S. aureus* has been studied in detail. However, ion-channels such as the outer membrane proteins of *E. coli* change their stochastic switching behavior in the presence of antibiotics such as ampicillin [48]. To exactly determine the concentration of the analyte, it is necessary that the switching behavior of the ion-channel itself is well known.

In order to obtain recordings of the ion-channel current signals, the channels themselves have to be embedded in a lipid bilayer membrane. This membrane itself does not allow ions to penetrate, thus it exhibits a high electrical resistance on the order of tens of Gigaohms. Isolating and accessing ion-channels of interest is accomplished using the patch-clamp technique in which a glass pipette with an orifice diameter of about 1 μm is brought in close contact to the outer membrane of a cell so that a Gigaseal forms. However, it is not guaranteed that the membrane patch will contain the channel of interest, in particular if the rate of expression of this channel in the outer membrane of a cell is low. To circumvent this problem, ion-channels of interest can be extracted from cells, purified and reconstituted into artificial lipid bilayer membranes. Ion-channel reconstitution offers the advantage of working with purified proteins, thus limiting the probability of observing current switching that is characteristic of other ion-channels that might be mistaken for analyte signatures. However, even in case of a known ion-channel, its stochastic signal might vary over time, even with no analyte present. Since lipid bilayers are fluidic entities, the spatial position of a single ion-channel within the membrane might not be constant and the baseline signature of the particular channel might be affected. Thus, it is necessary to establish a sound baseline for the stochastic signature before any analyte is added.

3.2 Setup for Outer Membrane Protein (OmpF) Experiments

To demonstrate the capabilities of the classification algorithms for assessing the baseline of an OmpF ion-channel, lipid bilayers were formed across apertures in silicon chips. These apertures were formed using dry reactive ion etching. To reduce the capacitance of the structure, photo-polymerizable epoxy resin (SU-8 2025) was applied and patterned on the back side of the samples. The surface of the chip was coated with a layer of plasma-polymerized polytetrafluorethylene (PTFE), rendering the surface hydrophobic for lipid bilayer attachment. Details of the microfabrication process have been described in [49, 50]. Samples were mounted in polystyrene holders that allow access to both sides of the silicon chip. Both compartments hold 1ml of 1M KCl solution, buffered with 20mM N-(2-hydroxyethyl) piperazine-N'-(2-ethanesulfonic acid) (HEPES) at a pH of 7.4. Electrical access to the solution wells was provided using As/AgCl wire electrodes. Lipid bilayers were formed using the bubble collapse (painting) method from a mixture of (1,2-dioleoyl-sn-glycero-3-phosphoethanolamine and 1,2-dioleoyl-sn-glycero-3-phosphocholine) (DOPE:DOPC, 4:1) lipids, dissolved in n-decane (10 mg/ml). OmpF ion-channels were reconstituted into these membranes by adding 0.5μl of OmpF stock solution to the *cis* compartment.

We used two identical chips in neighboring wells. The transimpedance amplifier used to record the ion-channel currents was constructed based on the circuit published by Sigworth [51], using off-the-shelf surface-mount components. The amplifier did not employ any analog filtering, resulting in a colored noise spectrum. The amplified signal was digitized using a National Instruments PCI-E 6021 DAQ card at a sample rate of 1 kHz. WinEDR [52] was used to acquire the signal as well as apply the stimulus voltage of 200 mV to the membrane containing the ion-channels.

3.3 Feature Extraction

Consider an ion-channel signal represented by the vector $\mathbf{x} = \{x_t\}_{t=0}^{T-1}$. The problem is to extract relevant features from the Fourier, wavelet and Walsh power spectra of $\mathbf{x}$. The power spectra are computed directly using the signal $\mathbf{x}$, without assuming any knowledge about the statistical characteristics of the ion-channel.

3.3.1 Preprocessing the Data

Ion-channel signals obtained from patch clamp technique often have low Signal-to-Noise Ratio (SNR). It has been observed that the background noise is colored with spectral density that increases over frequency [53]. This can mask the state transition events and change the signal characteristics drastically. Hence signal denoising is essential to improve the accuracy of the statistical model and feature space representation of the signal.

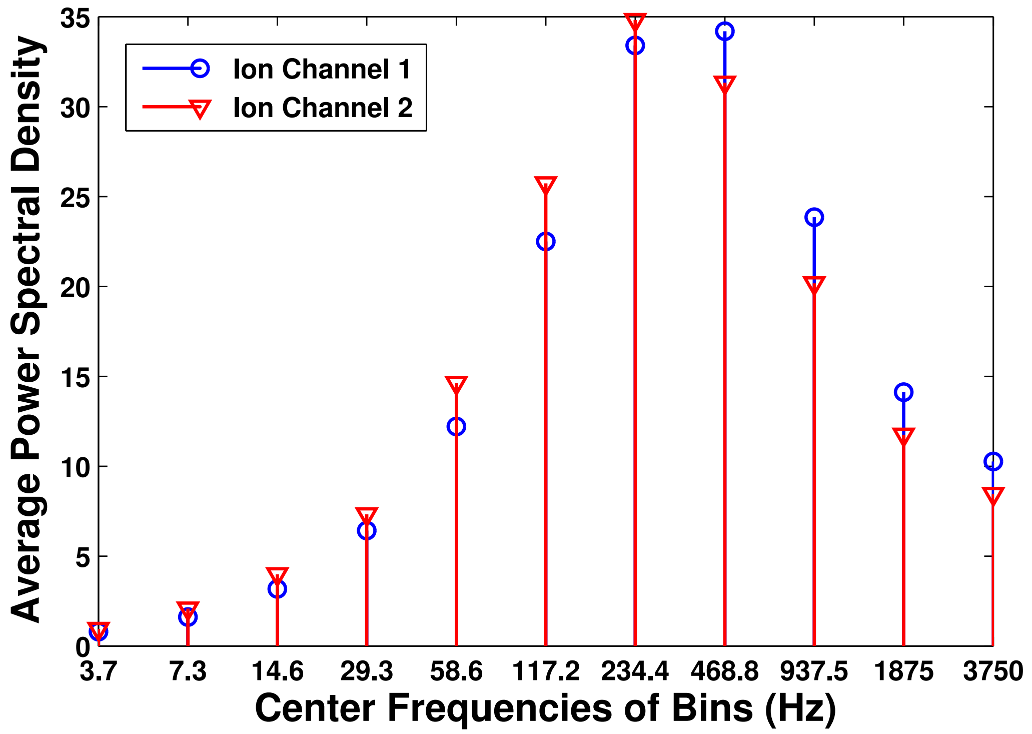


Figure 3.1: Average of Feature Vectors for Two Different Simulated Ion-Channel Signals in Fourier Domain.

The data preprocessing step involves using the DWT to perform denoising of the signal. The advantage of wavelet representation is that it can provide time and frequency parameters for specific dynamic signal events, i.e., *time-frequency localization*. We used the biorthogonal wavelet (bior3.7) with 4 levels of decomposition for preprocessing the data. Different wavelets and number of levels were tested and the ones that provided the best classification performance, as reported in Section 3.4, were chosen. Soft thresholding was used as it provides better performance than hard thresholding and the limit was chosen using the simple and effective universal threshold [54].

The Fourier domain features are obtained by computing the PSD of the ion-channel signal $\mathbf{x}$ using the Welch procedure given in [14]. Estimating the PSD involves windowing of the signal into M_f windows and averaging the modified periodograms computed over the windows to reduce the variance of the estimate. It is also known

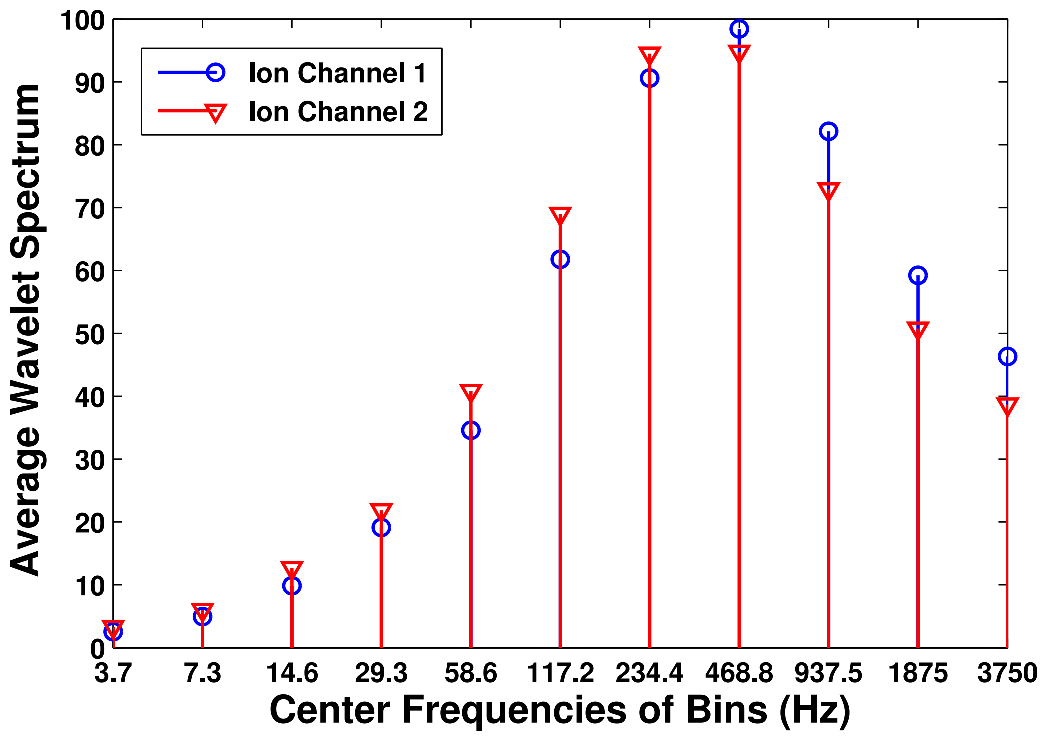


Figure 3.2: Average of Feature Vectors for Two Different Simulated Ion-Channel Signals in Wavelet Domain.

that the PSD obtained using the Welch procedure approximates the Blackman-Tukey type spectral density obtained in (2.15) [14]. The PSD is dependent only on the eigen decomposition of the state transition matrix and it will always exhibit low-pass characteristics. Furthermore, if we denote f_s as the sampling frequency and f_c as the frequency where the flat and sloping portions of the PSD intersect, $1/f_c$ represents the average opening time of the channel [15]. This shows that the Fourier power spectrum contains sufficient discriminatory information to distinguish between different ion-channel signals.

The PSD is divided into bins spaced in powers of two (dyadic bins), ignoring the DC component. The PSD values in each bin are summed to generate the feature vector. The dyadic binning scheme provides more weighting to the lower frequencies and hence better captures the signal power concentrated in the lower frequencies.

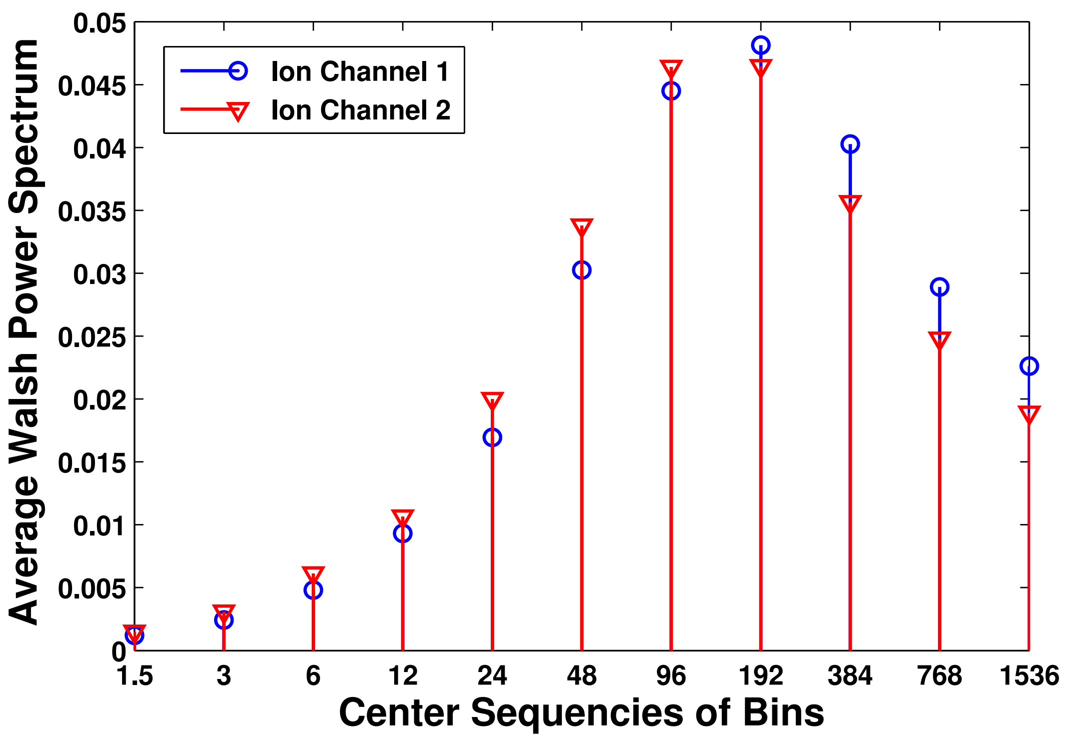


Figure 3.3: Average of Feature Vectors for Two Different Simulated Ion-Channel Signals in Walsh Domain.

Two ion-channel signals are simulated as 2-state discrete Markov processes using the QuB software [55] with rate parameters for the first ion-channel as $q_{12} = 1000$ and $q_{21} = 1000$ and the rate parameters for the second ion-channel as $q_{12} = 900$ and $q_{21} = 750$. States *1* and *2* have conductance levels of 0 pA and 1 pA respectively. A total of 60 signals were generated per ion-channel, each with 16384 samples at 10 KHz frequency. The average of Fourier domain features obtained from the estimated PSDs corresponding to the two ion-channels are shown in fig. 3.1. Each bin represents the frequency range from $f_s/2^{l+1}$ to $f_s/2^l$ and the center frequency of the bin is given by $3f_s/2^{l+2}$, where $l = \{1, ..., L\}$ is the index of the bin.

The scalogram defined in (2.18) is used as the wavelet domain feature and it is just a scaled version of the PDF described in [15]. We use the Haar wavelet, as it has a shape that correlates well with the general switching state structure of an ion-channel signal. The signal $\mathbf{x}$ is divided into M_ψ non-overlapping windows, and the scalogram is averaged across the windows to reduce the variance, similar to the case of estimating the PSD. The center frequency of the scale l in the scalogram is given by $3f_s/2^{l+2}$ and this feature also gives more importance to the lower frequencies (coarser scales) of the wavelet decomposition, which predominantly captures the signal characteristics. The average of wavelet domain features corresponding to the two simulated ion-channels given in Section 2.1.1 are illustrated in fig. 3.2.

The feature vector is generated by dividing the power spectrum into L dyadic bins of sequencies, ignoring the DC component, and summing the spectral estimates in each bin. Fig. 3.3 shows the average of the Walsh domain features corresponding to the two simulated ion-channels given in Section 2.1.1. The Walsh functions are similar to the Haar wavelet except that they lack the time localization properties of Haar wavelet. Hence, this feature also captures the key statistical properties of the ion-channel signals.

The different features described in Section 3.3 are extracted from the ion-channel signals and classified using Support Vector Machines (SVMs) [56]. The performance of the proposed features is evaluated using a set of signals from two simulated channels that are highly similar and also using experimental ion-channel data.

3.4 Simulated Data

The conductance level for the states *1* and *2* was 1 pA and that of states *3* and *4* was 0 pA. A total of 10^6 samples was generated per channel at a sampling rate of 10KHz and i.i.d. Gaussian noise with standard deviation σ was added to the channels. Using non-overlapping frames of size 16384, a dataset with 60 vectors was created for each ion-channel. Each dataset was randomly permuted to obtain a training set with 30 vectors and a test set with 30 vectors. To extract the feature vectors we used overlapping windows of 4096 samples for the cases of Fourier and Walsh while non-overlapping windows of 2048 samples were used for the wavelet domain. The 11-dimensional feature vectors obtained were used to train the SVM classifier and the performance was evaluated using the feature vectors of the test set. We also evaluated the classification performance of *ideal* HMMs, with a-priori knowledge of all parameters of the ion-channels, on the simulated dataset. Classification was performed for all the 120 vectors from both the datasets using a maximum likelihood approach. The performance of the *ideal* HMM is the maximum that can be achieved, as we assume knowledge of all the parameters that were used to obtain the simulated dataset. Note that we use *ideal* HMMs only as a baseline to compare our results, and this performance is not realizable in practice. In practical scenarios parameter estimation needs to be performed for HMMs and the performance will be much lesser.

The parameters used for evaluating the performance are classification accuracy, sensitivity and specificity. The accuracy of classification is the ratio of correctly associated samples to the total number of test samples. Sensitivity and specificity measure the proportion of the correctly identified positives and negatives respectively. The performance of the SVM and HMM classifiers were evaluated under various noise conditions with $\sigma = \{0, 0.1, ..., 0.7\}$. For the SVM classifier, we performed 1000 trial runs for each σ, with different realizations of noise and permutations of training and test sets in each trial. The experiments were repeated with features obtained after denoising. The performance reported represent the average values from multiple trials. The performance of the *ideal* HMM classifier was evaluated as an average of 10 trial runs.

The performance of all the three proposed features are similar and is not too far from the maximum performance achievable with the *ideal* HMMs. Except in the case of Walsh features, denoising lowers the performance at low noise variances since some

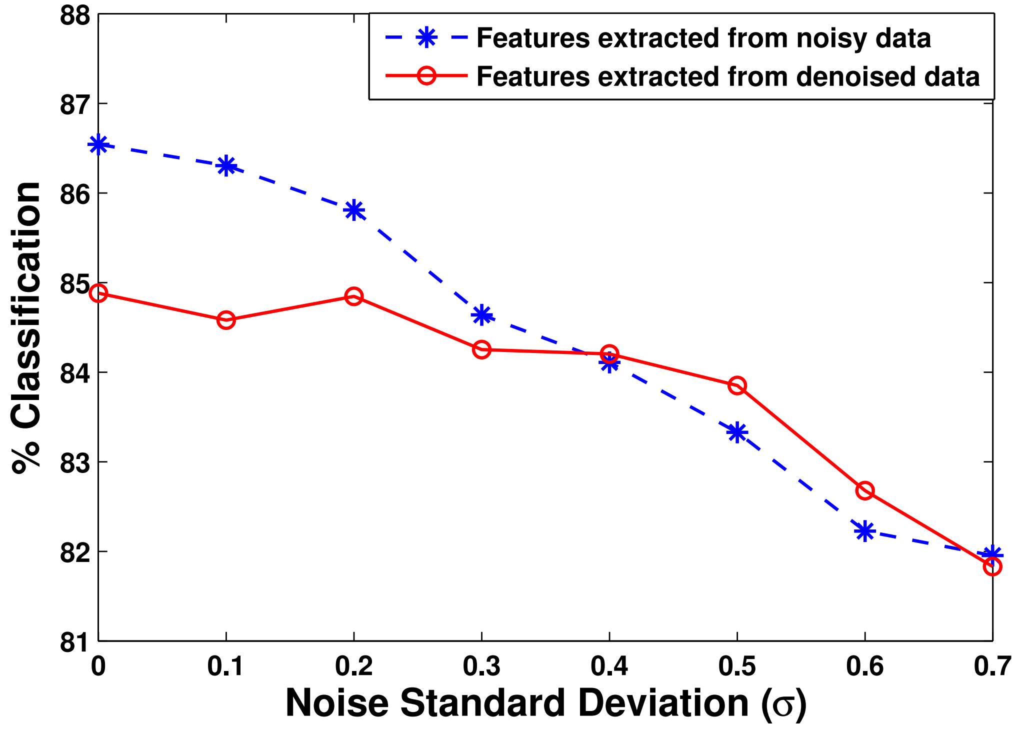


Figure 3.4: Ion-channel Signal Classification Results Using Linear SVM Kernel and Fourier Features.

information in the ion-channel signal is lost due to thresholding.

3.5 Experimental Data

We test the classifier with experimental data from two ion-channels obtained using the setup described in Section 3.2. The sampling rate was 1 KHz and a total of 31882 samples were collected for each channel. The data from each channel was divided into 14 segments each of length 4096, with a 50% overlap.

Classification was repeated for 100 trials with 7 randomly chosen segments from each channel for training and the remaining segments for testing. All the three features achieved more than 99% classification, sensitivity and specificity rates on an average. Using the proposed features, we determined that the ion-channel signals are stable over time and differentiated stochastically dissimilar channels. Note that we

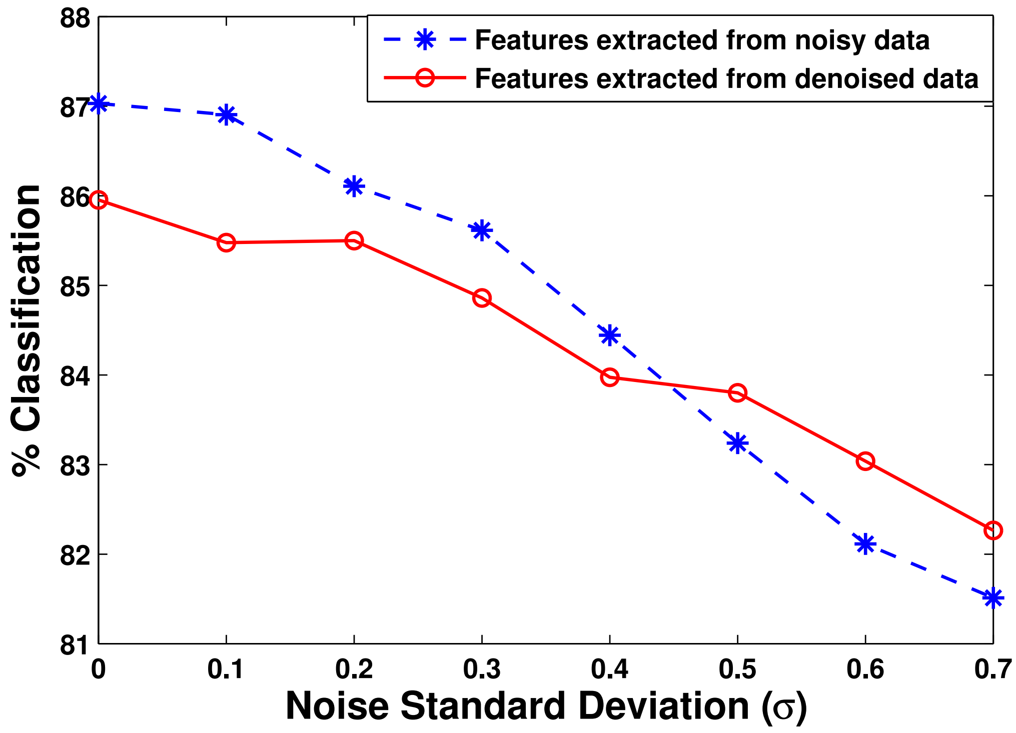


Figure 3.5: Classification Results Using Wavelet Features.

used the limited data available here and in order to characterize the performance of the features more accurately with real data, we need more experimental data. We now describe ways to exploit the structure in power spectral density (PSD) features for ion-channel signals to generate a more robust and discriminative global representations. We will show that the feature enhancing process can be framed as rank minimization problem with global and local regularizations. Exact and noisy matrix completion algorithms under low rank conditions have been proposed in [29]. The feature vectors of the noisy signal segments are extracted and stacked into a matrix. Under noiseless conditions this matrix is typically low rank, since the feature vectors of consecutive segments are similar. However, the presence of noise in the acquired data does not guarantee the low rank behavior of the feature matrix. Hence, the entries of the matrix with high variances are removed to build an incomplete matrix. We now perform

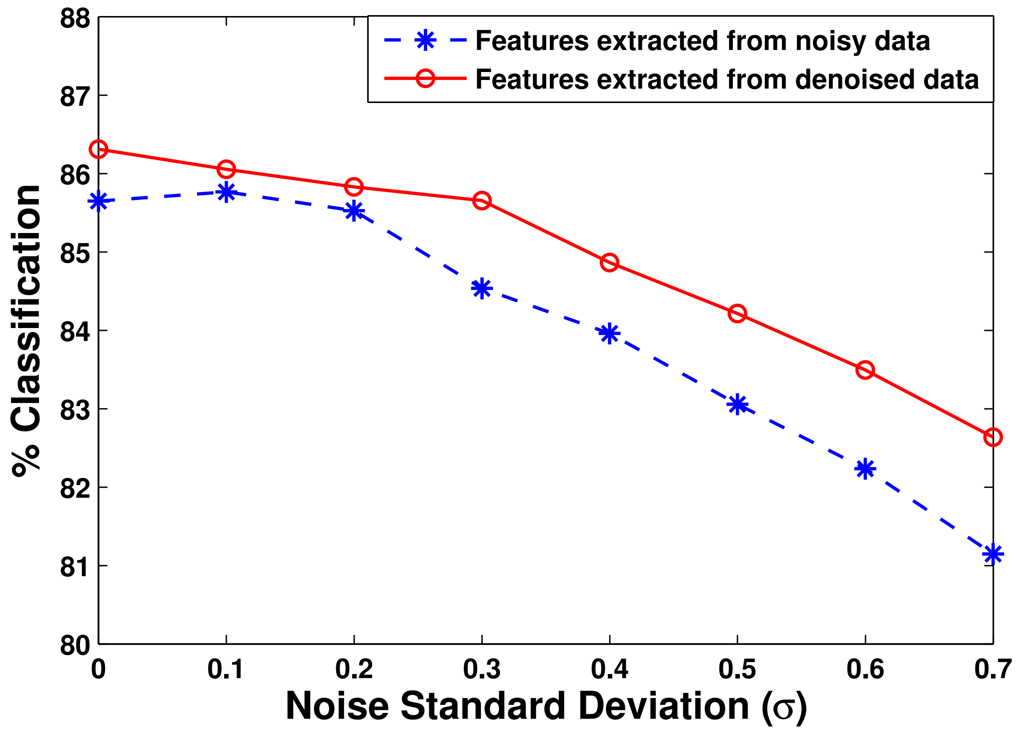


Figure 3.6: Classification Results Using Walsh Features.

matrix completion under a low rank condition and the columns of the completed matrix contain the robust PSD feature vectors. Simulation results obtained with synthetic single ion-channel data show that the stabilized features achieve improved classification performance in comparison to using the features extracted from the denoised signals. Furthermore, we demonstrate the effectiveness of the proposed robust features in reducing the false alarm rates when applied to analyte detection.

3.6 Generating Robust PSD Features Using Matrix Completion

In the problem of matrix completion, the missing entries of a matrix are inferred using a few observed entries, under some constraints. Assuming the matrix to be completed is of low rank and the observed entries are sampled from uniformly ran-

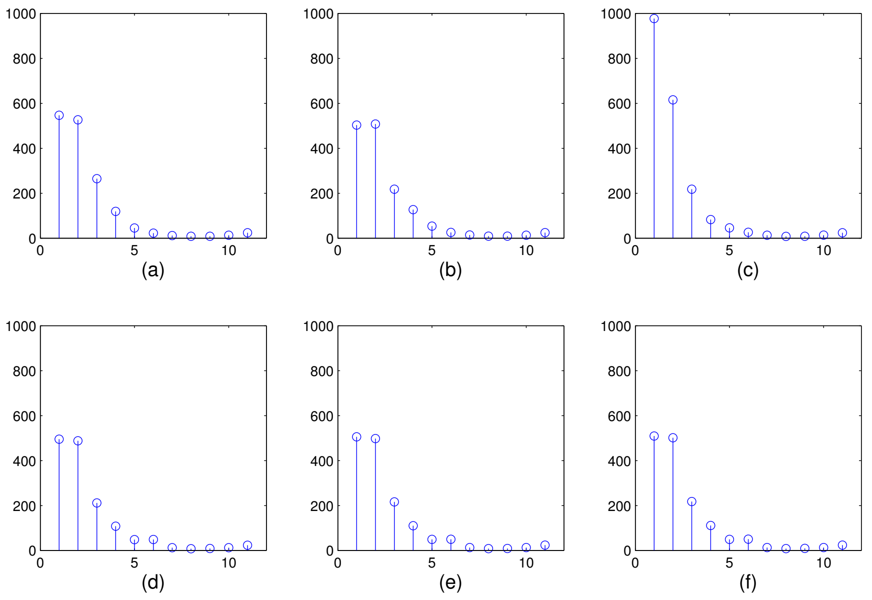


Figure 3.7: Robust PSD Features. (a)-(c) are the Original PSD Features for Three Segments of the Data, While (d)-(f) are the Corresponding Stabilized PSD Features. The X-axis Denotes the Frequency Bins and Y-axis Shows the Average Power Spectral Density.

dom locations in the matrix, exact recovery of the matrix is possible [29]. We pose the problem of stabilization of PSD features as a matrix completion problem. Stabilization here means that we eliminate the outliers in the PSD features make them robust.

Consider a matrix $\mathbf{M} \in \mathbb{R}^{n_1 x n_2}$ with missing entries. The indices of the observed entries $(i, j) \in \Omega$ where Ω is a subset of the cross-product set $\{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$. The sampling operator P_Ω applied to a matrix $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2}$ is given by

$$[P_\Omega(\mathbf{Y})]_{i,j} = \left\{ \begin{array}{cc} Y_{i,j} & (i,j) \in \Omega \\ 0 & \text{otherwise} \end{array} \right\} \tag{3.1}$$

A unique low rank matrix $\mathbf{Y}$ consistent with the observed entries of $\mathbf{M}$ exists when the singular vectors of the latter matrix obeys certain conditions. Such a matrix $\mathbf{Y}$

can be obtained by solving the following optimization problem.

$$\begin{aligned} &\text{minimize } \operatorname{rank}(\mathbf{Y}) \\ &\text{subject to } P_\Omega(\mathbf{Y}) = P_\Omega(\mathbf{M}) \end{aligned} \tag{3.2}$$

The conditions on the singular vectors of M are expressed as

$$\|u_k\|_{l_\infty} \leq \sqrt{\mu_B/n_1}, \|v_k\|_{l_\infty} \leq \sqrt{\mu_B/n_2} \tag{3.3}$$

where $k \in [r]$, r is the rank of the matrix $\mathbf{M}$, u_k and v_k are singular vectors of matrix $\mathbf{M}$ obtained using singular value decomposition (SVD). When μ_B is small the singular values are well spread and are not spiky.

The rank minimization problem in (3.2) is non-convex and NP-hard. The rank can be replaced by the *nuclear norm* defined as the sum of the singular values of the matrix. It has been shown that this is the tightest convex relaxation to the rank minimization problem [29]. The relaxed problem is given by

$$\begin{aligned} &\text{minimize } \|\mathbf{X}\|_* \\ &\text{subject to } P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{M}) \end{aligned} \tag{3.4}$$

where $\|\mathbf{X}\|_* = \sum_k \sigma_k$ is the *nuclear norm* of the matrix $\mathbf{X}$.

Let assume the vectors $\mathbf{b}_1,\ldots,\ \mathbf{b}_N$ are PSD features of N consecutive frames of an ion-channel signal. Note that each element in the feature vector corresponds to average PSD over a certain bin. These vectors are stacked column wise into a matrix $\mathbf{B}$. Ideally, this matrix should be low rank as consecutive frames are realization of the same Markov process and should have similar feature vectors. In order to identify the outlier feature samples and correct them, we assume every bin of the feature vector is a realization of an independent Gaussian random variable. In other words, each row of the matrix $\mathbf{B}$ contains realizations of a Gaussian random variable. Due to various types of noise, we may get some outliers in each bin. The outliers are identified by computing the variance of the entries in each column of $\mathbf{B}$ and identifying the samples whose values are more than an empirically decided threshold. We denote this incomplete matrix by $\mathbf{M}$. In cases where the entire column corresponding to frame has high variance, the column is removed altogether as this feature vector is not useful for classification. Furthermore, matrix completion algorithms cannot handle such scenarios.

Several algorithms have been proposed to solve the matrix completion problem efficiently. Few of the well known methods are Singular Value Thresholding (SVT) [57], Augmented Lagrange Multiplier (ALM) Method [58] and OptSpace [59]. We will now briefly describe the SVT algorithm which we use in this work. SVT algorithm iterates the following steps till stopping criterion is achieved.

$$\begin{aligned} \mathbf{Y}^k &= D_\tau(\mathbf{G}^{k-1}) \\ \mathbf{G}^k &= \mathbf{G}^{k-1} + \delta P_\Omega(\mathbf{M} - \mathbf{Y}^k) \end{aligned} \tag{3.5}$$

where $D_\tau(.)$ is the shrinkage operator which retains the singular values greater than τ of the argument matrix. Thus, the rank of $D_\tau(\mathbf{G})$ is considerably lower than that of matrix $\mathbf{G}$ if many of the singular values of $\mathbf{G}$ fall below τ. Further algorithmic and implementation details of SVT are given in [57]. The parameters τ and δ were experimentally set to 26 and 1.4 respectively. The stopping criterion was set to $1e{-}04$.

In order to evaluate the performance of the proposed robust features in ion-channel signal classification, we use the setup described in [60]. The dataset was randomly permuted to obtain a training set with 30 vectors and a test set with 30 vectors for each ion-channel. The Fourier domain PSD features and the proposed robust features are extracted and presented to a linear SVM for classification. Note that, the signals are denoised [60] prior to extracting the features. Table 3.1 shows the classification rates obtained with the original PSD features and the robust PSD features. Sensitivity and specificity measure the proportion of the correctly identified positives and negatives respectively. It can be clearly observed that the post processing of the PSD features leads to improved classification rates.

The classification setup described in the previous section cannot be directly used for analyte detection. The number of channels inserted in the lipid bilayer varies between experiments and training a classifier for all the possible cases is not possible. To overcome this problem, we proposed to use an array of ion-channel sensors [61] and detect the analyte (Ampicillin) by tracking the relative changes in PSD features among the sensors.

In the four chamber ion-channel sensor array, each chamber holds an ion-channel sensor. Three of the chambers act as the base signals and the other chamber is used as the test signal in which the analyte is introduced. The change in the signal generated in the test chamber can be attributed to: (a) the change in the number of channels inserted in the lipid bilayer and (b) the change in the driving state model due to the presence of an analyte. Support Vector Regression (SVR) is used to estimate the number of channels inserted. Similar to the procedure in [61], the robust PSD features are normalized using the estimate of the number of channels. The PSD features are compared across all chambers using a weighted Euclidean distance (WED) measure. A larger distance measure indicates the presence of the analyte.

Similar to the classification setup, we extract both PSD features and the robust features from each of the signal segments. A detection hit is defined as the case when the WED goes above an empirically obtained threshold. A signal segment corrupted with noise can produce a high WED even when the analyte is absent. Such cases are referred to as false hits. Table 3.2 shows the percentage of false hits obtained using the original PSD features and the stabilized features.

Table 3.1: Classification Performance Using the Original and Stabilized PSD Features for QUB Signals(Linear Kernel).

Transform Domain	**% Classification**	**% Sensitivity**	**% Specificity**
Original	92.1	91.4	92.8
Stabilized	96.6	98.2	95.1

Table 3.2: False Hits Percentage in Detection.

Features	**Percentage**
Original	13.33
Stabilized	1.67

3.7 LRSP Features

In the previous section, we described the Matrix completion method to obtain Robust PSD features. We used a heuristic method to build an incomplete matrix of stacked features and used low rank constraint to complete it. The features thus generated are closer in Euclidean sense for similar channels. This works well when the ion-channels in question are well distinct.

We now consider the case when two classes of ion-channels are quite close to each other in terms of their rate parameters and hence have close PSD features. This situation arises when the analyte is small in size and has minute effect of the closing and opening rate of the channel.We build database consisting of 12 ion-channel classes which form form 4 distinct groups based on their rate parameters. Each group contains 3 classes with similar rate parameters. The first two rows of Table 3.5 show the classification performance using PSD features and Matrix completed PSD when all the 12 classes are considered. It can be observed that the performance drops

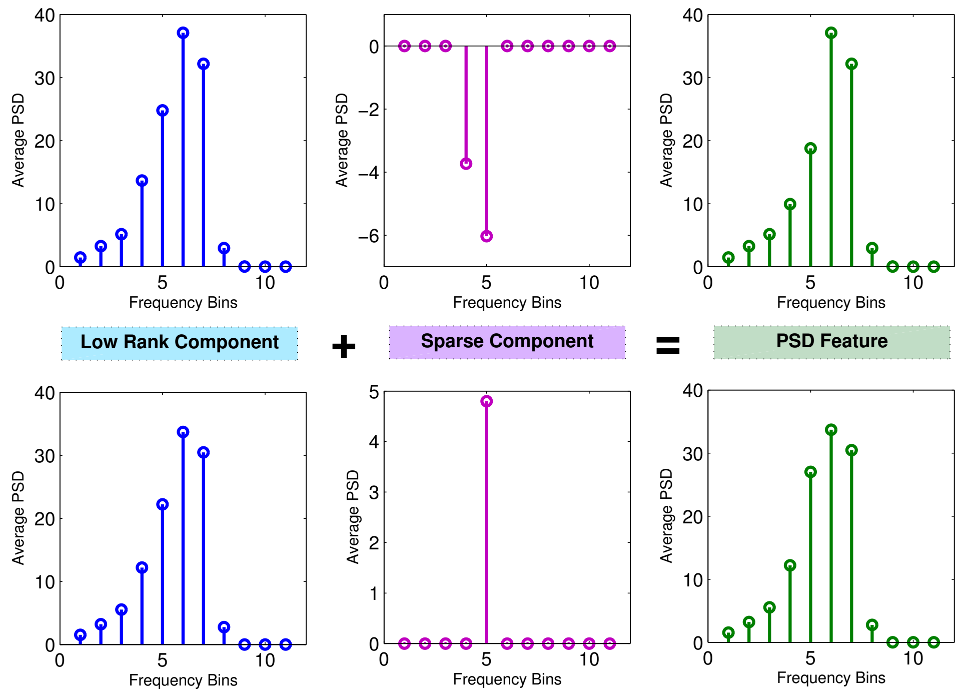


Figure 3.8: The Plots Show the Low Rank and Sparse Components of the PSD Features of Two Similar Classes.

considerably as the similar ion-channels are assigned similar features using low rank approximation and not discriminatory. We propose to decompose the features into two components to capture the group behaviour and also give importance to intra-group variation.

Consider the the features $[f_1, f_2, ..., f_M]$ obtained from M signals obtained from 12 classes of ion-channels. These are stacked in to matrix **F**. We would like to decompose the matrix **F** as following,

$$F = D * A + S + G \tag{3.6}$$

where, **D** is a pre-learned dictionary, **S** is a sparse error matrix and **G** is dense noise matrix. Further, we place a low rank constraint on $D * A$ so that features in same class have similar $D * \alpha$ or low rank or base component. We can form the joint optimization problem as below

$$\begin{aligned} \underset{A,S,L}{\arg\min} \quad & \|A\|_1 + \|S\|_1 + \|L\|_* + \gamma\|F - L - S\|_F^2 \\ & + \lambda\|L - D * A\|_F^2 \end{aligned} \tag{3.7}$$

where **L** is an extra variable added which is low rank and sparse in dictionary **D**.

The optimization problem can be broken down into following sub-problems:

Fix **A** and **S**,

$$L_t = \underset{L}{\arg\min} \|L\|_* + \gamma\|F - L - S_{t-1}\|_F^2 + \lambda\|L - D * A_{t-1}\|_F^2 \tag{3.8}$$

Fix **A** and **L**,

$$S_t = \underset{S}{\arg\min} \|S\|_1 + \gamma\|F - L_t - S\|_F^2 \tag{3.9}$$

Fix **L** and **S**,

$$A_t = \underset{A}{\arg\min} \|A\|_1 + \lambda\|L_t - D * A\|_F^2 \tag{3.10}$$

The dictionary **D** is formed by stacking the normalized PSD features into a matrix.

Each of the sub problems are convex and can be solved by different methods. One iteration of the joint problem involves solving each of the sub problems once with arguments as shown in the equations. We will now show the convergence of the joint problem which is similar to the approach described in GODec paper.

Let the objective value after solving the sub-problems be E_t^1, E_t^2 and E_t^3.

$$E_t^1 = \gamma\|F - L_t - S_{t-1}\|_F^2 + \lambda\|L_t - D * A_{t-1}\|_F^2 \tag{3.11}$$

$$E_t^2 = \gamma\|F - L_t - S_t\|_F^2 \tag{3.12}$$

$$E_t^3 = \lambda\|L_t - D * A_t\|_F^2 \tag{3.13}$$

Without affecting the optimization problem, we can modify the objective values as:

$$E_t^1 = \gamma\|F - L_t - S_{t-1}\|_F^2 + \lambda\|L_t - D * A_{t-1}\|_F^2 \tag{3.14}$$

$$E_t^2 = \gamma\|F - L_t - S_t\|_F^2 + \lambda\|L_t - D * A_{t-1}\|_F^2 \tag{3.15}$$

$$E_t^3 = \gamma\|F - L_t - S_t\|_F^2 + \lambda\|L_t - D * A_t\|_F^2 \tag{3.16}$$

Global optimality of S_t leads to $E_t^1 > E_t^2$ and global optimality of A_t leads to $E_t^2 > E_t^3$. Similarly, global optimality of L_{t+1} leads to $E_t^3 > E_{t+1}^1$. Thus, the objective values of the sub-problems monotonically decrease and hence the joint objective converges to local optimum point. After convergence, the matrices **S**, **L** and **A** give the sparse, low rank and sparse coded low rank version of the PSD features.

The classification performance is tested for PSD features, features obtained after matrix completion and features generated from low rank part and sparse part. We concatenate the low rank and sparse part and call it LS feature. Our proposed feature is the combination of sparse coded low rank part **A** and the sparse part **S**. We call the combined feature as LRSP code and is sparse. Table 3.4 shows the performance of all the feature variants when only the 4 distinct classes are used for training and testing. It can be observed that all the features perform well in this case and LRSP code gives equivalent performance with linear kernel SVM. Table 3.5 shows the classification rates when all the 12 classes are taken into consideration. The performance drops considerably as some channels are very close and are misclassified. The LS and LRSP perform better than others as the discriminative sparse part and representative low rank part are both given importance in these features. Table 3.3 shows the classification rates for one specific group. It can be seen that in this case the sparse part gives best performance and LRSP code gives close to best performance. Thus, to achieve best performance, we employ SVMs at two levels. First, we train an SVM to identify the group of the test ion-channel. The LRSP code is used at this stage. Group specific SVMs are trained to identify the exact ion-channel type.

3.8 Estimation of Number of Channels

In this section, we *discuss* the Support Vector Regression (SVR) method for function regression. We use SVR with auto-correlation wavelet kernel to estimate the number of channels inserted in each chamber. The signals *obtained across the membrane* using the patch clamp technique are corrupted with high noise and often have low signal-to-noise ratio (SNR). Thus, the state transition events can be masked and the parameters extracted can be erroneous. Discrete Wavelet Transform (DWT) based de-noising is performed as a preprocessing step. The advantage of *a* wavelet

Feature Type	**Noise level**		
	0	**0.1**	**0.3**
PSD Features	58.2	55.7	53.1
Matrix Completion on PSD	55.4	54.2	53.8
Sparse Component	**84.7**	**81.1**	**72.5**
Low Rank Component	55.8	55.1	54.8
Low Rank+Sparse Component	83.2	82.8	82.0
Proposed LRSP code	**82.5**	**81.9**	**80.9**

Table 3.3: Classification Accuracies in Percentage for Three Classes of Very Similar Ion-Channel Signals.

Feature Type	**Noise Level**		
	0	**0.1**	**0.3**
PSD Features	94.5	91.2	89.2
Matrix Completion on PSD	96.4	96.0	95.5
Sparse Component	**53.0**	53.8	48.3
Low Rank Component	98.6	97.7	97.3
Low Rank+Sparse Component	**98.2**	97.4	96.9
Proposed LRSP code	**98.0**	97.5	97.1

Table 3.4: Classification Accuracies in Percentage for Four Classes of Ion-Channel Signals which are Distant in the Feature Space.

representation is that it can provide time-frequency localization and help recover *the signals*. We used the biorthogonal wavelet (bior3.7) with 3 levels of decomposition for preprocessing the data. These parameters were obtained using *the* cross-validation method as described in [62]. Soft thresholding was used and the limit was chosen using the universal threshold [63].

The PSD of the signal is computed in Fourier domain using Welch procedure.

Feature Type	**Noise Level**		
	0	**0.1**	**0.3**
PSD Features	58.2	55.7	53.1
Matrix Completion on PSD	55.4	54.2	53.8
Sparse Component	48.6	45.9	43.6
Low Rank Component	55.8	55.1	54.8
Low Rank+Sparse Component	**70.5**	68.2	67.3
Proposed LRSP code	**70.8**	69.0	68.4

Table 3.5: Classification Accuracies in Percentage for Twelve Classes Comprised of Close and Distant Ion-channel Signals in the Feature Space.

This involves windowing the signal and *averaging* the periodograms obtained *from* each window. The PSD feature vector is generated by summing the PSD values in each of the dyadic frequency bins. Dyadic bins are bins spaced in powers of two. Therefore, dyadic binning gives importance to the signal power concentrated in the lower frequencies. The PSD feature values and the average energy are used to estimate the number of *operational* ion-channels.

3.8.1 Support Vector Regression

Let us suppose *that*, $\{f^1,...,f^N\}$ are the feature vectors of the signal segments and $\{y^1,...,y^N\}$ are the number of channels operational in each of them. We want to find a function

$$y = \mathbf{w}^T f + b, \tag{3.17}$$

by minimizing

$$\frac{1}{2}\|\mathbf{w}\|_2 + \frac{\gamma}{l}\sum_{i=1}^{N}(|y^i - \mathbf{w}^T f^i - b| - \epsilon)_+ \tag{3.18}$$

where $\gamma > 0$ is the regularization parameter controlling the bias-variance trade-off and $\epsilon > 0$ is a small number. The optimization problem can be solved by maximizing the following function,

$$-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_*^i - \alpha^i)(\alpha_*^j - \alpha^j)k(f^i, f^j) + \sum_{i=1}^{N}(\alpha_*^i - \alpha^i)y^i$$

subject to

$$\begin{aligned} \sum_{i=1}^{N}(\alpha_*^i - \alpha^i) &= 0, \\ \sum_{i=1}^{N}(\alpha_*^i + \alpha^i) &\leq \gamma\epsilon \\ \alpha^i, \alpha_*^i &\in [0, \gamma]. \end{aligned} \tag{3.19}$$

The estimated function is given by

$$y(f) = \sum_{i=1}^{N}(\alpha_*^i - \alpha^i)k(f, f^i) + b \tag{3.20}$$

where $k(f^i, f^j)$ is the kernel and b is *a* constant dependent on the support vectors obtained during *the* training of SVR.

The auto-correlation wavelet kernel is shown to work well for the regression described in [64]. The kernel is given by the following equation,

$$\begin{aligned} k(f^i, f^j) &= \psi(\frac{f^i - f^j}{a}) \\ \psi(u) &= \cos(1.75)e^{-u^2/2}. \end{aligned} \tag{3.21}$$

Table 3.6 shows the average error in the estimation of number of channels for the 4 simulated chamber signals. The signals have an identical state transition matrix. The PSD feature values, the energy of PSD feature values and their combination were tested as features for SVR. The SVR was trained for signals with 1, 4 and 6 ion-channels operational. The estimates of the number of channels obtained can be rounded off to *the nearest* integer values. Signals with 2, 3, 5 and 10 ion-channels were tested. Each of the four signals were split into 160 frames forming the 640 test signals. The difference between the rounded estimate and the true value of number of channels form the error for each frame. The energy of PSD features produced least error compared to the others. The three types of features produced maximum error for the signal when 10 ion-channels were operational. We observe that *the* average energy of *the* PSD features varies quasi-linearly with the number of channels operational. The same cannot be said about the PSD feature values where we consider the distribution of *power* among bins. Thus, least square fitting can be used instead of SVR while using the average energy of PSD features to estimate the cardinality of channels in *a* signal segment.

3.9 Detection Framework

The number of ion-channels inserted in each chamber is different, but the kinetic model driving the ion-channels is the same before any analyte is introduced. Assuming, each chamber has only one channel inserted, all the chamber signals will have similar feature vectors.

The matching of signals across chambers and detecting relative changes forms the detection framework for an analyte. The two main reasons for change in a chamber signal are: (a), the change in the number of ion-channels inserted in the lipid bilayer of the chamber; (b), the change in the model driving the ion-channel due to the presence of an analyte.

3.9.1 Normalized PSD Features

In our previous work [60], we used the PSD features to classify signals with single ion-channel operational but with different state transition matrices. In our present setting, we initially have four signals with different number of ion-channels operational all having identical state transition matrices. Fig. 3.7 shows the features of the four simulated signals. The average energy of the features $\frac{1}{L}\|f\|_2^2$ was observed to be varying with number of ion-channels inserted.

To match the signals across chambers, we use normalized feature vectors. The feature values are divided by the number of channels inserted. The normalization is in terms of the energy of the feature values. The normalized feature vector of signal with N number of channels have same energy as the signal with one channel. Table 3.7 shows the average energy of the feature values before and after normalization. The chamber signals with same model but different ion-channels have normalized features vectors which are similar in their energy and distribution.

3.9.2 Analyte Detection

We adopt a two step procedure for detection of analyte. The chamber signal containing the analyte is called the test signal and the other chamber signals are base signals. The unnormalized feature vectors are used to detect a change in the test signal. Once a change is detected, the normalized feature vectors are compared to the baseline signal. The distance between the feature vectors are measured using the weighted Euclidean distance,

$$\mathrm{d}(f_i, f_j) = \|\mathrm{diag}(\sigma_1, , \sigma_L)^{-1}(f_i - f_j)\|. \tag{3.22}$$

Chamber Signal	**Feature Values**	**Feature Energy**	**Feature Values+Energy**
A (N=2)	0.050	0.033	0.043
B (N=3)	0.069	0.045	0.063
C (N=5)	0.034	0.029	0.037
D (N=10)	0.087	0.071	0.092

Table 3.6: Errors in the Number of Channel Estimation.

where $\|.\|$ is the euclidean norm and σ_i is the variance of the i^{th} entry in L dimensional feature vector.

The small distance between the normalized feature vectors of the test and base signals indicate channel insertion and no change in the driving model of ion channels. Thus, it shows the absence of *an* analyte. And conversely, large distances indicate change in the kinetic model of ion-channels and show the presence of the analyte. This method was used on the synthetic data to detect simulated analyte and on the experimental data to detect Ampicillin. It was observed that the distance of test signal from the base signals in all the cases, was less than 1 before the introduction of analyte and greater than 10 after the introduction of analyte. Thus, we achieve a sufficient difference in distance between features for detection of analyte. Empirically, it can be said that if there is a tenfold increase in distance increases by 10 times, the ion-channels behavior has changed and an analyte is present.

Chamber Signal	**Before Normalization**	**After Normalization**
A (N=1)	407	407
B (N=2)	1533	383
C (N=3)	3776	419
D (N=10)	55850	496

Table 3.7: Average Energy of PSD Features Before and After Normalization.

Chapter 4

DEEP REPRESENTATIONS FOR ENVIRONMENT SOUND ANALYSIS

Building concise representations for multimedia is crucial for applications ranging from scene recognition, retrieval and personal life-logging systems to field robot navigation. A convenient approach to representing the content of multimedia data is to associate textual tags that can describe the underlying semantics. However, the effectiveness of semantic inference completely relies on the richness of the data, and there is an increasing need to utilize multiple data modalities to understand a physical process. For example, short time events such as an explosion can be effectively captured in the audio data, while the visual data from this event may be incomplete or unavailable due to the limited view or slow response of the camera. Consequently, research efforts have been focused on building tools that are specifically adapted for obtaining inferences from each data modality.

The growing interest in technologies for wearable computing, automatic life logging, and predictive inferences in robotics presents a huge potential for algorithms that characterize environmental sounds. A commonly adopted pipeline for processing such data involves extracting features to succinctly describe them, modeling the statistics of the features, and deriving predictors that reveal the underlying semantics. The quality of the extracted features is intimately tied to the subsequent stages for obtaining inferences.

A common modus operandi for audio feature extraction is to divide the signal into frames, and extract appropriate features from each frame. Features that reveal the Fourier domain characteristics, and those built on psycho-acoustic principles are typically adopted for representing audio data. However, these features do not often work well for environment sounds. This is evident from the unsatisfactory performance of features such as MFCCs (Mel-frequency cepstral coefficients) and psycho-acoustic features such as pitch, loudness, timbre, etc. in environmental sound recognition [3]. This can be attributed to the differences in the characteristics of natural sounds when compared to conventional speech and audio data. For example, unlike speech signals, which can be modeled as a sequence of phonemes, environmental sounds cannot be atomized into a small set of structures. Similarly, environmental sounds often lack any rhythmic patterns found in music data. On the other hand, methods such as Spectral Dynamic Features (SDF) which attempt to model the temporal behavior of the signals have been shown to provide an improved performance [65]. Alternately, learning features directly from data has been a recent and exciting approach when dealing with large-scale data [66]. For example, deep learning techniques, that can infer a hierarchy of features with increasing complexity, have been successful in modeling natural images [67]. This data-driven approach can be particularly suitable to our case since it can effectively capture the multitudes of variation in the data.

We develop a new approach for extracting hierarchical features that can be very effective in characterizing natural environmental sounds. Furthermore, we present an approach based on sparse representations to predict tags for novel test samples using the proposed features. In general, data-driven feature learning methods attempt to build meaningful representations by transforming data to a new domain in which the factors relevant to the task in hand are emphasized. This process of adapting features is computationally intensive for large-scale data, and can be severe when learning is carried out in multiple layers. Furthermore, these methods ensure that the representations are invariant to transformations that are not relevant to the desired application. For example, the spatial location of objects in images might not significantly affect the behavior of an object recognition system, and hence it is

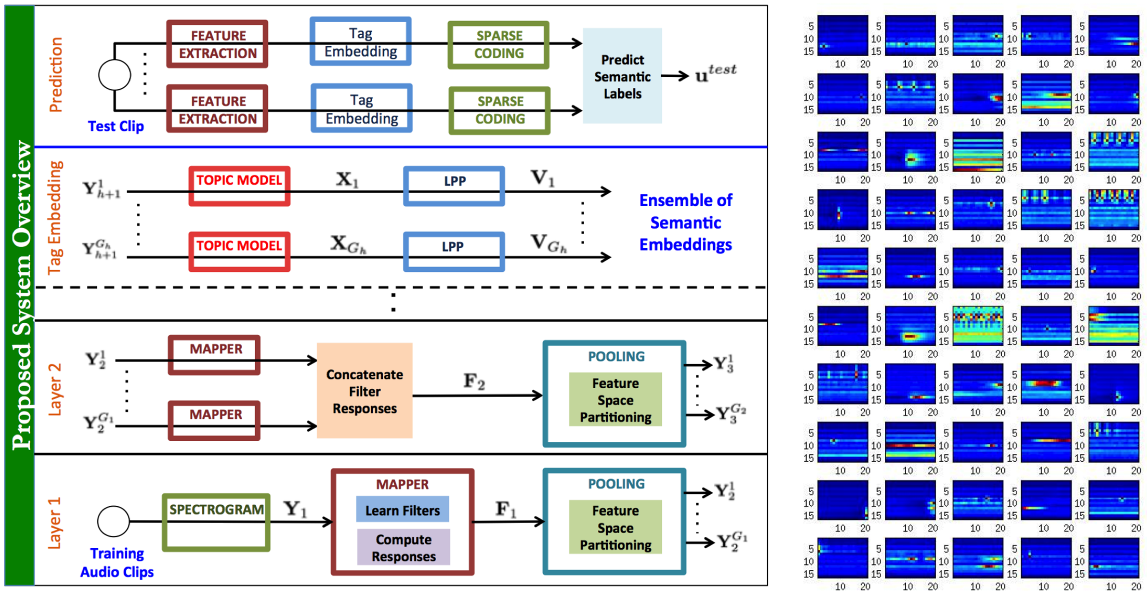


Figure 4.1: (*Left*) An Overview of the Proposed System. Each Layer in the Deep Feature Learning Architecture Involves a *Mapper* Function that Infers Filters and Evaluates Responses, and a *Pooling* Function that Partitions the Feature Space and Aggregates the Responses. By Constructing an Ensemble of Semantic Embeddings, the Underlying Tags can be Effectively Estimated. (*Right*) A Subset of Layer-1 Filters (*Dictionary Atoms*) Learned From the Dyadic Binned Spectrograms of the AASP Challenge Dataset, Where the X- and Y-axes Correspond to Time and Dyadic Frequency Bins, Respectively.

common to perform spatial aggregation (*pooling*) of local features using histograms, spatial pyramids etc. [68]. Existing strategies for pooling (e.g. max-pooling) do not work for environmental sounds, since capturing the signal dynamics, in both time and frequency, is crucial.

The proposed pipeline for feature extraction and semantic inference is illustrated in Figure 4. Instead of using standard audio features, we provide spectrograms as the input to our algorithm. In addition to being effective for modeling environmental sounds, spectrograms enable us to build a more robust pooling strategy. In a deep learning approach, at each layer, filters are inferred from the input, and the sparse filter responses are evaluated [8]. We refer to this two step process as the *Mapper* in Figure 4. We propose a pooling strategy, *correlation-pooling*, which can preserve the temporal dynamics in the resulting representation. To address the challenges pertinent to scalability, we propose to employ a mini-batch, damped K-hyperline clustering algorithm for inferring filters at each layer. The filters learned are atomic in nature and promote distributed representation of the data-points. Furthermore, we perform partitioning of the feature space based on their temporal similarity so that it is computationally tractable to learn filters in the subsequent layers. Interestingly, the partitioning provides a multi-view representation of the data and we exploit this by building an ensemble of topic models using the pooled features.

Finally, to perform semantic predictions, we construct low-dimensional semantic embeddings for the tag vectors, using graphs based on the topic models. For a test data sample, tag prediction is performed using ℓ_1 reconstruction with the low-dimensional embeddings corresponding to each of the topic models in the ensemble. The final predicted tag vector is obtained as the average of the individual predictions. Experiments with challenging datasets show that the proposed features outperform conventional spectral features used for audio classification. Furthermore, partitioning of the feature space to build an ensemble of representations enables the tag prediction algorithm to capture multiple, possibly unrelated, semantics in the data, simultaneously.

4.1 Learning Filters

At each layer of the hierarchy, we learn a set of filters, also referred as *dictionary atoms*, using which the data samples are encoded and subsequently aggregated to generate a succinct representation. We develop a modified version of the K-hyperline clustering algorithm (Section 2.2.2, [69]), referred to as mini-batch damped K-hyperline, to infer the dictionary atoms in each level. The input to the first layer of the hierarchy is the dyadic binned spectrograms of frames collected from all audio clips, denoted by the matrix $\mathbf{Y}_1 \in \mathbb{R}^{M_1 \times N_1}$ matrix. The filter responses in a level ℓ are appropriately aggregated to generate the inputs for the next level. The initial dictionary for a level, $\mathbf{D}_\ell^0$, is obtained as K_ℓ randomly chosen input samples (normalized to unit ℓ_2 norm). To combat the challenges with performing clustering on large datasets, we propose to use only a random subset of the input samples for updating the cluster centers during each iteration of the algorithm. This procedure is summarized in Table 4.1.

4.2 Feature Space Partitioning

Following the dictionary design, we encode all frames using the soft thresholding operator, $\mathbf{f}_{\ell,i} = max\{0, \mathbf{D}_\ell^T \mathbf{y}_{\ell,i} - \alpha\}$, where α is a tuned parameter. Since the filter responses $\{\mathbf{f}_{\ell,i}\}_{i=1}^{N_\ell}$ will be used to construct the input vectors for the subsequent layer, a large K_ℓ can make dictionary learning computationally challenging in the next layer. Consequently, it will be beneficial to partition the feature space into multiple (possibly overlapping) subspaces, and learn an ensemble of filters in the next layer. Though a natural choice is to perform random partitioning of the space, it can be highly suboptimal in describing the temporal dynamics. We propose a greedy partitioning technique that builds multiple $R-$dimensional (possibly overlapping) subspaces, where each subspace is constructed using filters whose responses exhibit similar temporal correlation structures. Let us denote the number of subspaces in layer ℓ as G_ℓ. At a layer ℓ, an audio file indexed by j contains N_ℓ^j frames. Hence, the total number of frames at level ℓ, in the dataset with S clips, can be obtained as $N_\ell = \sum_{j=1}^S N_\ell^j$. The filter responses can now be denoted by the matrix $\mathbf{F}_\ell \in \mathbb{R}^{K_\ell \times N_\ell}$.

In order to partition the feature space, we devise a novel similarity metric for comparing the temporal correlation structures of the filter responses. Let us consider the responses of all N_ℓ^j frames from sound clip j to the filter $\mathbf{d}_{\ell,p}$, denoted by the vector $\mathbf{F}_{\ell}^{p,j} \in \mathbb{R}^{N_{\ell}^{j}}$. We compute $L-$dimensional autocorrelation sequences for $N_{\ell+1}^{j}$ overlapping windows of $\mathbf{F}_{\ell}^{p,j}$. By stacking the correlation matrices $\mathbf{C}_{\ell}^{p,j} \in \mathbb{R}^{L \times N_{\ell+1}^{j}}$, $\forall j = 1 \cdots S$, we construct the overall correlation matrix $\mathbf{C}_{\ell}^{p} \in \mathbb{R}^{L \times N_{\ell+1}}$, where $N_{\ell+1} = \sum_{j=1}^{S} N_{\ell+1}^{j}$. The affinity between the correlation structures $\mathbf{C}_{\ell}^{p}$ and $\mathbf{C}_{\ell}^{r}$ corresponding to the filters $\mathbf{d}_{\ell,p}$ and $\mathbf{d}_{\ell,r}$ is measured as

$$A(p,r) = \frac{\mathbf{Tr}((\mathbf{C}_{\ell}^{p})^{T}\mathbf{C}_{\ell}^{r})}{\mathbf{Tr}((\mathbf{C}_{\ell}^{p})^{T}\mathbf{C}_{\ell}^{p}) + \mathbf{Tr}((\mathbf{C}_{\ell}^{r})^{T}\mathbf{C}_{\ell}^{r})}. \tag{4.1}$$

We adopt a greedy approach to partition the feature space into multiple overlapping subspaces. We begin by randomly choosing a filter and picking $R-1$ filters with the highest similarity measures (from (4.1)) to form a subspace. Following this, we randomly choose another filter, that has not been used in the earlier step, and construct the second subspace. We repeat this process until all filters are included in atleast one of the subspaces.

Since we consider only partially overlapping windows while computing the autocorrelation sequences, the number of frames in a clip j is reduced from N_{ℓ}^{j} to $N_{\ell+1}^{j}$. However, each new frame is now represented by a set of G_{ℓ} vectors of dimensions $R*L$ each, instead of a single K_{ℓ} dimensional response vector. Hence, we perform dimensionality reduction on each of the G_{ℓ} vectors using Random Projections (RP), where the resulting dimension $M_{\ell+1} << R*L$. According to the Johnson-Lindenstrauss lemma [70], RP can approximately preserve isometry in $O(\log(N)/\epsilon^{2})$ dimensions for N samples, where ϵ bounds the approximation error. The use of random projections is applicable here as we will further cluster the aggregated representations for learning filters in layer $\ell+1$. The pooled responses, at level ℓ, can now be denoted as $\{\mathbf{Y}_{\ell+1}^{g} \in \mathbb{R}^{M_{\ell+1} \times N_{\ell+1}}\}_{g=1}^{G_{\ell}}$, which are the inputs to the next layer.

In the $\ell + 1^{th}$ layer, we learn filters, independently, in each of the G_{ℓ} subspaces identified in the previous layer. The feature space partitioning and the pooling operations significantly reduce the computational cost for the mini-batch clustering. Let $K_{\ell+1}^{g}$ denote the number of filters learned in the subspace indexed by g $(g = 1 \cdots G_\ell)$. The total number of filters at level $\ell + 1$ is $K_{\ell+1} = \sum_{g=1}^{G_\ell} K_{\ell+1}^{g}$ and the filter response matrix, $\mathbf{F}_{\ell+1} \in \mathbb{R}^{K_{\ell+1} \times N_{\ell+1}}$, is obtained by stacking responses from all G_ℓ subspaces. We can then repeat the process of feature partitioning and pooling exactly as the previous layer. In our experiments, we employ a $2-$layer architecture, and the output of our feature extraction process are the matrices denoted by $\{\mathbf{Y}_3^g \in \mathbb{R}^{M_3 \times N_3}\}_{g=1}^{G_2}$. Long duration signals can benefit from more number of layers to further reduce the number of frames in the signal. More depth in the architecture can also be achieved by increasing the overlap length in the windowing operation during pooling. This reduces the rate of decrease in the number of frames for each clip across layers.

4.3 Predicting Semantic Labels

In order to generate effective representations for audio clips that will enable semantic predictions, we build an ensemble of Replicated Softmax Models (RSM) topic models on the pooled features. Note that, we infer a topic model for each of the subspaces from the feature partitioning. RSM belongs to the family of undirected, energy-based models known as restricted Boltzmann machines (RBM). The visible unit is modeled as a softmax variable instead of a Bernoulli variable as in a conventional RBM. Further details on this technique and its convergence properties can be found in [71]. Since RSM requires a histogram of words as input, we construct a bag-of-words model based histogram for the audio clips in each of the subspaces. Each of these models can be interpreted of as a local topic model based on a limited set of words, and can reliably predict a subset of tags or categories. Such an ensemble can provide a richer and robust information, and we show that it is particularly suitable for environment sounds which can manifest from numerous categories in real scenarios.

4.3.1 Ensemble Tag Embedding

In order to explore the relationships between the inferred topic models and the ground truth tag vectors (in training data), we compute low-dimensional semantic embeddings for the tag vectors based on graphs constructed using the corresponding topic models. In particular, we assume that the topic features follow a union-of-subspaces model, wherein samples in a subspace can be effectively reconstructed using other samples in that subspace, and construct graphs based on sparse representations. Each of the topic models in the ensemble provides a different semantic embedding that improves the predictability of some tags more than others. Furthermore, we assume that the tag vector for any audio clip is sparse, and that the ℓ_1 reconstruction of a test sample in both the topic feature space and the embedded tag space are similar.

We begin by considering one topic model in the ensemble, and let us denote the set of topic features using the matrix $\mathbf{X} \in \mathbb{R}^{P \times S}$, where P indicates number of topics. The tag (label) vectors for all sound clips are stored in the matrix $\mathbf{U} \in \mathbb{R}^{T \times S}$, where T denotes the total number of labels in the collection used to describe the training set. For the j^{th} clip, the entry $\mathbf{U}_{i,j}$ is set to 1 if the i^{th} tag is associated with that clip. We are interested in computing a linear projection matrix $\mathbf{V} \in \mathbb{R}^{T \times d}$, where $d \ll T$, which will result in similar low-dimensional embeddings for tag vectors with similar topic distributions. The projection directions are estimated using an approach similar to Locality Preserving Projections (LPP) [72], with a sparse coding graph obtained from the topic features. The edge weights in the graph correspond to sparse codes obtained for each feature vector using all other features as the dictionary. This approach is found to be more effective than Euclidean distance based neighborhood in identifying the semantic relationships.

Sparse coding of a topic feature $\mathbf{x}_i$ can be performed as

$$\min_{\boldsymbol{\alpha}_i} \|\mathbf{x}_i - \mathbf{B}\boldsymbol{\alpha}_i\| + \lambda\|\boldsymbol{\alpha}_i\|_1 \tag{4.2}$$

where the dictionary $\mathbf{B} \in \mathbb{R}^{P\times S-1}$ is designed using all features except $\mathbf{x}_i$ and $\|.\|_1$ denotes the ℓ_1 norm. Denoting the sparse coefficient matrix by $\mathbf{A} \in \mathbb{R}^{S\times S}$, we construct the adjacency matrix for the graph as $\mathbf{W} = |\mathbf{A}| + |\mathbf{A}^T|$.

Given the affinity matrix $\mathbf{W} \in \mathbb{R}^{S\times S}$, we compute the degree matrix $\mathbf{D}$ with each diagonal element containing the sum of the corresponding row or column of $\mathbf{W}$. The d projection directions for LPP are then computed by optimizing

$$\min_{\mathbf{V}} \sum_{i,j=1}^{S} \|\mathbf{V}^T\mathbf{u}_i - \mathbf{V}^T\mathbf{u}_j\|_2^2 w_{ij} \text{ s.t. } \sum_{i=1}^{S} \|\mathbf{V}^T\mathbf{u}_i\|_2^2 d_{ii} = 1.$$

where w_{ij} and d_{ii} are the corresponding elements of the affinity and the degree matrices. This optimization ensures that the embedding preserves the structure defined by the sparse coding based graph. Defining the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$, this optimization problem can be rewritten as

$$\min_{\text{trace}(\mathbf{V}^T\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{V})=\mathbf{I}} \text{trace}(\mathbf{V}^T\mathbf{U}\mathbf{L}\mathbf{U}^T\mathbf{V}). \tag{4.3}$$

and this can be solved using generalized eigen-value decomposition.

4.3.2 Tag Reconstruction

Assuming that our deep architecture for feature extraction contains h levels, for a test audio clip, we extract the features, and construct the G_h (size of the ensemble) topic distribution vectors $\{\mathbf{z}_g\}_{g=1}^{G_h}$. The goal here is to predict the tag vector for the test data using the topic features. For a test clip, we compute the similarity with training clips in terms of features using reconstruction coefficients as

$$\boldsymbol{\beta}_g = \underset{\boldsymbol{\beta}}{\text{argmin}} \|\mathbf{z}_g - \mathbf{X}_g\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1. \tag{4.4}$$

Let us denote the low-dimensional embeddings of the training tags corresponding to the g^{th} topic model as $\mathbf{E}_g = \mathbf{V}_g^T \mathbf{U}$. We perform out-of-sample extension for the test sample by estimating its embedding in the semantic space as, $\mathbf{e}_g^{test} = \mathbf{E}_g \boldsymbol{\beta}_g$. Since we know that the tag vector $\mathbf{u}_g^{test}$ is sparse, we solve the following inverse problem to predict the tag vector:

$$\mathbf{u}_g^{test} = \underset{\mathbf{u}}{\operatorname{argmin}} \|\mathbf{e}_g^{test} - \mathbf{V}_g \mathbf{u}\|_2^2 + \lambda \|\mathbf{u}\|_1 \tag{4.5}$$

Finally, we average the predictions from all models in the ensemble to estimate the tag vector: $\mathbf{u}^{test} = \frac{1}{G_h} \sum_{g=1}^{G_h} \mathbf{u}_g^{test}$.

4.4 Datasets and Results

First dataset was created for the recently organised IEEE AASP D-CASE scene classification challenge [1]. The dataset consists of 30 seconds long audio clips, recorded using a Soundman binaural microphone (PCM 44.1kHz 16 bit). The publicly available development dataset contains 100 clips with 10 samples from each of the following 10 classes: *bus, busy street, office, open market, park, quiet street, restaurant, supermarket, tube, tubestation.* It has an equal balance of indoor and outdoor scenes. We achieve a recognition accuracy of 71%, averaged over 100 runs, when compared to the baseline of 56% obtained using MFCCs. Figure 4.2 shows the recognition accuracies of each of the RSM topic models, and that of a majority vote classifier combining all of them. Table 4.2 shows the confusion matrix summed over 100 runs. The performance can be improved by using a larger dataset to learn filters compared to this dataset. Table 4.3 compares our performance to a few related entries from the challenge. The details on the methods used for comparison can be found in [1].

We will demonstrate tag prediction on the recently released large dataset containing 7960 audio files (10s long) [73]. These are mostly field recordings, and the

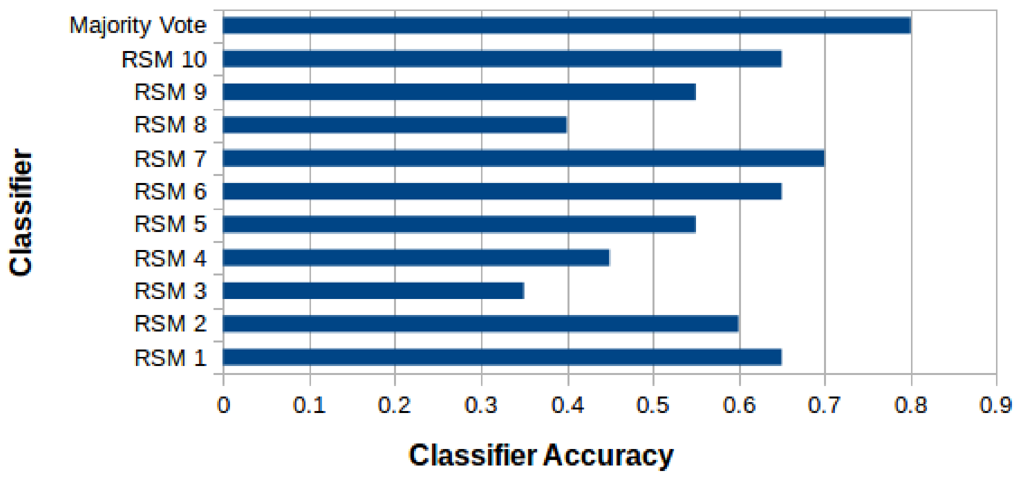


Figure 4.2: Effect of Using An Ensemble of Semantic Embeddings on the Classification Performance for a Particular Run on the AASP Challenge Dataset. A Majority Vote Classifier Outperforms the Individual Classifiers Learned Using the Corresponding RSM Topic Models.

total number of tags in the dataset is 7729. The tag *field-recording* was used as the search term to build the database from the freesound website. We use linear SVM for classification, and we report the performance for a subset of tags that have sufficient number of examples in the dataset. The accuracies and f1-scores are illustrated in in Figure 4.3. The tag prediction behavior of our system in the total space of 7729 tags is illustrated in Table 4.4. It can be observed that the semantically similar tags have been clustered in subspaces.

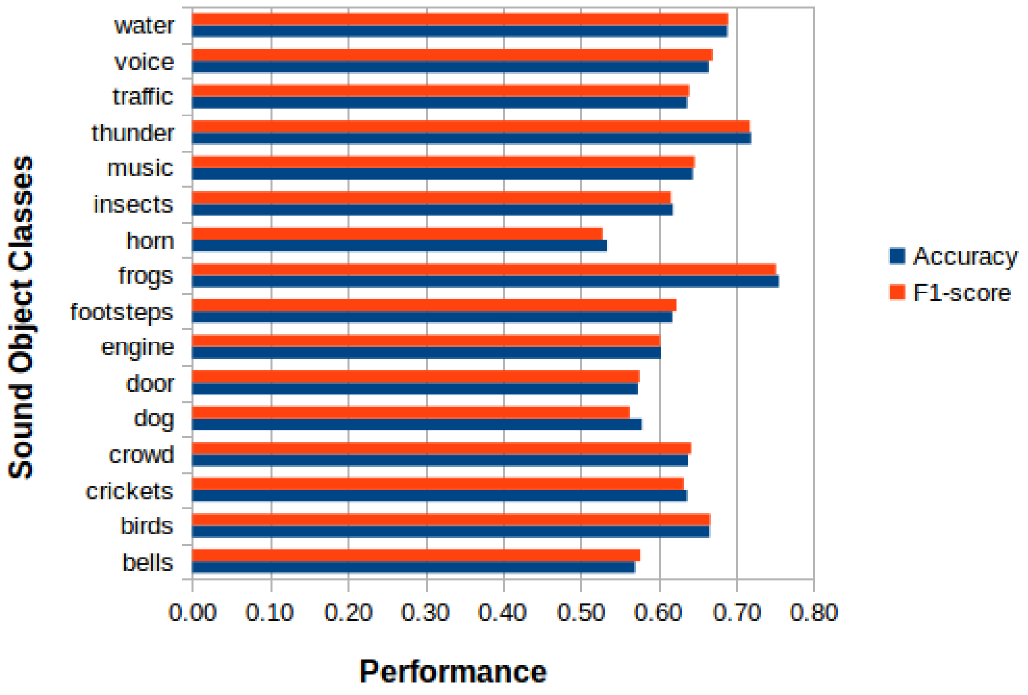


Figure 4.3: Prediction Performance (Accuracy and F1-score) of the Proposed Approach on a Subset of the Freefield1010 Dataset. This Subset was Chosen Such That It Contained a Sufficient Number of Examples for Each Tag.

Table 4.1: Mini-batch Damped K-hyperline Clustering for Inferring Filters in Each Layer of the Hierarchy.

```
Input:
$\mathbf{Y}_\ell$ - Input matrix of size $M_\ell \times N_\ell$
$K_\ell$ - Desired number of clusters
$T$ - Number of iterations
$B$ - Size of the mini-batch

Algorithm:
For $t = 1$ to $T$
  Draw a random subset $\mathbf{Y}_\ell^t = [\mathbf{y}_i^t]_{i=1}^B$ from the input matrix $\mathbf{Y}_\ell$.
  Loop for $G$ iterations
    For $i = 1$ to $B$
      - Compute $\mathbf{h}_i = (\mathbf{D}_\ell^{t-1})^T \mathbf{y}_i^t$.
      - Compute $j = \mathbf{argmax}_j(h_{ij})$, where $j = 1 \cdots K_\ell$.
      - Set $\mathbf{u}_j = h_{ij}\mathbf{y}_i + \mathbf{u}_j$.
      - Set $v_j = (h_{ij})^2 + v_j$.
    end
    For $i = 1$ to $K_\ell$
      - Set $\mathbf{d}_{\ell,j}^t = \frac{\mathbf{u}_j}{v_j} + \mathbf{d}_{\ell,j}^{t-1}$.
      - Normalize $\mathbf{d}_{\ell,j}^t$ to unit $\ell_2$ norm.
    end
  end
end
```

Table 4.2: Confusion Matrix for the AASP Scene Classification Dataset Averaged Over 100 Runs.

Estimated:	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
A (bus)	**82**	0	11	2	0	0	2	0	2	0
B (busy-street)	0	**100**	0	0	0	0	0	0	0	0
C (office)	5	0	**91**	0	0	1	1	2	0	0
D (open-market)	0	2	0	**87**	0	0	8	2	0	2
E (park)	10	10	36	0	**24**	12	6	0	0	1
F (quiet-street)	2	3	4	8	4	**72**	2	1	2	1
G (restaurant)	6	2	0	17	1	0	**66**	5	0	2
H (supermarket)	18	13	3	5	2	3	15	**20**	1	20
I (tube)	16	12	0	1	0	6	2	1	**56**	8
J (tube-station)	0	12	0	4	1	0	4	9	1	**68**

Table 4.3: Recognition Performance of the Proposed Approach on the Publicly Available AASP Development Dataset in Comparison to a Few Relevant Entries in the AASP Scene Classification Challenge [1].

Method	**Average Accuracy (%)**
Baseline MFCC + GMM	52
Cochleogram + Tonelikeness	55
Spectral and Temporal and Spatial Features	69
SparseRBM + Uniform Max-pooling	68
SparseRBM + Selective Max-pooling	75
i-vector analysis of MFCC + pLDA	65
Spectral and Temporal features + HMM	72
RQA features + MFCC	71
Our Method	71

Table 4.4: Illustration of Tag Prediction Behavior in the Subspaces Found. Table Shows the Top Predicted Tags in a Few Subspaces for a Particular Clip. Related Concepts are Clustered in a Subspace.

Ground Truth	street-sounds, traffic-noise, sidewalk-conversations, city-streets
SubSpace 1	male, voice, female, vocal, speech
SubSpace 2	traffic, driving, cars, roar, motorbike
SubSpace 3	city, voices, wind, male, station
SubSpace 4	voices, children, screeching, steps, child
SubSpace 5	racing, radio, accelerating, motor, car

Chapter 5

SPARSE FEATURE LEARNING AT SCALE

Performing fine-grain parallelization of the algorithms [74] can also reduce computation times. The inexpensive Graphical Processing Units (GPUs) [75] are a popular option for parallel processing.We implement the Orthogonal Matching Pursuit (OMP) algorithm to evaluate the sparse codes of thousands of signals in parallel using a learned dictionary. This implementation can be directly adapted to wavelet based de-noising using fixed dictionary. We present strategies to save memory and to achieve maximum speed-up. We present results that compare our implementation (GPU-OMP) to other CPU implementations.

5.1 Architecture

Using Graphic Processing Units (GPUs) as Stream Processors has enabled GPUs to perform parallel computations. Typically, a system comprises both a CPU and a GPU. On receiving the control and data from the CPU, GPU performs the computations and returns the processed data back to the CPU. Figure 5.1 illustrates the architecture of a GPU. A typical GPU has a series of Multiprocessors (MP), and each multiprocessor contains 8 Scalar Processors (SP). The memory bank of a MP can be accessed by all its SPs and a large global memory is shared across all the MPs. The scalar processor in turn has its own registers. Access to the registers and the shared bank in a MP is the fastest, while accessing the global memory is fast when the memory is coalesced. The slowest is the transfer between the CPU-RAM and the GPU-global memory.

CUDA (Compute Unified Device Architecture) is a parallel computing architec-

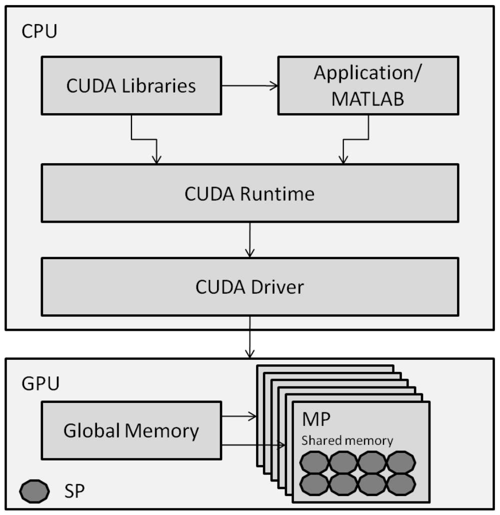


Figure 5.1: The NVIDIA GPU Architecture (NVIDIA and CUDA Are Registered Trademarks of NVIDIA Corporation).

ture developed by NVIDIA to implement algorithms in their GPUs. We use the C interface for CUDA and employ *nvmex* to generate *mex* files that can be invoked in MATLAB. A typical CUDA program consists of a host code and a device code, where CPU and GPU are the host and the device respectively. The host performs the non-parallel computations and passes data to the global memory in the GPU and launches a kernel. The kernel executes the computations using parallel threads on the SPs. The threads are grouped into blocks and blocks are grouped further into grids. The host code receives the processed data and returns to the MATLAB environment.

5.2 Dictionary Based Coding for Signals on GPU

The general problem of sparse coding can be stated as

$$\hat{\mathbf{a}} = \underset{\mathbf{a}}{\operatorname{argmin}} \|\mathbf{a}\|_0 \text{ s.t. } \|\mathbf{y} - \mathbf{\Psi}\mathbf{a}\|_2^2 \leq \epsilon, \tag{5.1}$$

where $\mathbf{y} \in \mathbb{R}^M$ is the data to be represented, $\mathbf{\Psi} \in \mathbb{R}^{M \times K}$ is the dictionary, $\mathbf{a} \in \mathbb{R}^K$ is the sparse coefficient vector, $\|.\|_0$ indicates the ℓ_0 norm, $\|.\|_2$ denotes the ℓ_2 norm and ϵ is the error goal for the representation.

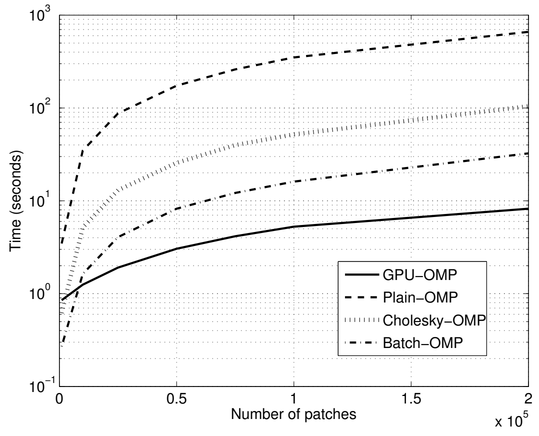


Figure 5.2: OMP Running Times - Total Execution Time to Perform OMP on Different Number of 64 Length Signal.

We use the NVIDIA GTX 460X GPU with 336 cores, installed on an AMD quad core machine. Since the transfer rate between CPU memory and global memory of a GPU is very slow, we transfer the signals to the GPU global memory in large chunks, limited by the GPU global memory size. The primary goal of our GPU implementation is to perform OMP on each signal in parallel using only a single CUDA kernel. The data transfer between the global memory of the GPU and the shared memory bank of a block causes additional overhead. We can minimize this transfer by reducing the number of kernel launches. Each signal and the dictionary are passed to a thread at the kernel launch. The sparse code is obtained from each thread and stored in the global memory of the GPU which is then finally transferred back to MATLAB.

The bottleneck in performing OMP using a single kernel is the limited availability of fast memory access (16 KB of shared memory bank) to each thread to compute the sparse code. Hence, there is a need to reduce the computational complexity and storage requirements of the OMP implementation. It can be observed that the computational overhead is in evaluation of the coefficients using a least square procedure. In the l^{th} step of the OMP algorithm, let $\mathbf{\Psi}_l \in \mathbb{R}^{M \times l}$ denote the set of l dictionary atoms chosen for the representation. To compute the sparse code, we need to solve $\min_{\mathbf{a}} \|\mathbf{y}_n - \boldsymbol{\Psi}_l \mathbf{a}\|_2^2$. This can be simplified by performing Gram-Schmidt orthogonalization of $\boldsymbol{\Psi}_l$ to compute $\mathbf{Q}$. The least squares problem can now be solved as $\|\hat{\mathbf{y}}_n - \hat{\boldsymbol{\Psi}}_l \mathbf{a}\|_2^2$, where $\hat{\boldsymbol{\Psi}}_l = \mathbf{Q}^T \boldsymbol{\Psi}_l$ and $\hat{\mathbf{y}}_n = \mathbf{Q}^T \mathbf{y}_n$. Thus, the size of the matrix whose inverse needs to be found at each step of OMP reduces from $M \times l$ ($\boldsymbol{\Psi}_l$) to $l \times l$ ($\hat{\boldsymbol{\Psi}}_l$). Finally, we use the iterative Newton's method to obtain the least squares solution [76]. The convergence to the true inverse is guaranteed when the initial estimate of the inverse of $\hat{\boldsymbol{\Psi}}_l$ is $\alpha \hat{\boldsymbol{\Psi}}_l^T$, where α is positive and sufficiently small. A commonly used initial estimate is

$$\boldsymbol{\Lambda}_0 = \frac{\hat{\boldsymbol{\Psi}}_l^T}{\|\hat{\boldsymbol{\Psi}}_l\|_1 \|\hat{\boldsymbol{\Psi}}_l\|_\infty} \tag{5.2}$$

where $\|\hat{\boldsymbol{\Psi}}_l\|_1$ and $\|\hat{\boldsymbol{\Psi}}_l\|_\infty$ are the maximum absolute row and column sum norms respectively. Figure 5.2 shows the comparison of the time taken by the Plain-OMP, Cholesky decomposition based OMP, Batch-OMP [77] and the GPU-OMP on different number of signals. The first three algorithms are implemented on the CPU. The 5-sparse representation of each 64 length signal is found using a dictionary of size 64×1024 . As it can be seen, when the number of signals is very less, the CPU performs better than then GPU. This is because, the number of signals is much lower than the total number of cores and computational overhead of data transfer between the CPU memory and the GPU is higher. With the increase in the number of signals, using a GPU achieves a significant speed-up in sparse code computations. Further speed-up can be achieved by using multiple GPUs.

5.3 Image Retrieval

We demonstrate the use of the proposed implementation in building features based on the sparse codes to (a) perform exact image search, which has been used in image monitoring for broadcast media and photo industry [78], [79], and (b) retrieve visually

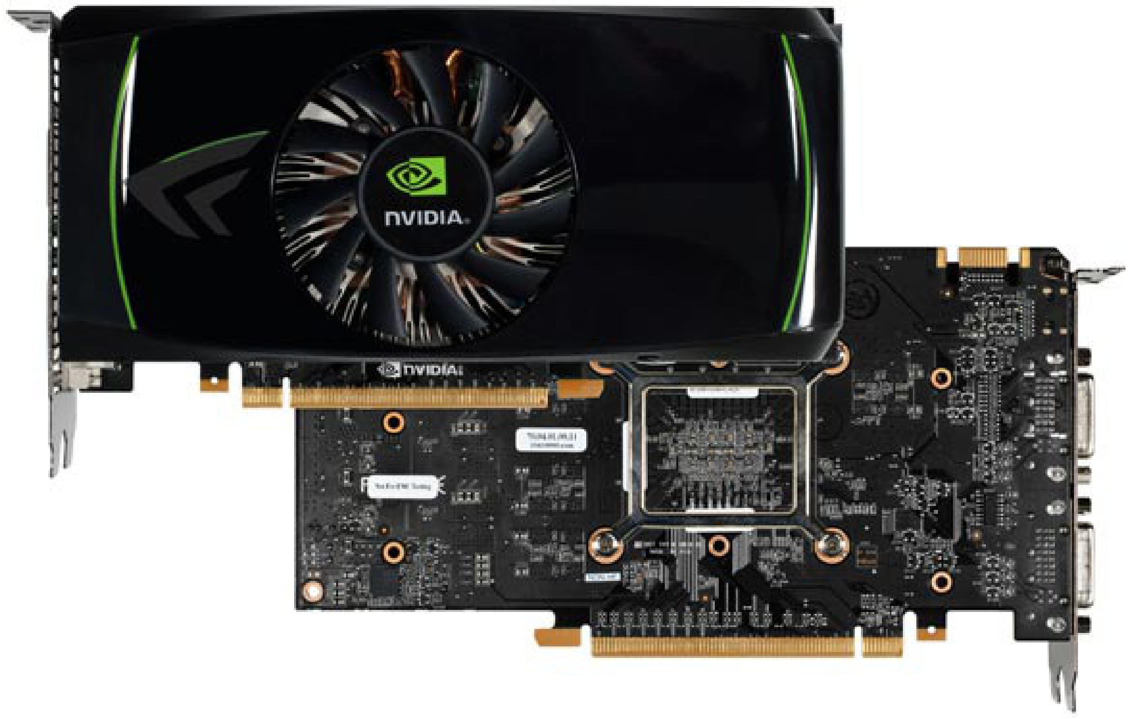


Figure 5.3: The NVIDIA GeForce GTX 460 GPU (NVIDIA is a Registered Trademark of NVIDIA Corporation).

similar images from the database. In order to facilitate exact image search, we can generate compact representations for images that can be easily stored and compared. We propose to perform exact search using a binary feature vector for each image generated from the sparse coefficients of its patches. In order to extract visually similar images we use local descriptors, invariant to commonly occurring transformations, that are coded using a dictionary and aggregated to generate image-level features. Retrieval results using images from the online *Flickr* database indicate that aggregated features yields better performance when compared to just using user tags.

5.3.1 Performing an Exact Match

Employing a nearest neighbor approach to perform an exact match in a large database and retrieving the associated information poses two important challenges. It is imperative for the algorithm to access all images to run the comparison. However the sizes of these databases are often overwhelming and hence modern image storage systems resort to using multiple servers. However, it is computationally expensive to retrieve images from several servers. This can be addressed by generating simple and unique image-level codes that allow for efficient representation and storage. The other challenge is to perform a fast comparison based on a distance metric. We simplify this by designing binary feature vectors for the images and using the Hamming distance for comparison.

The image-level feature is generated by first computing an $S-$sparse representation for the image patches using the GPU-OMP implementation. We then create an intermediate feature vector of length K from the matrix $\mathbf{A}$, using the absolute maximum of each row. This operation, referred to as max pooling, provides discriminative image information disregarding the spatial locality of the patches. In order to create binary feature vectors for images, we need to estimate the quantization threshold for the intermediate feature vectors. This is computed using a histogram of intermediate feature values for a subset of images from the database. The threshold is the value at which the histogram can be divided into approximately two equal regions. Finally, for each image we generate a $K \times 1$ binary feature vector by setting all the intermediate feature vector values greater than the threshold to 1 and the rest to 0. During retrieval, all the binary feature vectors from the image database are loaded into the GPU global memory. The GPU-OMP kernel and the GPU-Hamming-distance kernel are used to compute the binary feature vector for the test image and identify the exact match image from the database.

5.3.2 Obtaining Visually Similar Images

The standard online databases such as *Flickr* retrieve images based on user-defined tags. Since, user tags do not always hold relevance to the underlying content, user experience in image search is often compromised. Hence, there has been lot of interest in designing efficient content based search systems. Several image-specific features/descriptors have been used in order to retrieve similar images given a desired target image. However, performing feature based image retrieval is extremely challenging on large datasets.

We demonstrate a simple setup which initially runs a search based on user tags followed by an additional level of search based on content. Each image is divided into patches of size 24×24 with a grid spacing of 8 and one SIFT descriptor is extracted per patch. Global dictionaries are learned using a subset of the descriptors and we compute a S-sparse code for each SIFT descriptor using the GPU OMP implementation. The image-level feature is generated by counting the number of non-zero coefficients corresponding to each dictionary atom and normalizing it using the total number of non-zero coefficients, at different spatial scales. Finally, a nearest neighbor approach is used to retrieve the relevant images. Figure 5.4 shows results obtained with the proposed scheme in comparison to using a tag-based search. In each case, we use the first 250 results obtained with tag-based search [80] to perform feature based search. As it can be observed, the proposed scheme retrieves images that are very similar to the target image.

5.4 Image Annotation

Textual information or tags can be a useful meta-data for images. Large scale image retrieval becomes feasible by associating images with tags. Many a time users submit a textual query for image retrieval and hence tagging plays an important role in identifying images most relevant to the query. Furthermore, categorizing images and videos semantically can benefit from tag information. Human annotation is highly subjective and imprecise. Based on individual perception, the same image can

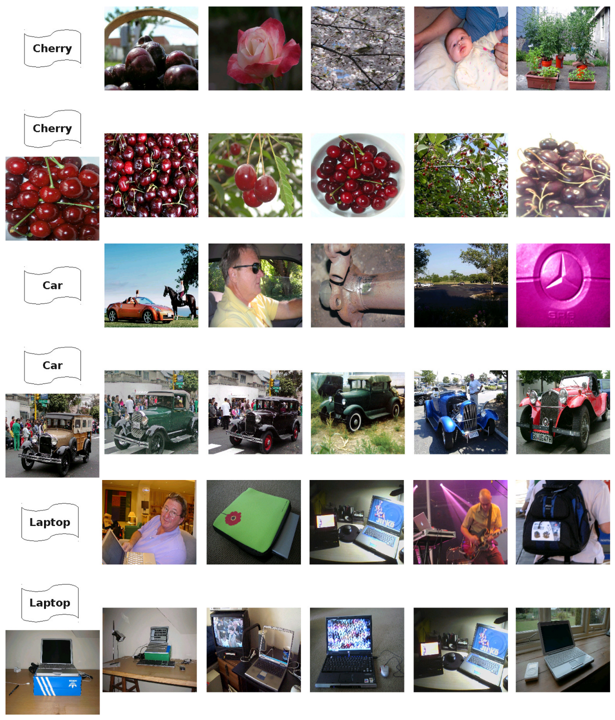


Figure 5.4: Image Retrieval: In Each Case, the First Row Shows The First 5 Results Obtained Using Tag-Based Search in *Flickr* and the Second Row Shows the Top 5 Results Obtained by Performing an Additional Level of Search by Providing a Target Image.

be associated with different tags. Hence, this tag annotation is variable and may not often convey the true semantic meaning of the image. It becomes a herculean task for humans to annotate images if the image database is large. The goal of automated image annotation is to assign suitable tags (or labels) to a given image, based on its content.

5.4.1 Related Work

The choice of suitable image descriptor is crucial for a variety of visual recognition problems. The use of multiple visual descriptors can lead to improved performance in image classification and tag annotation tasks. Since, combining a diverse set of descriptors is not straighforward, kernel methods have been successfully used for object classification based on multiple features [81]. The performance is sensitive to the dataset used for training and cannot be directly extended to image annotation. Understanding the contribution of each feature to image annotation is important, while combining the features. Furthermore, with image descriptors such as histograms, exploiting both the high-level correlations (comparing the whole histograms) and low-level correlations (comparing the individual bins in the histogram) might lead to improved annotation systems. Based on this intuition, the authors in [82] place a sparsity prior on the features and exploit their group clustering characteristics for effective feature selection. As a result, each bin in the histogram is assigned a different weight based on its contribution to the combined representation. We investigate the use of energy based Restricted Boltzman Machines (RBMs) to exploit the correlation among features at low-level bins and high level histograms. RBMs can learn the probability distribution of the input data and have been employed in a deep learning framework for dimensionality reduction [83].

It has been shown that meaningful low-dimensional representations can be obtained for the image features, particularly beneficial for complexity reduction in image retrieval applications [84]. The correlations between the tag and the features is highly non-linear and thus an RBM cannot be directly used on the raw tag data and features. There has been several attempts to obtain semantically meaningful embeddings for features, by directly incorporating the ground-truth tags. In [85], authors compute an embedding for tags and features in a latent semantic space using canonical correlation analysis (CCA). In [86] the embedding is estimated using sparse coding based graphs obtained from the tags. The semantically meaningful low-dimensional embedding can then be used to effectively predict tags for a test image, with suitable visual features.

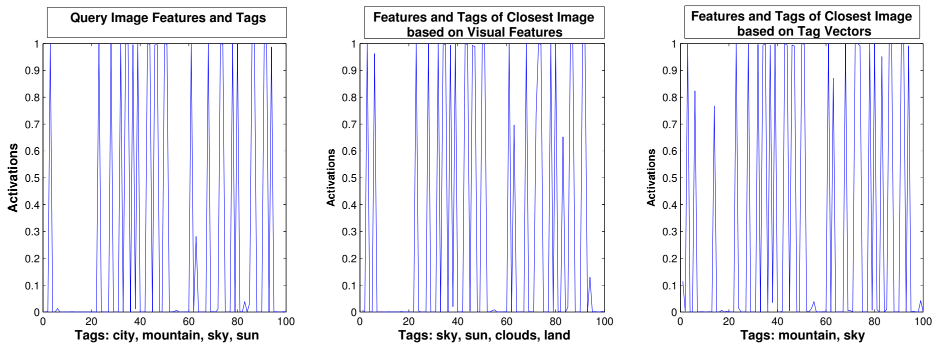


Figure 5.5: Figure Shows the Features and Tags of: (Left) Query Image, (Right) Image With Closest Visual Feature and (Right) Image With Closest Tag Vector.

5.4.2 Our Approach

We propose to perform automatic image annotation using sparse coding and nearest neighbour based methods in a semantic space. The semantic space is a low-dimensional linear subspace obtained from a sparse coding based similarity graph of training examples. The feature vector for each training example is obtained by fusing multiple descriptors using deep belief networks (DBNs) and the graph is computed from the unified feature-tag data obtained from training images. Practical image annotation algorithms have trade-off in their precision-recall characteristics. Therefore, we propose two different modifications to the proposed algorithm using random ensembles of tags or training data, in order to achieve higher precision or higher recall, without significantly compromising the other measure. Results show that sparse coding based tag prediction performs better in comparison to nearest-neighbour approach. Furthermore, the effectiveness of the randomized extensions to the algorithm in improving precision or recall is demonstrated.

5.5 Feature Extraction

We can extract hierarchical features by stacking layers of RBMs, known as a Deep Belief Network (DBN), where, output of one layer acts as input to the another. Each layer progressively captures higher order correlation between the data.

In our set-up, the $37,000$ dimensional input feature is obtained by stacking 15 different visual descriptors as described in [87]. Learning a network with these many visible units is computationally intractable, and would also require a large training set. Therefore, we use a greedy sequential method to generate features while using all the 15 descriptors. The input feature vector for each image is divided into 1000 dimensional chunks, and DBNs (number of nodes in each layer: 1000-1200-500-100) are learned sequentially. The DBN from the previous round is used to initialize the DBN in the current round. and the activations from from all DBNs are stacked to form a low dimension feature. The process is repeated for multiple passes over the whole $37,000$ dimensions to further refine the feature. Figure 5.5 shows the activations for the last round of the DBN for a sample query image. It can be observed from the plot that the features capture the visual correlations between images (leftmost and middle plots). But the images with similar tags (leftmost and rightmost) have degree of a dissimilarity in their visual features. This motivates us to develop a joint low-dimensional embedding of the visual features and tags to create a representation which directly capture correlations among the visual features as well as tags.

5.6 Computing Low-Dimensional Embeddings

Using the DBN features, we propose a procedure to compute a low-dimensional linear embedding for each image that incorporate the supervisory tag information. These reduced dimensional embeddings make it computationally feasible to perform image annotation using sparse coding on a large image database. The semantic information in the images is also encoded in the embedding since it is computed using both features and tags. The appropriate tags for a test image can then be predicted based on its relation to semantically similar images. As noted earlier, we assume that each image is described using more than one label, and we are provided with a training set with relevant ground truth labels.

Let us denote the set of image-level DBN features using the matrix $\mathbf{X} \in \mathbb{R}^{M \times T}$, where T indicates the total number of training images. The tag (label) matrix is denoted by $\mathbf{U} \in \mathbb{R}^{P \times T}$, where P denotes the total number of labels in the collection used to describe the training set. For the i^{th} image, the entry $\mathbf{U}_{j,i}$ is set to 1 if the j^{th} appropriately describes that image. We are interested in computing a projection matrix $V \in \mathbb{R}^{M \times d}$, where $d \ll M$, which will result in highly similar low-dimensional embeddings for images with similar tags. The multi label sparse coding (MLSC) technique described in [86] is one such method for creating a semantically meaningful embedding. In the MLSC approach, the embedding is estimated using the sparse coding graph obtained with the tags. The edge weights in the graph correspond to sparse codes obtained for each tag vector using all other tag vectors as the dictionary. These sparse coefficients are found to be more effective than Euclidean distance based neighbourhood in identifying the semantic relationships. Sparse coding based graphs are also more robust to noisy tags and adapt well to non-uniform sampling of the tag space. Based on the intuition that the similar images in the feature space will have similar tags, the authors in [86] create an embedding for the features such that the relations between the tag vectors are preserved. Such an approach has two main disadvantages: (a) its performance will be affected when there are insufficient number of tags or when there is severe noise in the tags, and (b) the computational complexity of obtaining the sparse coding graph is very high. In order to make the embedding even more robust to partial or noisy tags, we propose to explicitly model the correlation between the features and the tags. This is achieved by obtaining a sparse coding graph for the features and tag vectors together, using a learned dictionary with K elements. By fixing $K \ll T$, we can also avoid the high computational complexity usually associated with obtaining sparse coding graphs.

The proposed sparse coding and dictionary learning approach for the unified feature-tag space is denoted as

$$\min_{\mathbf{B},\mathbf{A}} \left\| \begin{bmatrix} \mathbf{X} \\ \mathbf{U} \end{bmatrix} - \mathbf{B}\mathbf{A} \right\|_F + \lambda \sum_i \|\mathbf{a}_i\|_1 \text{ s.t. } \forall k, \|\mathbf{b}_k\|_2^2 \leq 1, \tag{5.3}$$

where the dictionary $\mathbf{B} \in \mathbb{R}^{(M+P)\times K}$ models both the features and tag vectors of each training example using the same set of coefficients. $\mathbf{a}_i$ is the coefficient vector and the collection of T coefficients is denoted by the matrix $\mathbf{A} \in \mathbb{R}^{K\times T}$. Sparse codes of two images obtained from this model will be correlated only if their features and tag vectors are similar. The dictionary $\mathbf{B}$ can be obtained using sophisticated techniques, but we observed from our experiments that a simple K-means approach provided a sufficiently good dictionary for this application. Following this, we construct the weighted graph adjacency matrix, using the sparse codes obtained from (5.3), as $\mathbf{W} = |\mathbf{A}^T\mathbf{A}|$.

Low-dimensional projections can be obtained from the sparse coding graphs using the locality preserving projections (LPP) approach. Since the final goal is to predict tags using features obtained from test data, we compute the embedding only for the features in the training data, and not for the tags. Given the affinity matrix $\mathbf{W} \in \mathbb{R}^{T\times T}$, we compute the degree matrix $\mathbf{D}$ with each diagonal element containing the sum of the corresponding row or column of $\mathbf{W}$. The d projection directions for

LPP are then computed by optimizing

$$\min_{\mathbf{V}} \sum_{i,j=1}^{T} \|\mathbf{V}^T\mathbf{x}_i - \mathbf{V}^T\mathbf{x}_j\|_2^2 w_{ij} \text{ s.t. } \sum_{i=1}^{T} \|\mathbf{V}^T\mathbf{x}_i\|_2^2 \delta_{ii} = 1. \tag{5.4}$$

where w_{ij} and d_{ii} are the corresponding elements of the affinity and the degree matrices. This optimization ensures that the embedding preserves the structure defined by the sparse coding based graph. Defining the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$, (5.4) can be rewritten as

$$\min_{\text{trace}(\mathbf{V}^T\mathbf{X}\mathbf{D}\mathbf{X}^T\mathbf{V})=\mathbf{I}} \text{trace}(\mathbf{V}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{V}). \tag{5.5}$$

and this can be solved using eigen decomposition.

5.7 Tag Prediction

Let us denote the low-dimensional projection of the training features as $\mathbf{Y} = \mathbf{V}^T\mathbf{X}$. For a test image, we extract the feature $\mathbf{x}^t$ and compute the projection $\mathbf{y}^t = \mathbf{V}^T\mathbf{x}^t$. The goal here is to predict the tag vector for the test feature in the embedded space. From (5.4), it is clear that the sparse coding graph relations directly correspond to the Euclidean distances between the low-dimensional features. Hence, we can employ a simple local neighbourhood based least squares reconstruction for $\mathbf{y}^t$ in terms of the training samples $\{\mathbf{y}_i\}_{i=1}^T$. The reconstruction coefficients are given as

$$\begin{aligned} \boldsymbol{\alpha}^t = &\operatorname*{argmin}_{\boldsymbol{\alpha}^t} \|\mathbf{y}^t - \sum_{i=1}^{T} \boldsymbol{\alpha}_i^t \mathbf{y}_i\|_2^2 \\ &\text{s.t. } \boldsymbol{\alpha}_i^t = 0, \text{ if } i \notin \Omega_k(\mathbf{y}^t) \text{ and } \mathbf{1}^T\boldsymbol{\alpha}^t = 1. \end{aligned} \tag{5.6}$$

Here $\Omega_k(\mathbf{y}^t)$ indicates the set of k nearest neighbours of $\mathbf{y}^t$. However, this approach requires global similarities between the test feature and the training features for identifying the suitable tags. In other words, the k nearest neighbours for the test sample might not include other semantically related images because of the lack of global

Table 5.1: Precision-Recall Rates for the Corel-5k Dataset.

	Least Squares			**Sparse Coding**		
Algorithm	**Basic**	**Ext. 1**	**Ext. 2**	**Basic**	**Ext. 1**	**Ext. 2**
Rec. > 0	124	122	129	122	121	130
Results on All 260 Words						
Mean Prec.	0.22	0.24	0.2	0.23	0.26	0.22
Mean Rec.	0.27	0.26	0.29	0.27	0.27	0.3
Results on Best 50 Words (Precision)						
Mean Prec.	0.71	0.79	0.68	0.77	0.83	0.74
Mean Rec.	0.69	0.69	0.74	0.71	0.72	0.78

similarity. This can be addressed by computing the sparse code for the test feature with the training features as the dictionary.

Although the least squares approach may result in a reasonable performance, computing the reconstruction weights using a sparse coding approach may result in identifying training images that have higher semantic similarity to the test image. Therefore, we recast the problem in (5.6) as

$$\boldsymbol{\alpha}^t = \underset{\boldsymbol{\alpha}^t}{\operatorname{argmin}} \|\mathbf{y}^t - \mathbf{Y}\boldsymbol{\alpha}^t\|_2^2 + \lambda\|\boldsymbol{\alpha}^t\|_1 \tag{5.7}$$

to identify the reconstruction coefficients. Similar to the approach described in [86], we obtain the predicted tag vector for the test image using the tags of the training set and the reconstruction coefficients, $\mathbf{u}^t = \mathbf{U}\boldsymbol{\alpha}^t$.

5.7.1 Improving Tag Prediction Performance

Typically, automated image annotation performance is evaluated by comparing the predicted labels with the human-labelled ground truth. The metrics commonly used for evaluating the performance are the precision and recall for every label. Precision of a label is measured as the ratio between correctly annotated images and the total number of images annotated with that label. Recall is defined as the ratio between the number of correctly annotated images and the number of images that have the said label in the ground truth annotation. An effective image annotation system is characterized by high precision and recall. However in practical algorithms there is always a trade-off between precision and recall characteristics. Therefore it will be useful to incorporate suitable modifications to our algorithm to obtain higher precision or higher recall, depending of the requirements of the application, without significantly affecting the other. To achieve this, we propose extensions to the algorithm using randomized strategies either when the embedding is computed or when the tags are predicted. Note that these methods can be applied to both sparse coding based prediction and least squares based prediction.

Improving Precision

The effectiveness of the embedding step in preserving semantic relationships directly impacts the precision of the image annotation algorithm. The proposed approach in Section 5.6 can be improved by building multiple embeddings tuned for different random subsets of the tags. Each random subset indexed by s, contains only R randomly chosen tags out of the total P tags. The tag matrix thus obtained is denoted by $\mathbf{U}_s$. Note that it contains a random selection if R rows from $\mathbf{U}$. The resulting embedding retains only partial semantic relations, but since R is small, it can provide a high precision for those tags. We obtain one projection matrix per round, i.e., for one particular random choice of R tags. For a test image, we obtain a low-dimensional embedding for each projection and perform sparse coding or least squares reconstruction as described in Section 5.7. The labels of the test image are estimated using the final average coefficient vector obtained as the mean of coefficient vectors from multiple rounds. Furthermore, in order to simplify the dictionary learning procedure, we used dictionaries obtained as randomly chosen columns from $\begin{bmatrix} \mathbf{Y} \\ \mathbf{U}_s \end{bmatrix}$ instead of K-means dictionaries in each round. Note that, although we allow overlap between random subsets of tags, when computing the embeddings, some labels might participate in very few rounds and this could lead to a reduced recall rate for that label.

Improving Recall

The tag prediction approaches described in the earlier part of this section require the choice of a global parameter, such as sparsity penalty λ or the number of nearest neighbours k. These determine the number of training vectors that will participate in the representation. Relaxing this penalty will result in increased recall with a reduction in precision. To mitigate this effect to some extent, we propose to construct multiple dictionaries using random subsets of training features. When the size of this subset is reasonably high, the prediction algorithm can result in improved recall without affecting the precision significantly. Similar to the previous case, we compute the final average coefficient vector as the mean of coefficient vectors obtained from the multiple dictionaries. This final coefficient vector is used to estimate the tags of the test image.

5.8 Simulations

In order to test our proposed set-up, we use the Corel-5k dataset [88], which is typically used to benchmark image annotation tasks. The dataset contains 4500 images for training and 499 test images. Each image is annotated with a maximum of 5 tags. The dataset has a vocabulary size of 260 words. We derive image-level features using the procedure described in Section 5.5.

For the training data, we compute sparse codes using a dictionary of size $K = 512$, compute the sparse coding graph and obtain $d = 150$ dimensional embeddings. Sparse codes were obtained using the open-source SPAMS package [89]. In table 5.1, the tag prediction procedures mentioned as *Regular* are the basic least squares and sparse coding algorithms described in Section 5.7. *Randomized 1* is the variation of the basic algorithms for improving precision (Section 5.7.1), and in this case we fix $R = 40$, and use 30 random subsets. In order to improve recall (Section 5.7.1), we use the *Randomized 2* method in which during each round 3000 training examples are randomly chosen and this is repeated for 30 rounds.

Mean precision and recall are computed for all the 260 tags and a subset of best 50 tags (highest precision). The number of tags with non-zero recall is also a good indicator of the sensitivity of the algorithm to a wide range of tags. It can be observed that the sparse coding method performs better than least squares approach. As expected, the randomized methods are useful in improving the precision and recall respectively, while leading to a small reduction in the other measure in most of the cases. Hence the least squares method can be used if we need a fast and simple implementation, whereas further improvements in performance can be obtained with sparse coding for increased computational complexity.

Chapter 6

SUBSPACE BASED METHODS FOR ANALYSIS OF SILICON-PORE SENSOR DATA

The use of semiconductor nanostructures and novel nanomaterials such as conducting polymers for building biosensors is gaining popularity, since they can enable single molecule detection [90]. A silicon nanopore, for example, behaves like a protein channel and allows selective analyte transport [91, 92]. The ability to fabricate structures ranging from micrometers down to a few nanometers has led to the re-discovery of the Coulter counting principle. In Coulter counting experiments, the drop in ionic current through a micron-sized aperture is recorded upon passage of a particle, such as a blood cell. The shape and amplitude of the current signal then enables a discrimination between different cells, allowing for example a complete blood cell count to be generated. For a Coulter counter to be most efficient in discriminating between different particle sizes, the aperture through which the particles flow has to be on the same order of magnitude as the particles themselves. Thus, nanoscale particles are best detected using a nanopore. Using nanopores, however, restricts the range of particle sizes significantly. Moreover nanopores tend to become irreversibly blocked by larger particles or molecules, rendering them useless. Thus, in our experiments, we decided to use a micron-sized silicon pore, since it offers the best trade-off between sensitivity and detection range and is significantly easier to fabricate compared to a silicon nanopore.

The silicon micropore is setup to act as a Coulter counting device [93] with the silicon micropore chip itself being sandwiched between two chambers containing electrolyte solution, typically physiologically buffered saline (PBS). A constant voltage is applied across the silicon micropore using reversible Ag/AgCl reference electrodes. Since the current flowing through the aperture at the applied bias is on the order of a few nA, no separate counter and working electrodes are necessary to pass the current. To observe Coulter translocation events, silica microbeads were added to the electrolyte solution. Since the silica beads carry a surface charge, the applied bias causes the silica microbeads to enter the pore resulting in a drop in the value of the baseline current. By functionalizing the silica beads with biotin using amine linker chemistry, an immunoassay can be performed using the beads. By adding avidin to the solution, the beads will agglomerate and cause a drop in Coulter current that is proportional to that of a single silica bead passing through the micropore. The amount of drop in current and the corresponding increase in resistance can be used to detect and identify the presence of bead agglomeration, which can be extended to provide a direct biomolecule detection capability [42]. Previous experiments with signals originating from nanopore Coulter counting [94] and electromigration of beads through silicon nanopores [95] show the wide range of applicability of particle classification using ionic current drop measurements. The ability to employ signal processing to discriminate between different biochemical entities has been reported in prior work by the authors, including extracting features for ion-channel devices [96], classifying ion-channel signals using HMMs [42] as well as using neural networks [46] and SVM [97] to characterize and detect the presence of analytes in ion-channel signals.

We explore two features of the current drop to classify the object that is passing through the micropore: the amplitude of the drop and the time duration of the event. We discriminate between an event and a non-event based on the amplitude of the drop. The problem we are facing in the experiments when basing the event/non-event classification on the amplitude is that the signals obtained from the setup are inherently noisy. This is due to the noise sources being present, which is primarily the thermal noise of the aperture resistance itself. Since a lower resistance creates a higher thermal current noise, this limits the minimal achievable noise level of the system. Since the spectrum of the thermal noise is white, the only way to reduce the root-mean-square (rms) value of the noise is to limit the recording bandwidth. In a purely analog signal chain, this can be accomplished using higher order analog filters. These analog filters, however, lead to a modification of a step response in the signal. The analog filters that are typically employed in the analog signal processing chain are 8$^{\text{th}}$ order Bessel filters, which provide the best trade-off between signal integrity and steepness of the filter curve. However, aggressive setting of the corner frequency to achieve the lowest rms noise will have an effect on both the amplitude and the duration of the current drop in a Coulter signal, thereby affecting the ability to classify the particle or molecule passing through the micropore.

Wavelet based denoising is employed which preserves the amplitude, width and shape of the current drop [98]. After denoising, we obtain a series of current drops, with each one being a probable event caused by a bead passing through the micropore or a non-event caused by a bead bouncing off the walls. We develop the denoising protocol as a supervised learning problem, since we have knowledge of the anticipated amplitude of the drops for these two cases. After the wavelet-based denoising, we employ Support Vector Machines to classify drops as events and non-events.

Once we extract the events from the signal, the next step is to classify them and identify which biomolecule caused the event. Depending on the biomolecule size and shape, the amplitude and the time duration of the drop will change. In our experiments, we consider two or more silica beads coagulating as a representation of different biomolecules. Contrary to the denoising problem, where we know the amplitude of the current drop based on the geometry and size of a single silica bead, we do not know the current drop in case of bead agglomerates. Thus, we approached this problem as an unsupervised learning problem and we performed clustering in the feature domain.

6.1 Data Processing

6.1.1 Data Acquisition

Micropore data was generated using a Coulter counting element constructed using a Teflon chamber with two baths surrounding the micropore chip. The two baths were filled with 0.01M PBS electrolyte solution as shown in Figure 6.1. The Ag/AgCl electrodes are dipped into each bath of 0.01M PBS and the measurements are taken using a HEKA EPC-8 current amplifier, using a gain of 0.1mV/pA and analog filtering with a corner frequency of 1kHz. The analog current data was digitized using a National Instruments PCIe-6221 DAQ board, controlled by the WinEDR software [99]. Measurements of the current with only the 0.01M PBS solution on either side of the Teflon chamber provided the value of the baseline current for the PBS solution. This value was used to calculate the current drop due to the passing of silica microbeads of known size and geometry through the device. During the experiment, a bias of 400mV was applied across the silicon micropore and the baseline current was recorded. The sampling rate was set at 10kHz in all experiments.

6.1.2 Wavelet Transform Based Signal Denoising

The Discrete Wavelet Transform (DWT) is a linear transformation that can be used to analyze the temporal and spectral properties of non-stationary signals. The DWT decomposes a signal into the coarse approximation and the detail information coefficients. DWT is computed by successively applying pairs of analysis filters to the input signal. The advantage of the wavelet representation is that it can provide

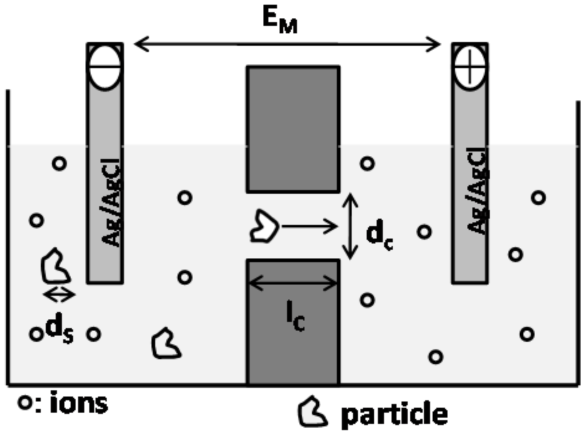


Figure 6.1: Teflon Chambers With the Device in the Center and the Analyte Within the Device (Micropore). The Electrodes Can Be Seen on Either Side of the Chambers.

both time and frequency parameters for specific dynamic signal events, i.e. time-frequency localization [62]. In contrast, the Fourier transform based filtering methods assume that the signal is stationary and thus cannot provide any information on the variations in the spectrum with respect to time. Denoising using the Discrete Wavelet Transform (DWT) is a nonlinear operation that involves appropriate thresholding of wavelet coefficients depending on the noise variance.

6.1.3 Features: Baseline Current, Drop Height and Drop Width

The useful features to be extracted from the micropore signals are the baseline current, peak height and peak width. The combination of the baseline current and the peak height indicates whether a bead has passed through the micropore completely or not. The peak amplitude is proportional to the baseline current, i.e., greater the baseline current I, greater will be the drop in current ΔI for the beads of the same diameter. The width of the peak is proportional to the diameter of the bead.

6.1.4 Event and Non-Event Classification

Experimental data of duration 30 minutes with a sampling rate of 10 kHz are available. A rectangular window of size $10,000$ samples with no overlap was used to segment the data. In each segment, drops were extracted using a gradient method. Each drop was labeled either as an event or a non-event. An event indicates that a bead passed through the micropore completely whereas a non-event indicates a bead bounced off the walls instead of passing through the micropore.

We use SVM to classify events and non-events as we have theoretical estimates of the drop values for events and non-events [91]. SVMs have been widely used for solving binary classification problems [56]. SVMs are decision machines that rely on transforming lower dimensional data into higher dimensional patterns, so that data from two categories can always be separated by a hyperplane, in accordance with Cover's theorem [100]. The SVM uses the concept of the margin, which is defined to be the smallest distance between the decision boundary and any of the samples. The support vectors are the training samples that are closest to the decision boundary and thus define the optimal separating hyperplane. In SVM, the decision boundary is chosen to be the one for which the margin is maximized. It can be shown that the largest margin minimizes the total generalization error [7]. The choice of the nonlinear function that maps the input into a higher dimensional space is usually dependent on the problem. Usually polynomial or radial basis functions are used to perform the mapping.

6.2 Dirichlet Process Mixture Model

The final step of the algorithm is to cluster the events. The overall number of clusters, N_{C}, is an input parameter for most of the algorithms. Estimating this quantity from the data itself is known as model order selection. Approaches such as Akaike's information criterion (AIC), Bayesian information criterion (BIC) and the minimum description length (MDL) are based on information theoretic principles. They identify the "true" number of clusters by setting a balance between the model complexity and the models capability to describe the given data, measured by the likelihood. In contrast, Bayesian Dirichlet Process (DP) based methods represent the number of clusters as a random variable [101]. An estimate of this random variable is obtained by sampling the model posterior under the observed data.

In the finite parametric mixture given in (6.1), different components of the mixture are defined by different values of the parameter vector θ_k [102]. If the mixture is estimated from the data, the mixture weights are chosen in proportion to the size of the classes, i.e. $c_\mathrm{k} = \frac{n_\mathrm{k}}{n}$, where n_k out of n total data points are assigned to the component k.

$$p(x|\mathbf{\Theta}) = \sum_{k=1}^{N_c} c_k F(x|\theta_k). \tag{6.1}$$

The finite mixture model (FMM) can be regarded as the result of integrating the distribution function F against

$$G_{\mathrm{FMM}}(\theta) := \sum_{k=1}^{N_\mathrm{C}} c_\mathrm{k} \delta_{\theta_\mathrm{k}}. \tag{6.2}$$

Here $\delta_{\theta_\mathrm{k}}$ denotes the Dirac function on Ω_θ, the parameter space, centered at θ_k. DPM models augment the expression in (6.2) by an extra term,

$$G_{\mathrm{DP}}(\theta) := \sum_{k=1}^{N_\mathrm{C}} c_\mathrm{k} \delta_{\theta_\mathrm{k}} + \alpha G_0(\theta). \tag{6.3}$$

The function G_0 is a probability distribution on the domain Ω_θ of mixture parameters, and $\alpha \in \mathbb{R}_+$ is a positive scalar parameter [103]. The difference between the models become clear when we consider a data generating process. In the case of FMM in (6.2), a new data $\mathbf{x}$ is always drawn from from one of the N_C parametric mixture components. In the case of FMM being a Gaussian mixture model, we sample $\theta \sim G_{FMM}$ and then $\mathbf{x} \sim F(.|\theta)$. In the case of DPM (6.3), $\mathbf{x}$ can be drawn either from the N_C components or from the new component $F(.|\theta^*_{N_C+1})$, with a probability proportional to α. The new parameter value $\theta^*_{N_C+1}$ is sampled from G_0. This describes the ability of DPM models to generate as many clusters as required by the observed data.

The standard Gibbs sampling algorithm for DPM models, such as the EM algorithm, also make use of latent variables to assign data points to the mixture components [104]. For each $\mathbf{x}_{\mathrm{i}}$, the discrete index variable S_i specifies the index of the mixture component to which $\mathbf{x}_{\mathrm{i}}$ is assigned [105]. Within the algorithm, a value of $S_i = 0$ indicates the generation of a new mixture component from G_0 as a model for $\mathbf{x}_{\mathrm{i}}$. The S_{i} are determined by computing the mixture proportions $\tilde{q}_{\mathrm{ik}}$ as

$$\tilde{q}_{i0} := \int_{\Omega_\theta} F(\mathbf{x}_{\mathrm{i}}|\theta) G_0(\theta) d\theta, \tag{6.4}$$

$$\tilde{q}_{ik} := n^{\text{-i}}_{\mathrm{k}} F(\mathbf{x}_{\mathrm{i}}|\theta_{\mathrm{k}}) \quad k = 1, ..., N_{\mathrm{C}}. \tag{6.5}$$

Here $n^{\text{-i}}_{\mathrm{k}}$ is the number of data points assigned to the component k with $\mathbf{x}_{\mathrm{i}}$ removed from the data set. The proportions are normalized to obtain the mixture probabilities

$$q_{\mathrm{ik}} := \frac{\tilde{q}_{\mathrm{ik}}}{\sum_{t=0}^{N_{\mathrm{C}}} \tilde{q}_{\mathrm{it}}}, \text{ for } k = 0, ..., N_{\mathrm{C}}. \tag{6.6}$$

For every $\mathbf{x}_i$, $q_{i0}, ..., q_{iN_C}$ are computed and $\mathbf{S}_i$ are sampled from these mixture probabilities. If $S_i = 0$, we create a new component and sample $\theta_{N_C+1} \sim F(\mathbf{x}_i|\theta_{N_C+1})$. And then for each $k = 1, ..., N_{\mathrm{C}}$, we sample $\theta_{\mathrm{k}} := G_0(\theta_{\mathrm{k}}) \prod_{i|S_{\mathrm{i}}=k} F(\mathbf{x}_{\mathrm{i}}|\theta_{\mathrm{k}})$.

The integral in (6.4) has a closed form solution if the likelihood function $F(\mathbf{x}_{\mathrm{i}}|\theta)$ and the prior distribution G_0 are conjugate pairs. Consider $F(\mathbf{x}_{\mathrm{i}}|\theta) = N(\mathbf{x}_{\mathrm{i}}; \mu, \Sigma)$, where the mean value μ and covariance Σ are independent. In this implementation, the prior probability distribution function of the mean value, μ, is a normal distribution $\mu \sim N(\mu_0, \Sigma_0)$; the prior probability distribution function of the covariance follows the inverse gamma distribution $\sigma^2 \sim \Gamma^{-1}(a, b)$.

$$\Sigma = I_{\mathrm{m}}\sigma^2. \tag{6.7}$$

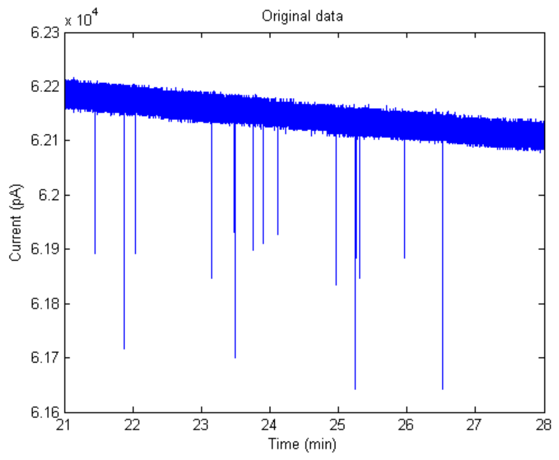
Here, I_{m} is unit diagonal matrix. The probability of generating a new class parameter q_0 is given by

$$\begin{aligned} q_0 &\propto \alpha \int F(\mathbf{x}_{\mathrm{i}}|\theta) G_0 d\theta. \\ q_0 &= \alpha \frac{1}{(2\pi)^{\mathrm{m}}(det(\Sigma\Sigma_0)^{0.5})} \times \\ &\quad \exp\left(\frac{-(\mathbf{x}_{\mathrm{i}} - \mu_0)^{\mathrm{T}}(\Sigma\Sigma_0)^{-1}(\mathbf{x}_{\mathrm{i}} - \mu_0)^{\mathrm{T}}}{2} \right). \end{aligned} \tag{6.8}$$

Here m is the dimension of the signal vector.

In our simulations, we determined using cross validation, that the biorthogonal wavelet gave the best performance for denoising the signals. Figure 6.2 shows the original and denoised version of one signal record. The signal was divided into parts to speed up computations, each part containing 10,000 samples at a sampling rate of 10kHz. To capture most of the features in the signal, the number of levels of wavelet decomposition was chosen to be 5. Two types of thresholding operations can be performed: hard and soft. In hard thresholding, the wavelet coefficients whose absolute value is less than the specified threshold limit T are set to zero. Soft thresholding is an extension of hard thresholding; it first performs hard thresholding, and then it shrinks the nonzero coefficients towards 0 by an amount T [63]. The value of the threshold T is dependent on the type of noise and the noise variance. In our case, soft thresholding was used and the threshold was set to be 17.9.

In one of the datasets, 1523 drops were extracted from the signal, out of which 43 drops were indicated as events by the SVM classifier. In another dataset, 241 events were detected. The Dirichlet Process Mixture Model (DPMM) was used on the above mentioned two sets of events with different number of types of events in them. One



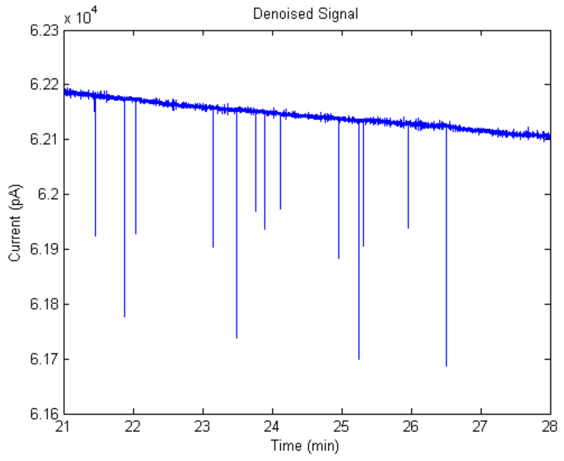


Figure 6.2: The Original Signal Showing Drops in Current Due to the Translocation Of Biotin-Coated Silica Beads Through a $5\mu M$ Diameter Pore and the Signal Denoised Using DWT.

such clustered events are shown in Figure 6.3. The concentration parameter α was chosen to be 2. The mean drop amplitudes obtained for the three clusters are -213.4 pA, -433.0pA and -1086.7 pA respectively. The mean of the three clusters can be interpreted as the current drop values for one bead passing, two bead passing together and three bead passing together. The objective of mixture distributions is to form a probabilistic model composed of a number of component clusters. Each cluster is characterized by a set of parameters describing the mean and variance of

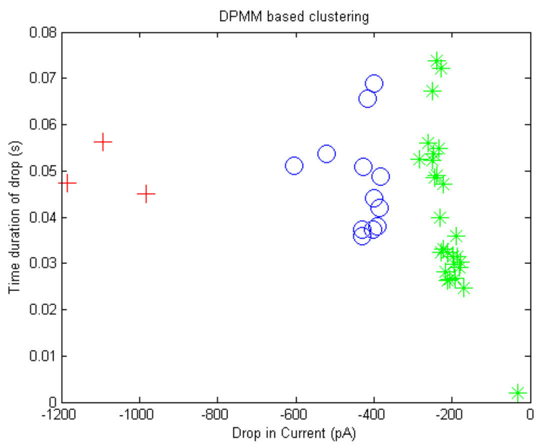


Figure 6.3: DPMM Based Clustering With Concentration Parameter $\alpha = 2$.

the component data. In order to robustly estimate the parameters, we need to first determine the number of clusters. This can be done by using a representative sample of training data and estimating the number of clusters and their parameters from this data. We have made the assumption that each clusters follows a multivariate Gaussian distribution.

The number of clusters K cannot be computed using maximum likelihood (ML). This is because the likelihood may always be made better by choosing a large number of clusters. Intuitively, the log likelihood may always be increased by adding more clusters since more clusters may be used to more accurately fit the data. However, a large model order makes the model complex and restricts its ability to generalize to new set of data. This problem of estimating the order of a model is known as order identification and numerous approaches to address this problem exist in literature.

6.3 Minimum Description Length (MDL) Principle

The MDL principle is a formalization of the Occam's Razor in which the best hypothesis for a given set of data is the one that leads to the best compression of the data. This estimator works by attempting to find the model order which minimizes the number of bits that would be required to code both the data samples and the parameter vector [106]. While a direct implementation of the MDL estimator may depend on the particular coding method used, Rissanen developed an approximate expression for the estimate based on assumptions given in [107] and the minimization of the expression,

$$\mathrm{MDL}(K, \theta) = -\log p(\mathbf{x}|k, \theta) + 0.5L\log(NM), \tag{6.9}$$

where θ is the representation for all the parameters and L is is the number of continuously valued real numbers required to specify the parameter, given by

$$L = K\left(1 + M + \frac{M(M+1)}{2}\right) - 1. \tag{6.10}$$

and M is the number of dimensions of the data. It can be observed that the expression in (6.9) contains a penalty term that depends on the total number of data values NM. In practice, this is important since otherwise more data will tend to result in overfitting of the model. In the proposed approach, we combine the MDL principle with the GMM and estimate the number of clusters.

In our dataset described in Section 2.1.2, 7891 drops were extracted from the signal, out of which 241 drops were indicated as events by the SVM classifier. The next step involves choosing K (number of clusters) for categorizing events and finding outlying clusters.

We use the MDL method described above to estimate the number of clusters which was found to be 4 in our dataset. Figure 6.4 shows the GMM based clustering using the EM algorithm and spectral clustering on the feature set. GMM based clustering attempts to fit multiple Gaussian distributions to the data. However, they clearly are not able to capture the natural shape of clusters in feature space. The main advantage of spectral clustering is that it can form arbitrary cluster shapes, thus is able to capture the natural cluster shape in our dataset and the outliers are clustered separately.

One of the goals of the analysis is to study the effect of time duration of events on categorization or clustering. Figure 6.4 shows the spectral clustering and GMM based clustering results on our dataset. The three bottom clusters obtained in the spectral clustering are the three categories of events and can be seen as the outcome of the variation in current drop. The category of any new event can be determined by comparing it with the three clusters. The top cluster can be seen as an outlying cluster formed of events with high time duration. It can be observed that current drop amplitude is the dominant feature responsible for the formation of the natural clusters in the spectral clustering. In order to perform robust identification of translocation and trapping events, we need to: (a) remove dense noise from the signal, (b) separate the baseline signal into trapping and sparse translocation events and (c) extract event statistics and cluster them into groups. Considering the example signal in figure 6.5(a), and denoting it as $\mathbf{d} \in \mathbb{R}^p$, we first divide it into non-overlapping frames and stack them column-wise into a matrix $\mathbf{D} \in \mathbb{R}^{mxn}$ such that $p = mn$. If $\mathbf{d}$ is noise-free with only trapping events, the matrix $\mathbf{D}$ will be of low rank. This is because most of the columns in this matrix are scaled versions of each other. Figure 6.5(b) shows the matrix as a scaled image of its entries. It can be observed that only two windows and hence two columns of matrix contain the step signals. All other columns have constant entries. Thus, the rank of the matrix is 2 which is also reflected in figure 6.5(c) with only 2 significant singular values. Figure 6.6 shows plots for the noisy version of signal. A noisy version of the matrix $\mathbf{D}$ can be expressed as

$$\mathbf{D} = \mathbf{L} + \mathbf{G}, \tag{6.11}$$

where the matrix $\mathbf{L}$ is of low rank and the matrix $\mathbf{G}$ represents dense Gaussian noise.

When sparse translocation events are present in the signal, the data matrix $\mathbf{D}$ can be expressed as

$$\mathbf{D} = \mathbf{L} + \mathbf{S} + \mathbf{G}, \tag{6.12}$$

where the matrix $\mathbf{S}$ is sparse and represents the translocation events.

6.4 De-noising and Event Extraction

We propose two approaches for de-noising and event extraction. In the first approach, we use the Discrete Wavelet based de-noising (DWT) [108] as a preprocessing step. The resulting signal matrix $\mathbf{C}$ is then decomposed into low rank ($\mathbf{L}$) and sparse components ($\mathbf{S}$) using the RPCA algorithm [109]. In the second approach, we decompose the noisy signal matrix $\mathbf{D}$ as $\mathbf{L} + \mathbf{S} + \mathbf{G}$, using an algorithm similar to that described in [110]. In this approach, de-noising is a part of the decomposition itself and not a separate preprocessing step. Note that the sparse translocation events in our case cannot be positive, as the current always drops during the passage of beads. Therefore, we force the matrix $\mathbf{S}$ to contain only non-positive values. We also explicitly add a piece-wise smoothness constraint using the *Total Variation (TV)* [111] regularization to the sparse component of the signals in order to obtain smooth drop events. The extracted events are grouped using multi-level clustering as described in our previous work [112], to identify long duration and short duration events and their internal clustering.

6.4.1 NpRPCA

We formulate the optimization problem of NpRPCA as,

$$\begin{aligned} &\text{minimize } \operatorname{rank}(\mathbf{L}) + \|\mathbf{S}\|_0, \\ &\text{subject to } \mathbf{C} = \mathbf{L} + \mathbf{S}, \end{aligned} \tag{6.13}$$

where the ℓ_0 norm, $\|.\|_0$, counts the number of non-zero elements in the argument matrix. The problem is non-convex and NP-hard because of the rank and the ℓ_0 norm constraints. Therefore, it can be made convex by replacing rank using its convex surrogate, the *nuclear norm*, which is defined as the sum of the singular values of the matrix. Similarly the ℓ_0 norm can be replaced by the convex ℓ_1 norm, which corresponds to the sum of absolute values of the entries of the matrix. The relaxed problem is given by

$$\begin{aligned}&\text{minimize}\|\mathbf{L}\|_* + \|\mathbf{S}\|_1,\\ &\text{subject to } \mathbf{C} = \mathbf{L} + \mathbf{S},\end{aligned} \tag{6.14}$$

where $\|\mathbf{L}\|_*$ is the *nuclear norm* defined as $\sum_k \sigma_k$, the sum of singular values of the matrix $\mathbf{L}$.

We use the Singular Value Thresholding (SVT) method described in [57] in order to solve (6.14). The SVT algorithm iteratively updates the low rank $\mathbf{L}_t = D_\tau(\mathbf{P}_{t-1})$ and sparse component $\mathbf{S}_t = T_\lambda(\mathbf{P}_{t-1})$ where $\mathbf{P}_t$ is the matrix obtained after correction based on the constraint $\mathbf{C} = \mathbf{L}+\mathbf{S}$, updated as $\mathbf{P}_t = \mathbf{P}_{t-1}+\delta(\mathbf{C}-\mathbf{L}_t-\mathbf{S}_t)$. $D_\tau(.)$ is the shrinkage operator which retains the singular values greater than τ of the argument matrix. $T_\lambda(.)$ is the thresholding operator which retains values less than λ and sets the rest to 0. In our simulations, the value of τ was set to $1e4$, δ as 0.9 and λ as 0.1.

6.4.2 NpGoDec

We would like to decompose the matrix $\mathbf{D}$ as $\mathbf{D} = \mathbf{L} + \mathbf{S} + \mathbf{G}$, where $\mathbf{S}$ is a non-positive sparse error matrix and $\mathbf{G}$ is a dense noise matrix. In order to extract the baseline components in the signals, we impose a low rank constraint on $\mathbf{L}$. As a result, we obtain the following joint optimization problem:

$$\begin{aligned}&\arg\min_{\mathbf{L},\mathbf{S}} \|\mathbf{D}-\mathbf{L}-\mathbf{S}\|_F^2,\\ &\text{subject to } \operatorname{rank}(\mathbf{L}) < r, \operatorname{card}(\mathbf{S}) < s, \operatorname{TV}(\mathbf{S}) < v.\end{aligned} \tag{6.15}$$

The solution for the optimization problem in (6.15) can be efficiently obtained by solving the following sub-problems:

$$\begin{aligned}&\mathbf{L}_t = \arg\min_{\mathbf{L}} \|\mathbf{D}-\mathbf{L}-\mathbf{S}_{t-1}\|_F^2,\\ &\text{subject to } \operatorname{rank}(\mathbf{L}) < r.\end{aligned} \tag{6.16}$$

$$\begin{aligned}&\mathbf{S}_t = \arg\min_{\mathbf{S}} \|\mathbf{D}-\mathbf{L}_t-\mathbf{S}\|_F^2,\\ &\text{subject to } \operatorname{card}(\mathbf{S}) < s, \operatorname{TV}(\mathbf{S}) < v.\end{aligned} \tag{6.17}$$

Each of the sub-problems are convex and can be solved using different methods. We employ the singular value hard thresholding method to solve the first sub-problem for obtaining $\mathbf{L}_t$. In the second sub-problem, we need to iteratively solve for the two constraints. First, entry-wise hard thresholding is used to obtain a sparse matrix $\mathbf{S}_t$, followed by *Total Variation (TV)* based de-noising [111] to incorporate both the constraints. One iteration of the joint problem involves solving each of the sub-problems once with arguments as shown in the (6.16) and (6.17). We will now discuss convergence of the joint optimization problem in (6.15) without the smoothness constraint (TV) as described in [110]. Let the optimal objective values of the two sub-problems, in iteration t, be $E_t^{(1)}$ and $E_t^{(2)}$ respectively, where $E_t^1 = \|\mathbf{D}-\mathbf{L}_t-\mathbf{S}_{t-1}\|_F^2$, and $E_t^2 = \|\mathbf{D}-\mathbf{L}_t-\mathbf{S}_t\|_F^2$. Global optimality of $\mathbf{S}_t$ leads to $E_t^{(1)} > E_t^{(2)}$. Similarly, global optimality of $\mathbf{L}_{t+1}$ leads to $E_t^{(2)} > E_{t+1}^{(1)}$. Hence, the objective values of the sub-problems monotonically decrease, and the joint objective converges to a local optimum. After convergence, the matrices $\mathbf{S}$, $\mathbf{L}$ provide the sparse, and low rank baseline components of the signals respectively. We empirically observe that the decomposition error $||\mathbf{D} - \mathbf{L} - \mathbf{S}||_F^2$ converges in the case of the approximate problem formulation satisfying additionally a TV based smoothness constraint.

Figures 6.7 and 6.8 illustrate the signal decomposition obtained using the original formulations of RPCA and GoDec, similarly figures 6.9 and 6.10 the proposed DWT+NpRPCA and NpGoDec methods, respectively. As it can be observed, though RPCA identifies the sparse events, the Gaussian noise is absorbed in the low rank component. As a result, RPCA results in a higher rank than expected due to the presence of noise. However, NpRPCA retains only non-positive events, denoting the drop caused by the passage of biomolecules. The GoDec method requires the user to provide the expected rank and cardinality information. We provide overestimated values as inputs to the algorithm. Choosing a higher rank allows lesser noise in the baseline signal, and choosing a higher cardinality overestimates the sparse events, which can be easily detected in the classification step. As a result, GoDec de-noises the baseline signal in addition to identifying the sparse translocation events. However, it also produces spurious positive peaks at the trapping event locations. By imposing non-positive constraints, these spurious peaks were eliminated in the proposed NpGoDec. Furthermore, the constraint on total variation leads to smooth drop events as expected.

6.5 Event Classification and Clustering

The candidate events extracted from a signal need to be further classified as passage events, reflected events and noise-artefacts. Passage events are the events in which a single bead or agglomerated beads completely go through the pore. Reflected events are the cases where the beads get reflected by the pore walls and cause only partial drop in current. Since, we have knowledge about the drop in current, and time duration in the case of one bead passing through completely, we use this information to hand label events into two classes as passage events and reflection events, respectively. The reflection class can contain noise induced events, and passage class contains events belonging to single/agglomerated beads passing through the pore. A two class Support Vector Machine (SVM) is used to perform the classification. The various measurements (pore length, diameter etc.) used in hand calculations of single bead drop features can have slight variations and lead to noisy classification threshold estimates and thus we erroneously label a few events. Such events were misclassified by the SVM or were close to the decision boundary. We reject such events and learn an SVM classifier using only trusted events.

The simulated noisy data contained 200 sparse translocation events, and 50 reflection events belonging to passage class. Linear SVM was trained using 50 hand-labeled examples with equal ratio of passage and reflection events. We relearn a SVM after rejecting 6 hand-labelled examples. In case of events extracted from DWT+NpRPCA, we achieved a classification accuracy of about 91 percent, after cross-validation, in correctly classifying the remaining 200 events. In the case of NpGoDec, we had an improved classification rate of 96 percent. This demonstrates that NpGoDec better preserves the drop shape characteristics compared to DWT+NpRPCA.

We are interested to group the passage events based on the number of agglomerated beads associated with each of these events using clustering methods. The number of such groups in a dataset and their shape are not known beforehand. We use Minimum Description Length (MDL) [113], an information theoretic based algorithm to estimate the number of groups in a given dataset. We use a two-fold clustering approach as described in our previous work [112]. Spectral clustering [33] is used at the first level which forms four groups as predicted by MDL. Three of the groups have similar drop durations and the dominating factor is the drop amplitudes.

The fourth group is of events which have high drop duration. We run a simple k-means clustering on this group using only drop amplitudes to get an estimate of the cardinality of agglomerated beads. In the simulated noisy data, clustering accuracies obtained using the DWT+NpRPCA and NpGoDec methods were comparable at 92.8% and 94.6% respectively. Furthermore, we observed that most errors, in both methods, occurred in labelling high duration events.

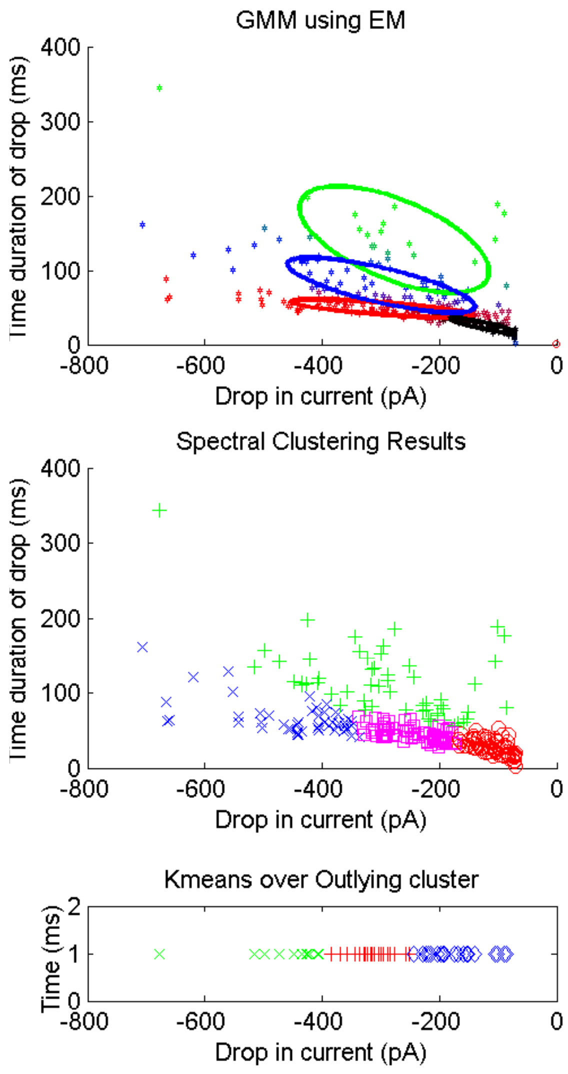


Figure 6.4: (Top) GMM Based Clustering and (Middle) Spectral Clustering on the Feature Set. (Bottom) K-means Clustering on the Outlying Cluster Based Only on the Variations in the Current Drop Amplitude. The Time Duration is Uniformly Set to $1ms$ for Plotting Purposes.

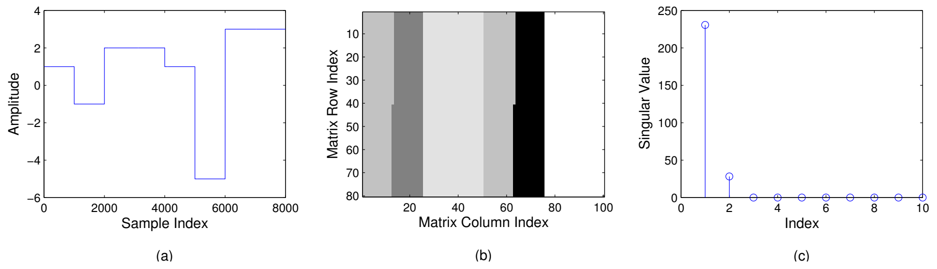


Figure 6.5: Stacking a Clean Signal **c** Into a Matrix **C**. (a) Noise Free Step Signal. (b) Scaled Image of Matrix **C**. (c) Singular Values of Matrix **C**. Figure Shows that the Rank of Matrix **C** is 2.

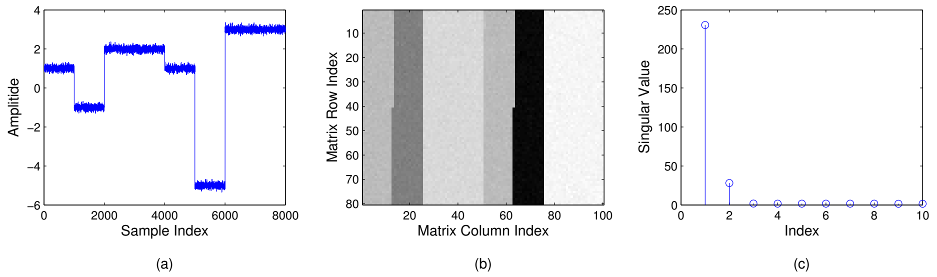


Figure 6.6: Stacking Noisy Signal **d** into a Matrix **D**. (a) Noisy Step Signal. (b) Scaled Image of Matrix **D**. (c) Singular Values of Matrix **D**. Figure Shows that the Rank of Matrix **D** is Approximately 2.

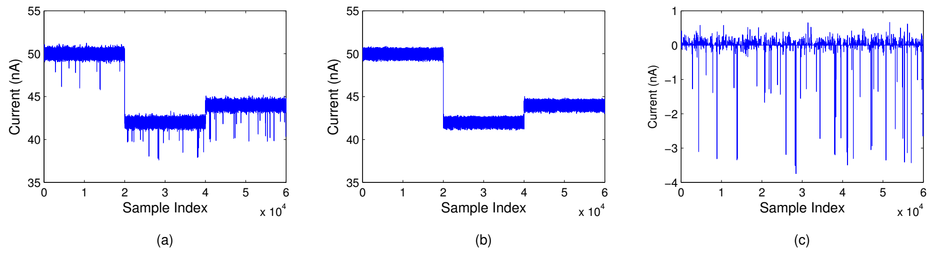


Figure 6.7: Signal Decompositions Using Original Formulations of RPCA. (a) Original Signal Containing Trapping and Translocation Events. (b) Baseline Signal Containing Only Trapping Events Obtained From Low Rank Component. (c) Candidate Translocation Events (With Positive and Negative Amplitudes) Obtained From Sparse Component.

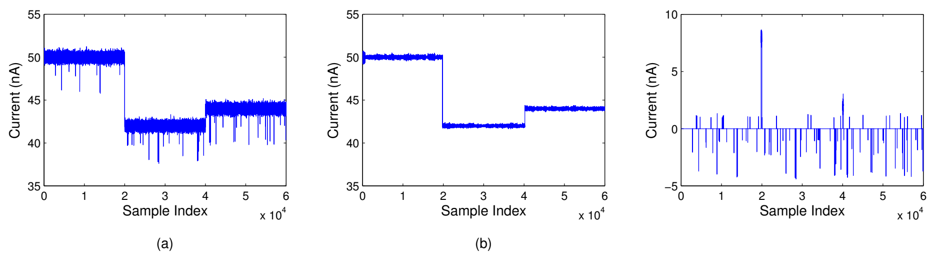


Figure 6.8: Signal Decompositions Using Original Formulations of GoDec. (a) Original Signal Containing Trapping and Translocation Events. (b) De-Noised Baseline Signal Containing Only Trapping Events Obtained From Low Rank Component. (c) Candidate Translocation Events (With Positive and Negative Amplitudes) Obtained From Sparse Component.

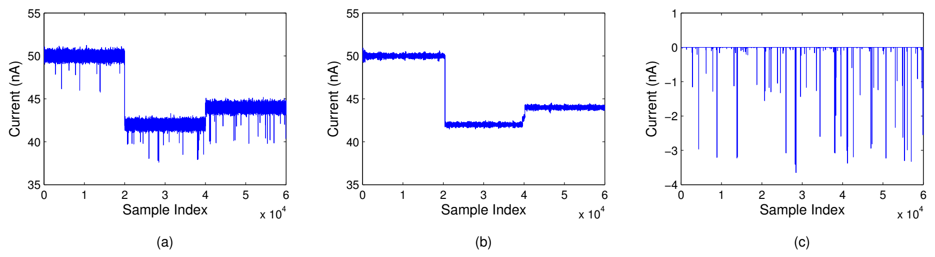


Figure 6.9: Signal Decompositions Using DWT and NpRPCA. (a) Original Signal Containing Trapping and Translocation Events. (b) De-Noised Baseline Signal Containing Only Trapping Events Obtained From Low Rank Component. (c) Candidate Translocation Events (With Only Negative Amplitudes) Obtained From Sparse Component.

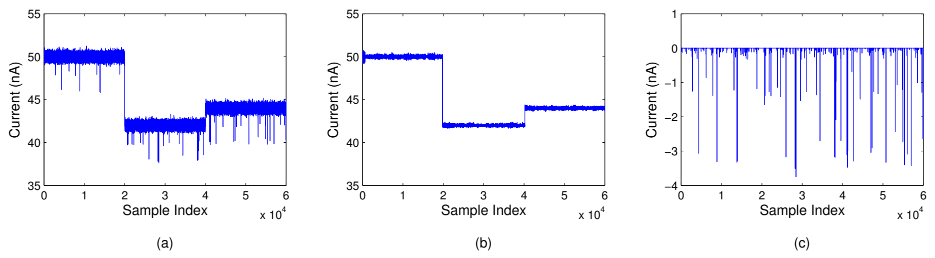


Figure 6.10: Signal Decompositions Using NpGodec. (a) Original Signal Containing Trapping and Translocation Events. (b) De-Noised Baseline Signal Containing Only Trapping Events Obtained From Low Rank Component. (c) Candidate Translocation Events (With Only Negative Amplitudes) Obtained From Sparse Component.

Chapter 7

SUMMARY AND FUTURE WORK

7.1 Summary

In this dissertation, different models to discover and improve feature representations for data were explored. Hand engineering features using domain knowledge is time-consuming and does not scale well to novel data and tasks. Recently, there have been many efforts to generate data-driven representations using clustering and sparse models. We explored some of the data-driven and unsupervised models for analyzing data. We propose methods to exploit the structure in the raw data and transformed data for supervised tasks such as recognition, detection and estimation. We began with the use of transform domain methods which find parsimonious representations for time-series data using fixed basis functions. We show that the transform domain based features contains discriminatory information and works well for simple two-state noise-free signal. Subspace based models such as sparse models and low-rank models can be used to improve the features which exploit the structure in the available data. We presented several models and algorithms using these paradigms and demonstrated improved performance in various inference tasks. Moving from simple time-series signals to complex audio signals a deep-learning based architecture was developed tuned to work for natural environment sounds. Sparse coding is an important and computationally expensive component in the models developed. To utilize larger datasets for learning and inference, there is a need to speed this step. A fast GPU optimized OMP algorithm was designed and implemented to this end. A detailed summary of the findings in this dissertation and possible future directions are presented below.

We built and improved transform domain features for ion-channel time-series signals. We proposed a classification setup to discriminate between different ion-channels using novel transform domain features obtained from their signals. Power distribution features, extracted from the frequency/sequency domains, are proposed that can effectively discriminate different ion-channel signals. We showed that the Fourier power spectra of an ideal ion-channel signal is directly dependent on the eigen decomposition of the state transition matrix characterizing the ion-channel. The features proposed based on the Fourier, wavelet and Walsh power spectra are suitable even for noisy data. Denoising the signals before extracting the features lead to an improved performance under high noise conditions. The best achievable performance obtained using HMMs with known ion-channel parameters is also presented as a baseline result for comparison.

We proposed a method to stabilize the PSD features for ion-channels using matrix completion. The performance of the robust PSD features were tested in a classification setup and a regression based analyte detection framework. The proposed features achieved better classification rates on the synthetic two class QUB data. These features were also used to analyze the signals obtained from the four chamber ion-channel sensor array device and detect Amplicillin. The features were decomposed into low-rank and sparse components to capture the group behaviour and also give importance to intra-group variation. Lower false detection rates were observed in the case of the improved features.

Simultaneous segmentation and feature extraction approaches for silicon-pores sensor data were considered. Low rank plus sparse matrix decomposition methods provide an efficient way for extracting sparse events and signal de-noising in silicon sensor signals. The redundancy across the segments of signals are exploited for analysis. Several variants of RPCA and GoDec suitable for this data were proposed. The proposed NpGoDec and NpRPCA method performs better than NpRPCA, coupled with DWT de-noising, in classification and clustering of events from silicon-pore signals. Furthermore, the shape of the drop events are well preserved in the former method. Analysis of the shape and duration of these current events enables us to estimate the properties of analytes. Spectral clustering was used to cluster the events in the feature domain as the shape of the clusters is unknown.

To generate features suitable for environment sound recognition we developed a feature learning architecture which preserves information about their highly dynamic nature. Environment sounds can pack complex semantics in a short time duration. For example, a short clip can comprise of a bird chirping, moving traffic, and people talking. We showed that, it is beneficial to partition the filters into groups, and learn topic models for each group. This ensemble representation provides more flexibility and robustness to the feature extraction algorithm. By incorporating this novel pooling strategy into a deep architecture, we obtained features that outperform other commonly adopted audio features. Algorithms were presented to perform partitioning of the feature space based on the temporal dynamics of the corresponding filters. The partitioning provides a multi-view representation of the data and makes learning computationally tractable in the subsequent layers. Ensemble of semantic embeddings using the multiple partitions improved tag prediction performance and helped in discovering semantics and clusters of tags. Proposed features outperformed traditional spectral features on challenging environmental sound recognition datasets namely Freefield1010 and IEEE AASP D-CASE.

With sparse coding as a key components in many of our algorithms we considered speeding up the OMP algorithm with GPUs. Specifically, several strategies were proposed to speed-up Orthogonal Matching Pursuit algorithm using CUDA kernel on a GPU are proposed. We demonstrated the use of sparse code from image patches to perform content based image retrieval. Implementations are developed as part of a large scale image retrieval system. Image-based exact search and visually similar search using the image patch sparse codes are performed. Results demonstrate large speed-up over CPU implementations and good retrieval performance was also achieved.

7.2 Future Directions

Exploring structure in data leads to developing representations which can encode this structure. Efficient representations can lead to disentanglement of factors of variation and thus improve the performance of the inference mechanisms. This can be done by using only labelled examples using supervised algorithms. The inherent generic structure can be discovered using unsupervised algorithms. These are beneficial when the label data is scarce and we intend to use the models learned to new datasets and novel tasks.

The unsupervised algorithms often encode the prior information in their structure, either explicitly or implicitly. Sparse coding assumes a union of subspaces model while manifold models assume that the data lies close to low-dimensional subspace. It will be useful to isolate these prior so that we can develop guarantees on performance and further tune them for different applications. Similarity, deep architectures encode the hierarchy information through depth and build invariances through pooling operations. The effect of various different pooling methods needs to be investigated from a theoretical point of view. It will be useful to devise schemes and metrics to measure the effect of the architectural changes such as pooling size and the pooling function as well as the depth of the network.

Textual information or tags can be a useful meta-data for multi-media data. Large scale retrieval becomes feasible by associating data with tags. Many a time users submit a textual query for image retrieval and hence tagging plays an important role in identifying images most relevant to the query. Furthermore, categorizing audio and video semantically can benefit from tag information. Human annotation is highly subjective and imprecise. Based on individual perception, the same image can be associated with different tags. Hence, this tag annotation is variable and may not often convey the true semantic meaning of the image. It becomes a herculean task for humans to annotate images if the image database is large. The research presented in this dissertation can be extended with efficient algorithm and software implementations to build a real-time automatic tag annotation framework. It will be useful to incorporate suitable modifications to the algorithms to obtain higher precision or higher recall, depending on the requirements of the application, without significantly affecting the other.

The prediction of tags for a test point was posed as an reconstruction problem. The sparse vectors were embedded into a low-dimensional space such that the mapping between this space and the feature space is smooth. The predictions are first performed in this embedded space and knowing that the full tag vector will be sparse we can pose the tag prediction as a sparse coding problem. This mapping of tags performed using only the features at the final layer. It will be be worthwhile to build and investigate a system which can learn the embeddings at multiple layers and find semantic relationships at different levels of hierarchy in the data representations.

The analysis of the the embedded tag vectors will be beneficial. It can be useful to understand the complex relationships between feature components and the appropriate semantic concepts. As an extension, the embeddings can be explicitly designed to attain algebraic or compositional properties to facilitate complex keyword based searches.

REFERENCES

[1] D. Giannoulis, E. Benetos, D. Stowell, M. Rossignol, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: An IEEE AASP challenge," in *IEEE Workshop on WASPAA*, 2013, pp. 1–4.

[2] H. Lee, "Tutorial on deep learning and applications," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learni.* Citeseer, 2010.

[3] S. Chachada and C.-C. J. Kuo, "Environmental sound recognition: A survey," in *IEEE APSIPA*, 2013, pp. 1–9.

[4] Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger *et al.*, "Comparison of learning algorithms for handwritten digit recognition," in *International conference on artificial neural networks*, vol. 60, 1995.

[5] A. Coates and A. Y. Ng, "Selecting receptive fields in deep networks," in *Advances in Neural Information Processing Systems*, 2011, pp. 2528–2536.

[6] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck, "Temporal pooling and multiscale learning for automatic annotation and ranking of music audio." in *ISMIR*, 2011, pp. 729–734.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning.* NJ, USA: Springer-Verlag New York, Inc., 2006.

[8] A. Coates and A. Y. Ng, "Learning feature representations with k-means," in *Neural Networks: Tricks of the Trade.* Springer, 2012, pp. 561–580.

[9] K. N. Ramamurthy, J. J. Thiagarajan, P. Sattigeri, M. Goryll, A. Spanias, T. Thornton, and S. M. Phillips, "Transform domain features for ion-channel signal classification," *Biomedical Signal Processing and Control*, vol. 6, no. 3, pp. 219–224, 2011.

[10] P. Sattigeri, J. Thiagarajan, K. N. Ramamurthy, A. Spanias, M. Goryll, and T. Thornton, "Robust psd features for ion-channel signals," in *Sensor Signal Processing for Defence (SSPD 2011).* IET, 2011, pp. 1–5.

[11] P. Sattigeri, J. Thiagarajan, M. Shah, K. N. Ramamurthy, and A. Spanias, "A scalable feature learning and tag prediction framework for natural environment sounds," in *Asilomar*, 2014.

[12] P. Sattigeri, J. J. Thiagarajan, K. N. Ramamurthy, and A. Spanias, "Implementation of a fast image coding and retrieval system using a gpu," in *Emerging Signal Processing Applications (ESPA), 2012 IEEE International Conference on.* IEEE, 2012, pp. 5–8.

[13] P. Sattigeri, J. Thiagarajan, K. N. Ramamurthy, A. Spanias, M. Goryll, and T. Thornton, "De-noising and event extraction for silicon pore sensors using matrix decomposition," in *Sensor Signal Processing for Defence (SSPD 2012)*. IET, 2012, pp. 1–5.

[14] P. Stoica and R. Moses, *Introduction to spectral analysis*. Prentice-Hall, 1997.

[15] Z. Chen, X. Guo, X. Zhang, W. Cram, and Z. Li, "A novel method for analysis of single ion channel signal based on wavelet transform," *Computers in Biology and Medicine*, vol. 37, pp. 559–562, 2007.

[16] B. Hille, *Ionic Channels of Excitable Membranes*. Sinauer Associates Inc., 1991.

[17] L.-Q. Gu, O. Braha, S. Conlan, S. Cheley, and H. Bayley, "Stochastic sensing of organic analytes by a pore-forming protein containing a molecular adapter," *Nature*, vol. 398, no. 6729, pp. 686–690, 1999.

[18] F. Qin, A. Auerbach, and F. Sachs, "A direct optimization approach to hidden markov modeling for single channel kinetics," *Biophysical Journal*, vol. 79, no. 4, pp. 1915–1927, 2000.

[19] J. R. Barry, E. A. Lee, and D. G. Messerschmitt, *Digital communications*. Springer, 2004.

[20] B. Vidakovic, *Statistical modeling by wavelets*. John Wiley & Sons, 2009, vol. 503.

[21] I. H. Dinstein and T. Silberberg, "Average walsh power spectrum for periodic signals," *IEEE Transactions on Electromagnetic Compatibility*, no. 4, pp. 407–412, 1981.

[22] J. Gibbs and F. Pichler, "Comments on transformation of" fourier" power spectra into" walsh" power spectra," *IEEE Transactions on Electromagnetic Compatibility*, no. 3, pp. 51–54, 1971.

[23] D. J. Field, "What is the goal of sensory coding?" *Neural Computation*, vol. 6, pp. 559–601, 1994.

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

[25] J. J. Thiagarajan, K. N. Ramamurthy, and A. Spanias, "Sparse representations for pattern classification using learned dictionaries," *Proceedings of Twenty-eighth SGAI International Conference on Artificial Intelligence*, 2008.

[26] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer, 2010.

[27] B. A. Olshausen and D. J. Field, “Sparse coding with an overcomplete basis set: A strategy employed by V1?” *Vision Research*, vol. 37, no. 23, pp. 3311–3325, Dec. 1997.

[28] J. A. Tropp, “Greed is good: Algorithmic results for sparse approximation,” *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.

[29] E. Cands and B. Recht, “Exact matrix completion via convex optimization,” *Foundations of Computational Mathematics*, vol. 9, pp. 717–772, 2009, 10.1007/s10208-009-9045-5. [Online]. Available: http://dx.doi.org/10.1007/s10208-009-9045-5

[30] G. Hinton, “A practical guide to training restricted boltzmann machines,” *Momentum*, vol. 9, p. 1, 2010.

[31] U. von Luxburg, “A tutorial on spectral clustering,” *Statistics and Computing*, vol. 17, pp. 395–416, 2007.

[32] M. Meila and J. Shi, “A random walks view of spectral segmentation,” in *8th International Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2001.

[33] A. Y. Ng, M. I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” in *Proc. of Neural Info. Processing Systems*, 2001.

[34] T. Serre, L. Wolf, and T. Poggio, “Object recognition with features inspired by visual cortex,” in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2. IEEE, 2005, pp. 994–1000.

[35] T. S. Lee, D. Mumford, R. Romero, and V. A. Lamme, “The role of the primary visual cortex in higher level vision,” *Vision research*, vol. 38, no. 15, pp. 2429–2454, 1998.

[36] I. Nelken, “Processing of complex stimuli and natural scenes in the auditory cortex,” *Current opinion in neurobiology*, vol. 14, no. 4, pp. 474–480, 2004.

[37] J. Sharma, A. Angelucci, and M. Sur, “Induction of visual orientation modules in auditory cortex,” *Nature*, vol. 404, no. 6780, pp. 841–847, 2000.

[38] S. Sukhbaatar, T. Makino, and K. Aihara, “Auto-pooling: Learning to improve invariance of image features from image sequences,” *arXiv preprint arXiv:1301.3323*, 2013.

[39] Y.-L. Boureau, J. Ponce, and Y. LeCun, “A theoretical analysis of feature pooling in visual recognition,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 111–118.

[40] A. Saxe, P. W. Koh, Z. Chen, M. Bhand, B. Suresh, and A. Y. Ng, “On random weights and unsupervised feature learning,” in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 1089–1096.

[41] H. Bayley, O. Braha, and L.-Q. Gu, "Stochastic sensing with protein pores," *Advanced Materials*, vol. 12, no. 2, pp. 139–142, 2000.

[42] A. Spanias, S. Goodnick, T. Thornton, S. Phillips, S. Wilk, and H. Kwon, "Signal processing for silicon ion-channel sensors," in *Signal Processing Applications for Public Security and Forensics, 2007. SAFE'07. IEEE Workshop on.* IET, 2007, pp. 1–4.

[43] D. Colquhoun and F. J. Sigworth, *Fitting and statistical analysis of single channel records.* New York: Plenum Press, 1983, pp. 191–264.

[44] F. Qin, A. Auerbach, and F. Sachs, "Estimating single-channel kinetic parameters from idealized patch-clamp data containing missed events," *Biophysical journal*, vol. 70, no. 1, pp. 264–280, 1996.

[45] F. Qin, "Restoration of single-channel currents using the segmental k-means method based on hidden markov modeling," *Biophysical journal*, vol. 86, no. 3, pp. 1488–1501, 2004.

[46] B. Konnanath, P. Knee, A. Spanias, and G. Wichern, "Classification of ion-channel signals using neural networks," in *Proceedings of Signal Processing, Pattern Recognition and Applications*, 2009.

[47] M. Diserbo, P. Masson, P. Gourmelon, and R. Catérini, "Utility of the wavelet transform to analyze the stationarity of single ionic channel recordings," *Journal of neuroscience methods*, vol. 99, no. 1, pp. 137–141, 2000.

[48] M. Kreir, C. Farre, M. Beckler, M. George, and N. Fertig, "Rapid screening of membrane protein activity: electrophysiological analysis of ompf reconstituted in proteoliposomes," *Lab on a chip*, vol. 8, no. 4, pp. 587–595, 2008.

[49] S. Wilk, L. Petrossian, M. Goryll, T. Thornton, S. Goodnick, J. Tang, and R. Eisenberg, "Integrated electrodes on a silicon based ion channel measurement platform," *Biosensors and Bioelectronics*, vol. 23, no. 2, pp. 183–190, 2007.

[50] M. Goryll and N. Chaplot, "Miniaturized silicon apertures for lipid bilayer reconstitution experiments," in *MRS Proceedings*, vol. 1191, no. 1. Cambridge Univ Press, 2009.

[51] F. Sigworth, "Electronic design of the patch clamp," *Single-channel recording*, pp. 95–127, 2009.

[52] J. Dempster, "A new version of the strathclyde electrophysiology software package running within the windows environment," *Journal of Physiology-London*, vol. 504P, p. 57, 1997.

[53] L. Venkataramanan, J. L. Walsh, R. Kuc, and F. J. Sigworth, "Identification of hidden markov models for ion channel currents. i. colored background noise," *Signal Processing, IEEE Transactions on*, vol. 46, no. 7, pp. 1901–1915, 1998.

[54] D. L. Donoho and J. M. Johnstone, “Ideal spatial adaptation by wavelet shrinkage,” *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[55] “QuB software,” Available online at: http://www.qub.buffalo.edu/.

[56] C. J. C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.

[57] J. F. Cai, E. J. Candes, and Z. Shen, “A Singular Value Thresholding Algorithm for Matrix Completion,” *ArXiv e-prints*, 2008.

[58] Z. Lin, M. Chen, and Y. Ma, “The Augmented Lagrange Multiplier Method for Exact Recovery of Corrupted Low-Rank Matrices,” *ArXiv e-prints*, Sep. 2010.

[59] R. H. Keshavan, A. Montanari, and S. Oh, “Matrix completion from noisy entries,” *J. Mach. Learn. Res.*, vol. 99, pp. 2057–2078, August 2010. [Online]. Available: http://portal.acm.org/citation.cfm?id=1859890.1859920

[60] K. N. Ramamurthy, J. J. Thiagarajan, P. Sattigeri, M. Goryll, A. Spanias, T. Thornton, and S. Phillips, “Transform domain features for ion-channel signal classification,” *Biomedical Signal Processing and Control*, vol. In Press, Corrected Proof, Available online 29 October 2010. [Online]. Available: http://www.sciencedirect.com/science/article/B7XMN-51BPDW6-1/2/5688c01402ddb9551a688dbf315d107d

[61] P. Sattigeri, K. N. Ramamurthy, J. J. Thiagarajan, M. Goryll, A. Spanias, and T. Thornton, “Analyte detection using an ion-channel sensor array,” in *17th International Conference on Digital Signal Processing*, 2011.

[62] A. V. Jagtiani, R. Sawant, J. Carletta, and J. Zhe, “Wavelet transform-based methods for denoising of coulter counter signals,” *Meas. Sci. Technol.*, vol. 19, pp. 1–15, 2008.

[63] D. L. Donoho, “De-noising by soft thresholding,” *IEEE Trans. on Information Theory*, vol. 41, pp. 613–627, 1995.

[64] G. Y. Chen and G. Dudek, “Auto-correlation wavelet support vector machine,” *Image and Vision Computing*, vol. 27, no. 8, pp. 1040 – 1046, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/B6V09-4TK92BX-2/2/ab213572262db7ae725338097dac548f

[65] M. Karbasi, S. Ahadi, and M. Bahmanian, “Environmental sound classification using spectral dynamic features,” in *IEEE ICICS*, 2011, pp. 1–5.

[66] Q. V. Le, “Building high-level features using large scale unsupervised learning,” in *IEEE ICASSP*. IEEE, 2013, pp. 8595–8598.

[67] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks.” in *NIPS*, vol. 1, no. 2, 2012.

[68] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *IEEE CVPR*, vol. 2. IEEE, 2006, pp. 2169–2178.

[69] J. J. Thiagarajan, K. N. Ramamurthy, and A. Spanias, "Mixing matrix estimation using discriminative clustering for blind source separation," *Digital Signal Processing*, vol. 23, no. 1, pp. 9–18, 2013.

[70] S. Dasgupta, "Experiments with random projection," in *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence.* Morgan Kaufmann Publishers Inc., 2000, pp. 143–151.

[71] R. Salakhutdinov and G. E. Hinton, "Replicated softmax: an undirected topic model." in *NIPS*, vol. 22, 2009, pp. 1607–1614.

[72] X. He and P. Niyogi, "Locality preserving projections," in *NIPS*, vol. 16, 2003, pp. 234–241.

[73] D. Stowell and M. D. Plumbley, "An open dataset for research on audio field recording archives: freefield1010," *arXiv preprint arXiv:1309.5275*, 2013.

[74] R. Raina, A. Madhavan, and A. Y. Ng, "Large-scale deep unsupervised learning using graphics processors," in *Proc. ICML*, 2009, pp. 873–880.

[75] J. Owens, M. Houston, D. Luebke, S. Green, J. Stone, and J. Phillips, "GPU computing," *Proc. IEEE*, vol. 96, no. 5, pp. 879 –899, May 2008.

[76] V. Pan and R. Schreiber, "An improved Newton interaction for the generalized inverse of a matrix with applications," *SIAM J. Sci. Stat. Comput.*, vol. 12, pp. 1109–1130, Sep. 1991. [Online]. Available: http://portal.acm.org/citation.cfm?id=114678.114687

[77] R. Rubinstein, M. Zibulevsky, and M. Elad, "Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit," CS Technion, Tech. Rep., 2008.

[78] M. Solli, "Topics in content based image retrieval," Nov. 2009.

[79] "Tineye.com," Available online at http://www.tineye.com/.

[80] J. Hays and A. Efros, "IM2GPS: estimating geographic information from a single image," in *IEEE CVPR*, Jun. 2008, pp. 1 –8.

[81] S. N. et al., "Multiple kernel learning for object classification," in *Proceedings of the 12th Workshop on Information-based Induction Sciences*, 2009.

[82] S. Z. et al., "Automatic image annotation using group sparsity," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on.* IEEE, 2010, pp. 3312–3319.

[83] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[84] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[85] Y. G. et al., "A multi-view embedding space for modeling internet images, tags, and their semantics," *arXiv preprint arXiv:1212.4522*, 2012.

[86] C. W. et al., "Multi-label sparse coding for automatic image annotation," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009, pp. 1643–1650.

[87] M. G. et al., "Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 309–316.

[88] P. D. et al., "Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary," in *Proceedings of the 7th European Conference on Computer Vision-Part IV*, ser. ECCV '02, 2002, pp. 97–112.

[89] J. M. et al., "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[90] T. Ito, L. Sun, R. R. Henriquez, and R. M. Crooks, "A carbon nanotube-based coulter nanoparticle counter," *Acc. Chem*, vol. 37, pp. 937–945, 2004.

[91] L. Petrossian, "Cylindrical solid state nanopores," Ph.D. dissertation, Arizona State University.

[92] S. J. Wilk, M. Goryll, G. M. Laws, S. M. Goodnick, T. J. Thornton, M. Saraniti, J. Tang, and R. S. Eisenberg, "Teflon (tm)-coated silicon apertures for supported lipid bilayer membranes," *Appl. Phys Lett*, vol. 85, pp. 3307–3309, 2004.

[93] H. Bayley and C. R. Martin, "Resistive-pulse sensing-from microbes to molecules," *Chemical Rev*, vol. 100, pp. 2575–2594, 2000.

[94] T. Mathew, P. Joshi, S. Prasad, M. Goryll, A. Spanias, and T. Thornton, "Silicon-based pore systems for emerging biosensor applications," in *Proceedings of the 2009 ASME International Mechanical Engineering Congress and Exposition*, 2009.

[95] P. Joshi, T. Mathew, L. Petrossian, S. Prasad, M. Goryll, A. Spanias, and T. Thornton, "Electromigration of charged polystyrene beads through silicon nanopores filled with low ionic strength solutions," in *Proceedings of the 2009 ASME International Mechanical Engineering Congress and Exposition*, 2009.

[96] H. Kwon, P. Knee, A. Spanias, S. Goodnick, T. Thornton, and S. Phillips, "Transform-domain features for ion-channel sensors," in *Signal Processing, Pattern Recognition, and Applications*, 2008.

[97] K. N. Ramamurthy, J. J. Thiagarajan, P. Sattigeri, B. Konnanath, A. Spanias, T. Thornton, S. Prasad, M. Goryll, S. Phillips, and S. Goodnick, "Transform domain features for ion-channel signal classification using support vector machines," in *9th International Conference on Information Technology and Applications in Biomedicine*, 2009.

[98] B. Konnanath, P. Sattigeri, T. Mathew, A. Spanias, S. Prasad, M. Goryll, T. Thornton, and P. Knee, "Acquiring and classifying signals from nanopores and ion-channels," in *IEEE ICANN 2009*, 2009.

[99] J. Dempster, *The Laboratory Computer: A guide for neuroscientists and physiologists.* Academic Press, London, 2001.

[100] B. D. Ripley, *Pattern Recognition and Neural Networks.* Cambridge University Press, 1996.

[101] T. Ferguson, "A bayesian analysis of some nonparametric problems," *Annals of Statistics*, vol. 1, pp. 209–230, 1973.

[102] C. E. Rasmussen, "The infinite gaussian mixture model." *Advances in Neural Information Processing Systems*, vol. 12, pp. 554–560, 2000.

[103] A. Ahmed and E. Xing, "Dynamic non-parametric mixture models and the recurrent chinese restaurant process: with applications to evolutionary clustering," *InSDM*, 2008.

[104] Neal and M. Radford, "Markov chain sampling methods for dirichlet process mixture models," *Technical Report 9815 Uni- versity of Toronto, Department of Statistics and Department of Computer Science*, 1998.

[105] M. A. T. Figueiredo, "Unsupervised learning of finite mixture models," *Pattern Analysis and Machine Intelligence*, pp. pp.381–396, 2002.

[106] P. D. Grunwald, "Minimum description length tutorial," *Advances in Minimum Description Length: Theory and Applications*, 2005.

[107] J. Rissanen, "Hypothesis selection and testing by the mdl principle," *The Computer Journal*, vol. 42, p. 260269, 1999.

[108] D. L. Donoho, "De-noising by soft-thresholding," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 613 –627, 1995.

[109] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *CoRR*, vol. abs/0912.3599, 2009.

[110] T. Zhou and D. Tao, "Godec: Randomized low-rank and sparse matrix decomposition in noisy case," in *Proc. of International Conference on Machine Learning*, 2011.

[111] M. A. Little and N. S. Jones, "Sparse bayesian step-filtering for high-throughput analysis of molecular machine dynamics," in *Proc. of International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2010, pp. 4162 –4165.

[112] P. Sattigeri, J. J. Thiagarajan, K. N. Ramamurthy, P. Joshi, A. Spanias, M. Goryll, and T. Thornton, “Analysis of coulter counting data from nanopores using clustering,” in *Proc. of Sensor Signal Processing for Defence*, 2010.

[113] P. D. Grunwald, “Minimum description length tutorial,” *Advances in Minimum Description Length: Theory and Applications*, 2005.