The Pennsylvania State University
The Graduate School

SEMISUPERVISED ACTIVE LEARNING AND GROUP ANOMALY DETECTION WITH UNKNOWN OR LABEL-SCARCE CATEGORIES

A Dissertation in
Electrical Engineering
by
Zhicong Qiu

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2017

The dissertation of Zhicong Qiu was approved [1] by the following:

David Miller
Professor of Electrical Engineering
Dissertation Advisor
Chair of Committee

George Kesidis
Professor of Electrical Engineering and Computer Science Engineering

John Doherty
Professor of Electrical Engineering

Sencun Zhu
Associate Professor of Computer Science Engineering

Kultegin Aydin
Department Head and Professor of Electrical Engineering

[1] Signatures on file in the Graduate School.

Abstract

This dissertation makes contributions to two major areas in machine learning, namely, semi-supervised active learning and anomaly detection, with general applicability but with demonstrated application to vehicle tracking and network intrusion detection. In both of these domains, some categories may be rare or unknown, with very few or no labeled samples to start with. For example, in a network intrusion detection system, following a standard statistical anomaly detection (AD) approach, one would train a null hypothesis, characterizing the normal behavior whose data sources are usually captured in a sandbox environment, and flag any sample that deviates from the norm (above a predefined threshold) as anomalies. Naively deploying the null model, however, will flood the administrator with too many uninteresting anomalies to further fully investigate. Thus, it makes sense to further discriminate many of the uninteresting anomalies from the truly interesting ones, and develop an active selection strategy to efficiently forward samples for oracle labeling that help to learn to discriminate these groups. Moreover, the interesting anomalies, which may amount to zero-day threats, are often highly skewed and may only manifest on a very small subset of features. Using all features to identify anomalies will not be advantageous because many features may not be informative/discriminative in practice. We try to design a rare category identification and characterization system that can benefit from 1) statistical AD, 2) a semi-supervised active learning based discriminant classifier, with zero weights given to the irrelevant features, and 3) an active sample selection strategy to select the most likely unknown sample, hence efficiently ranking/forwarding interesting anomalies for the administrator. In terms of 1), we also develop a purely unsupervised learning technique to extract group behavior that jointly exhibits anomalousness on a sample and feature subset.

In the case of semi-supervised learning, wherein the fundamental idea is to combine the usage of the limited number of labeled training samples and the abundance of unlabeled samples to learn a classifier, we focus on domains where some categories are rare or unknown, with very few or no labeled samples to start with. Unlike the conventional approaches in the mainstream literature, where unlabeled samples are perceived as belonging to one of the known categories and should be leveraged to minimize class

posterior uncertainty, we propose a semi-supervised objective that seeks to *preserve* the uncertainty among unlabeled samples, both to avoid overtraining and to help efficiently discover unknown classes. Specifically, using Shannon entropy as the measure of uncertainty, we propose to use max entropy regularization (maxEnt), rather than minimum entropy regularization (minEnt). Moreover, our proposed model in a two class classification problem is convex (unlike minEnt) and it has been shown to outperform in a variety of rare category characterization problems, compared to existing approaches. While semi-supervised learning focuses on exploiting the latent structures hidden in the sample distribution, active learning tries to explore the sample space with the least representation. We combine our maxEnt semi-supervised learning with a novel active learning strategy that together efficiently draw from the pool of unlabeled samples for oracle labeling, to achieve the best class discovery (exploration) and classification (exploitation).

In the case of anomaly detection, wherein the fundamental idea is to detect significant outliers that deviate from the one class or null hypothesis, we have two contributions to make. First, we propose a group based anomaly detection scheme that identifies the sample and feature subset that jointly manifest the potential anomalous group, with a Bonferroni approximation used to account for multiple testing. Unlike point-wise anomaly detectors, our model has the potential to jointly identify which of the sample and feature subsets are most atypical with respect to the null, thus avoiding most point-wise, superficial outliers and efficiently capturing group anomalies. Second, a singleton and pairwise Gaussian Mixture Models (GMMs) method is proposed as a novel feature representation to characterize atypicality on each single or pairwise dimension, using p-value as score/feature. This avoids the curse of dimensionality and achieves superior performance, compared to other non-informative feature mapping techniques in the literature.

Table of Contents

List of Figures viii

List of Tables x

Acknowledgments xi

Chapter 1
Introduction 1
1.1 Semi-supervised Learning 2
1.1.1 Data Dependent Regularization 3
1.2 Active Learning 4
1.3 Anomaly Detection 6
1.4 Other Related Work 6
1.5 Our Contributions 7

Chapter 2
Proposed MaxEnt Semi-supervised Active Learning Model 9
2.1 The Assumed Inductive Bias: the Unknown Class as a Subset of What is Anomalous 9
2.1.1 Discriminating Interesting from Uninteresting Anomalies 12
2.2 Comprehensive Low-order Null Modeling 13
2.2.1 Gaussian Mixture Model 14
2.2.2 Mixture-based P-Values as Derived Features 16
2.3 Classifier Model 17
2.3.1 Two Class Case 17
2.3.2 Extension for Multiple Common and Label-scarce Classes 19
2.3.3 Extension to Include Unknown Classes 20
2.4 Learning Objective Function for Semi-supervised and Active Learning . 21
2.5 Active Learning: Sample Selection Criteria for Oracle Labeling 27

2.6 Experimental Results for Semi-supervised Active Learning and New Class Discovery . . . 28
2.6.1 Performance Metrics . . . 28
2.6.2 Data Set Description . . . 29
2.6.3 Mixture P-values as Features for Supervised Learning . . . 29
2.6.4 Semi-supervised Learning Evaluation . . . 31
2.6.4.1 Majority v. Minority – a Two Class Classification Study 31
2.6.4.2 Multiple Common and Rare Classes . . . 32
2.6.5 Evaluating Method Variations . . . 34
2.6.5.1 Majority v. Minority – a Two Class Classification Study 34
2.6.5.2 Multiple Common and Rare Classes . . . 35
2.6.6 Active Learning Evaluation . . . 37
2.7 Conclusion . . . 40

Chapter 3
Applications of The Semi-supervised Active Learning with Maximum Entropy Regularization 48
3.1 Active Learning to Distinguish Suspicious from Innocuous Anomalies in a Batch of Vehicle Tracks . . . 48
3.1.1 Track Data and Its Raw Features . . . 50
3.1.2 P-Values as Features for Discriminating Suspicious from Merely Anomalous . . . 51
3.1.3 Classifier Model . . . 52
3.1.4 Semi-supervised Active Learning Objective Function . . . 53
3.1.5 Sample Selection Criteria for Oracle Labeling . . . 56
3.1.6 Experimental Results . . . 57
3.1.6.1 Performance Metrics . . . 58
3.1.6.2 Compare Different Alphas . . . 58
3.1.6.3 Compare Sample Selection Strategies . . . 59
3.1.6.4 Evaluation of Semi-supervised Learning . . . 60
3.1.6.5 Value of unlabeled tracks . . . 61
3.1.6.6 Weights on The Labeled Subsets . . . 62
3.1.6.7 Comparison with Support Vector Machines (SVMs) . . 62
3.1.7 Conclusion . . . 63
3.2 Flow Based Botnet Detection through Semi-supervised Active Learning 64
3.2.1 Problem Definition and Related Work . . . 66
3.2.2 Methodology . . . 68
3.2.2.1 Feature-Space Representation . . . 69
3.2.2.2 Anomaly Based Derived Features . . . 70
3.2.2.3 Classification Model and Learning Objective . . . 72
3.2.2.4 Active Learning Strategy . . . 74
3.2.3 Experimental Setup and Results . . . 74

3.2.3.1 Performance Metrics . . . 75
3.2.3.2 Experimental Results . . . 76
3.2.4 Discussion and Future Work . . . 79

Chapter 4
Detecting Clusters of Anomalies on Low-Dimensional Feature Subsets with Application to Network Traffic Flow Data **82**
4.1 Introduction . . . 82
4.2 Problem Definition and Related Work . . . 83
4.3 Proposed Model . . . 84
4.3.1 Mixture-based P-values for Singletons and Feature Pairs . . . 84
4.3.2 Scoring Clusters . . . 86
4.3.3 Identifying the Optimal Sample Subset I_c, Given Fixed J_c . . . 88
4.3.4 Overall Search Algorithm . . . 88
4.4 Experimental Setup and Results . . . 89
4.4.1 Feature Space Selection and Representation . . . 89
4.4.2 Performance Metrics . . . 90
4.4.3 Experimental Results . . . 91
4.5 Conclusion . . . 92

Chapter 5
Conclusion and Future Work **94**
5.1 MixEnt: Joint Update of Target Distribution and Posteriors . . . 94

Bibliography **99**

List of Figures

2.1 A three-cluster example with decision boundaries trained by different objectives: (MLE) Maximum Likelihood Estimate; (L2) MLE with L2-norm; (minEnt) Minimum entropy with weight decay (L2) [20]; (maxEnt) Proposed. . . . 27
2.2 Supervised classification experiment: average ROC AUC performance comparison of different feature mappings input to linear and RBF kernel SVMs on various UCI data sets. . . . 42
2.3 Page Block: ROC AUC. . . . 43
2.4 Page Block: Entropy. . . . 43
2.5 Page Block: Weight vector magnitudes. . . . 43
2.6 Semi-supervised classification experiment: average ROC AUC performance comparison of different regularizers applied to learn our posterior model on various data sets. . . . 44
2.7 Performance comparison of proposed model variations on various data sets. . . . 45
2.8 Active learning experiment for different sample selection strategies. (a) Page Block. (b) Statlog Shuttle. (c) Yeast. (d) Wine Quality. (e) KDD'99. (f) Spam Base. . . . 46
2.9 Active learning experiment: comparison of methods' ROC AUC performance and average classification accuracy versus number of oracle labelings, on various data sets. (a) Page Block. (b) Statlog Shuttle. (c) Yeast. (d) Wine Quality. (e) KDD'99. (f) Spam Base. . . . 47

3.1 Compare ROC of different Ks in Track Data. . . . 52
3.2 A suspicious track making a U-turn. . . . 57
3.3 A suspicious track with a loop. . . . 58
3.4 ROC AUC for different alphas. . . . 59
3.5 Weight magnitude for different alphas. . . . 60
3.6 Number of true detections for different sample selection mechanisms. . . . 61
3.7 ROC AUC performance for different sample selection mechanisms. . . . 61
3.8 ROC AUC performance with/without use of unlabeled samples and with $(\frac{1}{2}, \frac{1}{2})$ unlabeled targets. . . . 62
3.9 ROC AUC for different labeled subset weighting schemes. . . . 63

3.10 ROC AUC performance of our semi-supervised method, compared with linear and Gaussian kernel SVMs. 64
3.11 The overall AL system design. 74
3.12 Active learning experiment: comparison of different feature representations. 79
3.13 Active learning experiment: comparison of different AL strategies. . . . 80

4.1 Synthetic data experiment: comparison of different schemes with 2 independent Gaussian based anomalous feature subsets in (separate) 1-dim subspace. 92
4.2 Network traffic data experiment: comparison of different schemes with P2P or Zeus anomalies. 93

5.1 A three-cluster example with pathological labeled sample distribution from the two known categories. Synthetic experiment 1. 95
5.2 A three-cluster example with randomly chosen labeled samples from the two known categories. Synthetic experiment 2. 96

List of Tables

2.1 Data Set Properties: (N) number of samples. (D) number of attributes. (K) number of classes. (K_r) number of rare classes. (SCP) smallest class proportion in percentage. 29
2.2 Semi-supervised experiment for the unknown class scenario: ROC AUC performance. 33
2.3 Semi-supervised experiment for the label-scarce class scenario: ROC AUC and average accuracy per-class. (P) Proposed. (M) minEnt. (L2) L2 regularization. 34
2.4 Sparsity of the maxEnt model: average number of non-zero weight parameters for semi-supervised learning with 5% minority class proportion. 35
2.5 Compare model variations: average classification accuracy per-class and fraction of inactive model parameters in Λ. (P) Proposed. (Uc) Unconstrained Parameters. (Uw) Uniform Weighting. (MLE) Maximum Likelihood. 35
2.6 AL true detections and fraction of rare classes discovered out of 100 iterations for different sample selection strategies: (MLU) most likely unknown. (MU) most uncertain. (LC) least common and (R) random. . . 39

3.1 Illustrative examples of bidirectional flow's feature representation for the first N transmitted packets. $N = 6, D = 12$. C_i denotes the i^{th} packet from client to server, whereas S_i denotes the i^{th} packet from server to client. The three-way hand shake packets are not used. 69
3.2 Normal web and botnet flow sizes. 75
3.3 Standard flow based feature representation [33] [5]. 77
3.4 ROC AUC results for various supervised learning models, comparing various feature representations. 78

5.1 Empirical Results on both synthetic data sets and real world data sets. (SYN1) Synthetic data set illustrated in Figure 5.1. (SYN2) Synthetic data set illustrated in Figure 5.2. (PB) Page Block. (SH) Shuttle Statlog. 97

Acknowledgments

I would like to thank my thesis advisor Dr. David J. Miller, who helped me write better and better in English, and is very supportive during my PhD study, giving me invaluable advice. I would also like to thank Dr. George Kesidis, who gave me a lot of opportunities to apply my machine learning skills to network intrusion detection and network failure recovery, working with CISCO and DTRA. I would also like to thank my family and my friends who continue to inspire me throughout my life journey.

Chapter 1

Introduction

There are two basic paradigms in machine learning. One is supervised and the other unsupervised. In supervised learning, data are collected and annotated (labeled), often by human experts. Given the data set $\mathcal{X}$ and the associated label set $\mathcal{Y}$, the task involves learning a relationship/mapping between variables x and y based on a functional model (e.g., decision tree) and optimization of an objective criterion (e.g., maximum likelihood). If y is a discrete variable, we call it a classification problem; otherwise we call it a regression problem. Supervised learning methods have been applied in various applications, such as text classification, predictive modeling, and disease/fraud prevention. On the other hand, in unsupervised learning, data are collected in bulk without human annotation (e.g., survey data), and the task involves exploring interesting clusters/structures hidden in the data. Some other applications also involve dimensionality reduction and data visualization. Semi-supervised learning (SL) – a learning method to perform either transduction or inductive reasoning using both labeled and unlabeled samples – is usually conceived as a method to exploit both the information hidden in the latent clusters of the data and the discriminability of different classes/categories. The research on SL has been motivated by applications where annotation of data is scarce compared to the bulk of the unlabeled data that are easy to collect, e.g., in natural language processing. A detailed survey on semi-supervised learning is presented in [7]. This thesis focuses on inductive semi-supervised learning, *i.e.*, the design of a semi-supervised learning model for general class discrimination, using both labeled and (potentially a much larger number of) unlabeled samples. In the sequel, we will provide a brief introduction on semi-supervised learning and specifically data dependent regularization – in which unlabeled data are utilized as a kind of regularization source.

1.1 Semi-supervised Learning

In semi-supervised learning, we are given a set of training data $\mathcal{X} = \{\mathcal{X}_l, \mathcal{X}_u\}$, with $\mathcal{X}_l$ the labeled training examples and $\mathcal{X}_u$ the unlabeled examples, and the goal is to learn a mapping $F(.)$ that generalizes the distribution of $\mathcal{X}$ and/or prediction of $\mathcal{Y}$. As noted in [7], the presence of unlabeled samples can only be made useful when samples are obtained unbiasedly. If the data collection process is biased, the input distribution information available from the unlabeled samples tends to hurt the generalization performance. Transductive based semi-supervised learning, e.g., transductive SVM [62], concerns only with inferring labels for the unlabeled examples in $\mathcal{X}_u$, whereas inductive based semi-supervised learning tries to learn class boundaries for general classification besides the unlabeled samples on the sample space.

Inductive based semi-supervised learning can be further categorized into two groups: *generative* and *discriminative* models. A generative model seeks to estimate the joint distribution $P[x, y]$, using both labeled and unlabeled samples, typically assuming samples are drawn i.i.d.. For example, in [41] and [44], a stochastic generation mechanism is hypothesized for the observed data, treating the labels of those unlabeled samples as missing values, with the associated expectation-maximization (EM) learning to well-fit the data based on a maximum likelihood estimate. One can further extend it into the Bayesian framework by introducing a prior on the parameter space. On the other hand, in a discriminative model, e.g., [7], [4], [62], the focus is on learning a class posterior $P[y|x]$ (e.g., logistic regression) or a discriminant function (e.g., SVM [62]) that can well separate data from the different classes. If the philosophical principle of Vapnik [62] (which leads to the development of transductive SVM) is to be realized, *i.e.*, one does not solve a harder problem as an intermediate step towards solving a simpler problem, then discriminative approaches, by modeling $P[y|x]$ explicitly, with the need to estimate a (much) less number of parameters, can be more attractive in the case of regression or classification, compared to the generative models. In addition, the objective measure in selecting the best generative model is not discriminative in nature, so the best generative model we can estimate in practice may not be the best for prediction, especially when there is model mismatch. However, a discriminative model will not model the underlying sample distribution and hence is not suitable for tasks related to density estimation which requires the explicit knowledge of $P[x]$.

Based on various semi-supervised learning approaches, there are various assumptions associated with each of the methods used:

- **Smooth label assumption** : In a high density region, the closer the input samples x_1 and x_2, the likelier its labels y_1 and y_2 are the same.
- **Cluster assumption** : Samples belonging to the same cluster are more likely to belong to the same class, compared to samples belonging to different clusters.
- **Low-density separation** : The separating hyperplane should be generated from a low-density region. Note that this assumption is related to that of the first and the second one, because by guaranteeing that the decision boundary cuts through a low density region, clustered samples and high density regions are more likely assigned to the same class.
- **Manifold assumption** : To avoid the curse of dimensionality in modeling data in high dimension, the manifold assumption states that the high-dimensional data can be perfectly characterized in a low-dimensional manifold. Semi-supervised graphical models implicitly encode this assumption, by modeling each sample as a node and weights among data as their pairwise similarity measures [7], [2],[64].

In this thesis, we do not explicitly impose any of the above-mentioned assumptions and develop our semi-supervised learning model with a different paradigm that is suitable for rare and unknown category detection and characterization. Our work is closely related to data dependent regularization, which is discussed in the sequel.

1.1.1 Data Dependent Regularization

In this section, we discuss using unlabeled data as a source of regularization for a discriminative classifier.

Semi-supervised learning wherein unlabeled samples are used to achieve a type of "regularization" have been proposed in a number of prior works [20],[36],[17],[19],[10]. Penalizing solutions that cut through a high density region, by postulating low-density separation, all of these approaches introduce a cost term on the unlabeled samples which encourages solutions with minimum entropy on the unlabeled samples, *i.e.*, consistent with "most confident" decisions [20] and class label "smoothness" in identifiable local regions [10]. This has also been related to maximizing classifier margin [20]. [19] goes one step further than [20] by introducing an empirical estimate of the class prior. The proposed objective function in [19] is shown to work well in both clustering and semi-supervised learning tasks. [36] extends [20] by adding an expected class estimate regularization term, whose prior belief can be provided by a domain expert if available.

However, a concern with these approaches is that they encourage overtraining in the case where there are too few or no labeled examples from rare categories, the phenomenon of which will be experimentally demonstrated in Chapter 3. Accordingly, we modified these approaches, essentially by changing the sign on the unlabeled term – rather than minimizing decision uncertainty, our approach encourages maximizing decision uncertainty so as to limit the amount of classifier learning performed based on a limited number of supervising minority examples and so as to aid in efficient discovery of unknown classes. Moreover, our optimization problem for the standard multi-class classification problem is convex, whereas the minimum entropy problem is not. While we impose the complete opposite assumption of minEnt, the proposed maxEnt framework allows for unidentified categories samples to be the most uncertain; hence it is highly suitable in detecting unknown unknowns (unknown classes, zero-day phenomenon).

1.2 Active Learning

The main idea behind active learning is to allow the learner to ask questions. To put this into a machine learning context, a learning machine can query unknown/unlabeled samples to an oracle who knows the nature of the problem and hence can provide ground truth labeling to the machine [55]. By judiciously *choosing* the samples to be labeled by an oracle, one can potentially reduce the number of labeled examples that would otherwise be needed to learn an accurate classifier for the given domain. Such economization is of great practical importance, considering that ground-truth labeling is often an expensive and time-consuming exercise. Active learning also gives a flexible, user-interactive learning capability for domains where categorizations are highly subjective, e.g., consider learning to predict the music (or art) that an idiosyncratic user would find "interesting" or "uninteresting". Active learning systems involve a closed-loop learning cycle, with the classifier's model parameters adapted/retrained in light of newly ground truth labeled examples, and with the resulting model in turn used to help select additional samples for labeling. The initial model, which starts this process, is generally learned based on a (relatively small) initially available labeled training set. We focus on *pool-based* AL [55], where the additional (actively) labeled samples are chosen from a captured batch (pool) of unlabeled samples. A pool-based AL system is fully determined by:

1. The features and associated feature representation used as input to the classifier.

2. The chosen classifier model structure (e.g., decision tree, logistic regression).

3. The criterion or algorithm used for selecting samples to be labeled (e.g., choosing the sample with the greatest class uncertainty (entropy)).

4. The training objective function and associated optimization method, used for adapting the classifier's parameters in light of each new labeled instance.

In this work, we develop an active learning system suitable for detecting and learning unknown or label scarce categories. There are few prior works that address active learning with rare/unknown categories. The active learning review in [55] gives little discussion of this problem. [12] proposes a sampling scheme which exploits cluster structure (and class purity of same) in the data to potentially improve active learning efficiency. They demonstrated their method on the *Statlog Shuttle Data* domain, for which the rarest category occurs with frequency just 0.014%. Their scheme reduced the number of labelings required to identify a labeled example from every class by a factor of almost ten, compared with random selection. The most focused, notable research effort, of which we are aware, on active learning with rare categories is [26]. This thesis separately addresses rare category detection (identifying the first labeled instances of all classes), rare category characterization (to subsequently identify all instances of a rare category), and a related *unsupervised* problem, where rare groups should be identified in the absence of class labels for any samples in the pool. There are some fundamental differences between the approaches taken in [26] and the maxEnt framework we propose. First, [26] focuses on identifying examples of the rare category – not on overall classification accuracy. For a two-class problem with one unknown category, the most valuable samples to label may be the samples that are nearest to the true (optimal) decision boundary. It need not be the case that such samples predominantly come from the unknown class. In fact, if the unknown class *is* a rare class, one would expect, within the unlabeled pool, that *more* common category samples are close to the decision boundary than rare category samples. Accordingly, we expect that the most effective active selection strategy (to optimize overall classification performance) should sample accordingly. In fact, this was borne out by our experimental results – prioritizing the labeling of most/all rare category samples may not help a classifier to learn how to accurately discriminate common category samples that are near these rare category samples. Many of these rare category samples may be largely redundant. Second, some of the proposed methods in [26] assume the class priors are known. While this may be realistic for some domains (e.g., rare disease categories with known prevalence), such information will not be available in general. Third, in detecting rare category examples, [26] uses a nearest neighbor framework, with distances in general measured using all the features. By contrast, our

approach seeks to be robust to the presence of many noisy features that are irrelevant to (and which may thus confound) discrimination of the rare and common categories. Accordingly, it accommodates implicit (soft) feature selection.

1.3 Anomaly Detection

Anomaly detection refers to the process of detecting (interesting) samples that deviate from the normal behavior. A detailed survey can be found in [6]. For example, in network intrusion detection, the administrator is interested in detecting various kinds of abnormal traffic (DDoS attack pattern, botnet C&C traffic, etc.) in real time and to act upon them; in astronomy, scientists want to find interesting objects in a large batch of captured astronomical images, while many of them may contain anomalies that are uninteresting (noise, sensor mis-calibration, etc.) [46].

In both of these situations, the experts are only interested in a (very small) subset of anomalies and this is usually subjectively defined according to the particular situation. In this work, applying active learning, we allow the learning machine to actively exploit expert's subjective knowledge by judiciously choosing what samples to be forwarded for labeling, based on the amount of "information" they contain. We also show how this improves anomaly detection in the domain of vehicle tracking [51]. To characterize normal behavior, we hypothesize a null model exclusively based on normal behaviors, which will be described in Section 2.2.

1.4 Other Related Work

Training set class imbalance for supervised learning of classifiers has been addressed in a number of prior works. Widely used approaches include class rebalancing (by discarding samples from the majority class) [29] and class reweighting [8][22], where greater weights in the sample-wise learning objective function are given to samples from minority classes. Other approaches such as boosted ensembles [16] implicitly address this problem, by focusing the training of successive classifier stages on the most difficult examples. That is, the minority class examples and the examples from common classes nearest to them in feature space are likely the most difficult examples to correctly classify.

We also emphasize that our definition of an "unknown" class is very different from [40] and the "unsupervised" case considered in [26]. In these works, beyond wholly lacking initial labeled examples, "unknown categories" are not even in the predefined set,

i.e., they are completely *unanticipated* classes(clusters). Such classes can be identified by applying some type of clustering to a given (semi-supervised) data batch, with the unknown classes identified as detected clusters that do not contain any labeled examples [40]. These unanticipated classes might be latently present within a batch due to an incomplete knowledge of the classification domain (e.g., presence of an as yet undiscovered type of galactic object in an astronomical database ostensibly capturing predefined categories; presence of an unknown disease type "contaminating" a patient database for a known disease domain).

1.5 Our Contributions

There are a total of five novel contributions and applications in this thesis related to semi-supervised active learning and anomaly detection:

1. Our proposal of a novel (p-value based) feature representation and a companion class posterior model that together encode our inductive bias for semi-supervised learning in the presence of unknown (label-deficient) categories. When there are rare categories, this model will be shown to outperform conventional classifiers that have similar representation power and which are trained in the same way as our model, but which work on raw features rather than p-values. This will be highlighted in Chapter 2.

2. A novel semi-supervised learning framework with built-in capability for avoiding overtraining, as especially needed for SL and AL with unknown categories. This approach *preserves*, rather than minimizes, unlabeled decision uncertainty, and will be shown to give superior results for SL and AL under the unknown and LS class scenarios. This will be highlighted in Chapter 2.

3. A semi-supervised AL framework that exploits the first two contributions to achieve good classification accuracy with sparing use of oracle labeling. This approach will be shown to outperform several conventional benchmark methods, including support vector machine based AL. These observations and results will be highlighted in Chapter 2.

4. The application of the proposed maxEnt model and its variations in vehicle tracking and network intrusion detection systems will be discussed and their results are presented in Chapter 3.

5. A group anomaly detection (GAD) technique is proposed to jointly identify the sample and feature subset that together exhibit similar anomalous behaviors. This will be the primary focus in Chapter 4.

In Chapter 5, we will summarize our contributions and provide directions for future research.

Chapter 2

Proposed MaxEnt Semi-supervised Active Learning Model

2.1 The Assumed Inductive Bias: the Unknown Class as a Subset of What is Anomalous

The framework that we propose for SL and AL with unknown or label-scarce categories is inspired by statistical AD, and can be thought of as a generalization of (unsupervised) statistical AD. Specifically, considering a two-class problem, we will treat the common category (ω_1) as the null hypothesis (whose model can be learned based on the initial labeled training set of common category examples, $\mathcal{X}_l$), with the unknown (or LS) class denoted ω_2. Invoking a standard AD approach, given the learned null model, one would interrogate the unlabeled data batch $\mathcal{X}_u$, seeking to identify the set of *anomalous* samples – those which deviate most from the null, *i.e.*, those with lowest *p-values* [1]. The inductive bias [42] that we impose on the common versus unknown two-class problem is that the samples that come from the unknown (or label-scarce) class are a (special) subset of the *anomalous* samples in the data batch – those whose atypicalities (low p-values) are with respect to certain key (albeit *a priori* unknown) features or feature combinations. While other inductive bias should be more suitable for learning a classifier when there are plentiful labeled examples of both classes, for our label-scarce setting, we focus on *anomalies* with respect to the common class, in learning to discriminate between the common class and an unknown (or LS) class. The effectiveness of this choice will be

[1] A p-value is the probability of seeing a datum more extreme than the given observation, under the null hypothesis.

borne out by our experiments.

To exposit further, our inductive bias has three fundamental tenets:

1. A sample originates from the unknown class $\sim$ only if it is anomalous with respect to the common class – *i.e.*, if it is "typical" of the common class with respect to all features, it should not belong to the unknown class.

2. The samples in the unlabeled batch that come from the unknown class all exhibit anomalies with respect to a special (*a priori* unknown) common subset of the measured features, which we dub the *informative* feature subset.

3. The more atypical a sample is with respect to these (unknown) informative features, the more likely it is to originate from the unknown class.

The first tenet simply assumes the two classes are statistically distinct and discriminable to some extent – if so, the unknown class samples should be atypical *in some way*, referenced to the common class distribution. The second tenet suggests the anomalies of the unknown class samples are not expected to be random - since they come from the same class, these samples' anomalies are expected to exhibit a *pattern*. Finally, the third tenet links the strength of the anomalous pattern to the discriminability of the two classes.

Example: given a classification domain with a 100-dimensional feature vector $\underline{X} \in \Re^{100}$, the unknown class could consist of the samples in $\mathcal{X}_u$ that exhibit anomalies (low p-values) with respect to some subset of the following: the common class's (marginal) distributions for individual features X_{17} and X_{32}; the common class's marginal distributions for the feature pairs (X_{79}, X_{93}), (X_{45}, X_{67}), and (X_{17}, X_{52}).

There are several important observations to make here.

First, it should be clear from the above example, with a high-dimensional feature space and a small unknown subset of informative features, that, *in general*, an unknown class may not be identifiable without some supervising examples. At the same time, we note that there *is* prior work on purely *unsupervised* detection of unknown classes in a data batch [11][28], e.g., the approach in [28] *solely* exploits the low likelihood (under the null) that a (sizeable) subset of samples would manifest anomalies with respect to the same subset of features. In any event, in an AL setting, the supervision needed to make the unknown class identifiable/learnable will be oracle-supplied.

Second, note our inductive bias instructs that a special (nonlinear) feature transformation should be applied to the original, measured features - *p-values* (on individual

features and low-order feature combinations), treated as derived features (classifier inputs), are hypothesized to be effective for discriminating between the unknown and common classes.

Third, as illustrated by the above example, it is possible that the unknown class may only *conspicuously* manifest anomalies (low p-values) with respect to a *small* subset of the measured features (the *informative* features)[2]. The presence of *many* "uninformative" or nuisance features is plausible because, without any initial labeled examples of the (unknown) class, one does not in general know *a priori* which features will be needed to discriminate this class from the common class, *i.e.* one may need to "overprovision" with features; consequently, many uninformative features may be measured. Likewise, it will also be *a priori* unknown which low-order feature *combinations* (and associated p-values) will be class-discriminating. Thus, in general, one must agnostically measure *all* of them – e.g., restricting just to feature pairs to limit complexity, all $\frac{N(N-1)}{2}$. Thus, for N large, there may be many marginal and, especially, many pairwise, feature atypicalities that are uninformative. It is thus incumbent upon AL to "figure out" which (potentially sparse subset of) individual and low-order feature combination p-values are strongly discriminating. This amounts to a (possibly high-dimensional) feature selection problem. Note also that "distance-based" methods, such as [26], which discriminate on the basis of all (or most) features, should be suboptimal when many of the features are uninformative.

Fourth, suppose that we have identified the informative features. The three tenets together imply a constraint should be imposed on the classifier's parameter space. Specifically, as seen in the sequel, negative log p-values should be *non-negatively weighted*, with *zero* weights for irrelevant features (tenet 2), and with larger positive weights for more informative features (tenet 3). Finally, note that our above assumptions do *not* necessitate that an unknown class consist of a compact *cluster*, in either the full feature space (as assumed, e.g., in [26], [47]), or even in a subspace of the full feature space. That is, two (unknown class) samples that are atypical with respect to the same pair of features need *not* be close together in the corresponding 2-D feature space. As an example, consider a bivariate Gaussian null, whose equidensity contour is an ellipse. A pair of samples lying on the same ellipse will have the same p-value[3]. However, the two samples could be at

[2]The common and unknown class may have very similar distributions for all other features. Alternatively, common and unknown class samples in the data batch may *both* manifest anomalies with respect to these other features. Either way, these other features will not have much power to discriminate the two classes and are thus "uninformative". Moreover, considering finite sample sizes available for classifier training, it is well-known that use of features with weak discrimination power can *compromise* classification accuracy [61], *i.e.*, these "uninformative" features may effectively be nuisance features.

[3]Assuming that the p-value is obtained by integrating the bivariate null density over the region defined by the exterior of the ellipse.

opposite ends of the ellipse along the principal axis, and thus be quite distant from each other. What this suggests is that "clustering-based" approaches that work in the original feature space, e.g., [26],[47], may not be effective at identifying unknown classes (at least those consistent with our inductive bias). Our use of p-values as derived features can accordingly be considered an (intuitively motivated) alternative to agnostic use of a (e.g., Gaussian) kernel for achieving a nonlinear mapping of the original features. For some domains, use of such kernel-based mappings may give poor class discrimination, e.g., [25][4].

As aforementioned, we assume this inductive bias in SL and AL-based learning of classifiers for common versus unknown (or LS) class discrimination *in general* and will experimentally validate that the resulting classifiers achieve good performance on a variety of real-world (UC Irvine ML) data domains. However, to further motivate our approach, we next consider in more detail a domain where it is *natural* both i) to have unknown classes and ii) to expect that they are best discriminated using *p-values*.

2.1.1 Discriminating Interesting from Uninteresting Anomalies

In real AD deployment settings, one often finds that *too many* samples are detected as anomalies, flooding human operators with alerts [46],[58]. This may be due to limited prior knowledge/data for accurately estimating a model of what is "normal". It may also be due, e.g., to sensor mis-calibration, sensor failure, or human error in system configuration. What is desired is not to detect all anomalous samples, but rather to detect and prioritize what, for some domains, amounts to the most suspicious anomalies (for behavioral tracking, intrusion detection, email spam) and for others (scientific applications) amounts to the most interesting ones, *i.e.* new/emergent physical phenomena. There are several challenges associated with detecting the most suspicious (interesting) samples. First, "suspicious" may be a very small subset of what is anomalous, consisting of samples whose atypicalities are with respect to certain key (albeit *a priori* unknown) features (and feature interactions). Indiscriminately forwarding all detected anomalies for operator scrutiny will not scale; it will swamp the operator, making it very unlikely to identify truly suspicious anomalies[5]. Thus, in practice, it is necessary to automatically *discriminate* suspicious and innocuous anomalies that are output by an AD, and thus

[4]The Gaussian kernel, for example, implicitly defines generalized coordinates that include product features (and, thus, will combine informative and irrelevant features through their products). In this way, the contribution to the classifier from informative features may be distorted or masked by use of the kernel.

[5]If the detection rate is higher than the inspection rate, the operator will only be able to inspect and label a fraction of forwarded anomalies.

treat this as a supervised learning problem, as we have done recently [51]. However, because "suspicious" is rare, it may be an unknown (or LS) class. This by itself implies the need for AL (to identify some suspicious instances). Moreover, "suspicious" may be subjective, and thus may need to be informed by the current human operator. Second, we can expect feature selection will be especially important here – beyond "provisioning" for the unknown (suspicious) class, suspicious (e.g., malicious) entities may actively *obfuscate* their behavior, in which case the suspicious signature – its anomalousness – may only manifest with respect to a small subset of all measured features. If one restricts the features under consideration, one may wholly *miss* the suspicious signature. Feature selection can also make explicit and objective the basis upon which a (subjective) human operator distinguishes suspicious from innocuous anomalies.

2.2 Comprehensive Low-order Null Modeling

Suppose we are given a "training set" of known common (*i.e.*, null hypothesis) class (Ω_1) examples $\tilde{\mathcal{X}} = \{\underline{\tilde{x}}^{(i)}, i = 1, \ldots, T\}$, $\underline{\tilde{x}}^{(i)} \in \mathbb{R}^D$, from which to learn our probability model under the null[6]. One possibility is to learn the joint density function for $\underline{X} \in \mathbb{R}^D$ under Ω_1; then, for a given observed vector $\underline{x} \in \mathcal{X}_u$, one would measure a single p-value with respect to this joint density. However, there are several issues here. First, if D is large, the training database $\tilde{\mathcal{X}}$ may need to be huge in order to accurately estimate the joint density, *i.e.*, there is a curse of dimensionality [3][7]. This statement may hold even if one invokes strategies for limiting how the model size grows as a function of D, e.g., [15], [18] and [37], which efficiently model covariance matrices for Gaussian mixture model-based joint densities. Second, the anomalous signature may be too weak (the p-value too large) for large D. For example, suppose that the features are statistically independent under the null, and that a sample $\underline{x}$ exhibits an anomaly only for *one* (or very few) of the D features. In this case, the joint log-likelihood for $\underline{x}$ under the null is the sum of the marginal (single feature) log-likelihoods, with the effect of a single (anomalous) feature on the joint log-likelihood (and, thus, on the joint p-value) *averaged out* for increasing D.

To avoid both these problems, we instead propose to represent the null (and assess p-values) using *low-order* distributions, defined on sub-vectors of $\underline{X}$. Since it is *a priori*

[6]For the unknown class case, $\tilde{\mathcal{X}} = \mathcal{X}_l$. For the label-scarce case, $\tilde{\mathcal{X}}$ is the subset of $\mathcal{X}_l$ that originates from the common classes.

[7]This problem is exacerbated by the potential of finding poor local optima of the (generally non-convex) likelihood objective function that is maximized in learning the joint density function.

unknown which individual features and feature combinations may be informative about the unknown class, we must be unbiasedly comprehensive, considering *all* of them. Noting that there are $\binom{D}{k} = \frac{D!}{k!(D-k)!}$ such distributions at order k, even assuming that estimation of these distributions is performed "off-line", the highest order models that may be computationally practicable to learn (while at least capturing statistical dependencies between pairs of features) is $k = 2$. Accordingly, assuming large D, we propose to estimate all D marginal densities $\{f_{X_i}(\cdot), i = 1 \ldots, D\}$ and all $D(D-1)/2$ pairwise densities $\{f_{X_i,X_j}(\cdot), i, j = 1, \ldots, D, i < j\}$. Moreover, since distributions under the null will in general be complicated and multimodal, we will apply a widely invoked approach, modeling each marginal and pairwise feature density as a finite Gaussian mixture model (GMM). We learn each such model separately via the Expectation-Maximization (EM) algorithm [13], with the number of components chosen to minimize the Bayesian Information Criterion (BIC) [54] model order selection objective[8]. While this learning is computationally heavy, in practice it can often be performed *off-line*, prior to active learning. Moreover, this GMM learning is also highly parallelizable. In the next section, we describe our GMM modeling approach.

2.2.1 Gaussian Mixture Model

A widely used approach for learning a complex, multi-modal probability distribution for continuous data is a GMM. A GMM with M components is defined as follows:

$$P(\underline{x}) = \sum_{m=1}^{M} \alpha_m \mathcal{N}(\underline{x}|\underline{\mu}_m, \Sigma_m),$$

[8]Separately learning each marginal and pairwise feature GMM using the common training set $\tilde{\mathcal{X}}$ will not ensure *consistency* with respect to feature marginalizations. Specifically, a *marginal-consistent* collection of univariate and bivariate density functions should satisfy the following: if we consider any feature pairs (i, j) and (j, k), marginalizing out feature i from the (i, j) bivariate density and marginalizing out feature k from the (j, k) bivariate density should lead to the same marginal density for feature j. However, when the univariate and bivariate distributions are Gaussian mixtures, with a *non-convex* log-likelihood function (and with BIC-based model order selection separately applied to choose the number of components for each GMM), separate application of EM-plus-BIC to learn each GMM density function does not ensure a set of marginal-consistent distributions. This property is not centrally important here, however, since our main concern is only to learn marginal and pairwise density functions that allow accurate assessment of p-values. Accordingly, in this work we will apply EM-plus-BIC separately, to learn each low-order GMM.

One approach to obtain marginal-consistent low-order distributions is to simply learn the single GMM for the joint distribution on the *full* feature vector, $\underline{X}$. This *determines* (via marginalization) all lower-order distributions (which are also GMMs, and which are guaranteed to be marginal-consistent). However, if available data is limited, this strategy suffers from the curse of dimensionality [3]. Alternatively, it is possible to directly, jointly learn a marginal-consistent set of low-order GMMs.

where $\mathcal{N}(\underline{x}|\underline{\mu}_m, \Sigma_m)$ represents the m^{th} component of the mixture model, with its own unique mean and covariance matrix. α_m represents the m^{th} mixture coefficient (or component mass) and $\sum_m \alpha_m = 1$. Given a null training set $\tilde{\mathcal{X}} = \{\underline{\tilde{x}}^{(i)}, i = 1, \ldots, T\}$ with T observations, a standard approach to learn the GMM parameters Θ, with:

$$\Theta = \{(\alpha_1, \underline{\mu}_1, \Sigma_1), \ldots, (\alpha_M, \underline{\mu}_M, \Sigma_M)\},$$

is through maximum likelihood estimation (MLE):

$$argmax_\Theta \; P[\tilde{\mathcal{X}}|\Theta].$$

Assuming data are statistically independent and identically distributed, due to monotonicity of the log function, we can simply turn this into a maximum log likelihood estimate:

$$argmax_\Theta \; \sum_{i=1}^{T} log \sum_{m=1}^{M} \alpha_m \mathcal{N}(\underline{x}_i|\underline{\mu}_m, \Sigma_m). \tag{2.1}$$

Because of the presence of the mixture coefficients and the summation over M inside the logarithm, the log likelihood function is non-convex and the MLE for Θ does not have a closed-form analytical solution. We instead use the framework of expectation-maximization (EM) to find a local maximum of the above.

Specifically, applying the EM framework in learning the GMM (2.1), we have the following iterative process:

1. Initialize $\Theta^{(t)}$ through k-mean clustering.

2. Expectation (E) Step: calculate the posteriors of each sample for each component.

$$P[M = m|\underline{x}_i] = \frac{\alpha_m f_{\underline{x}|m}(\underline{x}_i|\theta_m)}{\sum_{k=1}^{M} \alpha_k f_{\underline{x}|k}(\underline{x}_i|\theta_k)}, \forall i \in \{1, \ldots, T\}, \forall m \in \{1, \ldots, M\}. \tag{2.2}$$

3. Maximization (M) Step, $\forall m \in \{1, \ldots, M\}$:

$$\begin{aligned} \underline{\mu}_m &= \frac{1}{T} \sum_{i=1}^{T} P[M = m|\underline{x}_i]\underline{x}_i \\ \Sigma_m &= \frac{1}{T} \sum_{i=1}^{T} P[M = m|\underline{x}_i](\underline{x}_i - \underline{\mu}_m)(\underline{x}_i - \underline{\mu}_m)^T \end{aligned}$$

$$\alpha_m \quad = \quad \frac{1}{T}\sum_{i=1}^{T} P[M=m|\underline{x}_i] \tag{2.3}$$

4. When convergence in (2.1) happens (e.g., when the relative difference from the previous update falls below a threshold), return. Otherwise, assign $\Theta^{(t)}$ to $\Theta^{(t+1)}$ and return to 2).

To select the number of component (M), we can apply the Bayesian Information Criterion (BIC) [3]:

$$BIC \quad = \quad (M(D+\frac{D(D+1)}{2})+M-1)\frac{log(T)}{2} - \sum_{i=1}^{T} log \sum_{m=1}^{M} \alpha_m f_{\underline{x}|m}(\underline{x_i}|\theta_m), \tag{2.4}$$

where the last summation is the incomplete data log likelihood in (2.1).

In summary, each single and pairwise GMM is learned by EM-plus-BIC, selecting the (M, Θ) pair that minimizes the BIC cost.

2.2.2 Mixture-based P-Values as Derived Features

Consider any pair of features $\underline{Y} = (X_i, X_j)$, modeled by a bivariate GMM under the null. Let $\{\alpha_m, m = 1, \ldots, M_{ij}\}$, $0 \leq \alpha_m \leq 1$, $\sum_{m=1}^{M_{ij}} \alpha_m = 1$, be the prior probabilities for the (M_{ij}) mixture components, with associated component densities $f_{\underline{Y}|m}(\underline{y}|\theta_m)$, $m = 1, \ldots, M_{ij}$, $\theta_m = (\underline{\mu}_m, \Sigma_m)$ the (mean vector, covariance matrix) parameter set for the m^{th} density. The mixture density is thus $f_{\underline{Y}}(\underline{y}) = \sum_{m=1}^{M_{ij}} \alpha_m f_{\underline{Y}|m}(\underline{y}|\theta_m)$. Given such a mixture null, we would like to calculate the p-value – the probability that a two-dimensional feature vector will be more extreme than the given observed vector $\underline{y} = (x_i, x_j)$. This has been previously considered in [48], with the result dubbed *mixture-based p-values*. First consider a single bivariate Gaussian density $\mathcal{N}(\underline{\mu}, \Sigma)$. We call a vector $\underline{y}'$ "more extreme" than $\underline{y}$ if it lies on a lower equidensity (elliptical) contour than $\underline{y}$. Accordingly, the p-value for $\underline{y}$, denoted $p_{ij}(\underline{y})$ is the integral of the bivariate density over the exterior of the ellipse defined by the squared Mahalanobis distance from $\underline{y}$ to $\underline{\mu}$:

$$r^2(\underline{y}) = (\underline{y}-\underline{\mu})'\Sigma^{-1}(\underline{y}-\underline{\mu}). \tag{2.5}$$

This integral can be calculated exactly by applying a whitening transformation [14], leading to the result that the p-value is: $p_{ij}(\underline{y}) = e^{-r^2(\underline{y})/2}$, *i.e.*, it is one minus the Rayleigh cdf evaluated at $r(\underline{y})$. Extending to a GMM by applying the law of total

probability and conditioning on the mixture component of origin, we find that the p-value for $\underline{y} = (x_i, x_j)$ is:

$$p_{ij}(\underline{y}) = \sum_{m=1}^{M_{ij}} P[M = m|\underline{y}]e^{-r_m^2(\underline{y})/2}. \tag{2.6}$$

Here, the mixture posterior is:

$$P[M = m|\underline{y}] = \frac{\alpha_m f_{\underline{Y}|m}(\underline{y}|\theta_m)}{\sum_{k=1}^{M_{ij}} \alpha_k f_{\underline{Y}|k}(\underline{y}|\theta_k)}, \tag{2.7}$$

and $r_m^2(\underline{y})$ is the squared Mahalanobis distance between $\underline{y}$ and $\underline{\mu}_m$ for the l^{th} GM component. Note that $p_{ij}(\underline{y})$ is the expected p-value, with the expectation taken with respect to the mixture posterior pmf. In a similar fashion, one can also calculate mixture-based p-values for single (univariate) features, denoted $p_i(x_i), i = 1, \ldots, D$. In this case, complementary error functions are used to measure the p-value conditioned on each mixture component, with the mixture-based p-value again the expected p-value.

2.3 Classifier Model

2.3.1 Two Class Case

For a given feature vector $\underline{x} \in \mathbb{R}^D$, define the set of low-order (one and two-dimensional) mixture p-values $\mathcal{P}(\underline{x}) = \{p_{ij}(x_i, x_j), i, j = 1, \ldots, D, i < j\} \cup \{p_i(x_i), i = 1, \ldots, D\}$[9], as defined in Section 2.2. Our active learning approach can be applied for various class posterior models. Here, it is developed assuming a "nearly standard" logistic regression class posterior form, albeit with negative logs of the p-values $\mathcal{P}(\underline{x})$ as derived feature inputs and with non-negative weight constraints. Specifically, for a two-class problem, with one common class (Ω_1) and one unknown or LS class (Ω_2), let

$$f(\underline{x}; \Lambda) = \exp(w_0 - \sum_{i=1}^{D} w_i \log p_i(x_i) - \sum_{j>i} \beta_{ij} \log p_{ij}(x_i, x_j)),$$

[9]We restrict to first and second order p-values in order to ensure scalability of the method with increasing D.

where the model parameters are $\Lambda = \{w_0, \{w_i\}, \{\beta_{ij}\}\}$. We then have:

$$\begin{aligned} P(\Omega_2|\mathcal{P}(\underline{x});\Lambda) &= \frac{f(\underline{x};\Lambda)}{1+f(\underline{x};\Lambda)}, w_i \geq 0, \beta_{ij} \geq 0, \forall i,j \\ P(\Omega_1|\mathcal{P}(\underline{x});\Lambda) &= 1 - P(\Omega_2|\mathcal{P}(\underline{x});\Lambda). \end{aligned} \tag{2.8}$$

We can make the following observations about this model:

- If we initialize at $w_1 = w_2 = ... = w_D = w > 0$ and set $\beta_{ij} = \beta > 0, \forall i,j$, then ordering samples by their *a posteriori* probability of belonging to the unknown class is equivalent to ordering them by their aggregate atypicality (the sum of the arithmetic averages of their first and second order log p-values), *i.e.*, *initially*, before there are any supervising examples, the model is unbiased about which features/combinations are informative, treating the "most anomalous" sample as the one most likely to come from the unknown class. Thus, before there are supervising examples, our model is akin to a conventional anomaly detector.

- We enforce non-negative weights on negative log p-values (which are positive), consistent with our inductive bias that "unknown" is a subset of what is anomalous with respect to Ω_1 – no increase in a feature's atypicality should *reduce* the probability of a sample belonging to the unknown class. However, if a single feature's (i) or a feature pair's $((i,j))$ atypicality is irrelevant to common-unknown discrimination, this can be properly reflected by setting $w_i = 0$ or $\beta_{ij} = 0$, respectively. In fact, as will be seen in the sequel, our approach learns *highly sparse* models, with very few non-zero weights. Thus, once learned, our model is very different from a conventional anomaly detector.

- *Monotonic Sample Ordering (Generalization) Property:* Suppose that the classifier parameters have been chosen so that $P(\Omega_2|\mathcal{P}(\underline{x}^{(n)})) < 0.5$, *i.e.* such that sample n is MAP-classified to Ω_1. Now consider any other sample $\underline{x}^{(m)}$ such that $p_i(x_i^{(m)}) \geq p_i(x_i^{(n)}), \forall i = 1, ..., D$ and $p_{ij}(x_i^{(m)}, x_j^{(m)}) \geq p_{ij}(x_i^{(n)}, x_j^{(n)}), \forall i,j$. Clearly, $P(\Omega_2|\mathcal{P}(\underline{x}^{(m)})) \leq P(\Omega_2|\mathcal{P}(\underline{x}^{(n)})) < 0.5$, *i.e.* if we have learned to maximum *a posteriori* (MAP)-assign sample n to Ω_1, we have in fact also learned to MAP-assign all samples more typical than n under the null to Ω_1. The same also holds in the alternative case, supposing that $\underline{x}^{(n)}$ is MAP-assigned to Ω_2 and considering the set of samples more *atypical* than $\underline{x}^{(n)}$. This is a type of *generalization* that comes from constraining the weights (on derived features that are positive) to be non-negative.

2.3.2 Extension for Multiple Common and Label-scarce Classes

Suppose now that there are *multiple* common classes $\Omega_{m_i} \in \mathcal{C}_m, i \in \{1, ..., M\}$ and multiple LS classes $\Omega_{r_j} \in \mathcal{C}_r, j \in \{1, ..., R\}$. In order to generalize our two-class approach to handle multiple common and LS classes, we propose a *hierarchical* class posterior, defined on all common and LS classes, to allow inference on common and rare/unknown sub-classes, *i.e.*,

$$P[C = \Omega_{r_j}|\underline{x}] = P[C \in \mathcal{C}_r|\underline{x}]P[C = \Omega_{r_j}|\mathcal{C}_r, \underline{x}], \tag{2.9}$$

and

$$P[C = \Omega_{m_i}|\underline{x}] = P[C \in \mathcal{C}_m|\underline{x}]P[C = \Omega_{m_i}|\mathcal{C}_m, \underline{x}]. \tag{2.10}$$

Here, the top-level posterior is the *two class subset* ($\{\mathcal{C}_r, \mathcal{C}_m\}$) posterior, *i.e.*,

$$P[C \in \mathcal{C}_r|\underline{x}; \Lambda] = \frac{f(\underline{x}; \Lambda)}{1 + f(\underline{x}; \Lambda)}, w_i \geq 0, \beta_{ij} \geq 0, \forall i, j$$

and

$$P[C \in \mathcal{C}_m|\underline{x}; \Lambda] = 1 - P[C \in \mathcal{C}_r|\underline{x}; \Lambda]. \tag{2.11}$$

The second-level posteriors $P[C = \Omega_{r_j}|\mathcal{C}_r, \underline{x}]$ and $P[C = \Omega_{m_i}|\mathcal{C}_m, \underline{x}]$ "divide up" the top-level probabilities $P[C \in \mathcal{C}_r|\underline{x}]$ and $P[C \in \mathcal{C}_m|\underline{x}]$ amongst $\Omega_{r_j} \in \mathcal{C}_r$ and $\Omega_{m_i} \in \mathcal{C}_m$, respectively.

The second-level posterior for the LS classes, $P[C = \Omega_{r_j}|\mathcal{C}_r, \underline{x}]$, is of the multi-logit form and can be defined either in the raw feature space or in p-value space, e.g., for raw feature space, let:

$$f(\underline{x}; A^j) = \exp(a_0^{(j)} + \sum_{k=1}^{D} a_k^{(j)} x_k + \sum_{l>k} a_{kl}^{(j)} x_k x_l),$$

where the model parameters are $A^j = \{a_0^{(j)}, \{a_k^{(j)}\}, \{a_{kl}^{(j)}\}\}$.

We then have, $\forall j \in \{1, ..., R\}$:

$$P[C = \Omega_{r_j}|\mathcal{C}_r, \underline{x}; A^j] = \frac{f(\underline{x}; A^j)}{\sum\limits_{\Omega_{r_{j'}} \in \mathcal{C}_r} f(\underline{x}; A^{j'})}. \tag{2.12}$$

Experimentally, we have found that while using p-values in the top level (to discriminate

between the common and LS class subsets) significantly outperforms use of raw features, there are only small differences in performance resulting from using p-values versus raw features for the second-level posterior to discriminate between the LS classes. This is not so surprising, since our inductive bias is really associated with the top layer posterior (discriminating the common class subset from the rare class subset). Note that one can also change the feature representation on the second-level posterior instead of using raw features.

The second-level posterior for the known classes $P[C = \Omega_{m_i}|\mathcal{C}_m, \underline{x}]$ is also of the multi-logit form and works in the raw feature space, *i.e.*, for the i^{th} common class, let:

$$f(\underline{x}; B^i) = \exp(b_0^{(i)} + \sum_{k=1}^{D} b_k^{(i)} x_k + \sum_{l>k} b_{kl}^{(i)} x_k x_l),$$

where the model parameters are $B^i = \{b_0^{(i)}, \{b_k^{(i)}\}, \{b_{kl}^{(i)}\}\}$. We then have, $\forall i \in \{1, ..., M\}$:

$$P[C = \Omega_{m_i}|\mathcal{C}_m, \underline{x}; B^i] = \frac{f(\underline{x}; B^i)}{\sum_{\Omega_{i'} \in \mathcal{C}_m} f(\underline{x}; B^{i'})}. \tag{2.13}$$

Note that both second-level posteriors are valid pmfs, summing to one over their respective class subsets, with the hierarchical posterior summing to one over all classes. Note also that, unlike the top-level posterior, there are no constraints on values for parameters of the bottom-level posterior pmfs.

2.3.3 Extension to Include Unknown Classes

In the AL setting, in addition to $\mathcal{C}_m$ and $\mathcal{C}_r$, we have the *unknown class subset*, $\mathcal{C}_u$, consisting of an (in general) unknown number of unknown classes, which need to be discovered through active learning. Once a new class is discovered via AL sample selection and oracle labeling, this category (now with one labeled instance) is moved from $\mathcal{C}_u$ to the LS class subset, $\mathcal{C}_r$. Thus, the subsets $\mathcal{C}_u$ and $\mathcal{C}_r$ change as a function of oracle labelings. To reflect this, we use the superscript (t), denoting $\mathcal{C}_u^{(t)}$ and $\mathcal{C}_r^{(t)}$ after the t-th oracle labeling. For concise notation in representing the event that a sample belongs to the unknown class subset, we assign (the unused) class $C = 0$, *i.e.* we let $C \in \mathcal{C}_u \Leftrightarrow C = 0$. Letting $C_{ru}^{(t)}$ denote $\mathcal{C}_r^{(t)} \cup \mathcal{C}_u^{(t)}$, the top level posterior for $\mathcal{C}_{ru}^{(t)}$ and $\mathcal{C}_m$ is of the same form as (2.11), with $P[C \in \mathcal{C}_{ru}^{(t)}|\underline{x}]$ and $P[C \in \mathcal{C}_m|\underline{x}]$, respectively. However, we modify the bottom level posterior for $\mathcal{C}_r$ as follows, to allow inference on the unknown class subset.

We have:

$$P[C = \Omega_{r_j} | C_{ru}^{(t)}, \underline{x}; A^j] = \frac{f(\underline{x}; A^j)}{1 + \sum\limits_{\Omega_{r_{j'}} \in \mathcal{C}_r^{(t)}} f(\underline{x}; A^{j'})} \quad \forall j \in \{1, ..., R\}, \tag{2.14}$$

and for $C \in \mathcal{C}_u^{(t)}$:

$$P[C = 0 | C_{ru}^{(t)}, \underline{x}; A^j] = \frac{1}{1 + \sum\limits_{\Omega_{r_{j'}} \in \mathcal{C}_r^{(t)}} f(\underline{x}; A^{j'})}. \tag{2.15}$$

Note that (2.14) and (2.15) define a valid pmf, *i.e.* $P[C = 0 | C_{ru}^{(t)}, \underline{x}; A^j] + \sum\limits_{j=1}^{R} P[C = \Omega_{r_j} | C_{ru}^{(t)}, \underline{x}; A^j] = 1$. Note also that, at the outset of AL, if there are no LS classes, *all* probability is assigned to the unknown class ($C = 0$). As the cardinality of $\mathcal{C}_r^{(t)}$ grows during AL, the probability of the unknown category ($C = 0$) tends to diminish. To control the amount of contribution from the unknown class, one can further replace the value "1" in both (7) and (8) with a hyper-parameter, and tune its value based on cross validation or a related measure. We leave consideration of this for future work.

2.4 Learning Objective Function for Semi-supervised and Active Learning

In this section, we focus on classifier learning, developing a novel semi-supervised learning objective well-suited both for SL and AL with unknown (and LS) classes. Our principal focus, in devising a learning framework for these scenarios, is to make appropriate, effective use of the (assumed to be many) unlabeled samples in the data batch ($\mathcal{X}_u$), in addition to the current cadre of labeled samples ($\mathcal{X}_l$). In this section, let $\mathcal{I}_l$ denote the sample index set for the labeled samples, $\mathcal{X}_l$, with $\mathcal{I}_u$ the index set for the unlabeled samples.

If we were to only make use of the labeled samples, a standard (supervised) approach for learning a class posterior model is maximization of the posterior log-likelihood over the labeled training samples [3]. For our purposes, it is convenient to exploit the equivalence between posterior log-likelihood maximization and minimization of a sum of cross entropies objective function, specified as follows. For a given labeled feature vector

instance $\underline{x}_n$ from class c_n, we define the "target" distribution vector:

$$\begin{aligned} Q[C|n] &= (Q[\Omega_1|n], \ldots, Q[\Omega_{|\mathcal{C}|}|n]) \\ &= (0, 0, \ldots, 1, 0, \ldots, 0), \end{aligned}$$

with the '1' in the position of $c_n \in \{\Omega_1, \ldots, \Omega_K\}$. Then, after the t-th oracle labeling of active learning, maximizing the log-likelihood of the posterior model $P[C|\underline{x}]$ given in (2.9),(2.10) is equivalent to minimizing

$$\begin{aligned} D^{(t)} &= \sum_{n \in \mathcal{I}_l^{(t)}} d(Q[C|n]||P[C|\underline{x}_n]) \\ &= -\sum_{\Omega \in \mathcal{C}_m \cup \mathcal{C}_r^{(t)}} \sum_{n \in \mathcal{I}_l^{(t)}: c(n) = \Omega} \log P[\Omega|\underline{x}_n] \end{aligned} \tag{2.16}$$

Here, $d(Q||P) = \sum_k q_k \log(q_k/p_k)$ is the Kullback-Leibler distance [30] (cross entropy) between probability mass functions $Q = \{q_k\}$ and $P = \{p_k\}$ defined on the same support, with Q the "target" distribution and P its (model-constrained) approximation. Note that in the AL case the labeled index set $\mathcal{I}_l$, the LS class set $\mathcal{C}_r$, and the unknown class set $\mathcal{C}_u$ are all functions of time. $\mathcal{I}_l$ grows with each oracle labeling, while $\mathcal{C}_r$ grows when a new unknown class is discovered through active labeling[10]. Note further that we do not consider here the case where a rare class Ω_{r_j} may transition (after AL has produced a sufficient number of labeled instances from this class) from $\mathcal{C}_r$ to $\mathcal{C}_m$[11].

We seek to modify (2.16) in two respects: i) primarily, to sensibly exploit for learning the (typically large) current subset of *unlabeled* samples, $\mathcal{X}_u^{(t)}$; ii) to also account for large imbalance between the number of labeled samples from common classes versus label-scarce classes. Let us consider the first goal. In [20], [17], and [19], semi-supervised learning objective functions were proposed by adding, to the purely supervised training objective function, a cost term which penalizes posterior solutions with high class uncertainty (*i.e.*, high class entropy) on the unlabeled sample subset. Specifically, following the approach taken in [20], one would modify the above training objective function to

[10] In the semi-supervised learning case, the superscript (t) should be omitted from these sets because, in this case, $\mathcal{X}_l$ and $\mathcal{X}_u$ remain fixed.

[11] For our experiments in the sequel, with the maximum number of oracle labelings set to 100, rare categories in general *remain* rare during our active learning.

add an entropy regularization term:

$$D^{(t)}_{\min-\mathrm{H}} = D^{(t)} + \mu \sum_{n \in \mathcal{I}_u^{(t)}} H(C|n), \tag{2.17}$$

where

$$H(C|n) = - \sum_{\Omega \in \mathcal{C}_m \cup \mathcal{C}_{ru}^{(t)}} P[C = \Omega | \underline{x}_n] \log P[C = \Omega | \underline{x}_n],$$

and with μ a non-negative hyperparameter, giving the weight on the entropy term[12]. This general approach, together with a weight decay term to avoid overfitting, has been dubbed "minimum entropy regularization" (minEnt), and has been motivated from the principle of making decisions with "confidence" [20]. A related approach given in [10] seeks to locally minimize class overlap. For the scenario considered here, however, with a deficient amount of labeled training data for some of the classes, it appears that the pivotal learning issue is how much training of the classifier's model parameters one should do, commensurate with the available labeled data, to glean as much information as possible from the labeled examples but while avoiding overtraining. The entropy penalty term in (3.3) will in general give a tendency for *more* learning, rather than less. This can be understood as follows. Consider the solution to the purely supervised posterior log-likelihood maximization problem, *i.e.* minimizing (2.16). Suppose that this solution achieves a level of unlabeled subset class entropy $H_0 = \sum_{n \in \mathcal{I}_u^{(t)}} H(C|n)$. Clearly, since the semi-supervised modification of (2.16) positively penalizes this class entropy, minimizing (3.3) should yield solutions with *lower* class entropy than those which minimize (2.16). However, such reduction in class entropy is generally achieved by increasing the magnitudes of the classifier's weights on the features as given in the posterior, *i.e.* by performing more learning. But this gives propensity for overtraining. In fact, in [20], [17] and [19], there is some "hedging" on the minimum entropy solution – in [20], the authors impose a smoothness constraint on weight vector, while in [17] an explicit, additional weight vector norm (L2) regularization term is introduced. Regardless, in both of these approaches, the impetus in the cost function coming from the unlabeled samples is to seek solutions with minimum class entropy.

By contrast, rather than penalizing solutions with high class decision uncertainty on the unlabeled samples, we penalize solutions with *low* decision uncertainty, as they may

[12]The value of the hyperparameter can be chosen, e.g., so as to minimize a cross-validated classification error rate measure.

be consistent with possible overtraining, especially during the early stages of AL and/or in the SL case, given label-scarce classes – *i.e., we propose maximum entropy (maxEnt), rather than minimum entropy regularization*[13].

Tikhonov-style regularization such as L2 or *Lasso*, by imposing a prior belief on the parameter space, also limits overtraining. However, such a universal prior has *no* dependence on (is unaffected by) unlabeled samples. By contrast, our maximum entropy regularizer is unlabeled-sample-dependent, seeking to keep the classifier as "noncommittal" as possible in regions of the feature space solely occupied by unlabeled samples. An example highlighting the difference between maxEnt and L2 regularization will be given in the sequel.

There are different ways maxEnt regularization can be mathematically formulated in practice. We effect such regularization by choosing the same (cross entropy) cost function for the unlabeled samples as for the labeled samples, albeit with a different choice for the target distribution. For every unlabeled sample, $n \in \mathcal{I}_u^{(t)}$, we introduce maximum entropy target distributions for both the top level and bottom level posteriors.

Denote $S(\Omega)$ as the class subset ($\mathcal{C}_{ru}$ or $\mathcal{C}_m$) to which Ω belongs. For the top level, we choose the uniform target distribution:

$$Q[S^{(t)}(C)|n] = (\frac{1}{2}, \frac{1}{2}).$$

Likewise, for the bottom level, we choose uniform target distributions. In the AL case, where we may have both $\mathcal{C}_r^{(t)}$ and $\mathcal{C}_u^{(t)}$, we choose:

$$\begin{aligned} Q[C|C \in C_{ru}^{(t)}, n] &= (\frac{1}{|\mathcal{C}_r^{(t)}| + 1}, \Omega \in \{0, \mathcal{C}_r^{(t)}\}), \\ Q[C|C \in \mathcal{C}_m, n] &= (\frac{1}{|\mathcal{C}_m|}, \Omega \in \mathcal{C}_m) \end{aligned}$$

In the SL case, if we only have $\mathcal{C}_r$, we choose:

$$\begin{aligned} Q[C|C \in \mathcal{C}_r, n] &= (\frac{1}{|\mathcal{C}_r|}, \Omega \in \mathcal{C}_r), \\ Q[C|C \in \mathcal{C}_m, n] &= (\frac{1}{|\mathcal{C}_m|}, \Omega \in \mathcal{C}_m) \end{aligned}$$

That is, at the top level, target probability is equally apportioned between $C_{ru}^{(t)}$ and

[13] In the latter stages of AL, when there are "sufficient" labels from all classes, minimizing class uncertainty on the unlabeled samples may be a good choice. However, in the early AL stages, a main concern should be avoiding over-training.

$\mathcal{C}_m$ and, at the bottom level, given a class subset, equally apportioned amongst the classes belonging to that subset[14]. Consistent with these target distributions, we have the following maximum entropy regularization costs (exposited for the AL case, where there are unknown classes). At the top level:

$$R_{\text{top}}^{(t)} = \sum_{n \in \mathcal{I}_u^{(t)}} d(Q[S^{(t)}(C)|n]||P[S^{(t)}(C)|\underline{x}_n]),$$

where $P[S^{(t)}(C)|\underline{x}] = (P[C_{ru}^{(t)}|\underline{x}], P[\mathcal{C}_m|\underline{x}])$, and at the bottom level:

$$R_{\text{bot}}^{(t)} = \sum_{n \in \mathcal{I}_u^{(t)}} (P[C_{ru}^{(t)}|\underline{x}_n] \cdot d_{QP}^r + P[\mathcal{C}_m|\underline{x}_n] \cdot d_{QP}^m),$$

where

$$\begin{aligned} d_{QP}^r &= d(Q[C|C \in C_{ru}^{(t)}, n]||P[C|C \in C_{ru}^{(t)}, \underline{x}_n])), \\ d_{QP}^m &= d(Q[C|C \in \mathcal{C}_m, n]||P[C|C \in \mathcal{C}_m, \underline{x}_n]). \end{aligned}$$

Note that the bottom-level regularizer weights each of the cross entropy terms by the probability that the sample originates from that class subset. Clearly, minimizing the costs $R_{\text{top}}^{(t)}$ and $R_{\text{bot}}^{(t)}$ will encourage entropy maximization on the unlabeled samples. Accordingly, our proposed semi-supervised objective function is a weighted sum of cross entropies over both the labeled and unlabeled data subsets, specified as follows[15]:

$$\begin{aligned} D_{\text{max-H}}^{(t)} = & - \sum_{\Omega \in \mathcal{C}_m \cup \mathcal{C}_r^{(t)}} \sum_{n \in \mathcal{I}_l^{(t)}: c(n) = \Omega} \alpha_\Omega^{(t)} \log P[\Omega|\underline{x}_n] \\ & + \alpha_{\text{top}}^{(t)} R_{\text{top}}^{(t)} + \alpha_{\text{bot}}^{(t)} R_{\text{bot}}^{(t)} \end{aligned} \tag{2.18}$$

Note that, in general, this cost function is nonconvex in its parameters – this is due to the regularization term $R_{\text{bot}}^{(t)}$. However, in the special case where there is a single known class and a single unknown (or LS) class (and, thus, no bottom layer regularization term $R_{\text{bot}}^{(t)}$), or for a standard multi-class, logistic regression based classification problem, the cost function is in fact convex. The objective (2.18) can be minimized by various optimization techniques, e.g., projected gradient or interior point methods, where the

[14]Note also that in the special case where there is a *single* common class and a *single* unknown class, there is only the top level, with target distribution $(\frac{1}{2}, \frac{1}{2})$.

[15]Again, in the SL case, the superscript (t) should be omitted since in this case the labeled and unlabeled data subsets are fixed.

descent direction can either be the gradient or Newton's direction. In our experiments, (2.18) is minimized, after the t-th oracle labeling, via a gradient descent procedure, which we have found to be effective. We applied the projected gradient descent method, *i.e.*, at each step, we update each parameter in its steepest descent direction. For parameters constrained to be non-negative, we project their values to 0 if the gradient-updated value goes negative. Inexact line search is used to select the learning rate (common to all parameters) at each iteration to ensure that each gradient update satisfies the Wolfe condition [45]. For each gradient update, with K known classes, N samples and D raw features, computational complexity of the gradient is $O(KND^2)$. To start the learning, the weights $\Lambda, \{A^j\}, \{B^i\}$ were all (unbiasedly) initialized to the same small value $\sim 10^{-6}$, except for $w_0, \{a_0^{(j)}\}$, and $\{b_0^{(i)}\}$, which were all initialized to zero.

A second important issue is how to choose the weights $\alpha_{\text{top}}^{(t)}$, $\alpha_{\text{bot}}^{(t)}$, and $\alpha_{\Omega}^{(t)}, \forall \Omega \in \mathcal{C}_r^{(t)} \cup \mathcal{C}_m$. We propose to choose the weight $\alpha_{\Omega}^{(t)}$ to balance the contribution to the objective function coming from the common and label-scarce class subsets. That is, we let $\alpha_{\Omega}^{(t)} = 1, \forall \Omega \in \mathcal{C}_m$ and $\alpha_{\Omega}^{(t)} = \frac{|n \in \mathcal{I}_l : c(n) \in \mathcal{C}_m|}{|n \in \mathcal{I}_l : c(n) \in \mathcal{C}_r^{(t)}|}, \forall \Omega \in \mathcal{C}_r^{(t)}$. It appears more difficult to find a principled, "universal" prescription for choosing $\alpha_{\text{top}}^{(t)}$ and $\alpha_{\text{bot}}^{(t)}$ – the "right" level of class decision uncertainty at the t-th labeling step may depend in a complicated way on the sizes of the labeled and unlabeled data subsets, on K, and could also be domain-dependent. We suggest in practice to set these hyperparameters to minimize (over a finite grid of candidate values) a cross-validated error rate measure, based on the labeled samples "seen until now"[16]. The choice of these hyperparameters is further discussed in Section 3.1.6.

While the design of our semi-supervised learning objective (2.18) and its maxEnt regularization is largely motivated to avoid overlearning, it is fundamentally different from Tikhonov-style regularization such as L2. To illustrate this and also compare with other objectives, we synthetically generated a three-cluster example shown in Figure 2.1. We labeled 10 samples from each of classes 1 and 2, treating the rest of the samples as unlabeled. Here, we want to discriminate between class 1 and 2, and make inference on all the unlabeled samples. Because the classification task involves two common categories and no rare categories, our hierarchical posterior reduces to a standard two-class logistic regression, using raw features as input, *i.e.*, no top level posterior and with bottom level posteriors solely on the two common classes. We trained different objective functions and plotted the resulting optimized decision boundaries. Note that MLE, L2 and maxEnt are

[16]Again, in the SL case, the superscript (t) on α_Ω, α_{top} and α_{bot} should be omitted because in this case the unlabeled and labeled data subsets are fixed.

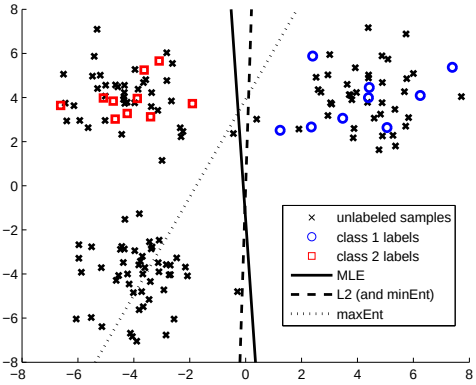


Figure 2.1: A three-cluster example with decision boundaries trained by different objectives: (MLE) Maximum Likelihood Estimate; (L2) MLE with L2-norm; (minEnt) Minimum entropy with weight decay (L2) [20]; (maxEnt) Proposed.

all convex functions, while minEnt is not. As seen in Figure 2.1, all the objectives are able to confidently discriminate between class 1 and 2. However, all objectives except maxEnt classify the unknown cluster (lower left) samples to class 2. On the other hand, maxEnt places the decision boundary in the middle of this high density region with no labeled samples, reflecting minimum classifier confidence (a maximum entropy posterior) in this region – this solution makes sense, since this is a region with purely unlabeled samples. Note that L2, MLE, and minEnt all give very similar solutions – the main difference is that with a smoothness constraint on the weight magnitudes, L2 achieves smaller margin between classes 1 and 2 than MLE. The cross validated minEnt solution is identical to L2's, because in this case, the cross validated error rate is 0 when the hyperparameter on the unlabeled sample's entropy term is set to 0. Note that the proposed maxEnt is highly suitable for unknown class discovery, since most of the unidentified samples are also the most uncertain ones as illustrated in this example.

2.5 Active Learning: Sample Selection Criteria for Oracle Labeling

Tailored for our semi-supervised AL learning objective function, we propose a new sampling criterion suitable for both classification and especially for new class discovery. Specifically, at the t-th oracle labeling, we select $\underline{x}_{i^*}$ to be the most likely unknown class

sample from $\mathcal{X}_u$:

$$i^* = \underset{i \in \mathcal{I}_u^{(t)}}{\text{argmax}} P[C \in \mathcal{C}_{ru}^{(t)} | \underline{x}_i] P[C = 0 | \mathcal{C}_{ru}^{(t)}, \underline{x}_i]. \tag{2.19}$$

Intuitively, the unlabeled sample most likely to belong to the unknown category should also deviate the most from the current common and LS categories, and once labeled, it is expected to either be from a new class or to significantly improve classification accuracy on the existing classes. At the initial AL phase, before any rare category is known, (2.19) reduces to $\underset{i \in \mathcal{I}_u^{(t)}}{\text{argmax}} P[C \in \mathcal{C}_{ru}^{(t)} | \underline{x}_i]$. Thus, our sample selection rule specializes, in this case, to choosing the sample least likely to belong to the common class subset. We call our new sample selection criterion (2.19) *most likely unknown*.

2.6 Experimental Results for Semi-supervised Active Learning and New Class Discovery

In this experimental results section, we will demonstrate:

1. the advantage of p-values as features over other feature mapping techniques (raw, RBF kernel) when one class is rare. Thus, we validate our choice of p-values as derived features;

2. that p-values in conjunction with weight constraints yield sparse solutions;

3. the advantage of the proposed entropy-preserving regularizer (maxEnt) as opposed to minimum entropy (minEnt) regularization in semi-supervised learning settings with unknown and label-scarce classes;

4. that, when applied to AL, our maxEnt semi-supervised learning approach yields a performance advantage over alternative methods, especially during the early phase of AL.

In all of our experiments, the available data set was partitioned into two mutually exclusive subsets – one used for learning and the other for (heldout) testing of generalization accuracy. This in fact was repeated multiple (five) times, with the test performance results averaged.

2.6.1 Performance Metrics

There are two fundamental performance measures of interest on the test batch: i) true detection rate – the proportion of truly unknown class samples (ω_2) that have been

correctly classified and ii) false alarm rate – the proportion of truly common class samples (ω_1) classified as unknown. We evaluate these measures in all our experiments and, in an active learning setting, for the "current" classifier (after every iteration of active labeling). As a comprehensive performance measure encompassing true detections and false positives, we measure average test set area under the ROC curve (ROC AUC). For discriminating among different common and rare classes, we also measure the per-class classification accuracy, averaged over all classes. In the AL setting, before an unknown class is discovered, all of its samples are assumed misclassified.

2.6.2 Data Set Description

Data sets were selected from the UCI ML repository, with both real and categorical features and with naturally rare classes. Most of the chosen data sets have been used in prior related studies [26], [27]. For each of these data sets, we define rare categories as those comprising less than 5% of the whole data set. The main properties of these data sets are shown in Table 2.1. The majority of these data sets have multiple normal and/or multiple rare classes. For the two-class domains (Spam Base and Seismic Pump), normal emails and normal recordings, respectively, are treated as the common category. Since Spam Base has balanced normal/spam email subsets, we randomly subsampled the spam subset to consist of only 5% of the whole batch.

Table 2.1: Data Set Properties: (N) number of samples. (D) number of attributes. (K) number of classes. (K_r) number of rare classes. (SCP) smallest class proportion in percentage.

Data Set	N	D	K	K_r	SCP
Page Block	5473	10	5	4	0.50%
Statlog Shuttle	58000	9	7	4	14e-3%
Yeast	1484	8	10	5	0.34%
(White) Wine Quality	4898	11	7	4	0.10%
Abalone (age 3-23)	4177	8	21	13	0.15%
Spam Base	4601	57	2	1	5.0%
Seismic Pump	2584	18	2	1	5.0%
KDD'99 (10%)	494021	42	15	14	2.4e-3%

2.6.3 Mixture P-values as Features for Supervised Learning

Here, we compare several feature mapping techniques on UCI ML repository data sets, using the support vector machine (SVM) as the supervised learning model. For the p-

value feature mapping, we first used all the available majority class samples in $\mathcal{X}_l$ to build GMM null models; then we measured first and second order mixture p-values as derived features for all the samples. For data sets with both categorical and continuous features, we measured the p-value of a single or a pair of categorical features using frequency counts. For a pair of continuous and categorical features, conditioned on each category, we modeled the first order GMM and measured the GMM-based mixture p-value; then the pair's p-value is the product of the categorical p-value and the mixture-based p-value. We then applied different kernel mappings (linear, RBF) to train an SVM, with the hyper-parameters (on margin slackness and on the RBF's width) chosen via 10-fold cross validation on the learning batch. We compared against linear and RBF kernel SVMs that use the original (raw) features, as well as against SVMs that use both raw and all pairwise feature products. We measured ROC AUC performance as a function of the minority category proportion (from 2% up to 20%). For data sets with multiple classes, we chose the most populous class as the majority class and randomly chose one of the other classes as the minority class. Similarly, for binary classification domains, we chose the most populous class as the majority and the least populous as the minority. Table 2.1 shows various data sets we used to evaluate performance. For each data set, we took the training set file and split it into a learning subset and a test subset, of equal sizes. We then randomly subsampled without replacement from the learning subset, to achieve specified majority/minority class proportions. Again, the learning/testing split was performed five times, with the test subset classification performance averaged over these five trials. Shown in Figure 2.2, on every data set, p-value feature mappings (linear and RBF) gave the overall best results, compared to other feature mappings. Moreover, consistent with our inductive bias, the performance advantage is most obvious when the class proportions are highly skewed (2% and 5%). Also, as demonstrated for Abalone in Figure 2.2c, p-value with the linear kernel gives an almost perfect ROC when few minority samples are present, but with the AUC decreasing as the number of minority samples is increased; on the other hand, p-values used with the RBF kernel mapping shows a reverse trend. This result is not surprising – as the size of the minority class grows, so does the VC dimension, and a more complicated (non-linear) decision boundary may be needed to well-separate the two classes.

2.6.4 Semi-supervised Learning Evaluation

2.6.4.1 Majority v. Minority – a Two Class Classification Study

Here, we compare the performance of maxEnt against minEnt regularization [20][19] and L2 regularization of the weight vector, with all of these methods learning our p-value based class posterior model. As in the previous subsection, five times we repeated a 50-50 split into learning and test subsets, with the classification performance averaged over these five trials. For the learning subset, we randomly selected 20% of both the majority and minority classes as labeled samples, treating the remaining samples as unlabeled ($\mathcal{X}_u$). The class proportions in the test batch were fixed to be the same as in the learning batch. The labeled majority samples in the learning batch were used to learn the null GMM models; we then evaluated both first and second order p-values as derived features for all samples. We measured the average ROC AUC performance for varying minority proportion, from 2% up to 20%. For the approach presented in [20], we used a pair of hyper-parameters, one on the unlabeled sample's entropy and one on the weight decay term (L2 norm). The hyper-parameter values were chosen via 10-fold cross validation applied to the learning batch. In the case of CV error tie, a smaller value on the unlabeled sample's entropy term and a larger weight decay have been selected. For [19], which claims to be an improvement over [20], we supplied the true class priors, which are needed by this model (even though they will not be available in practice). Shown in Figure 2.3 is the average ROC AUC performance on Page Block, along with each method's average entropy on the unlabeled samples and associated weight vector magnitude in Figure 2.4 and Figure 2.5. In addition to evaluating performance for varying minority fraction from 2% to 20%, we also measured the performance for the case where we only have labeled majority samples (*i.e.*, the unknown class scenario), with a 1% minority proportion amongst the unlabeled samples. This performance is shown as the zero percent proportion point in the figure. From the figure, maxEnt regularization gives the best performance, and this coincides with maxEnt having a much smaller weight magnitude and a much higher average entropy on the unlabeled samples than both minEnt methods and the L2 regularization approach. Both of these measures indicate that minEnt "overtrains" relative to maxEnt.

Note that when there are no labeled minority samples in the training batch, both of the minEnt-based models as well as the L2 regularization approach have *no* generalization capability (*i.e.*, the ROC AUC is 0.5 for all of these methods, for this case, because these methods do not correctly classify *any* minority class samples in the test

batch). Scrutinizing the minEnt objective function [20], one can see that when there are no labeled minority samples, the only impetus is to confidently correctly classify the majority class samples. Apparently, this gives no discrimination power for the minority class. These results affirm the importance of overtraining avoidance, especially when the two labeled classes are highly imbalanced.

Results on other data sets are shown in Figure 2.6, for minority labeled fractions from 2% up to 20%. We observe that the maxEnt regularizer gives the best performance in most cases, especially for low minority proportions. Also, as the proportion of labeled minority samples increases, the performance of minEnt approaches that of maxEnt, with even better results on Spam Base and Waveform for the higher class proportions. This is expected, since making confident decisions is plausible when there are "sufficient" labeled examples from both classes. As observed on Page Block, Shuttle, Yeast and Waveform, both minEnt based regularizers combined with the L2 norm perform worse overall than using the L2 norm alone. This again may be attributed to "overtraining", with the L2 norm penalty term insufficiently compensating for the minEnt impetus for "overtraining".

2.6.4.2 Multiple Common and Rare Classes

Here, we are interested in two scenarios of semi-supervised learning: i) with labeled samples from only the common classes (completely unknown rare categories), and ii) with label-scarce rare categories. For the first (unknown class) scenario, from the learning subset, we randomly labeled 20% of samples from each of the common classes, treating the remaining samples as unlabeled ($\mathcal{X}_u$). Thus, all rare categories are unknown. For the latter scenario, besides the 20% labeled common class samples, we also labeled one sample from each of the rare categories. The labeled common class samples in the learning batch were used to learn the 1st and 2nd order null GMM models; we then evaluated both first and second order p-values as derived features for all samples. We measured ROC AUC performance for the unknown class scenario, and measured both ROC AUC and average classification accuracy for the label-scarce case. For the minEnt approach presented in [20], we used a pair of hyperparameters, one on the unlabeled sample's entropy and the other on the L2 norm. The hyperparameter values were chosen via 10-fold CV applied to the learning batch, so as to minimize average classification error of the top level posterior. In the case of CV error ties, a smaller weight on the unlabeled sample's entropy term and a larger weight decay term were selected. For the unknown class scenario (with no ability to measure CV error), all hyperparameters were fixed at

$\frac{|\mathcal{X}_l|}{|\mathcal{X}_u|}$ for all the regularizers, so as to balance the effective sample size between the labeled and unlabeled data subsets.

Shown in Table 2.2 is the ROC AUC performance for the unknown class scenario. Note that when there are no labeled examples from the rare categories, both minEnt and L2 yield degenerate solutions, classifying all samples as coming from the common classes. Thus, these methods wholly fail under the first scenario. Also shown is the performance of two standard anomaly detectors: one using OC-SVM [53] [35] with the RBF kernel and the other a Gaussian Mixture Model (GMM), trained on the labeled common class samples, modeling the full-feature space. For these two methods, we ranked the test samples based on their respective anomaly scores (distance from origin in the case of OC-SVM and data likelihood for the GMM) and used the ordered list to sweep out an ROC curve. We also lower-bound ROC AUC by 0.5. Both of these anomaly detectors lag performance-wise behind our proposed model on most of the investigated data sets. Compared to these full-feature space models, our proposed method's inductive bias (p-values and non-negative constraints) yields a better classifier on these data sets. Table 2.3 shows both the ROC AUC and average classification accuracy for the second (label-scarce) scenario. Again, we see superior performance of our maxEnt regularizer compared to minEnt and L2-norm regularizers, presumably because maxEnt regularization avoids overtraining when there are few labeled rare samples, while minEnt encourages weight magnitudes to increase. However, during active learning, as the number of labeled rare samples increase, minEnt tends to perform well (see next subsection for further discussion). An uninformative prior (L2-norm) also underperforms compared to our proposed data-dependent regularizer.

Table 2.2: Semi-supervised experiment for the unknown class scenario: ROC AUC performance.

Data Set	**OC-SVM**	**GMM**	**Proposed**	**minEnt**	**L2**
Page Block	0.500	0.969	**0.971**	0.5	0.5
Statlog Shuttle	**0.946**	0.927	0.943	0.5	0.5
Yeast	0.802	0.706	**0.824**	0.5	0.5
Wine	0.747	**0.751**	0.738	0.5	0.5
Abalone	0.741	0.725	**0.780**	0.5	0.5
Spam Base	0.500	0.702	**0.874**	0.5	0.5
Seismic Pump	0.554	0.587	**0.704**	0.5	0.5
KDD 99	0.500	0.862	**0.985**	0.5	0.5

Table 2.3: Semi-supervised experiment for the label-scarce class scenario: ROC AUC and average accuracy per-class. (P) Proposed. (M) minEnt. (L2) L2 regularization.

Data Set	ROC AUC			Average Accuracy		
	P	**M**	**L2**	**P**	**M**	**L2**
Page Block	**0.97**	0.93	0.97	**0.41**	0.39	0.35
Statlog Shuttle	**0.93**	**0.93**	0.92	**0.78**	0.73	0.68
Yeast	**0.78**	0.74	0.71	**0.47**	0.42	0.39
Wine	**0.72**	0.70	0.71	**0.29**	0.23	0.17
Abalone	**0.66**	0.62	0.62	**0.21**	0.17	0.19
Spam Base	**0.91**	0.87	0.88	**0.88**	0.73	0.76
Seismic Pump	0.61	0.50	**0.62**	**0.54**	0.50	0.53
KDD 99	**0.98**	0.86	0.86	**0.50**	0.49	0.41

2.6.5 Evaluating Method Variations

2.6.5.1 Majority v. Minority – a Two Class Classification Study

Here, we investigate the performance of our maxEnt regularizer: i) with an additional weight decay (L2 norm) term, to further penalize overtraining; ii) with/without our "inductive bias," *i.e.*, either imposing positive constraints on the weights or allowing unconstrained weights (positive and negative-valued); iii) with our sample weighting scheme on the three data subsets $\mathcal{M}$, $\mathcal{S}$, and $\mathcal{U}$, or with the alternative of equal weighting applied to all samples.

We trained our maxEnt classifier and compared with these variations on several data sets, using the same experimental protocol as in the previous subsections. As shown in Figure 2.7, imposing our inductive bias gives superior performance in *all* of these experiments, compared to a maxEnt regularizer with unconstrained weights. When the two classes are highly skewed, our sample weighting scheme outperforms equal weighting of all samples, on all the data sets. However, as the minority sample size increases, our weighting scheme sometimes (apparently) "over-represents" the minority class, yielding poorer generalization performance, as observed on both Waveform and Abalone. The L2 weight decay's effect is domain-dependent: on Spam Base and Waveform, the weight penalty term helps improve maxEnt generalization performance, but on the other data sets it gives mixed results. Further investigation of different combinations of regularizers may be a good subject for future work.

Table 2.4 shows that, even though our basic maxEnt method (without an L2 norm term) does not explicitly encourage zero weights, on all the data sets there is a significant degree of sparsity in the solution – on Spambase, for the case of a 5% minority fraction,

only 3.7% of all the weight parameters are non-zero. On Waveform, only 6.35% are non-zero. The maximum degree of non-sparsity is on Statlog Shuttle, with 34.5% of weight parameters non-zero. Our inductive bias is thus seen to give propensity for sparse solutions, which is useful when dealing with many uninformative features.

Table 2.4: Sparsity of the maxEnt model: average number of non-zero weight parameters for semi-supervised learning with 5% minority class proportion.

Data Set	No. active parameters	(%) of active parameters
Page Block	9	16.36
Statlog Shuttle	16	34.5
Spam Base	61	3.7
Abalone	11	22.7
Yeast	10	27.78
Waveform	16	6.35
Seismic	15	8.77

2.6.5.2 Multiple Common and Rare Classes

Table 2.5: Compare model variations: average classification accuracy per-class and fraction of inactive model parameters in Λ. (P) Proposed. (Uc) Unconstrained Parameters. (Uw) Uniform Weighting. (MLE) Maximum Likelihood.

Data Set	**Average Accuracy**					**Inactive Parameters**				
	P	**Uc**	**Uw**	**Raw**	**MLE**	**P**	**Uc**	**Uw**	**Raw**	**MLE**
Page	**0.41**	0.39	0.25	0.37	0.37	0.78	0	**0.80**	0	**0.80**
Statlog	**0.78**	0.76	0.61	0.51	0.66	0.49	0	**0.54**	0	0.51
Yeast	0.47	0.47	0.27	0.48	**0.50**	0.78	0	**0.89**	0	**0.89**
Wine	**0.29**	0.22	0.19	0.23	0.23	**0.74**	0	0.42	0	**0.74**
Abalone	**0.21**	0.17	0.11	0.12	0.16	**0.45**	0	0.31	0	**0.45**
Spam	**0.88**	0.74	0.82	0.61	0.82	0.95	0	0.76	0	**0.97**
Seismic	**0.54**	0.50	0.51	0.49	0.53	0.84	0	**0.86**	0	**0.86**
KDD99	**0.50**	0.45	0.48	0.46	0.48	0.90	0	0.90	0	**0.91**

In a multiple common and rare classes scenario, we compare our proposed scheme (P), with maxEnt regularization, p-values as features, our hierarchical posterior, with non-negative constraints on weights in the top level, and employing our sample weighting choice for $\alpha_{\Omega}^{(t)}$ against alternatives that each vary some of our design choices: i) using raw and pairwise product features and unconstrained weights, instead of p-value based features in the top level (Raw); ii) allowing unconstrained weights (positive and negative-valued) on p-value features (Uc); iii) applying equal weighting to all samples

(Uw); iv) maximizing the posterior log-likelihood without performing maxEnt regularization (MLE). Ten times we repeated a 50-50 split into learning and test subsets, with generalization performance averaged over these ten trials. In each trial, in the learning subset we randomly labeled 20% of samples from each of the common classes, which are used to train the null GMM models. P-values for each sample are calculated from the null models. Besides the 20% labeled common samples, we also labeled one sample from each of the rare categories. Thus, all categories are known. Then we trained our proposed maxEnt classifier and these four method variations on several data sets. Hyperparameters ($\alpha_{\text{top}}^{(t)}$ and $\alpha_{\text{bot}}^{(t)}$) are chosen via 10-fold cross validation (CV). Although it seems that the best CV method is to minimize the hierarchical posterior's error, averaged over all classes, in the LS case, we do not have enough samples from each class to perform such validation. As an alternative, we minimized (over a finite grid of candidate values) a 10-fold cross-validated average error rate measure of the top level posterior. In the case of CV ties, a larger value of $\alpha_{\text{top}}^{(t)}$ and a larger value of $\alpha_{\text{bot}}^{(t)}$ were chosen.

Shown in Table 2.5, imposing a non-negative constraint on the top level weights gives superior performance in all of these experiments, compared to a scheme with unconstrained weights. There is large improvement that comes from zero-weighting the features that are deemed uninformative. Moreover, use of weight constrained p-value features improves generalization performance compared to the use of raw and pairwise product features. But when the top level weights are unconstrained, the use of p-value features performs similarly to the use of raw and pairwise product features on the majority of the data sets. Hence, our inductive bias (both non-negative weight constraints and p-value features) acts jointly to achieve superior performance. Third, maxEnt regularization is seen to be vital to achieving good performance, substantially outperforming classifiers trained without this regularization. That is, unlike MLE, the proposed regularizer avoids over-training based on the few labeled rare samples. Finally, the effect of sample weighting is data-set dependent, but when non-uniform weighting is effective, it gives a large improvement over equal weighting. Thus, the experimental comparison in Table 2.5 supports our method's use of i) maxEnt regularization, ii) p-values, iii) non-negative top level weights, and iv) non-uniform sample weighting.

Table 2.5 also shows that p-values as features and non-negativity constraints on top level weights act jointly to achieve a substantial level of parameter sparsity (zero weights) in the solution – P, Uw, and MLE, which all use p-values and non-negative top level weights, achieve high top level feature sparsity. Note also that this is achieved with only *one* labeled sample from each rare category. Thus, the degree of sparsity

does not appear to strongly depend on the number of labeled rare (unknown) category samples. On Spambase, only around 5% of top level weight parameters are non-zero for our proposed method. The maximum degree of non-sparsity for our method is on Statlog Shuttle and Abalone, with more than 50% of weight parameters non-zero. By contrast, the method variations that allow unconstrained weights and which use raw features rather than p-values achieve *no* degree of sparsity (all nonzero weights). Finally, note that sparsity by itself is not an indicator of good performance – the MLE method (without maxEnt regularization) achieves the highest degree of sparsity on all data sets, but it is outperformed, accuracy-wise, by the proposed method on all the data sets.

2.6.6 Active Learning Evaluation

In this section, we investigate the use of our semi-supervised maximum entropy learning approach within a pool-based AL setting, with samples chosen for oracle labeling one-by-one. In principle, the null model GMMs could be adapted when a new oracle labeling is from a common class. However, we fixed the null models after their initial training because there were an adequate number of common class samples in the initial $\mathcal{X}_l$.

Other than our proposed criterion, we also evaluate the following AL sample selection criteria:

Most uncertain: the unlabeled sample $\underline{x}_{i^*}$ with greatest class uncertainty (as measured by Shannon's entropy function $H(C|i^*)$ over all the known classes and the unknown class) is forwarded to the oracle. Intuitively, if an unlabeled sample is most uncertain for the current classifier, the removal of this uncertainty (by oracle labeling) should provide valuable information to the classifier.

Least common: the unlabeled sample $\underline{x}_{i^*}$ with largest *a posteriori* probability of not belonging to the common class subset (as measured by $P[\mathcal{C}_{ru}^{(t)}|\underline{x}_{i^*}]$) is forwarded to the oracle. This choice prioritizes accuracy in discriminating between common and LS/unknown classes.

Random: the unlabeled sample $\underline{x}_{i^*}$ is randomly chosen for oracle labeling. This choice gives a lower-bound baseline for AL performance.

To assess performance, we measured test set ROC AUC and average classification accuracy as a function of the number of oracle labelings. We assumed the unknown class scenario at the outset, with no initially known (labeled) rare class samples in the learning batch, and with 20% from each of the common classes samples labeled. The experimental protocol was the same as in previous subsections, with results averaged over ten learning/test splits.

Comparison of Sample Selection Schemes

Active learning performance may be affected by the choice of classifier structure, learning method, as well as by the scheme for choosing which samples to label. First, we compared different active sample selection strategies, using our maxEnt model as the classifier. Initially, since there are no labeled rare class samples, it may appear tempting to select for labeling the sample least likely (as measured by the model) to belong to the common class subset. Indeed, when using this strategy as the AL sample selection mechanism, we are able to identify many more rare class samples than either "most likely unknown" or "most uncertain" sampling. However, as next shown, the generalization performance of a classifier using this "least common" strategy can be poor. Besides the strategies mentioned in Section 2.5, we also show the performance achieved when all the samples in the learning batch are labeled (this should upper bound performance of AL).

In Figure 2.8, we show both ROC AUC and average classification accuracy as a function of oracle labelings, while in Table 2.6, we show the average number of true detections and the fraction of rare classes discovered after 100 oracle labelings. We emphasize that the "average number of true detections" is not a classification performance measure – it merely indicates how frequently the sample selection strategy forwards unknown/rare class samples to the oracle for labeling. As seen in Table 2.6, the "least common" strategy forwards many more unknown/rare class samples to the oracle on average than "most likely unknown" and "most uncertain", with uncertainty sampling forwarding the fewest. This can be explained by the fact that the number of common class samples close to the decision boundary dominate the rare class ones. However, as seen in Figure 2.8, both "least common" and "most uncertain" have poorer generalization performance than "most likely unknown". Note the initial drop in performance of "most uncertain" sampling on the Wine data set. Here, the labelings of "most uncertain" samples are clearly not the most helpful for classification. Moreover, while the "most likely unknown" and "least common" curves in Figure 2.8 start out together because the two criteria are *equivalent* until the first unknown category sample is labeled, the two curves diverge after this first labeling. The "least common" strategy has a tendency to identify many rare class samples that are very similar to each other, and far from the decision boundary – it does not tend to select informative samples that either are close to the decision boundary or are instances of novel classes. Thus, as shown in Table 2.6, it tends to have higher true detections but a lower fraction of rare classes discovered compared to "most likely unknown" (see results on Yeast and KDD). It also has poorer generalization performance than "most likely unknown" seen in Figure 2.8 (Note in particular results

on KDD, Wine, Page Block, and Yeast).

In some applications, where, e.g., the rare class samples are "suspicious" and hence urgently actionable, there may thus be a need for two (concurrent) operational labeling "streams" [51]. One is the most actionable samples: these are the unlabeled samples most likely to be rare (suspicious), whose identification may lead to subsequent action by a human operator. The other is the most informative samples: these are the samples whose labeling will help to improve the generalization accuracy of the classifier the most. The new "most likely unknown" method achieves the best results with the highest fraction of rare classes discovered, and sometimes comes close to the performance upper bound line.

Note that, for all these strategies, there is an initial drop in ROC AUC while the average accuracy increases, on all the data sets. The reason is that, when the first few rare class samples are labeled, the non-uniform weighting takes effect, and there is a trade-off between false positives and classification accuracy between the common and rare class subsets. As more rare samples are labeled and new classes are discovered in latter AL stages, ROC AUC tends to increase and surpass the initial level.

Table 2.6: AL true detections and fraction of rare classes discovered out of 100 iterations for different sample selection strategies: (MLU) most likely unknown. (MU) most uncertain. (LC) least common and (R) random.

Data Set	**AL True Detections**				**Rare Classes Discovered**			
	MLU	**MU**	**LC**	**R**	**MLU**	**MU**	**LC**	**R**
Page Block	35	12	**85**	15	**1.0**	**1.0**	**1.0**	0.75
Statlog Shuttle	60	4	**68**	2	**0.93**	0.75	**0.93**	0.12
Yeast	24	6	**49**	12	**0.93**	0.62	0.74	0.64
Wine	14	5	**34**	9	**1.0**	0.83	0.99	0.70
KDD'99	**33**	18	31	4	**0.71**	0.57	0.32	0.16
Spam Base	48	11	**49**	6	**1.0**	**1.0**	**1.0**	**1.0**

Comparison of Active Learning Models

Next, we compared our proposed maxEnt model (using the "most unlikely unknown" strategy for sample selection) with other alternatives: SVM based, minEnt based and a generative model based, elaborated as follows. For an SVM based approach, we first used OC-SVM with the RBF kernel and queried the most anomalous samples, until the first rare class sample was selected and labeled; then we switched to using the standard multi-class SVM with RBF kernel, labeling the most uncertain sample. However, this approach requires knowledge of the number of classes and, to initiate the learning, requires labeled instances from every class. As for a minEnt based approach (as opposed to our maxEnt

model), we chose the approach presented in [20]. We also compared with the recent model presented in [27], which we dub "Unified", a generative model that uses a Dirichlet Process to model the true class distribution, with expected classifier improvement as the sample selection mechanism. This approach was shown to outperform a number of previous works on rare category detection and characterization, e.g., [26].

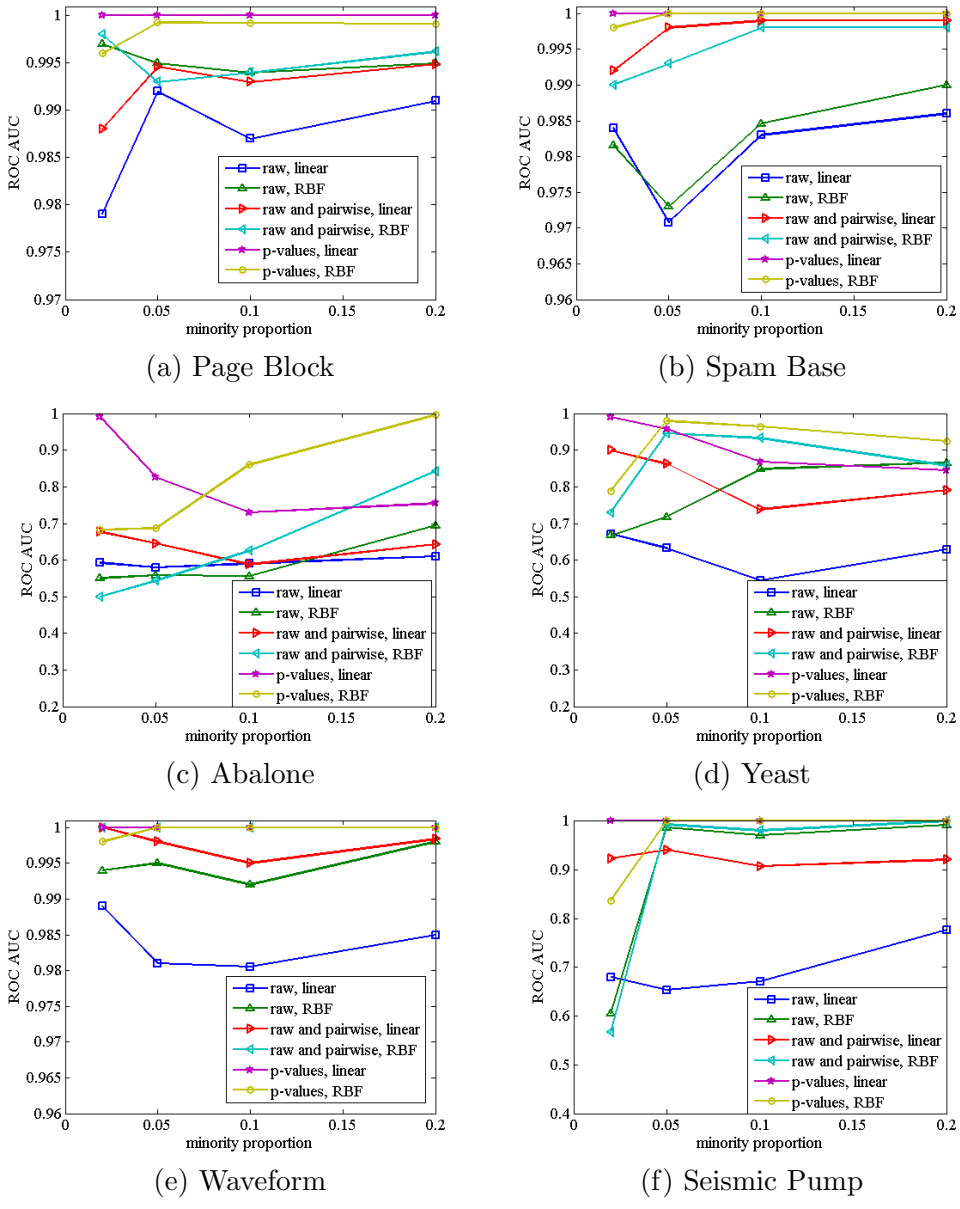
In our initial AL experiments, we found that the minEnt approach achieves *no* generalization capability – *i.e.*, the method classifies all samples as common class (see previous subsection and Table 2.2) and never selects any samples from the unknown class. This represents a fundamental breakdown of the minEnt approach within an "initially unknown class" AL setting. Subsequently, in order to at least allow minEnt to achieve some generalization performance, we "fed" the minEnt classifier with the labeled samples selected by our maxEnt approach. In this way, "informative" samples determined by our proposed model were also used to train the minEnt classifier. As seen in Figure 2.9, it is apparent that minEnt, even when fed the labeled samples selected by the maxEnt approach, achieves no generalization power in the early AL stage, on all the data sets. However, as observed for KDD'99, as more and more rare class samples are included into the labeled subset, minEnt improves, and performs quite well in the latter iterations, sometimes even better than maxEnt at a certain point. Compared with [27], our maxEnt approach achieves overall superior performance on all the data sets except KDD'99, especially during the early AL phase. Note that our model's ROC AUC curve outperforms all others on all data sets except Statlog Shuttle and Wine.

For comparing our method with the SVM, note that for the leftmost part of the curve, before any rare class samples have been labeled, we are really comparing our approach with OC-SVM. The SVM approach, which does not make effective use of unlabeled samples to regulate the learning, performs poorly on all the data sets.

2.7 Conclusion

In this chapter, we proposed novel SL and AL frameworks, along with a class posterior model, for learning to discriminate unknown from common classes. Our use of a hierarchical class posterior accommodates multiple common, multiple LS, and unknown classes. Our top-level class posterior imposes a constraint on the parameter space (non-negative weights on p-values) consistent with the inductive bias that what is unknown is a subset of what is anomalous relative to the common classes. Experimentally, we found that imposing this inductive bias leads to sparsity in the solution, with only a

small subset of feature p-values contributing to the class decision. Our SL approach exploits unlabeled samples in an unconventional way to control the amount of learning given few labeled samples. That is, we proposed a novel (maxEnt) sample-dependent semi-supervised regularizer, suitable for AL with unknown categories and for SL with high class imbalance in general. We also proposed a sample selection criterion for AL which focuses on the discovery of novel classes. Our framework is particularly suitable for learning a mapping between a human operator's concept of an unknown class (which could be based on complementary, human-digestible information sources, e.g., images or text) and an information rich, albeit possibly very low-level high-dimensional feature space. The classifier achieves this mapping, identifying which (low level) features prominently capture the operator's concept. Future work could investigate encoding higher order interactions into our posterior model. It could also explore a hybrid approach, with maxEnt regularization used during the early AL stage, transitioning to minEnt regularization in the later learning phase. We could also consider online learning of the null. We may also explore a hyperparameter to control the mass of the unknown category. Finally, we could investigate the effect of nonconvexity of the learning objective on the quality of classifier solutions we obtain.



(a) Page Block (b) Spam Base

(c) Abalone (d) Yeast

(e) Waveform (f) Seismic Pump

Figure 2.2: Supervised classification experiment: average ROC AUC performance comparison of different feature mappings input to linear and RBF kernel SVMs on various UCI data sets.

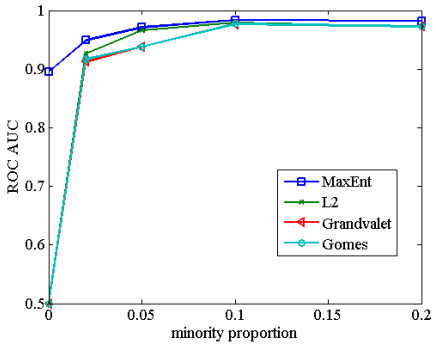


Figure 2.3: Page Block: ROC AUC.

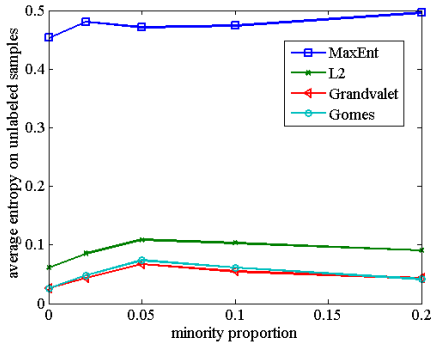


Figure 2.4: Page Block: Entropy.

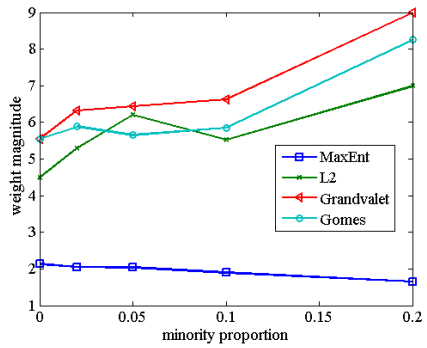
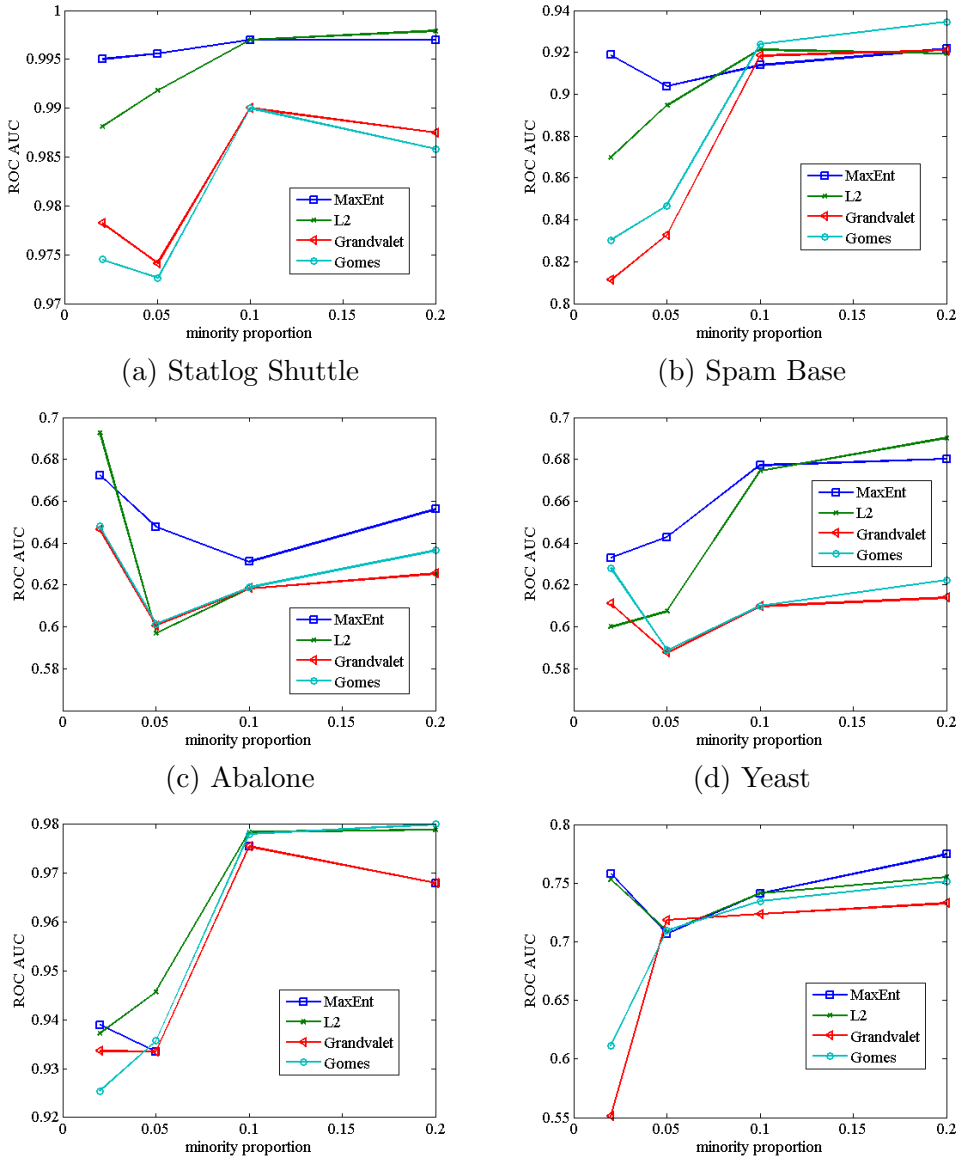


Figure 2.5: Page Block: Weight vector magnitudes.

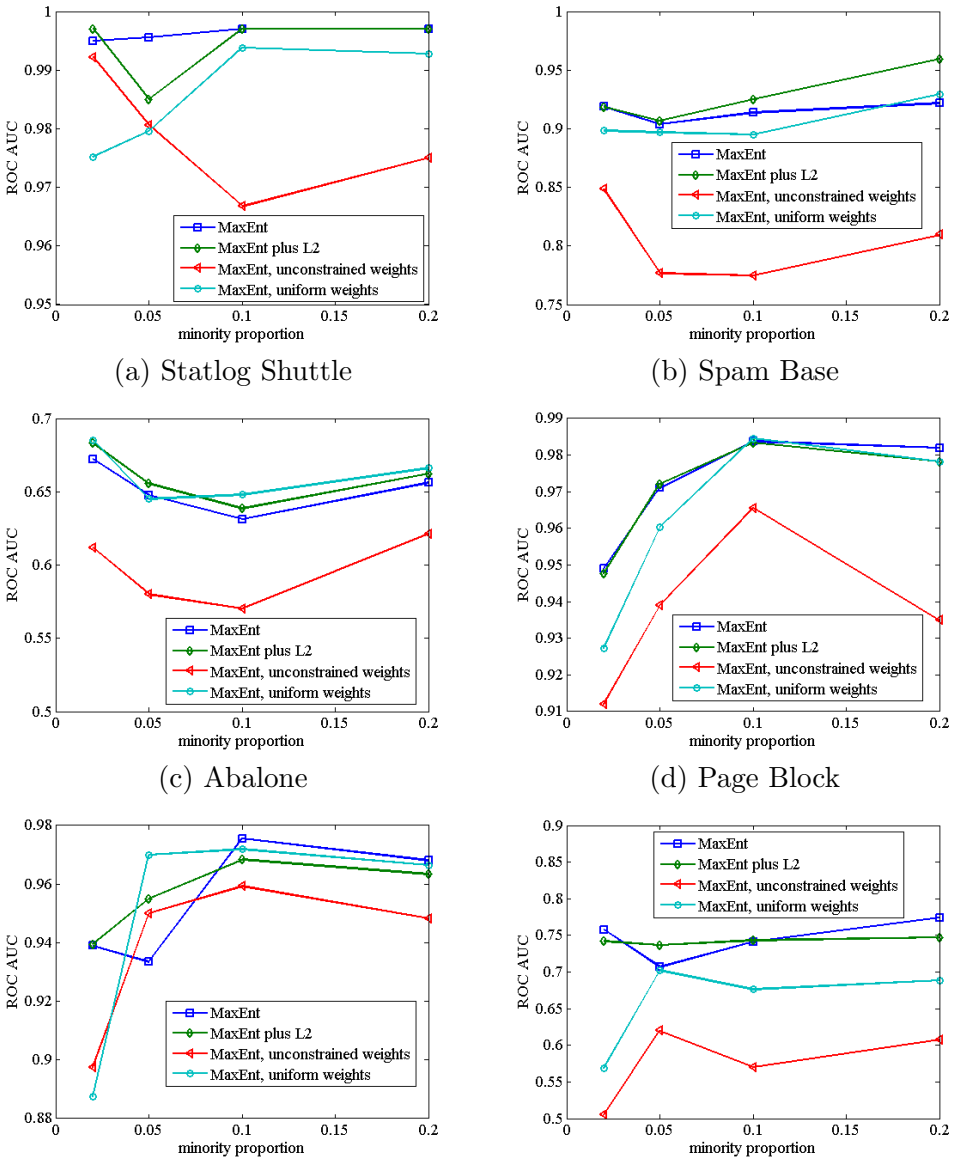


(a) Statlog Shuttle (b) Spam Base

(c) Abalone (d) Yeast

(e) Waveform (f) Seismic Pump

Figure 2.6: Semi-supervised classification experiment: average ROC AUC performance comparison of different regularizers applied to learn our posterior model on various data sets.



(a) Statlog Shuttle (b) Spam Base

(c) Abalone (d) Page Block

(e) Waveform (f) Seismic Pump

Figure 2.7: Performance comparison of proposed model variations on various data sets.

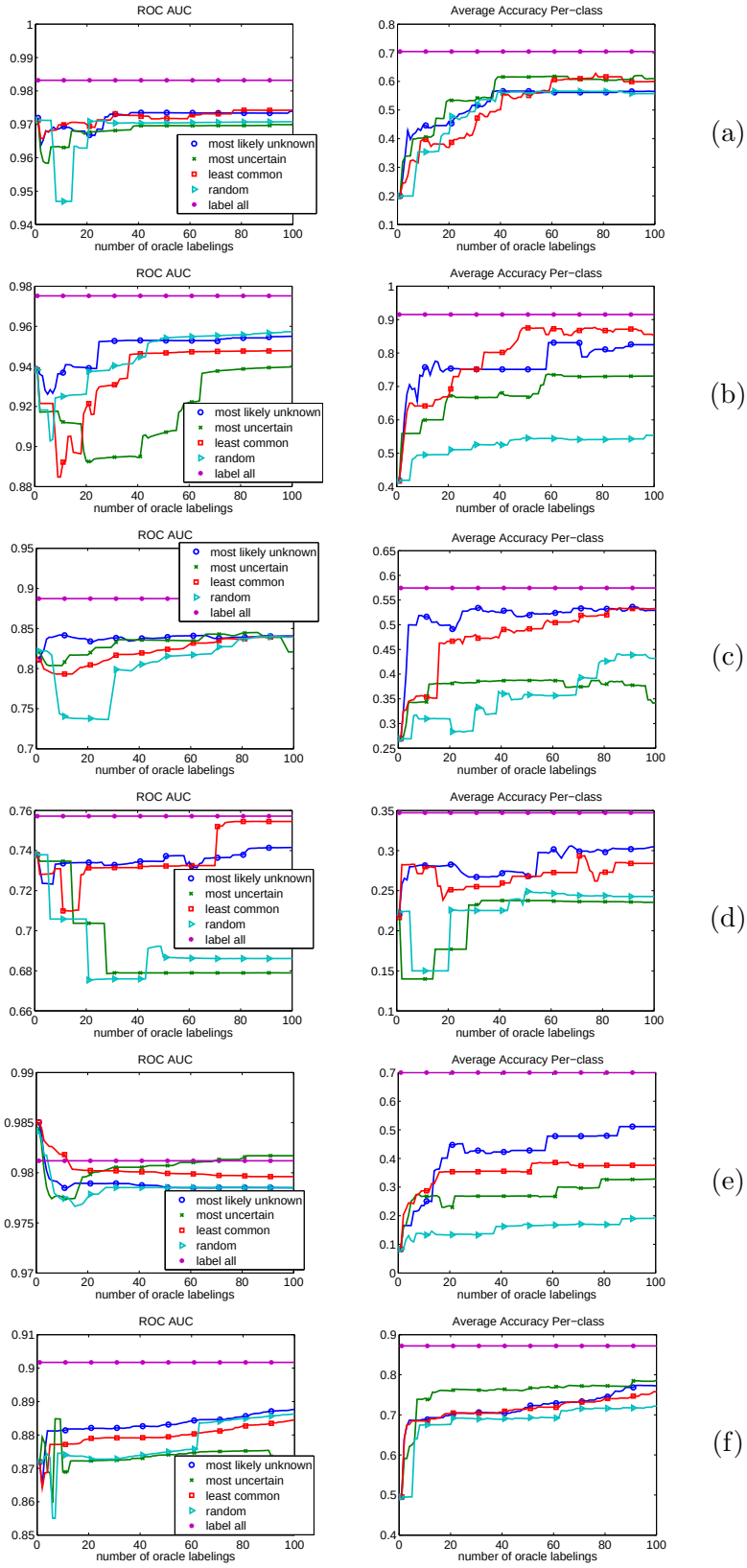


Figure 2.8: Active learning experiment for different sample selection strategies. (a) Page Block. (b) Statlog Shuttle. (c) Yeast. (d) Wine Quality. (e) KDD'99. (f) Spam Base.

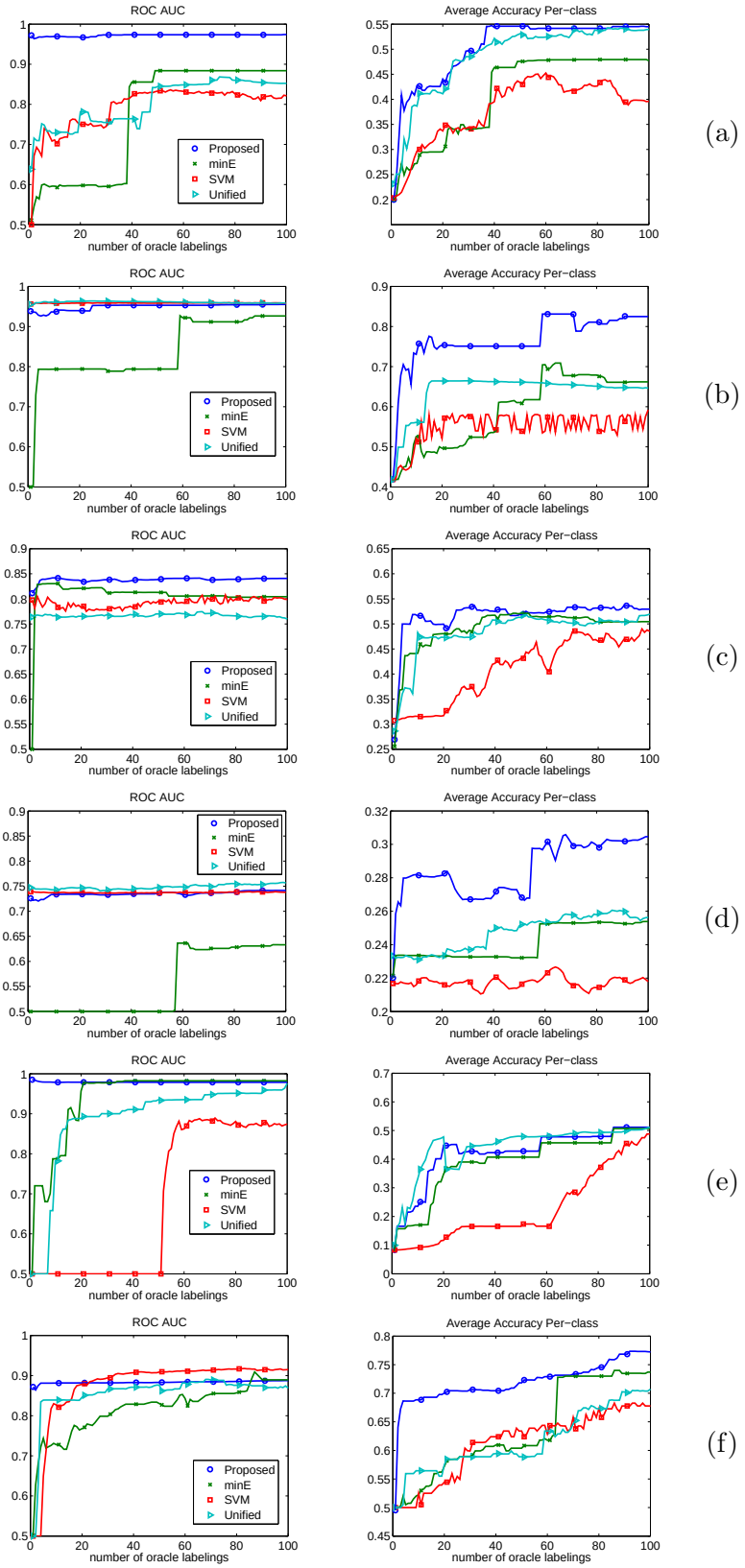


Figure 2.9: Active learning experiment: comparison of methods' ROC AUC performance and average classification accuracy versus number of oracle labelings, on various data sets. (a) Page Block. (b) Statlog Shuttle. (c) Yeast. (d) Wine Quality. (e) KDD'99. (f) Spam Base.

Chapter 3

Applications of The Semi-supervised Active Learning with Maximum Entropy Regularization

In this chapter, we investigate two applications of the semi-supervised active learning with maximum entropy regularization. The first section focuses on active detection of suspicious vehicle tracks [51]. The second section focuses on a network intrusion detection system [50].

3.1 Active Learning to Distinguish Suspicious from Innocuous Anomalies in a Batch of Vehicle Tracks

In anomaly detection, one often aims to identify the most atypical samples from a potentially large "test batch" of observed samples. This is generally assessed by evaluating the test batch relative to a reference model of what is "normal" (which may itself be learned based on a large representative training set of "normal" (null) instances). However, for a deployed system, in practice, one may find that many samples in the test batch will be detected as statistically significant anomalies. This may be due to limited prior knowledge and/or due to limited data available for accurately characterizing what is "normal". It may also be due to measurement inaccuracy, e.g., in astronomy many (uninteresting) anomalous objects may be detected if there is an equipment glitch, or due to human error in system configuration or data collection [46]. In fact, what is often genuinely desired is not to detect all the anomalous samples, but rather to detect and to prioritize what, for

some domains, amounts to the most suspicious anomalies (*i.e.*, for behavioral tracking, intrusion detection, email spam) and for others (scientific applications) amounts to the most interesting ones, *i.e.* those associated with heretofore unknown physical phenomena. For example, considering vehicle tracking, the main target domain in this section, speeding off-road or in a school zone may be highly suspicious, whereas speeding on a local highway or making a U-turn may be merely atypical. There are several challenges associated with detecting the most suspicious (most interesting) samples. First, "suspicious" may be a very small subset of the anomalous class, consisting of samples whose atypicalities are with respect to certain key (albeit *a priori* unknown) features and/or feature interactions. Indiscriminately forwarding all detected anomalies to an operator for scrutiny or action will not scale; it will swamp the operator, and likely fail to detect any of the truly suspicious anomalies[1]. Thus, in practice, it is necessary to discriminate between suspicious anomalies and innocuous ones. This seems to suggest treating the problem as one of standard supervised learning. However, there are several distinctive aspects of this problem. First, because anyway they tend to be rare, there may initially be no known (supervising) examples of what is truly suspicious/interesting. Moreover, what is *truly* suspicious may in fact be subjective, and thus may need to be informed by each (expert) consumer of an anomaly detection report. Thus, an *active learning* framework [55] and one that starts from *scratch* (*i.e.*, no initially labeled suspicious instances), is in general needed here. Second, for this suspicious/innocuous discrimination problem, there is in general an inherent, challenging feature selection problem. There are multiple reasons for this. First, since "suspicious" may be a wholly unknown class, it is *a priori* unknown which features may be needed to characterize it. Thus, many features may need to be measured and assessed. Second, some suspicious (e.g., malicious) activities may seek to be as *obfuscating* as possible, in which case the suspicious signature – its anomalousness – may only be exhibited with respect to a very small subset of all possible features. Thus, if one restricts the features under consideration, one may wholly *miss* the suspicious signature. On the other hand, many of the chosen features may be irrelevant (nuisance features) – anomalies with respect to these will not be indicative of suspiciousness. These features will degrade the classification accuracy unless they are eliminated or de-emphasized in the decision-making. It is also useful in general to identify what are the essential features and feature combinations – in this way, the classifier

[1]Even if the rate with which anomalies can be inspected by a human operator matches the rate at which anomalies are detected by a system, the likelihood of an operator to identify rare, suspicious anomalies will be low. But this is anyway far too optimistic – the rate of anomaly detection may be much higher than the inspection rate.

may make explicit and objective the basis upon which a (subjective) human operator distinguishes between anomalies that are suspicious (interesting) and those which are innocuous (uninteresting).

In summary, for numerous applications, ranging from vehicle tracking to email spam detection and Internet traffic monitoring, it is desirable to develop an active learning framework for prioritized identification of activities that are suspicious (interesting) as opposed to being merely atypical. In this work, we develop a novel method with such capability, focused particularly on the domain of entity tracking in wide area motion imagery (WAMI) video.

3.1.1 Track Data and Its Raw Features

The data we work with is based on less than one hour of video capturing an urban scene. We applied Toyon's in-house track detection and tracking software based in part on [1], which applies a variety of paradigms in achieving accurate, computationally efficient persistent entity tracking. Application of this system captured more than 1000 vehicle/entity tracks from the given scene. Based on Toyon's software, each tracked entity is represented by a (variable duration) multivariate time series ($\sim$ 1 Hz sampling) where, at each time point, there is instantaneous measurement of longitude, latitude, speed, and azimuth (heading). While these comprise the "raw" features, one can also compute derived features such as instantaneous acceleration, as well as track-level features such as range or total distance covered by a track, and track duration. Note that absolute timing features (time-of-day), while expected to be revealing in general (unusual vehicle activity at 4 A.M.), are not relevant for the data here, comprising less than one hour. Atypicality of instantaneous features such as speed and azimuth are best assessed by conditioning on the road segment of origin - 60 m.p.h. instantaneous speed is normal on a highway but abnormal on a suburban road. Likewise, an instantaneous azimuth inconsistent with the direction of the road being travelled (e.g., suggesting the vehicle is moving off-road) is unusual. Accordingly, we evaluate feature distributions under the null hypothesis (of no anomaly present) while conditioning on the road segment of origin [2]. Beyond speed and heading features, we also derive features that indicate whether a vehicle track consists of U-turns, looping (wherein the track crosses itself), as well as the proportion of the track that consists of off-road travel. U-turns are indicated by counting the number of pairs of points along the track trajectory that are within ϵ-distance of each other in opposite directions. Looping is indicated by pairs of "time-separated" points

[2] For the given scene, the road network was determined using Openstreetmap [24].

along a track trajectory that are within ϵ-distance.

3.1.2 P-Values as Features for Discriminating Suspicious from Merely Anomalous

The fundamental inductive bias that we exploit is that suspicious behavior is a subset of anomalous behavior - *i.e.*, the subset for which atypicalities are with respect to certain key (albeit *a priori* unknown) features and/or feature combinations. Consistent with this inductive bias, we treat, as feature inputs to our classifier, log p-values [3] with respect to the different measurement types (speed, azimuth, acceleration, and others) - a very negative log p-value indicates strong atypicality of a given feature type, which *may or may not* be indicative of suspiciousness – the active learning essentially needs to "figure this out" by learning suitable weights on these log p-value features. Measuring p-values on raw features before feeding to the classifier can also be thought of as an alternative (motivated by our inductive bias) to (agnostically) applying some nonlinear kernel transformation to the original raw features.

For each track, we computed track-level p-values associated with U-turns and track duration. We also computed, for each road visited along a track, speed, azimuth, acceleration, and change-in-azimuth p-values (one p-value is computed for each individual raw feature measurement). Each of these types of measurements, conditioned on the road of origin, is modeled by a mixture of Gaussians (unimodal, in some cases). For unimodal distributions, we separately measure both the left and right tail p-values (e.g., unusually high speed on some road could be innocuous, while atypically low speed may be suspicious). For features whose distributions are multimodal, we model using a mixture of Gaussians, and evaluate a *mixure-based p-value*, as detailed in Section 2.2. In order to fairly evaluate tracks that travel over different numbers of roads, for each track, for each feature type, the top K most extreme p-values are retained (K left-tail and K right-tail p-values, in unimodel case), considering all roads traveled over the duration of the track. In this way, for every track, the same number of log p-values of each type comprise the features that are the inputs to our classifier. We also included top K mixture p-values for every *pair* of feature types, based on bivariate Gaussian mixtures and mixture-based p-values, as detailed in Section 2.2. Experimentally, we found that $K = 10$, which allows capturing multiple atypical events during a track, is an effective choice. This leads to 142 total number of (log p-value) features, representing every track.

[3] A p-value is the probability of seeing a realization more extreme than an observed one, under a given null hypothesis. See Section 2.2.

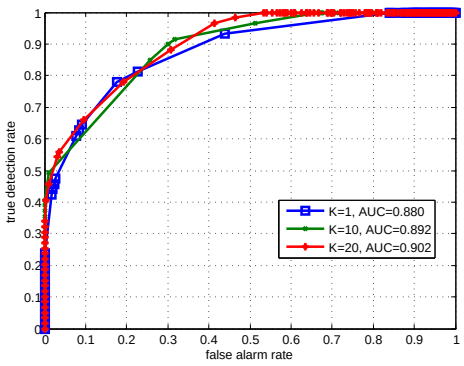


Figure 3.1: Compare ROC of different Ks in Track Data.

In order to understand how suspicious vs. innocuous classification accuracy on the Track domain varies with the number of p-value features, we evaluated the test set ROC as a function of K, with the classifiers trained in a supervised fashion, using all labeled training examples. While there is an increasing performance trend with increasing K, the accuracy gains are diminishing in going from $K = 10$ up to $K = 20$, as shown in Figure 3.1. Accordingly, we set $K = 5$ in our subsequent Track experiments. We further note that K is essentially practically limited by the *minimum* number of measurements for a track, across all tracks in the captured batch of tracks[4].

3.1.3 Classifier Model

Our active learning approach can be applied for various class posterior models. Here, it is developed assuming a "nearly standard" logistic regression classifier form, albeit with log p-values as features and with weight constraints:

$$P(C = \text{“suspicious”}|\underline{p}) \quad = \quad \frac{\exp(w_0 - \sum_{i=1}^{N}(w_i \log p_i + \sum_{j \neq i} \beta_{ij} \log p_{ij}))}{1 + \exp(w_0 - \sum_{i=1}^{N}(w_i \log p_i + \sum_{j \neq i} \beta_{ij} \log p_{ij}))} \quad (3.1)$$

based on the parameter set $\Theta = \{\{w_i\}, \{\beta_{ij}\}\}$, with $w_i \geq 0, \beta_{ij} \geq 0, \forall i, j$. Here, p_i is a p-value for feature i, with p_{ij} a p-value for the (i, j) feature type pair [5]. See Chapter

[4]For the few instances where a track does in fact have fewer than K measurements, we (uninformatively) "fill out" the missing log p-values with zeros (corresponding to high typicality).

[5]For clarity's sake, to make the above expression "uncluttered", the above equation does not distinguish, for unimodal distributions, between left and right tail p-values, and it assumes one (most extreme) p-value, for each of N feature types. More generally, our classifier will indeed make use of both left and right tail p-values for unimodel distributions, and it will use the top K p-values for each feature type to represent a given track.

2 for details in the properties (interpretability, monotonic sample ordering, etc.) and development of the model.

3.1.4 Semi-supervised Active Learning Objective Function

At the outset, we may have few or *no* labeled suspicious examples, but plentiful examples that can be (even without human inspection) reliably labeled as "innocuous" based on high typicality (relatively large aggregate p-values). Essentially, there are two classes, but with labeled instances initially available for only one of them. Thus, at the outset, our active learning problem amounts to a "one-class classification" problem [43]. We are proposing a pool-based active learning framework, with repeated selection of a sample from the test batch for labeling and then adaptation of the existing classifier model in light of this new labeled example. What distinguishes between different pool-based schemes are: 1) the criterion used for selecting the sample to forward to the oracle (operator) for labeling at each step and 2) the learning framework for adapting the classifier model in light of the newly labeled sample. In this section, we focus on the classifier learning, developing a novel semi-supervised learning framework, a variant of the general framework proposed in Chapter 2, that we believe is suitable for active learning of classifiers in general (not just for suspicious-innocuous discrimination). In section 3.1.6, we will show that this learning scheme, used in conjunction with our logistic regression classifier model, in fact gives improvement over support vector machine-based active learning (with both methods using uncertainty-based sample selection).

Our main focus, as detailed in Chapter 2, in devising a learning framework and objective function, is to make appropriate, effective use of the *many* unlabeled samples, in addition to the current cadre of labeled innocuous and suspicious samples. If we were only to make use of the labeled samples, a standard approach for (supervised) learning of a class posterior model is maximization of the posterior log-likelihood over the labeled training samples. Posterior log-likelihood maximization is in fact equivalent to minimization of a sum of cross entropies objective function, specified as follows. For a given labeled suspicious feature vector $\underline{p}_n$, we give the "target" distribution $(Q_{\text{target}}[C = 0|\underline{p}_n], Q_{\text{target}}[C = 1|\underline{p}_n]) = (0, 1)$, while for a labeled innocuous sample $\underline{p}_n$, we give the targets $(Q_{\text{target}}[C = 0|\underline{p}_n], Q_{\text{target}}[C = 1|\underline{p}_n]) = (1, 0)$. Then, after the t-th oracle labeling of active learning, maximizing the posterior log-likelihood model $P[C|\underline{p}]$ is equivalent to

minimizing

$$\begin{aligned} D^{(t)} &= \sum_{n\in\mathcal{I}^{(t)}} d(Q_{\text{target}}[C|\underline{p}_n]||P[C|\underline{p}_n]) + \sum_{n\in\mathcal{S}^{(t)}} d(Q_{\text{target}}[C|\underline{p}_n]||P[C|\underline{p}_n]) \\ &= -\sum_{n\in\mathcal{I}^{(t)}} \log P[0|\underline{p}_n] - \sum_{n\in\mathcal{S}^{(t)}} \log P[1|\underline{p}_n]. \end{aligned} \tag{3.2}$$

Here, $d(Q||P) = \sum_l Q_l \log(Q_l/P_l)$ is the Kullback-Leibler distance [30] (cross entropy) between probability mass functions Q and P, where $C = 0$ denotes "innocuous" and $C = 1$ denotes "suspicious". Q is interpreted as the "true" distribution, with P our (model-constrained) approximation of the true. Also, $\mathcal{I}^{(t)}$ and $\mathcal{S}^{(t)}$ are, respectively, the labeled innocuous and labeled suspicious subsets of the test batch after the t-th oracle labeling, where $t = 0$ means before any oracle labeling has been performed.

Similar to the development in Chapter 2, we seek to modify (3.2) in two respects: i) to sensibly incorporate and thus exploit for learning the (typically large) current subset of *unlabeled* samples, $\mathcal{U}^{(t)}$; ii) to account for large imbalance between the sizes of the labeled subsets $\mathcal{I}^{(t)}$ and $\mathcal{S}^{(t)}$. We will modify (3.2) to incorporate *prior* knowledge about the class statuses for samples in the unlabeled subset.

Let $(Q^{(t)}_{\text{prior}}[C = 0|\underline{p}_n], Q^{(t)}_{\text{prior}}[C = 1|\underline{p}_n])$ be the prior distribution for sample $n \in \mathcal{U}^{(t)}$. A consistent measure of the discrepancy between the distribution that we choose and a given prior distribution is, again, the cross entropy, but this time written $d(P[C|\underline{p}_n||Q_{\text{prior}}[C|\underline{p}_n]) = \sum_{c=0,1} P[C = c|\underline{p}_n] \log(P[C = c|\underline{p}_n]/Q_{\text{prior}}[C = c|\underline{p}_n])$. That is, with our model's distribution in the "posterior" position in the asymmetric measure $d(\cdot||\cdot)$, and with $Q_{\text{prior}}(\cdot)$ in the position of the prior [56]. Accordingly, we propose the following semi-supervised objective function:

$$\begin{aligned} D^{(t)} &= \alpha^{(t)}_u \sum_{j\in\mathcal{U}^{(t)}} (P[1|\underline{p}_j] \log(P[1|\underline{p}_j]/Q^{(t)}_{\text{prior}}[1|j]) + P[0|\underline{p}_j] \log(P[0|\underline{p}_j]/Q^{(t)}_{\text{prior}}[0|j])) \\ &\quad - \alpha^{(t)}_i \sum_{j\in\mathcal{I}^{(t)}} \log P[0|\underline{p}_j] - \alpha^{(t)}_s \sum_{j\in\mathcal{S}^{(t)}} \log P[1|\underline{p}_j], \end{aligned} \tag{3.3}$$

where $\alpha^{(t)}_u$, $\alpha^{(t)}_i$, and $\alpha^{(t)}_s$ are the weights on the unlabeled, labeled innocuous, and labeled suspicious terms at oracle labeling step t. (3.3) is clearly a semi-supervised generalization of (3.2). Moreover, directly, we shall see that the additional, unlabeled term can be seen as a special type of " regularizer" on posterior log-likelihood maximization.

A key unresolved issue is how to choose the prior distributions $(Q^{(t)}_{\text{prior}}[C = 0|\underline{p}_n], Q^{(t)}_{\text{prior}}[C = 1|\underline{p}_n])$, $n \in \mathcal{U}^{(t)}$. At $t = 0$, before there is any oracle labeling, a sensible choice, reflect-

ing maximum uncertainty, is: $(Q^{(0)}_{\text{prior}}[C=0|\underline{p}_n], Q^{(0)}_{\text{prior}}[C=1|\underline{p}_n]) = (\frac{1}{2}, \frac{1}{2}), \forall n \in \mathcal{U}^{(0)}$. Initially ($t=0$), if there are no labeled suspicious examples, $\mathcal{S}^{(0)}$ is empty, and $D^{(0)}$ thus consists only of a sum of cross entropies for the subsets $\mathcal{I}^{(0)}$ and $\mathcal{U}^{(0)}$. We first minimize $D^{(0)}$ with respect to our class posterior's model parameters, resulting in the learned class posteriors $(P^{(0)}[0|\underline{p}_j], P^{(0)}[1|\underline{p}_j])$. We then apply our active learning sample selection criterion (defined in the sequel) to select an unlabeled sample for (oracle) labeling. The chosen sample is removed from the unlabeled subset and included in the appropriate labeled subset, resulting in the new subsets $\mathcal{U}^{(1)}$, $\mathcal{I}^{(1)}, \mathcal{S}^{(1)}$. How then to choose the new prior distributions $Q^{(1)}_{\text{prior}}(\cdot)$ for the unlabeled samples, to be used in the *next* round of learning, minimizing $D^{(1)}$ (in light of the first oracle labeling result)? Unlike in Chapter 2, where we assign fixed target distribution throughout AL iterations, we develop an adaptive target assignment approach to reflect the current best estimate for unlabeled samples in each AL iteration. *Prior* to the initial round of learning, $(\frac{1}{2}, \frac{1}{2})$ is a suitable choice, reflecting maximal uncertainty. However, *post* this initial round, the new class posterior probabilities, denoted $(P^{(0)}[0|\underline{p}_n], P^{(0)}[1|\underline{p}_n]), n \in \mathcal{U}^{(1)}$ may be less uncertain, and in fact reflect our *best current estimate* for the true distributions $(P[0|\underline{p}_n], P[1|\underline{p}_n])$. Thus, we suggest this "current best estimate" as our new prior after the initial round and, in fact, also more generally, *i.e.*, we choose $(Q^{(t)}_{\text{prior}}[0|\underline{p}_n], Q^{(t)}_{\text{prior}}[1|\underline{p}_n]) = (P^{(t-1)}[0|\underline{p}_n], P^{(t-1)}[1|\underline{p}_n]), n \in \mathcal{U}^{(t)}$. Note that if we initialize the model parameters at the current solution after round $t-1$, we in fact initially have for round t that $(P^{(t)}[0|\underline{p}_n], P^{(t)}[1|\underline{p}_n]) = (P^{(t-1)}[0|\underline{p}_n], P^{(t-1)}[1|\underline{p}_n]) = (Q^{(t)}_{\text{prior}}[0|\underline{p}_n], Q^{(t)}_{\text{prior}}[1|\underline{p}_n]), n \in \mathcal{U}^{(t)}$, *i.e.*, before any adaptation of the model parameters is performed to account for the t-th oracle labeling, the sum of cross entropies term associated with the unlabeled sample subset is *zero*. That is, in adapting the parameters to account for the newly labeled sample at round t, we will decrease the labeled subset cross entropy terms, but will in general *increase* the unlabeled subset term from zero. From this, we can understand that the unlabeled data subset term is a type of "regularizer", minimizing the "amount" of adaptation performed in light of each newly labeled sample. We also note that this regularization is very different from e.g. [20], where class uncertainty is *minimized* over the unlabeled samples. By limiting the amount of learning in light of each new labeled sample, our unlabeled term forces *retention* of some class uncertainty[6]. In our experiments, the objective function $D^{(t)}$ is minimized, after the t-th oracle labeling, via a gradient descent procedure.

[6]In the latter stages of active learning, minimizing class uncertainty on the unlabeled samples may be a good choice. However, in the early stages, a main concern should be avoiding over-training on the small number of labeled examples.

Another important question is how to choose the weights $\alpha_u^{(t)}$, $\alpha_i^{(t)}$, and $\alpha_s^{(t)}$. We propose to choose the weight $\alpha_s^{(t)}$ to amplify the contribution of the few labeled suspicious samples, *i.e.*, to "equalize" the contribution to the objective function coming from the labeled suspicious and labeled innocuous subsets. That is, we let $\alpha_i^{(t)} = 1$ and choose $\alpha_s^{(t)} = \frac{|\mathcal{I}^{(t)}|}{|\mathcal{S}^{(t)}|}$. What remains is the choice of the weight on the unlabeled samples $\alpha_u^{(t)}$. It appears more difficult to find a principled, "universal" choice for the weight $\alpha_u^{(t)}$, since this controls the amount of regularization. Instead, we suggest to set $\alpha_u^{(t)}$ using a cross-validated error rate measure, based on the labeled samples "seen until now".

3.1.5 Sample Selection Criteria for Oracle Labeling

In this work we investigate the following five active learning strategies:

Most uncertain: the sample with the greatest class uncertainty is forwarded to the oracle. Intuitively, if an unlabeled vehicle track is most uncertain for the current classifier, the removal of this uncertainty (by oracle labeling) should provide valuable information to the classifier.

Most suspicious: the sample with largest *a posteriori* probability of being suspicious is forwarded to the oracle. This strategy prioritizes detection and forwarding of suspicious samples, since the truly suspicious tracks are the ones that may require further action, and possibly in a time-sensitive fashion. Forwarding the "most suspicious" track allows achievement of confirmation that it is truly suspicious (in which case, the labeling also serves as "notification" that something suspicious has been detected). However, labeling of the track most likely to be suspicious (as estimated by the classifier) may not give the most information for further improving the classifier – it may simply "confirm what is already known".

Random sampling: randomly selecting an unlabeled sample for forwarding to the oracle. This scheme is expected to provide a "worst case" performance bound, since random sampling should only work well if all samples' labels are equally informative to the classifier, which should not be valid for most realistic domains/data sets.

Largest expected decision change: selecting the sample whose labeling, in expectation, results in the most change (in aggregate) in the classifier's posterior probabilities over the unlabeled samples:

$$\begin{aligned} x^* = argmax_x P(S|x) * \sum_{x' \in U} |P(S|x') - P(S|x', L(x) = s)| \\ +(1 - P(S|x)) * \sum_{x' \in U} |P(S|x') - P(S|x', L(x) = I)| \end{aligned} \tag{3.4}$$

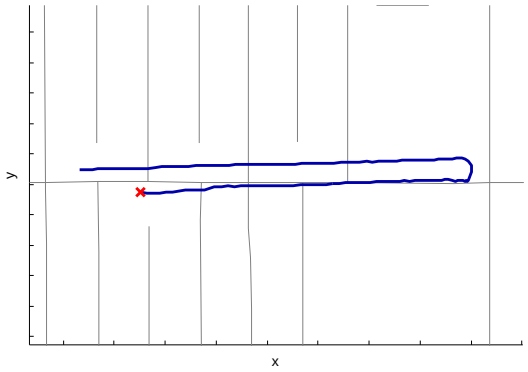


Figure 3.2: A suspicious track making a U-turn.

Largest expected improvement: recently in [27], the authors proposed a sample selection strategy which forwards the sample with the greatest estimated "expected improvement in classification error rate" for the current classifier (as measured over both the labeled and unlabeled samples).

The first three methods have relatively low computational complexity, while the latter two require substantial computation to trial re-train the classifier $2|\mathcal{X}_u|$ times (under each of two possible labelings, for all samples in the unlabeled batch). One can also expect that the "most suspicious" criterion may have the greatest propensity for choosing truly suspicious tracks. This will be investigated in the sequel.

3.1.6 Experimental Results

The road network used to capture vehicle tracks consists of several busy intersections, highways and some suburban areas where vehicle tracks are scarce. Figures 3.2 and 3.3 show some examples of tracks that are subjectively defined as suspicious (one with a U-turn and another with loops) by our human operator, where the cross point indicates the starting position. Grey lines are different roads in the scene. By using these u-turn/looping features by themselves, several suspicious tracks can be identified. But we include these features inside our active learning framework to let the operator decide if such tracks are indeed truly suspicious.

We separated the tracks into two independent batches, each with the same proportion of suspicious to innocuous tracks. One batch was used both for null modeling and active learning (but with innocuous examples that were used for null modeling segregated from those used for active learning), and the other for test set performance evaluation. We then switched the roles of the two batches and measured the average test set performance.

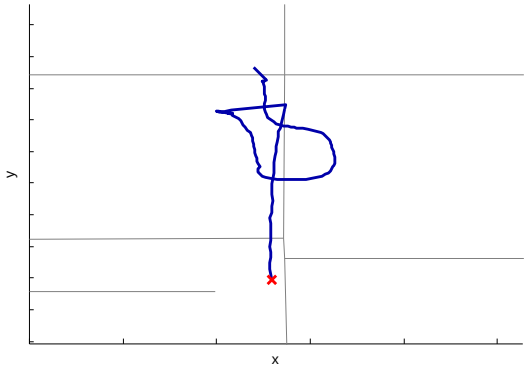


Figure 3.3: A suspicious track with a loop.

We chose $K = 10$. If a track has less than 10 measurements, its remaining p-values (to reach $K = 10$) were set to zero.

3.1.6.1 Performance Metrics

There are two fundamental performance measures of interest on the test batch: i) true detection rate – the proportion of truly suspicious tracks that have been correctly classified as suspicious and ii) false alarm rate – the proportion of innocuous tracks classified as suspicious. We evaluate these measures for our "current" classifier, after every iteration of active learning. As a comprehensive performance measure encompassing true detections and false positives, we measure the test set area under the ROC curve (ROC AUC), as a function of the number of active labelings. On the training batch, we also measure "true detections", *i.e.*, the number of truly suspicious tracks forwarded to the oracle, as a function of the number of active labelings. This is not really a performance measure – it simply indicates, for a given sample selection method, how many truly suspicious examples there are amongst the samples found to be "most informative" for refining the classifier.

3.1.6.2 Compare Different Alphas

In this section, we compare different choices of α_u on unlabeled samples during active labeling. Shown in Figure 3.4, we plot ROC AUC as a function of oracle labelings for different choices of α_u. We also plot the weight magnitudes after one hundred labelings in Figure 3.5. We observe that different α_u values give very different performance results and the associated weight magnitudes. A smaller α_u, which represents less amount of

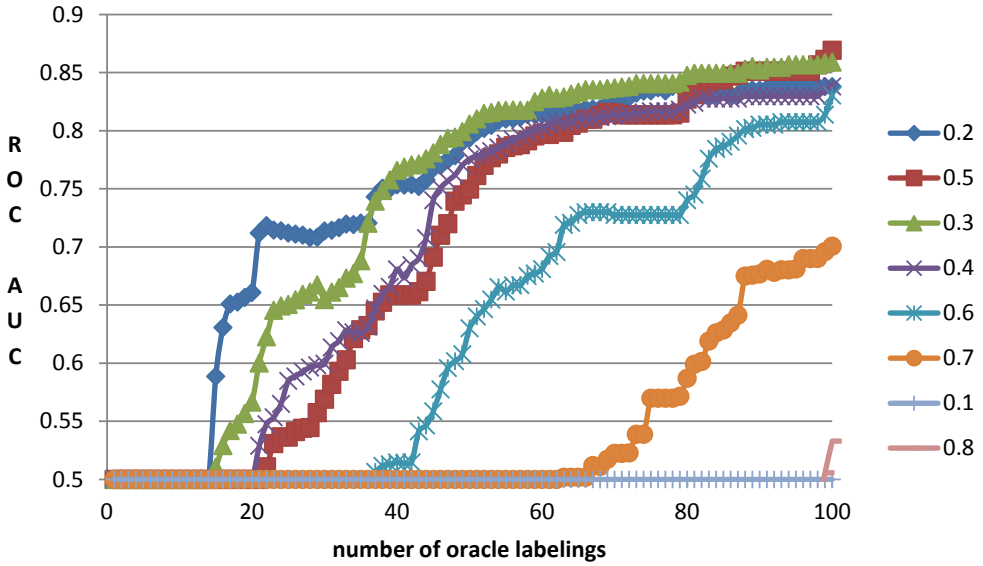


Figure 3.4: ROC AUC for different alphas.

regularization, tends to produce larger weight vector magnitude. The generalization performance is best when the weight magnitude is neither too large nor too small (*i.e.*, for α_u in the range from 0.2 to 0.6; otherwise there is very poor detection capability, as observed e.g. in the case of $\alpha_u = 0.1$ and 0.8;

3.1.6.3 Compare Sample Selection Strategies

In Figure 3.6 and Figure 3.7, we show the ROC AUC and training set "true detection" performances of the "most uncertain" and "most suspicious" strategies, when used with our classifier and semi-supervised learning framework. The two strategies provide very different results, with "most suspicious" choosing many more "informative examples" that are in fact truly suspicious. However, these labeled samples (including many truly suspicious samples) provide weak gains in classifier generalization, *i.e.*, in ROC AUC shown in Figure 3.7. Presumably, the reason is that the tracks being forwarded are very similar in their suspicious signatures – *i.e.*, the classifier does not learn much from the labeling of these samples. Uncertainty sampling, on the other hand, is seen to forward for labeling *very few* truly suspicious tracks – only 4 out of 50 – but its generalization performance is much better. Apparently, the samples selected by uncertainty sampling, being closer to the current decision boundary than the "most suspicious", are better able to help refine the boundary. Figure 3.7 also shows the performance of *random* selection (an effective performance lower bound), and that based on learning a classifier using the

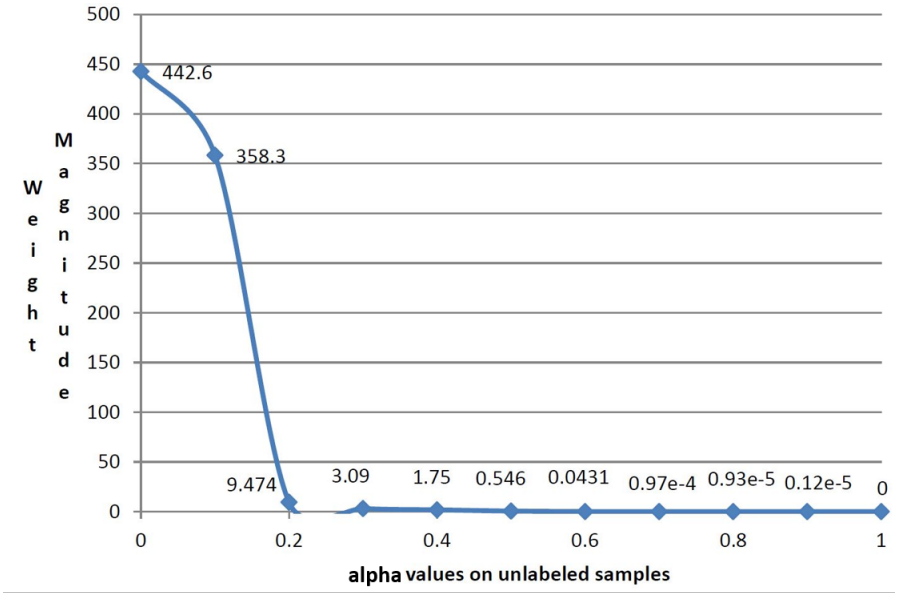


Figure 3.5: Weight magnitude for different alphas.

entire active learning batch as a labeled training set (which should upper bound active learning performance).

These results show that forwarding the "most uncertain" sample greatly outperforms forwarding the "most suspicious" sample. However, as we noted earlier, from an operational (actionable) standpoint, it is also clearly important to prioritize *confirmation* of the detection of suspicious tracks. This suggests the need for two (concurrent) operational labeling "streams" in a suspicious/innocuous active learning setting. One is the most actionable tracks: these are the unlabeled tracks estimated to be most suspicious, whose labeling may lead to subsequent action. The other is the most informative track(s): these are the tracks whose labeling will help to improve a suspicious/innocuous classifier the most. The "most uncertain" track is one reasonable candidate for the "most informative" track, evaluated here. However, alternative, more sophisticated criteria such as [27] can also be investigated, which we suggest for future work.

3.1.6.4 Evaluation of Semi-supervised Learning

In this section we investigate i) the value of unlabeled tracks in building up the classifier and ii) the proper choice of weights on the labeled subsets in our cross entropy-based learning.

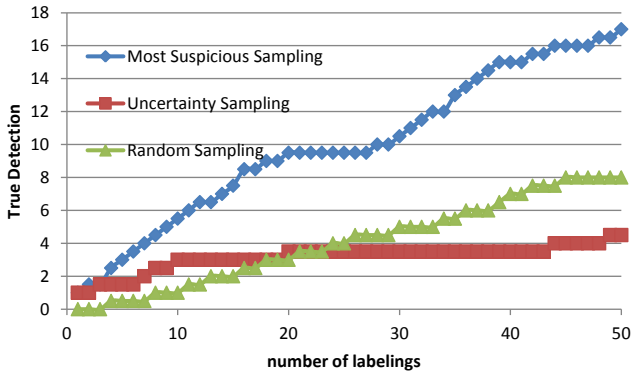


Figure 3.6: Number of true detections for different sample selection mechanisms.

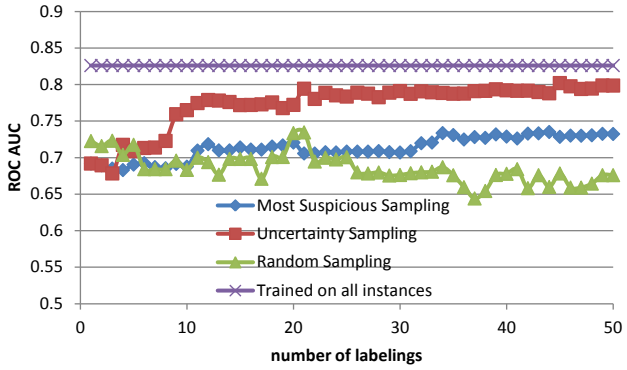


Figure 3.7: ROC AUC performance for different sample selection mechanisms.

3.1.6.5 Value of unlabeled tracks

To assess the value of the unlabeled tracks, we investigated excluding them (excluding the unlabeled subset) from our learning objective function. In this case, the learning is strictly supervised, and maximizes the class posterior log-likelihood. Moreover, since we start without any labeled suspicious tracks, initially, this amounts to maximizing the joint log-likelihood of the labeled innocuous samples.

In Figure 3.8, it is seen that inclusion of the unlabeled tracks within the classifier learning is essential for building up the classifier's accuracy – without the unlabeled term, uncertainty sampling is unable to detect any suspicious examples, and thus there is no ability to identify the suspicious class gleaned from active learning[7]. Also shown

[7]This is not surprising, since from the outset of the active learning, the class posterior is *solely* being fit to agree with the innocuous samples. With no unlabeled samples included in the learning, this situation never changes over the active learning iterations.

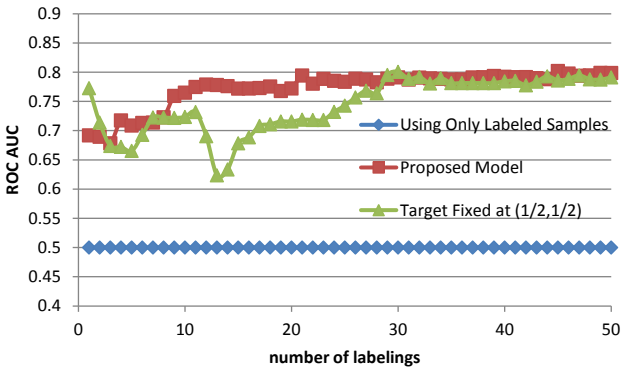


Figure 3.8: ROC AUC performance with/without use of unlabeled samples and with $(\frac{1}{2}, \frac{1}{2})$ unlabeled targets.

in Figure 3.8 is the performance of a classifier that uses our semi-supervised learning, but with the target distribution $(\frac{1}{2}, \frac{1}{2})$ for unlabeled samples, for all active learning iterations (t). While in the long run similar ROC AUC performances are achieved, use of our proposed unlabeled "regularizer" converges much more quickly, with far fewer labelings, to this "long run" performance than when the $(\frac{1}{2}, \frac{1}{2})$ targets are used. This in part validates our particular choice of unlabeled term "regularizer".

3.1.6.6 Weights on The Labeled Subsets

In Figure 3.9, we demonstrate the benefit of our proposed weighting scheme (which amplifies the weight on the suspicious subset, relative to the innocuous subset) by comparing with a scheme wherein every sample is given equal weight. As seen in the figure, ROC AUC is greatly improved by our unequal weighting scheme. Without the use of such weighting, the objective function will be biased towards the majority class, which is seen in the figure to hurt generalization performance.

3.1.6.7 Comparison with Support Vector Machines (SVMs)

Here we compared the performance of our method with that of active learning of SVMs [62]. Both methods use our proposed p-value features and uncertainty sampling. Since the SVM requires at least one sample from each class in order to learn, we "seed" SVM learning using the *one-class* SVM approach [53]. Once the one-class SVM (which selects the most outlying sample) is first successful in identifying a suspicious sample, we switch from one-class SVM to standard (two-class) SVM classifier learning. While the one-class SVM framework requires use of a nonlinear kernel, the two-class SVM could be applied

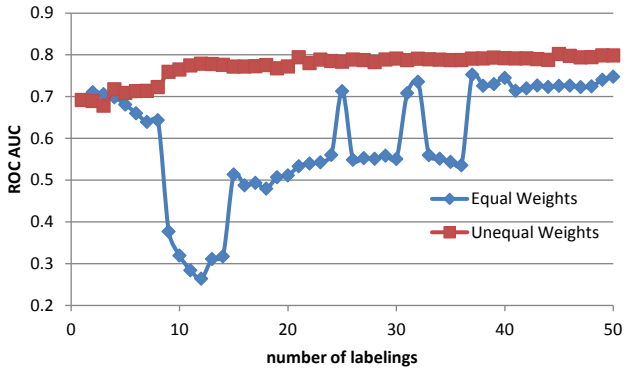


Figure 3.9: ROC AUC for different labeled subset weighting schemes.

with either a linear or a nonlinear kernel. We have found there is some improvement from using a nonlinear (Gaussian) kernel. Thus, we designed Gaussian kernel SVMs (for both the one and two-class stages). Hyperparameter selection for the SVM (choosing the slackness and Gaussian scale hyperparameters) was performed using a grid search, based on 10-fold cross validation error performance, evaluated on the current labeled training subset. As seen in Figure 3.10, the SVM, trained in a purely supervised fashion, initially has poor accuracy. But as it learns more and more samples close to the decision boundary, its performance boosts in the latter iterations, ultimately surpassing the ROC AUC of our model. We also show results for a linear kernel SVM, with similar representation power (linear feature dependence) as our logistic regression classifier. Note that the linear SVM's performance only approaches our model's performance in the latter iterations. Finally, we note that, in these results, for our method, we have not optimized the weight that we apply on the unlabeled sample subset – the weight on unlabeled samples was always set to one. Optimizing this hyperparameter, e.g., via cross validation, as well as using a non-linear kernel and consideration of alternative sample selection strategies could further enhance our method's performance.

3.1.7 Conclusion

In this section, we applied the adaptive semi-supervised active learning framework, along with a class posterior model, for learning to distinguish "suspicious" from "innocuous" anomalies (as well as from samples that are *not* anomalous). Our class posterior is chosen consistent with the inductive bias that "suspicious" is a subset of "anomalous". Our semi-supervised learning approach exploits unlabeled samples in an unconventional way – to control the amount of learning that is performed given few labeled samples. That is,

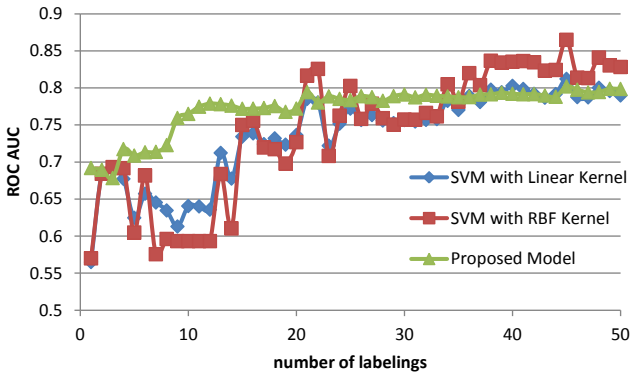


Figure 3.10: ROC AUC performance of our semi-supervised method, compared with linear and Gaussian kernel SVMs.

unlabeled samples are used to form a novel regularizer, which we suggest may be suitable for active learning in general. For the domain of detecting suspicious vehicle tracks, our approach (using a simple logistic regression classifier) performs quite favorably compared with a standard SVM-based active learning system. Our framework can be understood as learning a mapping between a human operator's concept of what is suspicious (based possibly on visual inputs) and an information-rich, albeit possibly very *low-level* high-dimensional feature space. The classifier achieves this mapping, and it also identifies which (possibly low-level) features capture the operator's concept.

3.2 Flow Based Botnet Detection through Semi-supervised Active Learning

Detecting botnet communication presents a major challenge for current IDSes. Coordinated bot malware is used to carry out malicious activities such as DDoS, spamming, and phishing, with huge cost passed on to the victims. One recent survey claimed that around 16% of host computers connected to the Internet today are compromised, becoming either active or passive bots, waiting to follow the bot master's commands. Neutralization of botnet activities involves technological, social, and political efforts, but successfully detecting botnets is usually a prerequisite for all other neutralization efforts. One of the most difficult challenges associated with detecting botnet communication is that both bot masters and slaves constantly modify their behavior to evade popular signature-based IDSes. For example, in order to avoid deep packet inspection based IDSes, botnet applications use secure transmission protocol (STP) to encrypt their command and control

(C&C) messages. Botnets also use fast flux to randomize both their port numbers and/or domain names, thus avoiding anomaly detectors that are based on usage of certain port numbers or domain names, such as [21]. To avoid timing based signatures, botnets try to randomly delay their transmissions or make their traffic round-trip-times (RTTs) similar to perceived "normal" sessions subject to very low implementation complexity. On the defender's end, according to [23], most current flow-based IDSes rely on payload information in the TCP packet, which is defeated by encryption. To avoid packet-size based features, a botnet may pad segment and/or pad its packets. Obviously, an *adaptive* IDS is required to quickly respond to the ever changing tactics of distributed botnets, which are sometimes driven by contextual information not immediately available to IDSes.

We propose an AL framework to adapt to unknown botnet behavior using only bidirectional packet sizes of a given flow[8] to derive application-discriminating features. To the best of our knowledge, this is the first work to apply AL to the domain of botnet detection. In our AL approach, the network security administrator is leveraged as an "oracle" to inform the IDS which AL-selected network flows are suspicious (botnets) or from one of the attack class, and which are known normal (e.g., web). Such decisions may be based on payload patterns, similarity to previously detected botnet flows, information from honeynets, *etc.* Since it is *a priori* unknown which feature anomalies are bot-indicative and which are merely spurious (leading to detection of *innocuous* outlier flows), we use machine learning and AL to learn the (potentially sparse) *informative* feature subset, starting from no ground truth about the botnet traffic, which may or in fact *may not* be present in a current captured batch of (unlabelled) flows. Using packet size and direction feature sets from available anonymized public datasets, we show our proposed flow-based feature representation leads to better detection performance than that of [33, 5] as well as that of [49]. Also unlike [49], we separate rare from "unknown-unknown" anomalies in this work while preserving the convexity of the objective of our optimization. This is an improvement over the multiclass, hierarchical approach developed in Chapter 2.

The rest of the chapter is organized as follows: We define our problem and present related works in Section 3.2.1. We then present our methodology in Section 3.2.2, followed by experimental results in Section 3.2.3. In Section 3.2.4, we conclude with a discussion, including future work and mention of suitable enterprise NIDS implementation platforms.

[8]Here, a packet flow is a bidirectional communication session between a pair of end-hosts.

3.2.1 Problem Definition and Related Work

Assume there is a batch of normal labeled web traffic available at the outset for training a null (normal) hypothesis model, *i.e.*, $\mathcal{X}_l = \{\underline{x}_i \in \mathbb{R}^D, i = 1, \ldots, T_l\}$, where $\underline{x}_i$ is a feature vector representing the i^{th} training (web) traffic-flow sample, and where we assume the number of training flows T_l is large enough to learn an accurate null model. The feature vector is based on the (possibly truncated, e.g., first ten packets) sequence of packet sizes in the bidirectional flow. We assume, as a reference, that the packets are alternating client-to-server and server-to-client, starting with a client-to-server packet. Thus, for a given flow, if there are two consecutive packets in the same direction, this is represented by inserting a zero between these two packet sizes. Illustrative examples are shown in Table 3.1. In this way, our feature vector preserves information about *both* the sequence of packet sizes and the sequence of packet directions. These training traffic flows can either be captured in a sandbox environment or sampled from a domain of interest (data warehouse, enterprise network) in real time under normal operating conditions – the LBNL data set [31] was captured in the latter way.

During real-time operation, an unlabeled flow-batch $\mathcal{X}_u$ is captured, with $T_u = |\mathcal{X}_u|$, which may or may not contain any suspicious botnet traffic masquerading as web. Applying pool-based AL [55], the learning algorithm will sequentially select samples to be labeled, aiming at each step to choose the most "informative" sample for labeling [9]. Based on some context information, the network administrator will ground-truth label a selected flow $\in \mathcal{X}_u$ as web or botnet; the web *vs* botnet classifier's parameters will then be adapted and improved in light of this new labeled example. Thereafter, another informative sample may be selected for labeling and the process is repeated. A flow chart of the overall learning system is shown in Figure 3.11.

An AL-based anomaly detection system is fully determined by: 1) the choice of the classifier's features; 2) the classification model (e.g., linear classifier, nearest neighbor classifier, decision tree); 3) the associated objective function optimized in learning the classifier's model parameters; 4) the optimization technique; and 5) the criterion used for determining the next sample to be oracle-labeled. The ultimate goal is to mine the current unlabeled flow-batch $\mathcal{X}_u$ such that, with as few labelings as possible, the active

[9] There are different possible definitions of "most informative". This could be the sample whose labeling is predicted to improve classification (bot vs. web) accuracy the most. Alternatively, it could be the sample estimated most likely to be an example of an unknown class (zero day threat) – this choice could lead to the most rapid discovery of unknown classes (zero day threats). It could even be the sample most likely to be a known bot (one that has been seen and labeled before) – however, while this flow may be the most *actionable* one by the system, labeling this sample may give less benefit in terms of improving the classifier's generalization accuracy, compared with other sample selection strategies.

learner discovers whether there are any suspicious anomalies (unknown bots) in $\mathcal{X}_u$ and, if there are, learns how to discriminate these suspicious anomalous flows from innocuous anomalies (mere outlier flows) from existing bot classes, and from normal web traffic. Moreover, the learning may identify a *sparse* set of informative, class-discriminating features, which aid interpretability of the classifier solution and which may thus assist the administrator in characterizing/identifying zero-day threats and their low-dimensional "signatures".

Machine-learning based botnet detection has been proposed in numerous previous works, with most of them focusing on the supervised learning scenario, assuming that ground truth examples of the botnet traffic are available at the outset [59, 57, 5]. Although the true detection rate can go nearly to 100% in these works, the assumption that we have ground truth examples for (possibly *unknown*) botnets limits the wide applicability of these IDSes. While some previous works address unsupervised learning with initially unknown botnet traffic [28] [48] [39], these approaches are inherently performance-limited, since supervised learning is not leveraged to label the anomalous samples (or clusters) detected by these approaches. Such ground-truthing could be used to refine the detector. For example, the AL IDS approach in [39] actively updates the one-class IDS to detect attacks that degrade of the network Quality of Service (QoS), but true anomalous traffic are not used when making a decision.

In this work, we try to address both of the above-mentioned limitations, assuming labeled botnet flows are *initially* unavailable and only become available within the AL framework when they are selected for labeling and then ground-truthed by the human oracle. These labeled examples are used under the (sparingly supervised) AL framework to learn and adapt a classifier that aims to discriminate the botnet from normal traffic (as well as from mere outlier flows). Moreover, our approach is *semi-supervised*, *i.e.*, it exploits for classifier learning not only the existing cadre of labeled samples, but also all captured *unlabeled* samples. We apply the very effective semi-supervised learning approach introduced in [49] and Chapter 2. We demonstrate the performance advantage of our proposed model over both unsupervised and standard supervised approaches in Section 3.2.3.

AL, with its judicious creation of a labeled pool of samples from the current unlabelled data-batch $\mathcal{X}_u$, has been shown to effectively reduce the number of labeled samples required for learning a classifier with a given level of generalization accuracy, compared to standard supervised learning (where the labeled training set is simply given up front, and may contain redundant, uninformative samples) [55]. The best-performing system

(of which we are aware) for AL with rare (and/or unknown) categories is [49], which consisted of: i) semi-supervised posterior log-likelihood maximization with *maximum entropy* unlabeled sample regularization as the objective function; ii) p-values (reflecting degree of atypicality relative to the null model) as derived features; and iii) a variant of a logistic regression classifier structure wherein the weights applied to negative log p-values were constrained to be non-negative, consistent with the inductive bias that the "suspicious" category is a subset of what is *anomalous*. It has successfully been applied in the domain of vehicle behavioral tracking, identifying suspicious or unknown track behaviors [51]. With sparing use of active labeling, this model generally converges within a few active labelings, and it has been demonstrated to perform better than other rare-category based anomaly detectors such as [27] and [26]. Again, we will use the same learning framework as proposed in [49], but propose a novel feature representation for botnet detection and separate rare from "unknown-unknown" anomalies to obtain a convex optimization objective.

3.2.2 Methodology

Obviously, in practice, NIDS may attempt to employ raw or derived features from packet payloads or timing (or other packet-flow attributes besides the packet size and direction sequence of a flow). Our primary aim herein is to demonstrate the efficacy of our proposed active learning approach so that a NIDS can adapt to new behavioral anomalies indicative of *new or evolving* threats. To this end, we make the best use of publicly available background and attack datasets.

To the best of our knowledge, botnet C&C activity is not present in the public LBNL background traces [31], particularly the public traces of the types of botnets used in the experiments reported herein. The LBNL traces were anonymized: payload-clipped and randomized IP addresses.

Timing-based features may lead to detection based on differences in the domains wherein the background (LBNL) and botnet traffic were recorded, *i.e.*, attack-salting artifacts [5]. As we are primarily interested in detecting botnet C&C activity masquerading as web (port-80) traffic, we did not employ port-number features of the packet flows [65].

So, to demonstrate the efficacy of our AL framework, we settled on the use of a flow feature-set consisting just of its packet size (taken as continuous numerical) and direction (categorical) sequence (*i.e.*, mixed feature types). This said, for purposes of comparison, we did evaluations using different feature sets, including with timing information, cf. Section 3.2.3.2.

Table 3.1: Illustrative examples of bidirectional flow's feature representation for the first N transmitted packets. $N = 6, D = 12$. C_i denotes the i^{th} packet from client to server, whereas S_i denotes the i^{th} packet from server to client. The three-way hand shake packets are not used.

Packet Sequence	**Feature Representation** $\underline{x} \in \mathbb{R}^D$											
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}
$C_1S_1C_2S_2C_3S_3$	C_1	S_1	C_2	S_2	C_3	S_3	0	0	0	0	0	0
$C_1C_2S_1S_2C_3C_4$	C_1	0	C_2	S_1	0	S_2	C_3	0	C_4	0	0	0
$S_1S_2S_3S_4S_5S_6$	0	S_1	0	S_2	0	S_3	0	S_4	0	S_5	0	S_6

3.2.2.1 Feature-Space Representation

In the command-and-control (C&C) phase[10], most botnet traffic involves master(s) periodically issuing C&C messages, whereas the slaves execute the given commands. Normal/background web traffic, on the other hand, tends to principally involve server-to-client communication. In the attack phase, botnet malware carries out malicious activity, periodically sending out beacon signals to the bot master via the previously established C&C channel, most of the time without packet transmissions from the bot master to bot slaves.

Hence, we specifically seek to preserve the bidirectional packet size sequence information as feature representation for different traffic flows. This feature representation was previously considered in [28, 48]. The authors used the first N (we set $N = 10$ in our experiments) packets after the three-way hand shake of each TCP flow. Then a feature vector of dimension $D = 2N$ was defined, specified by the sizes and directionalities of these N packets. Traffic is assumed to be alternating between client-to-server (CS) and server-to-client (SC). A zero packet size is thus inserted between two consecutive packets in the same direction to indicate an absence of a packet in the other direction. For example, if the bidirectional traffic is strictly SC, a zero will be inserted after each SC packet size. We demonstrate our flow feature representation using several examples in Table 3.1, with $N = 6$ for illustrative purposes. This $2N$-dimensional packet-size feature representation preserves bidirectional information of a given TCP flow, which is essential for discriminating between botnet and normal web traffic, as will be seen by our results. We use this feature representation as our "raw" features.

[10] *i.e.*, communication between Bot masters and slaves for attack coordination.

3.2.2.2 Anomaly Based Derived Features

As we discussed previously, both the presence of packets in given directions and the sizes of packets should be informative in identifying botnet traffic. We accordingly define a set of anomalous scores to quantify such. Considering the previously defined D-dimensional feature vector $\underline{x} = (x_1, x_2, \ldots, x_D)^T$, we use $\mathcal{I}(\underline{x}) = (I(x_1), I(x_2), \ldots, I(x_D))^T$, with $I(x) = 1$ if $x > 0$ and 0 otherwise, a binary vector, to specify the packet direction sequence. To reduce the number of parameters to model the joint distribution for $\mathcal{I}(\underline{x})$, we propose to model $\mathcal{I}(\underline{x})$ based on the Chow-Liu Bayesian Network variant [9] (An alternative is to treat all pairwise probabilities as the basis for derived features). Here, the joint distribution for a vector of discrete-valued random variables is the model which maximizes the likelihood over the training data under the constraint that the distribution factors as a product of first and second-order probabilities. Hence, based on this special Bayesian Network structure, $P[\mathcal{I}(\underline{x})]$ factorizes as:

$$P[\mathcal{I}(\underline{x})] \quad = \quad P[I(x_{j_1})]P[I(x_{j_2})|I(x_{j_1})] \ldots P[I(x_{j_D})|I(x_{j_{D-1}})],$$

where j_1 denotes the root node index of the learned Bayesian Network. To simplify notation in the sequel, we will use I_j to denote $I(x_j)$.

The maximum likelihood estimates of the probabilities are obtained from frequency counts. For all estimates, we added 1 in the numerator to avoid assigning 0 probabilities. That is, $P[I_j = 1] = \frac{N_j^+ + 1}{T_l + 2}$, with N_j^+ representing the number of web-flows belonging to $\mathcal{X}_l$ with positive packet size in the j^{th} position. Similarly, $P[I_j|I_m] = \frac{P[I_j, I_m]}{P[I_m]}$, with $P[I_j = 1, I_m = 1] = \frac{N_{jm}^{++} + 1}{T_l + 4}$ and N_{jm}^{++} representing the number of training web flows that have positive packet size in the $\{j, m\}$ position pair. Similarly, $P[I_j = 0, I_m = 1] = \frac{N_{jm}^{0+} + 1}{T_l + 4}$, $P[I_j = 1, I_m = 0] = \frac{N_{jm}^{+0} + 1}{T_l + 4}$, and $P[I_j = 0, I_m = 0] = \frac{N_{jm}^{00} + 1}{T_l + 4}$. Note that $P[\mathcal{I}(\underline{x})]$ is a product of D *unweighted* probabilities, quantifying the *unweighted* total anomaly score over the D dimensions, for the packet direction sequence. We will exploit the low-order *constituent* probabilities of $P[\mathcal{I}(\underline{x})]$ as the basis for derived features input to the classifier.

Next, for all single features and all pairs of features, considering only the *positive* entries (non-zero packet sizes), we propose to model these continuous distributions using Gaussian Mixture Models (GMMs) [11], and use both first and second order *mixture*

[11] We separately model each single feature and each pair of features by GMMs (while ignoring higher-order feature sub-collections) to avoid the curse of dimensionality in learning the parameters, while at least capturing dependencies between all pairs of features. This also avoids intractable marginalizations of the joint pdf on the full feature space, e.g., when considering evaluation of p-values involving the size

p-values [49] to quantify the flow anomalies by these independently learned GMMs. Following the intuition and development behind [49], as also detailed in Chapter 2, consider any pair of features $\underline{Y} = (X_i, X_j),\ 1 \leq i \neq j \leq D$, modeled by a bivariate GMM under the null. Let $\{\alpha_k, k = 1, \ldots, K_{ij}\},\ 0 \leq \alpha_k \leq 1,\ \sum_{k=1}^{K_{ij}} \alpha_k = 1$, be the prior probabilities for the (K_{ij}) mixture components, with associated component densities $\{f_{\underline{Y}|k}(\underline{y}|\theta_k), k = 1, \ldots, K_{ij}\}$, and $\theta_k = (\underline{\mu}_k, \Sigma_k)$ the (mean vector, covariance matrix) parameter set for the k^{th} density. The mixture density is thus $f_{\underline{Y}}(\underline{y}) = \sum_{k=1}^{K_{ij}} \alpha_k f_{\underline{Y}|k}(\underline{y}|\theta_k)$. Given such a mixture null, the p-value – the probability that a two-dimensional feature vector will be more extreme than the given observed vector $\underline{y} = (x_i, x_j)$ – is $p_{ij}^{+}(\underline{y}) = \sum_{k=1}^{K_{ij}} P[M = k|\underline{y}] e^{-r_k^2(\underline{y})/2}$. Here, the mixture posterior is $P[M = k|\underline{y}] = \frac{\alpha_k f_{\underline{Y}|k}(\underline{y}|\theta_k)}{\sum_{m=1}^{K_{ij}} \alpha_m f_{\underline{Y}|m}(\underline{y}|\theta_m)}$ and $r_k^2(\underline{y})$ is the squared Mahalanobis distance between $\underline{y}$ and $\underline{\mu}_k$ for the k^{th} GM component.

Note that $p_{ij}^{+}(\underline{y})$ is the expected p-value, with the expectation taken with respect to the mixture posterior pmf. In a similar fashion, one can also calculate a set of mixture-based p-values for single (univariate) features, denoted $\{p_i^{+}(x_i), i = 1 \ldots, D\}$.In this case, complementary error functions are used to measure the p-value conditioned on each mixture component, with the mixture-based p-value again the expected p-value. Based on the Bayesian Network and the collection of GMM null distributions, we can compute the vector of derived p-value based features for each flow from the raw features $(\underline{x}, \mathcal{I}(\underline{x}))$.

Let's define $p_i(x_i)$ and $p_{ij}(\underline{y})$ the following way:

$$p_i(x_i) = \begin{cases} p_i^{+}(x_i) & I_i = 1 \\ 1 & else \end{cases},\ p_{ij}(\underline{y}) = \begin{cases} p_{ij}^{+}(\underline{y}) & I_i = 1, I_j = 1 \\ 1 & else \end{cases}.$$

We then have the derived feature vector $\underline{z}$,

$$\begin{aligned} \underline{z} \ = \ & (P[I_{j_1}], P[I_{j_k}|I_{j_{k-1}}], p_i(x_i), p_{ij}(\underline{y}) \ : \\ & \forall i, j, k,\ 1 < k \leq D,\ 1 \leq i < j \leq D) \ \in \ (0,1]^{2D + \binom{D}{2}}. \end{aligned}$$

Note that this is a fixed-dimensional feature representation for every flow. Note also that we will take the log of the entries in $\underline{z}$, which means that p-values set to 1 have log p-values that are zero – taking the log thus effectively eliminates the effect of such

of packet j jointly with the binary presence/absence value for packet m.

features.

Finally, the derived feature set can be *augmented* to include the *conditional* p-value for packet size x_i given the direction bit for packet j, I_j, $\forall i, j$. This approach will introduce an additional $2D^2$ p-value features. During our experiments, however, we found these augmented features have little value in classifying the botnet applications we considered in this work, *i.e.*, nearly all the weights associated with the conditional p-value features were found to be 0. This prompts us not to use the conditional p-value based features for botnet detection.

3.2.2.3 Classification Model and Learning Objective

The classifier model is identical to that of developed in Chapter 2, which uses logs of the entries of the derived p-value based feature vector $\underline{z}$ and with *non-negative* constraints on the weights on these features. For the c^{th} class, let

$$f(\underline{z}; \underline{\beta}^{(c)}) = \exp(\beta_0^{(c)} - \sum_{i=1}^{2D+\binom{D}{2}} \beta_i^{(c)} \log z(i)),$$

where the model parameters for the c^{th} class are $\{\beta_i^{(c)}, i = 0, \ldots, 2D + \binom{D}{2}\}$. Using ω_2, ω_1, and ω_0 to respectively denote the known botnet, the known normal, and the unknown class (which represents the union of all "unknown unknown" classes, *i.e.*, those that have not yet been discovered or labeled), we then have:

$$P(\Omega = \omega_c|\underline{z}) = \frac{f(\underline{z}; \underline{\beta}^{(c)})}{\sum_{c'} f(\underline{z}; \underline{\beta}^{(c')})} \quad \text{with } \beta_i^{(c)} \geq 0, \forall i > 0, c = 0, 1, 2. \tag{3.5}$$

The inclusion of ω_0 allows for the possibility that there are *unknown* classes in a test data batch, beyond a botnet class (ω_2) that has already been discovered (for which there are labeled flow examples). Note also that (3.5) is for the case of one normal class and one known botnet class. This can of course be generalized if there are *multiple* known botnet classes. Moreover, initially in our scenario, there are *no* known botnet classes, *i.e.*, ω_2 is only instantiated once a sample from a botnet class is selected and actively labeled.

The non-negatively constrained logistic regression model has been shown to produce a highly *sparse* solution, in which only "informative features" have non-zero weights (See Chapter 2).

Let us assume at the t^{th} oracle labeling we have a set of labeled samples $\mathcal{Z}_l^{(t)} \in \mathbb{R}^{T_l^{(t)} \times (2D + \binom{D}{2})}$, with associated labels $C_l^{(t)}$ and unlabeled samples $\mathcal{Z}_u^{(t)} \in \mathbb{R}^{T_u^{(t)} \times (2D + \binom{D}{2})}$, with no ground truth. Using Q to denote the uniform distribution on $\{\omega_0, \omega_1, \omega_2\}$ if a sample from class ω_2 has already been labeled, and to denote the uniform distribution $\{\omega_0, \omega_1\}$ otherwise, *i.e.*,

$$\begin{cases} Q = \{q_{\omega_0}, q_{\omega_1}, q_{\omega_2}\} = \{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\} & if\ \omega_2 \in C_l^{(t)} \\ Q = \{q_{\omega_0}, q_{\omega_1}\} = \{\frac{1}{2}, \frac{1}{2}\} & otherwise. \end{cases} \tag{3.6}$$

Our AL-based, semi-supervised, regularized negative posterior log-likelihood learning objective, where we use novel *maximum entropy* regularization on the unlabeled sample subset, is:

$$\begin{aligned} \mathcal{J}_{\text{maxEnt}}^{(t)} &= -\sum_{(\underline{z},c) \in (\mathcal{Z}_l^{(t)}, C_l^{(t)})} \alpha_c^{(t)} \log P[\Omega = \omega_c | \underline{z}] \\ &\quad + \gamma \sum_{\underline{z} \in \mathcal{Z}_u^{(t)}} d(Q || P[\Omega | \underline{z}]). \end{aligned} \tag{3.7}$$

Here, in minimizing (3.7), we aim to maximize the class posterior log-likelihood on the labeled samples, but *also* to maximize the class *uncertainty* of the posterior on the unlabeled samples, where $d(Q||P) = \sum_c q_c log(q_c/p_c)$, the Kullback-Leibler distance [30] (cross entropy) between probability mass functions $Q = \{q_c\}$ and $P = \{p_c\}$. Compared to the previously proposed *minimum entropy* regularization approach [20], which minimizes decision uncertainty on unlabeled samples, (3.7) avoids over-training, especially when the rare category (botnet) is underrepresented, *i.e.*, during the early stages of AL.

(3.7) is a *convex* objective function [12], with a unique global minimum, unlike [20]. In [49], maximum entropy regularization was demonstrated to greatly outperform minimum entropy regularization, especially during the early phase of active learning, *i.e.*, either before an unknown class has been discovered, or when only a small subset of its samples have been labeled. During this phase, minimum entropy regularization may in fact *fail* to discover the unknown class, even after many oracle labelings.

We optimize (3.7) via projected gradient descent, which is guaranteed to reach the global minimum due to the objective's convexity. $\alpha_c^{(t)}$ is chosen to balance the effective sample size between the two classes, whereas γ is chosen via cross validation (CV). When

[12] Unlike the developed model in Chapter 2, we separate rare from unknown-unknown in this work primarily to preserve convexity.

there is not enough botnet traffic for CV, γ is set to $\frac{T_l^{(t)}}{T_u^{(t)}}$.

3.2.2.4 Active Learning Strategy

We use pool-based AL, wherein the oracle ground-truth labels the informative samples sequentially forwarded by the learner. There are several AL sample selection strategies we investigate in this work, including one of the best AL strategies proposed in [49]:

- *uncertainty sampling*: pick the unlabeled sample with the highest entropy, as measured by the current classifier's posterior (3.5).
- *most likely unknown (MLU) sampling*: pick the unlabeled sample that has the highest probability of belonging to the unknown class, as evaluated by $P(\Omega = \omega_0|\underline{z})$.
- *random sampling*: pick the unlabeled sample randomly.

Other sample selection criteria are also possible, e.g., we can sample traffic with the highest posterior for belonging to a known suspicious category. Moreover, we could use a (randomized) mixed strategy to simultaneously address multiple learning and inference objectives.

3.2.3 Experimental Setup and Results

The overall AL system is illustrated in Figure 3.11. Again, we obtained normal web traf-

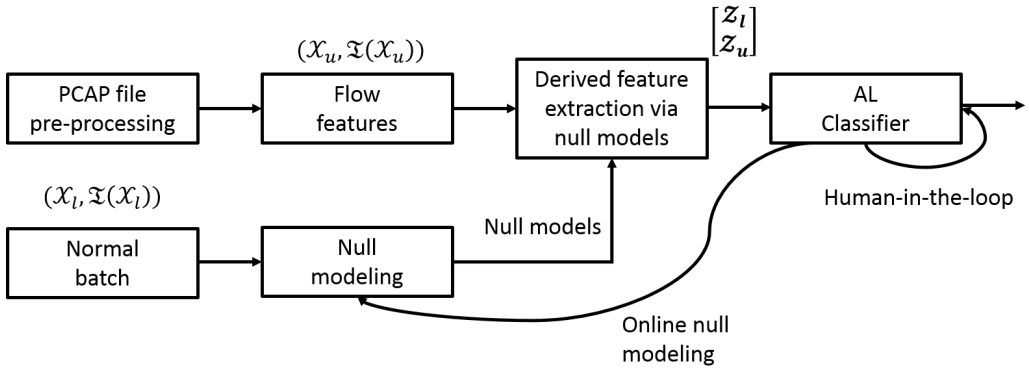


Figure 3.11: The overall AL system design.

fic from LBNL traces [31]. These traces, constituting more than 100 hours of recorded activity, were captured on a mid-size enterprise network, covering 22 subnets, from October 2004 through January 2005. This dataset contains web traffic on TCP port 80

with recorded time-of-day information. Specifically, the experiments in this paper are based on three PCAP files named "200412215-0510.port008", "20041215-1343.port008", and "20041215-1443.port010". There are respectively 2925, 5413, and 1634 web flows with at least 10 packets after the 3-way handshake. Flows with less than ten packets are not used in our experiments. We also investigate three distinct botnet applications, namely Zeus, Waledac and Storm bots. The Zeus bot is well-known for its detection evasion techniques such as randomization of proxy servers and/or port numbers, which make it very difficult to detect; it and its variants have become the most popular botnet application on the Internet today, especially for cybercrime activities. Waledac is currently one of the most prevalent P2P botnets and is widely considered the successor to the Storm botnet, with a more decentralized communication protocol [52]. We obtained Zeus PCAP files from [63] and [52], whereas both Waledac and Storm botnets are from the ISOT botnet data set [52]. Both C&C and non C&C traffic are combined and used in our experimentation.

In Table 3.2, the sample sizes of these web and botnet traffic traces shown. All traffic uses TCP as the transport protocol, and we use the first ten packets from each flow after the three-way handshake.

Table 3.2: Normal web and botnet flow sizes.

Application	**Number of Flows Used**
LBNL Web	9972
VRT Zeus	64
ISOT Zeus	23
ISOT Storm	3974
ISOT Waledac	4926

We could have involved a plurality of different modeled zero-day botnets in the current unlabelled flow-batch $\mathcal{X}_u$ and/or combine LBNL background traces together with a (different) deemed known botnet trace to inform the null. To simplify matters, the null was learned only using LBNL traces and only a single botnet appeared in $\mathcal{X}_u$.

3.2.3.1 Performance Metrics

For botnet anomaly detection, we are interested in the following generalization performance criteria. One is true positive rate or sensitivity: the ratio of botnet flows that are classified as truly botnet; the other is false positive rate or specificity: the ratio of web flows that are falsely classified as botnet flows. To give the comprehensive trade-off between the true positive and false positive rates, we use receiver operating characteristic

area under the curve (ROC AUC) as the generalization measure on the test batch, in all of our experiments. The ROC AUC is calculated as a function of the number of active labelings.

Note that other metrics could be used depending on the sensitivity to false positives (public ISP) or false negatives (military, intelligence, financial) of the NIDS deployment setting. For example, in a deployment setting with greater sensitivity to false positives, instead of the ROC AUC, the number of true positives before the first false positive is detected could be used to evaluate an anomaly detection or active learning framework, a quantity discernible from the (entire) ROC.

3.2.3.2 Experimental Results

For each experiment involving a background/normal LBNL trace and a botnet C&C trace:

1. One third of the normal batch is first randomly subsampled without replacement from the whole normal batch and used to train the Bayesian Network and all the marginal and pairwise packet size GMMs. The derived feature vector $\underline{z}$ is then obtained for each flow sample.

2. Half of the remaining two thirds of the normal traffic are treated as unlabeled and combined with half of the unlabeled botnet traffic for active learning (semi-superived AL training).

3. Finally, the remaining normal and attack samples are used for testing (measuring generalization performance of) our AL framework. Generalization performance is averaged over 5 random training-test splits.

We are particularly interested in the effectiveness of the proposed p-value based feature vector, compared with alternative feature representations. Besides the performance using associated raw features $(\underline{x}, \mathcal{I}(\underline{x}))$, we also compare with popular derived flow-based feature-sets in the current literature for network traffic classification. Through a correlation-based filtering process, [33] identified 8 among the 248 flow based traditional features as highly discriminative for different network flows. [5] additionally used IP-ratio and goodput in their experiment - we denote this feature set as CSET'11. Since the botnet applications are from different domains than the web traffic, we only use the time-independent features from [5], *i.e.*, RTT-samples and goodput are not used - these "standard" flow based features are shown in Table 3.3. Also compared is the feature

representation proposed in [49], *i.e.*, without the use of a Bayesian Network to capture presence/absence features for packets, but using GMM based p-values. We denote this feature set as TNNLS'16. We are also interested in the effectiveness of different AL strategies applied to the domain of botnet detection.

Table 3.3: Standard flow based feature representation [33] [5].

Abbreviation	**Description**
cnt-data-pkt	The count of all the packets with at least a byte of TCP data payload (client to server)
min-data-size	The minimum payload size observed (client to server)
avg-data-size	Data bytes divided by number of packets (server to client)
init-win-bytes	The total number of bytes sent in initial window(server to client & client to server)
IP-bytes-med	Median of total bytes in IP packet (client to server)
frame-bytes-var	Variance of bytes in (Ethernet) packet (server to client)
IP-ratio	Ratio between the maximum packet size and minimum packet size (server to client & client to server)

In Figure 3.12, we compare different feature representations, using MLU as the AL sample selection strategy. As expected, the proposed feature set greatly outperforms the CSET'11 feature set for Zeus and Storm botnet classification, as well as greatly outperforming the feature set proposed in [49] for general-purpose anomaly detection, especially for the two Zeus bots. This is because in [5], the absence of a data packet in a given direction is ignored; however, this information is seen to be highly discriminative between web and Zeus (Figure 3.12 (a)), and between web and Storm botnets (Figure 3.12 (c)). We also show supervised learning results for various feature representations in Table 4 with several standard supervised learning models. We use the following shorthand notations: (SVM) support vector machine. (C4.5) C4.5 decision tree. (T) TNNLS'16. (R) raw features $\underline{x}$. (C) CSET'11 features. (P) proposed features $\underline{z}$. (ALL) combined features Table 3 and $\underline{z}$. Again, we see the superior performance of the proposed feature representation in discriminating Zeus bots from web. Note that performance of the combined feature representation shown in Table 3.4 (ALL) has no advantage over the

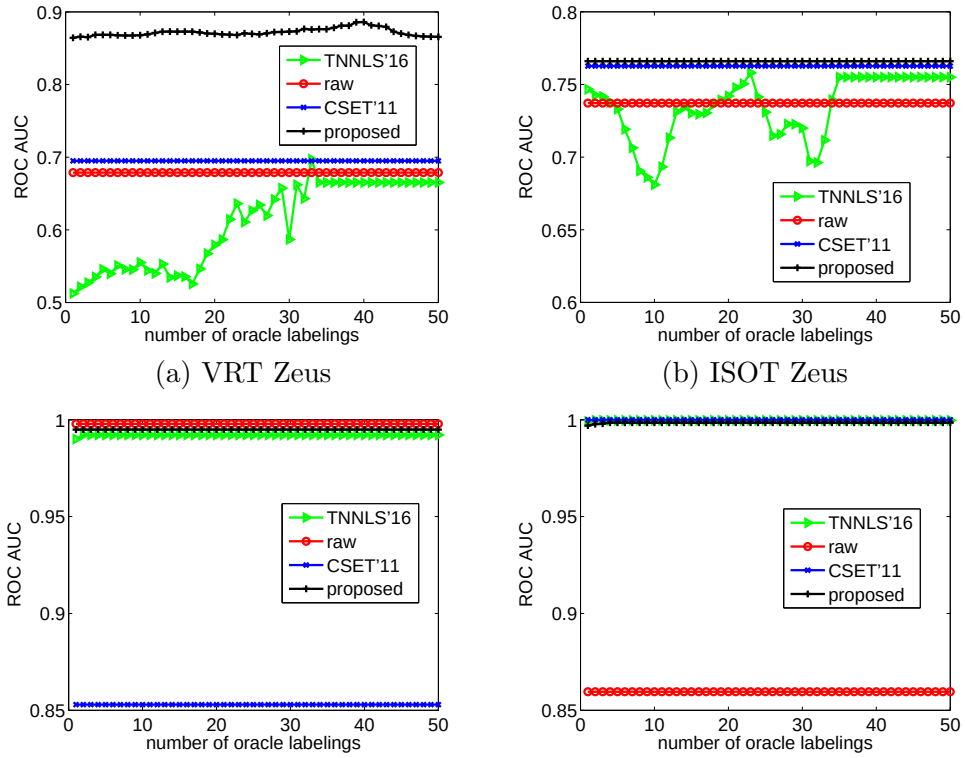
Table 3.4: ROC AUC results for various supervised learning models, comparing various feature representations.

Data Set	**SVM**					**C4.5**				
	T	R	C	P	ALL	T	R	C	P	ALL
VRT Zeus	0.84	0.84	0.83	**0.91**	**0.91**	0.84	0.83	0.83	**0.91**	**0.91**
ISOT Zeus	0.84	0.81	0.81	0.84	**0.85**	0.84	0.80	0.82	0.84	0.84
ISOT Storm	0.99	**1.0**	0.91	**1.0**	**1.0**	**1.0**	**1.0**	0.91	**1.00**	**1.0**
ISOT Waledac	**1.0**	0.99	**1.0**	**1.0**	**1.0**	**1.0**	0.99	**1.0**	**1.0**	**1.0**

standalone proposed features, in discriminating all the investigated botnet applications, suggesting that the information contained in standard features is well-captured by our proposed anomaly-based features. Note also that the AL-MLU performance for the proposed feature set in Figure 3.12 is very close to that of supervised learning (given botnet samples for training).

The proposed maxEnt model (3.7) produces a highly sparse solution, e.g., for VRT Zeus, 95.2% of the $\{\beta_i^{(2)}, i = 1, \ldots, 2D + \binom{D}{2}\}$ are zero by the fiftieth AL iteration, essentially eliminating the effects of these (spurious) features while still achieving comparable ROC AUC performance (about 88%) to that of the supervised learning models (about 90%) as shown in Table 3.

Next in Figure 3.13 , we compare different AL strategies and the standard minimum entropy based model (minEnt [20]) using the proposed feature vector. The premise behind the semi-supervised minEnt approach is to make classification decisions as confident as possible, consistent with margin maximization on all the samples. However, here we consider whether such a strategy is a sound one when there are few or no labeled samples for some classes initially. Seen in the figure, the MLU strategy outperforms the other two strategies, as was also claimed in [49]. Note that minEnt, with initially no labeled botnet traffic, will classify all unlabeled samples as normal web traffic due to its objective of minimizing unlabeled sample's entropy. Because this degenerate situation in minEnt will not change before observing the first botnet sample, minEnt will fail in the initially unknown botnet setting, always unbiasedly classifying all unlabeled samples as normal, *i.e.*, $\beta_0^{(2)} < 0, \beta_i^{(2)} = 0, \forall i$. Hence, to compare its model performance to that of maxEnt [49] during AL, we used the AL labeling stream from the maxEnt classifier with MLU, and fed these informative samples to minEnt. As seen in Figure 3.13, even when being fed informative labeled samples, minEnt stays flat at the lowest possible ROC AUC (0.5) during the initial learning phase (before the first botnet is labeled), and it requires more labeled samples to achieve the same level of generalization performance as that



(a) VRT Zeus (b) ISOT Zeus

(c) ISOT Storm (d) ISOT Waledac

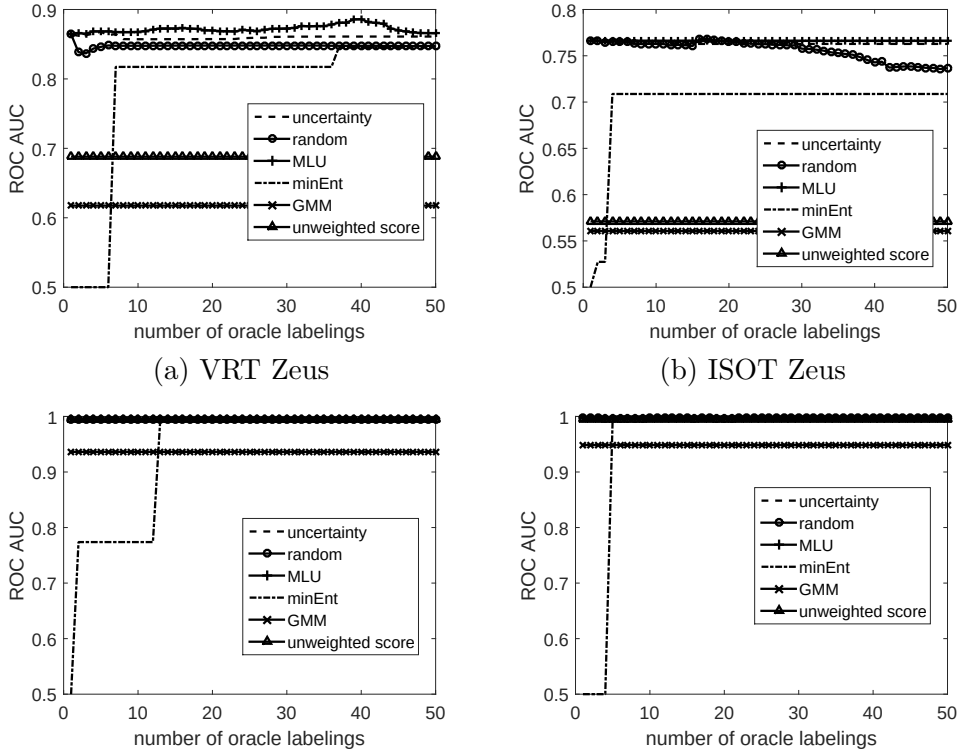
Figure 3.12: Active learning experiment: comparison of different feature representations.

of the maxEnt model. This shows that the maxEnt framework is much more suitable for unknown botnet detection, compared to minEnt. Moreover, the minEnt objective is non-convex, thus this approach is susceptible to finding poor local minima. As more and more samples are labeled, minEnt starts to learn and has very similar performance to that of maxEnt, as seen on Storm and Waledac.

Also compared is an unsupervised full GMM anomaly detector, trained on the labeled normal traffic, and an approach that unbiasedly sums up the log p-value based features (unweighted score). Both of these unsupervised learning approaches have lower ROC AUC compared to the AL based model except on ISOT Waledac, suggesting the value of maxEnt learning and the use of sparing labeling in improving generalization performance.

3.2.4 Discussion and Future Work

The proposed AL IDS can be mounted on enterprise network security platforms, including the Cisco Adaptive Security Appliance (ASA), Symantec Gateway Security (SGS), and IBM SiteProtector. Following the block diagram presented in Figure 3.11, the de-



(a) VRT Zeus (b) ISOT Zeus

(c) ISOT Storm (d) ISOT Waledac

Figure 3.13: Active learning experiment: comparison of different AL strategies.

veloped software can instantiate various active taps (probes) in different networks and subnetworks, analyzing the flow-level traffic sequence, and digesting/classifying locally aggregated real-time traffic periodically (e.g., hourly). The null model can be independently maintained and actively updated on separate taps by administrators, thus ensuring different security QoS requirements on various subnets.

By using the proposed feature representation, several common botnet evasion schemes become ineffective. For example, the random back-off presented in [60] can be completely overcome simply by ignoring (as above) timing-based features such as goodput or RTT, see [5]. Also, encryption of data packets may not affect performance using the proposed feature representation as it only requires packet-size information of the bidirectional TCP flows. Obviously, randomization of port number and domain names (fast flux) would also be ineffective when these features are not used, as above. However, flow "perturbation" or "noise injection" [60], can significantly impact the performance using our derived features. For example, if the intruder has detailed knowledge of the normal traffic patterns at the detection point, then they can adaptively groom their bidirectional

packet-size sequence accordingly to defeat a detection scheme predominantly based on such features. Alternatively, if the attacker knows the underlying AL approach and its state, then they can adaptively inject traffic sequences which adversely affect the labeling process and thus degrade classification performance, either through the introduction of highly uncertain traffic or extreme outliers or both. But such evasion techniques have significant overhead and require packet-traffic telemetry that may not be available to the bot slaves or master end-hosts. Hence, evasion is possible but may have very high implementation complexity and require adaptation overhead leading to less covert malware [60]. Note generally that active learning is a suitable defensive framework for such cat-and-mouse scenarios of jointly evolving attack and defense.

For future work, we will include other botnet applications in our performance evaluation. Given botnet activity observed *in situ* with background traffic, we will also consider the use of timing-based features (possibly including the inferred client OS type) to detect botnets. We will also consider detection of UDP-based anomalous traffic, e.g., botnet communication within streaming forms of media such as audio or video. A challenge here is that an entire UDP session may not be captured by a single network tap and so may need to be constructed by correlating UDP session-fragments from multiple taps.

Chapter 4

Detecting Clusters of Anomalies on Low-Dimensional Feature Subsets with Application to Network Traffic Flow Data

4.1 Introduction

Group anomaly detection has recently attracted much attention, with applications in astronomy, social media, disease/custom control, and network intrusion detection. In this work, we focus on group anomaly detection applied to network intrusion detection, where the anomalous groups are either distributed botnet (Zeus) or peer-to-peer (P2P) nodes generating traffic that deviates from the normal (Web traffic) behavior. Many existing intrusion detection systems (IDSs) only make sample-wise anomaly detections, e.g., in [58], the samples (individual flows, in the network IDS case) which deviate most from a normal (reference) model are flagged as anomalies/outliers. However, such an approach does not identify anomalous *groups* (e.g., a collection of botnet flows), whose samples all exhibit similar behavior. Identifying such groups could be essential for mounting some form of system response or defense. Moreover, individual samples may only be weakly atypical. Thus, a sample-wise IDS may either fail to detect most of the anomalous samples, or may incur high false positives when a low detection threshold is used. By contrast, (weakly) anomalous samples whose anomalies are all "similar to each other" may be *strongly* atypical when considered jointly. It should be noted that there is an

enormous number of candidate anomalous clusters, considering the conjoining of all possible sample subsets and all possible feature subsets. Thus, a GAD scheme will require some type of heuristic search over this huge space, aiming to detect the most statistically significant cluster candidates. In the sequel, we propose such a GAD scheme. Rather than assuming individual features or outlier events are statistically independent under the null hypothesis as in [38, 28], in our approach we capture and exploit statistical dependencies amongst the features defining a candidate cluster. Compared to previous works, as shown in our experiments, the proposed scheme is more effective in detecting group anomalies.

The chapter is organized as follows. Section 4.2 defines the problem and elaborates on related works. Section 4.3 describes the proposed model. Section 4.4 evaluates the system performance, and compares with some recent works, followed by conclusions in Section 4.5.

4.2 Problem Definition and Related Work

We assume there is a batch of normal labeled web traffic available at the outset as training set, *i.e.*, $X_l = \{\underline{\tilde{x}}_i, i = 1, ..., T_l, \underline{\tilde{x}}_i \in \mathbb{R}^D\}$, where $\underline{\tilde{x}}_i$ is a D-dimensional feature vector representing the i-th training traffic flow, and where we assume the number of training flows T_l is large enough to learn an accurate reference model (null hypothesis). These traffic flows can either be captured in a sandbox environment, or sampled from a domain of interest (data warehouse, enterprise network) in real time under normal operating conditions. Given a model of normal network traffic learned based on $\mathcal{X}_l$, our goal is to interrogate a captured batch of unlabeled traffic flows $\mathcal{X}_u = \{\underline{x}_i, i = 1, ..., T_u, \underline{x}_i \in \mathbb{R}^D\}$, seeking to detect latent groups of botnet or P2P traffic, with the flows in each such group exhibiting similar behavior. This has been previously considered in [28], where the authors used the samples in $\mathcal{X}_l$ to estimate bivariate Gaussian Mixture Models (GMMs), on all feature pairs, representing the null hypothesis. These bivariate GMMs were used to evaluate *mixture-based p-values* (the probability that an event is more extreme than the given observation) for all pairs of features. Assuming the features (tests) are statistically independent, a joint significance score function was defined for a given candidate cluster, specified by its sample subset and feature subset, with a Bonferroni approximation used to account for multiple testing. However, the independent tests assumption used in [28] becomes grossly invalid as more and more features are included in a cluster, which limits the proposed model's detection accuracy for increasing order K. Here, "order" is used

to denote the maximum feature subset dimension considered. A related framework was also proposed in [38], albeit assuming categorical attributes. The authors built a single, global null hypothesis Bayesian network based on $\mathcal{X}_l$. They then assigned categorical-based p-values to samples in X_u, with a cross entropy based scoring criterion used to efficiently search for the best feature and sample subset candidates. A limitation of this approach is that the statistical tests are again assumed to be independent.

Here, we describe and experiment with a GAD method that extends [38, 28]. The proposed method captures dependencies between the features in a candidate cluster by a dependence tree structure, and uses this model to help evaluate joint p-values for cluster candidates. As in [28], the Bonferroni corrected score is used as the objective function for evaluating the best cluster candidates (defined by their sample and feature subsets). The candidate with the best such score is detected as a cluster of anomalies. Whereas in [38] a single global (null model) Bayesian network is used to assess candidate clusters, in the current work a local, customized *cluster-specific* dependence tree model is used to assess each candidate cluster.

4.3 Proposed Model

4.3.1 Mixture-based P-values for Singletons and Feature Pairs

Consider a (sample, feature) index pair (i, j) and let $I_i^{(j)}$ be a binary indicator variable for the event that the j^{th} feature value of the i^{th} sample, $x_i^{(j)}$, is an outlier with respect to the null distribution for feature $X^{(j)}$. Let $O^{(j)}(x_i^{(j)})$ be a subset of the real line such that, $\forall y^{(j)} \in O^{(j)}(x_i^{(j)})$, $y^{(j)}$ is "more extreme" than the given observation $x_i^{(j)}$. One good definition for this set, consistent with evaluating a 2-sided p-value for a unimodal, symmetric null for $X^{(j)}$, is:

$$O^{(j)}(x_i^{(j)}; \mu^{(j)}) = \{y^{(j)} : |y^{(j)} - \mu^{(j)}| \geq |x_i^{(j)} - \mu^{(j)}|\},$$

where $\mu^{(j)}$ is a representative (mean) value for feature $X^{(j)}$. Given the component means $\mu_l^{(j)}, l = 1, ..., L_j$, of an L_j-component Gaussian mixture null, let $M^{(j)}(x)$ be a function that maps x to the mixture component index set $\{1, 2, ..., L_j\}$, *i.e.*, it indicates which mixture component generated x. Also, let Y_j be a random variable distributed according to the mixture density $f_{X_j}(x)$. Then, for a given observation $x_i^{(j)}$, we define the binary random variable $I_i^{(j)}$, where $I_i^{(j)} = 1$ if Y_j is more extreme under the null than $x_i^{(j)}$.

Then, we can write the singleton mixture p-value as:

$$\begin{aligned} & P[I_i^{(j)} = 1] \\ &= P[Y_j \in \cup_{l=1}^{L}((O^{(j)}(x_i^{(j)}; \mu_l^{(j)})) \cap (M^{(j)}(x_i^{(j)}) = l))] \\ &= \sum_{l=1}^{L} P[Y_j \in O^{(j)}(x_i^{(j)}; \mu_l^{(j)})] P[M^{(j)}(x_i^{(j)}) = l]. \end{aligned} \tag{4.1}$$

Here, an extreme outlier event is conditioned on $x_i^{(j)}$ having been generated by component density l. The probability $P[Y_j \in O^{(j)}(x_i^{(j)}; \mu_l^{(j)})]$ is the two-sided Gaussian p-value, integrating over the region $|y - \mu_l^{(j)}| \geq |x_i^{(j)} - \mu_l^{(j)}|$, while $P[M^{(j)}(x_i^{(j)}) = l]$ is the *a posteriori* probability that $x_i^{(j)}$ was generated by component l.

Similarly, for a *pair* of feature observations $(x_i^{(j)}, x_i^{(k)})$, we have the second order mixture p-value:

$$\begin{aligned} & P[I_i^{(j)} = 1, I_i^{(k)} = 1] \\ &= \sum_{l=1}^{L} P[Y_j \in O^{(j)}(x_i^{(j)}; \mu_l^{(j)}), Y_k \in O^{(k)}(x_i^{(k)}; \mu_l^{(k)})] \\ &\qquad \cdot P[M^{(j,k)}(x_i^{(j)}, x_i^{(k)}) = l]. \end{aligned} \tag{4.2}$$

Here, $P[Y_j \in O^{(j)}(x_i^{(j)}; \mu_l^{(j)}), Y_k \in O^{(k)}(x_i^{(k)}; \mu_l^{(k)})]$ integrates the l-th component bivariate Gaussian density over the region

$$\{(y_j, y_k) : |y_j - \mu_l^{(j)}| \geq |x_i^{(j)} - \mu_l^{(j)}|, |y_k - \mu_l^{(k)}| \geq |x_i^{(k)} - \mu_l^{(k)}|\}.$$

This region consists of the union of four unbounded rectangular regions in the plane. Also, $P[M^{(j,k)}(x_i^{(j)}, x_i^{(k)}) = l]$ is the *a posteriori* probability that $(x_i^{(j)}, x_i^{(k)})$ was generated by mixture component l.

In this work, a sample's anomalousness on a given feature subset is estimated by a joint p-value, with statistical dependencies between features accounted for by a dependence tree (DT) structure presented in [9]. Since the dependence tree is based on first and second order probabilities, the joint p-value will be based on the singleton and second order mixture p-values, as given above. A smaller joint p-value indicates a sample is more anomalous under the given feature subset.

4.3.2 Scoring Clusters

Let $\{I_c, J_c\}$ denote cluster candidate c, I_c its sample subset and J_c its feature subset. Let $T_c = |I_c|, N_c = |J_c|$. Note that p-values are uniformly distributed on $[0, 1]$ under the null. Thus, given a cluster with feature subset J_c, from a test batch of size T_u, the probability that at least one cluster with T_c samples has a smaller joint p-value than $\prod_{i \in I_c} P[\bigcap_{j \in J_c}(I_i^{(j)}) = 1]$ is:

$$1 - (1 - \prod_{i \in I_c} P[\bigcap_{j \in J_c}(I_i^{(j)}) = 1])^{C(T_u, T_c)} \tag{4.3}$$

Here, $C(T_u, T_c) = \binom{T_u}{T_c}$, *i.e.*, it is the number of combinations and accounts for all possible sample subset configurations in a cluster with T_c samples, from a test batch of size T_u. In statistical parlance, this exponent "multiple test corrects" for all possible sample subset configurations. In principle, (4.3) provides a sound basis at least for directly comparing all cluster candidates with the same feature subset J_c. However, it does not allow comparing pairs of cluster candidates with any configurations of (T_c, N_c), because all possible feature subset configurations at a given order, N_c, have not been accounted for – (4.3) only accounts for all cluster candidates with the same feature subset, J_c. Also, (4.3) requires evaluation of the joint p-value $P[\bigcap_{j \in J_c}(I_i^{(j)}) = 1]), \forall i \in I_c$, which in general depends on the joint null density function for $(X_{j_1}, X_{j_2}, ..., X_{j_{N_c}}), j_m \in J_c, m = 1, ..., N_c$. When D is large, it is not practically feasible to learn and store these $\binom{D}{N_c}$ joint null density functions, *i.e.*, for all possible combinations of features up to order N_c. Thus, it appears some tractable representation of $P[\bigcap_{j \in J_c}(I_i^{(j)}) = 1])$ is needed. An obvious temptation is to assume that $I_i^{(j)}$ and $I_i^{(j')}$ are statistically independent $\forall j, j' \in J_c, j' \neq j$. But this is a very poor assumption, consistent with assuming the features are independent.

To address the above problems, we seek to modify (4.3) in two respects. First, we propose to multiple test correct both for the different sample and the different feature subsets, given a cluster candidate with (T_c, N_c). In this approach, instead of the exponent being the number of combinations, it becomes the product of combinations on samples and combinations on features, *i.e.*, the probability that at least one cluster with (T_c, N_c) has a smaller joint probability than $\prod_{i \in I_c} P[\bigcap_{j \in J_c}(I_i^{(j)}) = 1]$ is:

$$S(I_c, J_c) = 1 - (1 - \prod_{i \in I_c} P[\bigcap_{j \in J_c}(I_i^{(j)}) = 1])^{\binom{D}{N_c}\binom{T_u}{T_c}}.$$

The Bonferroni (upper bound) approximation of $S(I_c, J_c)$ is derived as follows. Let p_l be the joint p-value for the l-th cluster candidate (out of $\binom{D}{N_c}\binom{T_u}{T_c}$ candidates with configuration (T_c, N_c)). Then, we have

$$S(I_c, J_c) = P\left[\bigcup_l (p_l \leq \prod_{i \in I_c} P[\bigcap_{j \in J_c} (I_i^{(j)}) = 1])\right] \leq$$
$$\sum_l P\left[p_l \leq \prod_{i \in I_c} P[\bigcap_{j \in J_c} (I_i^{(j)}) = 1]\right] =$$
$$\binom{T_u}{T_c}\binom{D}{N_c} \prod_{i \in I_c} P[\bigcap_{j \in J_c} (I_i^{(j)}) = 1]. \tag{4.4}$$

Here, we have used the union bound and the fact that joint p-values are uniformly distributed under the null distribution. We will take the right hand side upper bound expression as our *joint score function* for evaluating cluster candidates.

For this joint significance measure, we can efficiently determine the optimal *sample* subset, given a fixed feature subset, by greedy sequential sample inclusion, in sorted joint p-value order. This is due to the unimodality of this Bonferroni approximated joint significance measure, as a function of the number of samples included in a cluster's sample subset (see next subsection).

Second, a rich, tractable, joint probability mass function model that does capture statistical dependencies is a restricted form of Bayesian network, based exclusively on first and second order distributions, *i.e.*, the *dependence tree* (DT), which factorizes the joint distribution $P[\bigcap_{j \in J_c} (I_i^{(j)}) = 1])$ as a product of first and second order probabilities [9]. In [9], it was shown that, even though there is an enormous number of unique dependence tree structures, one can efficiently find the globally optimal dependence tree, over all such structures, maximizing the dataset's log-likelihood, by realizing that this can be recast as a maximum weight spanning tree problem, with the pairwise weights defined as the mutual information between the pairs of random variables. The maximum weight spanning tree can be efficiently solved via Kruskal's algorithm, with complexity $O(N_c^2 log(N_c))$. Hence, given any candidate feature subset J_c, Kruskal's algorithm can be applied to determine the DT that maximizes the likelihood measured on $\mathcal{X}_l$, *i.e.*, the null hypothesis is determined, consistent with the given candidate feature subset J_c.

Based on a given DT structure, $P[\bigcap_{j \in J_c} (I_i^{(j)}) = 1])$ factorizes as a product of first and

second order distributions, *i.e.*, $\forall i \in I_c$:

$$P[\underset{j\in J_c}{\cap}(I_i^{(j)})] = P[I_i^{(j_1)}]P[I_i^{(j_2)}|I_i^{(j_1)}]...P[I_i^{(j_{N_c})}|I_i^{(j_{N_c}-1)}],$$

where we use j_1 to denote the root node of the DT representing J_c.

It is apparent from the above that, for any feature subset, one can represent the joint p-value of a given sample by its first and second order mixture p-values. That is, for any feature pair (j,k), $P[I_i^{(j)}|I_i^{(k)}] = \frac{P[I_i^{(j)},I_i^{(k)}]}{P[I_i^{(k)}]}$. The numerator and denominator are, respectively, the second and first order mixture-based p-values that we defined earlier. Note that the mixture p-value definition is different from Chapter 2. Also note that, in order to evaluate the first order mixture p-value $P[I_i^{(k)}]$, we marginalize feature j from the bivariate GMM for the feature pair (j,k). This gives us the GMM for feature k.

4.3.3 Identifying the Optimal Sample Subset I_c, Given Fixed J_c

Given a fixed J_c and associated DT, we would like to choose the sample subset I_c to minimize the joint score function (4.4). Note that $\prod_{i\in I_c} P[\underset{j\in J_c}{\cap}(I_i^{(j)}) = 1]$ strictly decreases as more samples are included in the cluster. On the other hand, $\binom{T_u}{T_c}$ increases with T_c for $T_c < T_u/2$. However, $\binom{T_u}{T_c}$ *decreases* for $T_c > T_u/2$. The objective function is thus globally minimized (for $T_c \leq T_u/2$) by the following procedure: i) sort the samples in increasing order of their joint p-values $P[\underset{j\in J_c}{\cap}(I_i^{(j)}) = 1]$; ii) sequentially include the samples on the sorted list into I_c, until the objective function no longer decreases or until $T_c = T_u/2$. This procedure globally minimizes over I_c given fixed J_c. Note also that, by applying the condition that (new score $<$ old score), new samples are included so long as the newest sample $\tilde{i}$ satisfies $P[\underset{j\in J_c}{\cap}(I_{\tilde{i}}^{(j)}) = 1] < \frac{T_c+1}{T-T_c}$.

4.3.4 Overall Search Algorithm

First, using the normal samples in $\mathcal{X}_l$, all the first and second order null GMMs are separately trained via the EM algorithm, with the Bayesian Information Criterion used for model order selection. Mutual information for all feature pairs is then calculated based on the bivariate GMMs. This is achieved by generating $M = 10^6$ samples from a given bivariate GMM distribution, and then estimating the mutual information by $\frac{1}{M}\sum_{n=1}^{M}\log(\frac{f_{X_1X_2}(x_1^{(n)},x_2^{(n)})}{f_{X_1}(x_1^{(n)})f_{X_2}(x_2^{(n)})})$. We then detect clusters in $\mathcal{X}_u$ sequentially, in a rank-prioritized fashion, each time choosing the cluster which minimizes the joint score (4.4). The algorithm operates on an enormous space of candidate clusters even if the feature

space itself is only modestly sized (D). We start by sweeping over feature subset candidates at low orders and, for tractability, only the "most promising" candidates at higher orders, with candidate feature subsets at order K formed by "accreting" new features to the best-scoring candidates at order $K-1$. For each candidate feature subset J_c, its DT is first learned and its associated, optimal subset I_c is then determined using the method described in section 4.3.3. Evaluating all candidates at all feature subset orders, the one with the smallest joint score at each order N_c is recorded. The cluster with smallest Bonferroni-corrected score (4.4) is then forwarded as detected. Its samples are then removed from the test batch. Subsequent cluster detections are then made following the same procedure. Cluster detections are thus made (in general) in order of decreasing joint significance.

4.4 Experimental Setup and Results

Our experiments focus on detecting Zeus botnet and P2P traffic among normal Web traffic. The Web packet-flows are obtained from the LBNL repository [31]. This dataset contains Web traffic on TCP port 80, with specified time-of-day information. Specifically, the experiments in this paper are based on three datasets named "200412215-0510.port008", "20041215-1343.port008" and "20041215-1443.port010". The protocols to obtain normal, P2P and botnet network traffic are the same as in [28], *i.e.*, we used the port-mapper in [66] to identify P2P traffic in these files by a C4.5 decision tree pretrained in another domain (the Cambridge dataset [32]). The Zeus botnet traffic are obtained from another domain [63].

4.4.1 Feature Space Selection and Representation

First, we did *not* use layer-4 port number features for purposes of detection [66, 5]. Also, we did not consider timing information here because the Zeus activity was recorded on another domain. In [5], previous efforts were made to detect botnet and P2P traffic using the well-known feature representation for network intrusion detection from [34]. The authors found that these features, though able to detect some attack activity, could not successfully discriminate botnet or P2P from normal Web traffic, *i.e.*, botnet and P2P traffic appear as "normal" Web activity according to the features from [34, 5].

To capture the intrinsic behavior of botnet and P2P packet-traffic, we note that most Zeus botnet traffic involves masters giving command (control) messages, while slaves execute the given commands. In the case of P2P, nodes often communicate in a bidirectional

manner, exchanging relatively large packets in both directions. Normal/background Web traffic, on the other hand, tends to involve server-to-client communications.

Hence, we seek to preserve the bidirectional packet size sequence information as feature representation for different traffic flows. This feature representation was previously considered in [28]. The authors used the first N (we set $N = 10$ in our experiments) packets after the three-way hand shake of each TCP flow. Then a feature vector of dimension $2N$ is defined, specified by the sizes and directionalities of these N packets. Traffic are assumed to be alternating between client-to-server (CS) and server-to-client (SC). A zero packet size is thus inserted between two consecutive packets in the same direction to indicate an absence of a packet in the other direction. For example, if the bidirectional traffic is strictly SC, a zero will be inserted after each SC packet size. This $2N$-dimensional feature representation preserves bidirectional information of a given TCP flow, which is essential for discriminating between P2P, Zeus and normal Web traffic.

4.4.2 Performance Metrics

Our algorithm detects clusters (groups) in a sequential fashion. For each extracted group, we rank the samples in the group by their associated joint p-values on the given feature subset. These samples will be sequentially removed from the test batch, with the system then continuing to extract groups until the test set is depleted. Then we sweep out an ROC curve based on these rank-ordered detected samples. A larger area under the ROC curve indicates earlier detections of anomalous groups, which implies the effectiveness of the intrusion detection system. We compare our system's performance with a GMM based anomaly detector, trained by normal samples, on the whole feature space. For this detector, we rank the test samples based on their data likelihood under the GMM, and sweep out an ROC curve. We also compare with the approach presented in [28], which assumes significance tests are independent (denoted "Independence tests"), and with the recent work presented in [38] with a slight modification – instead of discretizing feature values as done in [38], we use a single dependence tree null distribution learned on $\mathcal{X}_l$ and our proposed joint p-values for continuous features, $P[\bigcap_{j \in J_c} (I_i^{(j)}) = 1]$. We denote this variation on the approach in [38] by "single Bayesian Net." There are two generalization performance measures of interest on the test set: one is the aforementioned ROC area under the curve (ROC AUC), as a function of the maximum feature subset size for a cluster, $K_{\max}$. The other is the top 100 precision rate, defined as the fraction of anomalous samples amongst the first 100 detected samples. Lastly, instead of exhaustively searching over all feature subsets at order K, we trial-add individual features to the top

candidate feature subsets from order $K-1$. At each order K, starting from order 2, we only consider the top 500 candidates from order $K-1$.

Two different sets of experiments were performed, one on synthetic data, and the other on the network data mentioned earlier. In the synthetic dataset experiment, we used one unimodal Gaussian with 10 dimensions to generate normal samples and two additional unimodal Gaussians to generate two distinct anomalous clusters. The two anomalous clusters use the same distribution as the normal distribution for nine of the ten features. Thus, they deviate from the normal (null) distribution only on a single feature dimension (this "informative" feature dimension was different for the two clusters). Their corresponding sample subsets consist of 2.5% of the whole data batch $\mathcal{X}_u$ (so the proportion of anomalous samples in $\mathcal{X}_u$ is 5% of the total). The variance of the informative features was chosen to be the same as that of the normal features, σ_n^2. Moreover the mean of the informative feature under an anomalous cluster was chosen to be two standard deviations away from the mean under the normal class, *i.e.*, $|\mu_n - \mu_a| = 2\sigma_n$, where we use subscripts n and a to denote 'normal' and 'anomalous', respectively. Thus, if we consider only the informative feature dimension, the Bayes error rate in discriminating normal from anomalous is 15.87%. After generating the synthetic data batch (with a size of ten thousand samples), we randomly chose 20% of normal samples as ground-truth and used them to train the null hypothesis. The remaining normal samples were used as part of the test batch, along with the samples from the two anomalous clusters. This was repeated 10 times, with the performance averaged.

For the network data, all the normal web flows from the three files were combined, making nearly ten thousand normal web flows. We randomly selected 20% of these flows as ground-truth normal samples to train the null, and treated the remaining normal flows as part of the test batch, combined with either P2P or Zeus anomalous flows. We separately experimented with P2P and Zeus flows. There were roughly 5 % of either P2P or Zeus flows in a given test batch. Experiments for each scenario were averaged over 10 random train-test splits.

4.4.3 Experimental Results

In Figure 4.1, we show the performance on the synthetic data. Note that both the proposed scheme and [28] effectively capture groups of anomalies when the maximum feature subset order is two. The first captured cluster (sample subset) consists of more than 95% anomalous samples on average. However, as the maximum feature subset order increases, the "independence tests" approach drops significantly in performance. This is because

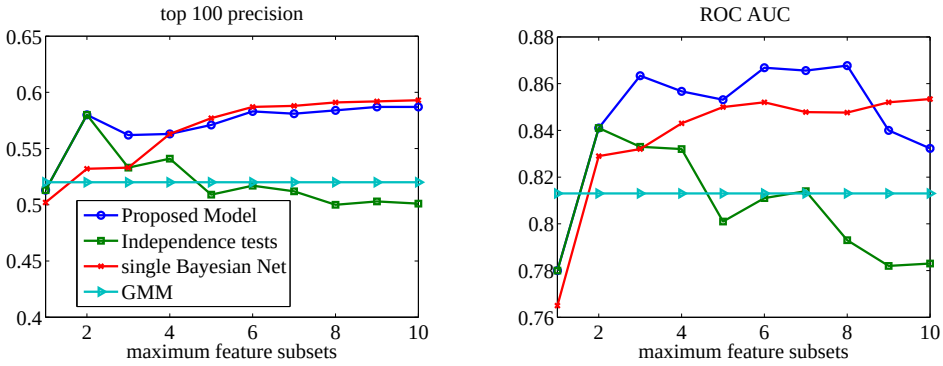


Figure 4.1: Synthetic data experiment: comparison of different schemes with 2 independent Gaussian based anomalous feature subsets in (separate) 1-dim subspace.

too many (assumed to be independent) pairwise tests create many redundant features that are all used to evaluate cluster anomalousness; use of these redundant features deemphasizes, within the score function, the important (low-order) feature subset. Also, we see an early advantage of using cluster-specific DTs, compared to the single Bayesian network approach. It appears that if an anomalous process is strictly generated from a low order subspace and normal in other feature dimensions (as is the case in this experiment) our cluster-specific DT approach outperforms a single Bayesian network approach. In Figure 4.2 a), we show the performance for normal-P2P discrimination. Compared to [28], which degrades in performance as more and more tests are included, we see superior performance for the proposed method. There is a large batch of anomalous samples captured at maximum order 6 by the proposed method, but both [38] and [28] did not capture this group effectively, as seen in the top 100 precision figure. Also, both of these methods are outperformed by the GMM baseline method. In Figure 4.2 b), we show the performance for normal-Zeus discrimination. Again, at maximum feature subset order 6 the proposed method captures a large portion of the anomalous flows – more than 50 Zeus flows were captured out of the first 100 flows detected by the proposed method. [28] performs poorly in this experiment, and again we observed that as the number of tests increase, the independence assumption degrades the detection performance. The single Bayesian network approach in [38] also performs relatively poorly on this dataset.

4.5 Conclusion

In this work, we proposed a GAD scheme to identify anomalous sample and feature subsets, accounting for dependencies between the features in a given subset. The proposed

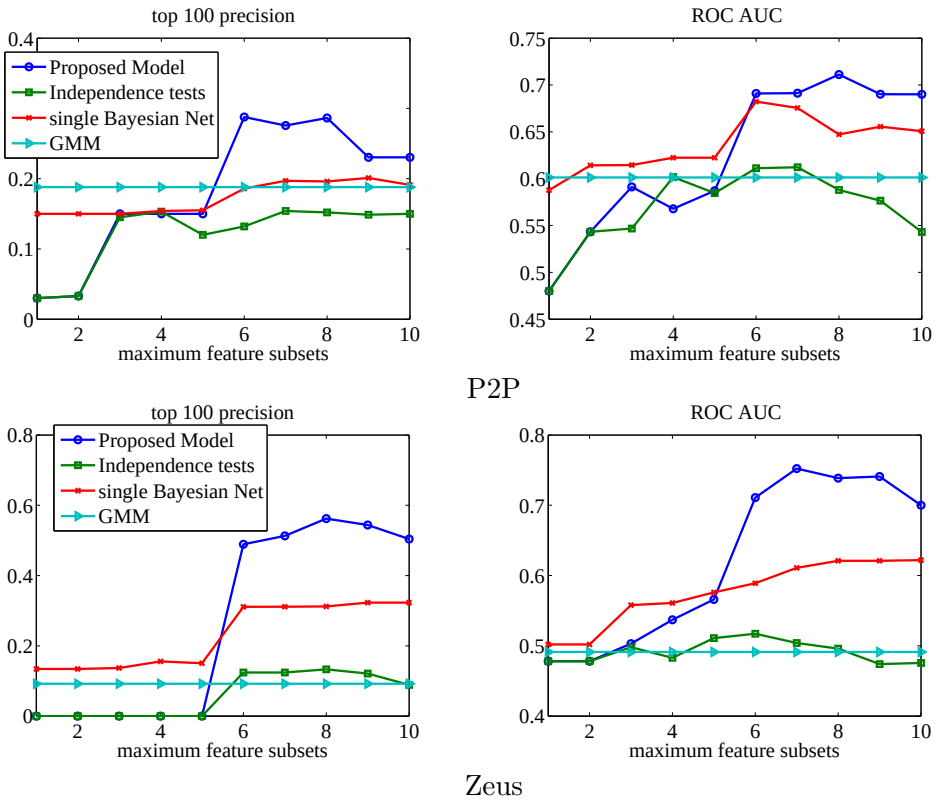


Figure 4.2: Network traffic data experiment: comparison of different schemes with P2P or Zeus anomalies.

model outperforms previous works that assume statistical tests are independent under the null. We demonstrated the effectiveness of our proposed system on both synthetic and real world data, with the latter drawn from the network intrusion detection domain, aiming to discriminate between normal and P2P/Zeus traffic. Our future work includes empirical p-value assessment and automatic determination of the maximum feature subset size of a cluster.

Chapter 5

Conclusion and Future Work

In this thesis, we developed a novel framework for semi-supervised active learning, using unlabeled samples as a source of regularization to preserve uncertainty among the unexplored high density region (maxEnt). Rather than minimizing the empirical entropy on the unlabeled samples, we seek to maximize it. We demonstrate its superior performance in both standard multi-class classification and rare category identification, compared to minEnt, especially when there are unknown categories. We have also successfully applied the maxEnt framework in vehicle tracking and network intrusion detection.

In the case of anomaly detection, we developed a novel group anomaly detection scheme to capture joint anomalies on a sample and feature subset, applied to network intrusion detection using bi-directional traffic flows. Rather than assuming individual features or outlier events are statistically independent under the null hypothesis as in [38, 28], in our approach we capture and exploit statistical dependencies amongst the features defining a candidate cluster. Compared to previous works, as shown in our experiments, the proposed scheme is more effective in detecting group anomalies. For future work in group anomaly detection, we would investigate the use of an alternative hypothesis to characterize the novel cluster.

Below, we provide one future research direction regarding the maxEnt framework.

5.1 MixEnt: Joint Update of Target Distribution and Posteriors

In a standard multi-class classification problem, consider a set of L labeled samples with K distinct categories, $\mathcal{X}_L = \{(\underline{x}_1, y_1), (\underline{x}_2, y_2), ..., (\underline{x}_l, y_l)\}$, where $\underline{x}_l \in \mathcal{R}^D, y_l \in$

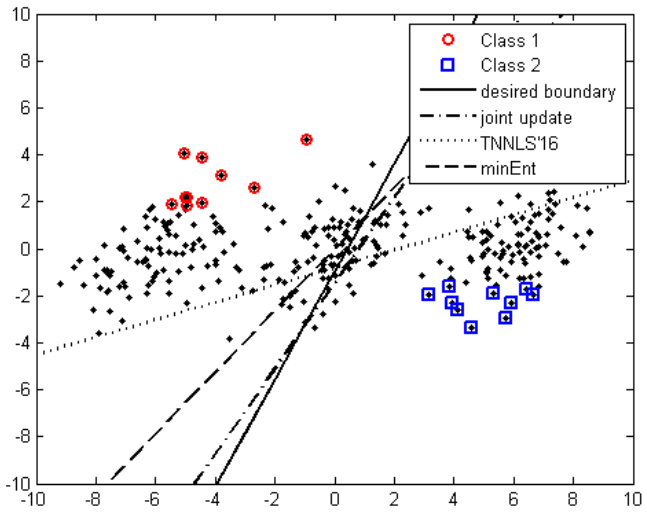


Figure 5.1: A three-cluster example with pathological labeled sample distribution from the two known categories. Synthetic experiment 1.

$\{1, 2, ..., K\}$, and $\mathcal{X}_U$ the set of U unlabeled samples whose categories are unknown, and each can either be from one of the K categories or from a hitherto unknown category. The previously proposed maxEnt learning objective in Chapter 2 is formulated as follows (we get rid of the rare categories and use raw features):

$$\begin{aligned} D_{\text{max-H}} &= -\sum_{(\underline{x}_n y_n)\in\mathcal{X}_l} \log P[y_n|\underline{x}_n] \\ &\quad +\alpha \sum_{\underline{x}_n\in\mathcal{X}_u^{(t)}} d(Q[C\in\{K,0\}]||P[C\in\{K,0\}|\underline{x}_n]) \end{aligned} \tag{5.1}$$

where $Q[C \in \{K, 0\}] = (\frac{1}{K+1}, \forall C)$, a uniform pmf, and we use index 0 to denote the unknown. Here, $d(Q||P) = \sum_k q_k \log(q_k/p_k)$ denotes the Kullback-Leibler divergence between probability mass functions $Q = \{q_k\}$ and $P = \{p_k\}$ defined on the same support, with Q the "target" distribution and P its (model-constrained) approximation. Clearly, minimizing (5.1) will globally maximize unlabeled samples' entropy. α is a non-negative hyperparameter that controls the amount of learning from the unlabeled samples. As α gets smaller, more weights will be put onto the labeled samples, tending to the maximum likelihood solution.

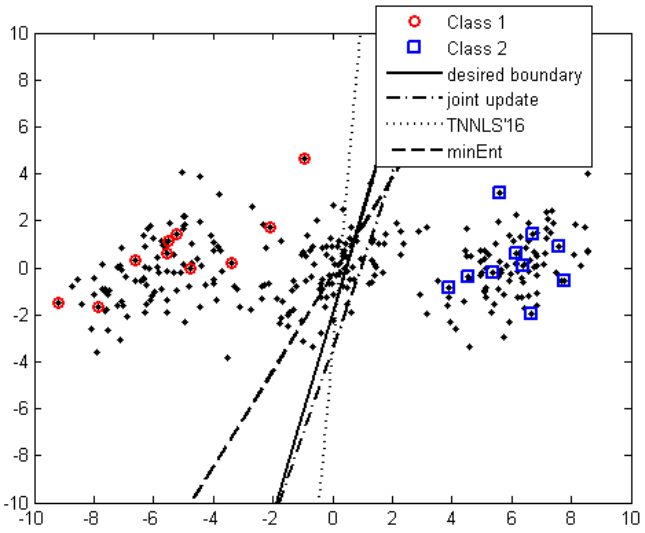


Figure 5.2: A three-cluster example with randomly chosen labeled samples from the two known categories. Synthetic experiment 2.

However, a (globally) maxEnt regularizer does not take into consideration the fact that some labeled and unlabeled samples within proximity ought to have the same label. One pathological example is shown in Figure 5.1 when the initial labeled samples are near the opposite ends of both known clusters: the maxEnt (as denoted by TNNLS'16 in the figure) solution cuts through both of the two supposedly well-formulated clusters and makes many more false predictions than the minEnt approach. The solution from minEnt could easily solve this problem, which as shown in the figure brings the decision boundary closer to the desired one (in which we assign the true target distribution for each sample, or equivalently, all the samples are labeled). But under different scenarios, minEnt can be overly confident about unknown clusters, hurting generalization performance, as shown in Figure 2.1. In order to preserve the advantage of minEnt on low density separation, while preserving uncertainty among local regions with purely unlabeled samples, we propose to jointly update both sample targets and parameters.

Accordingly, we seek to minimize the reformulated semi-supervised objective function, dubbed mixEnt, as follows:

$$min\ J(Z,\Theta) = -\sum_{(\underline{x}_n y_n)\in\mathcal{X}_l} \log P[y_n|\underline{x}_n] + \alpha \sum_{\underline{x}_n\in\mathcal{X}_u} d(z_n||P[C\in\{K,0\}|\underline{x}_n];\Theta) \qquad (5.2)$$

Table 5.1: Empirical Results on both synthetic data sets and real world data sets. (SYN1) Synthetic data set illustrated in Figure 5.1. (SYN2) Synthetic data set illustrated in Figure 5.2. (PB) Page Block. (SH) Shuttle Statlog.

Methods	**Avg.Entropy on unknown class**				**Error on known categories**			
	SYN1	SYN2	PB	SH	SYN1	SYN2	SH	PB
TNNLS'16	0.231	0.245	0.0564	0.236	5.00%	0	4.10%	24.0%
Joint Update	0.204	0.186	0.0249	0.236	0.00%	0	2.71%	24.0%
Desired	0.217	0.212	0.0447	0.203	0.00%	0	2.38%	23.9%
MinEnt	0.176	0.155	0	0.028	0.00%	0	2.49%	24.0%

where $Z \in \Re^{N\times(K+1)}$ and each row z_n denotes the target distribution for the n^{th} sample that have either Shannon entropy 0 (minEnt target, with probability 1 on one of the categories and zero on others. There are a total of $K+1$ of such target distributions) or it is a uniform distribution (maxEnt target, with probability $\frac{1}{K+1}$ equally distributed among all categories). We seek to optimize (5.2) by jointly determining Z and Θ through an EM-like iterative process, which is guaranteed to reach a local minimum:

1. Initialization: Z^t, with $z_n = \{\frac{1}{(K+1)}, \ldots, \frac{1}{(K+1)}\}, \forall n$, Θ^t
2. Assignment: in $Z^{(t+1)}$, $\forall n$, switch target z_n if doing so reduces the objective.
3. Gradient Descent on Θ: $\Theta^{(t+1)} = \Theta^t - \mu * \frac{\partial J}{\partial \Theta^t}$
4. End if $Z^{(t+1)} = Z^t$ and $\frac{|J^{(t+1)} - J^t|}{J^t} < 1e-6$. Else, $Z^{(t+1)} \leftarrow Z^t$, $\Theta^{(t+1)} \leftarrow \Theta^t$, and back to 2).

As observed in Figure 5.1, where we assume the labeled samples are pathologically chosen from the known clusters, the proposed mixEnt produces a decision boundary very close to the desired boundary. On the other hand, the method maxEnt in TNNLS'16 produces high entropy on *all* unlabeled samples, but with a decision boundary cut through the two known clusters, hurting generalization performance on the known categories, as seen in the error rate column in Table 5.1. minEnt approach, while achieving 0 error on the known categories, compared to the proposed mixEnt with joint update, has much lower entropy on the unknown category.

Figure 5.2 shows the case with randomly chosen labeled samples from the two known clusters. All methods have 0 error on the known categories. In this case, TNNLS'16 is the most preferred method because it produces the highest entropy on the unknown, as seen in Table 5.1.

In Table 5.1, we show the performance on a couple of real data sets (Page Block and Shuttle). These data set's property is described in Table 4.1. For each data set, we used only the majority classes for this experiment, randomly picking one of them as the unknown class and labeled 20% of the samples from each known class. Again, mixEnt has lower error rate on the known categories than the TNNLS'16 method, and has much higher entropy on the unknown than the minEnt method.

In conclusion, in a standard multi-class semi-supervised learning setting, the proposed method improves the error rate on the known categories while achieving a certain degree of uncertainty among the unlabeled samples, which provides a trade-off between maxEnt and minEnt. For future work, one can investigate more on the mixEnt strategy, by incorporating other information such as local regions, e.g., samples that belong to the same region should be assigned the same target distribution, and pairwise constraints on samples, and by developing theoretical guarantees on reaching a desired optimal solution. One can also investigate it in the AL setting, and its general applicability in characterizing rare categories and detecting unknown unknowns.

Bibliography

[1] C. S. Agate and K. J. Sullivan. Signature-aided tracking using association hypotheses. pages 44–55. International Society for Optics and Photonics, 2002.

[2] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *The Journal of Machine Learning Research*, 7:2399–2434, 2006.

[3] C. M. Bishop and N. M. Nasrabadi. *Pattern recognition and machine learning*. Springer New York, 2006.

[4] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, 1998.

[5] Z. B. Celik, J. Raghuram, G. Kesidis, and D. J. Miller. Salting public traces with attack traffic to test flow classifiers. In *Proc. USENIX Workshop on Cyber Security Experimentation and Test (CSET)*, San Francisco, CA, Aug. 2011.

[6] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *Computing Surveys*, 41(3):15, 2009.

[7] O. Chapelle, B. Scholkopf, and A. Zien. *Semi-supervised learning*, volume 2. 2006.

[8] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *Knowledge Discovery in Databases: PKDD 2003*, pages 107–119. Springer, 2003.

[9] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. on Information Theory*, 14(3):462–467, May. 1968.

[10] A. Corduneanu and T. Jaakkola. Data dependent regularization. In O. Chapelle, B. Schölkopf, A. Zien, et al., editors, *Semi-Supervised Learning*, pages 169–190. MIT Press Cambridge, 2006.

[11] K. Das, J. Schneider, and D. B. Neill. Anomaly pattern detection in categorical datasets. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 169–176. ACM, 2008.

[12] S. Dasgupta and D. Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning*, pages 208–215. ACM, 2008.

[13] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[14] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification, Second Edition*. John Wiley & Sons, 1999.

[15] C. Fraley and A. E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588, 1998.

[16] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, pages 23–37. Springer, 1995.

[17] A. Fujino, N. Ueda, and K. Saito. Semisupervised learning for a hybrid generative/discriminative classifier based on the maximum entropy principle. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):424–437, 2008.

[18] Z. Ghahramani, M. J. Beal, et al. Variational inference for bayesian mixtures of factor analysers. In *Advances in Neural Information Processing Systems*, pages 449–455, 1999.

[19] R. G. Gomes, A. Krause, and P. Perona. Discriminative clustering by regularized information maximization. In *Advances in Neural Information Processing Systems*, volume 23, pages 766–774, 2010.

[20] Y. Grandvalet, Y. Bengio, et al. Semi-supervised learning by entropy minimization. In *Advances in Neural Information Processing Systems*, volume 17, pages 529–536, 2004.

[21] G. Gu, R. Perdisci, J. Zhang, W. Lee, et al. Botminer: Clustering analysis of network traffic for protocol-and structure-independent botnet detection. In *Proc. The 17th USENIX Security Symposium*, pages 139–154, San Jose, CA, Jul. 2008.

[22] H. Guo and H. L. Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *ACM SIGKDD Explorations Newsletter*, 6(1):30–39, 2004.

[23] F. Haddadi, D. Le Cong, L. Porter, and A. N. Zincir-Heywood. On the effectiveness of different botnet detection approaches. In *Information Security Practice and Experience*, pages 121–135. Springer, 2015.

[24] M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *Pervasive Computing*, 7(4):12–18, 2008.

[25] T. Hastie, R. Tibshirani, J. Friedman, T. Hastie, J. Friedman, and R. Tibshirani. *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2001.

[26] J. He. *Rare category analysis*. PhD thesis, Carnegie Mellon University, 2010.

[27] T. M. Hospedales, S. Gong, and T. Xiang. A unifying theory of active discovery and learning. In *Computer Vision–ECCV 2012*, pages 453–466. Springer, 2012.

[28] F. Kocak, D. J. Miller, and G. Kesidis. Detecting anomalous latent classes in a batch of network traffic flows. In *Proceedings of the 48th Annual Conference on Information Sciences and Systems*, pages 1–6. IEEE, 2014.

[29] M. Kubat, S. Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *International Conference on Machine Learning*, volume 97, pages 179–186. Nashville, USA, 1997.

[30] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, pages 79–86, 1951.

[31] LBNL/ICSI Enterprise Tracing Project. `http://www.icir.org/enterprise-tracing`.

[32] W. Li, M. Canini, A. Moore, and R. Bolla. Efficient application identification and the temporal and spatial stability of classification schema. *Computer Networks*, 53(6):790–809, 2009.

[33] W. Li, M. Canini, A. W. Moore, and R. Bolla. Efficient application identification and the temporal and spatial stability of classification schema. *Computer Networks*, 53(6):790–809, Mar. 2009.

[34] W. Li and A. W. Moore. A machine learning approach for efficient traffic classification. In *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007. MASCOTS'07. 15th International Symposium on*, pages 310–317. IEEE, 2007.

[35] L. M. Manevitz and M. Yousef. One-class svms for document classification. *the Journal of Machine Learning Research*, 2:139–154, 2001.

[36] G. S. Mann and A. McCallum. Simple, robust, scalable semi-supervised learning via expectation regularization. In *Proceedings of the 24th International Conference on Machine Learning*, pages 593–600. ACM, 2007.

[37] S. C. Markley and D. J. Miller. Joint parsimonious modeling and model order selection for multivariate Gaussian mixtures. *IEEE Journal of Selected Topics in Signal Processing*, 4(3):548–559, 2010.

[38] E. McFowland, S. Speakman, and D. Neill. Fast generalized subset scan for anomalous pattern detection. *JMLR*, 14(1):1533–1561, 2013.

[39] E. Menahem, Y. Elovici, N. Amar, and G. Nakibly. ACTIDS: an active strategy for detecting and localizing network attacks. In *Proc. CCS Workshop on Artificial Intelligence and Security (AISec)*, pages 55–66, Berlin, Britain, Nov. 2013. ACM.

[40] D. J. Miller and J. Browning. A mixture model and EM-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1468–1483, 2003.

[41] D. J. Miller and H. S. Uyar. A mixture of experts classifier with learning based on both labelled and unlabelled data. *Advances in neural information processing systems*, pages 571–577, 1997.

[42] T. M. Mitchell. *Machine learning*. McGraw-Hill, New York, 1997.

[43] M. M. Moya and D. R. Hush. Network constraints and multi-objective optimization for one-class classification. *Neural Networks*, 9(3):463–474, 1996.

[44] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2):103–134, 2000.

[45] J. Nocedal and S. J. Wright. *Numerical Optimization, Second Edition*. Springer, 2006.

[46] D. Pelleg and A. W. Moore. Active learning for anomaly and rare-category detection. In *Advances in Neural Information Processing Systems*, volume 18, pages 1073–1080, 2004.

[47] L. Portnoy, E. Eskin, and S. Stolfo. Intrusion detection with unlabeled data using clustering. In *Proceedings of CSS Workshop on Data Mining Applied to Security*. ACM, 2001.

[48] Z. Qiu, D. J. Miller, and G. Kesidis. Detecting clusters of anomalies on low-dimensional feature subsets with application to network traffic flow data. In *Proc. 25th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, Boston, MA, Sep. 2015. IEEE.

[49] Z. Qiu, D. J. Miller, and G. Kesidis. A maximum entropy framework for semisupervised and active learning with unknown and label-scarce classes. *IEEE Trans. on Neural Network and Learning System*, Jan. 2016.

[50] Z. Qiu, D. J. Miller, and G. Kesidis. Flow based Botnet Detection through Semi-Supervised Active Learning. Technical Report CSE-16-010, CSE Dept, PSU, Sept. 12, 2016, http://www.cse.psu.edu/research/publications/tech-reports/2016/CSE-16-010.pdf.pdf/view.

[51] Z. Qiu, D. J. Miller, B. Stieber, and T. Fair. Actively learning to distinguish suspicious from innocuous anomalies in a batch of vehicle tracks. In *SPIE Defense and Security Symposium*. International Society for Optics and Photonics, 2014.

[52] S. Saad, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, J. Felix, and P. Hakimian. Detecting P2P botnets through network behavior analysis and machine learning. In *The 9th Annual International Conference on Privacy, Security and Trust (PST)*, pages 174–180, Montreal, QC, Canada, Jul. 2011. IEEE.

[53] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[54] G. Schwarz et al. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[55] B. Settles. *Active learning literature survey.* Computer Sciences Technical Report 1648, University of Wisconsin, Madison, 2009.

[56] J. Shore and R. Johnson. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *Information Theory, IEEE Transactions on*, 26(1):26–37, 1980.

[57] S. S. Silva, R. M. Silva, R. C. Pinto, and R. M. Salles. Botnets: A survey. *Computer Networks*, 57(2):378–403, Feb. 2013.

[58] R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of IEEE Symposium on Security and Privacy*, 2010.

[59] M. Stevanovic and J. M. Pedersen. Machine learning for identifying botnet network traffic. Technical report, Networking and Security Section, Department of Electronic Systems, Aalborg University, 2013.

[60] E. Stinson and J. C. Mitchell. Towards systematic evaluation of the evadability of bot/botnet detection methods. In *Proc. USENIX Workshop on Offensive Technologies (WOOT)*, volume 8, pages 1–9, Boston, MA, Jul. 2008.

[61] G. Trunk. A problem of dimensionality: A simple example. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (3):306–307, 1979.

[62] V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[63] VRT Labs - Zeus Trojan Analysis. `https://labs.snort.org/papers/zeus.html`.

[64] X. Zhu and J. Lafferty. Harmonic mixtures: combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. In *Proc. 22nd international conference on Machine learning*, pages 1052–1059. ACM, 2005.

[65] G. Zou, G. Kesidis, and D. Miller. A flow classifier with tamper-resistant features and an evaluation of its portability to new domains. *IEEE Journal on Selected Areas in Communications*, 29(7):1449–1460, 2011.

[66] G. Zou, G. Kesidis, and D. Miller. A flow classifier with tamper-resistant features and an evaluation of its portability to new domains. *IEEE J-SAC*, 29(7):1449–1460, 2011.

Vita

Zhicong Qiu

I received my B.S. and M.S. degrees from New York University Tandon School of Engineering, Brooklyn, NY, in 2012. From September 2011 through May 2012, I was a Teaching and Lab Assistant at NYU's Wireless Lab. I have been a Research Assistant at The Pennsylvania State University in the Electrical Engineering department since August 2012. I also worked as a research contractor during the summer of 2015 at Air Force Research Laboratory, Dayton, OH. My undergrad thesis advisors are Prof. Knox and Prof. Erkip, researching on Cooperative Wireless Communication. We implemented a software-defined radio (SDR) relay communication system using USRP board, and measured relay-and-forward strategy's information gain. My graduate thesis advisors are Prof. Xu, Prof. Xi and Prof. Chao, researching on solutions of Incast communication in MapReduce (most typical network architecture for SAN), developing a pacing TCP algorithm to mitigate the effect of Incast, simulating in ns-3.