EFFICIENT LEARNING WITH SOFT LABEL INFORMATION AND MULTIPLE ANNOTATORS

by

Quang Nguyen

BS, Moscow State University, 2006

Submitted to the Graduate Faculty of

the Dietrich School of Arts and Sciences in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2014

UNIVERSITY OF PITTSBURGH

COMPUTER SCIENCE DEPARTMENT

This dissertation was presented

by

Quang Nguyen

It was defended on

March 24th, 2014

and approved by

Milos Hauskrecht, PhD, Associate Professor, Computer Science

Janyce Wiebe, PhD, Professor, Computer Science

Jingtao Wang, PhD, Assistant Professor, Computer Science

Gregory Cooper, MD, PhD, Professor, Biomedical Informatics

Dissertation Director: **Milos Hauskrecht, PhD**, Associate Professor, Computer Science

EFFICIENT LEARNING WITH SOFT LABEL INFORMATION AND MULTIPLE ANNOTATORS

Quang Nguyen, PhD

University of Pittsburgh, 2014

Nowadays, large real-world data sets are collected in science, engineering, health care and other fields. These data provide us with a great resource for building automated learning systems. However, for many machine learning applications, data need to be annotated (labelled) by human before they can be used for learning. Unfortunately, the annotation process by a human expert is often very time-consuming and costly. As the result, the amount of labeled training data instances to learn from may be limited, which in turn influences the learning process and the quality of learned models. In this thesis, we investigate ways of improving the learning process in supervised classification settings in which labels are provided by human annotators. First, we study and propose a new classification learning framework, that learns, in addition to binary class label information, also from soft-label information reflecting the certainty or belief in the class label. We propose multiple methods, based on regression, max-margin and ranking methodologies, that use the soft label information in order to learn better classifiers with smaller training data and hence smaller annotation effort. We also study our soft-label approach when examples to be labeled next are selected online using active learning. Second, we study ways of distributing the annotation effort among multiple experts. We develop a new multiple-annotator learning framework that explicitly models and embraces annotator differences and biases in order to learn a consensus and annotator specific models. We demonstrate the benefits and advantages of our frameworks on both UCI data sets and our real-world clinical data extracted from Electronic Health Records.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PREFACE

I would like to thank everybody who made my journey to Ph.D. possible and enjoyable.

First of all, I want to express my deepest gratitude to my advisor, professor Milos Hauskrecht, who has always supported me in both professional career and personal life. Milos taught me machine learning and data mining, and how to conduct high quality research in these fields. He also encouraged and guided me through difficult moments during my graduate school years in Pittsburgh.

I would like to thank my thesis committee, Dr. Gregory Cooper, Dr. Janyce Wiebe and Dr. Jingtao Wang, for their valuable feedback and discussions on my thesis research.

Parts of this thesis are the results of collaboration with colleagues from University of Pittsburgh. In particular, I want to thank our former post-doc, Dr. Hamed Valizadegan (currently at NASA research), who worked with me on several papers and gave me insights in many machine learning and optimization techniques. I also would like to thank Dr. Gregory Cooper, Dr. Shyam Visweswaran and former and current members of Milos's lab: Charmgil Hong, Zitao Liu, Mahdi Pakdaman, Eric Heim, Dr. Iyad Batal (currently at GE research), Dr. Saeed Amizadeh (Yahoo Labs), Dr. Michal Valko (INRIA France), Dr. Tomas Singliar (Amazon Research) and Dr. Richard Pelikan (USC).

I am grateful to have many good friends in Pittsburgh, who made my life more enjoyable. In particular, I want to mention my best friend Nguyen Bach, who was always willing to help and gave me a lot of useful advices.

Finally, I am very grateful to my parents Le and Hung, my brother Vinh, my sister in law Hang and my niece Ha for their unlimited love, encouragement and support.

Thank you all !

1.0 INTRODUCTION

Nowadays, large real-world data sets are collected in various areas of science, engineering, economy, health care and other fields. These data sets provide us with a great opportunity to understand the behavior of complex natural and man-made systems and their combinations. However, many of these real-world data sets are not perfect and come with missing information we currently have no means to collect automatically. One type of such information is subjective labels provided by human annotators in the field that assigns data examples to one of the classes of interest. Take for example a patient health record, while some of the data (such as lab tests, medications given) are archived and collected, diagnoses of some conditions, or adverse events that occurred during the hospitalization are not. In the context of supervised learning, in order to analyze these conditions and predict them, individual patient examples must be first labeled by an expert or a group of experts. Moreover, supervised learning systems often perform well only if they are trained on a large number (hundreds, even thousands) of labeled examples. However, the process of labeling examples using subjective human assessments faces a number of problems:

First, collecting labels from human annotators can be extremely time-consuming and costly, especially in the domains where data assessment requires a high level of expertise. For example, in our disease diagnostics task, an experienced physician needs to spend about five minutes (on average) to review and evaluate one patient case [Nguyen et al., 2011a]; or in speech recognition tasks, [Zhu, 2005] reports that a trained linguist may take up to seven hours to annotate one minute of audio record (e.g. 400 times as long). The challenge is to find ways to reduce the number of examples that need to be reviewed and labeled by an expert while improving the quality of the models learned from these examples.

Second, because of the time it may take the human to review and annotate an example,

it is hard to expect that one annotator/expert will be able to label all examples. To address this, instead of using one annotator, we can use multiple annotators to label examples. However, different annotators may have different opinions, knowledge or biases, which lead to disagreements in the labeling. For example, one physician may diagnose a patient as having a certain disease, while another may say the opposite. Modelling and combining all agreements/disagreements among annotators is an important and interesting problem. Studying of this problem would help us get more insights into the labeling process, improve the quality of labels and the performance of learning models.

The first problem focuses more on the *quantity* of labels, while the second one focuses more on the *quality* of labels. In this thesis, we study and develop approaches to address both quantity and quality aspects of the annotation and learning process. Our ultimate goal is to increase the performance of classification models, while reducing and distributing the cost of labeling. Note that our work was originally motivated by applications in medical domain, so many examples and discussions in this thesis are related to this domain. However, in general, our proposed methods can be successfully applied in other domains, as we will demonstrate on many real-world benchmark data sets.

1.1 LEARNING WITH AUXILIARY SOFT-LABEL INFORMATION

The problem of optimizing the time and cost of labeling has been studied extensively in machine learning research community. One of the most popular research directions for this problem is Active Learning [Cohn et al., 1996]. The goal of active learning research is to develop methods that analyze unlabeled examples, prioritize them and select those that are most critical for the task we want to solve, while optimizing the overall data labeling cost. We explore another direction that is orthogonal to the active learning approach, which can help us to alleviate the costly labeling process in practice. The idea is based on a simple premise: a human expert who gives us a subjective label can often provide us with auxiliary information, which reflects his/her certainty in the label decision, at a cost that is insignificant when compared to the cost of example review and label assessment. To illustrate this

point, assume an expert reviewing electronic health record (EHR) data in order to provide some diagnostic assessment of the patient case. The complexity of clinical data prompts him/her to spend a large amount of time reviewing and analyzing the case. However, once the example is understood and the label decision is made, the cost of providing additional assessment of the confidence in this decision is relatively small. For example, according to our studies analyzing adverse clinical events (Heparin Induced Thrombocytopenia and Amiodarone Toxicity), experts spend four to six minutes before they make a decision on the condition and whether it is worthwhile to alert on it. In contrast, it takes them only a few seconds to provide an auxiliary assessment of confidence, e.g. the probability of having disease or the strength of the alert. Clearly, the cost to obtain this auxiliary information is insignificant compared to the whole data acquisition cost. The question is how to utilize it efficiently?

We propose and study a machine learning framework in which a classification learning problem relies, instead on just the binary class label information, on a more refined soft-label information reflecting the certainty or belief in this label. In general, this information can be a probabilistic/numeric score, e.g. chance of having disease is 0.7/1, 7/10, etc. or a qualitative category, e.g. weak, medium, strong belief in having disease. We expect this auxiliary soft-label information, when properly used, can help us to learn a better classification model. Briefly, we expect an instance with a high (or low) probability should be easier to classify by a proposed machine learning model, while an instance with probability close to 0.5 likely indicates the instance is harder to classify and is close to the decision boundary. The limitation of soft labeling is that assessments may be subject to noise. Studies by [Suantak et al., 1996, Griffin and Tversky, 1992, O'Hagan et al., 2007] showed that humans are not very good in assigning subjective probabilities. To address the problem, we propose novel methods based on the support vector machine and learning to rank methodologies, that are more robust to noise and able to learn high quality classifiers with smaller numbers of examples.

Note that our learning from auxiliary soft labels approach is complementary to active learning: while the later aims to select the most informative examples, we aim to gain more useful information from those selected. This gives us an opportunity to combine these two

approaches.

1.2 LEARNING WITH MULTIPLE ANNOTATORS

In practice the labeling process is often tedious and time consuming. At the same time the human expert effort is scarce. Therefore, it is hard to expect that one annotator/expert will be able to label all examples. To address this problem, we investigate and develop a learning framework that lets us use the labels obtained from multiple expert annotators. Our objective is to build a "consensus" model that agrees as much as possible with all experts and generalizes well on future unseen data. Note that this setting is different than the traditional supervised learning: instead of having a single annotator, we have a group of annotators labeling examples.

The biggest challenge in multiple-annotator learning is how to model and combine all agreements and disagreements among annotators. To develop a consensus model, we need to grasp the causes for the labeling disagreement of different annotators. The labeling disagreement may be rooted in (1) differences in the risks annotators associate with each class label assignment, (2) differences in the knowledge (or model) annotators use to label examples, and (3) differences in the amount of effort annotators spend for labeling each case. To illustrate the nature of these differences, let us consider the problem of diagnosing a patient from noisy and incomplete observations. First, diagnosing a patient as not having a disease when the patient has disease, carries a cost due to, for example, a missed opportunity to treat the patient, or longer patient discomfort and suffering. A similar, but different cost is caused by incorrectly diagnosing a patient. The differences in the annotator-specific utilities (or costs) may explain differences in their label assessments. Second, while diagnoses provided by different experts may be often consistent, the knowledge they have and features they consider when making the disease decision may differ, potentially leading to differences in labeling. It is not rare when two expert physicians disagree on a complex patient case due to differences in their knowledge and understanding of the disease. These differences are best characterized as differences in the model they use to diagnose the patient.

Third, different experts may spend different amounts of time to review the case and decide on the diagnosis. This may lead to mistakes in the labeling that are inconsistent with the expert's own knowledge and model. Our objective is to develop a learning framework that will embrace these types of differences and find a consensus model.

In this dissertation, I propose and develop a new multiple-annotator learning approach that takes into account the annotators' reliability as well as differences in the annotator-specific models and biases when learning a consensus model.

1.3 ORGANIZATION OF THE THESIS

This thesis is organized as follows:

Chapter 2 provides background and relevant research for supervised classification and approaches for label efficient learning. In particular, we start with an overview of supervised classification learning, with more details on logistic regression and maximum margin methods. Then we review relevant research in active learning, transfer learning, learning with auxiliary soft-label information and multi-annotator learning fields.

Chapter 3 describes our approach for learning with auxiliary soft-label information and the combination of active learning and auxiliary information in one learning framework. We demonstrate the benefits of our methods on a number of UCI data sets, while adding noise to auxiliary assessments. We also test our approach on real medical data representing experts assessment of the risk of the Heparin Induced Thrombocytopenia (HIT) [Warkentin et al., 2000] given a set of patients' observations and labs.

Chapter 4 describes our approach for learning with multiple annotators, followed by experimental results. We study our framework on synthetic (UCI-derived) datasets and on our real-world multiple expert learning problem in medical domain. First, for the synthetic data we start from the ground consensus model and show that we can recover it accurately from simulated experts' labels that may differ because of the expert-specific parameters. Second, we show benefits of our approach on HIT data.

Chapter 5 outlines our contributions and open research questions.

Finally, I would like to note that parts of this thesis have been previously published in [Nguyen et al., 2011a], [Nguyen et al., 2011b], [Nguyen et al., 2013], [Valizadegan et al., 2012], [Valizadegan et al., 2013].

2.0 BACKGROUND

This chapter outlines background and relevant research for methods we describe in this thesis. We start with the basics of supervised learning and classification methods, then discuss label efficient learning and related works. We use the following notation throughout this document: matrices are denoted by capital letters, vectors by boldface lower case letters and scalars by normal lower case letters.

2.1 SUPERVISED CLASSIFICATION LEARNING

Supervised learning is a sub-field of machine learning, where the task is to learn a mapping from input examples to desired output targets. In the standard supervised setting, training data consist of examples and corresponding labels (targets), which are given by a teacher (labeller). The goal is to learn a model that can accurately predict labels of future unseen examples. Formally, given training data $D = \{\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_N\}$ where $\mathbf{d}_i$ is a pair of $(\mathbf{x}_i, y_i)$, $\mathbf{x}_i$ is an input feature vector, y_i is a desired output given by a teacher, the objective is to learn a mapping function $f : X \rightarrow Y$ such that for a new future example $\mathbf{x}_{new}$, $f(\mathbf{x}_{new}) \approx y_{new}$. If desired outputs are continuous values then the learning problem is **regression**, otherwise the learning problem is **classification** if desired outputs are discrete.

In this thesis we focus on the classification learning, which has many applications in practice, for example, given historical clinical data, predict whether a (future) patient has disease or not, or given a text database, classify a new text document to one of possible topics (news, sport, etc.). In general, there may be any number of desired class outputs, however, the most frequently researched problem is binary classification, where outputs belong to

two classes, e.g. a disease is detected/not detected.

The exact form of the model $f : X \rightarrow Y$, and the algorithms used to learn it, can take on different forms. For example, the model can be based on linear discriminant analysis (LDA) [Fisher, 1936], logistic regression [McCullagh and Nelder, 1989], maximum margin (support vector machines) [Cortes and Vapnik, 1995], probabilistic models such a Naive Bayes model [Domingos and Pazzani, 1997] or a Bayesian belief network [Pearl, 1985], and classification trees [Breiman et al., 1984]. In addition, there are various ensemble methods, such as bagging [Breiman, 1996] and boosting [Schapire, 1990], where multiple individual (weak) learners are combined together to create a strong one.

In this chapter, we describe Logistic Regression [McCullagh and Nelder, 1989] and Maximum Margin [Cortes and Vapnik, 1995] methods in more details. These are the two of the most widely used baselines in classification learning research. Moreover, many methods mentioned in this thesis, including some of ours, are based on these methods, so a review of them would be useful.

2.1.1 Logistic regression

We want to learn a classification function $f : X \rightarrow Y$, that maps input (features) X to one of the class labels $\{0, 1..k\}$ in Y. One way to do this is to define a function $g_i(\mathbf{x}) : X \rightarrow R$ for each class $i \in \{0, 1, .., k\}$, then classify an input example $\mathbf{x}$ to the class with the highest value of $g_i(\mathbf{x})$, i.e. $f(\mathbf{x}) = \text{argmax}_{i \in \{0..k\}} g_i(\mathbf{x})$. Intuitively, function $g_i(\mathbf{x})$ represents a kind of class membership, and an example belongs to the class for which it has the highest membership value.

For illustration let us consider the binary classification case, i.e. $Y \in \{0, 1\}$. Figure 1 illustrates the input space with positive and negative examples and the decision boundary defined by $g_i(\mathbf{x})$, $i \in \{0, 1\}$. If $g_1(\mathbf{x}) > g_0(\mathbf{x})$ then $\mathbf{x}$ belongs to class 1 (positive), otherwise it belongs to class 0 (negative). The decision boundary is defined when $g_1(\mathbf{x}) = g_0(\mathbf{x})$, e.g. membership values are equal.

Function g_i can be designed in different ways. The logistic regression [McCullagh and

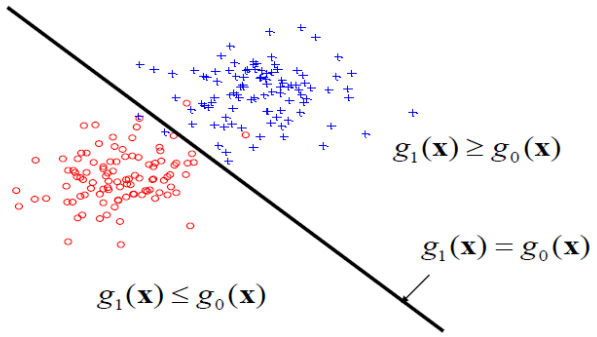

Figure 1: Discriminant functions and decision boundary.

Nelder, 1989], one of the most commonly used models, defines g_1 and g_0 as follows:

$$\begin{aligned} g_1(\mathbf{x}) &= \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}} = p(y=1|\mathbf{x},\mathbf{w}) \\ g_0(\mathbf{x}) &= p(y=0|\mathbf{x},\mathbf{w}) = 1 - p(y=1|\mathbf{x},\mathbf{w}) \end{aligned}$$

An example $\mathbf{x}$ will be assigned to class 1 if $g_1(\mathbf{x}) \geq 0.5$, and class 0 otherwise.

The main idea of logistic regression is to obtain a probabilistic interpretation of class membership by transforming a linear combination of input vector $\mathbf{x}$, i.e. $\mathbf{w}^T\mathbf{x}$, into a probabilistic value. The logistic function $\frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}}$ is useful because it can take any input $\mathbf{w}^T\mathbf{x}$ from $(-\infty, +\infty)$ range and transforms it into an output in (0,1) range.

2.1.1.1 Learning the logistic regression model Let $D_i = (\mathbf{x}_i, y_i)$ denotes the set of input examples $\mathbf{x}_i$ and labels y_i, and $\mu_i = p(y_i = 1|\mathbf{x}_i, \mathbf{w}) = g(\mathbf{w}^T\mathbf{x})$. We learn logistic regression model by maximizing the likelihood of data:

$$\mathbf{w}^* = \text{argmax}_{\mathbf{w}} L(D, \mathbf{w})$$

where

$$\begin{aligned} L(D,\mathbf{w}) &= \prod_{i=1}^{N} p(y=y_i|\mathbf{x}_i,\mathbf{w}) \\ &= \prod_{i=1}^{N} \mu_i^{y_i}(1-\mu_i)^{1-y_i} \end{aligned}$$

This is equivalent to maximizing the log-likelihood of data since the optimal weights are the same for both likelihood and log-likelihood:

$$\begin{aligned}
\mathbf{w}^* &= \text{argmax}_{\mathbf{w}}[\log L(D,\mathbf{w})] \\
&= \text{argmax}_{\mathbf{w}} l(D,\mathbf{w}) \\
&= \text{argmin}_{\mathbf{w}} - l(D,\mathbf{w}) \\
&= \text{argmin}_{\mathbf{w}} - \log \prod_{i=1}^{N} p(y = y_i|\mathbf{x}_i,\mathbf{w}) \\
&= \text{argmin}_{\mathbf{w}} - \log \prod_{i=1}^{N} \mu_i^{y_i}(1-\mu_i)^{1-y_i} \\
&= \text{argmin}_{\mathbf{w}} - \sum_{i=1}^{N} y_i \log \mu_i + (1-y_i)\log(1-\mu_i)
\end{aligned} \tag{2.1}$$

We can solve the optimization problem $\max_{\mathbf{w}} l(D,\mathbf{w}) = \min_{\mathbf{w}} -l(D,\mathbf{w})$ by taking the derivatives and updating the weight vector $\mathbf{w}$ by some scheme, for example, gradient descent. More specifically, the gradient of the log-likelihood of data for the logistic regression model becomes:

$$\nabla_{\mathbf{w}} - l(D,\mathbf{w}) = \sum_{i=1}^{N} -\mathbf{x}_i(y_i - f(\mathbf{w},\mathbf{x}_i))$$

The gradient descent procedure can be implemented by iteratively updating weights as:

$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} + \alpha(k) \sum_{i=1}^{N} -\mathbf{x}_i(y_i - f(\mathbf{w},\mathbf{x}_i))$$

where k indicates the k-th step of the updating process and $\alpha(k)$ is the learning rate scaling the iterative updates.

In this section we gave a brief review of logistic regression, more details about theory and analysis can be found in [McCullagh and Nelder, 1989].

2.1.1.2 Regularization Over-fitting problem is a problem in which a learner achieves high performance on training, but poor performance on test data. It can arise when the dimensionality of $\mathbf{x}$ is high while the number of training examples N is small. We can reduce the over-fitting effect by using one of the regularization approaches, such as the ridge (or L_2) regularization [Hoerl and Kennard, 1981], the lasso (or L_1) regularization [Tibshirani, 1996, Friedman, 2010], or their elastic network combination [Zou and Hastie, 2005]. Using regularization, the optimization in Equation 2.1 is modified to:

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}} Loss(D, \mathbf{w}) + Q(\mathbf{w})$$

where $Loss(D, \mathbf{w}) = -\log L(D, \mathbf{w})$ for logistic regression and $Q(\mathbf{w})$ is a regularization penalty. Examples of regularization penalties are: $Q(\mathbf{w}) = \lambda||\mathbf{w}||_1 = \lambda \sum_{j=1}^{d} |\mathbf{w}_j|$ for the L1 (lasso) regularization, or $Q(\mathbf{w}) = \lambda||\mathbf{w}||_2 = \lambda(\sum_{j=1}^{d} \mathbf{w}_j^2)^{\frac{1}{2}}$ for the L2 (ridge) regularization. λ is a constant that scales the regularization penalty. Typically this constant is optimized using the internal cross-validation approach.

2.1.2 Maximum Margin method

The main idea of the Maximum Margin method for classification is to find the decision hyper-plane that maximizes the margin between examples of the two classes. "Margin" is defined as the distance from the closest examples to the decision hyper-plane. The intuition is that among all possible decisions, the max-margin decision has the best generalization ability. In other words, it has the best chance to classify a future example correctly. This intuition was proved to be true. In fact, the idea has a strong foundation in statistical learning theory: [Vapnik, 1995] proved that the bound on generalization error is minimized by maximizing the margin.

Figure 2 illustrates this idea. In Figure 2-left positive and negative examples can be perfectly separated by many linear decision boundaries. However, as argued by [Vapnik, 1995] and [Cortes and Vapnik, 1995], the optimal solution is the decision boundary that maximizes the margin between positive and negative examples (Figure 2-right).

Note that the decision hyper-plane is determined only by the examples on the margin hyper-planes (circled points in Figure 2-right). Hence, these examples are called "support

vectors". In machine learning literature, Maximum Margin method for classification is often referred by the term "Support Vector Machines".

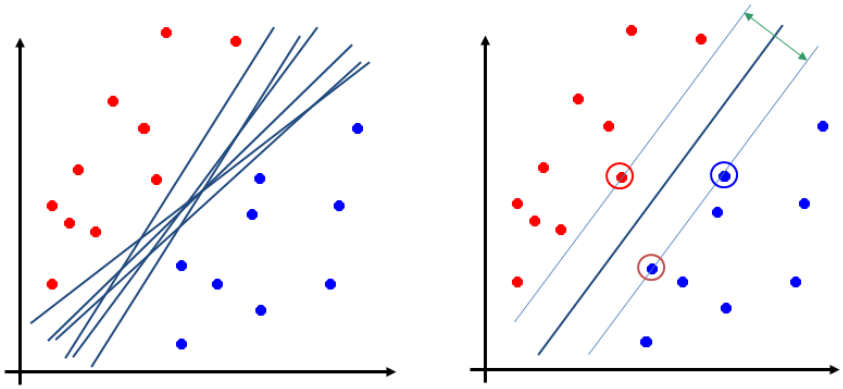

Figure 2: Maximum Margin (Support Vector Machines) idea. Left: many possible decisions; Right: maximum margin decision.

We briefly describe mathematical formulations of Linear SVM and Kernel SVM in the following sections.

2.1.2.1 Linear Support Vector Machines Let start with a simple case, when data are linearly separable. Figure 2 illustrates a 2-d example of this case.

Linear SVM can be formulated by the following constrained optimization problem:

$$\begin{aligned} &\min_{\mathbf{w},b} && Q(\mathbf{w}) \\ &\text{subject to:} && \\ &\forall i = 1..N: && y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \end{aligned}$$

where N is the number of examples in the training data, $\mathbf{w}$ is the weight vector - the model to be learned. $\mathbf{w}$ defines the direction of the decision boundary. b is the bias term, which defines the shift of the boundary. $\mathbf{x}_i$ and $y_i \in \{1, -1\}$ are feature vector and label, respectively, of example i. $Q(\mathbf{w})$ is a regularization function (Section 2.1.1.2), which is typically written in L2 norm in machine learning literature, but in general can be in L1 norm. For classification, a new example $\mathbf{x}$ is assigned "1" (positive) if $(\mathbf{w}^T\mathbf{x} + b) > 0$, otherwise "-1" (negative).

The above SVM formulation is called Hard-margin SVM, because it requires all examples of the two classes to be linearly separable. However, in practice, it is often impossible

to separate data perfectly with a linear boundary, as shown in Figure 3. To handle this case, we relax the above requirement by allowing SVM to make mistakes, but mistakes are penalized in the objective function. We have the following formulation of Soft-margin SVM, also called the **primal form** of (Soft) SVM:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & Q(\mathbf{w}) + C\sum_i \xi_i \\ \text{subject to:} & \\ \forall i = 1..N: \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{2.2}$$

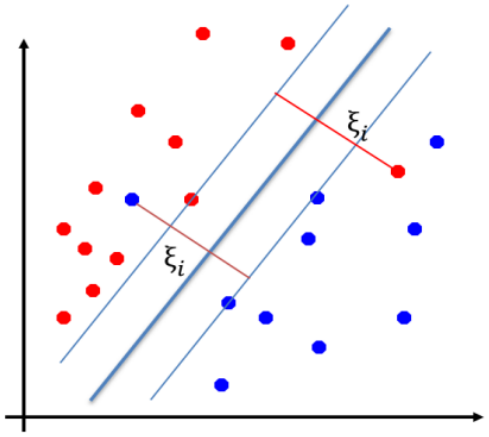

Figure 3: Soft-Margin SVM for the linearly non-separable case. Slack variables ξ_i represent distances between examples $\mathbf{x}_i$ and margin hyper-planes.

Slack variables ξ_i represent distances between examples $\mathbf{x}_i$ and margin hyper-planes. Note that $\xi_i = 0$ if $\mathbf{x}_i$ is located on the correct side of the margins, otherwise $\xi_i > 0$. $\xi_i = \max(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))$ is called the hinge loss. Constant C is a trade-off parameter that defines how much misclassified examples should be penalized. In fact, Hard-margin SVM is a special case of Soft-margin SVM with C set to infinity. Therefore, further in this document, the term "Support Vector Machines" refers to Soft-margin SVM.

Both Hard and Soft-margin formulations are convex optimization problems, which means that any local optimum is also the global optimum. This property is very important because it indicates that if we can find a best local solution, we are guaranteed to have the best

global solution. This is not the case for many other classification methods (logistic regression, neural networks, etc.), where we may be "trapped" in local optima and never find the global optimum.

2.1.2.2 Kernel Support Vector Machines Linear SVM with soft margins is a powerful tool when the non-separability is caused by a small number of (noisy) examples. However, if data are highly non-linear and are not separable by a linear boundary, e.g. data shown in Figure 4-left, then Linear SVM may not perform well. Kernel SVM was designed to solve this problem. The idea is to map features from the original space to a new higher dimensional space, where linear relations may exist. Figure 4 illustrates this idea: Figure 4-left shows positive and negative examples that cannot be separable in the 2-d space; Figure 4-right shows that mapping φ of input data from the original 2-d space to a 3-d space may introduce a linear boundary that can separate examples of two classes (in this case the linear boundary is a surface).

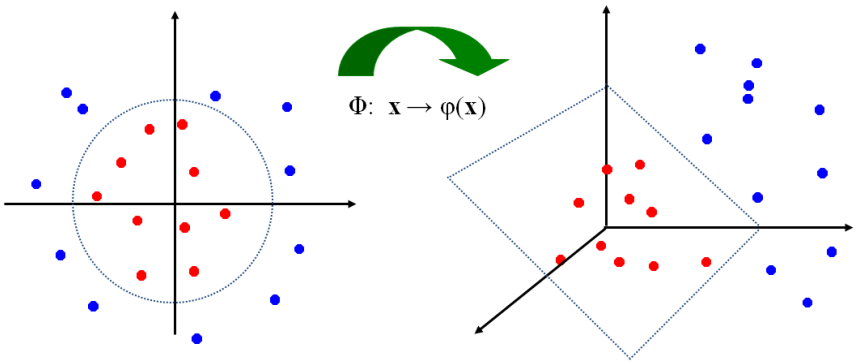

Figure 4: Kernel SVM idea. Left: original 2-d input space, positive and negative examples are not linearly separable; Right: function φ mapping original input space to a higher-dimensional (3-d) feature space, where positive and negatives can be linearly separable.

Solving the optimization problem 2.2 in the feature space is equivalent to solving the optimization of the following Lagrangian function:

$$\min_{\mathbf{w},b} \quad L(\mathbf{w},b,\mathbf{a}) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{N} a_i(y_i(\mathrm{w}^T\varphi(\mathbf{x}_i)+b)-1)$$

where $\mathbf{a} = (a_1, ..., a_N)^T$ is the vector of Lagrangian multipliers. Note that for the demonstration purpose we use L2 norm regularization $||\mathbf{w}||_2$, which is widely used in the machine learning literature.

Setting the derivatives of $L(\mathbf{w}, b, \mathbf{a})$ with respect to $\mathbf{w}$ and b equal to 0, we obtain the following two conditions:

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{N} a_i y_i \varphi(\mathbf{x}_i) \\ 0 &= \sum_{i=1}^{N} a_i y_i \end{aligned}$$

Plugging these conditions into $L(\mathbf{w}, b, \mathbf{a})$ gives the **dual form** of the maximum margin problem:

$$\begin{aligned} \max_{a_1..a_N} \quad & \sum_{i=1}^{N} a_i - \frac{1}{2} \sum_{i,j=1}^{N} a_i a_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to:} & \\ \forall i = 1..N: \quad & 0 \le a_i \le C \\ & \sum_{i=1}^{N} a_i y_i = 0 \end{aligned}$$

where $k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x})^T \varphi(\mathbf{x}')$ is a kernel function. For Linear SVM, $k(\mathbf{x}_i, \mathbf{x}_j)$ is the dot product of $\mathbf{x}_i$ and $\mathbf{x}_j$: $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$.

Solving constrained optimization problems in high dimensional spaces is difficult and computationally expensive. Therefore, kernel functions k should be designed so that SVM: (1) has the representation power of high dimensional spaces and (2) still be computationally efficient. This can be done by choosing a mapping from the input space I to a new feature space F: $\mathbf{x} \longrightarrow \varphi(\mathbf{x})$, such that $k(\mathbf{x}, \mathbf{x}') = \varphi(\mathbf{x}) \cdot \varphi(\mathbf{x}') \ \forall \mathbf{x}, \mathbf{x}' \in I$. Thus, we implicitly compute dot product in a high dimensional space F, in terms of operations in the original low dimensional space I. This is called the "kernel trick".

Many different types of kernels have been designed by the research community. For example, the two most widely used kernels are polynomial and radial basis functions (RBF):

- Polynomial-p: $k(\mathbf{x}, \mathbf{x}') = (c + \mathbf{x} \cdot \mathbf{x}')^p, p \in N, c \ge 0$

- RBF: $k(\mathbf{x}, \mathbf{x}') = exp(\frac{-||\mathbf{x}-\mathbf{x}'||^2}{2\sigma^2})$

In this section we gave a brief review of Support Vector Machines. More details about theory and analysis of SVM can be found in [Cortes and Vapnik, 1995] and [Bishop, 2006].

2.2 RELATED WORK FOR LABEL EFFICIENT LEARNING

By definition supervised learning systems rely on labels given in the training data, and in practice, they often must be trained on a large number of labelled examples in order to perform well. However, as mentioned in the Introduction chapter, the process of labeling examples using subjective human assessments faces two problems: (1) it can be extremely time-consuming and costly, which results in a limited number of labeled examples, and (2) in many cases labels are given by not only one but multiple annotators, which may introduce disagreements/contradictions due to differences in knowledge, opinions or biases. Since supervised learning methods rely on labeled examples, we need to find approaches to obtain more useful information (labels) with lower cost and utilize them efficiently. Again, in this thesis, we focus on classification learning, where our goal is to build classification models that can learn with smaller training data and make more accurate prediction on future unseen examples.

In this section we give an overview of research works that are relevant to our solutions for above problems. In Section 2.2.1 we review active learning - a sub-field of machine learning that aims to reduce labeling cost by selecting the most informative examples. In Section 2.2.2 we review transfer learning that aims to reducing labeling cost by transferring useful information from one domain/task to another domain/task. Then we give an overview of our alternative approach - learning with auxiliary soft labels and its relevant research. In Section 2.2.4 we summarize works in multi-annotator learning, where the two main objectives are estimating the "true" labels and learning a consensus model, given information collected from multiple annotators.

2.2.1 Active learning

Active Learning is a sub-field of (supervised) machine learning, where the primary goal is to reduce the cost of labeling examples. Active learning has been explored extensively by the data mining and machine learning communities in recent years. In traditional "passive" learning, the learner randomly picks examples from the database and requests labels for them. In contrast, active learning only requests labels for the most informative examples - ones that help to increase the performance of the current learning model. Intuitively, this may reduce the number of examples to be labeled and accelerate the learning process.

There are two common query scenarios: stream-based [Cohn et al., 1994] and pool-based [Lewis and Gale, 1994]. In the stream-based scenario, data come in a stream, one example at a time, the active learner set an "informativeness" threshold and decides whether to query for label or discard this example based on that threshold. In the pool-based scenario, the active learner has access to a pool (subset) of all unlabeled examples. It inspects examples in the pool and selects the k most informative examples to query for labels. While pool-based scenario is used much more common in practice, stream-based scenario is more appropriate in the case when limited processing power does not allow us to scan and inspect a pool of examples (e.g. mobile applications).

Figure 5 illustrates how (pool-based) active learning works. An active learner recursively performs three steps: (1) inspects unlabeled examples; (2) selects the most k informative examples and requests an annotator to label them, and (3) retrains the current learning model with the new set of labeled examples. This process is repeated until some stopping criteria is met, for example, N examples have been labeled, or the performance the model has reached some satisfied threshold.

In all the scenarios, the active learner needs to select example(s) based on some "informativeness" criteria. Different strategies to define the "informativeness" have been proposed. We summarize the most popular ones in the following paragraphs.

Uncertainty sampling [Lewis and Gale, 1994] is the simplest and most widely used query strategy. It queries the example that the current model predicts with the lowest confidence. For binary classification, uncertainty sampling selects the example that has predictive prob-

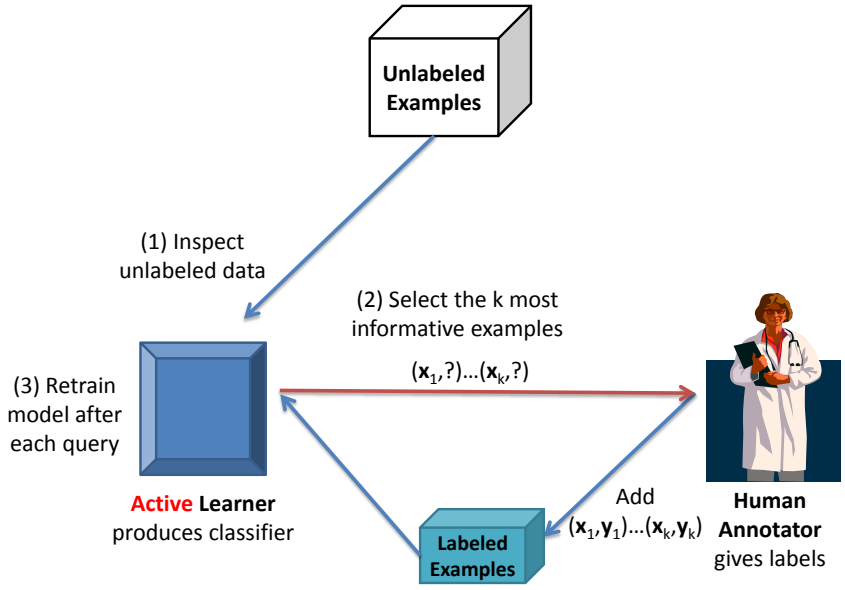

Figure 5: Active Learning scheme.

ability closest to 0.5. The idea is intuitive: if the current model can predict an example with high confidence then this example does not carry much information to improve the model; otherwise the example can benefit the model. For regression task, the learner queries the example for which the current learning model has the highest output variance in its prediction (e.g. least confidence).

Another popular strategy is query-by-committee (QBC) [Seung et al., 1992]. In QBC, one first constructs a committee of different models, which are all trained on the current labeled training data, but represent competing hypotheses. Then, committee members vote on the labeling of query candidates. The most informative example to be queried is the one that the committee members most disagree on. The construction of committee may be done in different ways. For example, the approach introduced by [Seung et al., 1992] simply samples randomly two hypotheses that are consistent with the current labeled training examples. Another approach is query-by-bagging [Abe and Mamitsuka, 1998]. It repeatedly samples subsets of labeled instances (using bagging [Breiman, 1996]) and trains committee models on them. The size of committee may vary, however previous works have shown that even a small committee (size two or three) could work well in practice ([Settles, 2010], [Seung et al., 1992], [Settles and Craven, 2008]).

Expected model change is the strategy that queries examples which cause the largest

change to the current model if we knew their labels. The intuition is that if an example changes the current learning model significantly then it likely carries much information and can make great impact on the learning process. One example of the expected model change strategy is the method by [Tong and Koller, 2000], where examples are selected to maximize the Kullback-Leibler divergence between the new posterior distribution obtained after training with the new queried label and the current posterior distribution (before the query). Another example of this query strategy is the "expected gradient length" method introduced by [Settles et al., 2008], where the model change was defined as the Euclidean length of the training gradient (the vector used to estimate parameter values during the optimization process). The disadvantage of the expected model change strategy is that "informativeness" can be over-estimated, for example, gradient may be too large if some parameters become large during the optimization. Therefore, some techniques, e.g. parameter regularization, need to be used to alleviate this problem.

Expected error reduction strategy was first proposed by [Roy and McCallum,]. It aims to directly reduce the generalization error of the learning model. The idea is that, for each example $\mathbf{x}_i$ in the unlabeled pool, the learning algorithm estimates its label and builds a model over the combination of $\mathbf{x}_i$ and the training labeled examples. Among them, the example that minimizes the generalization error is selected to query for label. The disadvantage of this strategy is the expensive computational cost.

Variance reduction strategy [Geman et al., 1992] was originally designed for regression task, but could be generalized for classification task. This strategy queries example that would minimize the prediction variance of the current model. The intuition is that the generalization error can be indirectly reduced by minimizing output variance. To use this strategy one must have a way to approximate the output variance, which is not a trivial problem, and also depends on specific learning models. Closed-form approximations of output variance were derived for Gaussian random fields [Cressie, 1993] and Neural networks [MacKay, 1992] .

2.2.2 Transfer learning

Transfer learning is an emerging field in machine learning research, that aims to improve the sample complexity of learning problems with the help of additional information and knowledge sources. The main idea is to transfer useful knowledge/information learned in one domain or task (source) to another (target) domain or task. This would help to increase the amount of information that the target learner can learn form, which leads to improved predictive performance. This is not a trivial problem because we need to find out what information is useful and beneficial for the target learner, and how to transfer that information.

2.2.2.1 Core concepts For the discussion of transfer learning we need to give the definitions of the core concepts: domain, task and transfer learning. For illustration let us consider an application example: disease diagnosis, where given patient health records, the goal is to classify them into certain classes.

Domain, denoted by $\mathscr{D}$, consists of two components: the feature space X and the marginal distribution $P(X)$.

Task, denoted by $\mathscr{T}$, consists of two components: the label space Y and a predictive function f that maps the feature space to the label space: $f : X \rightarrow Y$. From probabilistic point of view, f is the conditional probability $P(Y|X)$. The function f needs to be learned from the training data, and can be used to predict the label of a future example.

Source domain and **source task** are denoted by $\mathscr{D}_S$ and $\mathscr{T}_S$, whereas **target domain** and **target task** are denoted by $\mathscr{D}_T$ and $\mathscr{T}_T$, respectively.

Transfer Learning is a learning technique that aims to improve the predictive performance of the function f_T in $\mathscr{D}_T$ by using the information and knowledge in $\mathscr{D}_S$ and $\mathscr{T}_S$, where $\mathscr{D}_S \neq \mathscr{D}_T$ and/or $\mathscr{T}_S \neq \mathscr{T}_T$.

Given $\mathscr{D}_S$, $\mathscr{T}_S$, $\mathscr{D}_T$ and $\mathscr{T}_T$, we may have the following cases:

- The case when $\mathscr{D}_S = \mathscr{D}_T$ and $\mathscr{T}_S = \mathscr{T}_T$. This is the traditional classification learning, where we have one domain and one task.
- The case when $\mathscr{D}_S \neq \mathscr{D}_T$. In this case either (1) the feature spaces in source and target domains are different, e.g. patients have different feature vectors representing different

bio characteristics, or (2) the feature space is the same but feature distributions are different: $P(X_S) \neq P(X_T)$, e.g. patient data in source and target domains focus on different patient demographics (different countries, different age groups, etc.)

- The case when $\mathcal{T}_S \neq \mathcal{T}_T$. In this case we have either (1) the label spaces in source and target domains are different, e.g. there are two classes in the source domain and five classes in the target domains, or (2) the label space is the same but the label distributions are different: $P(Y_S|X_S) \neq P(Y_T|X_T)$, e.g. the source and the target set of patient records have very different ratios of positive/negative examples.
- The case when $\mathcal{D}_S \neq \mathcal{D}_T$ and $\mathcal{T}_S \neq \mathcal{T}_T$. This is the combination of the previous two cases.

2.2.2.2 Types of information to be transferred The most important question of transfer learning is "What to transfer ?". According to [Pan and Yang, 2010], there can be four different types of information to be transferred from the source domain/task to the target domain/task:

- **Instance transfer**. The main idea of this approach is to transfer a set of labeled training instances from the source data to the target data. The predictive performance is expected to increase because the target learner has acquired more labeled data to learn from. The source data usually cannot be used directly for the target task, so typically some re-weighting technique is developed to assign the weights or importance of the transferred data instances in the new domain/task. Some related works in this area are [Dai et al., 2007], [Jiang and Zhai, 2007], [Zadrozny, 2004].
- **Feature representation transfer**. The main idea of this approach is to utilize information from the source and target domains to learn a good feature representation that reduces the difference between these domains and decrease error rates. The predictive performance is expected to increase with the new (better) feature set. Some related works are [Argyriou et al., 2007a], [Lee et al., 2007], [Ruckert and Kramer, 2008]. A representative work is [Argyriou et al., 2007a], where the authors proposed to learn a low-dimensional feature representation that is shared between source and target tasks.
- **Parameter transfer**. The main idea of this approach is to assume that the source and target tasks have some shared parameters or prior distributions of hyper-parameters

and the approach aims to increase the performance of the target learner by exploiting these parameters. Some related works for the parameter transfer approach are ([Evgeniou and Pontil, 2004], [Gao et al., 2008], [Lawrence and Platt, 2004]). In Section 2.2.2.3 we will give more details about [Evgeniou and Pontil, 2004] - a representative work in this area.

- **Relation transfer**. The main idea of this approach is to assume that some relations among the data in the source and target domains are similar. Therefore, the learner in the target domain can benefit by transferring these relations from the source domain. Methods for relation transfer typically use some statistical learning technique (e.g. Markov Logic network) to transfer relation in data from the source to the target domain. Some related works in this area are ([Mihalkova et al., 2007], [Davis and Domingos, 2009]). Note that data are usually in relational domains (e.g. entities are predicates and their relations are in first-order logic), and not assumed to be independent and identically distributed (i.i.d).

2.2.2.3 Multitask learning Multitask learning is a special type of transfer learning. The reason we have a special interest in multitask learning is that it motivated our solution for the multi-annotator learning problem (Section 2.2.4 and Chapter 4). The main differences between multitask learning and (general) transfer learning are:

- In multitask learning we have many different tasks and the domain of all tasks is the same, whereas in transfer learning the relations between tasks and between domains could be in any combination (Section 2.2.2.1).
- While transfer learning focuses on improving the predictive performance of the target learner, multitask learning learns all tasks simultaneously and aims to improve the performance of all involved learners.

The main idea of multitask learning is that it assumes involving tasks are different but related, i.e. they share some related information that can help to improve the learning of all tasks simultaneously. Related works in this area aim to discover this shared information and utilize them to learn the tasks. For example, [Caruana, 1997], [Silver, 2001] transfer

the information among tasks through the shared hidden layer nodes in Neural Networks. [Yu et al., 2005], [Argyriou et al., 2007b], [Evgeniou and Pontil, 2004] transfer information through shared parameters of task-specific models.

A representative work in this area is [Evgeniou and Pontil, 2004], where the authors proposed a SVM-based method for multitask learning. The idea is to separate the weight vector $\mathbf{w}$ to be learned by the SVM into two components: a shared weight vector that is common for all tasks; and a task specific weight vector, one for each task:

$$\mathbf{w}_t = \mathbf{w}_0 + \mathbf{v}_t, \quad t \in \{1..T\}$$

where T is the number of tasks, $\mathbf{w}_0$ is the shared common weight vector, $\mathbf{w}_t$ is the weight vector for task t and $\mathbf{v}_t$ is the difference between $\mathbf{w}_0$ and $\mathbf{v}_t$.

$\mathbf{w}_0$ and $\mathbf{w}_t$ are incorporated into a single optimization problem and learned simultaneously during the optimization process:

$$\begin{aligned} \min_{\mathbf{w}_0, \mathbf{v}_T, \xi_{it}} \quad & \sum_{t=1}^{T}\sum_{i=1}^{m} \xi_{it} \quad + \quad \frac{\lambda_1}{T}\sum_{t=1}^{T} \|\mathbf{v}_t\|^2 + \lambda_2 \|\mathbf{w}_0\|^2 \\ \text{s.t.} \quad & \forall i \in \{1,2,...,m\} \quad \text{and} \quad t \in \{1,2,...,T\}: \\ & y_{it}(\mathbf{w}_0 + \mathbf{v}_t) \cdot \mathbf{x}_{it} \quad \geq \quad 1 - \xi_{it}, \\ & \xi_{it} \quad \geq \quad 0 \end{aligned}$$

where m is the number of training examples, ξ_{it} are slack variables that penalize misclassification errors, λ_1 and λ_2 are regularization constants and $\mathbf{x}_{it}$ are feature vectors.

Note that we adopt the idea of learning the common and specific tasks for solving the multi-annotator learning problem (Section 2.2.4). Our approach learns a common consensus model and annotator-specific models simultaneously. However, we go beyond the Evgeniou and Pontil's multitask learning approach by also incorporating the consistency and bias of different annotators into the optimization process. Details of our multi-annotator learning framework are described in Chapter 4.

2.2.3 Learning with auxiliary soft labels

We have done an overview of two popular label-efficient learning approaches: active learning and transfer learning. In this work we study an alternative approach: enrichment of labeled instances using auxiliary soft label and its incorporation into the classification learning. The idea is simple: we ask a human expert annotator to provide us, in addition to class label, also auxiliary soft label, reflecting his belief/certainty in the class label. This auxiliary information can be obtained with little extra cost because most of the time was already spent on example assessment and making class decision. Now, for each example, instead of having only class label, we have both class and soft labels. The question is how to efficiently utilize both kinds of labels to improve classification learning.

This problem is different than active learning: while active learning aims to select informative examples to label, we aim to obtain more useful information from those examples that are selected. This problem is also different than transfer learning because we work in the same domain, with the same data set, the auxiliary information is (always) directly relevant and available.

Since there are two kinds of labels available, this problem has close relationships to both classification, regression, and also preference learning. Previous works in the machine learning and AI community focused on one of these, but not their combination. For example, classification algorithms (e.g. logistic regression, SVM) would use only class labels. [Smyth, 1995] used probabilistic labels to learn classification of volcanoes from radar images of distant planets. They used a simple Gaussian model and relied only on the probabilistic information to learn the classifier. In preference/rank learning [Fürnkranz and Hüllermeier, 2010], the learner would rely only on the soft label as preference/ranking score and learns a ranking function that predicts this score. The final classification is made based on a threshold of the score. For example, SVMRank ([Herbrich et al., 1999], [Joachims, 2002]) learns a ranking function that satisfy pairwise order constraints on every pair of examples in the target rank.

In this work we investigate ways to improve existing algorithms by incorporating both class and auxiliary soft labels. Moreover, we also study the problem of noise in the soft

labels. Notice that most existing classification/regression/ranking algorithms assume that labels are reliable (golden standard). However, as we will see in Chapter 3, human subjective assessments can be noisy, which make the learning process more difficult.

2.2.4 Learning with multiple annotators

In traditional supervised learning, examples are assumed to be annotated by a single annotator - oracle, who gives ground truth labels . In contrast, in multiple-annotator learning, each example may be labeled by one or many annotators, who are not assumed to give ground truth labels. There are many application scenarios for multi-annotator learning, including, but not limited to, the following:

- **Each example is labeled by a large number of annotators**. In this scenario, the goal is to obtain a high-quality label for each example and the idea is, instead of acquiring the label from a reliable but expensive expert, one would acquire multiple labels from many (perhaps unreliable) annotators and try to come up with a consensus "true" label by some voting scheme. This scenario was motivated by the emerge of crowd-souring services (e.g. Amazon Mechanical Turk), where labels can be obtained at very low costs by online workers.
- **Different annotators label non-overlapping set of examples**. This is the scenario where each example in the training data is labeled by only one annotator, but overall there is more than one annotator labeling the training set. The goal is to distribute labeling efforts to different annotators. This scenario is motivated by the fact that in many domains (e.g. medical domain), the labeling task may be costly, time-consuming and tedious, so it is not expected that one annotator can label all examples. Note that in this scenario, the labeling task is typically very complicated and a high level of domain expertise is required.
- **Different annotators label overlapping sets of examples**. This scenario is in between of the above two scenarios, where each example can be labeled by only one or multiple annotators. This scenario may appear where, by the nature of the application, we may have some examples labeled by one and some labeled by many people, e.g. some

patients may be examined by one or several physicians, or some products may be rated by one or many consumers. In this case, the goal could include not only learning the consensus label or model, but also exploring the relations between different annotators through the labeling process.

Most of the current research focuses on the first case - crowd-sourcing application. However, note that in practice, there is no clear separation between the application scenarios, so methods designed for one application can be applied for another application.

So far the most commonly used approach in multi-annotator learning is the majority vote. For each example $i \in \{1 \dots N\}$ from the set of N examples, the "true" label z_i is estimated by voting: $z_i = 1$ if $\sum_{k=1}^{m} y_i^k \geq 0$, otherwise $z_i = -1$, where y_i^k is the label of example i given by annotator $k \in \{1 \dots K\}$. The majority vote is illustrated in Figure 6. In this figure z_i denotes the (true) consensus label and y_i^k denotes the label of an example provided by an annotator (in total there are N examples and K different annotators).

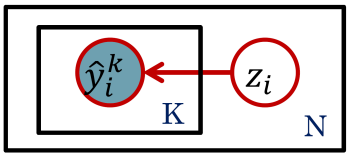

Figure 6: Graphical model for the majority vote method. N examples labeled by K annotators.

Note that majority vote focuses on estimating the "true" labels, it learns neither a consensus model nor annotator-specific models. Nevertheless, the data set D = $\{(\mathbf{x}_i, z_i)\}_{i=1}^{N}$ generated by majority vote can be used to train a consensus model later.

The majority vote comes with two drawbacks. First, it assumes that all annotators are equally reliable when labeling examples. However, when one reviewer is very reliable and the other ones are unreliable, the majority vote may sway the final labels and assign incorrect labels to some examples. Second, tight votes are ignored when learning the majority-based consensus model. The biggest advantage of majority vote is simplicity, which explains why it is being used in most practical applications, and as a common baseline to compare with more advanced methods in the field.

The research of more advanced techniques than the majority vote has been encouraged recently by the growing number of crowd-souring services. Using these services, one can hire many annotators to repeatedly label examples at low cost. [Sheng et al., 2008, Snow et al., 2008] showed that repeated labeling could help to learn better classification models. However, like the majority vote, they assume that all annotators have the same reliability, which is not true in practice. Other methods have been developed to address this limitation. In general, they can be divided into two main directions, depending on the primary goal of the learning process:

- **Learning the "true" label** which represents labels given by multiple annotators. This label can be used later to learn a predictive model.
- **Learning a consensus model** that is representative for models of different annotators, and this consensus model can be applied directly to predict future examples.

In the following sections, we give a brief overview of works in these two directions.

2.2.4.1 Learning the "true" label. This research direction is mainly motivated by the crowd-sourcing applications. In this line of research, [Dawid and Skene, 1979] is the pioneer work that serves as the foundation for most other works. The Dawid method is illustrated in Figure 7. In this figure z_i denotes the (true) consensus label and y_i^ks denotes the label of an example provided by an annotator (overall, there are N examples labeled by K annotators). These are the same as for the majority vote model (see Figure 6). In addition to these, Dawid's method adds a set of hidden variables π_k that represents the quality of reviews provided by each annotator. These extra variables are the elements of the contingency table for each annotator, i.e. true positive, false positive, true negative, and false positive rates. Dawid & Skene proposed an EM algorithm to learn the elements of the contingency table π_ks (M step) and the consensus labels z_is (E step).

Many works in this direction were based on this framework. [Smyth et al., 1995] used a similar method for estimating ground truth from subjective labels of Venus images. [Whitehill et al., 2009] extended the framework by modeling difficulty of examples. [Donmez et al., 2009] estimated the confidence interval for the reliability of each annotator.

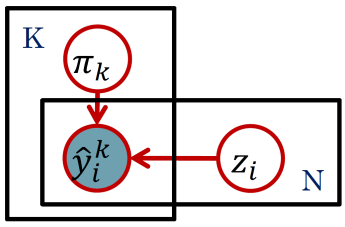

Figure 7: Graphical model for Dawid method [Dawid and Skene, 1979]. N examples labeled by K annotators.

The current state-of-the-art method is [Welinder et al., 2010], which models both annotator bias and example difficulty. Figure 8 shows the graphical model for this method. The Welinder method assumes that each annotator has their own version $\hat{\mathbf{x}}_i^k$ of the feature vector $\mathbf{x}_i$ generated from the class distribution and then utilizes the model $\mathbf{w}_k$ to produce their own label. They assume that the feature vector $\mathbf{x}_i$ is hidden and needs to be inferred from the labels generated by annotators. Note that this setting is common in crowd-sourcing where examples are labeled without the feature vectors. However, in many applications, the feature vectors of the examples are provided. Assuming that the feature vector $\mathbf{x}_i$ is observed, the Welinder model becomes a simple model. This is because the hidden variable $\hat{\mathbf{x}}_i^k$ can be removed and the relevant information can be directly incorporated into the reviewer specific model $\mathbf{w}_k$.

2.2.4.2 Learning the consensus model. In the second research direction, the primary goal is to learn a consensus model that can be used to predict labels for future examples. Yan et al. [Yan et al., 2010] and [Raykar et al., 2010] are state-of-the art methods in this direction.

[Raykar et al., 2010] used an EM procedure to jointly learn the annotator reliability and the consensus model. The Raykar method is illustrated in Figure 9, where γ_k and ϕ_k are sensitivity (true postive rate) and specificity (1-false positive rate) of annotator k, consensus model $\mathbf{u}$ generates the correct labels z_is using the feature vector $\mathbf{x}_i$.

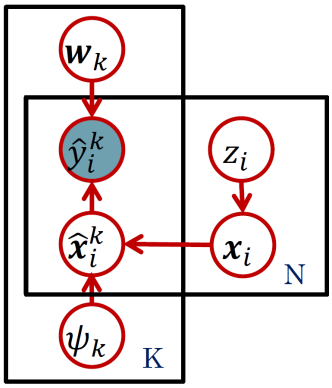

Figure 8: Graphical model for Welinder's method [Welinder et al., 2010]. N examples labeled by K annotators.

Raykar method aims to find the maximum-likelihood estimator of vectors $\boldsymbol{\gamma}, \boldsymbol{\phi}$ and $\mathbf{w}$:

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \{\hat{\boldsymbol{\gamma}}, \hat{\boldsymbol{\phi}}, \hat{\mathbf{u}}\} \\ &= \text{argmax}_{\boldsymbol{\theta}} \quad \ln \Pr(D|\boldsymbol{\theta}) \end{aligned} \tag{2.3}$$

where $D = \{\mathbf{x}_i, y_i^1, ..., y_i^K\}_{i=1}^N$ are the data that consist of N instances labeled by K annotators.

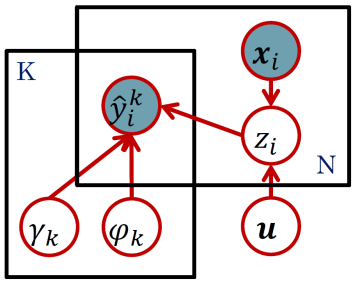

Figure 9: Graphical model for Raykar's method [Raykar et al., 2010]. N examples labeled by K annotators.

Raykar method assumes that training examples are independently sampled and y_i^k is conditionally independent of all other parameters given $\boldsymbol{\gamma}, \boldsymbol{\phi}$ and $\mathbf{u}$. Therefore, the likelihood

can be decomposed as follows:

$$\begin{aligned}
\Pr(D|\boldsymbol{\theta}) &= \prod_{i=1}^{N} \Pr(y_i^1, ..., .y_i^K | \mathbf{x}_i, \boldsymbol{\theta}) \\
&= \prod_{i=1}^{N} \Big[\Pr(y_i^1, ..., .y_i^K | z_i = 1, \boldsymbol{\gamma}) \Pr(z_i = 1 | \mathbf{x}_i, \mathbf{w}) + \\
&\quad \Pr(y_i^1, ..., .y_i^K | z_i = 0, \boldsymbol{\phi}) \Pr(z_i = 0 | \mathbf{x}_i, \mathbf{w}) \Big] \\
&= \prod_{i=1}^{N} \Big[\prod_{k=1}^{K} \Pr(y_i^k | z_i = 1, \gamma^k) \Pr(z_i = 1 | \mathbf{x}_i, \mathbf{w}) + \\
&\quad \prod_{k=1}^{K} \Pr(y_i^k | z_i = 0, \phi^k) \Pr(z_i = 0 | \mathbf{x}_i, \mathbf{w}) \Big] \\
&= \prod_{i=1}^{N} \Big[a_i p_i + b_i (1 - p_i) \Big] \qquad (2.4)
\end{aligned}$$

where

$$\begin{aligned}
p_i &= \Pr(z_i = 1 | \mathbf{x}_i, \mathbf{w}) \\
a_i &= \prod_{k=1}^{K} (\gamma^k)^{y_i^k} (1 - \gamma^k)^{1 - y_i^k} \\
b_i &= \prod_{k=1}^{K} (\phi^k)^{y_i^k} (1 - \phi^k)^{1 - y_i^k}
\end{aligned}$$

The optimization problem in Equation 2.3 can be solved by using the Expectation-Maximization (EM) [Dempster et al., 1977] procedure:

- **E-step**. Given the observation D and the current estimate of $\boldsymbol{\theta}$, and plugging Equation 2.4 into Equation 2.3, the conditional expectation of likelihood is:

$$\mathrm{E}\{\ln \Pr(D, \mathbf{z} | \boldsymbol{\theta})\}_{\Pr(\mathbf{z}|D,\boldsymbol{\theta})} = \sum_{i=1}^{N=1} \mu_i \ln p_i a_i + (1 - \mu_i) \ln(1 - p_i) b_i \qquad (2.5)$$

where $\mu_i = \Pr(z_i = 1 | y_i^1, ..., y_i^K, \mathbf{x}_i, \boldsymbol{\theta})$. μ_i can be computed using the Bayes theorem:

$$\begin{aligned}
\mu_i &\propto \Pr(y_i^1, ..., y_i^K | z_i = 1, \boldsymbol{\theta}) \cdot \Pr(z_i = 1 | \mathbf{x}_i, \boldsymbol{\theta}) \\
&= \frac{a_i p_i}{a_i p_i + b_i (1 - p_i)}
\end{aligned}$$

- **M-step**. Then $\hat{\boldsymbol{\theta}} = \{\hat{\boldsymbol{\gamma}}, \hat{\boldsymbol{\phi}}, \hat{\mathbf{u}}\}$ can be estimated by maximizing Equation 2.5:

$$\begin{aligned} \gamma^k &= \frac{\sum_{i=1}^{N} \mu_i y_i^k}{\sum_{i=1}^{N} \mu_i} \\ \phi^k &= \frac{\sum_{i=1}^{N} (1-\mu_i)(1-y_i^k)}{\sum_{i=1}^{N} (1-\mu_i)} \\ \mathbf{w}^{t+1} &= \mathbf{w}^t - \eta \mathbf{H}^{-1} \mathbf{g} \end{aligned}$$

  where $\mathbf{g}$ is the gradient vector and $\mathbf{H}$ is the Hessian matrix.

Raykar method shows superior performance over majority vote when the number of annotators is large (> 40). This is practical when the labeling task is easy and crowd-sourcing services (e.g. Amazon Mechanical Turk) can be utilized. However, it is not practical in domains where the annotation process is time-consuming and requires the work of experts, who are scarce resources and expensive. Thus, it is infeasible to hire a large number of annotators. An example is disease modeling and labeling of examples by physicians.

An approach similar to Raykar et al. was developed in [Yan et al., 2010], where the authors used a probabilistic model to model annotator expertise, which may vary and depend on the data observed. However, this method assumes that each example is labeled by all annotators in parallel. This is unrealistic in many applications, especially in medical domain, where it is common that an example is labeled by only one person.

3.0 LEARNING WITH AUXILIARY SOFT-LABEL INFORMATION

3.1 INTRODUCTION

In this chapter we propose and study a machine learning framework in which binary class label, that is used to learn classification models, is enriched by auxiliary soft-label reflecting the certainty or belief of a human annotator in the class label. In general, the soft label information can be presented either in terms of (1) a probabilistic score, e.g., the chance of a patient having a disease is 0.7, or, (2) a qualitative category, such as, weak or strong agreement with the patient having the disease. We expect the soft label information, when applied in the training (learning) phase, will let us learn binary classification models more efficiently with a smaller number of labeled examples. The idea of asking human annotators to provide auxiliary soft-labels is based on a simple premise: a human that gives us a subjective class label can often provide us with auxiliary information, which reflects his/her certainty in the class label decision, at a cost that is insignificant when compared to the cost of example review and label assessment.

Formally, our goal is to learn a classification model $g : X \rightarrow Y$, where X denotes the input (or feature) space and $Y = \{0, 1\}$ are two classes one can assign to each input. In the standard binary classification setting, the discriminant function is learned from examples with class labels $\{0, 1\}$ only. In our framework, in addition to class labels y, we also have access to auxiliary soft labels p associated with these class labels. Hence, each data entry d_i in the training data set $D = \{d_1, d_2, \cdots d_N\}$ consists of three components: $d_i = (\mathbf{x}_i, y_i, p_i)$ - an input $\mathbf{x}_i$, a binary class label y_i and a soft label p_i.

We first study the case when soft-labels p_i are presented in terms of probabilistic scores. We start by showing that we can modify a simple regression algorithm to incorporate the

soft-label assessments and learn a better classifier. Then we show this regression method may be very sensitive to the assessment inconsistencies and noise since it depends strongly on the accuracy of the human subjective estimates. To address the problem, we propose a novel method based on the support vector machines and learning-to-rank methodologies, that is more robust to noise and able to learn high quality classifiers with smaller numbers of labeled examples. Briefly, instead of relying on exact estimates of soft-labels, this method models the pair-wise ordinal relations between all training examples. The proposed method shows superior performance compared to traditional binary classifiers, e.g. SVM and logistic regression. However, it may not scale well with large data sets because it models the pair-wise ordinal relations between all pairs of examples, which means the number of constraints in optimization is (approximately) N^2, where N is the number of examples.

To address the quadratic complexity problem, we propose two methods to reduce the number of constraints. The first method discretizes all training examples into a constant number of bins based on the soft-label information, then models the pairwise ordinal relations between pairs of examples only when they belong to different bins. This helps to reduce the number constraints in practice. However, because there is still $O(N)$ examples in each bin, the total number of constraints is still $O(N^2)$. We propose the second method based on ordinal regression, which can reduce the number of constraints to $O(N)$. Briefly, instead of explicit modeling pairwise relations between examples, we model the relations between examples and the boundaries separating the bins. Since the number of bins and their boundaries is constant, the number of constraints is linear in the number of training examples. We show that, with a linear number of constraints in optimization, the performance of this method is still comparable to the proposed ranking method with $O(N^2)$ constraints and significantly outperforms other baseline methods.

We also study the case when soft-labels are presented in terms of qualitative ordinal categories, e.g. weak, medium, strong belief that a patient has a disease. We show that our proposed methods based on ranking and ordinal regression can also be applied to learn classifiers efficiently in this case. In fact, instead of making bins (or categories) by discretization of probabilistic soft-labels, we have categorical labels explicitly given by human annotators. These labels essentially create categories of different examples, therefore, the

proposed methods that work with discrete bins can be naturally applied to learn classifiers in this case.

We demonstrate advantages and disadvantages of all proposed methods on a number of UCI data sets. We also compare them on real-medical data representing expert assessment of the risk of the Heparin Induced Thrombocytopenia (HIT) [Warkentin et al., 2000] given a set of patients' observations and labs. The experimental results show that, whether soft-labels are represented in terms of probabilistic scores or ordinal categories, our methods still significantly outperform traditional binary classification methods.

Our framework is different than transfer learning. Transfer learning relies on training data and labels of some tasks/domains that are relevant to the target task; however, this source of data is not always available. Moreover, transfer learning often requires to learn/tune the "relevance" of other data sources to the current, which is a non-trivial problem, and sometimes can lead to "negative transfer", i.e. negative impact on learning of the main target task. In contrast, our approach aims to obtain more information by asking certainty labels, that are directly related to the target task, and based on the same training data as the main (binary) label decision.

Our approach is complimentary to active learning: while the later aims to select the most informative examples, we aim to gain more useful information from those selected. This orthogonality gives us an opportunity to combine these two approaches. In this work, we also investigate the extension of our framework for the active learning setting, where we actively select examples to query for labels based on the model learned from soft-label information.

In the following, we first study how to learn binary classification models efficiently from soft-labels presented in terms of probabilistic scores, then study how to learn classifiers from ordinal categorical soft-labels.

3.2 ALGORITHMS FOR LEARNING WITH PROBABILISTIC SOFT-LABELS

In this section we develop classification learning algorithms that let us accept and learn from probabilistic soft-labels $p_i \in [0,1]$. We start with modifying a simple discriminative model, and keep modifying the model to account for possible noise and inconsistencies in subjective probability estimates.

3.2.1 Discriminative regression methods

In the discriminative classification approach we want to learn a function $f : X \to \mathscr{R}$ that lets us discriminate examples in the two classes. Once the function f is known, the class decision is made with the help of a threshold σ such that for values $f(\mathbf{x}) \geq \sigma$ we classify the example as class 1, otherwise as class 0. In our framework, in addition to binary class labels $\{0,1\}$, we also have auxiliary probabilistic information associated with these class labels. The question is how this information can be used to learn a better discriminant function.

3.2.1.1 Linear regression One relatively straightforward solution is to assume the discriminant function is defined directly in terms of these auxiliary probabilities. In such a case, the learning of the discriminant function can be converted into a regression problem. One way to learn the function is to regress the features directly to probabilities, that is, we can learn a regression mapping f where $(\mathbf{x}_i, p_i)$ are the input-output pairs.

Assuming the function $f : X \to \mathscr{R}$ is formed by a linear model $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, the learning problem becomes a linear regression problem solved by minimizing the error function based on the sum of squared residuals:

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} \left(\mathbf{w}^\top \mathbf{x}_i - p_i\right)^2 + Q(\mathbf{w})$$

where $Q(\mathbf{w})$ is an optional regularization term that may help to prevent model over-fit (Section 2.1.1.2). The solution $\mathbf{w}^*$ yields a weight vector optimizing the linear model.

Further in this work, we refer to this method as to **LinRaux** (Linear regression with auxiliary soft-label information).

Defining the classification threshold. Once the weights of the discriminant function are learned, a classifier can be defined using a decision threshold σ. To find the optimal threshold, we can use class labels and minimize the overall loss in the training data.

3.2.1.2 Consistency with probabilistic assessment

Obviously, using an arbitrary function model, the outputs of the regression may not be consistent with probabilities. For example, by applying a linear regression directly to the input-probability pairs we may not guarantee the consistency of probabilistic labels once the model is learned, that is, some data points may fall outside $[0,1]$ interval. An alternative is to regress inputs to a new space in $\mathscr{R}$ obtained by transforming the probabilistic space, such that the transformation is monotonic in p_i, and its inverse lets us revert back to probabilities. An example of such a transformation is $t(p_i) = \ln \frac{p_i}{1-p_i}$ which is the inverse of the logistic function. In such a case the regression model is trained on $(\mathbf{x}_i, t(p_i))$ pairs. The results of the regression can be transformed back to the probability space by using the logistic function $g(s) = \frac{1}{1+e^{-s}}$ and the probabilities are consistent. Now, the learning problem becomes a linear regression problem solved by minimizing the following error function:

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} \left(\mathbf{w}^\top \mathbf{x}_i - t(p_i)\right)^2 + Q(\mathbf{w})$$

where $Q(\mathbf{w})$ is a regularization term (Section 2.1.1.2). The solution $\mathbf{w}^*$ yields a weight vector optimizing the linear model and if needed, the posterior probability is recovered as: $p(y=1|\mathbf{x},\mathbf{w}) = \frac{1}{1+e^{-\mathbf{w}^\top \mathbf{x}}}$.

Further in this work, we refer to this method as to **LogRaux** (Regression with log transformation on auxiliary soft-label information).

3.2.1.3 Soft-labels help to learn better classification models

To test whether the probabilistic soft-label information helps to learn better classification models, we compare AUC performance of the two regression methods learned from soft-labels - LinRaux and LogRaux, with two standard binary classifiers learned from binary class labels only - support vector machines (SVM) and logistic regression (LogR). We conduct experiments on five UCI data sets with simulated soft-labels and our medical data with real soft-labels given by

experts. For illustration purpose, we take one of the data sets - "Concrete", as the running example in the current and following sections. Details about experimental setup, how class and soft labels are given, and results on all other data sets are described in Experiment Sections 3.4 and 3.5.

Figure 10 shows the AUC performance of the four methods on "Concrete" data set. We see that LinRaux and LogRaux clearly outperforms SVM and logistic regression. Similar results are also observed on all other data sets. This demonstrates that soft-label information can help us to learn better classification models.

Note that the performance of the two regression methods LinRaux and LogRaux are very similar and overlapped on "Concrete" data as shown in Figure 10. It turns out that this is true also for all other data sets we experimented with (see Section 3.5 for full results on all data sets). Therefore, for simplicity sake, we sometimes omit LogRaux results in the remaining figures.

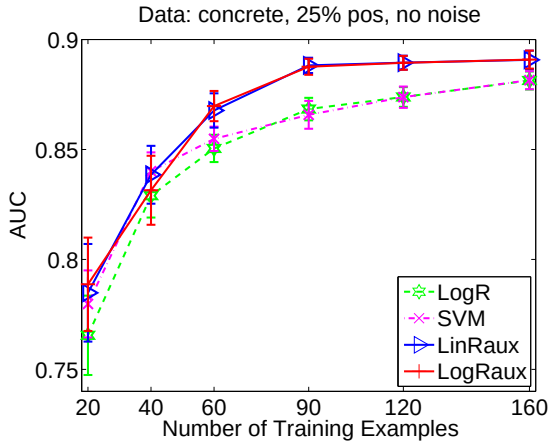

Figure 10: Regression methods LinRaux and LogRaux, that learn from probabilistic soft-label information, outperform binary classification models SVM and logistic regression (LogR), that learn from binary class labels only. The soft-labels are not corrupted by noise.

3.2.2 Noise in subjective estimates

Learning of the discriminant function directly from auxiliary probabilities raises a concern of what happens if these subjective probabilistic assessments are not consistent and subject

to variations due to inaccurate subjective human estimates. This issue is well documented in the literature; humans are not very good in providing well calibrated probabilistic estimates [Suantak et al., 1996, Griffin and Tversky, 1992, O'Hagan et al., 2007]. Therefore, the performance of predictive models trained on probabilistic soft-labels may be negatively influenced by these inaccurate estimates. Clearly, if the estimates differ widely one expects them to impact the quality of the learned discriminant function.

In order to test how noisy estimates influence the models trained on soft probabilistic labels, we have conducted experiments on UCI data sets, where we inject Gaussian noise in the soft labels. Figure 11 shows the AUC performance of methods from Figure 11 on "Concrete" data set when soft-labels the methods were trained on were corrupted by three different levels of noise: weak, medium and strong noise (details about noise levels are described in Section 3.5).

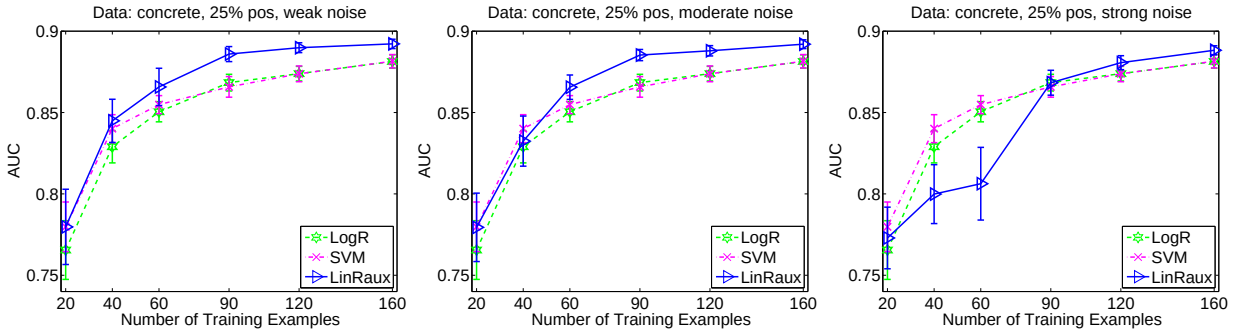

Figure 11: AUC of different models when the auxiliary soft-labels have three different levels of noise: (Left) weak noise, (Middle) Moderate noise, (Right) Strong noise. Regression method LinRaux clearly outperforms standard binary classifiers SVM and LogR when noise is weak. However, it's performance deteriorates quickly as the noise increases, and is outperformed by SVM and LogR when the noise is strong.

Figure 11.left shows the benefits of probabilistic soft-label information when the noise in the soft labels is weak. LinRaux method that uses soft-label information clearly outperforms SVM and LogR methods that rely on class labels only. Figure 11.middle shows the quality of learned models when these are trained with moderate noise. We see that the performance of LinRaux method somewhat deteriorates but still outperforms the methods trained on binary labels. However, when the models are trained on soft-labels with strong noise

(Figure 11.right), the benefit of soft-label information disappears and the binary label information leads to better classification models. This illustrates that the regression method that fits probabilistic soft-labels is sensitive to the noise and inconsistencies in soft-label assessments. The challenge is to develop methods that are more robust to these inconsistencies, and hence, can accept soft-labels based on subjective human estimates.

3.2.3 A ranking method to improve the noise tolerance

To address the soft-label noise problem we propose to adapt ranking methods that are more robust and tolerate the noise in the estimates better. Briefly, instead of relying strongly on exact probabilistic estimates, we try to model the relation in between the two probabilistic assessments only qualitatively, in terms of pairwise order constraints.

Let $f : X \rightarrow \mathscr{R}$ be a linear model $f(\mathbf{x}) = \mathbf{w}^{\top}\mathbf{x}$ that lets us discriminate between examples in class 0 and class 1. Now assume the same model represents a linear ranking function that lets us order individual data points such that if the instance $\mathbf{x}_1$ is ranked higher than $\mathbf{x}_2$ then $f(\mathbf{x}_1) > f(\mathbf{x}_2)$. Now assuming any two data points $\mathbf{x}_1$ and $\mathbf{x}_2$ are ordered according to their subjective probability p_1 and p_2, we expect the ranking function to preserve their order.

The learning to rank algorithms [Herbrich et al., 1999, Joachims, 2002] let us find the ranking function from the training data by minimizing the number of violated pairwise constrains between the data points and the amount of these violations. Such a formulation of a learning problem makes the problem of learning the discriminative model less dependent on exact subjective value estimates that are used to induce the pairwise ordering. Hence we hope this relaxation would allow us to better absorb some amount of noise in subjective probability estimates, eventually leading to more robust learning algorithms.

Let r^* be our target ranking order determined by the probabilistic information p_i associated with each example. Then for every pair of examples $\mathbf{x}_i$ and $\mathbf{x}_j$: $(\mathbf{x}_i, \mathbf{x}_j) \in r^*$ we can write a constraint $\mathbf{w}^{\top}(\mathbf{x}_i - \mathbf{x}_j) > 0$ we want the ranking function $f(\mathbf{x}) = \mathbf{w}^{\top}\mathbf{x}$ to satisfy. Just, like in the classification SVM, we allow some flexibility in building the hyperplane by adding slack variables $\xi_{i,j}$ representing penalties for the constraint violation and a constant C to regular-

ize these penalties. Now the learning-to-rank of N examples is equivalent to the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & Q(\mathbf{w}) + C\sum_{i,j}\xi_{i,j} \\ \text{subject to:} & \\ \forall(\mathbf{x}_i,\mathbf{x}_j)\in r^*: \quad & \mathbf{w}^\top(\mathbf{x}_i-\mathbf{x}_j)\geq 1-\xi_{i,j} \\ \forall i\forall j: \quad & \xi_{i,j}\geq 0 \end{aligned}$$

where $i,j = 1,2,...,N$ indexes examples, $Q(\mathbf{w})$ a regularization penalty, typically $\frac{1}{2}\mathbf{w}^\top\mathbf{w}$, and C is a constant. Solving this problem will give us the weight vector w and the discriminant function $f(\mathbf{x}) = \mathbf{w}^\top\mathbf{x}$ that violates the smallest number of constraints.

Combining class label and soft-label information in optimization. The ranking approach presented above improves the noise tolerance of the model to subjective probability estimates by ignoring their exact values and taking into account only their relative orders. However, because the human estimates are subjective, it is not uncommon that two different class labels may get probabilities that rank them opposite of their expected order, that is, the probability assigned to a class 0 example is higher than the subjective probability assigned to a class 1 example. The main question that arises is whether it is possible to address these inconsistencies by incorporating both class labels and auxiliary soft-label information by combining their loss functions into a single coherent optimization criterion. The intuition for doing this is to assure the model is driven by the class label first and refined with the auxiliary probabilistic information, if it is consistent with the labels. Based on this intuition, we propose to optimize:

$$\begin{aligned} \min_{\mathbf{w}} \quad & Q(\mathbf{w}) + B\sum_{i}\eta_i + C\sum_{i,j}\xi_{i,j} \\ \text{subject to:} & \\ \forall i: \quad & y_i(\mathbf{w}^\top\mathbf{x}_i+b)\geq 1-\eta_i \\ \forall(\mathbf{x}_i,\mathbf{x}_j)\in r^*: \quad & \mathbf{w}^\top(\mathbf{x}_i-\mathbf{x}_j)\geq 1-\xi_{i,j} \\ \forall i: \quad & \eta_i\geq 0 \\ \forall i\forall j: \quad & \xi_{i,j}\geq 0 \end{aligned} \tag{3.1}$$

where B and C are constants and $Q(\mathbf{w})$ is a regularization penalty.

This formulation assumes two sets of constraints, one defining the hinge loss for all examples and their class labels, the other one defining the loss for not respecting the orders induced by subjective probabilistic estimates. Once again, solving this problem will give us the weight vector $\mathbf{w}$ and the discriminant function $f(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x}$ that violates the smallest number of constraints. Note that by changing scaling constants B and C one can stress more either the label or the probabilistic order information. For example, if the noise in probabilistic labels is large then its influence can be decreased by decreasing C. In general, the settings of these parameters can be optimized using the internal cross-validation approach (3-fold cross-validation was used in our experiments). In this work we refer to this method as to **SvmAuxPair**.

Figure 12 compares the AUC performance of this ranking method (SvmAuxPair) with regression method LinRaux (Section 3.2.1.1) and standard binary classifiers SVM and LogR, on the "Concrete" data we used in Figure 11. Sub-figures 12.left, 12.middle and 12.right show the learning curves of the methods where soft-labels are corrupted by weak, moderate and strong noise, respectively. As we can see, when the noise is weak (Figure 12.left), both methods that learn from soft-labels (LinRaux and SvmAuxPair) clearly outperform methods that learn from class labels only (SVM and LogR). When the noise level is moderate (12.middle), LinRaux's performance decreases more than SvmAuxPair's, but these methods still outperform SVM and LogR. However, when the noise level is strong, LinRaux performs poorly and is worse than SVM and LogR, whereas SvmAuxPair is still the best. This shows that while the performance of LinRaux deteriorates quickly as the noise level increases, SvmAuxPair is very robust and consistently outperforms all other methods across different noise settings. This is because unlike LinRaux, SvmAuxPair does not rely on the exact estimates of probabilistic soft-labels, therefore does not suffer as much when the noise in soft-label is strong.

The proposed ranking approach SvmAuxPair has superior performance compared to other methods. However, since it models the pair-wise relations between all pairs of examples, the number of constraints in the optimization is quadratic in the number of examples used to train the model. Therefore, this method may not scale up well when the size

of the training data is large. In the following section we propose methods to address this disadvantage.

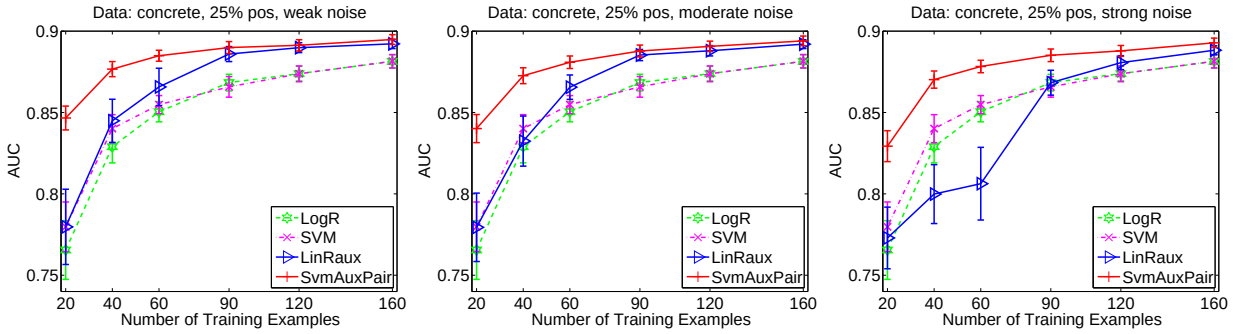

Figure 12: AUC of different models when the auxiliary soft-labels have three different levels of noise: (Left) weak noise, (Middle) Moderate noise, (Right) Strong noise. Ranking method SvmAuxPair is robust to noise and outperforms all other methods across different noise levels.

3.2.4 Ranking methods with reduced number of constraints

In this section we describe two methods that help us to reduce the number constraints in the above optimization problem (Equation 3.1) while still being robust to noise in soft-labels.

3.2.4.1 A ranking method with discrete bins of examples SvmAuxPair enforces one constraint for every pair of training examples. So, in order to reduce the number of constraints, the first idea is to enforce constraints only for pairs that are significantly different in terms of soft-labels. For example, we can sort all training examples by the value of their soft-label and distribute them equally into k bins. After that, we can apply the above ranking method, but include and enforce pairwise constraints only for examples that belong to different bins. In this work, we refer to this method as to **SvmAuxBinPair**.

The advantage of this method is that it helps us to reduce the number of constraints compared to SvmAuxPair method. However, the number of constraints is reduced only by a constant factor, which means we still have $O(N^2)$ constraints in the optimization. Another disadvantage of this method is that we need to decide on the number of bins and how probabilistic soft-labels are distributed into these bins.

3.2.4.2 A ranking method with linear number of constraints To address the limitation of SvmAuxBinPair, we propose a new method for which the number of constraints is only linear in the size of training data. This method is based on [Chu and Keerthi, 2005], where the authors used SVM in the context of ordinal regression learning. We adapt this idea for our task, where we learn a classification model from both binary class labels and ordinal soft-labels given by human experts.

To illustrate the idea, let us have examples distributed into four bins according to their soft-labels, as shown in Figure 13. These examples are mapped to the real line by a decision function $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$, the value of which indicates how likely an example being positive: larger $f(\mathbf{x})$ means $\mathbf{x}$ is more likely positive, and vice versa. Our objective is to learn $f(\mathbf{x})$.

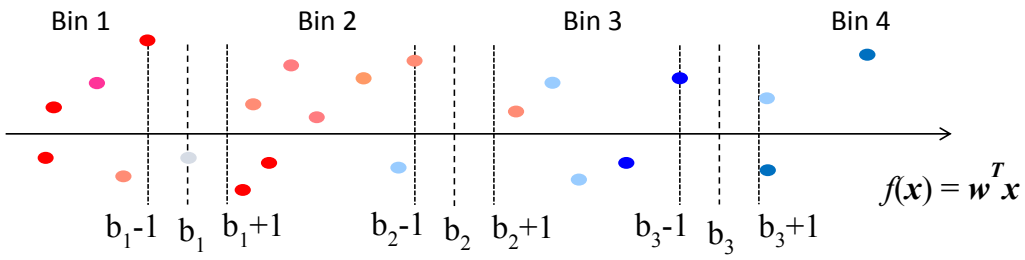

Figure 13: Ranking based on discrete bins. Examples are distributed to different bins based on their soft-labels. Optimization constraints are defined for examples and their relative positions to bin boundaries. Red and blue data points denote negative and positive examples, respectively.

The examples in different bins are projected (mapped) to the real line and bins are separated by some boundaries: b_1, b_2, b_3 as shown in Figure 13. Instead of enforcing order constraints for every pair of examples in different bins ($O(N^2)$ pairs), we try to enforce that every example $\mathbf{x}$ falls on the correct side of each boundary defining the bins. That is, if the example $\mathbf{x}$ is located on the left side of a boundary b_j then its decision value is smaller than the lower margin of b_j: $f(\mathbf{x}) \leq b_j - 1$, otherwise its decision value is greater than the upper margin of b_j: $f(\mathbf{x}) \geq b_j + 1$. This means that for each boundary there are N constraints, so in total, for $r-1$ boundaries, there are $(r-1)N$ constraints, which is linear in the number of training examples ($O(N)$).

Because of the noise, we cannot expect that all constraints will be always satisfied. In

that case, we allow violations of constraints but penalize errors by slack variables $\xi_{jk}, j = 1..r-1, k = 1..n_j$, where n_j is the number of examples in bin j. In addition to order constraints between examples and bins, we also enforce order constraints for binary class labels, that is, if $\mathbf{x}_i$ belongs to class 0 ($y_i = 0$) and $\mathbf{x}_j$ belongs to class 1 ($y_i = 1$) then $f(\mathbf{x}_i) < f(\mathbf{x}_j)$. The violations of class label constraints are penalized by slack variables η_i.

To wrap up, we propose to learn the decision function $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$ by solving the following optimization problem:

$$
\begin{aligned}
\min_{\mathbf{w},b,b_j,\eta_i,\xi_{jk}} \quad & Q(\mathbf{w}) + B\sum_{i=1}^{N}\eta_i + C\sum_{j=1}^{r-1}\sum_{k=1}^{n_j}\xi_{jk} \\
\text{subject to:} & \\
y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \eta_i \quad & \forall i = 1..N \\
(\mathbf{w}^\top \mathbf{x}_i - b_j) \leq -1 + \xi_{jk} \quad & \forall j = 1..r-1, k = 1..n_j, \mathbf{x}_i \in bins\, 1,..,j \\
(\mathbf{w}^\top \mathbf{x}_i - b_j) \geq 1 - \xi_{jk} \quad & \forall \mathbf{x}_i \in bins\, j+1,..,r \\
\eta_i \geq 0 & \\
\xi_{jk} \geq 0 &
\end{aligned}
$$

where $Q(\mathbf{w})$ is a regularization term, $B\sum_{i=1}^{N}\eta_i$ is the total penalty for violating class label constraints and $C\sum_{j=1}^{r-1}\sum_{k=1}^{n_j}\xi_{jk}$ is the total penalty for violating bin order constraints.

In this work, we refer to this method as to **SvmAuxOrd**.

Figure 14 shows the AUC performance of the two new ranking methods SvmAuxBinPair and SvmAuxOrd on the "Concrete" data set with three levels of noise. First, note that all three ranking methods SmvAuxPair, SvmAuxBinPair and SvmAuxOrd outperform other methods across all noise levels (weak, moderate, strong noise shown in left, middle, right sub-figures, respectively). Second, SvmAuxBinPair and SvmAuxOrd are comparable with SvmAuxPair. This means that SvmAuxOrd that uses only $O(N)$ optimization constraints still performs as well as SvmAuxPair and SvmAuxBinPair, which have $O(N^2)$ optimization constraints.

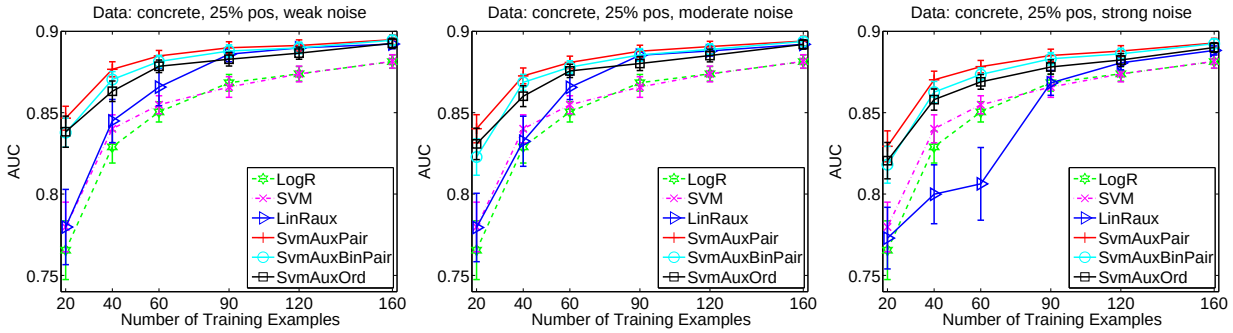

Figure 14: Ranking method SvmAuxOrd, with a linear number of optimization constraints, is robust to soft-label noise and is comparable with SvmAuxPair and SvmAuxBinPair, which have a quadratic number of constraints.

3.3 LEARNING WITH AUXILIARY CATEGORICAL SOFT-LABELS

As mentioned in section 3.1, auxiliary soft-labels p_i may be presented in terms of probabilistic scores or qualitative ordinal categories. For example, in our study for the problem of monitoring the risk of heparin-induced thrombocytopenia (HIT) (see more details in Section 3.5.1), we ask experts to assess the certainty in raising an alert on HIT, in terms of ordinal categories: "strongly-disagree", "weakly-disagree", "weakly-agree", "strongly-agree" with the alert. In the case of learning with ordinal categorical soft-labels, the regression methods (Sections 3.2.1.1 and 3.2.1.2) cannot be applied because target soft-labels are no longer numeric. However, the above ranking methods using discrete bins (SvmAuxBinPairs and SvmAuxOrd, Sections 3.2.4.1 and 3.2.4.2) can be applied naturally since we already have discrete categories given by a human. In fact, the only difference in this case is that we apply the methods directly to the categorical soft-labels without doing any discretization/distribution of examples into bins. This is an advantage because we no longer need to decide on the number of bins (categories) and the way to distribute examples.

3.4 EXPERIMENTS WITH UCI DATA SETS

We have conducted two sets of experiments to test our framework and methods. The first set of experiments uses five UCI data sets and simulated probabilistic labels. The second set of experiments uses a real-world clinical data set with human assessments of a life threatening condition – the heparin induced thrombocytopenia [Warkentin et al., 2000, Warkentin, 2003] (section 3.5).

In this section we show experiments on UCI data sets with simulated probabilistic labels. We use the data to first demonstrate the benefits of soft-label information for learning classification models. Second, we show the robustness of our methods to noise in the soft-label information. Finally, we use the data to show how our approach can alleviate the learning problem when the data are unbalanced.

3.4.1 Experimental set-up

Since there is no UCI data set with both class labels and auxiliary probabilistic soft-labels, we used five UCI regression data sets and generated probabilities from their continuous outputs. Their properties are summarized in Table 1. For all these data sets we normalized the continuous output values and interpreted them as probabilities. We defined a binary class variable by using a threshold on the underlying continuous variable. For example, the output variable representing the strength of concrete in the Concrete data set was used to define two classes: a concrete with a good strength (class 1) and a concrete with a bad strength (class 0). Specific thresholds used to define the binary class variable depend on the the percentage of positive examples we want to generate, for example, to generate a data set with 25% of examples being positive we assign class 1 to examples with top 25% largest output values, and class 0 to the rest.

We used the following models in our comparisons:

- **LogR:** The standard logistic regression with L1 regularization trained on binary labels,
- **LinRaux:** The linear regression model with L1 regularization trained on auxiliary probabilistic information (from Section 3.2.1.1),

- **SVM:** The standard linear SVM with the hinge loss and L1 regularization trained on binary labels only, and
- **SvmAuxPair:** The new SVM-based ranking model (from Section 3.2.3) with the L1 regularization penalties and two hinge losses: one for binary labels and the other for pairwise order constraints between all pairs of examples.
- **SvmAuxBinPair:** The new SVM-based ranking model (from Section 3.2.4.1) that discretizes examples into 5 bins according to the soft-labels and enforces constraints between examples of different bins.
- **SvmAuxOrd:** The new ranking model (from Section 3.2.4.2) based on ordinal regression, that discretizes examples into 5 bins according to their soft-labels and enforces constraints between examples and boundaries of the different bins.

The constants C and B for SVM-based ranking models were optimized using 3-fold cross-validation approach.

We evaluated performance of the different methods by calculating the Wilcoxon statistic (the area under the ROC curve). Each data set was split into training and test set (2/3 and 1/3 of all data, respectively). We randomly selected samples from the training set to train the models. The training process was repeated 100 times. We reported the average AUC on the test set.

Table 1: UCI data sets used in the experiments

Data set	# examples	# features
ailerons	7154	40
concrete	1030	8
bank8	4499	8
housing	506	13
pol	5000	48

3.4.2 Effect of the training data size on the model quality

To test the benefit of the methods and the impact of probabilistic soft-label information on the sample complexity we trained the new models with training data of different size and compared them with models that learn from the binary labels only. Figure 15 compares the

performance of the different methods on all five UCI data sets by varying the number of samples selected for training. The error bars show 95% confidence interval.

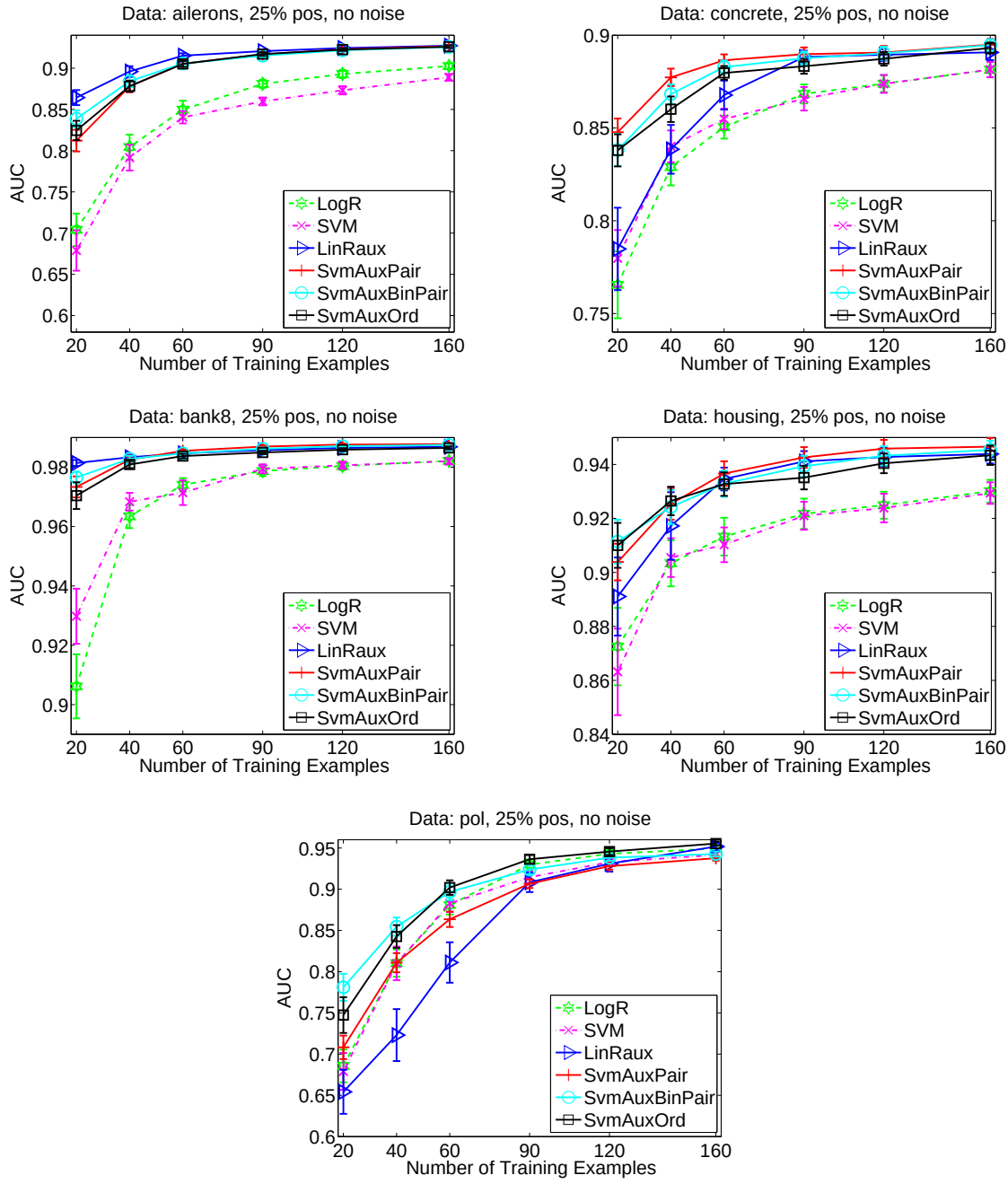

Figure 15: The benefit of learning with auxiliary probabilistic information on five different UCI data sets. The quality of resulting classification models for different training sample sizes is shown in terms of the Area under the ROC curve statistic.

The results on all five data sets clearly show the benefit of learning with auxiliary prob-

abilistic information. Methods trained with the auxiliary information - SvmAuxPair, SvmAuxBinPair, SvmAuxOrd and LinRaux consistently outperform the standard binary classifiers - SVM and logistic regression, and the sample complexity for training the model of equivalent quality was greatly reduced. SvmAuxPair is the best method on two of the data sets, SvmAuxOrd is the best on one, and the linear regression on soft-label information is the best method on two data sets.

3.4.3 Effect of noise on the auxiliary soft-label information

Our first experiment assumed the class label is defined directly by the probabilistic information. It meant to show that the auxiliary information may help. However, in practice the probabilistic information is often imprecise and subject to noise. This may effect its utility for learning the classification models. Our second experiment aims to demonstrate the robustness of our method to such a noise.

Figure 16 shows the performance of five methods from the previous experiment when auxiliary probabilistic information is corrupted by a noise. To obtain the noisy estimates each auxiliary probability value was modified by adding Gaussian noise as follows: noisy auxiliary value = original auxiliary value * (1 + $noiselevel$ * N(0,1)). We assume four different levels of noise: no, weak, moderate and strong noise, which are specified as follows:

- No noise: 0;
- Weak noise: Gaussian noise from 0.05 * N(0,1);
- Moderate noise: Gaussian noise from 0.15 * N(0,1);
- Strong noise: Gaussian noise from 0.30 * N(0,1).

Intuitively, this means that the average noise to signal ratios for weak, moderate, strong noises are 5%, 15% and 30%, respectively. To avoid inconsistent probability values, we assured all auxiliary values always fell in the interval [0,1].

When the noise is injected into the probabilistic information the linear regression model trained with probabilistic information is sensitive and its performance drops. We see the logistic regression model trained on binary labels in some instances outperforms the regression model with auxiliary information. However, our ranking approaches are more robust

and outperform both the baseline logistic regression and the baseline SVM for all three noise levels. This shows the robustness of our approaches to noisy soft-label estimates.

3.4.4 Effect of the training size on the training time

In the previous section (3.4.3) we have shown that: (1) the performance of our ranking-based methods trained with both class and soft labels (SvmAuxPair, SvmAuxBinPair, SvmAuxOrd) clearly outperform baseline methods (LinRaux, logistic regression and SVM), and (2) the performance of SvmAuxPair, SvmAuxBinPair and SvmAuxOrd are comparable. In this section, we compare the training running time of these methods when the size of the training data varies. Results are shown in Figure 17. Note that the training time depends on the number of examples but not on the noise level, so we show results only for the moderate level of noise. The results for other noise levels are similar.

We can see that the training time of SvmAuxOrd method scales up with the training size much better than SvmAuxPair and SvmAuxBinPair do. This is because, as discussed in Section 3.2.4.2, the number of optimization constraints for SvmAuxOrd is only $O(N)$ (N is the number of examples), whereas that number for SvmAuxPair, SvmAuxBinPair is $O(N^2)$.

We have discussed the complexity of different methods in terms of the number of constraints in the optimization task. But, what is the complexity in terms of the number of training examples? First, for the standard SVM formulation these two complexities turn out to be the same because there is one constraint for each training example defining the optimization task. Second, SvmAuxPair, SvmAuxBinPair and SvmAuxOrd essentially solve the SVM problem for an augmented set of training examples generated from pairs of original examples. The three methods differ in what pairs are included in the augmented set. For example, in the case of SvmAuxPair, the augmented set is constructed from all pairs of original examples. As a result, similar to SVM, the complexity in the number of constraints is the same as the complexity in the number of (augmented) examples. Then, one may ask: what is the (time) complexity in terms of the number of examples? The answer depends on what SVM solver one uses to solve the optimization for SVM and whether a non-linear kernel is used or not.

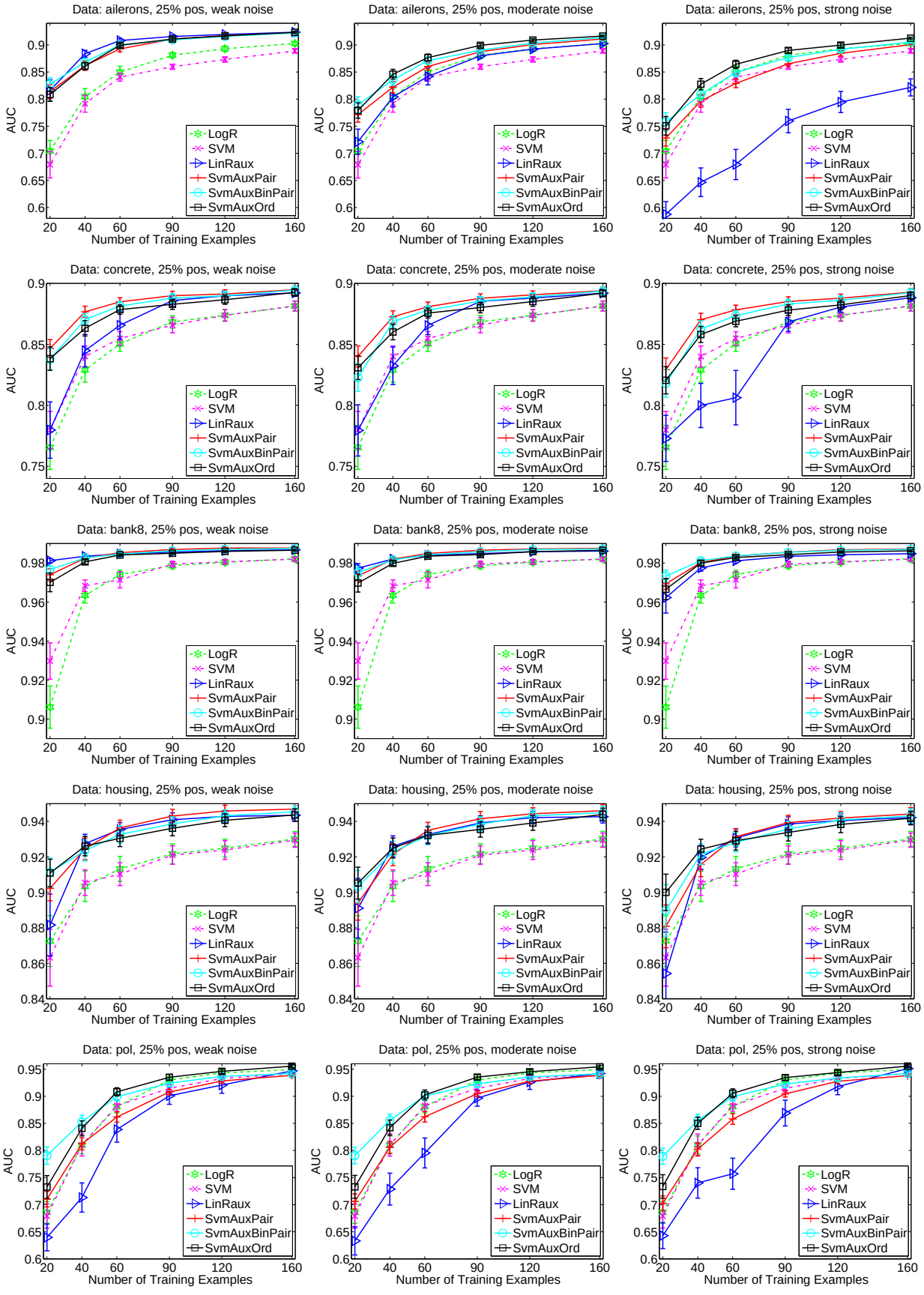

Figure 16: Learning with noise in soft-labels. AUC vs. sample size for different learning methods trained on data with soft-label information corrupted by three levels of noise: left column - weak noise (5%), middle - moderate noise (15%), right - strong noise (30%). One row of figures for each data set.

In our experiments we used the state-of-the-art SVM solver implemented in Liblinear package [Chang et al., 2008], which is specially designed for the linear SVM model. This solver globally converges at linear rate [Chang et al., 2008]. This means the time complexity of SVM and SvmAuxOrd is $O(N)$, whereas that of SvmAuxPair and SvmAuxBinPair is $O(N^2)$. Other optimization methods for SVM may have different time complexity. For example, Sequential Minimal Optimization (SMO) scales somewhere between linear and quadratic [Platt, 1999], depending on specific problems and kernels used. This means if SMO was used, the total time complexity of SVM and SvmAuxOrd would be between $O(N)$ and $O(N^2)$ and that of SvmAuxPair and SvmAuxBinPair would be between $O(N^2)$ and $O(N^4)$.

In conclusion, our SvmAuxOrd method is able to achieve AUC performance that is close to the ranking-based approach and, at the same time, has good scalability of standard baseline methods. This is important for many practical applications where learning systems are typically trained on very big data sets.

3.4.5 Effect of auxiliary soft-label information when learning with unbalanced data sets

The binary labels in all previous experiments were generated using the probabilistic information such that the number of positive and negative examples was 25% and 75% of data respectively. The question we investigate now is how the probabilistic information influences the learning process when different proportions of positive and negative examples are observed. In general, we expect that learning from labeled data when data set is unbalanced is much harder than with a balanced data set.

Experiment. Figure 18 compares five learning methods on five UCI data sets where positive labels were restricted to top 10, 25 and 50 percent of examples with the highest observed outcome values respectively. After labels were generated, the weak level of noise was applied to corrupt the probabilistic information.

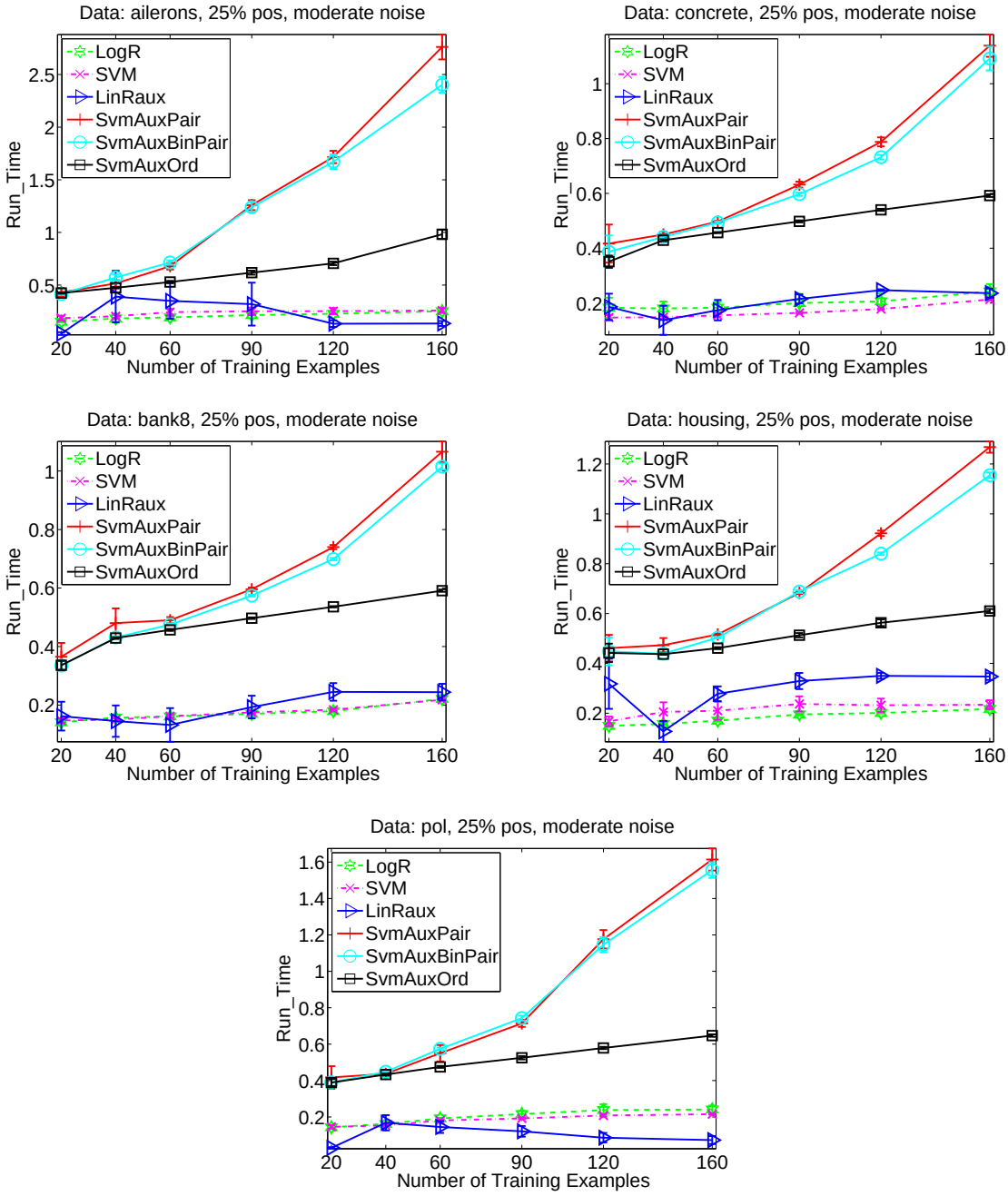

Figure 17: Effect of the training data size (number of examples) on the training running time (seconds).

The results in Figure 18 show that our methods (LinRaux, SvmAuxPair, SvmAuxBinPair and SvmAuxOrd) that learn with auxiliary information outperform methods that learn only from binary labels (LogR and SVM) in all three levels of data balance. The advantage of our methods is clearer when data are more unbalanced (advantage at 10% is higher than at

25%, and much higher than at 50%). This is important because in many domains, especially medical domain, data are often highly unbalanced, i.e. the number of positive examples (patients diagnosed with some disease) is much smaller than the number of negative examples (without disease). This means that auxiliary labels are very useful for practical learning tasks.

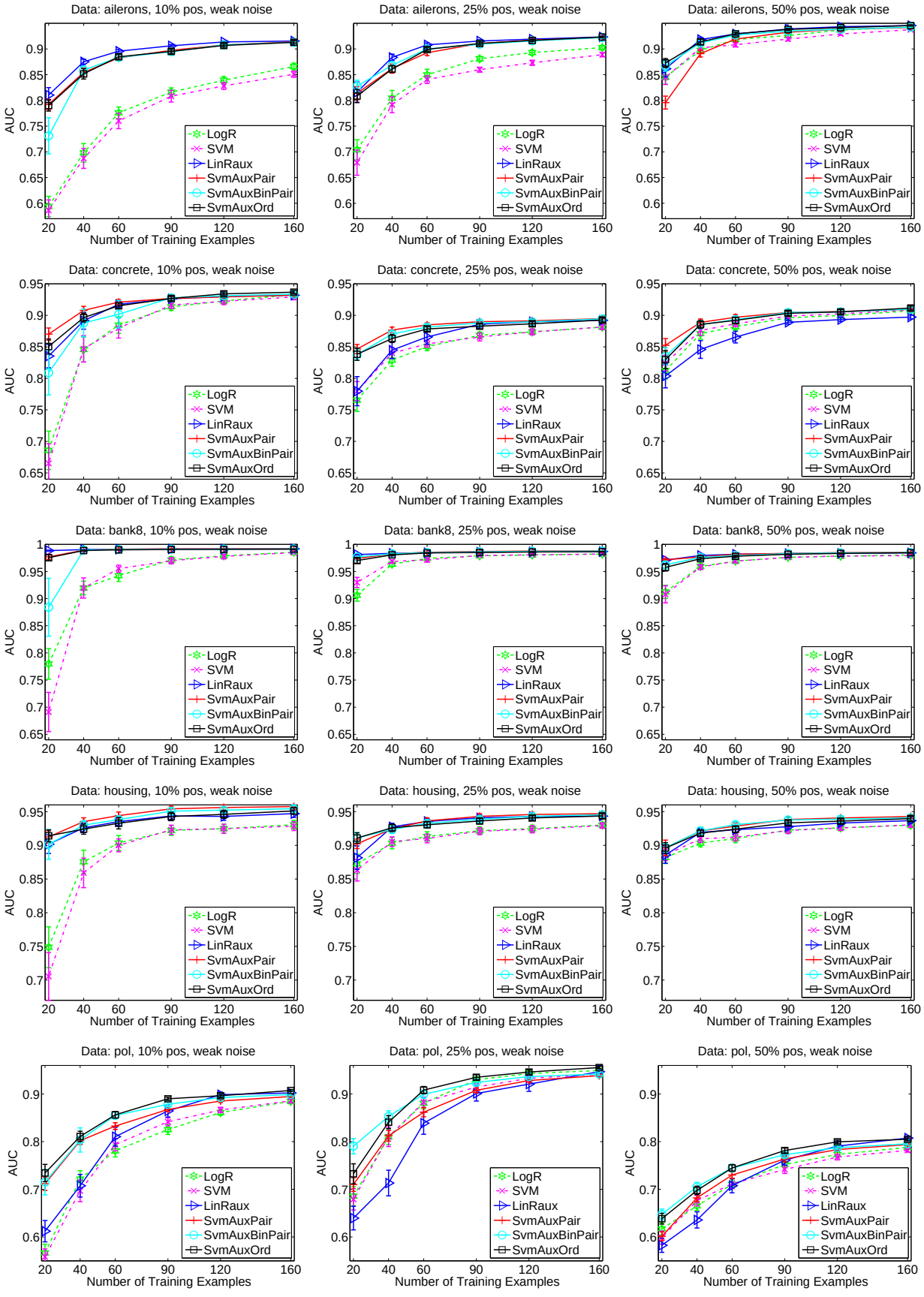

Figure 18: Learning with unbalanced data. Area under the ROC curve vs. sample size for different learning methods trained on data with different ratios of positive examples: left column - 5%, middle - 25%, right - 50%. One row of figures for each data set.

3.5 EXPERIMENTS WITH CLINICAL DATA

All experiments so far were conducted on synthetic data where noise in soft-labels was simulated. The question is whether the proposed approaches really work on data labeled by humans. To demonstrate that, we apply the proposed methods to the problem of monitoring the risk of heparin-induced thrombocytopenia (HIT) [Warkentin et al., 2000].

3.5.1 HIT and HIT data set

3.5.1.1 HIT HIT is an adverse immune reaction that may develop if a patient is treated with heparin for a longer period of time. If the condition is not detected and treated promptly it may lead to further complications, such as thrombosis, and even to death. An important problem is the monitoring and detection of patients who are at risk of developing the condition. We investigate the possibility of building a HIT alert model from patient data using the assessment of HIT by an expert. This corresponds to the problem of learning a binary classification model from data.

3.5.1.2 HIT data The data used in the experiments were extracted from the Post-Surgical Cardiac (PCP) database [Hauskrecht et al., 2010, Hauskrecht et al., 2013] of EHRs of 4486 postsurgical cardiac patients. The extracted data consisted of over 51,000 patient-state instances obtained from EHRs using 24-hour segmentation procedure proposed in Hauskrecht et al. [Hauskrecht et al., 2010]. Out of these we selected 377 instances that were labeled (independently) by three clinical pharmacists with respect to HIT. The instances for the study were selected using a special stratified sampling approach aimed to increase the proportion of HIT alert instances the expert would agree with in the data set. For example, one stratum with a larger proportion of positives was built using patient instances for which an HPF4 test (that is used to confirm the HIT) was ordered in next 24 hours. The strata covered the full instance space and the sampling was biased to strata with expected higher proportions of positive instances. All 377 examples sampled by this procedure were recorded with weights reflecting how likely they would be if they were obtained by an unbiased random

sampling process. The weights let us correct biases due to stratified selection. Please note, that the reason for introducing stratified sampling for labeling the patient instances was that the data and their labels were intended also for other related projects and the aim of one of them was to obtain and analyze a larger sample of positive HIT alerts. Given the fixed review budget the stratification of patient instances and stratified sampling was the best option to achieve our goals.

3.5.1.3 Temporal feature extraction The EHR consists of complex multivariate time series data that reflect sequences of lab values, medication administrations, procedures performed, etc. In order to use these for building HIT prediction models, a small set of temporal features representing well the patient state with respect to HIT for any time t is needed. However, finding a good set of temporal features is an extremely challenging task [Hauskrecht and Fraser, 1998, Hauskrecht and Fraser, 2000, Batal et al., 2009, Combi et al., 2010, Batal et al., 2011, Batal et al., 2012, Batal et al., 2013]. Briefly, the clinical time series, are sampled at irregular times, have missing values, and their length may vary depending on the time elapsed since the patient was admitted to the hospital. All these make the problem of summarizing the information in the time series hard. In this work, we address the above issues by representing the patient state at any (segmentation) time t using a subset of pre-defined temporal feature mappings proposed by Hauskrecht et al [Hauskrecht et al., 2010, Valko and Hauskrecht, 2010, Hauskrecht et al., 2013], that let us convert patient's information known at time t to a fixed length feature vector. The feature mappings define temporal features such as last observed platelet count value, most recent platelet count trend, or, the length of time the patient is on medication, etc. We used feature mappings for five clinical variables useful for the detection of HIT: Platelet counts, Hemoglobin levels, White Blood Cell Counts, Heparin administration record, Major heart procedure. The full list of features generated for these variables is listed in Table 6 in Appendix A.1. Briefly, temporal features for numeric lab tests: Platelet counts, Hemoglobin levels and White Blood Cell Counts used feature mappings illustrated in Figure 19 plus additional features representing the presence of last two values, and pending test. The heparin features summarize if the patient is currently on the heparin or not, and the timing of the administration, such

as the time elapsed since the medication was started, and the time since last change in its administration. The heart procedure features summarize whether the procedure was performed or not and the time elapsed since the last and first procedure. The feature mappings when applied to EHR data let us map each patient instance to a vector of 50 features. These features were then used to learn the models in all subsequent experiments. The alert labels assigned to patient instances by experts were used as class labels.

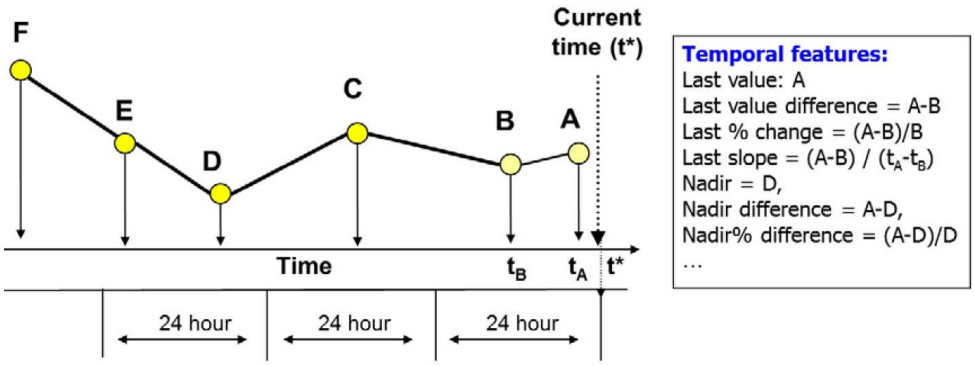

Figure 19: The figure illustrates a subset of 10 temporal features used for mapping time-series for numerical lab tests.

3.5.1.4 HIT data assessment For each patient instance, we asked three experts in clinical pharmacy the following two questions:

- **Question 1**. How strongly the clinical evidence indicates that the patient is at risk of HIT? The answer was as a numeric score in the range from 0 to 100, which we interpreted as a probabilistic score by converting it to interval [0,1].
- **Question 2**. Assume you have received a HIT alert for this patient. Please indicate to what extent do you agree/disagree with the alert? The answer was one of the four ordinal categories: "strongly-disagree", "weakly-disagree", "weakly-agree", "strongly-agree".

We used answers to Question 2 to define the binary class label "Agree with alert". Briefly, the label is positive if the expert agrees (weakly or strongly) with the alert in Question 2, otherwise it is negative. Our ultimate objective was to build a classification model that predicts this binary class label. Answers to questions 1 and 2 can be then viewed as its soft label refinement: answers to question 1 define probabilistic labels, and answers to question 2 ordinal category labels. In order to make the qualified judgment, the experts were able

to see EHR up to the time of the alert assessment, which is what the experts would see if the instances were encountered prospectively. Questions 1 and 2 were asked, and the answers to both questions were recorded on the same electronic input form. The answers were submitted at the same time by the expert by pressing the submit button on the form. We note that inclusion of both questions on the same form and their specific ordering could have influenced their respective answers. However, at this point we do not have any evidence (for or against) to believe this skewed the results in any significant way.

Table 2: Basic statistics for the auxiliary soft label information collected from the experts. Each expert labels 377 patient instances. The second and third columns show the distribution of probabilities for positive and negative examples. The remaining columns show the counts of strong and weak subcategories for positive and negative examples.

	Mean (std) probability for positives	Mean (std) probability for negatives	Number of strong positives	Number of weak positives	Number of weak negatives	Number of strong negatives
Expert 1	0.51 (0.16)	0.14 (0.17)	3	85	141	148
Expert 2	0.59 (0.06)	0.28 (0.16)	0	53	235	89
Expert 3	0.48 (0.13)	0.39 (0.14)	1	105	181	90

Table 2 shows the basic statistics related to auxiliary soft label information collected from the experts and used in the experiments. The second and third columns are related to probabilistic label information; the remaining columns to the ordinal category labels. We see that while the strong HIT alert (strong positive) option was used only rarely by the experts, strong and weak no-alerts are quite frequent and likely to be useful for learning a good discriminative model.

Appendix A.2 gives detailed agreement matrices for all pairs of experts, and their corresponding agreement, kappa [Cohen, 1960] and weighted kappa [Cohen, 1968] statistics. Fleiss' kappa [Fleiss et al., 1971] statistics is used to assess the agreement for all three experts combined. Briefly, for binary labels the pairwise kappa ranges in between 0.33 and 0.57, while Fleiss' kappa is 0.47. For ordinal categorical labels the weighted kappa for pairwise analysis ranges in between 0.31 and 0.51, and Fleiss' kappa is 0.34.

3.5.2 Experimental setup

We evaluated the performance of the soft-label classification methods using the Area under ROC (AUC) score [Hanley and McNeil, 1982]. To perform the evaluation, the data set of 377 examples with weights (reflecting the stratification) was first split randomly (by ignoring the weights) using a 2:1 ratio into the disjoint training and testing groups. This split assured that instances in the training group were used for training the models, while instances in the testing group for their evaluation. The weights associated with the instances were then used to subsample the two groups in order to generate the training and test sets. This process lets us generate un-biased and non-overlapping training and testing data sets. The process (data splitting and sub-sampling of data according to weights) was repeated 100 times and the average AUC and 95% confidence interval on the test set were calculated.

To investigate the impact of the soft label information on the sample complexity we trained the new models that accept soft label information with data of different sizes and compared them to models that learn from the binary "Agree with alert" and "Disagree with alert" labels only. In all experiments, the soft label information was used only to aid the training (learning) process, and it was never used to test the binary model. The training sizes were varied from 20 to 250. The upper limit (250) was chosen because the performance of the methods at that point stabilized and further increases in the training size did not result in any significant changes. We performed two experiments testing the impact of learning with auxiliary probabilistic labels (Experiment 1), and ordinal categorical labels (Experiment 2). Table 3 summarizes all methods used for these experiments.

L1 regularization penalty was used to implement Q(w) for all models. The constants C and B for SVM-based ranking models were optimized using 3-fold cross-validation approach. Please note that because the number of samples in strong positive subcategory is very small, the multiclass method is trained only on three subcategories (positives, weak negatives and strong negatives).

Table 3: Methods used in experiments with probabilistic soft-labels (experiment 1) and ordinal categorical soft-labels (experiment 2). Note: in the "Labels used" column, "prob" and "categ" mean probabilistic and categorical soft-labels, respectively.

Method	**Short description**	**Labels used**			**Experiment**	
		Class	Prob	Categ	1	2
SVM (baseline)	Standard binary SVM	+			Y	Y
LogR (baseline)	Standard logistic regression	+			Y	Y
Multiclass (baseline)	Soft-max regression model [Lin et al., 2007] that learns from categorical labels, but ignores the ordinal information			+		Y
LinRaux	Linear regression model with probabilistic soft-labels (section 3.2.1.1)		+		Y	
SvmAuxPair	The new ranking model with order constraints on all example pairs (section 3.2.3)	+	+	+	Y	Y
SvmAuxBinPair	The new ranking model that discretizes examples into 5 bins and enforces constraints only between examples of different bins (section 3.2.4.1)	+	+		Y	
SvmAuxOrd	The new ranking model that enforces constraints between examples and boundaries of bins/categories (section 3.2.4.2)	+	+	+	Y	Y

3.5.3 Results and discussion

The results of all our experiments are shown graphically in Figure 20 and 22. In addition, the same results are tabulated together with pairwise statistical significance test in Appendix A.3.

3.5.3.1 Experiment 1: learning with probabilistic soft-labels Figures 20(a),20(b), 20(c) compare the performance of different methods on HIT data and soft labels expressed in terms of probabilistic scores (answers of the experts to question 1 for expert 1, expert 2 and expert 3, respectively). The x-axis shows the number of examples the models are trained on

and the y-axis shows the AUC. The baseline methods (SVM and LogR) are illustrated using dashed lines. Methods that utilize soft labels are shown using solid lines.

The results in Figure 20 show that the simple approach utilizing the probabilistic information LinRaux is worse than classification models that learn from the binary label information only (SVM and LogR). This is true for all three experts. The new ranking approaches (SvmAuxPair, SvmAuxBinPair, SvmAuxOrd) that ignore exact probabilistic assessments, but at the same time try to preserve the relative order of patient instances, perform the best and outperforms all alternatives on two of the experts (expert 1 and 2) and are comparable to binary classifiers for expert 3.

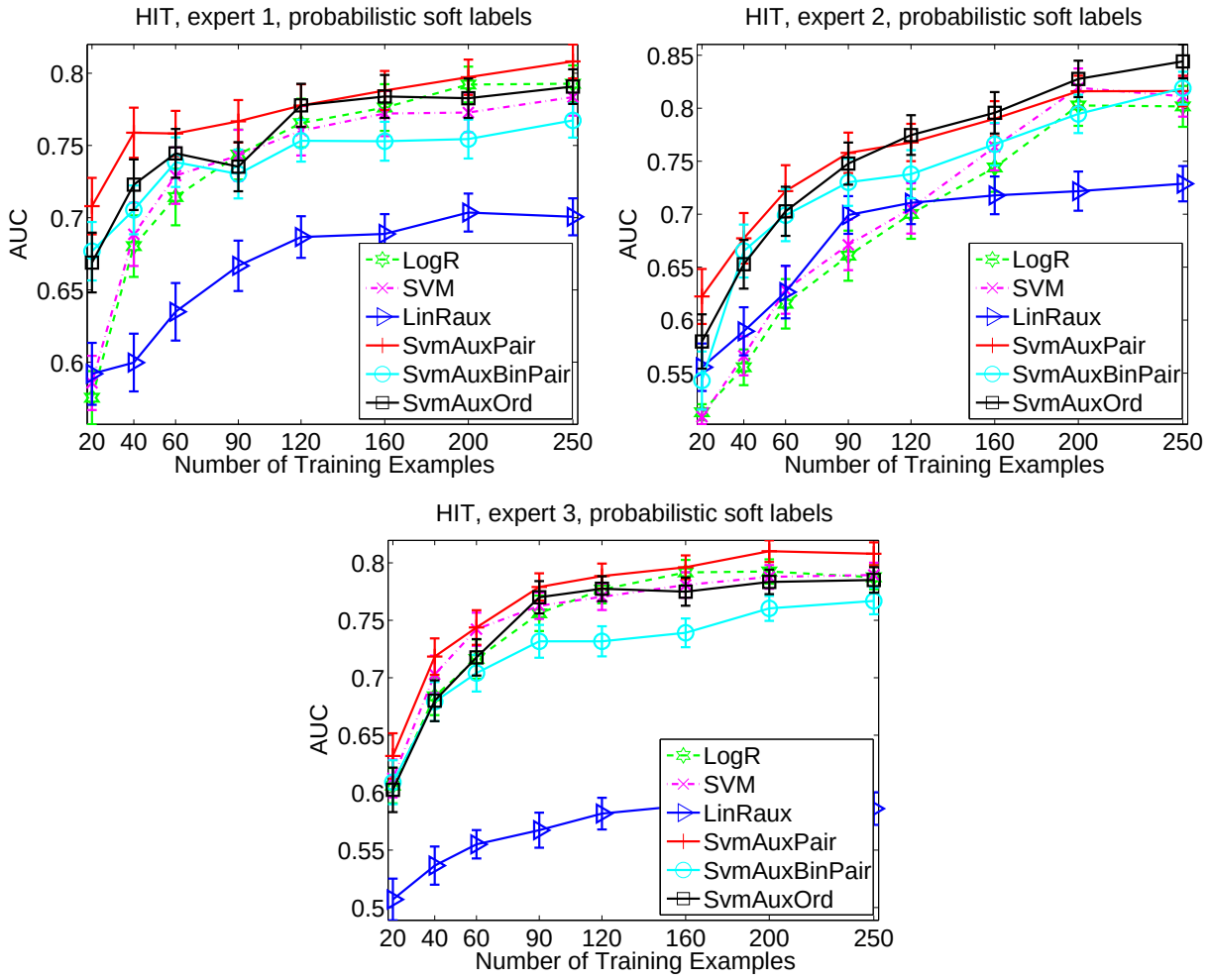

Figure 20: AUC for the different learning methods trained on probabilistic soft labels from three different experts and for the different training sample sizes.

Discussion. LinRaux method that attempts to fit probabilistic estimates to define the discriminant function perform the worst. At first glance this is somewhat surprising. However, these results can be explained by inconsistences and biases in subjective probability estimates provided by experts. The fact that subjective probability estimates are often not well calibrated is a widely documented problem in the literature [Suantak et al., 1996, Griffin and Tversky, 1992, O'Hagan et al., 2007]. Hence, it is not realistic to expect that probabilistic assessments for all instances are perfect both in absolute terms (that is, each probability assessment is a perfect estimate of the true probability) and relative terms (when pairs of instances and their probability differences are considered). This in turn may influence the model based on such assessments, especially when the model tries to 'closely' fit the estimates. Figure 21 illustrates this problem on the data in our study. It shows the distribution of probability estimates for positive and negative examples for Expert 1, 2 and 3, and overlaps of the two regions. These overlaps and inconsistencies influence the discriminatory performance of the models trained on probabilistic soft labels and translate to the results observed in Figure 20. More specifically, notice that while the probability estimates of expert 2 assigned to positive and negative labels in Figure 21 are rather well separated, the estimates of expert 1 and particularly expert 3 are much harder to separate and there is a great deal of overlap in the regions defining positive and negative labels. These differences translate to the results in Figure 20. In particular, LinRaux that learn their models only from probability estimates get more benefit from probabilistic information provided by expert 2 than expert 1 and expert 3. Our ranking models SvmAuxPair, SvmAuxBinPair and SvmAuxOrd, that rely on both probability estimates and binary labels, are more robust and able to absorb the inconsistences much better. However, please notice that for expert 3 (with the most inconsistent probability assignments) the benefit of probabilistic information diminishes and the ranking models are comparable to models that learn from the binary information only.

In summary, the new ranking approaches (SvmAuxPair, SvmAuxBiPair, SvmAuxOrd) that ignore exact probabilistic assessments, but at the same time preserve the relative order of patient instances, are more robust to noisy probabilistic estimates and outperform all alternatives, hence demonstrating the benefit of soft probabilistic labels for learning classi-

fication models.

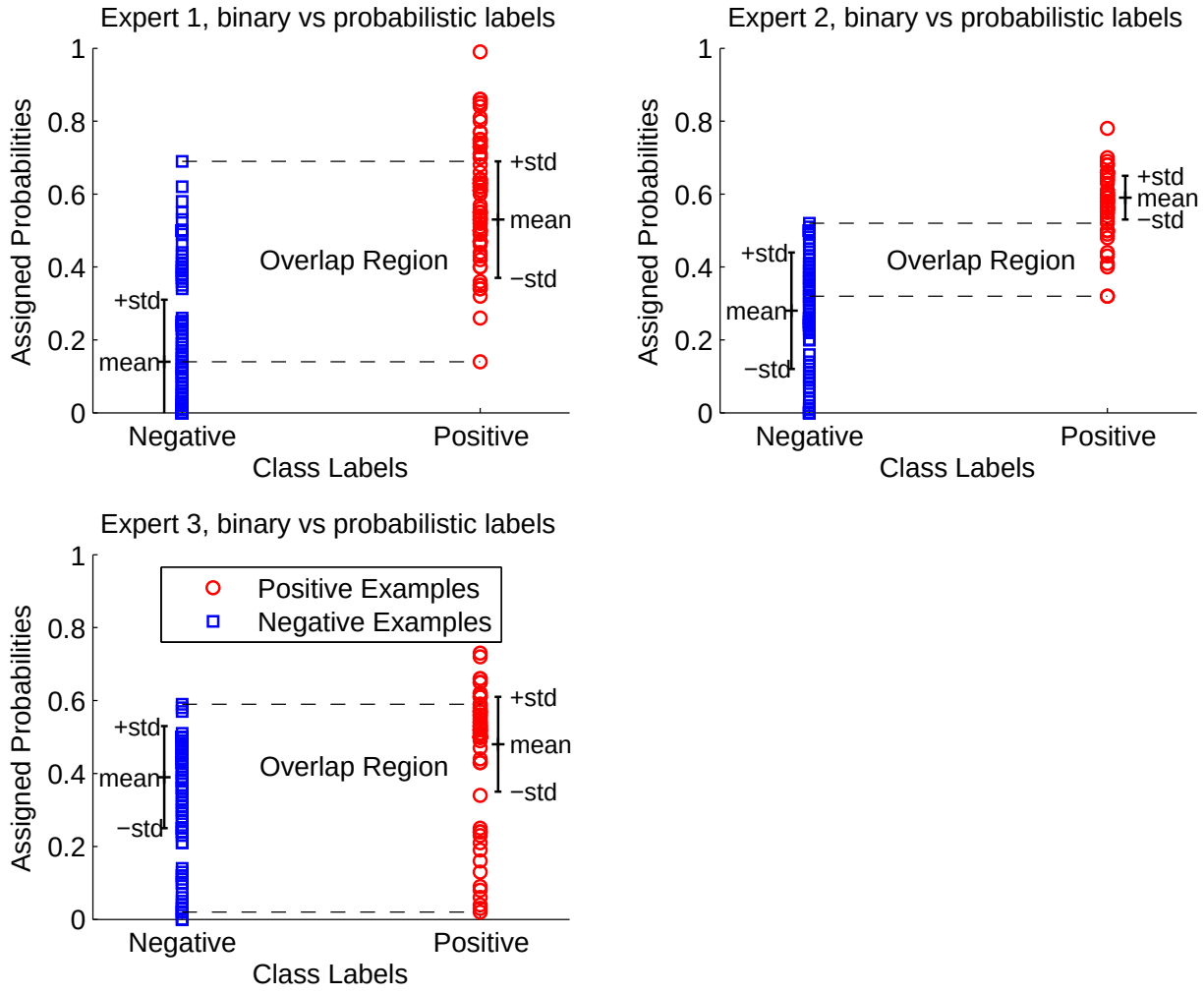

Figure 21: Distribution of probabilities assigned to negative (left) and positive (right) examples by experts 1, 2 and 3, respectively. Left and right vertical bars show mean and standard deviation of assigned probabilities to negative and positive examples, respectively. The 'Overlap Region' between horizontal dash lines is where probability estimates for positive and negative examples overlap. These inconsistencies may lead to deterioration of models trained based on such probabilities.

3.5.3.2 Experiment 2: learning with ordinal categorical soft-labels Figure 22 compares the performance of the methods on HIT data and the soft labels expressed in terms of ordinal categories: 'strongly-disagree', 'weakly-disagree', 'weakly-agree' and 'strongly-agree'

with the HIT alert. Figures 22(a), 22(b), 22(c) show the AUC of models learned for expert 1, expert 2 and expert 3, respectively.

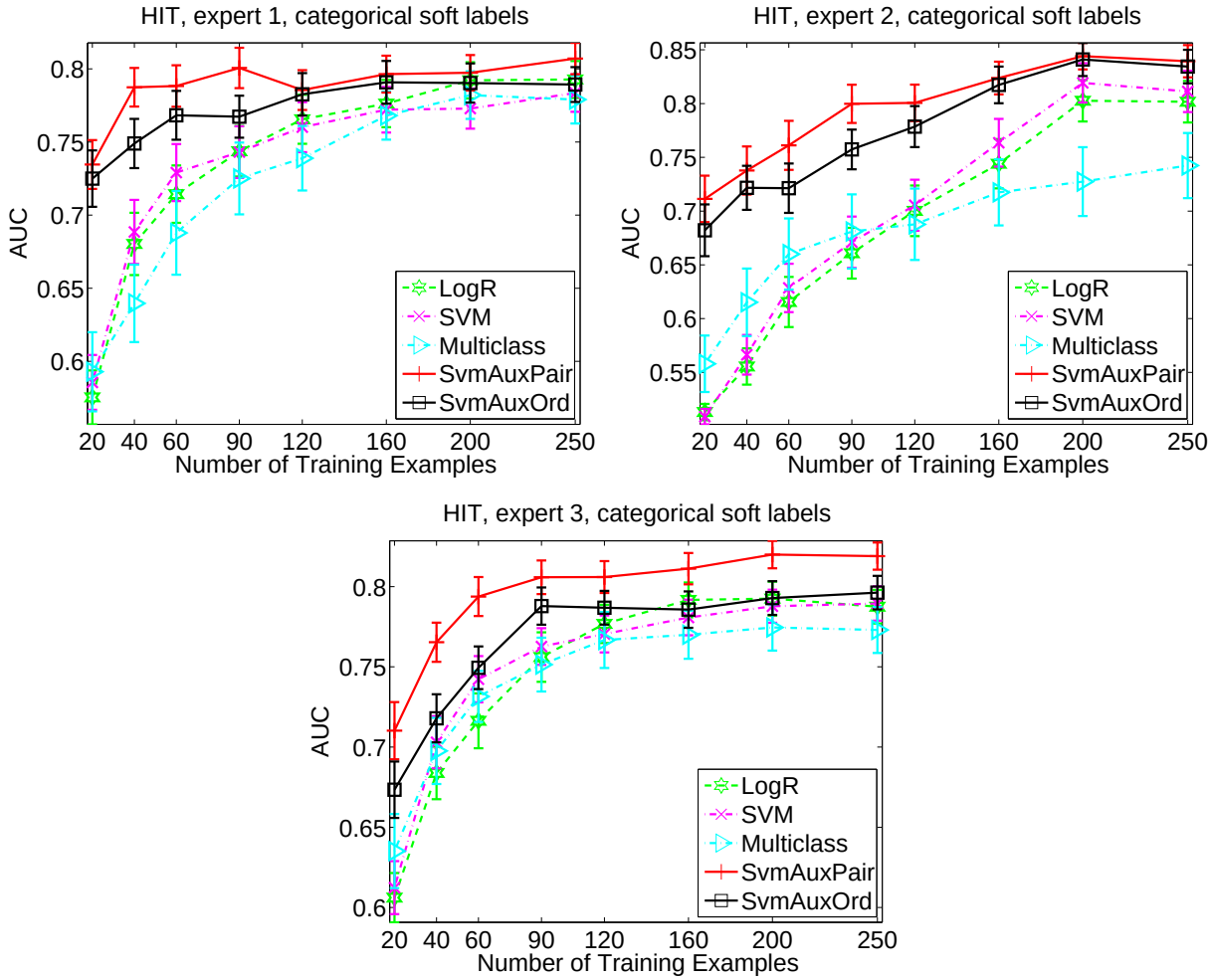

Figure 22: AUC for the different learning methods trained on ordinal categorical labels from three different experts and for the different training sample sizes.

The SvmAuxPair method, that enforces $O(N^2)$ constraints for all pairs of examples, is the best for all three experts, followed closely by the SvmAuxOrd method that enforces only $O(N)$ constraints. These two ranking methods outperform the two baselines that rely on the binary class information (SVM and LogR) and the multiclass learning method that tries to learn different categories but ignores their order.

Discussion. The results demonstrate that binary labels, when further refined to ordinal subcategories, can lead to improved classification models. In particular, splitting positive

and negative labels to strong and weak positives and negatives, and mapping them on the same discriminative function while assuring their order helped us to converge faster to a better discriminative function. However, we also observe that when these subcategories are taken in isolation (without ordering) as in the multiclass method, the performance tends to be worse. This can be attributed to the fact that multiclass models require more parameters and in general more samples are needed to fit them accurately.

The refinement of the two alert classes to four ordinal subcategories clearly helped us to learn better models for all three experts. In contrast to this, the probabilistic information in experiment 1 was helpful, but the margin of the improvement was smaller and for expert 3 who was the least consistent in assigning probabilities to examples the benefit was marginal. Our experiments with probabilistic assessments (experiment 1) given by human suggest the assessments may suffer from consistency/calibration problems which may reduce the utility of the probabilistic information for aiding the learning process. This also suggests that the utility of soft labeling may differ and vary with the resolution and the number of soft categories the expert may choose from. An interesting and open question is how many categories to use in order to benefit from the soft labeling the most.

3.6 ACTIVE LEARNING WITH SOFT-LABEL INFORMATION

Active learning is a learning technique that aims to reduce the cost of labeling by actively seeking the most informative training examples and asking for their labels ([Lewis and Gale, 1994], [Roy and McCallum,], [Tomanek and Olsson, 2009]). This is different than traditional (passive) supervised learning, where the learner randomly samples training examples from an unlabeled data set and queries for their labels. We have given an overview of active learning in Section 2.2.1. In general, an active learner works as follows:

- Step 1. Start with an initial set of unlabeled examples; query labels for this set.
- Step 2. Train a model on the labeled set.
- Step 3. Inspect unlabeled examples and select the k most informative examples based on the current model.

- Step 4. Query labels for the selected examples; add these examples to the labeled set.
- Step 5. Repeat steps 2-4 until some stopping criteria is matched, e.g. N examples (depending on the labeling budget) have been labeled, or the performance of the learned model has reached some satisfied threshold.

In this work we have proposed a new cost-efficient learning approach: learning with soft-label information. As mentioned earlier, our approach is complementary to active learning: while active learning aims to select the most useful examples to request for labeling, the soft-label learning aims to obtain more information from examples that were selected. This means that our learning approach can work together with active learning. In this section, we conduct preliminary investigation of combining auxiliary soft-label information with active learning.

3.6.1 A query strategy for active learning with soft-label information

For active learning, the most important question is the query strategy, i.e. how to find the most informative examples that could help us to learn a model faster. In practical applications and research studies, the single most commonly used strategy is uncertainty sampling ([Lewis and Gale, 1994], [Settles, 2010]). Briefly, the uncertainty strategy selects examples that the current learner is most uncertain about, i.e. examples with predictive scores closest to 0.5 or examples closest to the decision boundary. The intuition for this strategy is that the examples closest to decision boundary would most likely influence the decision if we knew their labels, therefore they are the most informative. Other strategies have been proposed in the machine learning community (Section 2.2.1), however they were designed for only either classification (query class label) or regression setting (query continuous values). In our problem, we have both class labels and auxiliary soft-labels, and none of the existing active learning works have specifically addressed this setting.

The query strategy we propose to utilize the soft-label information divides examples into different regions (intervals) based on their predictive scores given by the current learned model, and then selects examples from the regions that have the most discrepancy between predictive scores and soft-labels given by the human annotator. The intuition for this strat-

egy is that if in some region, the model and the human annotator do not agree on the labeling of examples, then acquiring more labels in that region would help us to resolve the discrepancy between the model and the human and get more useful information for learning. In contrast, if the current model and the human annotator already agree on the labeling of examples in a region, then acquiring more labels from that region would not provide more valuable information for learning.

In the following, we give the description and explanation of our strategy.

3.6.1.1 Description of the query strategy

Let us consider the following notation:

- Let U denotes the set of unlabeled examples, L the set of labeled examples that is initially empty and D the complete set of examples such that $D = U \cup L$.
- X - the feature space.
- X^D, X^L, X^U - feature vectors of all, labeled and unlabeled examples, respectively.
- $f : X \rightarrow [0,1]$ - the current model that maps input X to probabilities.
- P_h - probabilistic soft-label given by the human annotator.
- MSE - mean square error estimate.
- ΔE - confidence interval for error estimate.
- Std - standard deviation.

Given this notation, the query strategy with soft-label information is described as follows:

- **Step 1**. Select randomly an initial small subset of examples from U; query a human annotator for class and soft labels and add the labeled examples to L.
- **Step 2**. Train a model f on the current labeled set L.
  Model f can be trained using any learning algorithm, e.g. a standard binary classifier, such as SVM, or a learning method that uses soft-label information, such as SvmAuxOrd (Section 3.2.4.2).
- **Step 3**. Compute $f(X^L)$ and $f(X^U)$; divide the (0,1) interval equally into m continuous regions with n labeled examples in each region based on $f(X^L)$; distribute all examples to the regions based on $f(X^L)$ and $f(X^U)$ (see Figure 23).

In Figure 23 the values of $f(X^L)$ and $f(X^U)$ are projected onto the horizontal axis and the values of soft-labels $P_h(X^D)$ are projected onto the vertical axis. The black points are labeled examples (X^L).

- **Step 4**. For each region $i = 1..m$ compute $MSE_i = \frac{1}{n}(f(X_i^L) - P_h(X_i^L))^2$ and $\Delta E_i = 1.96 * Std[(f(X_i^L) - P_h(X_i^L))^2]/\sqrt{n}$.

  The mean square error term MSE_i determines the discrepancy between the predictive scores $f(X_i^L)$ given by the model f and the soft-label estimates $P_h(X_i^L)$ given by the human annotator h in region i. ΔE determines the 95% confidence of the MSE estimate.

- **Step 5**. Select interval i with a probability proportional to $MSE_i + \Delta E_i$; sample randomly an example in the selected interval and query the human annotator for its class and soft labels; add the labeled example to the labeled set L.

  In this step we prefer to select the intervals that have the more discrepancy between the predictive scores $f(X^L)$ and the soft-labels $P_h(X^L)$. Note that in Figure 23, more preferred intervals, e.g. region 2 and 3 , would have labeled examples spread out further from the line connecting points {0,0} and {1,1}. The intuition for this selection strategy is that if in some region there is much discrepancy between the learned model f and the human annotator h then acquiring more labeled examples in that region would help us to get more information to resolve this discrepancy, and therefore have a better revision of the model f in the next learning iteration. In contrast, if in some region the learned model f and the human h already agree on the labeling (e.g. region 4 in Figure 23) then we probably do not want to put more labeling effort to explore that region further.

- **Repeat steps 2-5** until a stopping criteria is matched.

In this work, we refer to this query strategy as to **SLDiscr** (Soft Label Discrepancy).

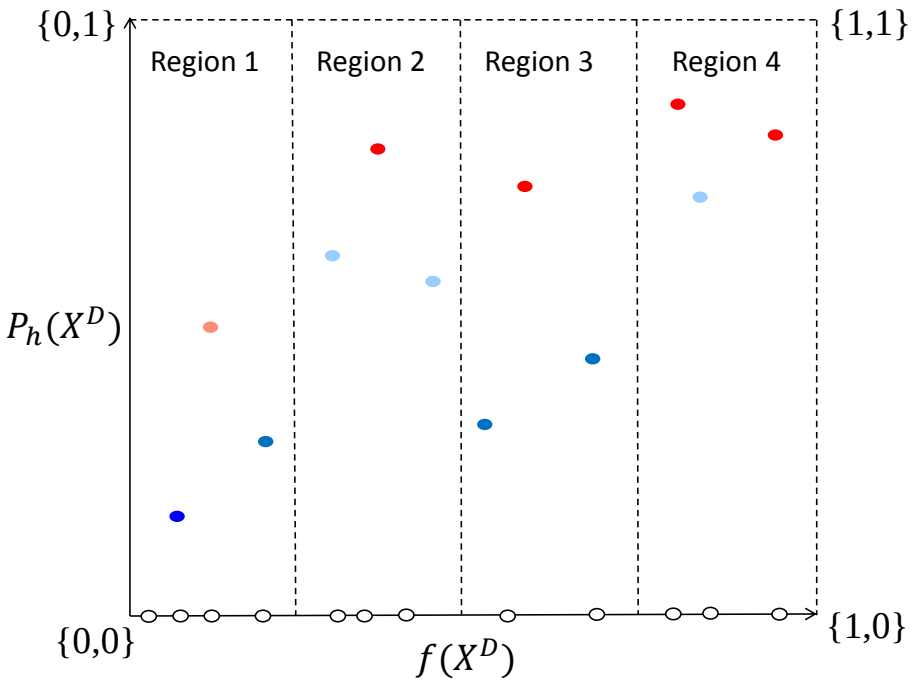

Figure 23: The query strategy (SLDiscr) that uses soft-label information. The black points are labeled examples. The x-coordinate represents the probability of the example as estimated by the current model, while the y-coordinate shows the probability that is assigned to it by a human.

3.6.2 Experiments

In our preliminary investigation we have conducted two sets of experiments, one on UCI data sets and one on our HIT data, to investigate how active learning may work together with soft-label information. In general, active learning strategies may differ in what algorithm should be used for training model and what query strategy should be utilized to select the most informative examples.

For training model we test two learning algorithms: (1) standard SVM and (2) SvmAux-Ord - the proposed ranking method that distributes examples to discrete bins and enforces $O(N)$ optimization constraints between examples and bin boundaries (Section 3.2.4.2).

For query strategy we test three approaches: (1) Random selection, (2) Uncertainty sampling, and (3) SLDiscr - the query strategy that aims to resolve the discrepancy between the

soft-label information and the model prediction.

There are two learning algorithms and three query strategies, so in total we have six different methods. Table 4 summarizes the six methods that we compare in experiments.

Table 4: Methods for the experiments of learning with soft-labels in active learning setting.

Method	**Use soft label for training**	**Use soft label for query**
SVM Random	No	No
SVM Uncertainty	No	No
SVM SLDiscr	No	Yes
SvmAuxOrd Random	Yes	No
SvmAuxOrd Uncertainty	Yes	No
SvmAuxOrd SLDiscr	Yes	Yes

3.6.2.1 Experiments on UCI data sets Similar to experiments in Section 3.4, we compare the above six methods on five UCI data sets ("ailerons", "concrete", "bank8", "housing" and "pol") with three different levels of noise in the soft-label: weak, moderate and strong. Figure 24 shows the results for the full set of experiments. We have the following observations.

First, regardless of the query strategy, methods that use soft-label for training model (SvmAuxOrd-Random, SvmAuxOrd-Uncertainty, SvmAuxOrd-SLDiscr) outperform methods that do not use soft-label for training (SVM-Random, SVM-Uncertainty, SVM-SLDiscr). This suggests that the benefit of soft-label information alone (when models are trained on randomly sampled examples) is greater than the benefit of active learning when it is applied in binary label settings.

Second, all methods that use auxiliary soft-label information for training the model (SvmAuxOrd-Random, SvmAuxOrd- Uncertainty, SvmAuxOrd-SLDiscr), whether the examples used for training were selected by active learning or random sampling, have comparable performance. This suggests that the combination of active learning and soft-label information may not be as beneficial as we hoped for. However, we would like to note that we have proposed and tried only one query strategy (SLDiscr) and it is possible that a better query strategy for active learning that utilizes soft-label information exists. Hence, the investigation of the combination of the two approaches and its benefits remains an open research question.

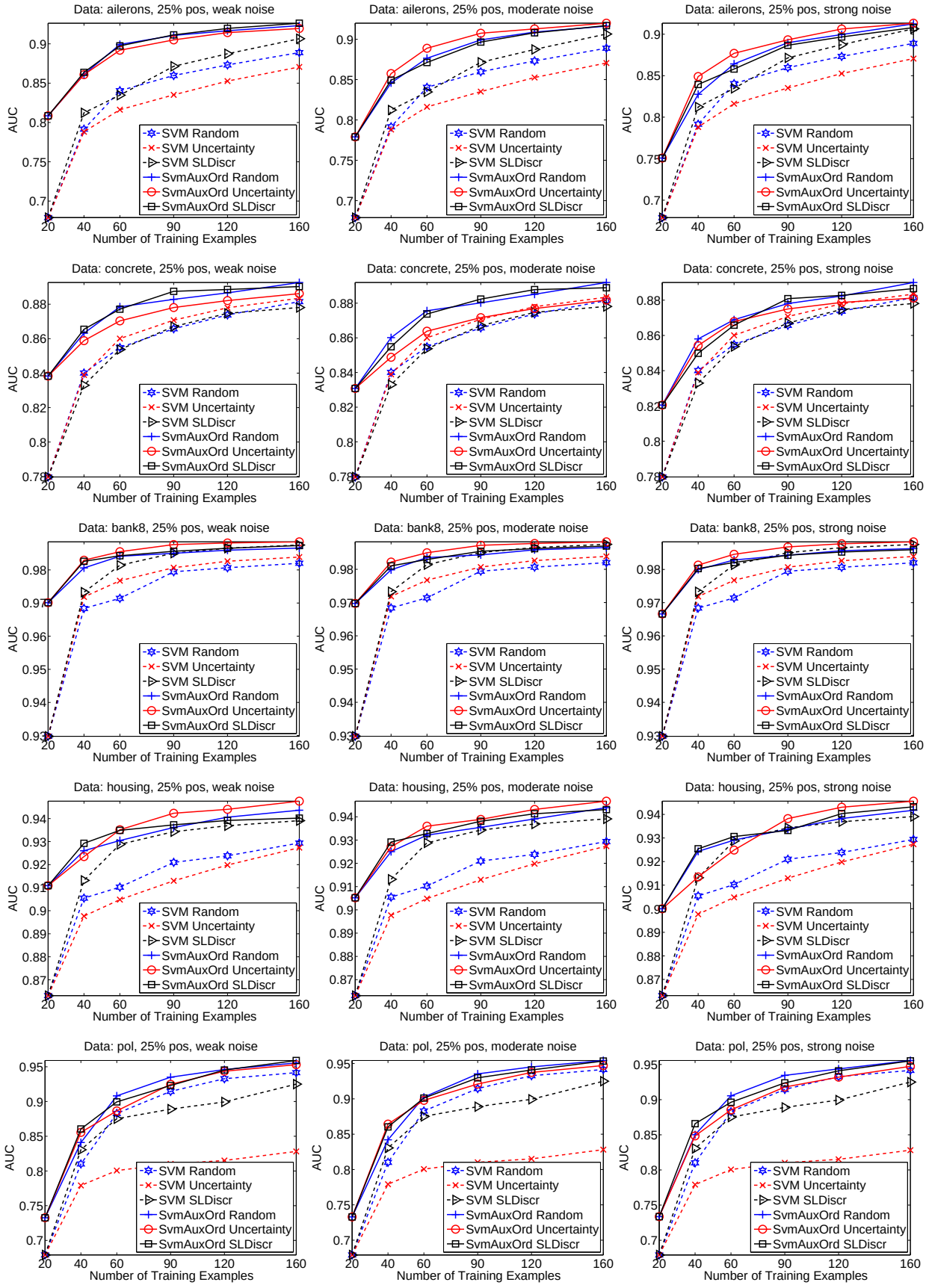

Figure 24: AUC vs. sample size for different learning methods and query strategies trained on data with auxiliary soft-label information corrupted by three levels of noise: left column - weak noise (5%), middle - moderate noise (15%), right - strong noise (30%). One row of figures for each data set.

3.6.2.2 Experiments on HIT data set In this section we compare different methods on HIT data (3.5.1). Figures 25 and 26 show the results for learning with probabilistic soft-label and categorical soft-label, respectively. The results confirm the observations made on UCI data sets (see above). In particular:

First, methods SvmAuxOrd-SLDiscr and SvmAuxOrd-Random, that use soft-label for training model, outperform all methods that do not use soft-label for training (SVM-Random, SVM-Uncertainty, SVM-SLDiscr). This confirms that a passive learning model but trained with soft-label, outperforms an active learning model trained on binary labels only.

Second, SvmAuxOrd-SLDiscr outperforms SvmAuxOrd-Random for expert 1 but they are comparable for the other two experts. This observation suggests that once we use soft-label for training model, the random sampling strategy may be good enough to learn a high-quality model.

In addition, note that methods with uncertainty strategy do not perform well, especially for experts 1 and 3. This is because for experts 1 and 3, the overlapping regions of probabilistic soft-label estimates are large (as shown in Figure 21), which make the uncertainty strategy struggle to find a good decision boundary and determine the closest examples to it.

3.7 SUMMARY

Making use of many real-world data sets often prompts one to fill additional information with subjective human labels. However, this process is often very time consuming and costly, so different ways of reducing the labeling cost need to be sought. In this chapter we investigate a new framework for reducing this cost by reducing the number of examples one must label. The trick is to use auxiliary soft-label information that reflects how strongly the human annotator believes in the class label, which can be extracted quickly and virtually at no additional cost.

We proposed multiple methods that use this information to make the learning more sample-efficient. First, we introduced regression-based methods that learn directly from soft-label information. These methods outperform traditional binary classification methods

when there is little noise in the soft-labels. However, they rely primarily on exact estimates of auxiliary labels, so their performance quickly degrades when the level of noise increases. Since in practice, human subjective estimates are often noisy and inconsistent, we proposed new methods that are more robust to noise. These methods are based on ranking methodology and support vector machines, that learn from both class and auxiliary soft-labels.

In general, auxiliary soft labels may be present in the form of probabilities or qualitative ordinal categories. We proposed different regression-based and SVM-based methods to deal with each of this case. We evaluated our methods on five UCI data sets and our clinical data set. The experimental results shown that our methods significantly outperform traditional binary classifiers in both cases: when soft labels are probabilities, and when they are qualitative categories. This confirms that the proposed learning with auxiliary soft labels framework can help to build better classification models with lower cost.

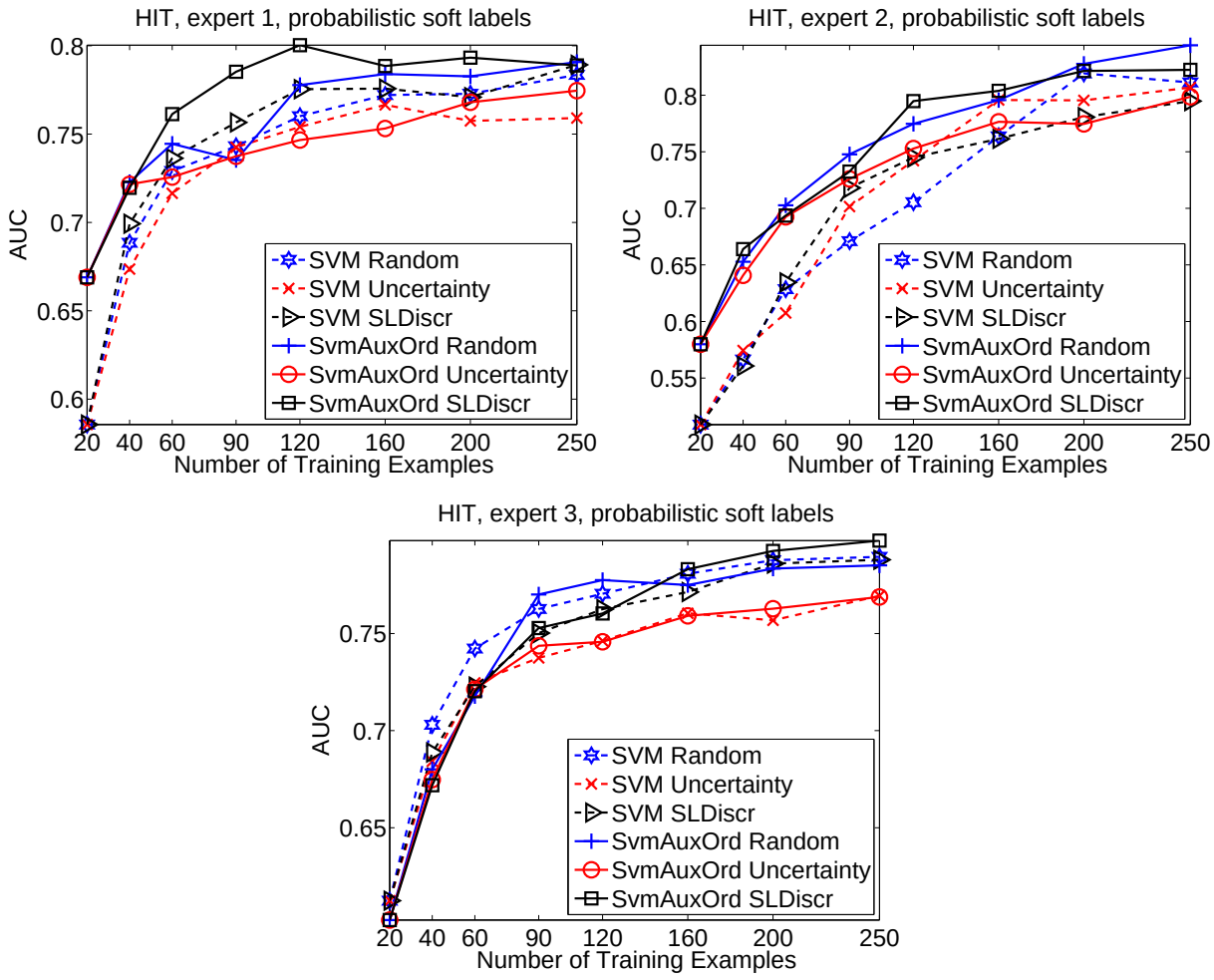

Figure 25: AUC for different learning methods and query strategies trained on probabilistic soft labels from three different experts and for different training sample sizes.

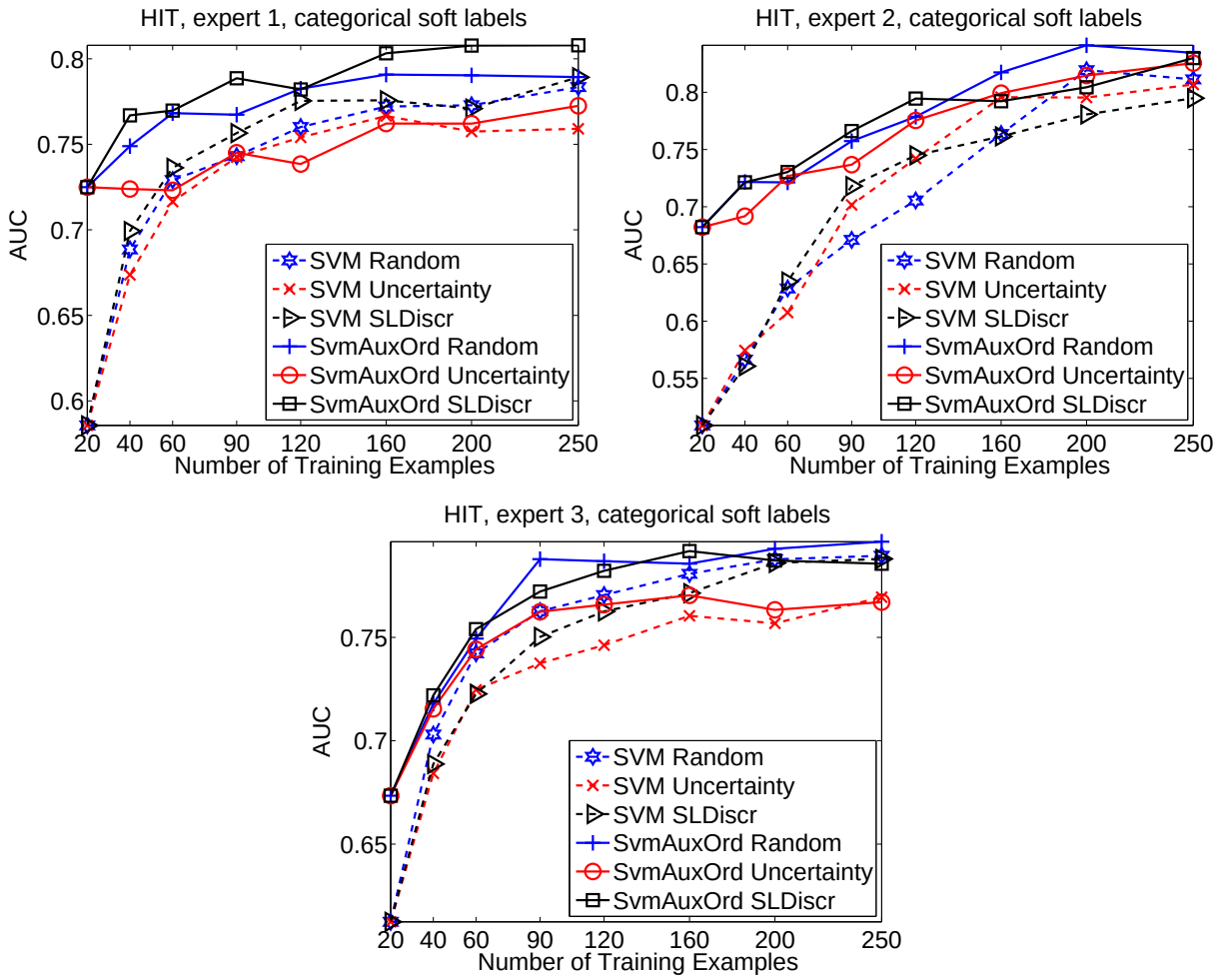

Figure 26: AUC for different learning methods and query strategies trained on ordinal categorical labels from three different experts and for different training sample sizes.

4.0 LEARNING WITH MULTIPLE ANNOTATORS

In this chapter we describe a new multiple-annotator learning approach that combines information obtained from different annotators/experts to learn a consensus model, that generalizes well to future unseen data. Moreover, our method also learns and produces models for individual annotators, which are useful for modeling/predicting behavior of specific annotators.

Our approach takes into account the annotator reliability as well as differences in the annotator-specific models when learning a consensus model. These considerations allow us to identify different characteristics of the annotators, including bias, domain expertise, and carefulness during the label assignment process. We study our framework on public UCI medical data sets and on our real-world multiple expert learning problem in medical domain. First, for the UCI data we start from the ground consensus model and show that we can recover it accurately from simulated experts' labels that may differ because of the expert-specific parameters. Second, we demonstrate benefits of our approach on medical data representing experts assessment of the risk of the Heparin Induced Thrombocytopenia (HIT).

4.1 INTRODUCTION

The standard machine learning framework assumes the class labels are assigned to instances by a uniform labeling process. However, in many practical applications the labels may come from different annotators with different domain knowledge and interpretation of data. For example, in medical domain, a patient can be diagnosed by a group of physi-

cians with different experience levels, or in natural language processing, an article can be annotated by people with different background and education. In this work we take medical domain as the running example to discuss our multiple-annotator learning framework. However, in general, our framework can be applied to learn predictive models in other domains.

Let us consider the multiple-annotator learning problem in the context of disease diagnosis. Class labels are given to patient instances by physicians/experts, and the goal is to learn a classification model that correctly predict class label for future patients. Typically the class labels are either acquired (1) during the patient management process and represent the decision of the human expert that is recorded in the Electronic Health Records (EHR), say, diagnosis, or (2) retrospectively during a separate annotation process based on past patient data. In the first case, there may be different physicians who manage different patients, hence the class labels naturally originate from multiple experts. Whilst in the second (retrospective) case, the class label can in principle be provided by one expert, the constraints on how much time a physician can spend on patient annotation process often requires to distribute the load among multiple experts.

Accepting the fact that labels are provided by multiple experts, the complication is that different experts may have different subjective opinion about the same patient case. The differences may be due to experts' knowledge, subjective preferences and utilities, and expertise level. This may lead to disagreements in their labels, and variation in the patient case labeling due to these disagreements. However, we would like to note that while we do not expect all experts to agree on all labels, we also do not expect the expert's label assessment to be random; the labels provided by different experts are closely related by the condition (diagnosis, an adverse event) they represent.

Given that the labels are provided by multiple experts, two interesting research questions arise. The first question is whether there is a model that would represent well the labels the group of experts would assign to each patient case. We refer to such a group model as to the (group) consensus model. The second question is whether it is possible to learn such a consensus model purely from label assessments of individual experts, that is, without access to any consensus/meta labels, and do this as efficiently as possible.

To address the above issues, we propose a new multi-expert learning framework that starts from data labeled by multiple experts and builds: (1) a *consensus model* representing the classification model the experts collectively converge to, and (2) *individual expert models* representing the class label decisions exhibited by individual experts. Figure 27 shows the relations between these two components: the experts' specific models and the consensus model. We would like to emphasize again that our framework builds the consensus model without access to any consensus/meta labels.

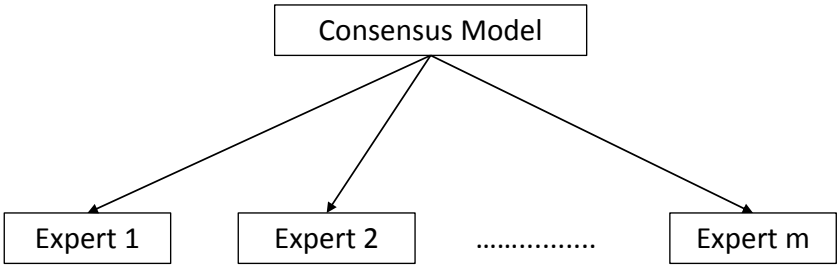

Figure 27: The consensus model and its relation to individual expert models.

To represent relations among the consensus and expert models, our framework considers different sources of disagreement that may arise when multiple experts label a case and explicitly represents them in the combined multi-expert model. In particular our framework assumes the following sources for expert disagreements:

- **Differences in the risks annotators associate with each class label assignment:** diagnosing a patient as not having a disease when the patient has disease, carries a cost due to, for example, a missed opportunity to treat the patient, or longer patient discomfort and suffering. A similar, but different cost is caused by incorrectly diagnosing a patient. The differences in the expert-specific utilities (or costs) may easily explain differences in their label assessments. Hence our goal is to develop a learning framework that seeks a model consensus, and that, at the same time, permits experts who have different utility biases.
- **Differences in the knowledge (or model) experts use to label examples:** while diagnoses provided by different experts may be often consistent, the knowledge they have and features they consider when making the disease decision may differ, potentially

leading to differences in labeling. It is not rare when two expert physicians disagree on a complex patient case due to differences firmly embedded in their knowledge and understanding of the disease. These differences are best characterized as differences in their knowledge or model they used to diagnose the patient.

- **Differences in time annotators spend when labeling each case:** different experts may spend different amount of time and care to analyze the same case and its subtleties. This may lead to labeling inconsistency even within the expert's own model.

4.2 FORMAL PROBLEM DESCRIPTION

In the standard classification learning, we have training data set $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ consisting of n data examples, where $\mathbf{x}_i$ is a d-dimensional feature vector and y_i is the corresponding binary class label. The objective is to learn a classification function: $f : \mathbf{x} \rightarrow y$ that generalizes well to future data.

In the multiple-annotator learning setting, we have m different annotators/experts who assign labels to examples. Let $D^k = \{(\mathbf{x}_i^k, y_i^k)\}_{i=1}^{n_k}$ denotes training data specific for the expert k, such that $\mathbf{x}_i^k$ is a d-dimensional input example and y_i^k is the binary label assigned by expert k. Given the data from multiple experts, our main goal is to learn the classification mapping: $f : \mathbf{x} \rightarrow y$ that would generalize well to future examples and would represent a good consensus model for all these experts. In addition, we can learn the expert specific classification functions $g_k : \mathbf{x} \rightarrow y^k$ for all $k = 1, \ldots, m$ that predicts as accurately as possible the label assignment for that expert. The learning of f is a difficult problem because (1) the experts' knowledge and reliability could vary, and (2) each expert can have different preferences (or utilities) for different labels, leading to different biases towards negative or positive class. Therefore, even if two experts have the same relative understanding of a patient case their assigned labels may be different. Under these conditions, we aim to combine the subjective labels from different experts to learn a good consensus model.

4.3 METHODOLOGY

We aim to combine data labeled by multiple experts and build (1) a unified consensus classification model f for these experts and (2) expert-specific models g_k, for all $k = 1, \cdots, m$ that can be applied to future data. Figure 28 illustrates the idea of our framework with linear classification models. Briefly, let us assume a linear consensus model f with parameters (weights) $\mathbf{u}$ and b from which linear expert-specific models g_ks with parameters $\mathbf{w}_k$ and b_k are generated. Given the consensus model, the consensus label on example $\mathbf{x}$ is positive if $\mathbf{u}^\top \mathbf{x} + b \geq 0$, otherwise it is negative. Similarly, the expert model g_k for expert k assigns a positive label to example $\mathbf{x}$ if $\mathbf{w}_k^\top \mathbf{x} + b_k \geq 0$, otherwise the label is negative. To simplify the notation in the rest of the chapter, we include the bias term b for the consensus model in the weights vector $\mathbf{u}$, the biases b_k in w_ks, and extend the input vector $\mathbf{x}$ with constant 1.

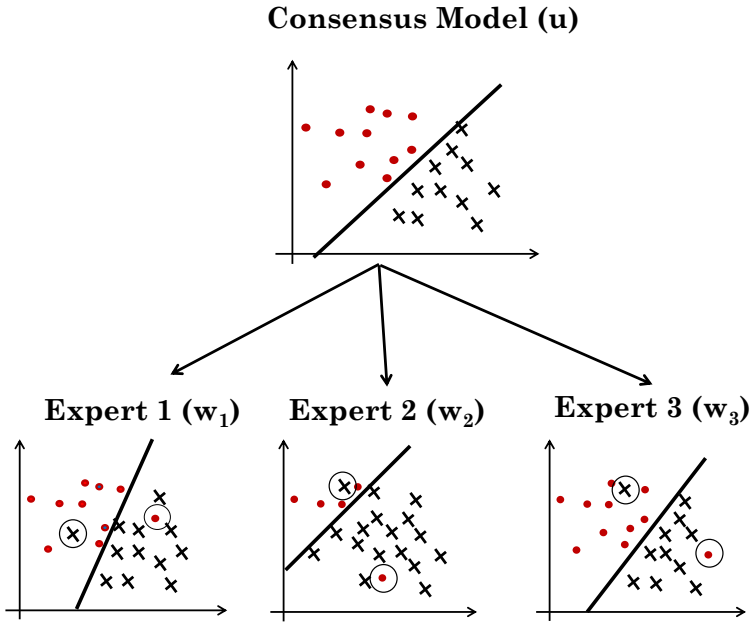

Figure 28: The experts' specific linear models $\mathbf{w}_k$ are generated from the consensus linear model $\mathbf{u}$. The circles show instances that are mislabeled with respect to individual expert's models and are used to define the model self consistency.

The consensus and expert models in our framework and their labels are linked together using two reliability parameters:

1. α_k: the self-consistency parameter that characterizes how reliable the labeling of expert k is; it is the amount of consistency of expert k within his/her own model $\mathbf{w}_k$.
2. β_k: the consensus-consistency parameter that models how consistent the model of expert k is with respect to the underlying consensus model $\mathbf{u}$. This parameter models the differences in the knowledge or expertise of the experts.

We assume, all deviations of the expert specific models from the consensus model are adequately modeled by these expert-specific reliability parameters. In the following we present the details of the overall model and how reliability parameters are incorporated into the objective function.

4.3.1 Multiple Experts Support Vector Machines (ME-SVM)

Our objective is to learn the parameters $\mathbf{u}$ of the consensus model and parameters $\mathbf{w}_k$ for all expert-specific models from the data. We combine this objective with the objective of learning the expert specific reliability parameters α_k and β_k. We have expressed the learning problem in terms of the objective function based on the max-margin classification framework [Scholkopf and Smola, 2001, Valizadegan and Jin, 2007] which is used, for example, by Support Vector Machines. However, due to its complexity we motivate and explain its components using an auxiliary probabilistic graphical model that we later modify to obtain the final max-margin objective function.

Figure 29 shows the probabilistic graphical model representation [Bishop, 2006, Koller and Friedman, 2009] that refines the high level description presented in Figure 28. Briefly, the consensus model $\mathbf{u}$ is defined by a Gaussian distribution with zero mean and precision parameter η as:

$$p(\mathbf{u}|\mathbf{0}_d,\eta) \quad = \quad \mathcal{N}(\mathbf{0}_d,\eta^{-1}\mathbf{I}_d) \tag{4.1}$$

where $\mathbf{I}_d$ is the identity matrix of size d, and $\mathbf{0}_d$ is a vector of size d with all elements equal to 0.

The expert-specific models are generated from a consensus model $\mathbf{u}$. Every expert k has his/her own specific model $\mathbf{w}_k$ that is a noise corrupted version of the consensus model $\mathbf{u}$;

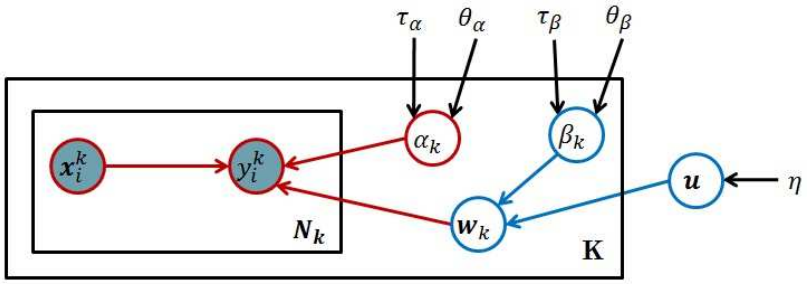

Figure 29: Graphical representation of the auxiliary probabilistic model that is related to our objective function. The circles in the graph represent random variables. Shaded circles are observed variables, regular (unshaded) circles denote hidden (or unobserved) random variables. The rectangles denote plates that represent structure replications, that is, there are k different expert models $\mathbf{w}_k$, and each is used to generate labels for N_k examples. Parameters not enclosed in circles (e.g. η) denote the hyperparameters of the model.

that is, we assume that expert k, $\mathbf{w}_k$, is generated from a Gaussian distribution with mean $\mathbf{u}$ and an expert-specific precision β_k:

$$p(\mathbf{w}_k|\mathbf{u},\beta_k) = \mathcal{N}(\mathbf{u},\beta_k^{-1}\mathbf{I}_d),$$

The precision parameter β_k for the expert k determines how much $\mathbf{w}_k$ differs from the consensus model. Briefly, for a small β_k, the model $\mathbf{w}_k$ tends to be very different from the consensus model $\mathbf{u}$, while for a large β_k the models will be very similar. Hence, β_k represents the consistency of the reviewer specific model $\mathbf{w}_k$ with the consensus model $\mathbf{u}$, or, in short, consensus-consistency.

The parameters of the expert model $\mathbf{w}_k$ relate examples (and their features) $\mathbf{x}$ to labels. We assume this relation is captured by the regression model:

$$p(y_i^k|\mathbf{x}_i^k,\mathbf{w}_k,\alpha_k) = \mathcal{N}(\mathbf{w}_k^\top\mathbf{x}_i^k,\alpha_k^{-1})$$

where α_k is the precision (inverse variance) and models the noise that may corrupt expert's label. Hence α_k defines the self-consistency of expert k.

Please also note that although y_i^k is binary, similar to [Evgeniou and Pontil, 2004] and [Zhang and Yeung, 2010], we model the label prediction and related noise using the Gaussian distribution. This is equivalent to using the squared error loss as the classification loss.

We treat the self-consistency and consensus-consistency parameters α_k and β_k as random variables, and model their priors using Gamma distributions. More specifically, we define:

$$\begin{aligned} p(\beta_k|\theta_\beta,\tau_\beta) &= \mathcal{G}(\theta_\beta,\tau_\beta), \\ p(\alpha_k|\theta_\alpha,\tau_\alpha) &= \mathcal{G}(\theta_\alpha,\tau_\alpha), \end{aligned} \tag{4.2}$$

where hyperparameters θ_{β_k} and τ_{β_k} represent the shape and the inverse scale parameter of the Gamma distribution representing β_k. Similarly, θ_{α_k} and τ_{α_k} are the shape and the inverse scale parameter of the distribution representing α_k.

Using the above probabilistic model we seek to learn the parameters of the consensus $\mathbf{u}$ and expert-specific models W from data. Similarly to Raykar et al. [Raykar et al., 2010] we optimize the parameters of the model by maximizing the posterior probability $p(\mathbf{u},W,\alpha,\beta|X,\mathbf{y},\xi)$, where ξ is the collection of hyperparameters $\eta,\theta_{\beta_k},\tau_{\beta_k},\theta_{\alpha_k},\tau_{\alpha_k}$.

The posterior probability can be rewritten as follows:

$$\begin{aligned} &p(\mathbf{u},W,\alpha,\beta|X,\mathbf{y},\xi) \propto \\ &\quad p(\mathbf{u}|\mathbf{0}_d,\eta)\left(\prod_{k=1}^{m} p(\beta_k|\theta_\beta,\tau_\beta)p(\alpha_k|\theta_\alpha,\tau_\alpha)p(\mathbf{w}_k|\mathbf{u},\beta_k)\prod_{i=1}^{n_k} p(y_i^k|\mathbf{x}_i^k,\alpha_k,\mathbf{w}_k)\right) \end{aligned} \tag{4.3}$$

where $X=[\mathbf{x}_1^1;\ldots;\mathbf{x}_{n_1}^1;\ldots;\mathbf{x}_1^m;\ldots;\mathbf{x}_{n_m}^m]$ is the matrix of examples labeled by all the experts, and $\mathbf{y}=[y_1^1;\ldots;y_{n_1}^1;\ldots;y_m^1;\ldots;y_{n_m}^m]$ are their corresponding labels. Similarly, X^k and $\mathbf{y}^k$ are the examples and their labels from expert k.

Direct optimization (maximization) of the above function is difficult due to the complexities caused by the multiplication of many terms. A common optimization trick to simplify the objective function is to replace the original complex objective function with the logarithm of that function. This conversion reduces the multiplication to summation [Bishop, 2006]. Logarithm function is a monotonic function and leads to the same optimization solution as

the original problem. Negative logarithm is usually used to cancel many negative signs produced by the logarithm of exponential distributions. This changes the maximization to minimization. We follow the same practice and take the negative logarithm of the above expression to obtain the following problem (see Appendix A for the details of the derivation):

$$\begin{aligned}\min_{\mathbf{u},\mathbf{w},\boldsymbol{\alpha},\boldsymbol{\beta}} \frac{\eta}{2}\|\mathbf{u}\|^2 \quad &+ \quad \frac{1}{2}\sum_{k=1}^{m}\alpha_k \sum_{i=1}^{n_k}\|y_i^k - \mathbf{w}_k^\top \mathbf{x}_i^k\|^2 \\ &+ \quad \frac{1}{2}\sum_{k=1}^{m}\beta_k\|\mathbf{w}_k - \mathbf{u}\|^2 \\ &+ \quad \sum_{k=1}^{m}\big(-\ln(\beta_k) - n_k\ln(\alpha_k)\big) \\ &+ \quad \sum_{k=1}^{m}\big(-(\theta_{\beta_k}-1)\ln(\beta_k) + \tau_{\beta_k}\beta_k\big) \\ &+ \quad \sum_{k=1}^{m}\big(-(\theta_{\alpha_k}-1)\ln(\alpha_k) + \tau_{\alpha_k}\alpha_k\big)\end{aligned} \tag{4.4}$$

Although we can solve the objective function in Equation 4.4 directly, we replace the squared error function in Equation 4.4 with the hinge loss[1] for two reasons: (1) the hinge loss function is a tighter surrogate for the zero-one (error) loss used for classification than the squared error loss [Scholkopf and Smola, 2001], (2) the hinge loss function leads to the sparse kernel solution [Bishop, 2006]. A sparse solution means that the decision boundary depends on a smaller number of training examples. Sparse solutions are more desirable specially when the models are extended to the non-linear case where the similarity of the unseen examples needs to be evaluated with respect to the training examples on which the decision boundary is dependent.

By replacing the squared errors with the hinge loss we obtain the following objective

[1]Hinge loss is a loss function originally designed for training large margin classifiers such as support vector machines. The minimization of this loss leads to a classification decision boundary that has the maximum distance to the nearest training example. Such a decision boundary has interesting properties, including good generalization ability. [Scholkopf and Smola, 2001, Vapnik, 1995]

function:

$$\begin{aligned}\min_{\mathbf{u},\mathbf{w},\boldsymbol{\alpha},\boldsymbol{\beta}} \frac{\eta}{2}\|\mathbf{u}\|^2 &+ \frac{1}{2}\sum_{k=1}^{m}\alpha_k \sum_{i=1}^{n_k}\max\left(0,1-y_i^k\mathbf{w}_k^\top\mathbf{x}_i^k)\right) \\ &+ \frac{1}{2}\sum_{k=1}^{m}\beta_k\|\mathbf{w}_k-\mathbf{u}\|^2 \\ &+ \sum_{k=1}^{m}\left(-\ln(\beta_k)-n_k\ln(\alpha_k)\right) \\ &+ \sum_{k=1}^{m}\left(-(\theta_{\beta_k}-1)\ln(\beta_k)+\tau_{\beta_k}\beta_k\right) \\ &+ \sum_{k=1}^{m}\left(-(\theta_{\alpha_k}-1)\ln(\alpha_k)+\tau_{\alpha_k}\alpha_k\right)\end{aligned} \tag{4.5}$$

We minimize the above objective function with respect to the consensus model $\mathbf{u}$, the expert specific model $\mathbf{w}_k$, and expert specific reliability parameters α_k and β_k.

4.3.2 Optimization

We need to optimize the objective function in Equation 4.5 with regard to parameters of the consensus model $\mathbf{u}$, the expert-specific models $\mathbf{w}_k$, and expert-specific parameters α_k and β_k. Similar to the SVM, the hinge loss term: $\max\left(0,1-y_i^k\mathbf{w}_k^\top\mathbf{x}_i^k\right)$ in Equation 4.5 can be replaced by a constrained optimization problem with a new parameter ϵ_i^k. Briefly, from the optimization theory, the following two equations are equivalent [Boyd and Vandenberghe, 2004]:

$$\min_{\mathbf{w}_k}\max\left(0,1-y_i^k\mathbf{w}_k^\top\mathbf{x}_i^k\right) \tag{4.6}$$

and

$$\begin{aligned}&\min_{\epsilon_i^k,\mathbf{w}_k}\epsilon_i^k \\ &s.t. \quad y_i^k\mathbf{w}_k^\top\mathbf{x}_i^k > 1-\epsilon_i^k\end{aligned}$$

Now replacing the hinge loss terms in Equation 4.5, we obtain the equivalent optimization problem:

$$
\begin{aligned}
\min_{\mathbf{u},\mathbf{w},\boldsymbol{\epsilon},\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & \frac{\eta}{2}\|\mathbf{u}\|^2 + \frac{1}{2}\sum_{k=1}^{m}\alpha_k\sum_{i=1}^{n_k}\epsilon_i^k \\
& + \frac{1}{2}\sum_{k=1}^{m}\beta_k\|\mathbf{w}_k-\mathbf{u}\|^2 \\
& + \sum_{k=1}^{m}\left(-\ln(\beta_k)-n_k\ln(\alpha_k)\right) \\
& + \sum_{k=1}^{m}\left(-(\theta_{\beta_k}-1)\ln(\beta_k)+\tau_{\beta_k}\beta_k\right) \\
& + \sum_{k=1}^{m}\left(-(\theta_{\alpha_k}-1)\ln(\alpha_k)+\tau_{\alpha_k}\alpha_k\right) \\
\text{s.t.} \quad & y_i^k\mathbf{w}_k^\top\mathbf{x}_i^k \geq 1-\epsilon_i^k,\ k=1\ldots m,\ i=1\ldots n_k \\
& \epsilon_i^k \geq 0, \qquad\qquad k=1\ldots m,\ i=1\ldots n_k
\end{aligned}
\tag{4.7}
$$

where $\boldsymbol{\epsilon}$ denote the new set of ϵ_i^k parameters.

We optimize the above objective function using the alternating optimization approach [Bezdek and Hathaway, 2002]. Alternating optimization splits the objective function into two (or more) easier subproblems, each depends only on a subset of (hidden/learning) variables. After initializing the variables, it iterates over optimizing each set by fixing the other set until there is no change of values of all the variables. For our problem, diving the learning variables into two subsets, $\{\alpha,\beta\}$ and $\{\mathbf{u},\mathbf{w}\}$ makes each subproblem easier, as we describe below. After initializing the first set of variables, i.e. $\alpha_k = 1$ and $\beta_k = 1$, we iterate by performing the following two steps in our alternating optimization approach:

- **Learning $\mathbf{u}$ and $\mathbf{w}_k$:** In order to learn the consensus model $\mathbf{u}$ and expert specific model $\mathbf{w}_k$, we consider the reliability parameters α_k and β_k as constants. This will lead to an SVM form optimization to obtain $\mathbf{u}$ and $\mathbf{w}_k$. Notice that ϵ_i^k is also learned as part of SVM optimization.
- **Learning α_k and β_k:** By fixing $\mathbf{u}$, $\mathbf{w}_k$ for all experts, and $\boldsymbol{\epsilon}$, we can minimize the objective function in Equation 4.7 by computing the derivative with respect to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. This

results in the following closed form solutions for α_k and β_k:

$$\alpha_k = \frac{2(n_k + \theta_{\alpha_k} - 1)}{\sum_{y_i^k=1} \epsilon_i^k + 2\tau_{\alpha_k}} \tag{4.8}$$

$$\beta_k = \frac{2\theta_{\beta_k}}{\|\mathbf{w}_k - \mathbf{u}\|^2 + 2\tau_{\beta_k}}. \tag{4.9}$$

Notice that ϵ_i^k is the amount of violation of label constraint for example $\mathbf{x}_i^k$ (i.e. the i^{th} example labeled by expert k) thus $\sum_{i=1} \epsilon_i^k$ is the summation of all labeling violations for model of expert k. This implies that α_k is inversely proportional to the amount of misclassification of examples by expert k according to its specific model $\mathbf{w}_k$. As a result, α_k represents the consistency of the labels provided by expert k with his/her own model. β_k is inversely related to the difference of the model of expert k (i.e. $\mathbf{w}_k$) with the consensus model $\mathbf{u}$. Thus it is the consistency of the model learned for expert k from the consensus model $\mathbf{u}$.

4.4 EXPERIMENTS

In this section, we conduct two sets of experiments, one with two public UCI medical data sets, and the other with our medical data set. We demonstrate that: (1) our method ME-SVM learns better classification models than baseline methods do; (2) ME-SVM performs well with different numbers and different settings of annotators; (3) ME-SVM learns the consistency parameters α and β that help us to quantify the quality of the annotators.

4.4.1 Methods

We compare the following methods:

- **Majority:** An SVM model is trained using the training set obtained from majority voting.
- **Model-Majority** A separate model is trained using the training examples provided by each reviewer. The label of a test example is computed by majority voting of the output of these reviewer specific models.

- **Dawid:** An SVM model is trained using the labels provided by the classical Dawid & Skene's algorithm [Dawid and Skene, 1979].
- **Raykar:** This is the algorithm developed by Raykar et. al. [Raykar et al., 2010]. It computes two reliability parameters for each reviewer: sensitivity and specificity of each reviewer in order to combine the labels and obtain a consensus model.
- **Welinder:** An SVM model is trained using the consensus labels generated by the Welinder et. al. [Welinder et al., 2010] algorithm (this method does not directly generate a consensus model).
- **ME-SVM:** This is the new method we propose in this work. We set $\eta = 1$ and used a similar gamma prior for both α and β: $\tau_{\alpha_k} = \tau_{\beta_k} = \theta_{\alpha_k} = \theta_{\beta_k} = 1$.

These methods were described in more detail in section 2.2.4. Note that our method and Raykar method utilize information provided by the feature vectors while Dawid and Welinder do not. The difference between our method and Raykar's method is that the Raykar's does not consider the existence of reviewers' specific models and assumes that reviewers have access to the consensus label and all the labeling disagreements are due to the fact that reviewers perturb these consensus labels.

For all the baseline methods, we used the setting recommended by the authors. We also set the trade-off parameter $C = 1$ in SVM (default value in LIBSVM [Chang and Lin, 2011]). Notice that the parameter setting is not straightforward since we do not know the true labels of examples.

We have conducted experiments on our own medical data (HIT condition) and two publicly available medical data sets. We divided the data sets randomly to 2/3 for training and 1/3 for future (test) examples. We repeated all experiments 100 times and reported the average performance and 95% confidence interval. These data sets and the experimental results will be described in the following subsections.

4.4.2 Experiments on public medical data sets: Breast Cancer and Parkinsons

To study the performance of ME-SVM under different settings of reviewers' parameters (similar to the approach taken by [Raykar et al., 2010, Welinder et al., 2010]), we used UCI

data sets "Wisconsin Diagnostic Breast Cancer" (WDBC) and "Parkinsons" and generated reviewers with different values of these parameters. We performed the following steps to construct synthetic reviewers: (1) We learned a binary classifier $\mathbf{u}$ and bias term b using SVM and the true labels provided in these data sets. (2) To model the different level of expertise (consensus-consistency) for reviewers, we generated $\mathbf{z}_k$ for reviewer k based on a Gaussian distribution with mean 0 and different values of variance $[0, 0.1^*, 0.2, 0.3, 0.4, 0.5]$ and obtained the reviewer-specific model $\mathbf{w}_k = \mathbf{u} + \mathbf{z}_k$. (3) After obtaining the labeling based on this specific models, we flipped the labels of a fraction of examples; the fraction's value is based on a Gaussian distribution with different values of mean $[0, 0.1, 0.2, 0.3^*, 0.4, 0.5]$ and variance 0.1. The values pointed by * are the default values in experiments when they are not specified. The default value for the number of reviewers is 3.

The 3rd and 4th columns in Table 5 show the performance of different methods for "Breast Cancer" ("Wdbc") and "Parkinson" data sets and three reviewers. The results show that the proposed method, ME-SVM, significantly outperforms the other baselines. To investigate the effect of different settings of the reviewer-specific parameters, we investigate how different methods work under controlled setting of these parameters. Notice that we fix all parameters to their default value, as described earlier in this section, except the parameter under study.

Table 5: AUC of different methods on medical data sets with 3 reviewers

Method/Data set	HIT	Breast Cancer (Wdbc)	Parkinsons
# Examples/Features	377/50	569/31	195/23
Majority	83 ± 1	91 ± 1	73 ± 2
Model-Majority	80 ± 1	85 ± 3	70 ± 3
Dawid	80 ± 1	91 ± 1	69 ± 3
Raykar	83 ± 1	91 ± 1	73 ± 2
Welinder	83 ± 1	88 ± 2	72 ± 2
ME-SVM	$\mathbf{86 \pm 1}$	$\mathbf{97 \pm 1}$	$\mathbf{81 \pm 2}$

4.4.2.1 Effect of the number of reviewers To study the effect of the number of reviewers, we generated different numbers of reviewers with levels of expertise, noise, and bias set to default values. Figure 30 shows the result of this experiment. Note that our method ME-

SVM significantly outperforms baseline methods, especially when the number of reviewers is small. This is important for medical domain because in practice, it is hard to have a large number (e.g. more than 20) of medical experts working on patient cases, due to availability and high labelling cost.

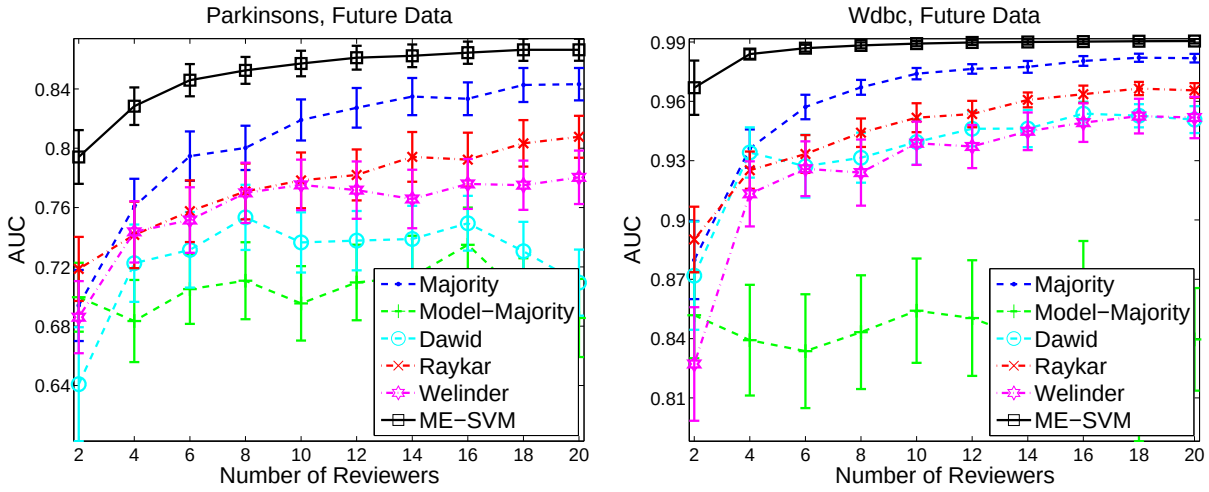

Figure 30: UCI data: effect of number of reviewers

4.4.2.2 Effect of different levels of expertise (model-consistency) To study the effect of different expertise levels, we varied the variance of $\mathbf{z}_k$ (model noise) as specified in Section 4.4.2. We report the results in Figure 31. Notice that the performance of all methods decreases when the model noise increases. However, our method ME-SVM is more robust and outperforms other baselines regardless of the noise level.

4.4.2.3 Effect of different levels of self-consistency To study the effect of different self-consistency levels, we varied the mean value of the fraction of examples with flipped labels (flipping noise) as specified in Section 4.4.2. The results are reported in Figure 32. Notice that when the flipping noise is extreme (close to 0 or 0.5) the performance of all methods are similar. This is because the flipping noise close to 0 indicates that reviewers are very consistent and rarely make careless mistakes, so all methods would perform well, whereas flipping noise close to 0.5 indicates that reviewers make a lot of random mistakes, which makes learning almost impossible for any method. However, when the flipping noise

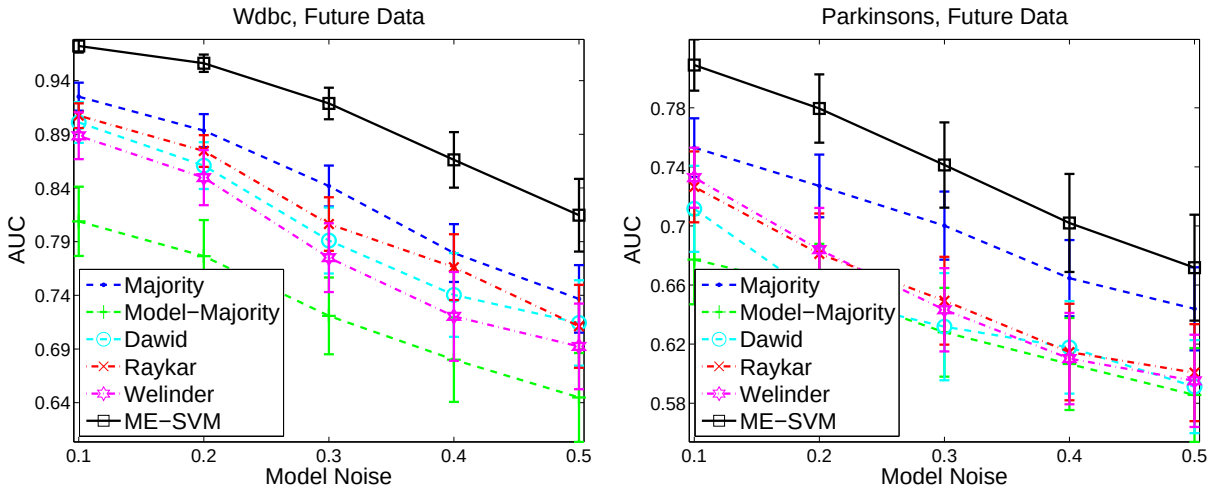

Figure 31: UCI data: effect of model noise

is average (0.2 - 0.3) our method ME-SVM shows superior performance compared to the baselines. This is important because the case of average noise is common in practice, where human annotators do make random mistakes, but not too frequently, due to rush, tiredness, etc.

4.4.3 Experiments on HIT data set

We test the performance of our method on clinical data obtained from EHRs for post surgical cardiac patients (PCP) and the problem of monitoring and detection of the Heparin Induced Thrombocytopenia (HIT) [Warkentin et al., 2000, Warkentin, 2003]. Details about HIT and HIT data were described in section 3.5.1. In this work, we investigate the possibility of building a detector from patient data and human expert assessment of patient cases with respect to HIT and the need to raise the HIT alert. This corresponds to the problem of learning a classification model from data where expert's alert or no-alert assessments define class labels.

4.4.3.1 HIT data assessment We asked three clinical pharmacists to provide us with labels showing if the patient is at the risk of HIT and if they would agree to raise an alert on

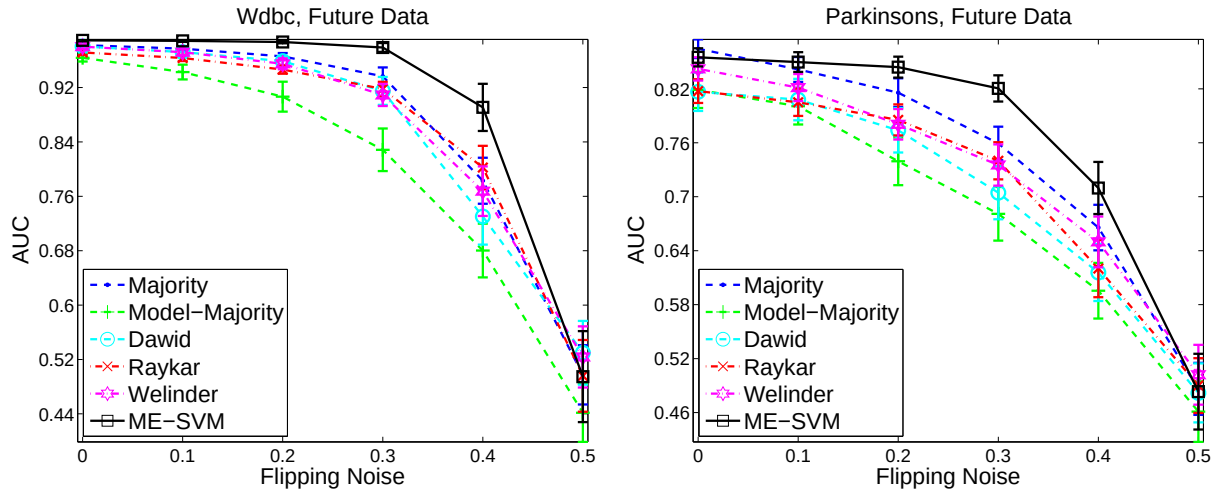

Figure 32: UCI data: effect of flipping noise

HIT if the patient was encountered prospectively. The assessments were conducted using a web-based graphical interface (called PATRIA) we have developed to review EHRs of patients in the PCP database and their instances. All three pharmacists worked independently and labeled 377 patient instances. Note that we collected both binary class and soft-labels as mentioned in Section 3.5.1. However, for a fair comparison with baseline methods, we used only binary class labels to train models.

After the first round of expert labeling (with three experts) we asked a senior expert on HIT condition to label the data, but this time, the expert, in addition to information in the EHR, also had access to the labels of the other three experts. This process led to 88 positive and 289 negative labels. We used the judgment and labels provided by this expert as consensus labels.

We note that alternative ways of defining consensus labels in the study would be possible. For example, one could ask the senior expert to label the cases independent of labels of other reviewers and consider expert's labels as surrogates for the consensus labels. Similarly one can ask all three experts to meet and resolve the cases they disagree on. However, these alternative designs come with the different limitations. First, not seeing the labels of other reviewers the senior expert would make a judgment on the labels on her own and

hence it would be hard to speak about consensus labels. Second, the meeting of the experts and the resolution of the differences on every case in the study in person would be hard to arrange and time consuming to undertake. Hence, we see the option of using senior expert's opinion to break the ties as a reasonable alternative that (1) takes into account labels from all experts, and, (2) resolves them without arranging a special meeting of all experts involved.

In addition, we would like to emphasize that the labels provided by the senior expert were only used to evaluate the quality of the different consensus models. That is, we we did not use the labels provided by that expert when training the different consensus models, and only applied them in the evaluation phase.

4.4.3.2 Experiment: learning consensus model

The cost of labeling examples in medical domain is typically very high, so in practice we may have a very limited number of training data. Therefore, it is important to have a model that can efficiently learn from a small number of training examples. We investigate how different methods perform when the size of training data varies. For this experiment we randomly sample examples from the training set to feed the models and evaluate them on the test set. We tested two different ways of labeling the examples used for learning the model: (1) every example was given to just one expert and every expert labeled the same number of examples, and (2) every example was given to all experts, that is, every example was labeled three times. The results are shown in Figure 33. The x-axis shows the total number of cases labeled by the experts. The left and right plots respectively show the results when labeling options 1 and 2 are used.

First notice that our method that explicitly models experts' differences and their reliability consistently outperforms other consensus methods in both strategies, especially when the number of training examples is small. This is particularly important when labels are not recorded in the EHRs and must be obtained via a separate post-processing step, which can turn out to be rather time-consuming and requires additional expert effort. In contrast to our method the majority voting does not model the reliability of different experts and blindly considers the consensus label as the majority vote of labels provided by different experts. The SVM method is a simple average of reviewer specific models and does not con-

sider the reliability of different experts in the combination. The Raykar method, although modeling the reliability of different experts, assumes that the experts have access to the label generated by the consensus model and report a perturbed version of the consensus label. This is not realistic because it is not clear why the expert perturb the labels if they have access to consensus model. In contrary, our method assumes that different experts aim to use a model similar to consensus model to label the cases however their model differs from the label of the consensus model because of their differences in the domain knowledge, expertise and utility functions. Thus, our method uses a more intuitive way and realistic approach to model the label generating process.

Second, by comparing the two strategies for labeling patient instances we see that option 1, where each reviewer labels different patient instances, is better (in terms of the total labeling effort) than option 2 where all reviewers label the same instances. This shows that the diversity in patient examples seen by the framework helps and our consensus model is improving faster, which is what we intuitively expect.

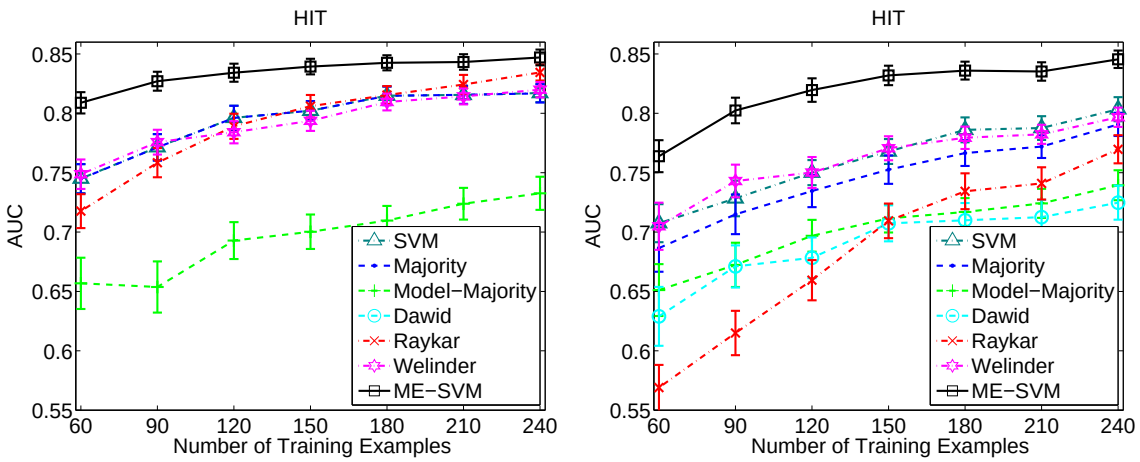

Figure 33: Effect of the number of training examples on the quality of the model when: (Left) every example is labeled by just one expert; (Right) every example is labeled by all three experts

4.4.3.3 Experiment: running time for learning the consensus model In this section we compare the running times required to train consensus models for the different methods when the examples provided by each reviewer fully overlap. Figure 34 shows the

results for ME-SVM and three baseline methods: SVM, Majority and Model-Majority. All of these methods rely on the linear SVM solver implemented in the Liblinear package [Chang et al., 2008] to optimize the parameters of their respective models. This package is optimized for training speed and it scales linearly in the number of training examples. However, we note that it only models and optimizes linear models; there is no option to use a non-linear kernel in this package.

We can see that Majority and Model-Majority approaches scale linearly and are the fastest in terms of their training time. This is because both of these methods only perform a voting step (it takes $O(kN)$ time for N training examples and k annotators) and a training step when the parameters of the linear model are optimized (it takes $O(N)$ time), so the total running time complexity is $O(kN)$. The SVM approach also scales linearly in N and k. However, it tends to be slower than Majority and Model-Majority approaches because the model training dominates the execution time, and the SVM model is trained on Nk examples (k annotators times N examples). In contrast to this, the Majority and Model-Majority approaches always run the SVM learning on N examples only.

ME-SVM uses the alternate optimization process, similar to the Expectation Maximization (EM) procedure, to learn the consensus model. Therefore, the total time complexity of ME-SVM is the number of iterations in the alternate optimization procedure multiplied by time complexity of solving SVM in each iteration ($O(kN)$). We currently do not have any rigorous complexity analysis of the rate of convergence of the iterative ME-SVM algorithm. However, the running time curve for ME-SVM in Figure 34 and its quadratic shape suggests the procedure is quadratic in N.

The running times for Raykar, Dawid and Welinder methods are not shown in Figure 34 because these methods (their implementation) use different optimization solvers that are not optimized for speed like the Liblinear package. Therefore, a side -by-side running-time comparison would not be fair for these methods. Nevertheless, Raykar, Dawid and Welinder methods rely on iterative EM procedures similar to the solution we use for ME-SVM, so assuming that these methods would use a solver comparable to Liblinear package in terms of the speed, their running times would be similar to that of ME-SVM.

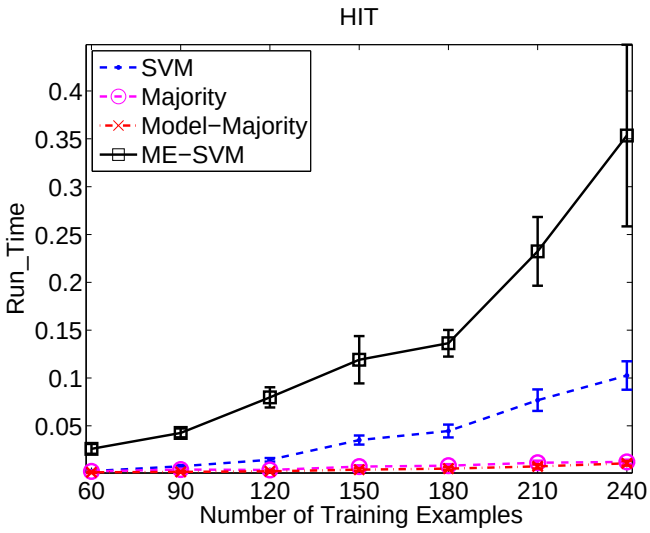

Figure 34: Number of training examples versus running time required for learning the consensus model (in seconds).

4.4.3.4 Experiment: modeling individual experts One important and unique feature of our framework when compared to other multi-expert learning frameworks is that it models explicitly the individual experts' models $\mathbf{w_k}$, not just the consensus model $\mathbf{u}$. In this section, we study the benefit of the framework for learning the expert specific models by analyzing how the model for any of the experts can benefit from labels provided by other experts. In other words we investigate the question: *Can we learn an expert model better by borrowing the knowledge and labels from other experts?* We compared the expert specific models learned by our framework with the following baselines:

- **SVM:** We trained a separate SVM model for each expert using patient instances labeled only by that expert. We use this model as a baseline.
- **Majority*:** This is the Majority model described in the previous section. However, since Majority model does not give expert specific models, we use the consensus model learned by the Majority method in order to predict the labels of each expert.
- **Raykar*:** This is the model developed by Raykar et. al. [Raykar et al., 2010], as described in the previous section. Similarly to Majority, Raykar's model does not learn expert specific models. Hence, we use the consensus model it learns to predict labels of

individual experts.

- **ME-SVM:** This is the new method we propose in this work, that generates expert specific models as part of its framework.

Similarly to Section 4.4.3.2, we assume two different ways of labeling the examples: (1) every example was given to just one expert, and every expert labeled the same number of examples, and (2) every example was given to all experts, that is every example was labeled three times.

We are interested in learning individual prediction models for three different experts. If we have a budget to label some number of patient instances, say, 240, and give 80 instances to each expert, then we have can learn an individual expert model from: (1) all 240 examples by borrowing from the instances labeled by the other experts, or (2) only its own 80 examples. The hypothesis is that learning from data and labels given by all three experts collectively is better than learning each of them individually. The hypothesis is also closely related to the goal of multi-task learning, where the idea is to use knowledge, models or data available for one task to help learning of models for related domains.

The results for this experiment are summarized in Figure 35, where x-axis is the number of training examples fed to the models and y-axis shows how well the models can predict individual experts' labels in terms of the AUC score. The first (upper) line of sub-figures shows results when each expert labels a different set of patient instances, whereas the second (lower) line of sub-figures shows results when instances are always labeled by all three experts. The results show that our ME-SVM method outperforms the SVM trained on experts' own labels only. This confirms that learning from three experts collectively helps to learn expert-specific models better than learning from each expert individually and that our framework enables such learning. In addition, the results of Majority* and Raykar* methods show that using their consensus models to predict expert specific labels is not as effective and that their performance falls bellow our framework that relies on expert specific models.

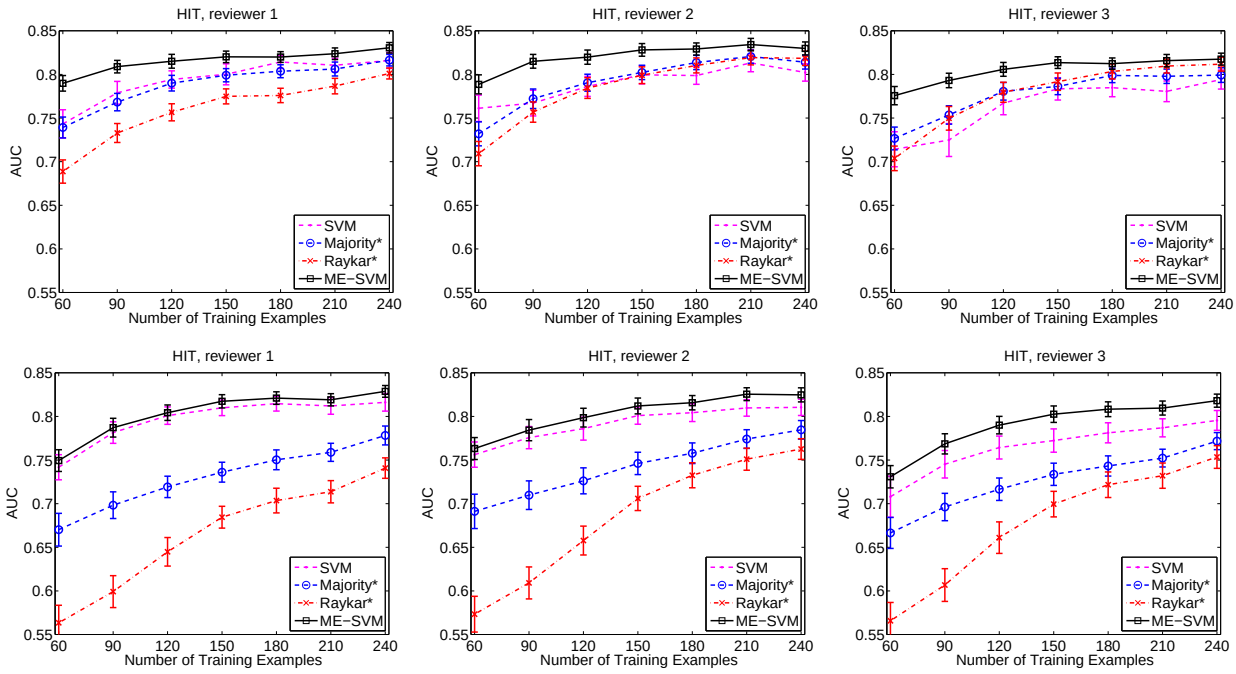

Figure 35: Learning of expert-specific models. The figure shows the results for three expert specific models generated by the ME-SVM and the standard SVM methods, and compares them to models generated by the Majority* and Raykar* methods. First line: different examples are given to different experts; Second line: the same examples are given to all experts.

4.4.3.5 Experiment: self-consistency and consensus-consistency As we described in Section 4.3, we model self-consistency and consensus-consistency with parameters α_k and β_k. α_k measures how consistent the labeling of expert k is with his/her own model and β_k measures how consistent the model of expert k is with respect to the consensus model. The optimization problem we proposed in Equation 4.7 aims to learn not just the parameters $\mathbf{u}$ and $\mathbf{w}_k$ of the consensus and experts' models, but also the parameters α_k and β_k, and this without having access to the labels from the senior expert.

In this section, we attempt to study and interpret the values of the reliability parameters as they are learned by our framework and compare them to empirical agreements in between the senior (defining the consensus) and other experts. Figure 36(a) shows the agreements of labels provided by the three experts with labels given by the senior expert,

which we assumed gives the consensus labels. From this figure we see that Expert 2 agrees with the consensus labels the most, followed by Expert 3 and then Expert 1. The agreement is measured in terms of the absolute agreement, and reflects the proportion of instances for which the two labels agree.

Figures 36(b) and 36(c) show the values of the reliability parameters α and β, respectively. The x-axis in these figures shows how many training patient instances per reviewer are fed to the model. Normalized Self-Consistency in Figure 36(b) is the normalized value of α_k in Equation 4.7. Normalized Consensus-Consistency in Figure 36(c) is the normalized inverse value of Euclidean distance between an expert specific model and consensus model: $1/||\mathbf{w}_k - \mathbf{u}||$, which is proportional to β_k in Equation 4.7. In Figure 36(d) we add the two consistency measures in an attempt to measure the overall consistency in between the senior expert (consensus) and other experts.

As we can see, at the beginning when there is no training data all experts are assumed to be the same (much like the majority voting approach). However, as the learning progresses with more training examples available, the consistency measures are updated and their values define the contribution of each expert to the learning of consensus model: the higher the value the larger the contribution. Figure 36(b) shows that expert 3 is the best in terms of self-consistency given the linear model, followed by expert 2 and then expert 1. This means expert 3 is very consistent with his model, that is, he likely gives the same labels to similar examples. Figure 36(c) shows that expert 2 is the best in terms of consensus-consistency, followed by expert 3 and then expert 1. This means that although expert 2 is not very consistent with respect to his own linear model his model appears to converge closer to the consensus model. In other words, expert 2 is the closest to the expected consensus in terms of the expertise but deviates with some labels from his own linear model than expert 3 does[2]

Figure 36(d) shows the summation of the two consistency measures. By comparing Figure 36(a) and Figure 36(d) we observe that the overall consistency mimics well the agreements in between the expert defining the consensus and other experts, especially when the

[2]We would like to note that the self-consistency and consensus-consistency parameters learned by our framework are learned together and hence it is possible one consistency measure may offset or compensate for the value of the other measure during the optimization. In that case the interpretation of the parameters as presented may not be as straightforward.

number of patient instances labeled and used to train our model increases. This is encouraging, since the parameters defining the consistency measures are learned by our framework only from the labels of the three experts and hence the framework never sees the consensus labels.

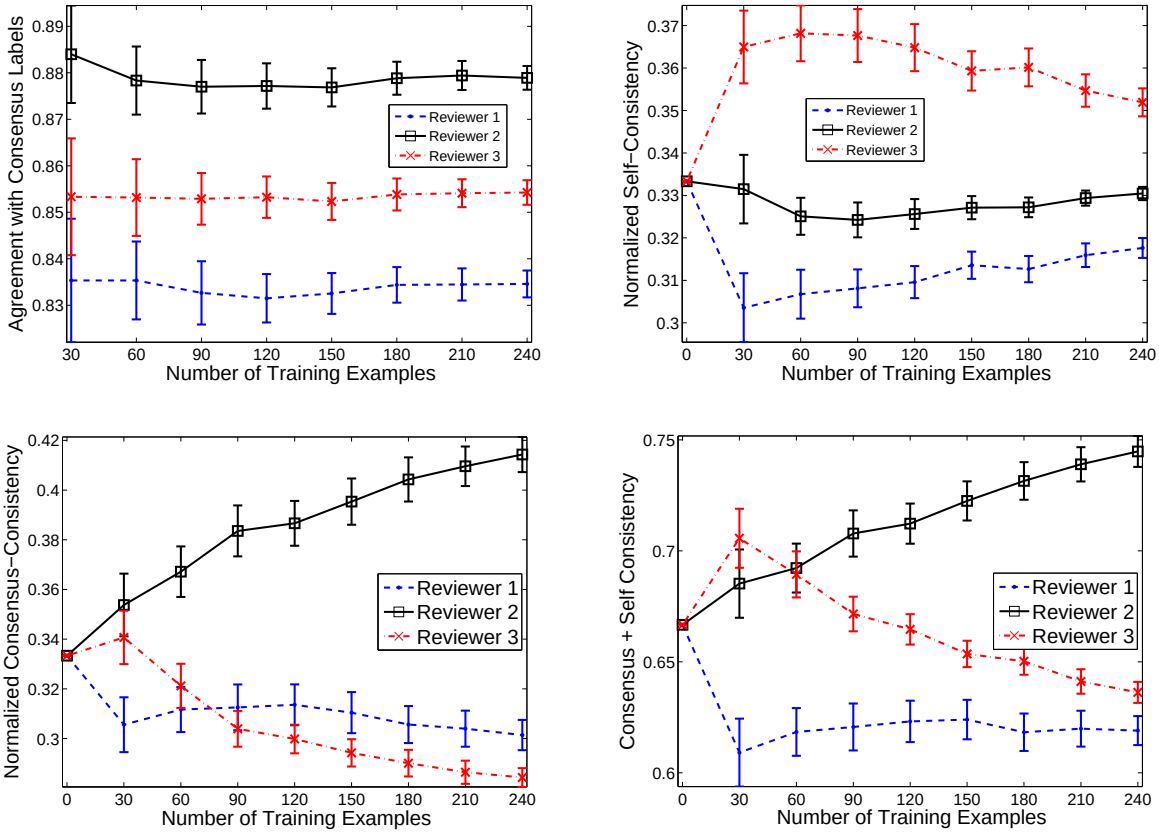

Figure 36: (left top) Agreement of experts with labels given by the senior expert; (right top) Learned self consistency parameters for Experts 1-3; (left bottom) Learned consensus consistency parameters for Experts 1-3; (right bottom) Cumulative self and consensus consistencies for Expert 1-3.

4.5 SUMMARY

In many real world applications, labelling examples may be very tedious and time consuming, so it is not expected that one annotator will label all examples. Instead, we may have

a learning problem where a group of annotators provides assessments on a set of examples. The challenge with this problem is that annotators often disagree on labelling, sometimes even give contradictory assessments on the same examples. This is common in practice because different annotators may have different levels of expertise, bias, or take different mount of time and effort for labelling. In this work, we have investigated a solution for the multiple annotator learning problem, where the goal is to efficiently combine labeled data collected from different annotators to learn better predictive models. We propose a framework that takes into account and learns three characteristics of annotators: (1) their individual labeling model, (2) consistency with their own model, and (3) their bias towards positive/negative class.

We evaluated our method on public UCI medical data sets and our clinical data. The experimental results show the benefits of our method:

- It learns a consensus model that outperforms baseline methods, regardless of the number of reviewers, the levels of annotator expertise, self-consistency or bias. The advantage is especially clear when the number of reviewers is low and the level of noise in labels is average. This scenario is very common in practice, particularly in medical domain, because typically we do not have a large number of annotators (medical experts are scarce resources), and human annotators do make mistakes, but not too frequently.
- It also learns reviewer specific models and parameters very well. This is important because it provides more insights into the labelling process and identifies which annotators are better and make more contributions to the consensus model.
- It shows that learning annotator models collectively (in parallel) is more efficient than learning them individually. This observation shows a connection to real world situations, when we see that people learn/work more efficiently when they collaborate in a group.

5.0 CONCLUSIONS

5.1 CONTRIBUTIONS

In many supervised classification tasks training examples need to be labeled by human annotators before they can be used for learning models. The labeling process may be very costly and tedious, especially in domains where data are complicated and a high level of expertise is required. An example of such domains is medical diagnosis. In this dissertation, I studied and proposed solutions to address the cost-efficient learning problem, where the objective is to learn better classification models while reducing and efficiently distributing the cost of labeling. The main contributions of this work are summarized as follows.

- We presented a learning from soft-label framework, where we ask the human annotator to provide us with, in addition to class label, also soft-label reflecting his/her belief in the class label, and incorporate this information into the learning process. In general, the soft-label can be presented in terms of probability, e.g. 0.85, or qualitative categories, e.g. weak/strong belief. Our framework was motivated by the observation that the cost of labeling is mostly in the example review. Once an example was reviewed and class label was given, the human annotator can give us the auxiliary soft-label at an insignificant cost. We have done a study in medical domain to confirm this conjecture.
- We showed that the soft-label information can help to significantly increase the performance of classification models. However, we pointed out that the soft-label given by human may be noisy and negatively influence the learning process. To address this problem, we proposed a ranking-based method that is very robust to noise and outperforms standard binary classifiers even with a strong level of noise in the soft-label. The idea

of this method is to model the pair-wise ranking relations between training examples, and combine that with the class information, instead of relying on the exact values of soft-labels or class label alone.

- We showed that, while the above ranking method learns high-quality classifiers, it does not scale up well with large training data because of the $O(N^2)$ number of ranking constraints in the optimization. To address this issue, we proposed a ranking method that is able to reduce the number of constraints to $O(N)$ while retaining the high classification performance. The idea is to distribute training examples to a constant number of discrete bins based on the soft-label information, then enforce optimization constraints on examples and bin boundaries. Another advantage of this method, besides the linear number of constraints, is that it can be naturally applied in the case when soft-labels are presented in qualitative categories. In this case the discrete categories/bins are given directly by the human annotators.
- We presented a novel multi-annotator learning framework that takes into account different aspects of the labeling process and efficiently combines information from different annotators to learn better classification models. This framework addresses a problem of the traditional supervised learning framework: it assumes that, either there is a single annotator labeling all examples, or the labeling process is uniform and labels from different annotators are treated similarly. This is often not the case in practice. First, the labeling process may be tedious, so we cannot expect that a single human annotator will be able to work on the whole training data. Second, in some practical applications, there are naturally many annotators, e.g. a group of physicians examining the same patient, or many people annotating the same news article. Lastly, it is important to accept the fact that the labeling process is not uniform - people often have different opinions on the same subject.
- Our multi-annotator learning approach learns better classification models than state-of-the art methods. The current methods mainly come from the learning-from-crowds field, which has been motivated by the growing number of crowd-sourcing services. These methods typically learn a voting model that weights the labels collected from the crowds of online workers. The methods often rely on a large number of annotators and labels

which can be had at low costs. However, in many practical tasks, e.g. disease diagnosis, it is not feasible to acquire a large number of annotators because the labeling process requires people with high levels of expertise, which is an expensive and scarce resource. Our approach does not rely on a large number of labels, instead, it takes into account and learns different characteristics of the (expert) annotators. Briefly, it models the domain knowledge, the bias and the consistency of different annotators. This approach results in successful learning of a consensus model that represents the collective wisdom of the experts.

- Another advantage of our multi-annotator learning approach is that it also learns the model and other characteristics of each annotator, such as consistency, reliability and bias. This is useful in practice because we can use the individual models to predict labeling patterns of the experts. Moreover, this gives us more insights into the labeling process, which helps us to tune the models and the labeling process itself for better predictive performance and higher quality of data in the future (e.g., we may prefer to collaborate with more reliable annotators).

5.2 OPEN QUESTIONS

We have proposed approaches to learn better classification models while reducing and efficiently distributing the cost of labeling. Our approaches show superior performance compared to existing approaches. However, there are still many challenges and open questions that are subject to further investigation.

- In learning with soft-label information framework, we proposed a ranking method that has $O(N)$ number of constraints in the optimization, therefore it scales up well with large training data. However, in the case when soft-labels are presented in terms of probabilistic labels, this methods requires a discretization and distribution of examples into different bins based on the soft-label information. There are two immediate questions: (1) how many bins should we have, and (2) how to distribute examples into these bins ? In this work we set the number of bins to five and equally distribute examples

into five bins. In practice, the performance of classification models may vary and depend on these settings.

- The baseline methods are mostly designed for crowd-sourcing applications, so they perform well with larger numbers of annotators. Currently we have only three annotators in our real medical data set. Therefore, it is worthwhile to investigate how our multi-annotator learning framework works with larger numbers of annotators compared to the other methods on our data.
- Active selection of annotators. As mentioned before, our method can learn individual annotator models, as well as consistency parameters of annotators. Therefore, we can learn which annotator is better for the labeling task. This opens a new research opportunity: apply this method to actively select annotators who can make more contributions to our task.
- An open research direction is to extend our multiple-annotator learning framework to incorporate auxiliary information. Notice that our current multiple-annotator learning framework is designed for the classification task with the availability of class labels, but does not take into account auxiliary soft labels. As discussed in Chapter, we modeled the expertise (model-consistency), bias and self-consistency of annotators by a set of annotator-specific parameters. In the new setting, the question is how to model annotators with a new set of soft labels, in addition to the set of binary class labels? Can we assume that if an annotator is consistent with class labels then he will also be consistent with auxiliary soft-labels? Adding more parameters to the current framework would make it more difficult to optimize and possibly increase over-fitting.

APPENDIX

STATISTICS AND MATHEMATICAL DERIVATIONS

A.1 FEATURES USED FOR CONSTRUCTING THE PREDICTIVE MODELS

Table 6: Features used for constructing the predictive models. The features were extracted from time series data in electronic health records using methods from [Hauskrecht et al., 2010, Valko and Hauskrecht, 2010, Hauskrecht et al., 2013]

Clinical variables	Features	
Platelet count (PLT)	1	last PLT value measurement
	2	time elapsed since last PLT measurement
	3	pending PLT result
	4	known PLT value result indicator
	5	known trend PLT results
	6	PLT difference for last two measurements
	7	PLT slope for last two measurements
	8	PLT % drop for last two measurements
	9	nadir HGB value
	10	PLT difference for last and nadir values
	11	apex PLT value
	12	PLT difference for last and apex values
	13	PLT difference for last and baseline values
	14	overall PLT slope
Hemoglobin (HGB)	15	last HGB value measurement
	16	time elapsed since last HGB measurement
	17	pending HGB result
	18	known HGB value result indicator
	19	known trend HGB results
	20	HGB difference for last two measurements
	21	HGB slope for last two measurements
	22	HGB % drop for last two measurements
	23	nadir HGB value
	24	HGB difference for last and nadir values
	25	apex HGB value
	26	HGB difference for last and apex values
	27	HGB difference for last and baseline values
	28	overall HGB slope
White Blood Cell count (WBC)	29	last WBC value measurement
	30	time elapsed since last WBC measurement
	31	pending WBC result
	32	known WBC value result indicator
	33	known trend WBC results
	34	WBC difference for last two measurements
	35	WBC slope for last two measurements
	36	WBC % drop for last two measurements
	37	nadir WBC value
	38	WBC difference for last and nadir values
	39	apex WBC value
	40	WBC difference for last and apex values
	41	WBC difference for last and baseline values
	42	overall WBC slope
Heparin	43	Patient on Heparin
	44	Time elapsed since last administration of Heparin
	45	Time elapsed since first administration of Heparin
	46	Time elapsed since last change in Heparin administration
Major heart procedure	47	Patient had a major heart procedure in past 24 hours
	48	Patient had a major heart procedure during the stay
	49	Time elapsed since last major heart procedure
	50	Time elapsed since first major heart procedure

A.2 STATISTICS OF AGREEMENT BETWEEN EXPERTS

Tables 7 - 12 in this appendix summarize pairwise agreement matrices for experts who reviewed and evaluated the patient cases. Tables 13 and 14 summarize agreement statistics among all experts. There are four possible values for categorical labels they could choose from: "Strongly-disagree", "Weakly-disagree", "Weakly-agree", "Strongly-agree" with alert. When interpreted in terms of agreements we get binary labels: "Agree" and "Disagree" with alert. The results show the agreement matrices for both binary and ordinal categorical labels.

For each pairwise agreement table (7 - 12) we calculate both simple agreement and kappa [Cohen, 1960] statistics. Since the basic kappa statistic ignores the ordering of categories, for agreement tables with ordinal categories "Strongly-disagree", "Weakly-disagree", "Weakly-agree", "Strongly-agree" we also calculate weighted kappa [Cohen, 1968] that reflects how far the assessments are in terms of their ordering.

For agreement among all experts (tables 13 and 14) we calculate Fleiss' kappa [Fleiss et al., 1971], which assesses the reliability of agreement between all experts.

Table 7: Expert 1 versus Expert 2, binary labels. Absolute agreement = 0.85, Kappa = 0.53

Expert 1	Expert 2	
	Disagree with alert	Agree with alert
Disagree with alert	279	10
Agree with alert	45	43

Table 8: Expert 1 versus Expert 2, ordinal categorical labels. Absolute agreement = 0.61, Kappa = 0.39, Weighted Kappa = 0.47

Expert 1	Expert 2			
	Strongly-disagree	Weakly-disagree	Weakly-agree	Strongly-agree
Strongly-disagree	71	77	0	0
Weakly-disagree	13	118	10	0
Weakly-agree	5	39	41	0
Strongly-agree	0	1	2	0

Table 9: Expert 1 versus Expert 3, binary labels. Absolute agreement = 0.84, Kappa = 0.57

Expert 1	Expert 3	
	Disagree with alert	Agree with alert
Disagree with alert	249	40
Agree with alert	22	66

Table 10: Expert 1 versus Expert 3, ordinal categorical labels. Absolute agreement = 0.60, Kappa = 0.40, Weighted Kappa = 0.51

Expert 1	Expert 3			
	Strongly-disagree	Weakly-disagree	Weakly-agree	Strongly-agree
Strongly-disagree	75	70	3	0
Weakly-disagree	15	89	37	0
Weakly-agree	0	22	62	1
Strongly-agree	0	0	3	0

Table 11: Expert 2 versus Expert 3, binary labels. Absolute agreement = 0.77, Kappa = 0.33

Expert 2	Expert 3	
	Disagree with alert	Agree with alert
Disagree with alert	254	70
Agree with alert	17	36

Table 12: Expert 2 versus Expert 3, ordinal categorical labels. Absolute agreement = 0.55, Kappa = 0.26, Weighted Kappa = 0.34

Expert 2	Expert 3			
	Strongly-disagree	Weakly-disagree	Weakly-agree	Strongly-agree
Strongly-disagree	45	36	8	0
Weakly-disagree	44	129	62	0
Weakly-agree	1	16	35	1
Strongly-agree	0	0	0	0

Table 13: All three experts, binary labels. Fleiss' Kappa = 0.47.

	Disagree with alert	Agree with alert
All three experts	884	247

Table 14: All three experts, ordinal categorical labels. Fleiss' Kappa = 0.34.

	Strongly-disagree	Weakly-disagree	Weakly-agree	Strongly-agree
All three experts	327	557	243	4

A.3 AUC AND SIGNIFICANCE TEST RESULTS FOR LEARNING WITH AUXILIARY INFORMATION EXPERIMENTS

The tables in this appendix list exact AUC values for the different learning methods and the different number of training examples shown in Figures 20(a), 20(b), 20(c) and 22(a), 22(b), 22(c). In addition to exact AUC values, the tables summarize the results of the pairwise statistical test assessing the significant differences between the best performing method and the rest of the methods.

Experiment 1 (Figures 20(a), 20(b), 20(c))

The following tables show the AUC for the different methods and different number of training examples, for expert 1, 2 and 3, respectively. The methods use probabilistic labels. The best method is shown in bold. A star next to a number indicates the method is statistically significantly different (at p=0.05) than the best method.

Table 15: Expert 1 (figure 20(a))

Method	**Number of training examples**							
	20	40	60	90	120	160	200	250
LogR	57.5*	68.0*	71.4*	74.4*	76.6	77.6	79.2	79.3
SVM	58.6*	68.9*	72.9*	74.3*	76.0	77.2	77.3	78.3
LinRaux	59.2*	60.0*	63.5*	66.7*	68.7*	68.9*	70.4*	70.1*
SvmAuxPair	70.8	75.9	75.8	76.7	77.8	78.8	79.7	80.8
SvmAuxBinPair	67.7*	70.5*	73.8*	73.0*	75.3*	75.3*	75.4*	76.7*
SvmAuxOrd	66.9*	72.3*	74.5	73.5	77.8	78.4	78.3	79.1

Table 16: Expert 2 (figure 20(b))

Method	**Number of training examples**							
	20	40	60	90	120	160	200	250
LogR	51.3*	55.5*	61.5*	66.1*	70.0*	74.4*	80.3	80.2*
SVM	50.9*	56.6*	62.9*	67.1*	70.5*	76.4*	81.9	81.1*
LinRaux	55.6*	58.9*	62.6*	69.9*	71.1*	71.8*	72.2*	72.9*
SvmAuxPair	62.2	67.7	72.2	75.8	76.8	79.0	81.6	81.6
SvmAuxBinPair	54.3*	66.5	69.9	73.0	73.8	76.6	79.5	81.9
SvmAuxOrd	58.0	65.3	70.3	74.8	77.5	79.6	82.8	84.4

Table 17: Expert 3 (figure 20(c))

Method	**Number of training examples**							
	20	40	60	90	120	160	200	250
LogR	60.6	68.4*	71.6	75.6	77.7	79.2	79.3	78.7
SVM	61.2	70.3	74.2	76.3	77.0	78.1	78.8	79.0
LinRaux	50.7*	53.7*	55.5*	56.7*	58.2*	58.9*	58.0*	58.6*
SvmAuxPair	63.2	71.8	74.4	77.9	78.9	79.6	81.0	80.8
SvmAuxBinPair	60.9	67.9*	70.4*	73.2*	73.2*	73.9*	76.0*	76.7*
SvmAuxOrd	60.2	68.0*	71.8	77.0	77.8	77.5*	78.3*	78.5*

Experiment 2 (Figures 22(a), 22(b), 22(c))

The following tables show the AUC for the different methods and the different number of training examples. The methods use ordinal category labels. The best method is shown in bold. A star next to a number indicates the method is statistically significantly different (at p=0.05) than the best method.

Table 18: Expert 1 (figure 22(a))

Method	**Number of training examples**							
	20	40	60	90	120	160	200	250
LogR	57.5*	68.0*	71.4*	74.4*	76.6	77.6	79.2	79.3
SVM	58.6*	68.9*	72.9*	74.3*	76.0	77.2	77.3*	78.3*
Multiclass	59.3*	64.0*	68.8*	72.5*	73.9*	76.8*	78.2	77.9
SvmAuxPair	73.5	78.7	78.8	80.1	78.6	79.6	79.7	80.7
SvmAuxOrd	72.5	74.9*	76.8	76.7*	78.3	79.1	79.0	78.9

Table 19: Expert 2 (figure 22(b))

Method	**Number of training examples**							
	20	40	60	90	120	160	200	250
LogR	51.3*	55.5*	61.5*	66.1*	70.0*	74.4*	80.3*	80.2*
SVM	50.9*	56.6*	62.9*	67.1*	70.5*	76.4*	81.9	81.1
Multiclass	55.8*	61.5*	66.0*	68.1*	68.8*	71.7*	72.7*	74.2*
SvmAuxPair	71.1	73.8	76.1	80.0	80.1	82.4	84.4	83.9
SvmAuxOrd	68.2	72.2	72.1	75.7*	77.9	81.7	84.1	83.4

Table 20: Expert 3 (figure 22(c))

Method	**Number of training examples**							
	20	40	60	90	120	160	200	250
LogR	60.6*	68.4*	71.6*	75.6*	77.7*	79.2*	79.3*	78.7*
SVM	61.2*	70.3*	74.2*	76.3*	77.0*	78.1*	78.8*	79.0*
Multiclass	63.5*	69.8*	73.2*	75.1*	76.7*	77.0*	77.5*	77.3*
SvmAuxPair	71.0	76.5	79.4	80.6	80.6	81.1	82.0	81.9
SvmAuxOrd	67.3*	71.8*	74.9*	78.8	78.7	78.6*	79.3*	79.6*

A.4 DERIVATION OF EQUATION 4.4

In this appendix, we give a more detailed derivation of Equation 4.4 from 4.3:

$$
\begin{aligned}
& p(\mathbf{u}, W, \boldsymbol{\alpha}, \boldsymbol{\beta} | X, \mathbf{y}, \xi) \\
\propto \quad & p(\mathbf{u}|\mathbf{0}_d, \eta) \\
& \times \prod_{k=1}^{m} p(\beta_k|\theta_\beta, \tau_\beta) \\
& \quad \times p(\alpha_k|\theta_\alpha, \tau_\alpha) p(\mathbf{w}_k|\mathbf{u}, \beta_k) \\
& \quad \times \prod_{i=1}^{n_k} p(y_i^k|\mathbf{x}_i^k, \alpha_k, \mathbf{w}_k) \\
= \quad & \mathcal{N}(\mathbf{u}|\mathbf{0}_d, \eta^{-1}\mathbf{I}_d) \\
& \times \prod_{k=1}^{m} \mathcal{G}(\beta_k|\theta_\beta, \tau_\beta) \\
& \quad \times \mathcal{G}(\alpha_k|\theta_\alpha, \tau_\alpha) \\
& \quad \times \mathcal{N}(\mathbf{w}_k|\mathbf{u}, \beta_k^{-1}\mathbf{I}_d) \\
& \quad \times \prod_{i=1}^{n_k} \mathcal{N}(y_i^k|\mathbf{w}_k^\top \mathbf{x}_i^k, \alpha_k^{-1}) \\
= \quad & \frac{\eta}{\sqrt{2\pi}} e^{-\frac{\eta\|\mathbf{u}\|^2}{2}} \\
& \times \prod_{k=1}^{m} \frac{1}{\Gamma(\theta_\beta)} \tau_\beta^{\theta_\beta} \beta_k^{\theta_\beta - 1} e^{-\tau_\beta \beta_k} \\
& \quad \times \frac{1}{\Gamma(\theta_\alpha)} \tau_\alpha^{\theta_\alpha} \alpha_k^{\theta_\alpha - 1} e^{-\tau_\alpha \alpha_k} \\
& \quad \times \frac{\beta_k}{\sqrt{2\pi}} e^{-\frac{\beta_k\|\mathbf{w}_k - \mathbf{u}\|^2}{2}} \\
& \quad \times \prod_{i=1}^{n_k} \frac{\alpha_k}{\sqrt{2\pi}} e^{-\frac{\alpha_k\|y_i^k - \mathbf{w}_k^\top \mathbf{x}_i^k\|^2}{2}}
\end{aligned}
$$

Taking the negative logarithm of $p(\mathbf{u}, W, \boldsymbol{\alpha}, \boldsymbol{\beta} | X, \mathbf{y}, \xi)$ that lets us to convert the maxi-

mization problem to minimization, we get:

$$
\begin{aligned}
&-\ln p(\mathbf{u}, W, \boldsymbol{\alpha}, \boldsymbol{\beta} | X, \mathbf{y}, \xi) = \\
&- \quad \ln(\eta) - log(\sqrt{2\pi}) + \frac{1}{2}\eta\|\mathbf{u}\|^2 \\
&+ \quad \sum_{k=1}^{m} \left(\ln(\Gamma(\theta_\beta)) - \theta_\beta \ln(\tau_\beta) - (\theta_\beta - 1)\ln(\beta_k) + \tau_\beta \beta_k\right) \\
&+ \quad \sum_{k=1}^{m} (\ln(\Gamma(\theta_\alpha)) - \theta_\alpha \ln(\tau_\alpha) - (\theta_\alpha - 1)\ln(\alpha_k) + \tau_\alpha \alpha_k) \\
&+ \quad \sum_{k=1}^{m} \left(-\ln(\beta_k) + \ln(\sqrt{2\pi}) + \frac{1}{2}\beta_k \|\mathbf{w}_k - \mathbf{u}\|^2\right) \\
&+ \quad \sum_{k=1}^{m} \sum_{i=1}^{n_k} \left(-\ln(\alpha_k) + \ln(\sqrt{2\pi}) + \frac{1}{2}\alpha_k \|y_i^k - \mathbf{w}_k^\top \mathbf{x}_i^k\|^2\right)
\end{aligned}
$$

Rewriting the above equation we get:

$$
\begin{aligned}
&-\ln p(\mathbf{u}, W, \boldsymbol{\alpha}, \boldsymbol{\beta} | X, \mathbf{y}, \xi) = \\
&\quad \frac{1}{2}\eta\|\mathbf{u}\|^2 \\
&+ \quad \sum_{k=1}^{m} \left(-(\theta_\beta - 1)\ln(\beta_k) + \tau_\beta \beta_k\right) \\
&+ \quad \sum_{k=1}^{m} (-(\theta_\alpha - 1)\ln(\alpha_k) + \tau_\alpha \alpha_k) \\
&+ \quad \sum_{k=1}^{m} \left(-\ln(\beta_k) + \frac{1}{2}\beta_k \|\mathbf{w}_k - \mathbf{u}\|^2\right) \\
&+ \quad \sum_{k=1}^{m} \sum_{i=1}^{n_k} \left(-\ln(\alpha_k) + \frac{1}{2}\alpha_k \|y_i^k - \mathbf{w}_k^\top \mathbf{x}_i^k\|^2\right) \\
&+ \quad A
\end{aligned}
$$

where A sums all constant terms that can be ignored during the optimization step, and that include terms involving hyper-parameters η, θ_α, τ_α, θ_β that are constants. By ignoring A and rearranging the remaining terms we get Equation 4.4.

BIBLIOGRAPHY

[Abe and Mamitsuka, 1998] Abe, N. and Mamitsuka, H. (1998). Query learning strategies using boosting and bagging. In *Proceedings of the 15th International Conference on Machine Learning*, ICML '98, pages 1–9, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Argyriou et al., 2007a] Argyriou, A., Evgeniou, T., and Pontil, M. (2007a). Multi-task feature learning. In *Advances in Neural Information Processing Systems*. MIT Press.

[Argyriou et al., 2007b] Argyriou, A., Micchelli, C. A., Pontil, M., and Ying, Y. (2007b). A spectral regularization framework for multi-task structure learning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc.

[Batal et al., 2012] Batal, I., Fradkin, D., Harrison, J., Moerchen, F., and Hauskrecht, M. (2012). Mining recent temporal patterns for event detection in multivariate time series data. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 280–288. ACM.

[Batal et al., 2009] Batal, I., Sacchi, L., Bellazzi, R., and Hauskrecht, M. (2009). Multivariate time series classification with temporal abstractions. *Florida Artificial Intelligence Research Society Conference*.

[Batal et al., 2011] Batal, I., Valizadegan, H., Cooper, G. F., and Hauskrecht, M. (2011). A pattern mining approach for classifying multivariate temporal data. In *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 358–365. IEEE.

[Batal et al., 2013] Batal, I., Valizadegan, H., Cooper, G. F., and Hauskrecht, M. (2013). A temporal pattern mining approach for classifying electronic health record data. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(4):63.

[Bezdek and Hathaway, 2002] Bezdek, J. C. and Hathaway, R. J. (2002). Some notes on alternating optimization. In *Proceedings of the 2002 AFSS International Conference on Fuzzy Systems. Calcutta: Advances in Soft Computing*, AFSS '02, pages 288–300, London, UK, UK. Springer-Verlag.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

[Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, New York, NY, USA.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

[Breiman et al., 1984] Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA.

[Caruana, 1997] Caruana, R. (1997). Multitask learning. *Machine Learning*, 28(1):41–75.

[Chang and Lin, 2011] Chang, C.-C. and Lin, C.-J. (2011). *LIBSVM: A library for support vector machines*. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[Chang et al., 2008] Chang, K.-W., Hsieh, C.-J., and Lin, C.-J. (2008). Coordinate descent method for large-scale l2-loss linear support vector machines. *J. Mach. Learn. Res.*, 9:1369–1398.

[Chu and Keerthi, 2005] Chu, W. and Keerthi, S. S. (2005). New approaches to support vector ordinal regression. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 145–152, New York, NY, USA. ACM.

[Cohen, 1960] Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37.

[Cohen, 1968] Cohen, J. (1968). Weighted kappa: nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological bulletin*.

[Cohn et al., 1994] Cohn, Atlas, and Ladner (1994). Improving generalization with active learning. *Machine Learning*.

[Cohn et al., 1996] Cohn, D. A., Ghahramani, Z., and Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145.

[Combi et al., 2010] Combi, C., Keravnou-Papailiou, E., and Shahar, Y. (2010). *Temporal information systems in medicine*. Springer Publishing Company, Incorporated.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20:273–297.

[Cressie, 1993] Cressie, N. A. C. (1993). *Statistics for Spatial Data*. Wiley-Interscience, revised edition.

[Dai et al., 2007] Dai, W., Yang, Q., Xue, G.-R., and Yu, Y. (2007). Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 193–200, New York, NY, USA.

[Davis and Domingos, 2009] Davis, J. and Domingos, P. (2009). Deep transfer via second-order markov logic. In *Proceedings of the 26th International Conference on Machine Learning (ICML-09)*, Montreal, Canada.

[Dawid and Skene, 1979] Dawid, A. P. and Skene, A. M. (1979). Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied Statistics*, 28(1):20–28.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of The Royal Statistical Society, Series B*, 39(1):1–38.

[Domingos and Pazzani, 1997] Domingos, P. and Pazzani, M. (1997). On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130.

[Donmez et al., 2009] Donmez, P., Carbonell, J. G., and Schneider, J. (2009). Efficiently learning the accuracy of labeling sources for selective sampling. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[Evgeniou and Pontil, 2004] Evgeniou, T. and Pontil, M. (2004). Regularized multi–task learning. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117, New York, NY, USA. ACM.

[Fisher, 1936] Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(7):179–188.

[Fleiss et al., 1971] Fleiss, J. et al. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

[Friedman, 2010] Friedman, J. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of The Royal Statistical Society, Series B*, 33(1).

[Fürnkranz and Hüllermeier, 2010] Fürnkranz, J. and Hüllermeier, E. (2010). Preference learning. In Sammut, C. and Webb, G. I., editors, *Encyclopedia of Machine Learning*, pages 789–795. Springer.

[Gao et al., 2008] Gao, J., Fan, W., Jiang, J., and Han, J. (2008). Knowledge transfer via multiple model local structure mapping. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 283–291.

[Geman et al., 1992] Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58.

[Griffin and Tversky, 1992] Griffin, D. and Tversky, A. (1992). The weighing of evidence and the determinants of confidence. *Cognitive Psychology*, 24(3):411 – 435.

[Hanley and McNeil, 1982] Hanley, J. A. and McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36.

[Hauskrecht et al., 2013] Hauskrecht, M., Batal, I., Valko, M., Visweswaran, S., Cooper, G. F., and Clermont, G. (2013). Outlier detection for patient monitoring and alerting. *Journal of Biomedical Informatics*, 46(1):47–55.

[Hauskrecht and Fraser, 1998] Hauskrecht, M. and Fraser, H. (1998). Modeling treatment of ischemic heart disease with partially observable markov decision processes. In *Proceedings of the Annual American Medical Informatics Association Symposium*, pages 538–542.

[Hauskrecht and Fraser, 2000] Hauskrecht, M. and Fraser, H. (2000). Planning treatment of ischemic heart disease with partially observable markov decision processes. *Artificial Intelligence in Medicine*, 18(3):221–244.

[Hauskrecht et al., 2010] Hauskrecht, M., Valko, M., Batal, I., Clermont, G., Visweswaram, S., and Cooper, G. (2010). Conditional outlier detection for clinical alerting. In *Proceedings of the Annual American Medical Informatics Association Symposium*, pages 286 – 290.

[Herbrich et al., 1999] Herbrich, R., Graepel, T., and Obermayer, K. (1999). Support vector learning for ordinal regression. In *International Conference on Artificial Neural Networks*, pages 97–102.

[Hoerl and Kennard, 1981] Hoerl, A. E. and Kennard, R. W. (1981). Ridge regression. advances algorithms and applications. *American Journal of Mathematical and Management Sciences*, 1(1):5–83.

[Jiang and Zhai, 2007] Jiang, J. and Zhai, C. (2007). Instance weighting for domain adaptation in nlp. In *Annual Meeting of the Association for Computational Linguistics*, pages 264–271.

[Joachims, 2002] Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142.

[Koller and Friedman, 2009] Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

[Lawrence and Platt, 2004] Lawrence, N. D. and Platt, J. C. (2004). Learning to learn with the informative vector machine. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 65–, New York, NY, USA. ACM.

[Lee et al., 2007] Lee, S.-I., Chatalbashev, V., Vickrey, D., and Koller, D. (2007). Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the International Conference on Machine Learning*, volume 227 of *ACM International Conference Proceedings Series*, pages 489–496. ACM.

[Lewis and Gale, 1994] Lewis, D. D. and Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Proceedings of the 17th ACM International Conference on*

Research and Development in Information Retrieval, pages 3–12, Dublin, IE. Springer Verlag, Heidelberg, DE.

[Lin et al., 2007] Lin, C., Weng, R. C., and Keerthi, S. S. (2007). Trust region newton method for large-scale logistic regression. In *Proceedings of the 24th International Conference on Machine Learning*.

[MacKay, 1992] MacKay, D. J. C. (1992). Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604.

[McCullagh and Nelder, 1989] McCullagh, P. and Nelder, J. A. (1989). *Generalized linear models (Second edition)*. London: Chapman & Hall.

[Mihalkova et al., 2007] Mihalkova, L., Huynh, T., and Mooney, R. J. (2007). Mapping and revising markov logic networks for transfer learning. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, AAAI'07, pages 608–614. AAAI Press.

[Nguyen et al., 2011a] Nguyen, Q., Valizadegan, H., and Hauskrecht, M. (2011a). Learning classification with auxiliary probabilistic information. In *IEEE International Conference on Data Mining*, pages 477–486.

[Nguyen et al., 2011b] Nguyen, Q., Valizadegan, H., and Hauskrecht, M. (2011b). Sample-efficient learning with auxiliary class-label information. In *Proceedings of the Annual American Medical Informatics Association Symposium*, pages 1004–1012.

[Nguyen et al., 2013] Nguyen, Q., Valizadegan, H., and Hauskrecht, M. (2013). Learning classification models with soft-label information. *Journal of American Medical Informatics Association*.

[O'Hagan et al., 2007] O'Hagan, A., Buck, C., Daneshkhan, A., Eiser, R., Garthwaite, P., Jenkinson, D., Oakley, J., and Rakow, T., editors (2007). *Uncertainty judgements Eliciting experts' probabilities*. John Wiley and Sons.

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

[Pearl, 1985] Pearl, J. (1985). Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of Cognitive Science Society (CSS-7)*.

[Platt, 1999] Platt, J. C. (1999). Advances in kernel methods. chapter Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pages 185–208. MIT Press, Cambridge, MA, USA.

[Raykar et al., 2010] Raykar, V. C., Yu, S., Zhao, L. H., Valadez, G. H., Florin, C., Bogoni, L., and Moy, L. (2010). Learning from crowds. *Journal of Machine Learning Research*, 11:1297–1322.

[Roy and McCallum,] Roy, N. and McCallum, A. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the 18th International Conferenceon on Machine Learning*.

[Ruckert and Kramer, 2008] Ruckert, U. and Kramer, S. (2008). Kernel-based inductive transfer. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, volume 5212 of *Lecture Notes in Computer Science*, pages 220–233. Springer.

[Schapire, 1990] Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, 5(2):197–227.

[Scholkopf and Smola, 2001] Scholkopf, B. and Smola, A. J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

[Settles, 2010] Settles, B. (2010). Active learning literature survey. Technical report.

[Settles and Craven, 2008] Settles, B. and Craven, M. (2008). An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079.

[Settles et al., 2008] Settles, B., Craven, M., and Ray, S. (2008). Multiple-instance active learning. In *In Advances in Neural Information Processing Systems*, pages 1289–1296. MIT Press.

[Seung et al., 1992] Seung, H. S., Opper, M., and Sompolinsky, H. (1992). Query by committee. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 287–294, Pittsburgh, Pennsylvania. ACM Press.

[Sheng et al., 2008] Sheng, V. S., Provost, F., and Ipeirotis, P. G. (2008). Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 614–622, New York, NY, USA. ACM.

[Silver, 2001] Silver, D. L. (2001). Selective functional transfer: Inductive bias from related tasks. In *International Conference on Artificial Intelligence and Soft Computing*, pages 182–189. ACTA Press.

[Smyth, 1995] Smyth (1995). Learning with probabilistic supervision. *Computational Learning Theory and Natural Learning System 3*, pages 163–182.

[Smyth et al., 1995] Smyth, Fayyad, Burl, Perona, and Baldi (1995). Inferring ground truth from subjective labeling of venus images. *Advances in Neural Information Processing Systems*, pages 1085–1092.

[Snow et al., 2008] Snow, R., O'Connor, B., Jurafsky, D., and Ng, A. Y. (2008). Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In

Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 254–263, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Suantak et al., 1996] Suantak, L., Bolger, F., and Ferrell, W. R. (1996). The hard-easy effect in subjective probability calibration. *Organizational Behavior and Human Decision Processes*, 67(2):201 – 221.

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of The Royal Statistical Society, Series B*, 58(1):267–288.

[Tomanek and Olsson, 2009] Tomanek, K. and Olsson, F. (2009). A web survey on the use of active learning to support annotation of text data. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies Workshop on Active Learning for Natural Language Processing*, pages 45–48, Boulder, Colorado. Association for Computational Linguistics.

[Tong and Koller, 2000] Tong, S. and Koller, D. (2000). Active learning for parameter estimation in bayesian networks. In *Advances in Neural Information Processing Systems*, pages 647–653. MIT Press.

[Valizadegan and Jin, 2007] Valizadegan, H. and Jin, R. (2007). Generalized maximum margin clustering and unsupervised kernel learning. In *Advances in Neural Information Processing Systems*, pages 1417–1424, Cambridge, MA. MIT Press.

[Valizadegan et al., 2012] Valizadegan, H., Nguyen, Q., and Hauskrecht, M. (2012). Learning medical diagnosis models from multiple experts. In *Proceedings of the Annual American Medical Informatics Association*, pages 921–30.

[Valizadegan et al., 2013] Valizadegan, H., Nguyen, Q., and Hauskrecht, M. (2013). Learning classification models from multiple experts. *Journal of Biomedical Informatics*, pages 1125–1135.

[Valko and Hauskrecht, 2010] Valko, M. and Hauskrecht, M. (2010). Feature importance analysis for patient management decisions. In *Proceedings of the 13th International Congress on Medical Informatics*, pages 861–865.

[Vapnik, 1995] Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

[Warkentin, 2003] Warkentin, T. (2003). Heparin-induced thrombocytopenia: pathogenesis and management. *British Journal of Haematology*, pages 535 – 555.

[Warkentin et al., 2000] Warkentin, T., Sheppard, J., and Horsewood, P. (2000). Impact of the patient population on the risk for heparin-induced thrombocytopenia. *Blood*, pages 1703 – 1708.

[Welinder et al., 2010] Welinder, P., Branson, S., Belongie, S., and Perona, P. (2010). The multidimensional wisdom of crowds. In *Advances in Neural Information Processing Systems*.

[Whitehill et al., 2009] Whitehill, J., Ruvolo, P., fan Wu, T., Bergsma, J., and Movellan, J. (2009). Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise. In *Advances in Neural Information Processing Systems*, pages 2035–2043.

[Yan et al., 2010] Yan, Y., Fung, G., Dy, J., and Rosales, R. (2010). Modeling annotator expertise: Learning when everybody knows a bit of something. In *International Conference on Artificial Intelligence and Statistics*.

[Yu et al., 2005] Yu, K., Tresp, V., and Schwaighofer, A. (2005). Learning gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pages 1012–1019, New York, NY, USA. ACM.

[Zadrozny, 2004] Zadrozny, B. Z. (2004). Learning and evaluating classifiers under sample selection bias. In *International Conference on Machine Learning*, pages 903–910.

[Zhang and Yeung, 2010] Zhang, Y. and Yeung, D.-Y. (2010). A convex formulation for learning task relationships in multi-task learning. In *Conference on Uncertainty in Artificial Intelligence*.

[Zhu, 2005] Zhu, X. (2005). *Semi-supervised learning with graphs*. PhD thesis, Pittsburgh, PA, USA. AAI3179046.

[Zou and Hastie, 2005] Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320.