UNIVERSITY OF COPENHAGEN
FACULTY OF SCIENCE

# PhD thesis

Jan Kremer

# Active and Adaptive Learning from Biased Data with Applications in Astronomy

Supervisor: Christian Igel

Submitted: 10/06/2016

**Summary**

This thesis addresses the problem of machine learning from biased datasets in the context of astronomical applications. In astronomy there are many cases in which the training sample does not follow the true distribution. The thesis examines different types of biases and proposes algorithms to handle them.

During learning and when applying the predictive model, active learning enables algorithms to select training examples from a pool of unlabeled data and to request the labels. This allows for selecting examples that maximize the algorithm's accuracy despite an initial bias in the training set. Against this background, the thesis begins with a survey of active learning algorithms for the support vector machine.

If the cost of additional labeling is prohibitive, unlabeled data can often be utilized instead and the sample selection bias can be overcome through domain adaptation, that is, minimizing the discrepancy between training sample and the true distribution. A simple method consists of weighting the elements of the training sample such that the empirical risk becomes an unbiased estimator of the true distribution's risk. The respective weights can be computed as the probability density ratio of training and test distribution. A model selection criterion—which is known in the context of kernel-based weight estimators—is proposed to be combined with a nearest neighbor density ratio estimator. It is shown to compare favorably to alternative approaches when applied to large-scale problems with low-dimensional feature spaces: a common setting in astronomical applications such as photometric redshift estimation.

Another form of bias stems from label noise. This thesis considers the scenario in which unreliable labels can be replaced by highly accurate labels at a certain cost. This is, for example, the case in crowd-sourcing, where unreliable labelers can be corrected by experts, or in astronomy, where a labeling based on photometric data can be improved by spectroscopic observations. An algorithm to actively select objects for correction under a limited re-labeling budget is presented. It is shown empirically to converge faster to the maximally attainable accuracy than the state-of-the-art.

# Acknowledgments

This thesis would not have been possible without the help of many people, the most important of whom I would like to mention here. First and foremost, I thank my supervisor Christian Igel. He was always available for stimulating discussions and encouraged me when things failed. The diverse collection of pens he forgot on my desk bears witness to his constant support. I would also like to thank Kim Steenstrup Pedersen for serving as my second supervisor and for helping me whenever I needed it.

I thank Fei Sha for hosting me in his group at UCLA and for the great discussions we had. His enthusiasm for science is truly inspiring. Furthermore, I would like to thank my colleague Fabian Gieseke for his constant encouragement, and helping me especially in the beginning of this endeavor.

I thank my colleagues and office mates Niels Dalum Hansen, Oswin Krause and Kristoffer Stensbo-Smidt for great scientific and non-scientific discussions, blackboard sessions and comedy. Many thanks to Susan Nasirumbi Ipsen for helping me with administrative matters and the rest of the image section for making this a nice place to work at.

I thank my parents Otto and Margret for always supporting any of my interests. Last but not least, I would like to thank my girlfriend Adriana Ion for putting things into perspective and for being there for me.

I would also like to acknowledge the Danish Council for Independent Research for supporting this work through the project "SkyML".

# Contents

# Chapter 1

# Introduction

In many application areas the available datasets violate a basic assumption underlying most supervised machine learning algorithms: Training and test data are not drawn from the same distribution, that is, we have a sample $S_{\text{train}}$ of labeled examples $(x, y)$ drawn from a distribution $p_{\text{train}}$, which is different from the test distribution $p_{\text{test}}$. We aim at learning a hypothesis $h$ from $S_{\text{train}}$ which minimizes the generalization error over $p_{\text{test}}$. In general, if $p_{\text{train}}$ and $p_{\text{test}}$ are unrelated, this is not possible. In this thesis, we will consider scenarios in which this difference is caused by a biased sample selection or by noisy labeling. We show that if we make certain assumptions on the type of bias or label noise, or have the possibility to obtain additional labels, we can still learn from biased training data.

In astronomy, biased datasets are common. High-quality measurements that provide ground-truth labels are costly and only available for certain regions of the input space. This creates a sample selection bias. Furthermore, certain annotations have to be performed by human labelers who are often unreliable or make systematic errors. Learning algorithms which fail to address these biases will in many cases perform sub-optimally.

Suitable bias correction strategies depend on the type of bias and on the acquisition costs. If we can afford to take additional measurements, we can select candidates for observation by active learning—a learning method which chooses the examples it finds most useful to learn from. This has two advantages. First, we may need to only examine a fraction of examples in comparison to sampling uniformly at random from $S_{\text{train}}$. Second, we can compensate for a selection bias by selecting examples that are under-represented in the current training set $S_{\text{train}}$ with respect to $p_{\text{test}}$.

Often, however, it is not possible to label additional examples, for instance, because annotation costs are prohibitive. Each example might have to be examined by an expert or additional expensive measurements are needed to derive the correct label. An example from astronomy is redshift estimation. The redshift of a galaxy measures how much its observed wavelength is shifted towards longer wavelengths. We can use spectroscopy, which measures the photon count over a wide range of wavelengths, to determine the redshift with high precision. Spectroscopy, however, is very time-consuming, it can only measure few objects

at the same time, and its depth is limited. It is therefore very costly and the measurements are biased towards observations closer to the observer.

Photometric measurements, which measure an object's intensity in a few bands, provide an inexpensive alternative to spectroscopy. As they only require an image of the sky, many objects can be observed at the same time. Photometric data is also available for objects at great distance to the observer, but has the drawback of providing less information in comparison to spectroscopic data. Building a photometric redshift model based on known photometry-spectroscopy correspondences remains a challenge in astronomy. It is complicated by the selection bias caused by spectroscopic observations.

If an unlabeled sample drawn from $p_{\text{test}}$ is available at training time, we can minimize the discrepancy between training and test set by importance-weighting. This is a simple strategy that re-weights training examples so that the empirical risk of $h$ on $S_{\text{train}}$ becomes an unbiased estimator of the risk over the test distribution $p_{\text{test}}$. The respective weights can be computed as the probability density ratio between $p_{\text{train}}$ and $p_{\text{test}}$.

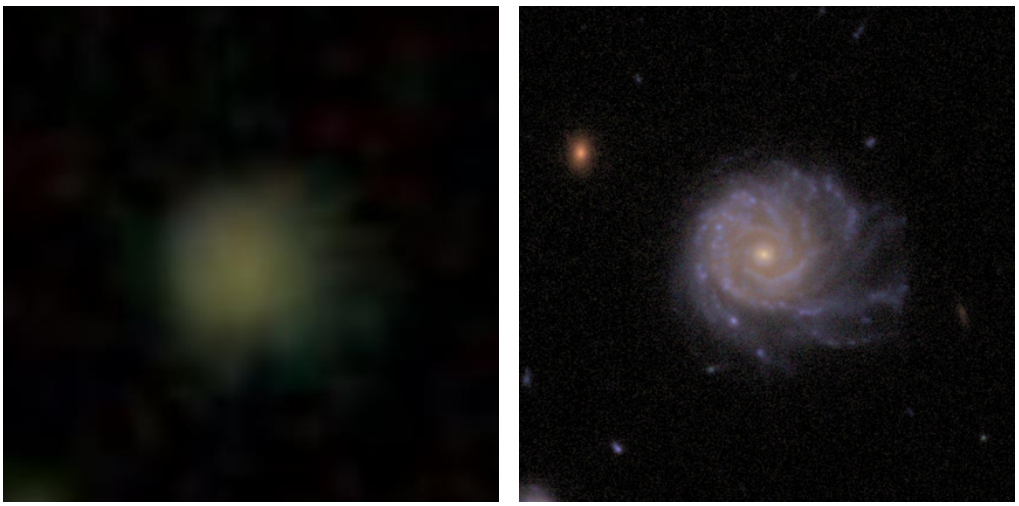
In some cases, expert annotations can be replaced by crowd-sourcing the labeling process. A successful example in the astronomical domain is the Galaxy Zoo project.[1] It collected millions of labels from volunteers who classified images observed by the Sloan Digital Sky Survey (SDSS) telescope and the Hubble Space Telescope. The task was to determine the galaxy's morphology which describes its visual appearance, for instance, whether it looks elliptical or spiral.

While crowd-sourcing provides an inexpensive mechanism to collect a large number of labels, the accuracy may suffer due to missing knowledge or low motivation of the annotators. But even when experts obtain the labels, the quality of the annotation is ultimately limited by the quality of the observation. Figure 1.1 shows an example of a galaxy observed by the ground-based SDSS telescope and by the space-based Hubble telescope. While an annotator might label this galaxy as elliptical based on the left image, it is clear from the right image, that it is in fact a spiral. When a new telescope becomes available, simply using a label which is based on an old measurement would not provide the most accurate annotation given the new observation. Both crowd-sourcing and low-quality measurements thus can lead to label noise.

The effect of label noise can be mitigated by minimizing a surrogate loss which takes into account a noise model. A simple and well-studied noise model is class-conditional label noise. It assumes that the probability of observing the wrong label is independent of the observed example given the true label. While this assumption is most likely violated in many real-world classification tasks, it has the advantage that the number of model parameters only depends on the number of classes. This makes it easier to learn than sample-dependent noise models.

In addition to using a label-noise robust model, one way to mitigate label noise is simply to correct false labels manually. In practice the number of corrections is usually limited by annotation costs. Active label correction aims at correcting the

[1] Visit `http://www.galaxyzoo.org` to classify a few galaxies yourself.



(a) Ground-based SDSS telescope (b) Hubble Space Telescope

Figure 1.1: A spiral galaxy as imaged by the Sloan Digital Sky Survey (SDSS) ground-based telescope on the left and by the Hubble Space Telescope on the right. Due to the lack of atmospheric distortion, the Hubble image is much better resolved and reveals the spiral character of the galaxy. An annotator would label this galaxy most likely as an elliptical if they based their annotation on the left image [Galaxy Zoo, 2012].

label which increases the accuracy of the classifier the most to lessen annotation costs. In the following we will present the background of the aforementioned concepts in more detail.

## 1.1 Active Learning

Active learning aims at optimally utilizing a limited budget for acquiring labeled data for supervised learning [Hanneke, 2009, Settles, 2012]. In many practical applications data are not scarce, but labels are. Acquiring them is costly, and, thus, the limited budget should be spent on labels that provide the maximum insight to the learning algorithm. It has been shown empirically and theoretically that in many cases one can do better than passive learning, that is, sampling uniformly at random from the training set [Beygelzimer et al., 2010].

Another case in which active learning can be beneficial is large-scale learning. Even if large amounts of labeled data are available, running time constraints may not allow for training on the entire set. By exploiting informative regions of the input space, performance gains over passive learning can be achieved. In addition, active learning can mitigate the effects of a sample selection bias by choosing examples that balance misrepresentation in the training set [Richards et al., 2012].

Any active learning strategy necessarily has the drawback that the acquired sample is not independent and identically distributed (i.i.d), an assumption which

underlies most supervised learning algorithms. Luckily, this bias is introduced by the learner itself and can thus be controlled, for instance, by importance-weighting [Sugiyama, 2005, Bach, 2006, Beygelzimer et al., 2009]: If an example is selected with too high probability it receives a small weight and vice versa, if it is sampled with too small probability it receives a large weight.

Support vector machines exhibit excellent empirical performance while being well-understood theoretically [Cortes and Vapnik, 1995, Steinwart and Christmann, 2008]. Due to their limited scalability to large datasets they can benefit greatly from active learning. Moreover, they can inform active learning strategies by providing a distance to the separating hyperplane as a measure of how informative an example is [Lewis and Gale, 1994b, Tong and Chang, 2001]. To correct a selection bias, sample weights can be introduced easily [Zadrozny et al., 2003].

## 1.2 Domain Adaptation

If there is no budget available to acquire additional labels, we can instead utilize unlabeled data. If we have access to a large sample of unlabeled data that follows the distribution of the test sample $p_{\text{test}}$, we can apply unsupervised domain adaptation techniques [Daume III and Marcu, 2006, Jiang, 2008]. These aim at minimizing the discrepancy between the marginal distributions $p_{\text{train}}(x)$ and $p_{\text{test}}(x)$ of training and test data [Ben-David et al., 2006].

As in active learning, importance weights can be used to alleviate the selection bias [Huang et al., 2007, Cortes et al., 2008]. By weighting examples in the training set by the probability density ratio of $p_{\text{test}}$ to $p_{\text{train}}$, an unbiased estimate of the true risk can be computed. As in this case the weights are solely determined by the data and not controlled by the algorithm as in active learning, the variance of the estimator has to be controlled with care [Cortes et al., 2010].

The naive solution is estimating the probability densities separately before computing their ratio. Sugiyama et al. [2010b] show empirically that this leads to sub-optimal results. This might be because a small error in the estimation of the denominator density can lead to a large error in the estimated ratio. Several authors have proposed to use a one-step procedure to estimate the importance weights directly [Sugiyama et al., 2008, Bickel et al., 2009]. Most estimators in the literature rely on kernel-based methods [Huang et al., 2007, Kanamori et al., 2012, Izbicki et al., 2014]. When applied to large-scale datasets these require sub-sampling or approximations.

Nearest neighbor-based algorithms constitute a conceptually simple alternative to kernel-based estimators [Lima et al., 2008, Loog, 2012]. They perform well when the dimensionality of the input space is low and sample sizes are large, a scenario which is common in astronomical applications such as photometric redshift estimation [Lima et al., 2008]. They rely on counting test examples within a radius that is defined by the $K$th neighbor in the training sample, thereby flexibly handling sparse regions in the input space.

## 1.3 Learning with Noisy Labels

Apart from sample selection bias, datasets may be affected by label noise [Frenay and Verleysen, 2014]. For instance, labels which are annotated by non-experts through crowd-sourcing are often unreliable [Raykar et al., 2010]. Even though these effects can be mitigated by a weighted averaging or by modeling the labelers, a certain amount of data will remain labeled incorrectly. Furthermore, even expert labelers may have difficulties annotating examples which are not well resolved, or belong to a class that is easily confused with another. For example, in astronomy the quality of observations is influenced by weather conditions, resolution and other factors. In the case of galaxy morphology classification, an object that has been identified in an astronomical survey as an elliptical galaxy might in fact be a spiral. The particular details that would have helped identifying the object could have been lost due to a limited resolution of the imaging telescope.

Label noise can be addressed by robust surrogate losses that assume a certain label noise model. This model can, for instance, depend on the true class [Bootkrajang and Kabán, 2012, Natarajan et al., 2013], the examples [Xiao et al., 2015], or the distance to an assumed true hyperplane separating the data [Du and Cai, 2015]. Class-conditional noise models work well when the label noise is asymmetric, that is, one class is mistaken more easily for another than vice versa. These models may make strong assumptions about the label noise, but require fewer parameters in comparison to sample-dependent models. Azadi et al. [2015] show that the impact of label noise can also be mitigated by regularization. Their regularizer, however, depends on the accuracy of a prior model pre-trained on well-labeled data. Filtering noisy examples is another approach [Boser et al., 1992]. It has the drawback that it might reduce the training sample size significantly.

## 1.4 Active Label Correction

Even when label-noise robust loss functions are minimized, these can only lessen the influence of label noise. It can be advantageous to correct as many labels as possible with the help of an expert [Zeng and Martinez, 2001]. In practice, any re-labeling budget will be limited and this raises the question of how it can be spent effectively.

Active label correction is similar in spirit to active learning [Rebbapragada et al., 2012]. The difference is that the queried examples are not unlabeled; their labels are given, but are not trustworthy. This scenario is also known as learning from weak teachers [Urner et al., 2012]. In contrast to the crowd-sourcing literature, the label noise is assumed to be inherent, not the result of a distribution of labels from different labelers [Sheng et al., 2008, Zhao et al., 2011].

Previous works have considered active learning strategies like uncertainty sampling to pick examples for correction without employing a label noise model [Rebbapragada et al., 2012]. In their purely theoretical work, Urner et al. [2012] considered label noise, where the noise depends on the labels of examples in the neighborhood.

## 1.5 Outline of the Thesis

First, the thesis discusses the problem of sample selection bias with an emphasis on astronomical applications. Chapter 2 gives an overview of machine learning and image analysis algorithms in astronomy. Then active learning is discussed, which can be used to mitigate the problem of sample selection bias by actively picking examples minimizing it. As the support vector machine has several advantageous theoretical properties and works well in practice, in chapter 3 existing methods that make use of this classifier in active learning scenarios are surveyed.

In chapter 4 a nearest neighbor-based algorithm to estimate probability density ratios is discussed. These estimates can be used as importance weights in domain adaptation. The algorithm can be combined with a model selection criterion, which originated from the kernel literature [Sugiyama et al., 2007], to select the optimal number of neighbors. It is shown that it empirically outperforms kernel-based methods in photometric redshift estimation, an astronomical application where sample sizes are large, but the input dimensionality is often small.

Then in chapter 5 the thesis examines the problem of label noise. Active label correction utilizes the current model to select examples for re-labeling by an expert. We develop an algorithmic framework in which examples that maximize the expected model change are chosen for correction. Within this framework we derive three active label correction algorithms and show that an algorithm which employs a label-robust maximum likelihood estimator performs best among these. Furthermore, we demonstrate empirically that in the case of class-conditional noise it outperforms algorithms that do not take into account the underlying label noise distribution. Chapter 6 concludes the thesis and gives an outlook on possible future work.

## 1.6 Included Manuscripts and Published Articles

The main contributions of this thesis are the following manuscripts and published articles:

- J. Kremer, K. Steenstrup Pedersen, and C. Igel. Active learning with support vector machines. *Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery*, 4(4):313–326, 2014

- J. Kremer, F. Gieseke, K. Steenstrup Pedersen, and C. Igel. Nearest neighbor density ratio estimation for large-scale applications in astronomy. *Astronomy and Computing*, 12:67–72, 2015

- J. Kremer, F. Sha, and C. Igel. Active label correction for class-conditional noise. Submitted, 2016a

- J. Kremer, K. Stensbo-Smidt, F. Gieseke, K. Steenstrup Pedersen, and C. Igel. Big universe, big data: Machine learning and image analysis for astronomy. Submitted, 2016b

# Chapter 2

# Big Universe, Big Data: Machine Learning and Image Analysis for Astronomy

*This chapter is based on the manuscript J. Kremer, K. Stensbo-Smidt, F. Gieseke, K. Steenstrup Pedersen, and C. Igel. Big universe, big data: Machine learning and image analysis for astronomy. Submitted, 2016b*

**Abstract**

Astrophysics and cosmology are rich with data. The advent of wide-area digital cameras on large aperture telescopes has led to ever more ambitious surveys of the sky. The data volume of an entire survey from a decade ago can now be acquired in a single night and real-time analysis is often desired. Thus, modern astronomy requires big data know-how, in particular it demands highly efficient machine learning and image analysis algorithms. But scalability is not the only challenge: Astronomy applications touch several current machine learning research questions, such as learning from biased data and dealing with label and measurement noise. We argue that this makes astronomy a great domain for computer science research, as it pushes the boundaries of data analysis. We present this exciting application area for data scientists. The article focuses on exemplary results, discusses main challenges, and highlights some recent methodological advancements in machine learning and image analysis triggered by astronomical applications.

## 2.1 Introduction

Astrophysics and cosmology are rich with data. The advent of wide-area digital cameras on large aperture telescopes has led to ever more ambitious surveys of the sky. The data volume of an entire survey from a decade ago can now be acquired in a single night and real-time analysis is often desired. Thus, modern astronomy requires big data know-how, in particular it demands highly efficient machine

Figure 2.1: An example of two morphology categories: on the left, the spiral galaxy M101; on the right, the elliptical galaxy NGC 1132 (credit: NASA, ESA, and the Hubble Heritage Team (STScI/AURA)-ESA/Hubble Collaboration).

learning and image analysis algorithms. But scalability is not the only challenge: Astronomy applications touch several current machine learning research questions, such as learning from biased data and dealing with label and measurement noise. We argue that this makes astronomy a great domain for computer science research, as it pushes the boundaries of data analysis. In the following, we will present this exciting application area for data scientists. We will focus on exemplary results, discuss main challenges, and highlight some recent methodological advancements in machine learning and image analysis triggered by astronomical applications.

## 2.2 Ever-Larger Sky Surveys

One of the largest astronomical surveys to date is the Sloan Digital Sky Survey (SDSS). Each night, the SDSS telescope produces 200 GB of data and now provides close to a million field images, in which more than 200 million galaxies, and even more stars, have been detected. A subset of the most visible galaxies formed the foundation for the crowd-sourced Galaxy Zoo project, in which volunteers classified more than 900,000 galaxies into one of six morphology categories (see Figure 2.1). The project was followed by Galaxy Zoo 2, which focused on the 300,000 brightest and largest of the original Galaxy Zoo galaxies [Willett et al., 2013]. Here, volunteers measured more detailed morphological features of the galaxies, resulting in 16 million classifications. The images and classifications are all publicly available, making this a highly valuable dataset for the development of image analysis and machine learning algorithms.

Upcoming surveys will provide even larger data volumes. *Euclid* is a space-based telescope selected by the European Space Agency (ESA) for launch in 2019, which will survey the sky for galaxies and map the large-scale structure of the Universe. It will generate about 300 GB of image data per night, each image with a resolution comparable to that of the Hubble Space Telescope. This will enable precision measurements of the large-scale structure and the expansion of the Universe, which can help solving some of the hardest and most exciting

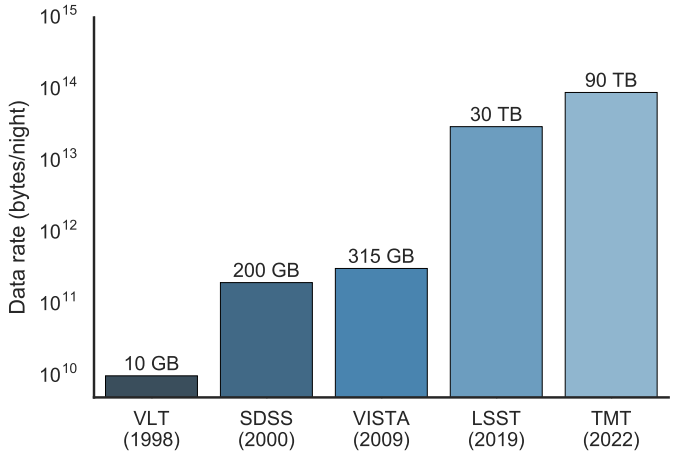



Figure 2.2: Increasing data volumes of existing and upcoming telescopes: Very Large Telescope (VLT), Sloan Digital Sky Survey (SDSS), Visible and Infrared Telescope for Astronomy (VISTA), Large Synoptic Survey Telescope (LSST) and Thirty Meter Telescope (TMT).

challenges faced by physicists today, for example explaining dark matter and dark energy.

Another promising future survey is the *Large Synoptic Survey Telescope* (LSST). It will deliver wide-field images of the sky, exposing galaxies that are too faint to be seen today. One of the main objectives of LSST is to discover *transients*, objects that change brightness over time-scales of seconds to months. These changes are due to a plethora of reasons; some may be regarded as uninteresting while others will be extremely rare events, which cannot be missed. LSST is expected to see millions of transients per night, which need to be detected in real-time to allow for follow-up observations. With staggering 30 TB of images being produced per night, efficient and accurate detection will be a major challenge. Figure 2.2 shows how the data rates have increased and will continue to increase as new surveys are initiated. Future missions will collect hundreds of measurements for each of more than a billion objects.

What do these measurements look like? Surveys usually take either *spectroscopic* or *photometric* observations, see Figure 2.3. Spectroscopy measures the photon count at thousands of wavelengths. The resulting spectrum allows for identifying the chemical components of the observed object and thus enables determining many interesting properties. Photometry takes images using a CCD, and these are typically acquired through only a handful of broad-band filters, making photometry much less informative than spectroscopy.

While spectroscopy provides measurements of high precision, it has two drawbacks: First, it is not as sensitive as photometry, meaning that distant or otherwise faint objects cannot be measured. Second, only few objects can be captured at the same time, making it more expensive than photometry, which allows for acquiring images of thousands of objects in a single image. Photometry

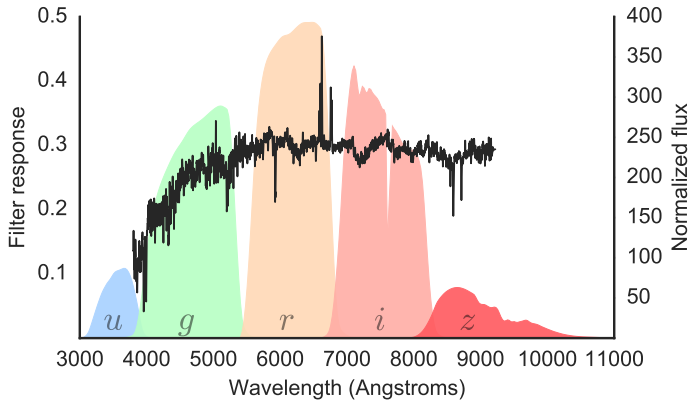



Figure 2.3: The spectrum of galaxy NGC 5750 (black line), as seen by SDSS, with the survey's five photometric broad-band filters $u$, $g$, $r$, $i$, and $z$, ranging from ultraviolet ($u$) to near-infrared ($z$). For each band the galaxy's brightness is captured in an image.

can capture objects that may be ten times fainter than what can be measured with spectroscopy. A faint galaxy is often more distant than a bright one—not just in space, but also in time. Discovering faint objects therefore offers the potential of looking further back into the history of the universe, over time-scales of billions of years. Thus, photometric observations are invaluable to cosmologists, as they help understanding the early universe.

Once these raw observations have been acquired, a pipeline of algorithms needs to extract information from them. Much image-based astronomy currently relies to some extent on visual inspection. A wide range of measurements is still carried out by humans, but needs to be addressed by automatic image analysis in light of growing data volumes. Examples are 3D orientation and chirality of galaxies, and the detection of large-scale features, such as jets and streams. Challenges in these tasks include image artifacts, spurious effects, and discerning between merging galaxy pairs and galaxies that happen to overlap along the line of sight. Current survey pipelines often have trouble correctly identifying these types of problems, which then propagate into the databases.

A particular challenge is that cosmology relies on scientific analyses of long-exposure images. As such, the interest in image analysis techniques for preprocessing and de-noising is naturally great. This is particularly important for the detection of faint objects with very low signal-to-noise ratios. Automatic object detection is vital to any survey pipeline, with reliability and completeness being essential metrics. Completeness refers to the amount of detected objects, whereas reliability measures how many of the detections are actual objects. Maximizing these metrics requires advanced image analysis and machine learning techniques. Therefore, data science for astronomy is a quickly evolving field gaining more and more interest. In the following, we will highlight some of its success stories.

## 2.3 Big Data Analysis in Astronomy

Machine learning methods are able to uncover the relation between input data (e.g., galaxy images) and outputs (e.g., physical properties of galaxies) based on input-output samples, and they have already proved successful in astrophysical contexts. For example, Mortlock et al. [2011] use Bayesian analysis to find the most distant quasar to date. These are extremely bright objects forming at the center of large galaxies and thus, are still visible from great distances. They are, however, also extremely rare. In this case, Bayesian model comparison has helped the scientists to select a few most likely objects for re-observation from thousands of plausible candidates.

Another astrophysical application is photometric redshift estimation. Redshift is caused by the Doppler effect, which shifts the spectrum of an object towards longer wavelengths when it moves away from the observer. As the universe is expanding uniformly, we can infer the velocity of a galaxy by its redshift and, thus, its distance to Earth. Hence, redshift estimation is a useful tool for determining the geometry of the universe. Redshift can be determined with high precision by taking a spectroscopic measurement. An important question in astronomy is, whether it can also be inferred from photometric observations, since these are easier to acquire. We can build a training set from photometric measurements whose redshifts are spectroscopically confirmed and subsequently use, for instance, a neural network to learn a model for predicting the redshifts [Collister and Lahav, 2004].

With easy access to the publicly available large databases containing data from astronomical surveys, like SDSS, most big data-oriented work in astronomy naturally use these as the starting point. Most quantities in these databases are derived from images taken by the survey telescope(s). The derivation is usually carried out by the survey consortia, and offered freely to the entire scientific community with a delay of a year or less. The survey images themselves are, however, often also freely available online, and offer an enormous potential for image analysis to enable novel discoveries.

The measurement of galaxy morphologies from images is of major interest to cosmologists. Morphology can tell astronomers about a galaxy's formation and history, which are valuable pieces of information when trying to understand the Universe as a whole. While the majority of science on galaxy morphology traditionally involves visual inspection, image analysis algorithms are gaining momentum. In particular, convolutional neural networks (CNNs) have seen a growing use in astronomy in the past years. As an example, Galaxy Zoo 2 has been used to train a CNN to predict galaxy morphologies [Dieleman et al., 2015].

Image analysis is also widely applied in solar physics. The Sun is continuously being monitored in high resolution by multiple satellites. Image analysis plays a vital part in detection and classification of both sunspots and the solar eruptions, such as flares and coronal mass ejections, which they may produce. Eruptions can have serious consequences for the Earth, as millions of tons of charged particles are accelerated out from the Sun. An eruption can short-circuit satellites around the Earth, including the International Space Station, and, if large enough, electric

systems on the ground. This can knock out power grids for long periods of time, and especially navigation systems on planes crossing the polar regions are at risk during high solar activity. Solar eruptions are therefore monitored constantly by automated software; but not all events are detected [Robbrecht and Berghmans, 2004]. Continuous alert systems using advanced image analysis techniques hold the potential to increase safety world-wide.

This glimpse of success stories of big data analysis in astronomy is by no means exhaustive. An overview of machine learning methods that find application in astronomy can be found in the survey by Ball and Brunner [2010].

## 2.4 Astronomy Driving Data Science

In the following, we present three examples from our own work showing how astronomical data analysis can trigger methodological advancements in machine learning and image analysis.

### Describing the Shape of a Galaxy

A galaxy's morphology is difficult to quantize in a concise manner. It is a reasonable choice to assign a galaxy a class based on its appearance. Indeed, such a classification approach has been used since galaxies were first discovered: in a subjective way by manual inspection. More objective measures of morphology have been studied, but none have conveyed the same amount of information as the century-old classification scheme.

Image analysis does not only allow for automatic classification, but can also inspire new ways to look at morphology [Pedersen et al., 2013, Polsterer et al., 2015]. For instance, we examined how well one of the most fundamental measures of galaxy evolution, the star-formation rate, could be predicted from the *shape index*. The shape index measures the local structure around a pixel going from dark blobs over valley-, saddle point- and ridge-like structures to white blobs. It can thus be used as a measure of the local morphology on a per-pixel scale, see Figure 2.4. The study showed that the shape index does indeed capture some fundamental information about galaxies, which is missed by traditional methods.

### Dealing with Sample Selection Bias

A challenging theoretical and practical problem in machine learning is caused by *sample selection bias*. In supervised machine learning, the models are constructed based on labeled examples, that is, observations (e.g., images, spectra, photometric features) together with their outputs (also referred to as labels, e.g., the corresponding redshift or galaxy type). Most machine learning algorithms are built on the assumption that the training set has been sampled uniformly at random from the population of interest (i.e., training and future test data follow the same distribution). This allows for generalization, enabling the model built from labeled examples in the training set to accurately predict the target variables in an unlabeled test set. In real-life applications this assumption is

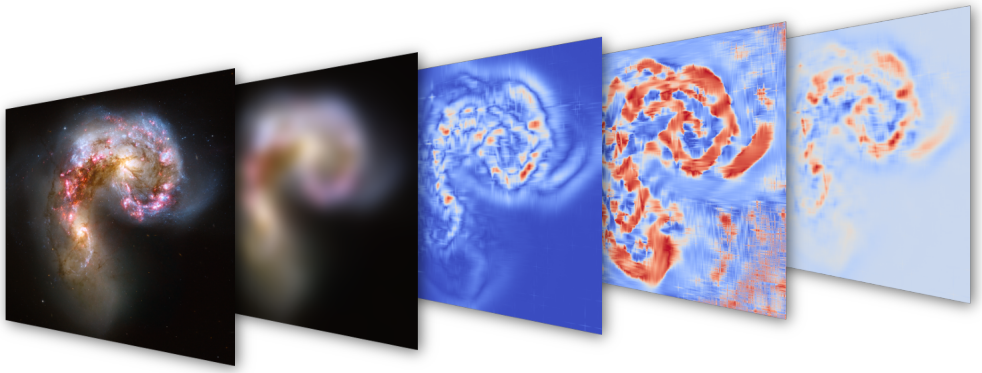


Figure 2.4: From left to right: The original image of a galaxy merger, the scale-space image of the galaxies, the curvedness (a measure of how pronounced the local structure is), the shape index, and finally the shape index weighted by the curvedness. The image shows the Antennae galaxies as seen by the Hubble Space Telescope (credit: NASA, ESA, and the Hubble Heritage Team (STScI/AURA)-ESA/Hubble Collaboration).

often violated—we refer to this as sample selection bias. Certain examples are more likely to be labeled than others due to factors like availability or acquisition cost regardless of their representation in the population. Sample selection bias can be very pronounced in astronomical data sets, and machine learning methods have to address this bias to achieve good generalization.

In astronomy, the mismatch between training and testing data can arise for several reasons. Often only training data sets from old surveys are initially available, while upcoming missions will probe never-before-seen regions in the astrophysical parameter space. Crowd-sourcing efforts like the Galaxy Zoo project are susceptible to a selection bias, too. As objects have to be recognizable to the citizen scientists, candidates to be labeled are restricted to a limited depth in space. Thus, again, certain regions of the input space will be underrepresented in the training sample.

Fortunately, acquiring large datasets without labels is typically not a problem anymore. Often we can obtain some labels, for example, by letting an expert annotate some data or by taking spectroscopic measurements, which are both costly procedures. This offers a remedy to the sample selection bias problem. We can either utilize unlabeled examples to improve our model, or we can obtain the labels of some unlabeled data points. As the latter is costly, we must try to find the observations that would improve our model the most after labeling.

When the learning algorithm is allowed to choose the examples that maximally improve the model, we speak of *active learning*. In the context of sample selection bias, we can then choose examples that minimize the difference between training and test distribution. Richards et al. [2012] have demonstrated how active learning helps to compensate sample selection bias when classifying variable stars (i.e., different types of stars that change their brightness over time).

When we do not have the budget or even the possibility to label additional examples, we can resort to *domain adaptation algorithms*. These methods assume that we not only have access to a labeled training set that may be subject to a selection bias, but also to an unlabeled test set that follows the distribution of the population, and the latter is used to correct for the bias. *Importance-weighting* is a simple, yet effective domain adaptation algorithm. The idea is to give more weight to examples in the training sample which lie in regions of the feature space that are underrepresented in the test sample and, likewise, give less weights to examples whose location in the feature space is overrepresented in the test set. If these weights are estimated correctly, the model we learn from the training data is an unbiased estimate of the model we would learn from a sample that follows the population's distribution. The challenge lies in estimating these weights reliably and efficiently. Given a sufficiently large sample, a simple strategy can be followed: Using a nearest neighbor-based approach, we can count the number of test examples that fall within a hypersphere whose radius is defined by the distance to the $K$th neighbor of a training example. The weight is then the ratio of the number of these test examples over $K$. This flexibly handles regions which are sparse in the training sample. In the case of redshift estimation, we could alleviate a selection bias by utilizing a large sample of photometric observations to determine the weights for the spectroscopically confirmed training set [Kremer et al., 2015].

### Scaling-up Nearest Neighbor Search

Nearest neighbor methods are not only useful to address the sample selection bias, they have proven to provide excellent prediction results in astrophysics and cosmology. For example, they are used to generate candidates for quasars at high redshift [Polsterer et al., 2013]. Nearest neighbor methods work particularly well when the number of training examples is high and the input space is low-dimensional. This makes them a good choice for analyzing large sky surveys where the objects are described by photometric features (e.g., the five intensities in the frequency bands shown in Figure 2.3). However, searching the nearest neighbors becomes a computational bottleneck in this big data setting.

In general, powerful compute servers can be employed to reduce the running time of machine learning algorithms, where the individual nodes conduct parts of the overall task. However, making use of supercomputers might become very expensive and parallelizing the work may not be straight-forward. There are several ways to address this issue, namely the application of special data structures, the approximation of the underlying problem, or the development of algorithms that allow for a massively-parallel implementation.

To compute the nearest neighbors for a given query object, spatial search structures such as k-d trees are an established way to reduce the computational requirements. If the input space dimensionality is moderate (say, up to 30), the running time can often be reduced by several orders of magnitude. One could also apply approximate nearest neighbor search [Arya et al., 1994]. However, we are interested in the exact solutions for our astronomical data analysis. Thus, we

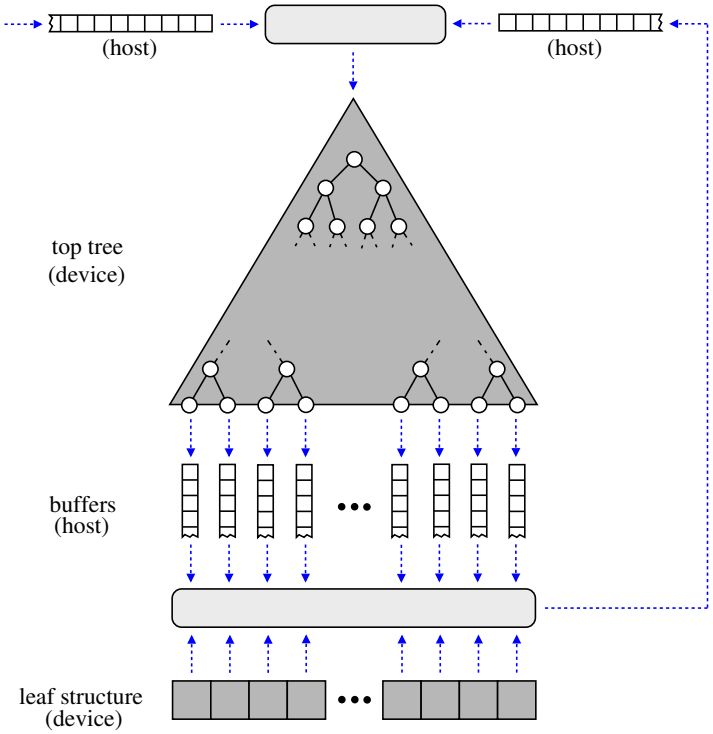



Figure 2.5: The *buffer k-d tree* data structure depicts an extension of classical $k$-d trees and can be used to efficiently process huge amounts of nearest neighbor queries using GPUs. The gray components (top tree and leaf structure) are stored on the device (GPU); the remaining ones (input/reinsert queue and buffers) are stored on the host CPU system. Nearest neighbor queries are repeatedly distributed to appropriate leaves and large chunks of queries are processed together in a massively-parallel manner on the GPU [Gieseke et al., 2014].

resort to inexpensive massively-parallel devices, graphics processing units (GPUs), to accelerate the involved computations. Unfortunately, nearest neighbor search based on spatial data structures cannot be parallelized in an obvious way. To this end, we developed a new algorithm that allows for efficient massively-parallel traversal of spatial search structures as sketched in Figure 2.5, which can achieve a significant running time reduction at a much lower cost compared to traditional parallel computing architectures. The corresponding framework can be used to efficiently search for nearest neighbors given large training and test sets with hundreds of millions of data points [Gieseke et al., 2014]. Such variants of classical approaches that can handle massive amounts of data efficiently and at low cost will be crucial for the upcoming data-intensive analyses in astronomy.

## 2.5 Physical Models vs. Machine Learning Models

The biggest concern data scientists meet when bringing forward data-driven machine learning models in astrophysics and cosmology is arguably lack of interpretability. There are two different approaches to predictive modeling in astronomy: physical modeling and data-driven modeling. The traditional one is to build physical models, which can incorporate all necessary astrophysical background knowledge. These models can be used for prediction, for example, by running Monte Carlo simulations. Ideally, this approach ensures that the predictions are physically plausible. In contrast, there may be the risk that extrapolations by a purely data-driven machine learning model violate physical laws. The decisive feature of physical models is that they also provide an understanding of *why* certain observations could have been made. This interpretability of predictions is typically not provided when using a machine learning approach.

Physical models have the drawbacks that they are difficult to construct and that inference may take a long time (e.g., in the case of Monte Carlo simulations). Most importantly, the quality of the predictions depends on the quality of the physical model, which is typically limited by necessary simplifications and incomplete scientific knowledge. In our experience, data-driven models typically outperform physical models in terms of prediction accuracy [Stensbo-Smidt et al., 2013, 2015]. Thus, we strongly advocate data-driven models when we are mainly interested in accurate predictions. And this is indeed often the case, for example, if we want to estimate properties of objects in the sky for quickly identifying observations worth a follow-up investigation or for conducting large-scale statistical analyses. Generic machine learning methods are not meant to replace physical modeling, because they typically do not provide scientific insights beyond the predicted values. Still, we argue that if prediction accuracy is what matters, one should favor the more accurate model, whether it is interpretable or not. Having said this, while the black-and-white portrayal of the two approaches may help to illustrate common misunderstandings between data scientists and physicists, it is of course shortsighted. Physical and machine learning modeling are not mutually exclusive: Physical models can inform machine learning algorithms, and machine learning can support physical modeling. A simple example for the latter is using machine learning to estimate the error residuals of a physical model [Pedersen et al., 2013].

Dealing with uncertainties is a major issue in astronomical data analysis. Data scientists are asked to provide error bars for their predictions and have to think about how to deal with input noise. In astronomy, both input and output data have (non-Gaussian) errors attached to them. Often these measurement errors have been quantified (e.g., by incorporating the weather conditions at the day the data was obtained), and it is desirable to consider these errors in the prediction. Bayesian modeling and Monte Carlo methods simulating physical models offer solutions, however, often they do not scale for big data. Alternatively, one can modify machine learning methods to process error bars, as attempted for nearest neighbor regression by modifying the distance function [Polsterer et al., 2013].

## 2.6 A Peek into the Future

The future looks bright for data science in astronomy. Within the next few years, image analysis and machine learning systems that can process terabytes of data in near real-time with high accuracy will be essential.

There are great opportunities for making novel discoveries, even in databases that have been available for decades. The volunteers of Galaxy Zoo have demonstrated this multiple times by discovering structures in the SDSS images that have later been confirmed to be new types of objects. These volunteers are not trained scientists, yet they make new scientific discoveries.

Even today, only a fraction of the images of SDSS have been inspected by humans. Without doubt, the data still hold many surprises, and upcoming surveys, such as LSST, are bound to image previously unknown objects. It will not be possible to manually inspect all images produced by these surveys, making advanced image analysis and machine learning algorithms of vital importance.

One may use such systems to answer questions like how many types of galaxies there are, what distinguishes the different classes, whether the current classification scheme is good enough, and whether there are important sub-classes or undiscovered classes. These questions require data science knowledge rather than astrophysical knowledge, yet the discoveries will still help astrophysics tremendously.

In this new data-rich era, astronomy and computer science can benefit greatly from each other. There are new problems to be tackled, novel discoveries to be made, and above all, new knowledge to be gained in both fields.

# Chapter 3

# Active Learning with Support Vector Machines

*This chapter is based on the article J. Kremer, K. Steenstrup Pedersen, and C. Igel. Active learning with support vector machines.* Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery*, 4(4):313–326, 2014*

**Abstract**

In machine learning, *active learning* refers to algorithms that autonomously select the data points from which they will learn. There are many data mining applications in which large amounts of unlabeled data are readily available, but labels (e.g., human annotations or results from complex experiments) are costly to obtain. In such scenarios, an active learning algorithm aims at identifying data points that, if labeled and used for training, would most improve the learned model. Labels are then obtained only for the most promising data points. This speeds up learning and reduces labeling costs. Support vector machine (SVM) classifiers are particularly well-suited for active learning due to their convenient mathematical properties. They perform linear classification, typically in a kernel-induced feature space, which makes measuring the distance of a data point from the decision boundary straightforward. Furthermore, heuristics can efficiently estimate how strongly learning from a data point influences the current model. This information can be used to actively select training samples. After a brief introduction to the active learning problem, we discuss different query strategies for selecting informative data points and review how these strategies give rise to different variants of active learning with SVMs.

## 3.1 Introduction

In many applications of supervised learning in data mining, huge amounts of unlabeled data samples are cheaply available while obtaining their labels for training a classifier is costly. To minimize labeling costs, we want to request labels only for potentially informative samples. These are usually the ones that we

expect to improve the accuracy of the classifier to the greatest extent when used for training. Another consideration is the reduction of training time. Even when all samples are labeled, we may want to consider only a subset of the available data because training the classifier of choice using all the data might be computationally too demanding. Instead of sampling a subset uniformly at random, which is referred to as *passive learning*, we would like to select informative samples to maximize accuracy with less training data. *Active learning* denotes the process of autonomously selecting promising data points to learn from. By choosing samples actively, we introduce a selection bias. This violates the assumption underlying most learning algorithms that training and test data are identically distributed: an issue we have to address to avoid detrimental effects on the generalization performance.

In theory, active learning is possible with any classifier that is capable of passive learning. This review focuses on the support vector machine (SVM) classifier. It is a state-of-the-art method, which has proven to give highly accurate results in the passive learning scenario and which has some favorable properties that make it especially suitable for active learning: (i) SVMs learn a linear decision boundary, typically in a kernel-induced feature space. Measuring the distance of a sample to this boundary is straightforward and provides an estimate of its informativeness. (ii) Efficient online learning algorithms make it possible to obtain a sufficiently accurate approximation of the optimal SVM solution without retraining on the whole dataset. (iii) The SVM can weight the influence of single samples in a simple manner. This allows for compensating the selection bias that active learning introduces.

## 3.2 Active Learning

In the following we focus on supervised learning for classification. There also exists a body of work on active learning with SVMs in other settings such as regression [Demir and Bruzzone, 2012] and ranking [Brinker, 2004, Yu, 2005]. A discussion of these settings is, however, beyond the scope of this article.

The training set is given by $\mathcal{L} = \{(x_1, y_1), ..., (x_\ell, y_\ell)\} \subset \mathcal{X} \times \mathcal{Y}$. It consists of $\ell$ labeled samples that are drawn independently from an unknown distribution $\mathcal{D}$. This distribution is defined over $\mathcal{X} \times \mathcal{Y}$, the cross product of a feature space $\mathcal{X}$ and a label space $\mathcal{Y}$, with $\mathcal{Y} = \{-1, 1\}$ in the binary case. We try to infer a hypothesis $f : \mathcal{X} \to \mathcal{Z}$ mapping inputs to a prediction space $\mathcal{Z}$ for predicting the labels of samples drawn from $\mathcal{D}$. To measure the quality of our prediction, we define a loss function $L : \mathcal{Z} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$. Thus, our learning goal is minimizing the expected loss

$$R(f) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[L(f(x), x, y)\Big] \ , \tag{3.1}$$

which is called the *risk* of $f$. We call the average loss over a finite sample $\mathcal{L}$ the *training error* or *empirical risk*. If a loss function does not depend on the second argument, we simply omit it.

In sampling-based active learning, there are two scenarios: stream-based and pool-based. In *stream-based active learning*, we analyze incoming unlabeled

samples sequentially, one sample at a time. Contrary, in *pool-based active learning* we have access to a pool of unlabeled samples at once. In this case, we can rank samples based on a selection criterion and query the most informative ones. Although, some of the methods in this review are also applicable to stream-based learning, most of them consider the pool-based scenario. In the case of pool-based active learning, we have, in addition to the labeled set $\mathcal{L}$, access to a set of $m$ unlabeled samples $\mathcal{U} = \{x_{\ell+1}, ..., x_{\ell+m}\}$. We assume that there exists a way to provide us with a label for any sample from this set (the probability of the label is given by $\mathcal{D}$ conditioned on the sample). This may involve labeling costs, and the number of queries we are allowed to make may be restricted by a budget. After labeling a sample, we simply add it to our training set.

In general, we aim at achieving a minimum risk by requesting as few labels as possible. We can estimate this risk by computing the average error over an independent test set not used in the training process. Ultimately, we hope to require less labeled samples for inferring a hypothesis performing as well as a hypothesis generated by passive learning on $\mathcal{L}$ and completely labeled $\mathcal{U}$.

In practice, one can profit from an active learner if only few labeled samples are available and labeling is costly, or when learning has to be restricted to a subset of the available data to render the computation feasible. A list of real-world applications is given in the general active learning survey [Settles, 2012] and in a review paper which considers active learning for natural language processing [Olsson, 2009].

Active learning can also be employed in the context of transfer learning [Shi et al., 2008]. In this setting, samples from the unlabeled target domain are selected for labeling and included in the source domain. A classifier trained on the augmented source dataset can then exploit the additional samples to increase its accuracy in the target domain. This technique has been used successfully, for example, in an astronomy application [Richards et al., 2012] to address a *sample selection bias*, which causes source and target probability distributions to mismatch [Quionero-Candela et al., 2009].

## 3.3 Support Vector Machine

Support vector machines (SVMs) are state-of-the-art classifiers [Boser et al., 1992, Cortes and Vapnik, 1995, Mammone et al., 2009, Salcedo-Sanz et al., 2014, Schölkopf and Smola, 2002, Shawe-Taylor and Cristianini, 2004]. They have proven to provide well-generalizing solutions in practice and are well understood theoretically [Steinwart and Christmann, 2008]. The *kernel trick* [Schölkopf and Smola, 2002] allows for an easy handling of diverse data representations (e.g., biological sequences or multimodal data). Support vector machines perform linear discrimination in a kernel-induced feature space and are based on the idea of large margin separation: they try to maximize the distance between the decision boundary and the correctly classified points closest to this boundary. In the following, we formalize SVMs to fix our notation, for a detailed introduction we refer to the recent WIREs articles [Mammone et al., 2009, Salcedo-Sanz et al., 2014].

An SVM for binary classification labels an input $x$ according to the sign of a decision function of the form

$$f(x) = \langle \mathbf{w}, \phi(x) \rangle + b = \sum_{i=1}^{\ell} \alpha_i y_i \kappa(x_i, x) + b \ , \tag{3.2}$$

where $\kappa$ is a positive semi-definite kernel function [Aronszajn, 1950] and $\phi(x)$ is a mapping $\mathcal{X} \to \mathcal{F}$ to a kernel-induced Hilbert space $\mathcal{F}$ such that $\kappa(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$. We call $\mathcal{F}$ the feature space, which includes the weight vector $\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \phi(x_i) \in \mathcal{F}$. The training patterns $x_i$ with $\alpha_i > 0$ are called *support vectors*. The decision boundary is linear in $\mathcal{F}$ and the offset from the origin is controlled by $b \in \mathbb{R}$.

The distance of a pattern $(x, y)$ from the decision boundary is given by $|f(x)/\|\mathbf{w}\||$. We call $yf(x)$ the *functional margin* and $yf(x)/\|\mathbf{w}\|$ the *geometric margin*: a positive margin implies correct classification. Let us assume that the training data $\mathcal{L}$ is linearly separable in $\mathcal{F}$. Then $m(\mathcal{L}, f) = \min_{(x,y)\in\mathcal{L}} yf(x)$ defines the margin of the whole data set $\mathcal{L}$ with respect to $f$ (in the following we do not indicate the dependency on $\mathcal{L}$ and $f$ if it is clear from the context). We call the feature space region $\{x \in \mathcal{X} \mid |f(x)| \leq 1\}$ the *margin band* [Campbell et al., 2000].

A *hard margin SVM* computes the linear hypothesis that separates the data and yields a maximum margin by solving

$$\begin{aligned} \max_{\mathbf{w},b,\gamma} \quad & \gamma \\ \text{subject to} \quad & y_i(\langle \mathbf{w}, \phi(x_i) \rangle + b) \geq \gamma \quad , \quad i = 1, \ldots, \ell \\ & \|\mathbf{w}\| = 1 \end{aligned} \tag{3.3}$$

with $\mathbf{w} \in \mathcal{F}$, $b \in \mathbb{R}$ and $\gamma \in \mathbb{R}$ [Shawe-Taylor and Cristianini, 2004]. Instead of maximizing $\gamma$ and keeping the norm of $\mathbf{w}$ fixed to one, one can equivalently minimize $\|\mathbf{w}\|$ and fix a *target margin*, typically $\gamma = 1$.

In general, we cannot or do not want to separate the full training data correctly. *Soft margin SVMs* mitigate the concept of large margin separation. They are best understood as the solutions of the regularized risk minimization problem

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{\ell} C_i L_{\text{hinge}}(\langle \mathbf{w}, \phi(x_i) \rangle + b, y_i) \ . \tag{3.4}$$

Here, $L_{\text{hinge}}(f(x_i), y_i) = \max(0, 1 - y_i f(x_i))$ denotes the *hinge loss*. An optimal solution $\mathbf{w}^* = \sum_{i=1}^{\ell} \alpha_i^* y_i \phi(x_i)$ has the property that $0 \leq \alpha_i^* \leq C_i$ for $i = 1, \ldots, \ell$. For soft margin SVMs, the patterns in $\mathcal{L}$ need not be linearly separable in $\mathcal{F}$. If they are, increasing the $C_i$ until an optimal solution satisfies $\alpha_i^* < C_i$ for all $i$ leads to the same hypothesis as training a hard margin SVM. Usually, all samples are given the same weight $C_i = C$.

## 3.4 Uncertainty Sampling

It seems to be intuitive to query labels for samples that cannot be easily classified using our current classifier. Consider the contrary: if we are very certain about the class of a sample, then we might regard any label that does not reflect our expectation as noise. On the other hand, uncovering an expected label would not make us modify our current hypothesis.

*Uncertainty sampling* was introduced by Lewis and Gale [Lewis and Gale, 1994a]. The idea is that the samples the learner is most uncertain about provide the greatest insight into the underlying data distribution. Figure 3.1 shows an example in the case of an SVM. Among the three different unlabeled candidates, our intuition may suggest to ask for the label of the sample closest to the decision boundary: the labels of the other candidates seem to clearly match the class of the samples on the respective side or are otherwise simply mislabeled. In the following, we want to show how this intuitive choice can be justified and how it leads to a number of active learning algorithms that make use of the special properties of an SVM.

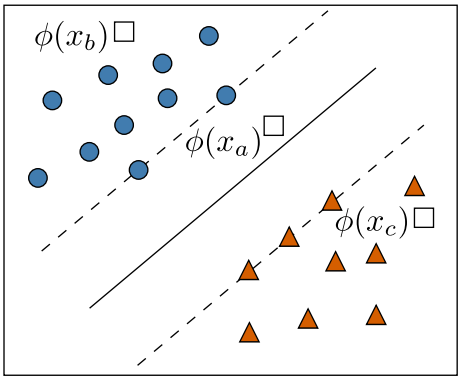



Figure 3.1: The three rectangles depict unlabeled samples while the blue circles and orange triangles represent positively and negatively labeled samples, respectively. Intuitively, the label of the sample $x_a$ might tell us the most about the underlying distribution of labeled samples, since in the feature space, $\phi(x_a)$ is closer to the decision boundary than $\phi(x_b)$ or $\phi(x_c)$.

### Version Space

The *version space* is a construct that helps to keep track of all hypotheses that are able to perfectly classify our current observations [Mitchell, 1982]. Thus, for the moment, we assume that our data are linearly separable in the feature space. The idea is to speed up learning by selecting samples in a way that minimizes the version space rapidly with each labeling.

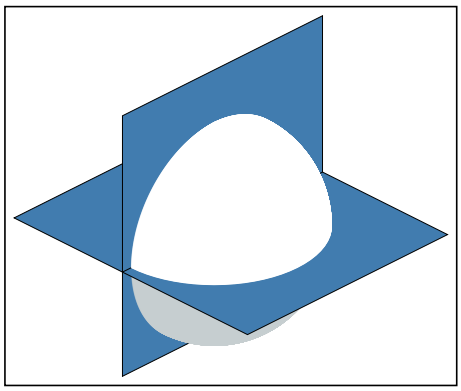
We can express the hard margin SVM classifier (3.3) in terms of a geometric representation of the version space. For this purpose, we restrict our consideration to hypotheses $f(x) = \langle \mathbf{w}, \phi(x) \rangle$ without bias (i.e., $b = 0$). The following results, however, can be extended to SVMs with $b \neq 0$.

The version space $\mathcal{V}(\mathcal{L})$ refers to the subset of $\mathcal{F}$ that includes all hypotheses consistent with the training set $\mathcal{L}$ [Mitchell, 1982]:

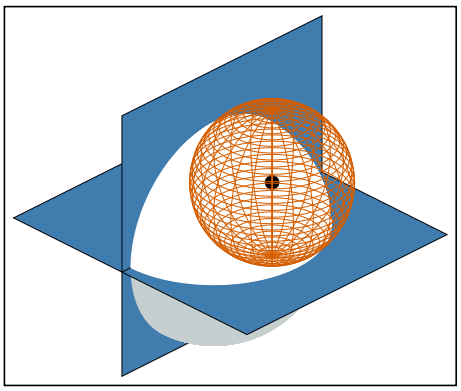
$$\mathcal{V}(\mathcal{L}) := \left\{ \mathbf{w} \in \mathcal{F} \mid \|\mathbf{w}\| = 1,\ yf(x) > 0, \forall (x, y) \in \mathcal{L} \right\} \tag{3.5}$$

In this representation, we can interpret the hypothesis space as the unit hypersphere given by $\|\mathbf{w}\| = 1$. The surface of the hypersphere includes all possible hypotheses classifying samples that are mapped into the feature space $\mathcal{F}$. We define $\Lambda(\mathcal{V})$ as the area the version space occupies on the surface of this hypersphere. This is depicted in Figure 3.2. The hypothesis space is represented by the big sphere. The white front, which is cut out by the two hyperplanes, depicts the version space.

Each sample $x$ can be interpreted as defining a hyperplane through the origin of $\mathcal{F}$ with the normal vector $\phi(x)$. Each hyperplane divides the feature space into two half-spaces. Depending on the label $y$ of the sample $x$, the version space is restricted to the surface of the hypersphere that lies on the respective side of the hyperplane. For example, a sample $x$ that is labeled with $y = +1$, restricts the version space to all $\mathbf{w}$ on the unit hypersphere for which $\langle \mathbf{w}, \phi(x) \rangle > 0$, i.e., the ones that lie on the positive side of the hyperplane defined by the normal vector $\phi(x)$. Thus, the version space is defined by the intersection of all half-spaces and the surface of the hypothesis hypersphere. Figure 3.2a illustrates this geometric relationship.



(a) The sphere depicts the hypothesis space. The two hyperplanes are induced by two labeled samples. The version space is the part of the sphere surface (in white) that is on one side of each hyperplane. The respective side is defined by its label.



(b) The center (in black) of the orange sphere depicts the SVM solution within the version space. It has the maximum distance to the hyperplanes delimiting the version space. The normals of these hyperplanes, which are touched by the orange sphere, correspond to the support vectors.

Figure 3.2: Geometric representation of the version space in 3D following Tong [Tong, 2001].

If we consider a feature mapping with the property $\forall x, z \in \mathcal{X} : \|\phi(x)\| = \|\phi(z)\|$, such as normalized kernels, including the frequently used Gaussian kernel,

then the SVM solution (3.3) has a particularly nice geometric interpretation in the version space. Under this condition, the decision function $f(x) = \langle \mathbf{w}, \phi(x) \rangle$ maximizing the margin $m(\mathcal{L}, f)$ (i.e., the minimum $y\langle \mathbf{w}, \phi(x) \rangle$ over $\mathcal{L}$) also maximizes the minimum distance between $\mathbf{w}$ and any hyperplane defined by a normal $\phi(x_i)$, $i = 1, \ldots, \ell$. The solution is a point within the version space, which is the center of a hypersphere, depicted in orange in Figure 3.2b. This hypersphere yields the maximum radius possible without intersecting the hyperplanes that delimit the version space. The radius is given by $r = \frac{y\langle \mathbf{w}, \phi(x) \rangle}{\|\phi(x)\|}$, where $\phi(x)$ is any support vector. Changing our perspective, we can interpret the normals $\phi(x_i)$ of the hyperplanes touching the hypersphere as points in the feature space $\mathcal{F}$. Then, these are exactly the support vectors since they have the minimum distance to our decision boundary defined by $\mathbf{w}$. This distance is $m(\mathcal{L}, f)$, which turns out to be proportional to the radius $r$.

### Implicit Version Space

An explicit version space, as defined above, only exists if the data are separable, which is often not the case in practice. The Bayes optimal solution need not have vanishing training error (as soon as under $\mathcal{D}$ we have $p(y_1|x) \geq p(y_2|x) > 0$ for some $x \in \mathcal{X}$ and $y_1, y_2 \in \mathcal{Y}$ with $y_1 \neq y_2$). Thus, minimizing the version space might exclude hypotheses with non-zero training error that are in fact optimal. In agnostic active learning [Balcan et al., 2006], we do not make the assumption of an existing optimal zero-error decision boundary. An algorithm that is theoretically capable of active learning in an agnostic setting is the $A^2$-algorithm [Balcan et al., 2006]. Here, a hypothesis cannot be deleted due to its disagreement with a single sample. If, however, all hypotheses that are part of the current version space agree on a region within the feature space, this region can be discarded. For each hypothesis the algorithm keeps an upper and a lower bound on its training error (see the work by Balcan et al. [Balcan et al., 2006] for details). It subsequently excludes all hypotheses which have a lower bound that is higher than the global minimal upper bound. Despite being intractable in practice, this algorithm forms the basis of some important algorithms compensating the selection bias as discussed below.

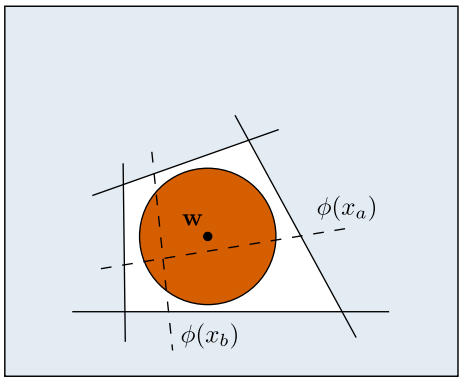
### Uncertainty-Based Active Learning

Although the version space is restricted to separable problems, it motivates many general active selection strategies. Which samples should we query to reduce the version space? As we have seen previously, each labeled sample that becomes a support vector restricts the version space to one side of the hyperplane it induces in $\mathcal{F}$. If we do not know the correct label of a sample in advance, we should always query the sample that ideally halves the version space. This is a safe choice as we will reduce it regardless of the label. Computing the version space in a high-dimensional feature space is usually intractable, but we can approximate it efficiently using the SVM. In the version space, the SVM solution $\mathbf{w}$ is the center of the hypersphere touching the hyperplanes induced by the support vectors.

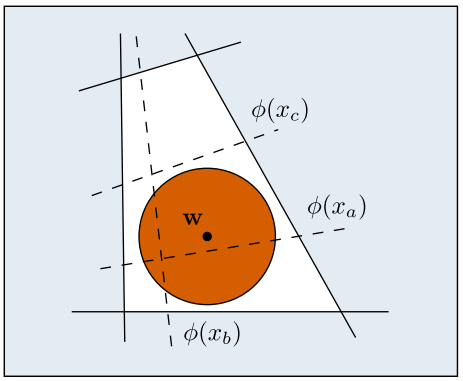
Each hyperplane delimits the version space. Assuming that the center of this hypersphere is close to the center of the version space, we can use it as an approximation. If we now choose a hyperplane that is close to this center, we approximately bisect the version space. Therefore, we want to query the sample $\hat{x}$ that induces a hyperplane as close to $\mathbf{w}$ as possible:

$$\hat{x} = \underset{x \in \mathcal{U}}{\operatorname{argmin}} \, |\langle \mathbf{w}, \phi(x) \rangle| = \underset{x \in \mathcal{U}}{\operatorname{argmin}} \, |f(x)| \tag{3.6}$$

This strategy queries the sample closest to the current decision boundary and is called *Simple Margin* [Tong, 2001]. Figure 3.3 shows this principle geometrically, projected to two dimensions.




(a) *Simple Margin* will query the sample that induces a hyperplane lying closest to the SVM solution. In this case, it would query sample $x_a$.




(b) Here the SVM does not provide a good approximation of the version space area. *Simple Margin* would query sample $x_a$ while $x_c$ might have been a more suitable choice.

Figure 3.3: The version space area is shown in white, the solid lines depict the hyperplanes induced by the support vectors, the center of the orange circle is the weight vector $\mathbf{w}$ of the current SVM. The dotted lines show the hyperplanes that are induced by unlabeled samples. This visualization is inspired by Tong [Tong, 2001].

By querying the samples closest to the separating hyperplane, we try to minimize the version space by requesting as few labels as possible. However, depending on the actual shape of the version space, the SVM solution may not provide a good approximation and another query strategy would have achieved a greater reduction of the version space area. This is illustrated in Figure 3.3b. The strategy of myopically querying the samples with the smallest margin may even perform worse than a passive learner.

Therefore, we can choose a different heuristic to approximate the version space more accurately [Schohn and Cohn, 2000, Tong and Koller, 2002]. For instance, we could compute two SVMs for each sample: one for the case we labeled it positively and one assuming a negative label. We can then, for each case, compute the margins $m^+ = +\langle \mathbf{w}, \phi(x) \rangle$ and $m^- = -\langle \mathbf{w}, \phi(x) \rangle$. Finally, we query the sample which gains the maximum value for $\min(m^+, m^-)$. This

quantity will be very small if the corresponding version spaces are very different. Thus, we take the maximum to gain an equal split. This strategy is called *MaxMin Margin* [Tong, 2001] and allows us to make a better choice in case of an irregular version space area, as we can see in Figure 3.4. This, however, comes with the additional costs of computing the margin for each potential labeling.

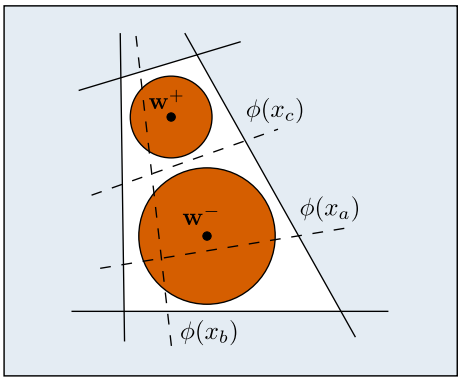



Figure 3.4: In this case, the *MaxMin Margin* strategy would query sample $x_c$. Each of the two orange circles correspond to an SVM trained with a positive and a negative labeling of $x_c$ [Tong, 2001].

Uncertainty sampling can also be motivated by trying to minimize the training error directly [Campbell et al., 2000]. Depending on the assumptions made, we can arrive at different strategies. Considering the classifier has just been trained on few labeled data, we assume the prospective labels of the yet unlabeled data to be uncorrelated with the predicted labels. Therefore, we want to select the sample for which we can expect the largest error, namely

$$\hat{x} = \underset{x \in \mathcal{U}}{\operatorname{argmax}} \frac{1}{2}\Big[\max(0, 1 - f(x)) + \max(0, 1 + f(x))\Big] \ , \tag{3.7}$$

where we assume the hinge loss and $f(x)$ as defined in (3.2). Assuming a separable dataset, we are only interested in uncertain samples, i.e., those within the margin band. Under these constraints, any choice of $x$ leads to the same value of the objective (3.7): we select the sample at random in this case. However, as soon as some labeled samples are available for SVM training, the prediction of the SVM for an unlabeled point $x$ is expected to be positively correlated with the label of $x$. Thus, we assume correct labeling and look for each sample at the minimum error we gain regardless of the labeling. We want to find the sample that maximizes this quantity, i.e.,

$$\hat{x} = \underset{x \in \mathcal{U}}{\operatorname{argmax}} \min\Big\{\max(0, 1 - f(x)), \max(0, 1 + f(x))\Big\} \tag{3.8}$$

$$= \underset{x \in \mathcal{U}}{\operatorname{argmin}} |f(x)| \ , \tag{3.9}$$

which gives us the same selection criterion as in (3.6), the *Simple Margin* strategy. If all unlabeled samples meet the target margin (i.e., the hinge loss of the samples

is zero) and if we assume the SVM labels them correctly (i.e., $|f(x)| \geq 1$), it seems that we have arrived at a hypothesis that already generalizes well. Both, picking samples near or far away from the boundary appears to be non-optimal. Therefore, we simply proceed by choosing a random sample from the unlabeled data.

In practice, we may start by training on a random subset and perform uncertainty sampling until the user stops the process or until all unlabeled samples meet the target margin. In this case, we query another random subset as a validation set and estimate the error. We may repeat the last step until we reach a satisfactory solution.

### Expected Model Change

Instead of trying to minimize an explicit or implicit version space, we can find the most informative sample by selecting it based on its expected effect on the current model, the *expected model change*. In gradient-based learning, this means selecting the sample $x \in \mathcal{U}$ that, if labeled, would maximize the expected gradient length, where we take the expectation $\mathbb{E}_y$ over all possible labels $y$.

Non-linear SVMs are usually trained by solving the underlying quadratic optimization problem in its Wolfe dual representation [Cortes and Vapnik, 1995, Schölkopf and Smola, 2002]. Let us assume SVMs without bias. If we add a new sample $(x_{\ell+1}, y_{\ell+1})$ to the current SVM solution and initialize its coefficient with $\alpha_{\ell+1} = 0$, the partial derivative with respect to $\alpha_{\ell+1}$ of the dual problem $W(\boldsymbol{\alpha})$ to be maximized is

$$g_{\ell+1} = \frac{\partial W(\boldsymbol{\alpha})}{\partial \alpha_{\ell+1}} = 1 - y_{\ell+1} \sum_{i=1}^{\ell} \alpha_i y_i \kappa(x_i, x_{\ell+1}) = 1 - y_{\ell+1} f(x_{\ell+1}) \ . \tag{3.10}$$

As $\alpha_i$ is constrained to be non-negative, we only change the model if the partial derivative is positive, that is, if $y_{\ell+1} f(x_{\ell+1}) < 1$. Note that $y_{\ell+1} f(x_{\ell+1}) > 1$ implies that $(x_{\ell+1}, y_{\ell+1})$ is already correctly classified by the current model and meets the target margin.

Let us assume that our current model classifies any sample perfectly and that the dataset is linearly separable in the feature space:

$$p(y|x) = \begin{cases} 1 & \text{if } y\, f(x) > 0 \\ 0 & \text{otherwise} \end{cases} \ . \tag{3.11}$$

If we just consider the partial derivative $g_{\ell+1}$ in the expected model change selection criterion, we arrive at selecting

$$\begin{aligned} \hat{x} &= \operatorname*{argmax}_{x \in \mathcal{U}} \Big( p(y=1|x)|1 - f(x)| + p(y=-1|x)|1 + f(x)| \Big) \\ &= \operatorname*{argmax}_{x \in \mathcal{U}} \Big( p(y=1|x)(1 - f(x)) + p(y=-1|x)(1 + f(x)) \Big) \\ &= \operatorname*{argmax}_{x \in \mathcal{U}} \begin{cases} 1 - f(x) & \text{if } f(x) > 0 \\ 1 + f(x) & \text{if } f(x) < 0 \end{cases} = \operatorname*{argmin}_{x \in \mathcal{U}} |f(x)| \ . \end{aligned} \tag{3.12}$$

Thus, uncertainty sampling can also be motivated by maximizing the expected model change. Next, we want to have a look at approaches that try to exploit the uncertainty of samples and simultaneously explore undiscovered regions of the feature space.

## 3.5 Combining Informativeness and Representativeness

By performing mere uncertainty sampling, we may pay too much attention to certain regions of the feature space and neglect other regions that are more representative of the underlying distribution. This leads to a sub-optimal classifier. To counteract this effect, we could sample close to the decision boundary, but also systematically include samples that are farer away [Guyon et al., 2011, Ho et al., 2011].

In Figure 3.5, we see an example where uncertainty sampling can mislead the classifier. Although $\phi(x_a)$ is closest to the separating hyperplane, it is also far away from all other samples in feature space and thus may not be representative of the underlying distribution. To avoid querying outliers, one would like to select samples not only based on their informativeness, but also based on representativeness [Dasgupta, 2011]. In our example, selecting sample $x_b$ would be a better choice, because it is located in a more densely populated region where a correct classification is of more importance to gain an accurate model.

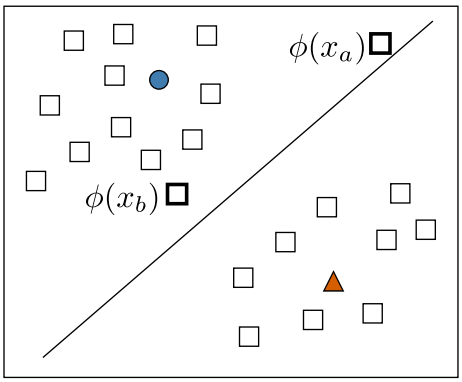
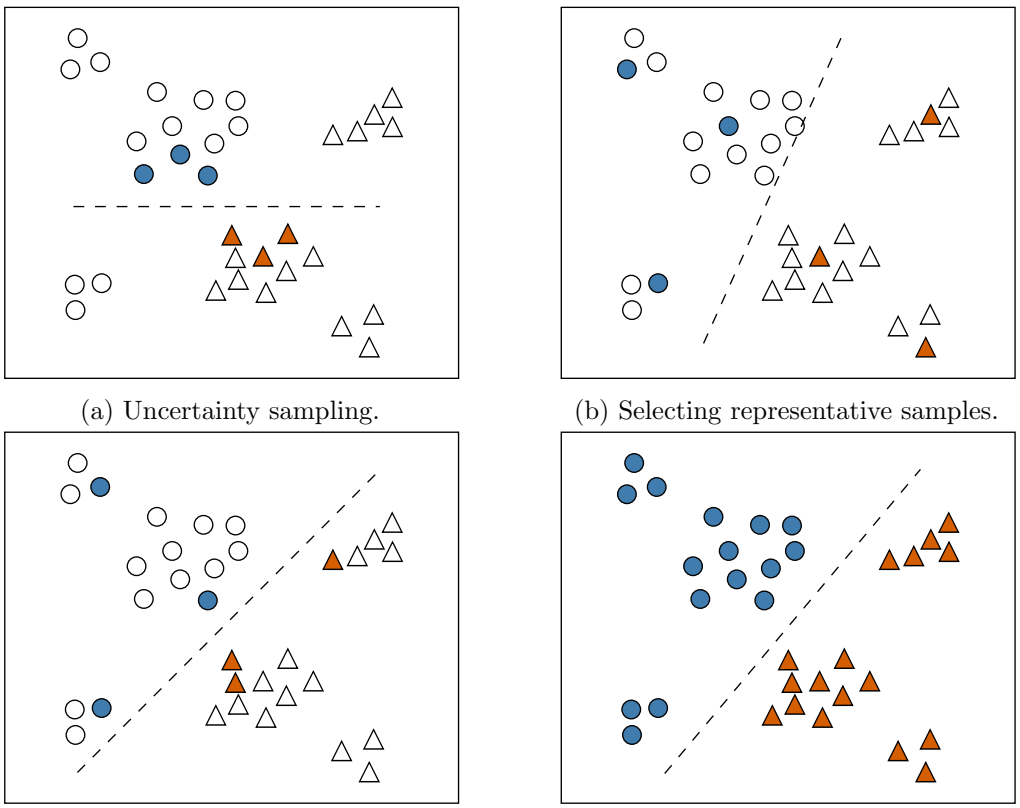



Figure 3.5: The white rectangles depict unlabeled samples. The blue circle and the orange triangle are labeled as positive and negative, respectively. In feature space, $\phi(x_a)$ lies closer to the separating hyperplane than $\phi(x_b)$, but is located in a region, which is not densely populated. Using pure uncertainty sampling, e.g., *Simple Margin*, we would query the label of sample $x_a$.

Informativeness is a measure of how much querying a sample would reduce the uncertainty of our model. As we have seen, uncertainty sampling is a viable method to exploit informativeness. Representativeness measures how well a sample represents the underlying distribution of unlabeled data [Settles, 2012]. By using a selection criterion that maximizes both measures, we try to improve our

models with less samples than a passive learner while carefully avoiding a model that is too biased. In Figure 3.6 we can see a comparison of the different strategies using a toy example where we sequentially query six samples. Figure 3.6a shows a biased classifier as the result of uncertainty sampling. While the solution in Figure 3.6b is closer to the optimal hyperplane, it also converges slower, as it additionally explores regions where we are relatively certain about the labels. Combining both strategies, as shown in Figure 3.6c, yields a decision boundary that is close to the optimal (Figure 3.6d) with fewer labels.



(a) Uncertainty sampling.

(b) Selecting representative samples.

(c) Combining informativeness and representativeness.

(d) Optimal hyperplane, obtained by training on the whole dataset.

Figure 3.6: Binary classification with active learning on six samples and passive learning on the full dataset.

### Semi-Supervised Active Learning

To avoid oversampling unrepresentative outliers, we can combine uncertainty sampling and clustering [Xu et al., 2003]. First, we train an SVM on the labeled samples and then apply $k$-means clustering to all unlabeled samples within the margin band to identify $k$ groups. Finally, we query the $k$ medoids. As only samples within the margin band are considered, they are all subject to high uncertainty. The clustering ensures that the informativeness of our selection is increased by avoiding redundant samples.

However, when using clustering, one has to decide what constitutes a cluster [Dasgupta and Hsu, 2008]. Depending on the scale and the selected number of clusters, different choices could be equally plausible. To avoid this dilemma, we can choose another strategy to incorporate density information [Huang et al., 2010]. We build on the min-max formulation [Hoi et al., 2008] of active learning and request the sample

$$\hat{x} = \underset{x_s \in \mathcal{U}}{\operatorname{argmin}} \max_{y_s \in \{-1,+1\}} \min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{(x,y)\in\mathcal{L}_s} L(f(x), x, y) \tag{3.13}$$

where $\mathcal{L}_s = \mathcal{L} \cup (x_s, y_s)$. We take the minimum regularized expected risk when including the sample $x_s \in \mathcal{U}$ with the label $y_s$ that yields the maximum error. Selecting the sample $\hat{x}$ minimizing this quantity can be approximated by uncertainty sampling (e.g., using *Simple Margin*).

In this formulation, however, we base our decision only on the labeled samples and do not take into account the distribution of unlabeled samples. Assuming we knew the labels for each sample in $\mathcal{U}$, we define the set $\mathcal{L}_{su}$ containing the labeled samples $(x, y)$ for $x \in \mathcal{U}$ and the training set $\mathcal{L}$. We select the sample

$$\hat{x} = \underset{x_s \in \mathcal{U}}{\operatorname{argmin}} \min_{\mathbf{y}_u \in \{\pm 1\}^{n_u - 1}} \max_{y_s \in \{-1,+1\}} \min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{(x,y)\in\mathcal{L}_{su}} L(f(x), x, y) \ , \tag{3.14}$$

where $n_u = |\mathcal{U}|$ and $\mathbf{y}_u$ is the label vector assigned to the samples $x \in \mathcal{U}$ without the label for $x_s$. Thus, we also maximize the representativeness of the selected sample by incorporating the possible labelings of all unlabeled samples. By using a quadratic loss-function and relaxing $\mathbf{y}_u$ to continuous values, we can approximate the solution through the minimization of a convex problem [Huang et al., 2010].

Both clustering and label estimation are of high computational complexity. A simpler algorithm, which the authors call *Hinted SVM*, considers unlabeled samples without resorting to these techniques [Li et al., 2012]. Instead, the unlabeled samples are taken as so-called hints that inform the algorithm of feature space regions which it should be less confident in. To achieve this, we try to simultaneously find a decision boundary that produces a low training error on the labeled samples while being close to the unlabeled samples, the hints. This can be viewed as semi-supervised learning [Chapelle et al., 2006]. It is, however, in contrast to typical semi-supervised SVM approaches that push the decision boundary away from the pool of unlabeled samples.

The performance of this algorithm depends on the hint selection strategy. Using all unlabeled samples might be too costly for large datasets while uniform sampling of the unlabeled pool does not consider the information provided by labeled samples. Therefore, we can start with the pool of all unlabeled samples and iteratively drop instances that are close to already labeled ones. When the ratio of hints to all samples is below a certain threshold, we can switch to uncertainty sampling.

Another problem with uncertainty sampling is that it assumes our current hypothesis to be very certain about regions far from the decision boundary. If this assumption is violated, we will end up with a classifier worse than obtained using passive learning. One way to address this issue is to measure the uncertainty of our current hypothesis and to adjust our query strategy accordingly [Mitra et al., 2004]. To achieve this, we compute a heuristic measure expressing the confidence that the current set of support vectors will not change if we train on more data. It is calculated as

$$c = \frac{2}{|\mathcal{L}_{\mathrm{SV}}| \cdot k} \sum_{(x,y) \in \mathcal{L}_{\mathrm{SV}}} \min(k_x^+, k_x^-) \ , \tag{3.15}$$

where $\mathcal{L}_{\mathrm{SV}}$ are the support vectors and $k_x^+$ and $k_x^-$ are the number of positively and negatively labeled samples within the $k$ nearest neighbors of $(x, y) \in \mathcal{L}_{\mathrm{SV}}$. In the extremes, we get $c = 1$ if $k_x^+ = k_x^-$ and $c = 0$ if $k_x^+ = 0 \vee k_x^- = 0$ for all $(x, y) \in \mathcal{L}_{\mathrm{SV}}$. We can use this measure to decide whether a labeled data point $(x, y)$ should be kept for training by adding it with probability

$$p(x) = \begin{cases} c & \text{if } yf(x) \leq 1 \\ 1 - c & \text{otherwise.} \end{cases} \tag{3.16}$$

to the training data set. This means that more samples are queried within the margin if we are very confident that the current hyperplane represents the optimal one. In the following, we discuss a related idea, which not only queries samples with a certain probability, but also subsequently incorporates this probability to weight the impact of the training set samples.

### Importance-Weighted Active Learning

When we query samples actively instead of selecting them uniformly at random, the training and test samples are not independent and identically distributed (i.i.d.). Thus, the training set will have a sample selection bias. As most classifiers rely on the i.i.d. assumption, this can severely impair the prediction performance.

Assume that we sample the training data points from the biased sample distribution $\tilde{\mathcal{D}}$ over $\mathcal{X} \times \mathcal{Y}$, while our goal is minimizing the risk (3.1) with respect to the true distribution $\mathcal{D}$. If we know the relationship between $\tilde{\mathcal{D}}$ and $\mathcal{D}$, we can still arrive at an unbiased hypothesis by re-weighting the loss for each sample [Cortes et al., 2008, Zadrozny et al., 2003]. We introduce the weighted loss $L_w(z, x, y) = w(x, y)L(z, x, y)$ and define the weighting function

$$w(x, y) = \frac{p_{\mathcal{D}}(x, y)}{p_{\tilde{\mathcal{D}}}(x, y)} \tag{3.17}$$

reflecting how likely it is to observe $(x, y)$ under $\mathcal{D}$ compared to $\tilde{\mathcal{D}}$ under the assumption that the support of $\mathcal{D}$ is included in $\tilde{\mathcal{D}}$. This leads us to the basic

result

$$\mathop{\mathbb{E}}_{(x,y)\sim\tilde{\mathcal{D}}}\Big[L_w(f(x),x,y)\Big] = \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{\tilde{\mathcal{D}}}(x,y)\frac{p_{\mathcal{D}}(x,y)}{p_{\tilde{\mathcal{D}}}(x,y)}L(f(x),y)\mathrm{d}(x,y) \quad (3.18)$$

$$= \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} p_{\mathcal{D}}(x,y)L(f(x),y)\mathrm{d}(x,y) \quad (3.19)$$

$$= \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\Big[L(f(x),x,y)\Big] \ . \quad (3.20)$$

Thus, by choosing appropriate weights we can modify our loss-function such that we can compute an unbiased estimator of the generalization error $R(f)$. This technique for addressing the sample selection bias is called *importance weighting* [Beygelzimer et al., 2009, 2010].

We define a weighted sample set $\mathcal{L}_w$ as the training set $\mathcal{L}$ augmented with non-negative weights $w_1, \ldots, w_\ell$ for each point in $\mathcal{L}$. These weights are used to set $w(x_i, y_i) = w_i$ when computing the weighted loss. For the soft margin SVM minimizing the weighted loss can easily be achieved by multiplying each regularization parameter $C_i$ in (3.4) with the corresponding weight, i.e., $C_i = w_i \cdot C$ [Zadrozny et al., 2003]. While the weighting gives an unbiased estimator, it may be difficult to estimate the weights reliably and the variance of the estimator may be very high. Controlling the variance is a crucial problem when using importance weighting.

The original importance-weighted active learning formulation works in a stream-based scenario and inspects one sample $x_t$ at each step $t > 1$ [Beygelzimer et al., 2009]. Iteration $t$ of the algorithm works as follows:

1. Receive the unlabeled sample $x_t$.
2. Choose $p_t \in [0, 1]$ based on all information available in this round.
3. With probability $p_t$, query the label $y_t$ for $x_t$, add $(x_t, y_t)$ to the weighted training set with weight $w_t = 1/p_t$, and retrain the classifier.

In step 2 the query probability $p_t$ has to be chosen based on earlier observations: this could be, for instance, the probability that two hypotheses disagree on the received sample $x_t$.

This algorithm can also be adapted to the pool-based scenario [Ganti and Gray, 2012]. In this case, we can simply define a probability distribution over all unlabeled samples in the pool. We set the probability for each point in proportion to its uncertainty, i.e., its distance to the decision boundary. This works well if we assume a noise-free setting. Otherwise, this method suffers from the same problems as other approaches that are based on a version space. Given a mislabeled sample (i.e., the label has a very low probability given the features), the active learner can be distracted and focus on regions within the hypothesis space which do not include the optimal decision boundary.

One way to circumvent these problems is to combine importance weighting with ideas from agnostic active learning [Zhao et al., 2012]. We keep an ensemble of SVMs $\mathcal{H} = \{f_1, ..., f_K\}$ and train each on a bootstrap sample subset from

$\mathcal{L}$, which may be initialized with a random subset of the unlabeled pool $\mathcal{U}$ for which labels are requested. After the initialization, we choose points $x \in \mathcal{U}$ with selection probability

$$p_t(x) = p_{\text{threshold}} + (1 - p_{\text{threshold}})(p_{\max}(x) - p_{\min}(x)) \ , \tag{3.21}$$

where $p_{\text{threshold}} > 0$ is a small minimum probability to ensure that $p_t(x) > 0$. Using Platt's method [Platt, 1999a], we define $p_i(x) \in [0, 1]$ to be the probabilistic interpretation of an SVM $f_i$ with $p_{\min} = \min_{1 \le i \le K} p_i(x)$ and $p_{\max} = \max_{1 \le i \le K} p_i(x)$. Thus, $p_t$ is high if there is a strong disagreement within the ensemble and low if all classifiers agree. This allows to deal with noise, because no hypothesis gets excluded forever.

## 3.6 Multi-Class Active Learning

The majority of research on active learning with SVMs focuses on the binary case, because dealing with more categories makes estimating the uncertainty of a sample more difficult. Furthermore, multi-class SVMs are in general more time consuming to train. There are different approaches to extend SVMs to multi-class classification. A popular way is to reduce the learning task to multiple binary problems. This is done by using either a one-vs-one [Hastie and Tibshirani, 1998, Platt et al., 2000] or a one-vs-all [Rifkin and Klautau, 2004, Vapnik, 1998] approach. However, performing uncertainty sampling with respect to each single SVM may cause the problem that one sample is informative for one binary task, but bears little information for the other tasks and, thus, for the overall multi-class classification.

It is possible to extend the version space minimization strategy to the multi-class case [Tong, 2001]. The area of the version space is proportional to the probability of classifying the training set correctly, given a hypothesis sampled at random from the version space. In the one-vs-all approach for $N$ classes, we maintain $N$ binary SVM models $f_1, \ldots, f_N$ where the $i$th SVM model is trained to separate class $i$ from the other classes. We consider minimizing the maximum product of all $N$ version space areas to select

$$\hat{x} = \operatorname*{argmin}_{x \in \mathcal{U}} \max_{y \in \{-1,1\}} \prod_{i=1}^{N} \Lambda(\mathcal{V}_{x,y}^{(i)}) \tag{3.22}$$

where $\Lambda(\mathcal{V}_{x,y}^{(i)})$ is the area of the version space of the $i$-th SVM if the sample $(x, y) : x \in \mathcal{U}$ was included in the training set. To approximate the area of the version space, we can use *MaxMin Margin* for each binary SVM. The margin of a sample in a single SVM only reflects the uncertainty with respect to the specific binary problem and not in relation to the other classification problems. Therefore, we have to modify our approximation if we want to extend the *Simple Margin* strategy to the multi-class case.

We can interpret each $f_i(x)$ as a quantity that measures to what extent $x$ splits the version space. As we have $N$ different classifiers that influence the

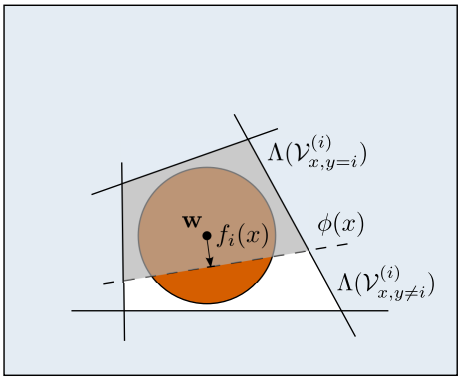



Figure 3.7: Single version space for the multi-class problem with $N$ one-vs-all SVMs [Tong, 2001]. The area $\Lambda(\mathcal{V}^{(i)}_{x,y=i})$ corresponds to the version space area if the label $y = i$ for the sample we picked. The area $\Lambda(\mathcal{V}^{(i)}_{x,y\neq i})$ corresponds to the case where $y \neq i$. In the multi-class case, we want to measure both quantities to approximate the version space area.

area of the version space, we have to quantify this influence for each one. In Figure 3.7, we can see the version space for one single binary problem where we want to discriminate between class $i$ and the rest. Thus, we want to approximate the area $\Lambda(\mathcal{V}^{(i)}_{x,y=i})$. If we choose a sample $x$ where $f_i(x) = 0$, we approximately halve the version space, for $f_i(x) = 1$, the area approximately stays the same and for $f_i(x) = -1$, we gain a zero area; similarly for the area $\Lambda(\mathcal{V}^{(i)}_{x,y\neq i})$. Therefore, we can use the approximation

$$\Lambda(\mathcal{V}^{(i)}_{x,y}) = \begin{cases} 0.5 \cdot (1 + f_i(x)) \cdot \Lambda(\mathcal{V}^{(i)}) & \text{if } y = i \\ 0.5 \cdot (1 - f_i(x)) \cdot \Lambda(\mathcal{V}^{(i)}) & \text{if } y \neq i \end{cases} . \tag{3.23}$$

We can also employ one-versus-one multi-class SVMs [Luo et al., 2004]. Again, we use Platt's algorithm [Platt, 1999a] to approximate posterior probabilities for our predictions. We simultaneously fit the probabilistic model and the SVM hyperparameters via grid-search to derive a classification probability $p_k(x)$, $k = 1, \ldots, K$, for each of the $K$ SVMs given the sample $x$. We can use these probabilities for active sample selection in different ways. A simple approach is to just select the sample with the least classification confidence. This corresponds to the selection criterion

$$\hat{x}_{\mathrm{LC}} = \operatorname*{argmin}_{x \in \mathcal{U}} \min_{k \in \{1,\ldots,K\}} p_k(x) \ . \tag{3.24}$$

This approach suffers from the same problem we mentioned earlier: the probability is connected only to each single binary problem instead of providing a measure that relates it to the other classifiers. To alleviate this, we can choose another approach, called *breaking ties* [Luo et al., 2004]. Here, we select the sample with the minimum difference between the highest class confidences, namely

$$\hat{x}_{\mathrm{BT}} = \operatorname*{argmin}_{x \in \mathcal{U}} \min_{k,l \in \{1,\ldots,K\}, k \neq l} p_k(x) - p_l(x) \ . \tag{3.25}$$

This way, we prefer samples which two classifiers claim to be certain about, and thus, avoid considering the uncertainty of one classifier in an isolated manner.

## 3.7 Efficient Active Learning

In passive learning, selecting a training set comes almost for free, we just select a random subset of the labeled data. In active learning, we have to evaluate whether a sample should be added to the training set based on its ability to improve our prediction. A simple strategy to reduce computational complexity is to not only select single samples, but to collect them in batches instead. However, to profit from this strategy, we have to make sure to create batches that minimize redundancy.

Another consideration that makes efficient computation necessary is that for many algorithms we have to retrain our model on different training subsets. Usually, these subsets differ only by one or the few samples we consider for selection. Thus, we can employ online learning to train our model incrementally. In particular, this makes sense if we analyze the effect that adding a sample has on our model.

### Online Learning

After we have selected a sample to be included in the training set, we have to retrain our model to reflect the additional data. Using the whole dataset for retraining is computationally expensive. A better option would be to incrementally improve our model with each selected sample through online learning. LASVM [Bordes et al., 2005, Glasmachers and Igel, 2008] is an SVM solver for fast online training. It is based on a decomposition method [Platt, 1999b] solving the learning problem iteratively by considering only a subset of the $\alpha$-variables in each iteration. In LASVM, this subset considers one unlabeled sample (corresponding to a new variable) in every second iteration, which can, for instance, be picked by uncertainty sampling [Bordes et al., 2005].

### Batch-Mode Active Learning

When confronted with large amounts of unlabeled data, estimating the effect of single samples with respect to the learning objective is costly. Besides online learning, we can also gain a speed-up by labeling samples in batches. A naive strategy is to just select the $n$ samples that are closest to the decision boundary [Tong and Chang, 2001]. This approach, however, does not take into account that the samples within the batch might bear a high level of redundancy.

To counteract this redundancy, we can select samples not only due to their individual informativeness, but also if they maximize the diversity within each batch [Brinker, 2003]. One heuristic is to maximize the angle between the

hyperplanes that the samples induce in version space:

$$|\cos(\angle(\phi(x_i), \phi(x_j))| = \frac{|\langle\phi(x_i), \phi(x_j)\rangle|}{\|\phi(x_i)\|\|\phi(x_j)\|} \tag{3.26}$$

$$= \frac{|\kappa(x_i, x_j)|}{\sqrt{\kappa(x_i, x_i)\kappa(x_j, x_j)}} \tag{3.27}$$

Let $\mathcal{S}$ be the batch of samples, which we initialize with one sample $x_{\mathcal{S}}$. We subsequently add the sample $\hat{x}$ whose corresponding hyperplane minimizes the maximum angle between any other hyperplane induced by a sample in the batch. It is computed as

$$\hat{x} = \operatorname*{argmin}_{x \in \mathcal{U} \setminus \mathcal{S}} \max_{z \in \mathcal{S}} |\cos(\angle(\phi(x), \phi(z))| \ . \tag{3.28}$$

We can form a convex combination of this diversity measure with the well-known uncertainty measure (distance to the hyperplane) with trade-off parameter $\lambda \in [0, 1]$. Then, samples that should be added to the batch are iteratively chosen as

$$\hat{x} = \operatorname*{argmin}_{x \in \mathcal{U} \setminus S} \left( \lambda |f(x)| + (1 - \lambda) \max_{z \in S} |\cos(\angle(\phi(x), \phi(z))| \right) \ . \tag{3.29}$$

We can choose $\lambda = 0.5$ to give equal weight to the uncertainty and diversity measure. An optimal value, however, might depend on how certain we are about the accuracy of the current classifier.

## 3.8 Conclusion

Access to unlabeled data allows us to improve predictive models in data mining applications. If it is only possible to label a limited amount of the available data due to labeling costs, we should choose this subset carefully and focus on patterns carrying the information most helpful to enhance the model. Support vector machines (SVMs) have convenient properties that make it easy to evaluate how unlabeled samples would influence the model if they were labeled and included in the training set. Therefore, SVMs are particularly well-suited for active learning. However, there are several challenges we have to address, such as efficient learning, dealing with multiple classes, and that actively choosing the training data introduces a selection bias. Importance weighting seems to be most promising to counteract this bias, and it can be easily incorporated into an active SVM learner. Devising parallel algorithms for sample selection can speed up learning in many cases. Most of the research in active SVM learning so far has focused on binary decision problems. A challenge for future research is to develop efficient active learning algorithms for multi-class SVMs that address the nature of the multi-class decision in a more principled way.

# Chapter 4

# Nearest Neighbor Density Ratio Estimation for Large-Scale Applications in Astronomy

*This chapter is based on the article J. Kremer, F. Gieseke, K. Steenstrup Pedersen, and C. Igel. Nearest neighbor density ratio estimation for large-scale applications in astronomy.* Astronomy and Computing*, 12:67–72, 2015*

**Abstract**

In astronomical applications of machine learning, the distribution of objects used for building a model is often different from the distribution of the objects the model is later applied to. This is known as sample selection bias, which is a major challenge for statistical inference as one can no longer assume that the labeled training data are representative. To address this issue, one can re-weight the labeled training patterns to match the distribution of unlabeled data that are available already in the training phase. There are many examples in practice where this strategy yielded good results, but estimating the weights reliably from a finite sample is challenging. We consider an efficient nearest neighbor density ratio estimator that can exploit large samples to increase the accuracy of the weight estimates. To solve the problem of choosing the right neighborhood size, we propose to use cross-validation on a model selection criterion that is unbiased under covariate shift. The resulting algorithm is our method of choice for density ratio estimation when the feature space dimensionality is small and sample sizes are large. The approach is simple and, because of the model selection, robust. We empirically find that it is on a par with established kernel-based methods on relatively small regression benchmark datasets. However, when applied to large-scale photometric redshift estimation, our approach outperforms the state-of-the-art.

## 4.1 Introduction

In many machine learning applications labeled (training) and unlabeled (test) data do not follow the same distribution. One reason can be that the labeled patterns have not been sampled randomly. In astronomy such a sample selection bias arises because objects that are expected to show more interesting properties are preferred when it comes to costly high-quality spectroscopic follow-up observations; other objects whose scientific value may not be that obvious (e.g., seemingly star-like objects) may be overlooked [Mortlock et al., 2011]. One way to address this bias is to weight the labeled training sample according to the ratio between the two probability distributions [Huang et al., 2007]. As this true ratio is usually not available, one has to estimate it from a finite sample. The crucial point is to control the variance of the estimator. Empirically, it seems promising to reduce the variance of the estimator by accepting a slightly higher bias [Sugiyama et al., 2008]. This gives rise to ratio estimators that, in practice, perform better than the naïve approach of estimating the two densities separately.

In this work, we improve a simple nearest neighbor density ratio estimator [Lima et al., 2008] by combining it with a principled way of performing model selection [Sugiyama and Müller, 2005]. The approach compares well to established kernel-based estimators on a variety of standard, small-sized regression datasets. Furthermore, by selecting proper hyperparameters and by taking huge amounts of patterns into account, we experimentally show that the estimator yields better results compared to the state-of-the-art on a large-scale astronomical dataset.

Let each data point be represented by a feature vector $\mathbf{x}$ from a domain $\mathcal{X}$ with a corresponding label $\mathbf{y}$ from a domain $\mathcal{Y}$. We consider scenarios in which the learner has access to some labeled (source) data $S$ sampled from $p_s(\mathbf{x}, \mathbf{y})$ and a large sample of unlabeled (target) data $T$ sampled from $p_t(\mathbf{x}, \mathbf{y})$. While $p_s(\mathbf{x}, \mathbf{y})$ and $p_t(\mathbf{x}, \mathbf{y})$ may not coincide, we assume that $p_s(\mathbf{y}|\mathbf{x}) = p_t(\mathbf{y}|\mathbf{x})$ for all $\mathbf{x}$ and that the support of $p_t$ is a subset of the support of $p_s$. This is usually referred to as *covariate shift*, a particular type of *sample selection bias*. In this case the probability density ratio between target and source distribution at a given point reduces to $\beta(\mathbf{x}) = \frac{p_t(\mathbf{x})}{p_s(\mathbf{x})}$.

Different strategies have been proposed to address covariate shift, such as finding a common feature space or re-weighting the source patterns. The latter is conceptually simple, and there are several approaches to estimate appropriate weights via density ratio estimation [Huang et al., 2007, Lima et al., 2008, Sugiyama and Müller, 2005, Bickel et al., 2007, Cortes et al., 2008, Loog, 2012, Quionero-Candela et al., 2009, Izbicki et al., 2014, Kanamori et al., 2009]. These methods are, for example, based on reducing the problem to probabilistic classification between the target and source dataset [Bickel et al., 2007], on using kernel-based methods to match means in an induced Hilbert space [Huang et al., 2007], or on using nearest neighbor queries to estimate the mismatch between the densities by counting patterns in local regions [Lima et al., 2008, Loog, 2012]. It is crucial to control the variance of such an estimator via regularization. Depending on the algorithm at hand, the regularization can take the form of, for example, a kernel bandwidth [Huang et al., 2007], the rank of a low-rank kernel

matrix approximation [Izbicki et al., 2014], or a weight norm [Kanamori et al., 2009]. The involved parameters are often set by heuristics such as the median of pairwise distances for the kernel bandwidth [Schölkopf and Smola, 2002]. As an alternative, Sugiyama et al. [Sugiyama and Müller, 2005] suggest a model selection criterion that is unbiased under covariate shift. In the following, we employ this criterion for selecting the neighborhood size of the nearest neighbor estimator via cross-validation. Then, we empirically show that the resulting algorithm can outperform the computationally more expensive state-of-the-art kernel-based estimator due to its ability to consider larger samples in less time.

This article is structured as follows: in Section 4.2 we briefly discuss two state-of-the-art kernel-based estimators that serve as a baseline in our experimental evaluation. In Section 4.3 we present a nearest neighbor-based density ratio estimator and show how it can be extended to perform automatic model selection. In Section 4.4 we evaluate the proposed nearest neighbor density ratio estimator with integrated model selection in comparison to other methods on a medium-sized regression benchmark and on a large-scale astronomical dataset for photometric redshift estimation. In Section 4.5 we conclude and give possible directions for future work.

## 4.2 Kernel-based Density Ratio Estimation

In density ratio estimation, kernel-based estimators are considered the state-of-the-art [Sugiyama et al., 2010a]. Among these, kernel mean matching (KMM) [Huang et al., 2007] and the spectral series estimator [Izbicki et al., 2014] have shown to perform particularly well.

Given some input space $\mathcal{X}$, a kernel is a positive semi-definite function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for which $\forall x, z \in \mathcal{X} : k(x, z) = \langle \Phi(x), \Phi(z) \rangle_{\mathcal{H}}$, where $\Phi : \mathcal{X} \to \mathcal{H}$ maps elements of the input space to a kernel-induced Hilbert space $\mathcal{H}$ [Aronszajn, 1950]. Kernel mean matching aims at matching the means of two distributions in $\mathcal{H}$ by solving the problem

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \ \left\| \frac{1}{N_s} \sum_{i=1}^{N_s} \beta_i \Phi(\mathbf{x}_i^{(s)}) - \frac{1}{N_t} \sum_{i=1}^{N_t} \Phi(\mathbf{x}_i^{(t)}) \right\|_{\mathcal{H}}^2 \tag{4.1}$$

$$\text{subject to } \ \beta_i \in [0, B] \ \text{ and } \ |\sum_{i=1}^{N_s} \beta_i - N_s| \leq N_s \epsilon \ , \tag{4.2}$$

where $N_s$ is the number of source domain patterns and $N_t$ is the number of target domain patterns. The parameter $B$ restricts the maximum possible weight and $\epsilon$ bounds the deviation of the mean weight from 1. Cortes et al. [Cortes et al., 2008] show that the solution to Eq. (4.1) converges with high probability to the true density ratio if the kernel induced by $\Phi(\mathbf{x})$ is universal [Steinwart and Christmann, 2008]. The kernel function, which implicitly defines $\Phi$ and $\mathcal{H}$, is typically chosen from a parameterized family of functions, and the kernel parameters are parameters of KMM-based approaches.

The spectral series estimator [Izbicki et al., 2014], although motivated differently, minimizes an unconstrained version of Eq. (4.1) for computing training

weights. Instead of bounding the weights via $B$ and their mean via $\epsilon$, the solution is regularized by the rank $J$ of a low-rank approximation of the kernel Gram matrix between training points – which results when expanding Eq. (4.1). Unlike KMM, the spectral series estimator can compute weights not only for the source sample, but also for arbitrary patterns. This allows for selecting the kernel parameters and $J$ via cross-validation, as we shall see later.

Negative theoretical results in the analysis of weighting methods [Ben-David et al., 2010, Ben-David and Urner, 2012] suggest that sample sizes have to be prohibitively large to guarantee reliable weights. However, empirically it has been found that re-weighting often does improve results. Our method is motivated by typical tasks in astronomy, where we deal with large labeled samples and huge unlabeled samples in feature spaces of relatively low dimensionality (e.g., up to $\mathbb{R}^{10}$). For such rather benign scenarios, we aim at estimating weights with high accuracy by taking into account hundreds of thousands of labeled and unlabeled patterns. However, both KMM as well as the spectral series estimator involve $|S| \times |T|$ kernel matrices in their general form. Thus, they are not directly applicable to scenarios with hundreds of thousands of patterns. Special cases might be addressed in a more efficient way. Still, the general cases with non-linear kernel functions involve the computation of such kernel matrices and, depending on the method, quadratic programming, matrix inversion, or eigenvalue decomposition, which exhibit at least a quadratic running time [Bern and Eppstein, 2001, Golub and Van Loan, 1989, Kojima et al., 1989]. Therefore, we are considering nearest neighbor-based density ratio estimation, which can be implemented more efficiently.

For the matrix decompositions in the spectral series estimator we used an efficient $\mathcal{O}(n^2)$-algorithm [Dhillon, 1998]. Both, decomposition as well as the nearest neighbor search, could be sped up by using approximation schemes (e.g., see [Arya et al., 1994, Halko et al., 2011]), but we decided not to introduce such approximations with corresponding hyperparameters in our study.

## 4.3 Nearest Neighbor Density Ratio Estimation Revisited

We consider the algorithm proposed by Lima et al. [Lima et al., 2008] to estimate appropriate ratios via nearest neighbor queries, see Algorithm 1. The efficiency of the approach is ensured via the use of $k$-d trees. For the sake of completeness, we briefly sketch how these spatial data structures can be used to speed up nearest neighbor search before outlining the details of the density ratio estimator.

### Nearest Neighbor Search in Low Dimensions

A classical *$k$-$d$ tree* [Bentley, 1975] is a binary tree constructed from a $d$-dimensional point set $S \subset \mathbb{R}^d$. The inner nodes correspond to hyperplanes splitting the data in $\mathbb{R}^d$ and the leaf nodes define a partitioning of $S$. The tree can be built recursively in $\mathcal{O}(|S| \log |S|)$ time. Starting from the root node numbered by 0 and $S_0 = S$, each inner node $v$ with children $u$ and $w$ partitions

the data $S_v$ into two almost equal-sized subsets $S_u$ and $S_w$. If $S_v$ contains only a single point (or a predefined number of points), $v$ becomes a leaf node. At tree level $j$, the datasets are split according to the median in dimension $j \bmod d + 1$.

To efficiently search for the nearest neighbor of a given query point $\mathbf{q} \in \mathbb{R}^d$, one can make use of the hierarchical subdivision induced by a $k$-d tree: The tree is traversed in two phases. During the first phase, the tree is processed from top to bottom to find the $d$-dimensional leaf (box) that contains the query point (the search is guided by the median values). The query point is then compared with all points that are stored in the corresponding leaf, which yields the first nearest neighbor candidate. Afterwards, in the second phase, the tree is processed from bottom to top and on the way back to the root, neighboring boxes are checked for points that are potentially closer to $\mathbf{q}$ than the current candidate. In case the distance of $\mathbf{q}$ to the splitting hyperplane is larger than the distance between $\mathbf{q}$ and its current nearest neighbor candidate, one can safely ignore the whole subtree that has not yet been visited. These distance checks can be performed efficiently by resorting to the associated median values. The generalization to $k > 1$ neighbors is straightforward (see, e.g., [Bentley, 1975], or [Gieseke et al., 2014], for details).

In the best case, all nearest neighbors are contained in the leaf that stems from the first phase and no further subtrees need to be processed on the way back to the root. For such queries, the runtime is logarithmic in the number $|S|$ of points. This also holds for the expected case as shown by [Friedman et al., 1977] (given constant $d$). In the worst case, however, the complete $k$-d tree needs to be processed, which leads to a linear instead of a logarithmic runtime per query. From a practical perspective, the running time depends on the dimensionality of the feature space: for moderate dimensions (e.g., up to $d = 30$), a logarithmic running time behavior can be expected, while for larger $d$ the performance often decreases significantly due to the *curse of dimensionality* [Hastie et al., 2009].

### Nearest Neighbor Density Ratio Estimator

We are now ready to outline the details of Algorithm 1: In Step 1, $k$-d trees for the source and target patterns are built. In Steps 2 to 8, all query patterns $\mathbf{x}_j^{(q)}$ are processed. For each query pattern, the $K$ nearest neighbors w.r.t. the source patterns in $S$ are computed. This is followed by the computation of the number $l_j$ of nearest neighbors in $T$ whose distance to $\mathbf{x}_j^{(q)}$ is less than or equal to the previously computed $K$-th nearest neighbor. Finally, this result is re-weighted according to the number $N_s$ of source patterns, the number $N_t$ of target patterns and the number $K$ of nearest neighbors. Hence, the true density ratio $\beta(\mathbf{x}_j^{(q)})$ of target and source distribution at a point $\mathbf{x}_j^{(q)}$ in the feature space $\mathbb{R}^d$ is approximated via

$$\widehat{\beta}(\mathbf{x}_j^{(q)}) = l_j \cdot \frac{N_s}{K \cdot N_t} \ . \tag{4.3}$$

As $k$-d trees speed up nearest neighbor computation for low-dimensional feature spaces, we get good running time results in this scenario: the construction of the trees for the source and target patterns takes $\mathcal{O}(N_s \log N_s)$ and $\mathcal{O}(N_t \log N_t)$

**Algorithm 1** NEAREASTNEIGHBORRATIOESTIMATOR

**Require:** A set $S = \{\mathbf{x}_1^{(s)}, \ldots, \mathbf{x}_{N_s}^{(s)}\} \subset \mathbb{R}^d$ and a set $T = \{\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_{N_t}^{(t)}\} \subset \mathbb{R}^d$ of patterns from the source and target domain, respectively, a query set $Q = \{\mathbf{x}_1^{(q)}, \ldots, \mathbf{x}_{N_q}^{(q)}\} \subset \mathbb{R}^d$, and a number $K > 1$.

**Ensure:** Weights $\widehat{\beta}(\mathbf{x}_1^{(q)}), \ldots, \widehat{\beta}(\mathbf{x}_{N_q}^{(q)}) \in \mathbb{R}$ for the patterns in $Q$.

1: Construct $k$-d trees $\mathcal{T}_s$ and $\mathcal{T}_t$ for $S$ and $T$, respectively.
2: $W = \{\}$
3: **for** $j = 1, \ldots, N_q$ **do**
4: Compute the $K$ nearest neighbors for $\mathbf{x}_j^{(q)}$ in $S$ (via $k$-d tree $\mathcal{T}_s$).
5: Let $r_j$ be the Euclidean distance between $\mathbf{x}_j^{(q)}$ and its $K$-th nearest neighbor.
6: Compute number $l_j$ of nearest neighbors in $T$ with distance less than $r_j$ to $\mathbf{x}_j^{(q)}$ (via $k$-d tree $\mathcal{T}_t$).
7: $W = W \cup \left\{ l_j \cdot \frac{N_s}{K \cdot N_t} \right\}$
8: **end for**
9: **return** $W$

time, respectively. For each query pattern, nearest neighbors are computed via these trees. The number $K$ of neighbors is crucial for the accuracy of the algorithm, and the question of how to choose it is not discussed in [Lima et al., 2008]. We propose to select $K$ via cross-validation by minimizing the model selection criterion proposed in [Sugiyama and Müller, 2005]. It seeks to minimize the least-squares error between true and estimated density ratio, as in regression. However, we usually do not have access to the true density ratio. Therefore, we use a substitution to estimate the minimizer of the least-squares error up to a constant. The expected least-squares loss between true and estimated density ratio over the source probability density $p_s(\mathbf{x})$ is given by

$$\begin{aligned} L(\beta, \widehat{\beta}) &= \int (\beta(\mathbf{x}) - \widehat{\beta}(\mathbf{x}))^2 \, p_s(\mathbf{x}) \mathrm{d}\mathbf{x} \\ &= \int \widehat{\beta}(\mathbf{x})^2 p_s(\mathbf{x}) \mathrm{d}\mathbf{x} - 2 \int \widehat{\beta}(\mathbf{x}) \beta(\mathbf{x}) p_s(\mathbf{x}) \mathrm{d}\mathbf{x} + \int \beta(\mathbf{x})^2 p_s(\mathbf{x}) \mathrm{d}\mathbf{x} \\ &= \int \widehat{\beta}(\mathbf{x})^2 p_s(\mathbf{x}) \mathrm{d}\mathbf{x} - 2 \int \widehat{\beta}(\mathbf{x}) p_t(\mathbf{x}) \mathrm{d}\mathbf{x} + \int \beta(\mathbf{x})^2 p_s(\mathbf{x}) \mathrm{d}\mathbf{x} \ , \end{aligned} \tag{4.4}$$

where we substituted the true density ratio $\beta(\mathbf{x}) = \frac{p_t(\mathbf{x})}{p_s(\mathbf{x})}$. As the third term does not depend on the estimated ratio $\widehat{\beta}(\mathbf{x})$, we can estimate $L(\beta, \widehat{\beta})$ up to a constant by

$$\widehat{L}(\beta, \widehat{\beta}) = \frac{1}{|S|} \sum_{\mathbf{x} \in S} \widehat{\beta}(\mathbf{x})^2 - \frac{2}{|T|} \sum_{\mathbf{x} \in T} \widehat{\beta}(\mathbf{x}) \ . \tag{4.5}$$

Here, we have replaced the expectations in the first two terms by their empirical estimates. As long as $p_s(\mathbf{x})$ and $p_t(\mathbf{x})$ do not change, the constant term in Eq. (4.5) will not change and thus, we can safely ignore it when comparing different weight estimates $\widehat{\beta}$.

It is important to note that we evaluate Eq. (4.5) *post hoc* on trained density ratio estimators. Direct unconstrained minimization of Eq. (4.5) with respect

to $\widehat{\beta}(\mathbf{x})$ for $\mathbf{x} \in S$ (i.e., the values needed for re-weighted training) would lead to the trivial solution $\widehat{\beta}(\mathbf{x}) = 0$ for $\mathbf{x} \in S$. An open-source Python package of the nearest neighbor estimator with integrated model selection is available on Github.[1]

## 4.4 Experiments

We consider two experiments: re-weighted regression on standard domain adaptation benchmarks and weight computation for photometric redshift estimation.

### Regression Benchmarks

We compared our approach to kernel mean matching (KMM) [Huang et al., 2007] and the spectral series estimator [Izbicki et al., 2014] following the protocol of the experiments in [Cortes et al., 2008]. For each of the eight regression datasets, which are rather small (the largest having 16 512 labeled and 9511 unlabeled patterns), we created a biased subset $S$ of the original dataset $T$. As defined in [Cortes et al., 2008], each point is moved from $T$ to $S$ with probability

$$p(s = 1|\mathbf{x}) = \frac{e^v}{1 + e^v} \ , \tag{4.6}$$

where $v$ is defined as

$$v = \frac{4\mathbf{w} \cdot (\mathbf{x} - \overline{\mathbf{x}})}{\sqrt{\mathrm{Var}(\mathbf{w} \cdot (\mathbf{x} - \overline{\mathbf{x}}))}} \ , \tag{4.7}$$

for a pattern $\mathbf{x} \in \mathbb{R}^d$, and $\mathbf{w} \in \mathbb{R}^d$ chosen uniformly at random from $[-1, 1]^d$. Thus, the bias is only determined by the covariate $\mathbf{x}$. The ideal method, which we consider as a baseline, weights the points in $S$ with $\frac{1}{p(s=1|\mathbf{x})}$. For each dataset, we selected the $\mathbf{w}$ that maximized the difference in regression loss between ideal and unweighted method among 10 trials.

After having estimated the weights, we use them to re-weight the loss function of a linear regularized least-squares estimator. Here, we select the regularization parameter $\lambda \in \{2^n : n \in \{-3, \ldots, 4\}\}$ via leave-one-out cross-validation. Since KMM has no mechanism for automatically choosing its hyperparameter, we chose the bandwidth $\sigma = \sqrt{d/2}$ for $\mathbf{x} \in \mathbb{R}^d$ [Cortes et al., 2008]. For the spectral series estimator and the nearest neighbor method, we chose their parameters by performing 5-fold cross-validation using Eq. (4.5). We selected the bandwidth parameter $\epsilon$ of the spectral series estimator from $\{\epsilon_0^{-1}, \epsilon_0^0, \epsilon_0^1, \epsilon_0^2\}$, with $\epsilon_0 = \mathrm{median}(\{\|\mathbf{x} - \mathbf{y}\|_2^2 : \mathbf{x}, \mathbf{y} \in S\})/8$ [Schölkopf and Smola, 2002], and the rank $J$ from $\{1, \ldots, \lfloor N_s \times 4/5 \rfloor\}$. For the nearest neighbor estimator we selected a $K \in \{2, 3, 4, 5, 8, 16, 32\}$ and also considered a variant with fixed $K = 2$ to demonstrate the influence of model selection (the value $K = 2$ corresponds to the most frequent choice for $K$ in the model selection algorithm).

Figure 4.1 shows the relative normalized mean squared error (NMSE) and the standard deviation over 10 different trials for each method on a test set not

[1] `https://github.com/kremerj/nnratio`.

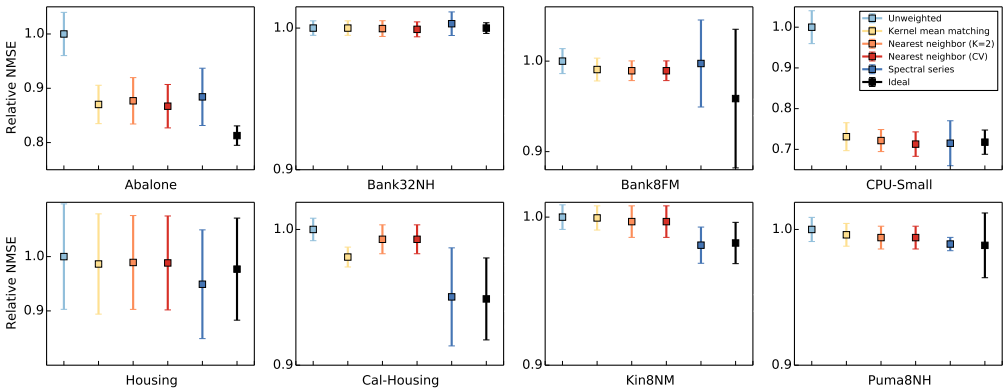



Figure 4.1: The relative normalized mean squared errors (NMSE) for kernel- and nearest neighbor-based ratio estimators on different regression datasets. The error bars indicate the standard deviations over 10 different samplings using the selection probability $p(s = 1|\mathbf{x})$ for a fixed $\mathbf{w}$.

used for training or estimation of weights. For each dataset we scaled the results linearly so that the unweighted NMSE yielded 1.0. The weighted methods almost always perform better than unweighted regression. Although the sample sizes are small, the nearest neighbor estimator performs on a par with the kernel methods among which the spectral series estimator performs best. Because of the sample sizes, our method cannot tap its full potential and using a fixed $K = 2$ seems to be a viable approach. However, the picture changes when moving to our real-world large-scale application.

### Redshift Estimation

We evaluated our method on a large-scale astronomical dataset [Izbicki et al., 2014]. The problem we consider is photometric redshift estimation of galaxies. The redshift phenomenon is caused by the Doppler effect which shifts the spectrum of an object towards longer wavelengths if it is moving away from the observer. Because the universe is expanding uniformly, we can infer a galaxy's velocity by its redshift and, thus, its distance to Earth. Hence, redshift estimation is a useful tool for determining the geometry of the universe. A photometric observation contains the intensities of an object (in our case, galaxies) in 5 different bands ($u$,$g$,$r$,$i$,$z$), ranging from ultraviolet to infrared. Spectroscopy, in contrast, measures the photon count at certain wavelengths. The resulting spectrum allows for identifying the chemical components of the observed object and thus, enables determining many interesting properties, including the redshift. Spectroscopy, however, is much more time-consuming than photometric observation and therefore, costs could be greatly reduced if we could predict suitable candidates for follow-up spectroscopy from low-quality low-cost photometry. Figure 4.2 shows examples of corresponding photometric and spectroscopic observations.

For each of the 5 bands a point spread function (*model*) and a composite model (*cmodel*) are fit to the photometric observation. We take the 4 magnitude

differences between adjacent bands and the magnitude in the red band for *model* and *cmodel*. Thus, we arrive at $2 \times (4 + 1) = 10$ covariates for each galaxy.

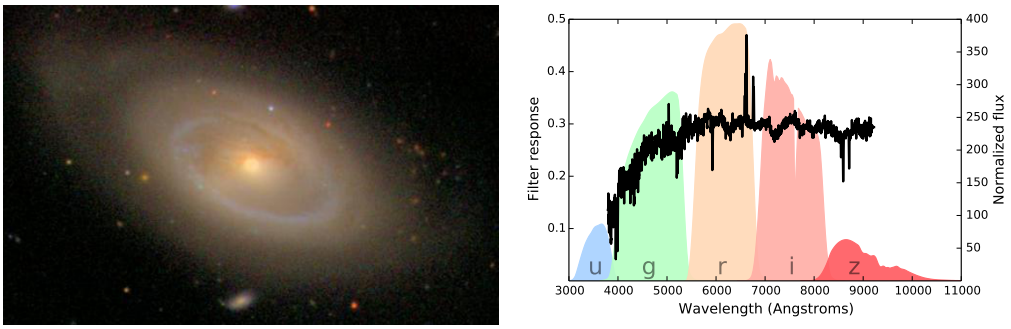



Figure 4.2: An example from the *Sloan Digital Sky Survey* (SDSS) [Aihara et al., 2011]. (a) An image of the spiral galaxy NGC 5750. (b) Its associated spectrum overlapping the five photometric intensity band filters *u,g,r,i,z*.

The dataset contains a sample of 467 710 galaxies whose redshift has been confirmed by spectroscopy and an unconfirmed sample of 540 237 galaxies. The task is to estimate the redshift of the unconfirmed (target) sample by training on the spectroscopically confirmed (source) sample. As we do not have ground-truth labels for the target sample, we simply recorded the estimated loss given by Eq. (4.5) as in [Izbicki et al., 2014], see Figure 4.3. Interestingly, the absolute estimates are more accurate when we consider the dataset as-is. In Figure 4.3(b) we consider a preprocessed dataset where we standardized the covariates to have zero mean and unit variance, as is common for methods that rely on pattern distances. Here, we see that the nearest neighbor estimator with model selection outperforms the other methods even clearer, although the absolute estimated loss is higher than the one for the original data, see Figure 4.3(a). If the task can benefit from re-weighting, then the performance is likely to improve with more accurate weights.

In our experiment, we trained the ratio estimators with increasing sample sizes from 5000 to 400 000 patterns (each from source and target sample) and estimated the weights on hold-out test samples of size 50 000 (source and target) not used for training. As KMM cannot produce out-of-sample weights, we only compared the nearest neighbor estimator (with $K$ either being fixed or chosen by model selection) and the spectral series estimator using the same parameters as in the first experiment. As running times become prohibitively large for the spectral series estimator, we only recorded it up to sample sizes of 20 000 patterns. Figure 4.4 shows the running time per sample size on an AMD Opteron 6380. The time measured includes the time used for cross-validation on a single-core machine. It should be noted that the cross-validation procedure is parallelizable to the point that its additional costs for the gained accuracy are minimal. For comparable running times the nearest neighbor estimator is able to use more samples than the spectral series estimator and thus, estimate weights more accurately. Furthermore, selecting the parameter $K$ via cross-validation performs better than our default choice $K = 2$ (which was the most frequently selected

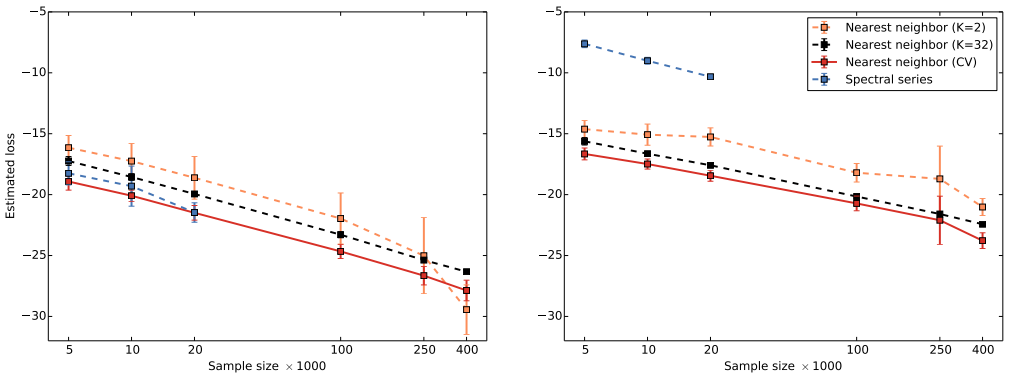



Figure 4.3: The estimated loss for the nearest neighbor and spectral series estimator on an astronomical dataset typically used in the context of photometric redshift estimation. (a) The original dataset. (b) Results with covariates transformed to have zero mean and unit variance.

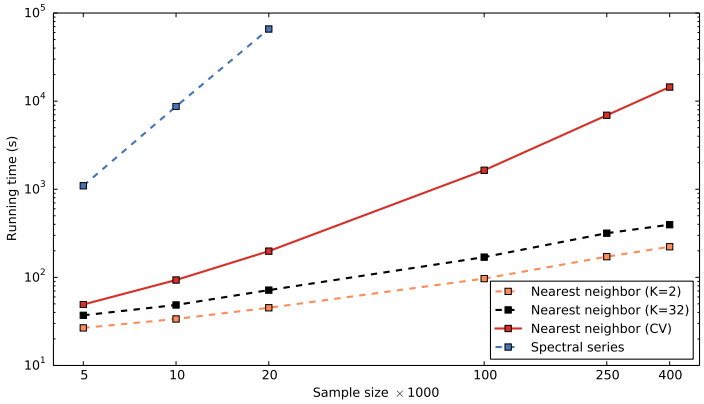



Figure 4.4: The running times per sample size for the different estimators, including the time used for cross-validation. The nearest neighbor estimators can utilize considerably larger samples than the spectral series estimator given the same time constraints.

value in the model selection experiments on the benchmark datasets).

## 4.5 Conclusion

Sample selection bias is a common problem in astronomy [Richards et al., 2012], where datasets are typically large and the feature space dimensionality is often low. For this scenario, we suggest to use a nearest neighbor density ratio estimator combined with a model selection criterion, which is unbiased under covariate shift, for choosing the neighborhood size. The resulting algorithm is simple, robust due to the systematic hyperparameter choice, and—as we experimentally demonstrate—highly efficient and accurate. Future work will consider the theoretical properties of the estimator and an implementation

on GPUs [Gieseke et al., 2014] for handling datasets with billions of patterns efficiently and at low cost.

## Acknowledgments

The authors gratefully acknowledge support from The Danish Council for Independent Research through the project *SkyML* (FNU 12-125149). The authors also would like to thank R. Izbicki for providing the astronomical dataset. Funding for SDSS-III has been provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Science Foundation, and the U.S. Department of Energy Office of Science. The SDSS-III web site is http://www.sdss3.org/.

SDSS-III is managed by the Astrophysical Research Consortium for the Participating Institutions of the SDSS-III Collaboration including the University of Arizona, the Brazilian Participation Group, Brookhaven National Laboratory, Carnegie Mellon University, University of Florida, the French Participation Group, the German Participation Group, Harvard University, the Instituto de Astrofisica de Canarias, the Michigan State/Notre Dame/JINA Participation Group, Johns Hopkins University, Lawrence Berkeley National Laboratory, Max Planck Institute for Astrophysics, Max Planck Institute for Extraterrestrial Physics, New Mexico State University, New York University, Ohio State University, Pennsylvania State University, University of Portsmouth, Princeton University, the Spanish Participation Group, University of Tokyo, University of Utah, Vanderbilt University, University of Virginia, University of Washington, and Yale University.

# Chapter 5

# Active Label Correction for Class-Conditional Noise

*This chapter is based on the article J. Kremer, F. Sha, and C. Igel. Active label correction for class-conditional noise. Submitted, 2016a*

**Abstract**

Active label correction addresses the problem of learning from input data for which noisy labels are available (e.g., from imprecise measurement devices or crowd-sourcing) and each true label can be obtained at a significant cost (e.g., by taking additional measurements or asking human experts). To minimize the labeling costs, we are interested in identifying training patterns for which knowing the true labels maximally improves the learning performance. This paper devises active label correction algorithms for classification under the assumption of class-conditional noise, where the true label is conditionally independent of the input given the observed label. To select labels for correction, we adopt the active learning strategy of maximizing the expected model change. We consider the change in regularized empirical risk functionals that use different pointwise loss functions for patterns with noisy and true labels, respectively. Three different choices of loss functions for the noisy data points then lead to different active label correction algorithms. Two of the loss functions consider the label noise rates, which are estimated during learning, where importance weighting compensates for the sampling bias due to active learning. Experiments show that viewing the true label as a latent variable and computing the maximum likelihood estimate of the model parameters improves over the state-of-the-art.

## 5.1 Introduction

Acquiring data with noisy labels for supervised learning is often cheap and simple, while obtaining reliable labels remains difficult and/or costly. For instance, in astronomy huge amounts of photometric data from sky surveys are available. Noisy labels can be obtained using crowd-sourcing or automated labeling, but getting

a reliable label may require an expert or even additional costly spectroscopic measurements. Another example are medical images, which can be labeled either unreliably by medical students or by expensive experts [Urner et al., 2012]. If we are willing to invest in getting high quality labels for some of our training data in order to increase the generalization performance of a machine learning model, two fundamental questions arise. First, how should we learn from both noisy and true labels? Second, which training examples should be re-labeled? This study addresses these questions by devising tailored loss functions and corresponding *active label correction* strategies, which try to identify examples for which obtaining the true labels would be most helpful. Active label correction has been considered before by Rebbapragada et al. [2012] and as *learning from weak teachers* by Urner et al. [2012]. We incorporate a label noise model, based on which we can derive algorithms for learning and re-labeling in a principled way. This noise model must be simple so that its parameters can be estimated efficiently during training. We assume that the label noise is class-conditional [Angluin and Laird, 1988], that is, the label noise rates depend only on the true class, but are independent of the covariate. For selecting the examples for correction, we adopt the strategy from active learning to select those points that promise to change the model the most [Settles and Craven, 2008, Settles et al., 2008].

In the following, we summarize additional related work and our main contributions. Section 2 introduces the general active label correction framework, three different pointwise loss functions for noisy examples, and the resulting general label correction algorithms. Section 3 makes these algorithms concrete for logistic regression. Section 4 presents an importance-weighting method for estimating the noise model during learning. We present experimental results on a range of datasets using logistic regression and on a convolutional neural network example in section 5.

## Related Work

Label noise can degrade the accuracy of a learning algorithm to a great extent, and there are different ways of dealing with this problem, see the survey by Frenay and Verleysen [2014]. One way is to incorporate a label noise model into the loss function [Bootkrajang and Kabán, 2012, Natarajan et al., 2013, Reed et al., 2015, Sukhbaatar and Fergus, 2015, Xiao et al., 2015]. We follow this approach to guide label corrections and to mitigate the effects of label noise on not yet corrected training examples. The work most similar to ours is by Rebbapragada et al. [2012], although they do not model the label noise. They use uncertainty sampling to select the next example to correct and show that this improves over random selection. The purely theoretical paper by Urner et al. [2012] makes assumptions on the noisy labeling which are very different from our noise model and which are based on a notion of neighborhood in the input space. Roughly speaking, if the labels are homogeneous in a neighborhood, then the noise rate in that neighborhood has to be low, if the labels are heterogeneous, the noise rate has to be high. Label correction has also been considered in the

crowd-sourcing community [Sheng et al., 2008, Zhao et al., 2011]. In these works the authors examine the problem of noisy labelers and consider the trade-off between sampling new examples and asking for additional labels for an already sampled example. In our case we assume that labels are corrected in a reliable way, that the dataset is fixed, and that the noise is inherent (i.e., there is not necessarily a distribution of labels from different labelers).

### Main Contributions

(i) We introduce noise-aware loss functions for active label correction. These loss functions attenuate the influence of noisy examples and inform the selection for re-labeling. (ii) We adopt the maximum expected model change strategy for the proposed regularized risk functionals and devise three novel algorithms for active label correction. (iii) It is shown how to simultaneously learn noise and classification model parameters using importance-weighting. (vi) We provide an empirical comparison demonstrating that *robust maximum likelihood weighted uncertainty re-labeling (ML-WURL)*, which is based on maximum likelihood estimates of the model parameters, outperforms the state-of-the-art.

## 5.2 Active Label Correction by Maximizing Expected Model Change

We consider classification problems with input space $\mathcal{X}$, label space $\mathcal{Y}$, and pointwise loss function $l$. The goal is to minimize the risk $R(h) = \mathbb{E}_{(x,y)\sim p}[l(h(x), y)]$ over hypotheses $h \in \mathcal{H}$ for an unknown distribution $p$. For a training pattern $(x, y) \sim p$, we may initially only know $(x, \tilde{y})$, the input $x$ and a corresponding *noisy label* $\tilde{y} \in \mathcal{Y}$. However, we can obtain the *true label* $y$ at a considerable cost. Although we refer to $y$ as *the* true label, we do not presume zero Bayes risk.[1]

We assume *class-conditional label noise* [Angluin and Laird, 1988]. That is, the noisy label is conditionally independent of the input given the true label, $p(\tilde{y} \,|\, x, y) = p(\tilde{y} \,|\, y)$. This noise model is well-studied in the label noise literature [Bootkrajang and Kabán, 2012, Natarajan et al., 2013, Menon et al., 2015, Liu and Tao, 2016]. The model has the advantage that learning its parameters typically requires only a few observations of noisy labels with corresponding true labels. In the following, we focus on binary classification and define the noise rates as

$$\rho_{+1} := p(\tilde{Y} = -1 | Y = +1), \;\; \rho_{-1} := p(\tilde{Y} = +1 | Y = -1), \;\; \rho_{+1} + \rho_{-1} < 1 \;\; ,$$

where $Y, \tilde{Y} \in \mathcal{Y} = \{-1, +1\}$ are random variables for the true and the noisy label, respectively.

Our learning strategy is minimizing the regularized empirical risk

$$L(h) := \sum_{(x,y)\in\mathcal{S}} l(h(x), y) + \sum_{(x,\tilde{y})\in\tilde{\mathcal{S}}} \tilde{l}(h(x), \tilde{y}) + \lambda\Omega(h) \;\; , \tag{5.1}$$

[1] For all subsequent considerations it is actually not necessary to be able to obtain the true label, a significantly more accurate label is sufficient.

given a *noisy training set* $\tilde{\mathcal{S}}$ containing inputs with noisy labels $(x, \tilde{y})$ and a *clean training set* $\mathcal{S}$ containing $(x, y) \sim p$. This study will investigate different choices for the noise-aware pointwise loss function $\tilde{l}$, which may depend on (estimates of) the noise rates.

We assume that we have the possibility to correct labels. That is, we can query the true label $y$ for $(x, \tilde{y}) \in \tilde{\mathcal{S}}$. We query single (or small batches of) labels in an iterative process. For any $(x, \tilde{y})$ for which we obtain the true label $y$, we remove $(x, \tilde{y})$ from $\tilde{\mathcal{S}}$ and add $(x, y)$ to the clean training set $\mathcal{S}$. As re-labeling is assumed to be costly, we need a method for selecting the potentially most informative examples for correction.

We propose to greedily select the example(s) having the strongest expected influence on the error measure in (5.1). This criterion was suggested by Settles et al. [2008] in the context of multiple-instance active learning. It has the computational advantage that retraining of the model is not required for the selection process, as it is, for instance, in *expected error reduction* [Roy and McCallum, 2001]. We approximate the expected model change by the difference between the gradient of the error measure before and after correcting the respective label. After we have selected an example $(x_j, \tilde{y}_j)$ and corrected its label to $y_j$ the regularized empirical risk changes to

$$L_j(h) := \sum_{(x,y) \in \mathcal{S} \cup \{(x_j, y_j)\}} l(h(x), y) + \sum_{(x,\tilde{y}) \in \tilde{\mathcal{S}} \setminus \{(x_j, \tilde{y}_j)\}} \tilde{l}(h(x), \tilde{y}) + \lambda \Omega(h) \ . \quad (5.2)$$

We assume a differentiable error measure $L$ and define

$$g(h) := \frac{\partial L(h)}{\partial w} \ \text{ and } \ g_j(h) := \frac{\partial L_j(h)}{\partial w}$$

as the gradients of $L$ and $L_j$ with respect to the model parameter $w$ of the hypothesis $h$. Our approach is to pick

$$\begin{aligned} (x^*, \tilde{y}^*) &= \operatorname*{arg\,max}_{(x_j, \tilde{y}_j) \in \tilde{\mathcal{S}}} \ \mathbb{E}_{y_j | x_j, \tilde{y}_j} \Big[ \| g_j(h) - g(h) \| \Big] \\ &= \operatorname*{arg\,max}_{(x_j, \tilde{y}_j) \in \tilde{\mathcal{S}}} \ \mathbb{E}_{y_j | x_j, \tilde{y}_j} \left[ \left\| \frac{\partial}{\partial w} \Big( l(h(x), y) - \tilde{l}(h(x), \tilde{y}) \Big) \right\| \right] \end{aligned} \quad (5.3)$$

for re-labeling. Different noise-aware loss functions $\tilde{l}$ lead to different algorithms. We consider the following natural choices for $\tilde{l}$.

**Noise-agnostic estimator.** The simplest way to deal with label noise is to neglect it and just choose the noise-aware loss $\tilde{l}$ to coincide with the standard loss $l$.

**Unbiased estimator.** If we know the noise rates $\rho_{-1}$ and $\rho_{+1}$, we can define a loss $l_{\mathrm{u}}$ on the noisy data as an unbiased estimator of the standard loss $l$ on the clean data,

$$\mathbb{E}_{\tilde{y}} \Big[ l_u(h(x), \tilde{y}) \Big] = l(h(x), y)$$

for all $x$, $y$, and $h$. For binary classification, Natarajan et al. [2013] have shown that this holds for

$$l_{\mathrm{u}}(h(x), y) = \alpha_y l(h(x), y) - \beta_y l(h(x), -y) \ ,$$

with

$$\alpha_y := \frac{1 - \rho_{-y}}{1 - \rho_{-1} - \rho_{+1}} \text{ and } \beta_y := \frac{\rho_y}{1 - \rho_{-1} - \rho_{+1}} \ .$$

**Maximum likelihood estimator.** Following Bootkrajang and Kabán [2012], we can consider the true label $y$ as a latent variable and write the posterior probability of a noisy label with class-conditional noise as

$$\begin{aligned} p(\tilde{y}|x) &= \sum_y p(\tilde{y}, y|x) \\ &= \sum_y p(\tilde{y}|y) p(y|x) \\ &= \big(1 - \rho_{\tilde{y}}\big) p(y = \tilde{y}|x) + \rho_{-\tilde{y}}\, p(y = -\tilde{y}|x) \ . \end{aligned}$$

If we assume that we can model $p(y|x)$ with the current hypothesis $h \in \mathcal{H}$, we can write the likelihood of the model parameters $w$ given a single training example as

$$\mathcal{L}(w|x, \tilde{y}) \propto p(\tilde{y}|x, w) = \big(1 - \rho_{\tilde{y}}\big) h(x) + \rho_{-\tilde{y}} \big(1 - h(x)\big) \ .$$

Taking the negative log-likelihood, the noise-aware maximum likelihood loss can be defined as

$$l_{\mathrm{ML}}(h(x), \tilde{y}) := -\log \Big( \big(1 - \rho_{\tilde{y}}\big) h(x) + \rho_{-\tilde{y}} \big(1 - h(x)\big) \Big) \ . \tag{5.4}$$

This maximum likelihood approach coincides with the unbiased estimator only in the noiseless case.

## 5.3 Active Label Correction with Logistic Regression

In the following, the active label correction strategies are applied to logistic regression. Logistic regression has the advantages that its output can be interpreted as a probability (allowing its use for the noise-aware maximum likelihood estimator), that the unbiased estimator is convex [Natarajan et al., 2013], and that it can be easily extended to multi-class classification and to deep neural network architectures (e.g., convolutional neural networks, see experiments in section 5.6). For logistic regression we have

$$h(x) := \sigma(x) = \frac{1}{1 + \exp(-w^\top x)} \ .$$

A natural choice for the loss is the cross-entropy

$$l(h(x), y) := [y = +1] \log \frac{1}{h(x)} + [y = -1] \log \frac{1}{1 - h(x)} \ ,$$

where $[\cdot]$ is the Iverson bracket. Combining both components gives

$$l(h(x), y) = -\log \sigma(yx) \ .$$

To compute the expectation in the criterion (5.3), we re-write the probabilities using Bayes' rule and the noise model $\tilde{Y} \perp\!\!\!\perp X|Y$. Assuming that our current output $\sigma(yx)$ models the posterior probability $p(Y|X)$, we get

$$\begin{aligned} p(Y = -\tilde{y}|\tilde{Y} = \tilde{y}, X = x) &\propto p(\tilde{Y} = \tilde{y}|Y = -\tilde{y})p(Y = -\tilde{y}|X = x) \\ &= \rho_{-\tilde{y}}\sigma(-\tilde{y}x) \\ p(Y = +\tilde{y}|\tilde{Y} = \tilde{y}, X = x) &\propto p(\tilde{Y} = \tilde{y}|Y = +\tilde{y})p(Y = +\tilde{y}|X = x) \\ &= (1 - \rho_{\tilde{y}})\sigma(\tilde{y}x) \ . \end{aligned}$$

Now, we derive three novel active label correction algorithms. These pick the next example $(x^*, \tilde{y}^*)$ to be corrected based on the aforementioned loss functions.

### Weighted Uncertainty Re-Labeling (WURL)

Using standard regularized logistic regression, the gradient $g$ becomes

$$g(\sigma) = -\sum_{(x,y)\in\mathcal{S}} yx\sigma(-yx) - \sum_{(x,\tilde{y})\in\tilde{\mathcal{S}}} \tilde{y}x\sigma(-\tilde{y}x) + \frac{\partial}{\partial w}\Omega(w) \ . \tag{5.5}$$

The gradient $g_j$, which measures the change rate after replacing $\tilde{y}_j$ with $y_j$ is then

$$\begin{aligned} g_j(\sigma) = -\sum_{(x,y)\in\mathcal{S}\cup\{(x_j,y_j)\}} & yx\sigma(-yx) \\ & - \sum_{(x,\tilde{y})\in\tilde{\mathcal{S}}\setminus\{(x_j,\tilde{y}_j)\}} \tilde{y}x\sigma(-\tilde{y}x) + \frac{\partial}{\partial w}\Omega(w) \ . \end{aligned} \tag{5.6}$$

Inserting (5.5) and (5.6) into (5.3) gives

$$\begin{aligned} (x^*, \tilde{y}^*) =& \underset{(x_j,\tilde{y}_j)\in\tilde{\mathcal{S}}}{\arg\max} \ \mathbb{E}_{y_j|x_j,\tilde{y}_j}\Big[\|g_j(\sigma) - g(\sigma)\|\Big] \\ =& \underset{(x_j,\tilde{y}_j)\in\tilde{\mathcal{S}}}{\arg\max} \ \|\tilde{y}_j x_j \sigma(\tilde{y}_j x_j) + \tilde{y}_j x_j \sigma(-\tilde{y}_j x_j)\| \, p(Y = -\tilde{y}_j|\tilde{Y} = \tilde{y}_j, X = x_j) \\ =& \underset{(x_j,\tilde{y}_j)\in\tilde{\mathcal{S}}}{\arg\max} \ \|x_j\| \, p(Y = -\tilde{y}_j|\tilde{Y} = \tilde{y}_j, X = x_j) \\ =& \underset{(x_j,\tilde{y}_j)\in\tilde{\mathcal{S}}}{\arg\max} \ \rho_{-\tilde{y}_j}\|x_j\| \, \sigma(-\tilde{y}_j x_j) \\ :=& \underset{(x_j,\tilde{y}_j)\in\tilde{\mathcal{S}}}{\arg\max} \ s_{\mathrm{W}}(x_j, \tilde{y}_j) \ , \end{aligned}$$

where we assume that $p(y \,|\, x)$ can be replaced by $\sigma(yx)$. The criterion $s_{\mathrm{W}}$ suggests to pick the example which has the least confidence predicting its given label,

weighted by the length of the sample vector $\|x\|$ and the label flip probability $\rho_{-\tilde{y}}$ of the opposite class. Empirical results suggest that the bias towards input patterns with larger norm does not affect performance in practice [Settles et al., 2008, Settles and Craven, 2008]. Note that we do not assume that the given model classifies the clean points perfectly. We only assume that our current model is a good predictor for $p(Y|X)$. If we assume $\rho_{+1} = \rho_{-1}$ and that all inputs have the same norm $\|x\|$, the criterion reduces to the one used in *uncertainty re-labeling* as proposed by Rebbapragada et al. [2012].

### Robust Unbiased Weighted Uncertainty Re-Labeling (U-WURL)

Choosing the unbiased estimator $l_{\mathrm{u}}$ for the noise-aware loss $\tilde{l}$ gives

$$L(\sigma) = \sum_{(x,y)\in\mathcal{S}} l(\sigma(x), y) + \sum_{(x,\tilde{y})\in\tilde{\mathcal{S}}} \Big(\alpha_{\tilde{y}} l(\sigma(x), \tilde{y}) - \beta_{\tilde{y}} l(\sigma(x), -\tilde{y})\Big) + \Omega(w) \quad (5.7)$$

with gradient

$$g(\sigma) = -\sum_{(x,y)\in\mathcal{S}} yx\sigma(-yx) - \sum_{(x,\tilde{y})\in\tilde{\mathcal{S}}} \Big(\alpha_{\tilde{y}}\tilde{y}x\sigma(-\tilde{y}x) + \beta_{\tilde{y}}\tilde{y}x\sigma(\tilde{y}x)\Big) + \frac{\partial}{\partial w}\Omega(w) \ .$$

Following our selection criterion (5.3), we then correct the example $(x^*, \tilde{y}^*)$ that maximizes

$$\begin{aligned}
s_{\mathrm{U}}(x,\tilde{y}) &:= \mathbb{E}_{y|x,\tilde{y}}\Big[\|g_j(\sigma) - g(\sigma)\|\Big] \\
&= \mathbb{E}_{y|x,\tilde{y}}\Big[\|x\|\big|-y\sigma(-yx) + \tilde{y}\alpha_{\tilde{y}}\sigma(-\tilde{y}x) + \tilde{y}\beta_{\tilde{y}}\sigma(\tilde{y}x)\big|\Big] \\
&= \|x\|\Big(\big|\tilde{y}\sigma(\tilde{y}x) + \tilde{y}\alpha_{\tilde{y}}\sigma(-\tilde{y}x) + \tilde{y}\beta_{\tilde{y}}\sigma(\tilde{y}x)\big|\rho_{-\tilde{y}}\sigma(-\tilde{y}x) \\
&\quad + \big|-\tilde{y}\sigma(-\tilde{y}x) + \tilde{y}\alpha_{\tilde{y}}\sigma(-\tilde{y}x) + \tilde{y}\beta_{\tilde{y}}\sigma(\tilde{y}x)\big|(1-\rho_{\tilde{y}})\sigma(\tilde{y}x)\Big) \\
&= \|x\|\Big(\alpha_{\tilde{y}}\rho_{-\tilde{y}}\sigma(-\tilde{y}x) + \beta_{\tilde{y}}(1-\rho_{\tilde{y}})\sigma(\tilde{y}x)\Big) \qquad (5.8) \\
&\propto \|x\|\Big((1-\rho_{-\tilde{y}})\rho_{-\tilde{y}}\sigma(-\tilde{y}x) + \rho_{\tilde{y}}(1-\rho_{\tilde{y}})\sigma(\tilde{y}x)\Big) \qquad (5.9) \\
&= \|x\|\Big((1-\rho_{-1})\rho_{-1}\sigma(-x) + \rho_{+1}(1-\rho_{+1})\sigma(x)\Big) \ .
\end{aligned}$$

To arrive at (5.8) we use $\alpha_y - \beta_y = 1$ and $\sigma(-yx) = 1 - \sigma(yx)$. The proportionality in (5.9) is due to multiplying out the positive constant factor $(1 - \rho_{+1} - \rho_{-1})$ and because one can substitute $\tilde{y}$ with 1 as the term is a symmetric function of $\tilde{y}$. We define

$$\begin{aligned}
a &:= \rho_{+1}(1-\rho_{+1}) - \rho_{-1}(1-\rho_{-1}) \text{ and} \\
b &:= \rho_{-1}(1-\rho_{-1})
\end{aligned}$$

to get the selection criterion

$$\begin{aligned}(x^*, \tilde{y}^*) &= \underset{(x_j,\tilde{y}_j)\in\tilde{\mathcal{S}}}{\arg\max}\ \mathbb{E}_{y_j|x_j,\tilde{y}_j}\Big[\|g_j - g\|\Big] \\ &= \underset{(x_j,\tilde{y}_j)\in\tilde{\mathcal{S}}}{\arg\max}\ \|x_j\|\Big(a\,\sigma(x_j) + b\Big) \\ &= \underset{(x_j,\tilde{y}_j)\in\tilde{\mathcal{S}}}{\arg\max}\ s_{\mathrm{U}}(x_j, \tilde{y}_j)\ .\end{aligned}$$

Thus, we see that selecting the next example for correction consists of an affine transformation of the logistic function, weighted by the example magnitude. This method has the advantage that the optimization of (5.7) is a convex problem in the case of the logistic loss [Natarajan et al., 2013]. However, it leads to a selection criterion which is independent of the noisy label.

### Robust Maximum Likelihood Weighted Uncertainty Re-Labeling (ML-WURL)

For logistic regression, the maximum likelihood loss in (5.4) takes the form as derived by Bootkrajang and Kabán [2012]:

$$\begin{aligned}l_{\mathrm{ML}}(\sigma(x), \tilde{y}) &= -\log\Big((1-\rho_{\tilde{y}})\sigma(\tilde{y}x) + \rho_{-\tilde{y}}\big(1-\sigma(\tilde{y}x)\big)\Big) &(5.10)\\ &= l(\sigma(x), \tilde{y}) - \log\Big(1 + \rho_{-\tilde{y}}\exp(-\tilde{y}w^\top x) - \rho_{\tilde{y}}\Big) &(5.11)\end{aligned}$$

Unfortunately, minimizing $l_{\mathrm{ML}}$ is not a convex problem. The form of (5.11), however, suggests that we might be able to employ DC programming by interpreting it as a difference of two convex functions [Tao, 1997]. If we use $l_{\mathrm{ML}}$ as the noise-aware loss, we get

$$\begin{aligned}L(\sigma) = &\sum_{(x,y)\in\mathcal{S}} l(\sigma(x), y) \\ &+ \sum_{(x,\tilde{y})\in\tilde{\mathcal{S}}}\Big(l(\sigma(x),\tilde{y}) - \log\big(1 + \rho_{-\tilde{y}}\exp(-\tilde{y}w^\top x) - \rho_{\tilde{y}}\big)\Big) + \Omega(w)\end{aligned}$$

with gradient

$$\begin{aligned}g(\sigma) = &-\sum_{(x,y)\in\mathcal{S}} yx\sigma(-yx) \\ &+ \sum_{(x,\tilde{y})\in\tilde{\mathcal{S}}}\tilde{y}x\Big(\frac{\rho_{-\tilde{y}}}{\rho_{-\tilde{y}} + (1-\rho_{\tilde{y}})\exp(\tilde{y}w^\top x)} - \sigma(-\tilde{y}x)\Big) + \frac{\partial}{\partial w}\Omega(w)\end{aligned}$$

leading to

$$\|g_j(\sigma) - g(\sigma)\| = \|x\|\Big| - y\sigma(-yx) - \tilde{y}\Big(\frac{\rho_{-\tilde{y}}}{\rho_{-\tilde{y}} + (1-\rho_{\tilde{y}})\exp(\tilde{y}w^\top x))} - \sigma(-\tilde{y}x)\Big)\Big|\ .$$

Thus, we select the example $(x^*, \tilde{y}^*)$ for correction that maximizes

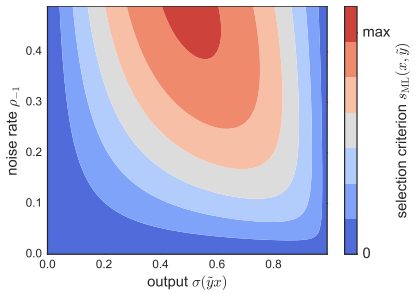
$$
\begin{aligned}
s_{\mathrm{ML}}(x, \tilde{y}) &:= \mathbb{E}_{y|x,\tilde{y}}\Big[\|g_j(\sigma) - g(\sigma)\|\Big] \\
&= \|x\|\Big(\big(1 - \frac{\rho_{-\tilde{y}}}{\rho_{-\tilde{y}} + (1-\rho_{\tilde{y}})\exp(\tilde{y}w^\top x)}\big)\rho_{-\tilde{y}}\sigma(-\tilde{y}x) \\
&\quad + \frac{\rho_{-\tilde{y}}}{\rho_{-\tilde{y}} + (1-\rho_{\tilde{y}})\exp(\tilde{y}w^\top x)}(1-\rho_{\tilde{y}})\sigma(\tilde{y}x)\Big) \\
&= \|x\|\frac{(1-\rho_{\tilde{y}})\rho_{-\tilde{y}}}{\rho_{-\tilde{y}} + (1-\rho_{\tilde{y}})\exp(\tilde{y}w^\top x)}\Big(\exp(\tilde{y}w^\top x)\sigma(-\tilde{y}x) + \sigma(\tilde{y}x)\Big) \\
&= \|x\|\frac{2(1-\rho_{\tilde{y}})\rho_{-\tilde{y}}\sigma(\tilde{y}x)}{\rho_{-\tilde{y}} + (1-\rho_{\tilde{y}})\exp(\tilde{y}w^\top x)} \\
&\propto \|x\|\frac{(1-\rho_{\tilde{y}})\rho_{-\tilde{y}}}{\rho_{-\tilde{y}} + (1-\rho_{\tilde{y}})\exp(\tilde{y}w^\top x)}\sigma(\tilde{y}x) \\
&= \|x\|\frac{(1-\rho_{\tilde{y}})\rho_{-\tilde{y}}}{1 - \rho_{\tilde{y}} + \rho_{-\tilde{y}} + \rho_{-\tilde{y}}\exp(-\tilde{y}w^\top x) + (1-\rho_{\tilde{y}})\exp(\tilde{y}w^\top x)} \\
&= \|x\|\frac{\rho_{-\tilde{y}}\sigma(\tilde{y}x) \times (1-\rho_{\tilde{y}})\sigma(-\tilde{y}x)}{\rho_{-\tilde{y}}\sigma(\tilde{y}x) + (1-\rho_{\tilde{y}})\sigma(-\tilde{y}x)} \ .
\end{aligned}
$$

The selection criterion $s_{\mathrm{ML}}$ suggests picking examples uniformly at random in the noiseless case. If the noise rates are equal, it proposes to pick examples with the highest uncertainty, weighted by the noise level. In Figure 5.1a we see how the selection criterion changes as a function of the symmetric label noise rate $\rho = \rho_{-1} = \rho_{+1}$ and the output of the classifier $\sigma(\tilde{y}x)$ indicating its uncertainty. We see that with increasing noise more weight is put on the uncertainty of the classifier, whereas with decreasing noise the selection criterion approaches a uniform distribution, see Figure 5.1b. The higher the noise, the more important the classifier's confidence becomes. It is also interesting to note that the selection criterion is not symmetric w.r.t. to $\sigma(\tilde{y}x)$, giving more weight to higher certainty than to lower. This also holds for $\rho_{-1} = \rho_{+1}$.

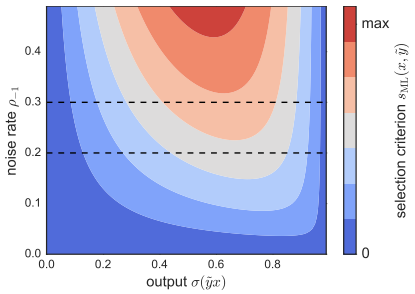
## 5.4 Active Label Correction with Softmax Regression

The aforementioned algorithms can all be extended to multi-class versions. To do so, we generalize our results from binary to multinomial logistic regression, also known as softmax regression. Different from the binary case, the output of the classifier is a categorical distribution over the label space. We have the true and noisy labels $Y, \tilde{Y} \in \mathcal{Y} = \{1, \ldots, K\}$, a $D$-dimensional feature vector $z \in \mathbb{R}^D$ and the output is $h(z) \in \mathbb{R}_+^K$ with $\sum_{j=1}^K h_j(z) = 1$. The output $h$ is computed by the softmax function

$$
h_y(z) := \sigma_y(z) = \frac{\exp(z_y)}{\sum_{y' \in \mathcal{Y}} \exp(z_{y'})} \ ,
$$




(a) Behavior of the selection criterion $s_{\mathrm{ML}}$ in the symmetric case $\rho_{-1} = \rho_{+1}$.




(b) Behavior of the selection criterion $s_{\mathrm{ML}}$ in the asymmetric case, where $\rho_{+1} = 0.1$.

Figure 5.1: The selection criterion $s_{\mathrm{ML}}$ as a function of the negative noise rate $\rho_{-1}$ and the output of the classifier $\sigma(x)$ for the case $\tilde{y} = +1$. On the left the symmetric case $\rho_{-1} = \rho_{+1}$ is shown, on the right we see the behavior of the selection criterion when $\rho_{+1} = 0.1$. The dashed lines show the two cases we considered in our experiments.

which models the posterior probability $p(y|z)$. Using the cross-entropy loss as in the logistic regression case leads to the pointwise loss

$$l(h(z), \tilde{y}) = -\log \sigma_{\tilde{y}}(z) \ .$$

This loss can be readily inserted into the standard and the unbiased loss, leading to multi-class equivalents of WURL and U-WURL. Note that the feature vector $z$ of the pointwise loss functions does not have to be a linear function of the inputs, but can also be a non-linear feature representation within a deep neural network (see below). To compute the loss and the selection criterion of ML-WURL, we re-write (5.10) as

$$l_{\mathrm{ML}}(h(z), \tilde{y}) = -\log \sum_{j=1}^{K} p(\tilde{Y} = \tilde{y}|Y = j)\, \sigma_j(z) = -\log \sum_{j=1}^{K} \gamma_{j\tilde{y}}\, \sigma_j(z) \ ,$$

where we have defined the probability that a label has been flipped from the true label $j$ to the observed label $k$ as

$$\gamma_{jk} := p(\tilde{Y} = k|Y = j) \ .$$

Following Bootkrajang and Kabán [2012], the corresponding gradient with respect to the elements $z_c$ of the feature vector $z$ is computed as

$$\frac{\partial l_{\mathrm{ML}}(h(z), \tilde{y})}{\partial z_c} = \sigma_c(z) - \frac{\gamma_{c\tilde{y}}\, \sigma_c(z)}{\sum_{j=1}^{K} \gamma_{j\tilde{y}}\, \sigma_j(z)} \ ,$$

which reduces in the noiseless case ($\gamma_{c\tilde{y}} = [c = \tilde{y}]$) to the gradient

$$\frac{\partial l_{\mathrm{ML}}(h(z), \tilde{y})}{\partial z_c} = \sigma_c(z) - [c = \tilde{y}] \ .$$

To compute the complete gradient in the case of a deep neural network, we can apply the chain rule efficiently using back-propagation. As in the previous section we can then compute the selection criterion as the expected model change $\mathbb{E}_{y|x,\tilde{y}}[\|g_j(\sigma) - g(\sigma)\|]$. This strategy works with any (deep) multi-class architecture. In our empirical evaluation (see section 5.6), we used a convolutional neural network (CNN) to demonstrate its performance.

## 5.5 Estimating the Noise Rates

The noise-aware selection criteria assume that the noise rates are known. In practice, it is unlikely that they are. In this case we can draw an initial sample uniformly at random and estimate the noise rates by counting the number of corrected labels:

$$\hat{\rho}_k = \frac{\sum_{(x,y,\tilde{y})\in\mathcal{S}_C}[\tilde{y} = -k, y = k]}{\sum_{(x,y,\tilde{y})\in\mathcal{S}_C}[y = k]} \ , \tag{5.12}$$

where $[\cdot]$ is the Iverson bracket, and $\mathcal{S}_C$ is the set of all corrected examples with their noisy and true labels $(x, y, \tilde{y})$. In case $\mathcal{S}_C$ is drawn uniformly at random, $\hat{\rho}_k$ is an unbiased estimator of the corresponding true noise rate. This approach has the drawback that in this initial phase the active learning does not yet exploit the gathered information about the noise rates.

Thus, we want to simultaneously estimate the noise model parameters while using the current noise model for active learning. Drawing examples actively (i.e., non-uniformly) has the drawback that (5.12) becomes biased. One way of dealing with this bias is importance-weighting. Instead of deterministically picking examples that maximize our criterion, in each iteration $t$, we define a sampling probability distribution $p_s(x, \tilde{y}, t)$ over the noisy sample $\tilde{\mathcal{S}}$. This distribution assigns examples with a higher score in the selection criterion a higher probability of being picked. When an example is chosen, it is given an importance weight, defined as the inverse of its sampling probability $p_s$. To avoid infinite importance weights, we also have to make sure that the sampling probability is bounded away from zero by adding a minimum probability $p_{\min} \leq \frac{1}{n}$. Thus, similar to Ganti and Gray [2012], we can define our sampling probability distribution as

$$p_s(x, \tilde{y}, t) := p_{\min}(t) + (1 - n \cdot p_{\min}(t))\frac{s(x, \tilde{y})}{\sum_{(x,\tilde{y})\in\mathcal{S}\cup\tilde{\mathcal{S}}} s(x, \tilde{y})} \ , \tag{5.13}$$

where $n = |\mathcal{S} \cup \tilde{\mathcal{S}}|$ and $s(x, \tilde{y})$ is a non-negative selection criterion, for example one of the three criteria $s_{\mathrm{W}}$, $s_{\mathrm{U}}$ , $s_{\mathrm{ML}}$ we introduced above. If $s(x, \tilde{y}) = 0$ for all $(x, \tilde{y})$, we just sample uniformly at random by setting $p_s(x, \tilde{y}, t) = \frac{1}{n}$. Following Ganti and Gray [2012], we define the minimum probability $p_{\min}(t) = \frac{1}{nt^{\kappa}}$, where $\kappa$ is a hyperparameter that tunes the trade-off between exploiting the criterion and exploring new examples for noise rate estimation. Note that in order to draw each example independently, we draw with replacement. Thus, it is possible that an example is selected multiple times. In this case, we just re-use its previously corrected example at no cost.

To account for the non-uniform selection, we can estimate the noise rates in iteration $t$ by

$$\hat{\rho}_k(t) = \frac{\sum_{\tau=1}^{t} \sum_{(x,y,\tilde{y})\in\mathcal{S}_C} w(x,\tilde{y},\tau)[\tilde{y}=-k, y=k]}{\sum_{\tau=1}^{t} \sum_{(x,y,\tilde{y})\in\mathcal{S}_C} w(x,\tilde{y},\tau)[y=k]} \ , \tag{5.14}$$

where we define $w(x,\tilde{y},\tau) := \frac{1}{p_s(x,\tilde{y},\tau)}$ and $k \in \{-1,1\}$ is the label of interest. Although $\hat{\rho}_k(t)$ is not an unbiased estimator of the true unknown noise rate either, one can show that its bias vanishes for $t \to \infty$.

By employing importance weights we are able to simultaneously estimate the noise rates while maximizing the accuracy through active learning. In order to avoid an unstable start-up phase, it is possible to integrate a prior probability. This prior can be informed by methods that estimate noise rates from noisy samples only, e.g., [Liu and Tao, 2016, Menon et al., 2015].

## 5.6 Experiments

We randomly sampled training sets of 2000 patterns from different benchmark datasets and flipped their labels with probabilities $\rho_{-1} \in \{0.2, 0.3\}$ and $\rho_{+1} = 0.1$. We evaluated the accuracies of the classifiers on separate test sets of 5000 samples (in the case of the dataset 'ad' on 359 as more data are not available). We averaged each experimental outcome over 30 trials.

The result achieved by training the predictive model on the full training set without label noise is called the *clean baseline*. It indicates the performance limit achieved by correcting the whole dataset. For the active label correction, we started with 0 corrected examples ($\mathcal{S} = \emptyset$) and stop when half of the training set is corrected ($|\mathcal{S}| = 1000$). For all experiments we set the trade-off parameter $\kappa = 0.5$, as suggested by Theorem 3 for the squared loss in Ganti and Gray [2012]. To start with a stable estimated noise rate, we employ a burn-in phase of sampling $n_{\text{burn-in}} = 50$ examples uniformly at random, as in passive sampling.

The algorithms we devised in this paper are referred to as *weighted uncertainty re-labeling (WURL)* and *robust ML weighted uncertainty re-labeling (ML-WURL)*, see section 5.3 and 5.3, respectively. For comparison, we consider *passive re-labeling* for the standard loss, that is, choosing an example to correct uniformly at random. The algorithm presented by Rebbapragada et al. [2012] is referred to as *uncertainty re-labeling*, which amounts to selecting the example with the closest absolute distance to the decision hyperplane. As robust *unbiased weighted uncertainty re-labeling (U-WURL)* always performed worse than the others proposed methods, it is omitted from the main plots (for U-WURL results see section 5.8).

### Logistic Regression

The algorithms are evaluated on the binary classification benchmark datasets 'a1a', 'ad', 'covtype', 'w1a', 'mushrooms', 'cod-rna' and 'ijcnn1' from the LIBSVM

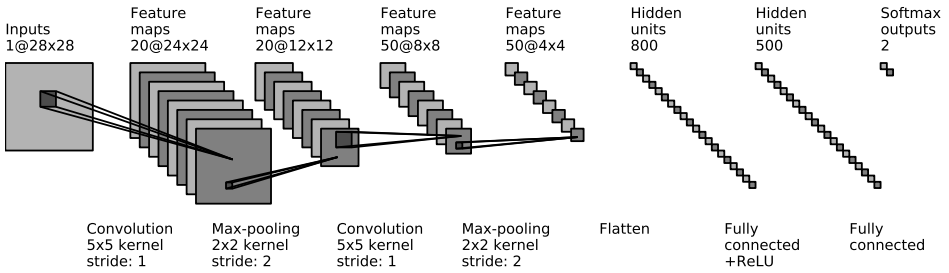



Figure 5.2: Architecture of the CNN.[3] Each layer is processed in mini-batches of 64 examples. The loss is computed as the cross entropy between the softmax-output and the labels in the noise-agnostic case. For ML-WURL we used the loss detailed in section 5.4.

data repository.[2] We evaluated each classifier after selecting one additional example and assigning it its true label. We tested $\ell_2$-regularization $\Omega(h) = \frac{1}{2}\|x\|^2$ with $\lambda \in \{1, 10\}$. Each loss is optimized using L-BFGS and each iteration is warm-started.

Figure 5.3 shows selected empirical results. We can see that on all shown datasets ML-WURL reaches the results of the clean baseline fastest. Uncertainty sampling dominates in the beginning only in the case of the single dataset 'w1a', however, the robust maximum likelihood estimator still converges faster to the clean baseline. Additional experiments on the remaining datasets and with different parameter settings can be found in section 5.8.

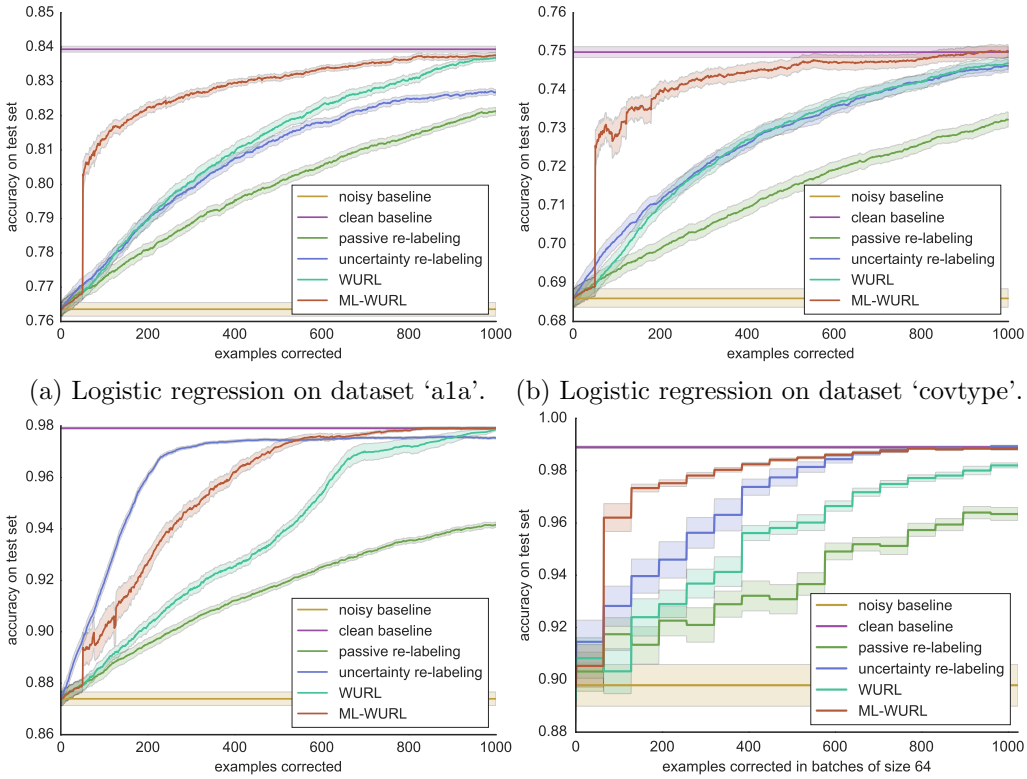
### Convolutional Neural Network

To show that our approach also works well with non-linear, deep architectures [LeCun et al., 2015], we have included one experiment with a convolutional neural network (CNN). We trained a CNN using the 'LeNet'-architecture on the numbers '8' and '9' from the MNIST dataset [LeCun et al., 1998] with Rectified Linear Unit (ReLU) activations instead of sigmoids. It consists of two convolution layers, each followed by a max-pooling layer applied with a stride length of 2, and two fully connected layers. The outputs are computed using the softmax function. The architecture of the CNN is shown in Figure 5.2.

Different from the previous experiments, in each round we selected a batch of the 64 examples that had the highest values of the selection criterion. In each round we trained the CNN for $10,000$ iterations using stochastic gradient descent with weight decay of 0.0005, momentum of 0.9 and a learning rate of 0.01 using a modified version of the Caffe-library to use the label-robust losses [Jia et al., 2014]. The CNN experiment confirmed that ML-WURL outperforms the alternative algorithms.

[2] `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`

[3] The illustration was created using the code at `https://github.com/gwding/draw_convnet`




(a) Logistic regression on dataset 'a1a'.

(b) Logistic regression on dataset 'covtype'.

(c) Logistic regression on dataset 'w1a'.

(d) CNN classifying 8-vs-9 on MNIST.

Figure 5.3: Empirical results on different benchmark datasets. Shown are the mean test set errors and standard deviations over 30 trials for logistic regression ($\lambda = 1.0$) and CNN. The active learning parameters were $\rho_{-1} = 0.3$, $\rho_{+1} = 0.1$, $n_{\text{burn-in}} = 50$, $\kappa = 0.5$. Experiments on further datasets and for different parameter settings can be found in section 5.8.

## 5.7 Conclusion

We presented a principled approach to active label correction (or learning from weak teachers). We propose to employ loss functions that depend on a noise model and to apply the maximum expected model change criterion to the corresponding regularized risk functionals. Class-conditional noise was assumed as a model for the true noise. We demonstrated how to learn the parameters of the noise model during the active learning process. Different choices of the loss function were considered and corresponding algorithms were derived. On a range of datasets, the algorithm ML-WURL (viewing the true label as a latent variable and computing the maximum likelihood estimate of the model parameters) consistently gave the best results, outperforming the state-of-the-art approach uncertainty re-labeling. ML-WURL is an anytime algorithm that simultaneously estimates the label noise rates while minimizing the risk. The approach can be extended to multi-class problems and other loss functions as well as other hypotheses classes.

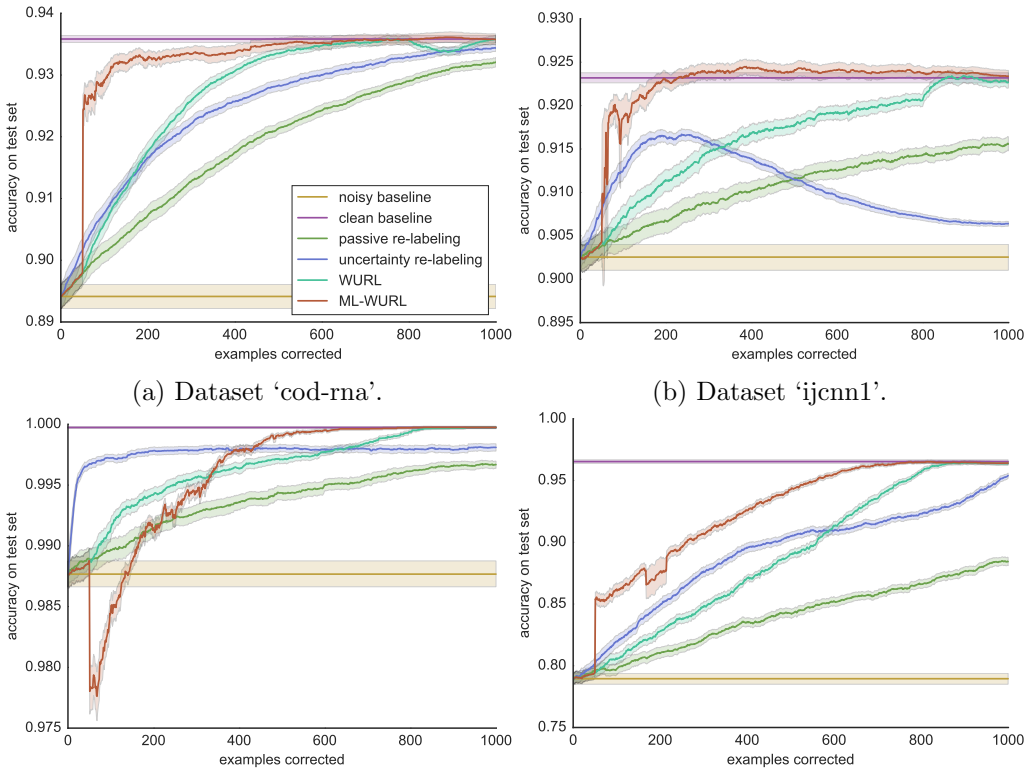
## 5.8 Additional Experiments

We present results for the same datasets as in the main section plus several additional ones with various parameter settings to demonstrate that the overall performance of the proposed algorithm relative to its competitors is unaffected by specific choices. Furthermore, we show the performance of robust unbiased weighted uncertainty re-labeling (U-WURL).

In general, the behaviors of the algorithms match the ones shown on the datasets in the main section. There are only minor differences: In the 'mushrooms' experiments the noise rate is initially estimated incorrectly and the performance degrades, but the algorithm is able to quickly recover from the bad start. On the dataset 'ijcnn1' the performance of uncertainty re-labeling degrades quickly after the start, while the others perform as expected.

### Additional Datasets

Figure 5.4 depicts some more results on additional datasets using the same parameter settings as in the main section. The algorithms showed the same behavior as with the parameter settings considered earlier.




(a) Dataset 'cod-rna'.

(b) Dataset 'ijcnn1'.

(c) Dataset 'mushrooms'.

(d) Dataset 'ad'.

Figure 5.4: Empirical results on the additional datasets with the parameters as in the main section: $\rho_{-1} = 0.3$, $\rho_{+1} = 0.1$, $\lambda = 1.0$, $n_{\text{burn-in}} = 50$, $\kappa = 0.5$.

### Different Choice of Noise Rates $\rho_{-1}$ and $\rho_{+1}$

We set the noise rates to values with smaller difference, that is, $\rho_{-1} = 0.2$ and $\rho_{+1} = 0.1$. The results are shown in Figure 5.5. It can be seen that the maximum likelihood estimator still dominated. It showed comparable performance as in the case of the larger difference in the noise rates.

### Different Choice of Regularization Parameter $\lambda$

We set the regularization parameter to $\lambda = 10.0$, shown in Figure 5.6. The curves show the same behavior as in the main section with the maximum likelihood estimator typically dominating.

### Different Choice of Burn-In Sample Size $n_{\text{burn-in}}$

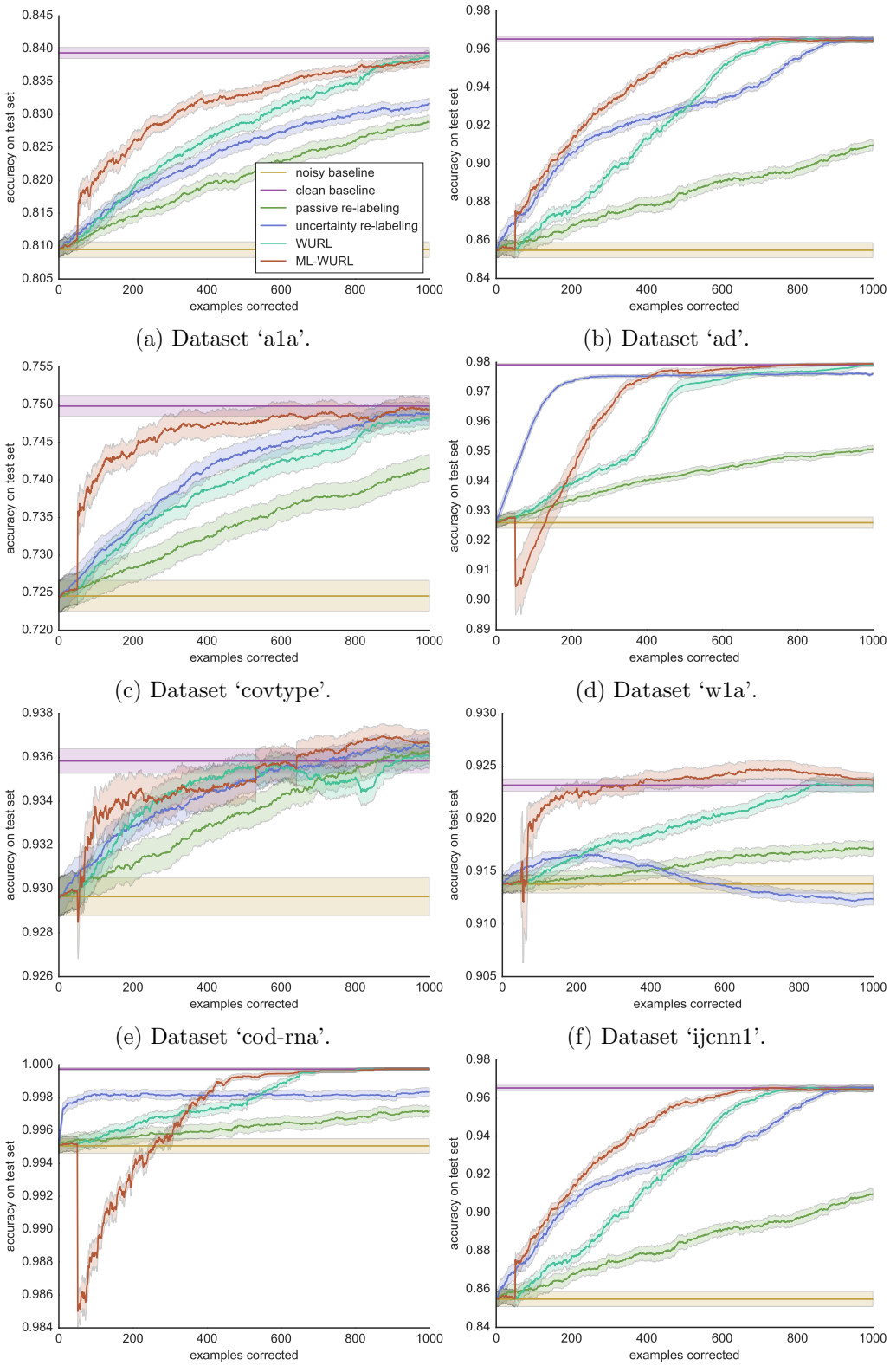
We set the burn-in sample size to $n_{\text{burn-in}} = 0$, shown in Figure 5.7. We see that during the first few iterations the algorithm fluctuated more than without initial uniform sampling for stabilization. After the initial fluctuations, it recovered quickly and converged to the clean baseline as fast as in the other settings.

### Different Choice of Exploration Parameter $\kappa$

We set the exploration parameter $\kappa = 0.1$, shown in Figure 5.8. This lets the probability of picking an example uniformly at random decay more slowly with the number of re-labeled examples. We see that the algorithm behaves more stable during the first few iterations.

### Performance of the Unbiased Estimator

Due to its inferior performance we excluded U-WURL from the main section. Here we include it for completeness, see Figure 5.9. Although in some datasets it performed better than passive re-labeling, it performed always worse than the other active methods.




(a) Dataset 'a1a'.

(b) Dataset 'ad'.

(c) Dataset 'covtype'.

(d) Dataset 'w1a'.

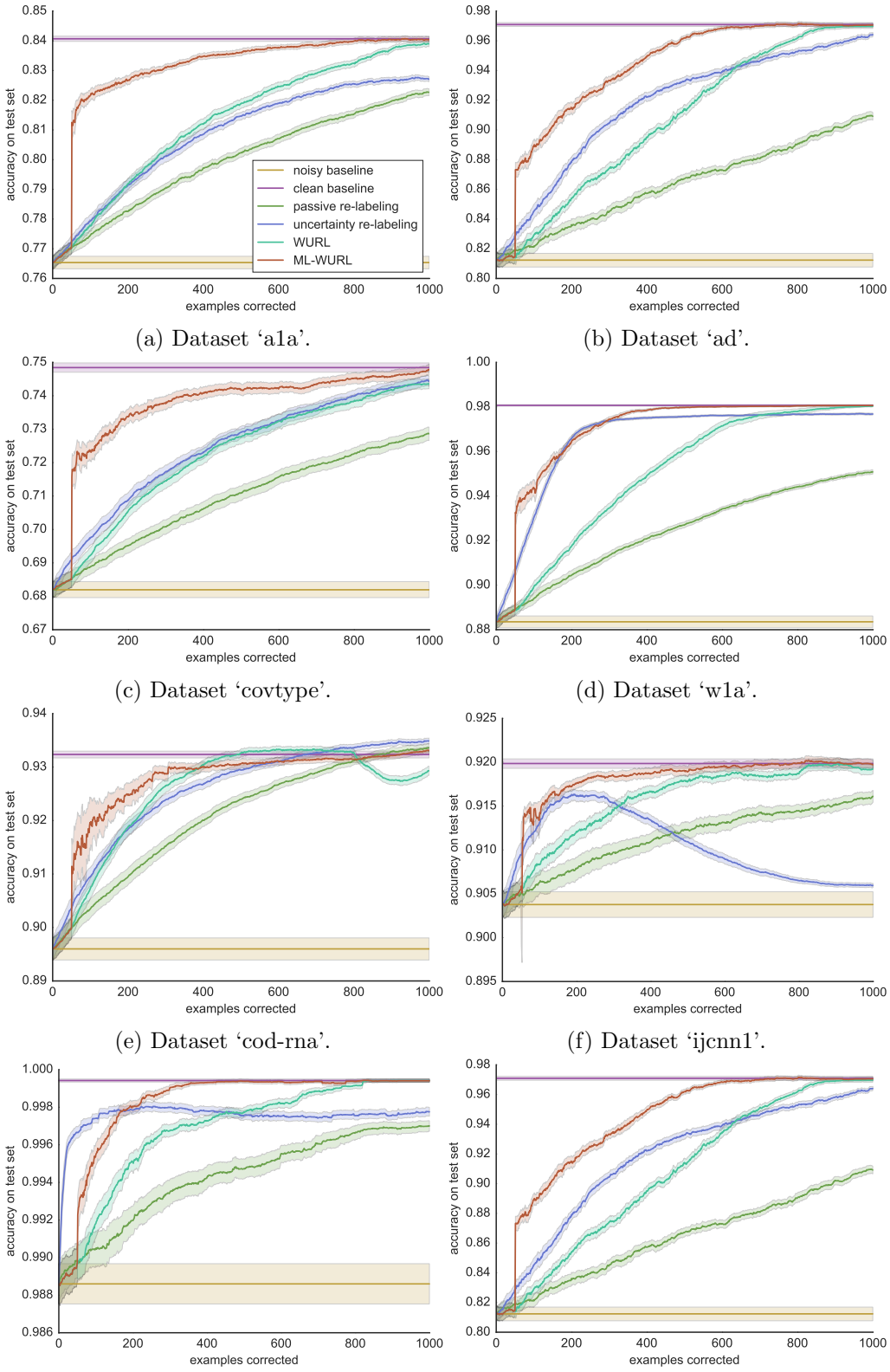
(e) Dataset 'cod-rna'.

(f) Dataset 'ijcnn1'.

(g) Dataset 'mushrooms'.

(h) Dataset 'ad'.

Figure 5.5: Setting noise rates $\rho_{-1} = 0.2$ and $\rho_{+1} = 0.1$. Empirical results on different benchmark datasets with parameters: $\rho_{-1} = 0.2$, $\rho_{+1} = 0.1$, $\lambda = 1.0$, $n_{\text{burn-in}} = 50$, $\kappa = 0.5$.




(a) Dataset 'a1a'.

(b) Dataset 'ad'.

(c) Dataset 'covtype'.

(d) Dataset 'w1a'.

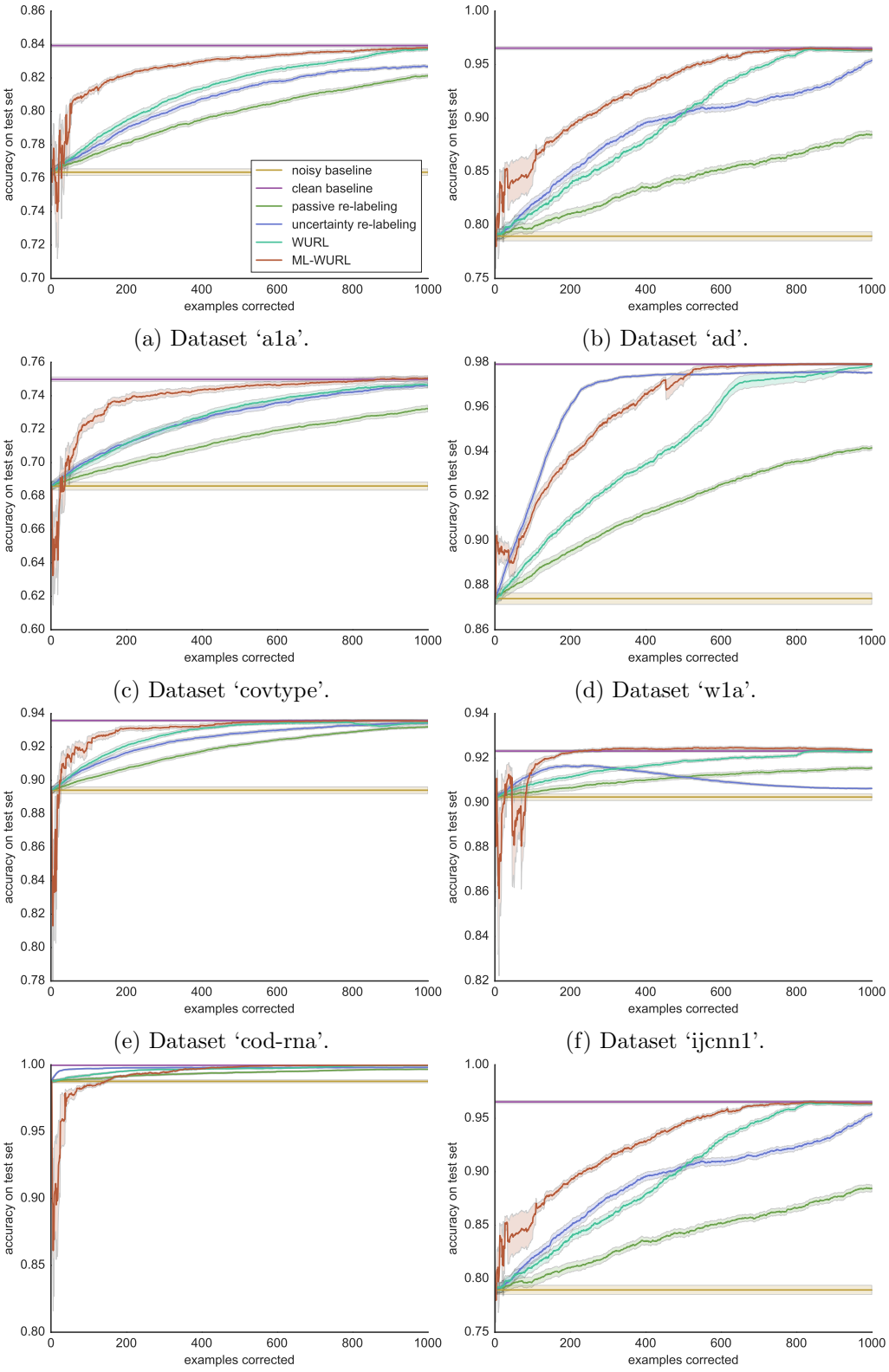
(e) Dataset 'cod-rna'.

(f) Dataset 'ijcnn1'.

(g) Dataset 'mushrooms'.

(h) Dataset 'ad'.

Figure 5.6: Setting regularization parameter $\lambda = 10.0$. Empirical results on different benchmark datasets with parameters: $\rho_{-1} = 0.3$, $\rho_{+1} = 0.1$, $\lambda = 10.0$, $n_{\text{burn-in}} = 50$, $\kappa = 0.5$.




(a) Dataset ‘a1a’.

(b) Dataset ‘ad’.

(c) Dataset ‘covtype’.

(d) Dataset ‘w1a’.

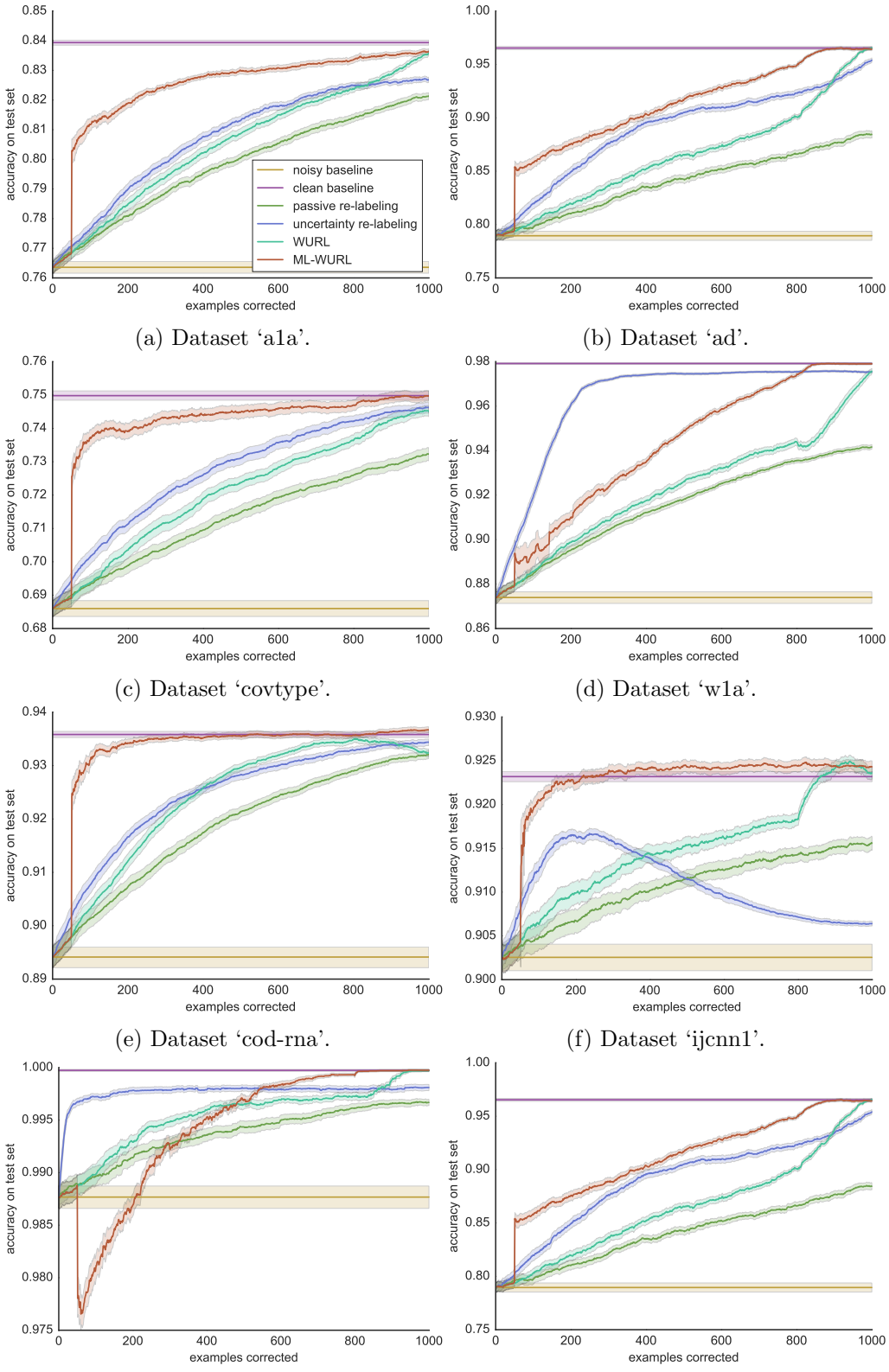
(e) Dataset ‘cod-rna’.

(f) Dataset ‘ijcnn1’.

(g) Dataset ‘mushrooms’.

(h) Dataset ‘ad’.

Figure 5.7: Setting burn-in sample size $n_{\text{burn-in}} = 0$. Empirical results on different benchmark datasets with parameters: $\rho_{-1} = 0.3$, $\rho_{+1} = 0.1$, $\lambda = 1.0$, $n_{\text{burn-in}} = 0$, $\kappa = 0.5$.




(a) Dataset ‘a1a’.

(b) Dataset ‘ad’.

(c) Dataset ‘covtype’.

(d) Dataset ‘w1a’.

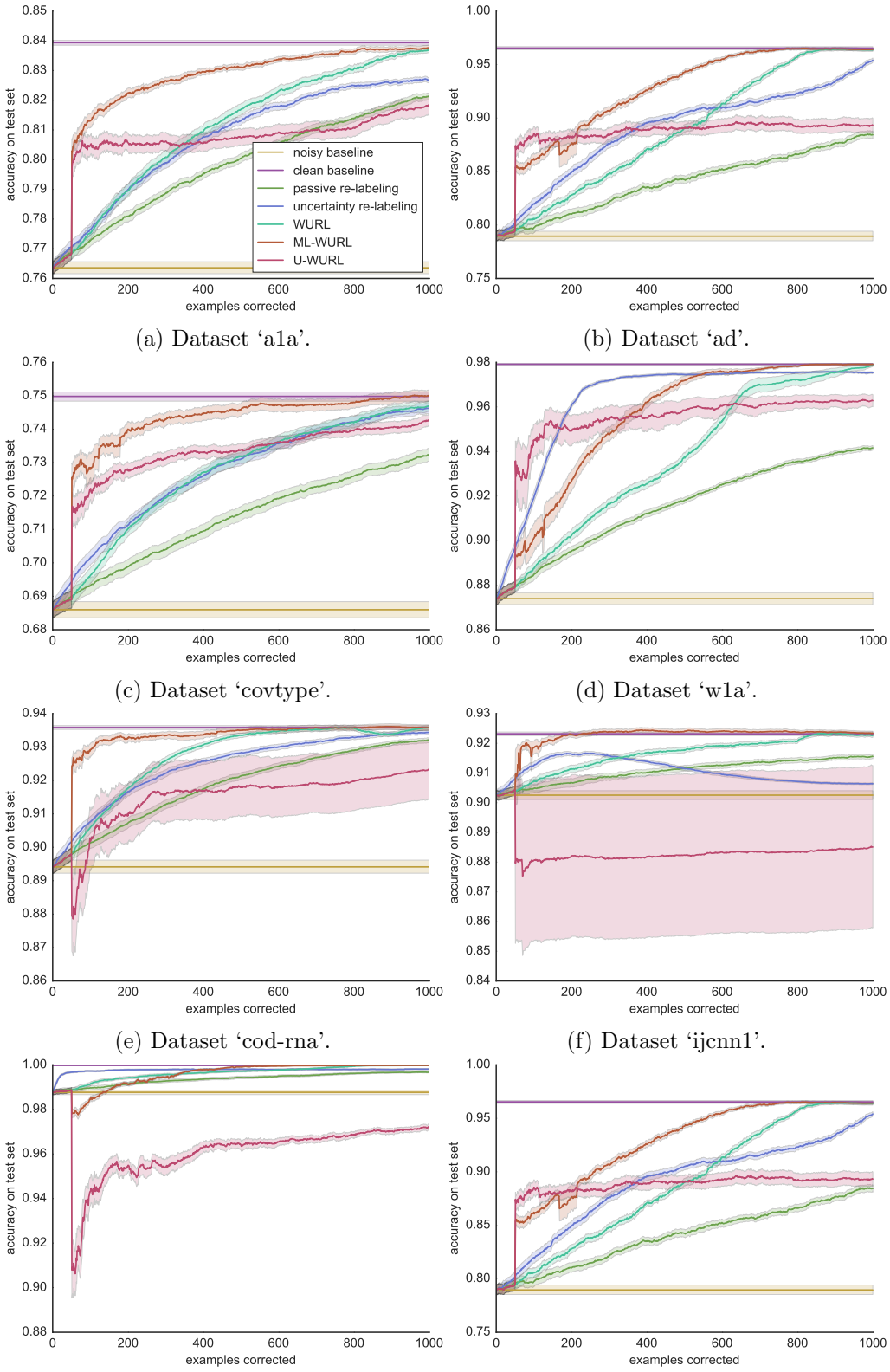
(e) Dataset ‘cod-rna’.

(f) Dataset ‘ijcnn1’.

(g) Dataset ‘mushrooms’.

(h) Dataset ‘ad’.

Figure 5.8: Setting the exploration parameter $\kappa = 0.1$. Empirical results on different benchmark datasets with parameters: $\rho_{-1} = 0.3$, $\rho_{+1} = 0.1$, $\lambda = 1.0$, $n_{\text{burn-in}} = 50$, $\kappa = 0.1$.




(a) Dataset ‘a1a’.

(b) Dataset ‘ad’.

(c) Dataset ‘covtype’.

(d) Dataset ‘w1a’.

(e) Dataset ‘cod-rna’.

(f) Dataset ‘ijcnn1’.

(g) Dataset ‘mushrooms’.

(h) Dataset ‘ad’.

Figure 5.9: Empirical results including the unbiased estimator with parameters: $\rho_{-1} = 0.3$, $\rho_{+1} = 0.1$, $\lambda = 1.0$, $n_{\text{burn-in}} = 50$, $\kappa = 0.5$.

# Chapter 6

# Conclusion and Future Work

In this thesis we considered problems that arise when learning from biased datasets in the context of astronomical applications.

In chapter 2, we reviewed current machine learning and image analysis algorithms for solving large-scale astronomical problems. We illustrated the need to develop new algorithms that are able to handle ever-larger sky surveys. Furthermore, label and measurement noise have to be controlled. We argued that in astronomical problems in which only the accuracy of the predictor matters, data-driven machine learning models can outperform interpretable physical models. We focused on exemplary results, discussed main challenges, and highlighted some recent methodological advancements in machine learning and image analysis triggered by astronomical applications.

In chapter 3, we surveyed algorithms for active learning with support vector machines. After a brief introduction to the active learning problem, we discussed different query strategies for selecting informative data points and reviewed how these strategies give rise to different variants of active learning with support vector machines (SVMs). These have convenient properties that make it easy to evaluate how unlabeled samples would influence the model if they were labeled and included in the training set. Therefore, SVMs are particularly well-suited for active learning. However, we showed that there are several challenges that still need to be addressed, such as efficient learning, dealing with multiple classes, and that actively choosing the training data introduces a selection bias. Importance weighting seems to be most promising to counteract this bias, and it can be easily incorporated into an active SVM learner. Most of the research in active SVM learning so far has focused on binary decision problems. A challenge for future research is to develop efficient active learning algorithms for multi-class SVMs that address the nature of the multi-class decision in a more principled way.

In chapter 4, we considered the scenario in which we do not spend a labeling budget. Instead, we used large amounts of unlabeled data to importance-weight the training sample to minimize a possible domain shift. We considered an efficient nearest neighbor density ratio estimator that can exploit large samples to increase the accuracy of the weight estimates. To solve the problem of choosing the right neighborhood size, we proposed to use cross-validation on a model selection criterion that is unbiased under covariate shift. The resulting algorithm

is our method of choice for density ratio estimation when the feature space dimensionality is small and sample sizes are large. The approach is simple and, because of the model selection, robust.

We showed empirically that it is on a par with established kernel-based methods on relatively small regression benchmark datasets. However, when applied to large-scale photometric redshift estimation, our approach outperformed the state-of-the-art. Subsequent works have validated our findings and shown that this estimator holds state-of-the-art results in redshift estimation [Izbicki et al., 2016]. Future work could consider the theoretical properties of the estimator and an implementation on GPUs for handling datasets with billions of patterns efficiently and at low cost.

In chapter 5 we examined the setting in which the training data is labeled, but the annotations are noisy, for example, due to imprecise measurement or crowd-sourcing. We assumed that each true label can be obtained at a significant cost (e.g., by taking additional measurements or asking human experts). To minimize the labeling costs, we aimed at identifying training patterns for which knowing the true labels maximally improves the learning performance. We devised active a framework for label correction algorithms under the assumption of class-conditional noise, where the true label is conditionally independent of the input given the observed label. To select labels for correction, we adopted the active learning strategy of maximizing the expected model change. We considered the change in regularized empirical risk functionals that use different pointwise loss functions for patterns with noisy and true labels, respectively.

Three different choices of loss functions for the noisy data points then led to different active label correction algorithms. Two of the loss functions considered the label noise rates, which are estimated during learning, where importance weighting compensates for the sampling bias due to active learning. Our experiments showed that on a range of datasets, the algorithm we termed maximum likelihood weighted uncertainty relabeling (ML-WURL) consistently gave the best results, outperforming the state-of-the-art approach uncertainty re-labeling. This approach views the true label as a latent variable and computes the maximum likelihood estimate of the model parameters. ML-WURL is an anytime algorithm that simultaneously estimates the label noise rates while minimizing the risk. We also extended ML-WURL to softmax regression and showed empirically that it also works well with deep models like convolutional neural networks (CNNs). In future work, ML-WURL could be extended to multi-class problems and other loss functions as well as other hypotheses classes.

In the future, there will be a growing interest in the research directions followed in this thesis. With the new astronomical surveys producing terabytes of data per night, the importance of computer science for astronomy will steadily increase. The algorithms for dealing with biased data presented in this thesis were developed with astronomical applications in mind, however, they address general problems. In many practical applications of data analysis, the assumption of i.i.d. data is violated and dealing with the resulting biases is one of the biggest challenges in machine learning.

# Bibliography


H. Aihara, C. A. Prieto, D. An, S. F. Anderson, E. Aubourg, E. Balbinot, T. C. Beers, A. A. Berlind, S. J. Bickerton, and D. e. a. Bizyaev. The eighth data release of the sloan digital sky survey: First data from SDSS-III. *The Astrophysical Journal Supplement Series*, 193(2):29, 2011.

D. Angluin and P. Laird. Learning from noisy examples. *Machine Learning*, 2 (4):343–370, 1988.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

S. Arya, D. M. Mount, N. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1994.

S. Azadi, J. Feng, S. Jegelka, and T. Darrell. Auxiliary image regularization for deep CNNs with noisy labels. *InternationaI Conference on Learning Representations (ICLR)*, 2015. URL `http://arxiv.org/abs/1511.07069`.

F. R. Bach. Active learning for misspecified generalized linear models. In B. Schölkopf, J. C. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 19, pages 65–72. Curran Associates, Inc., 2006.

M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In W. W. Cohen and A. Moore, editors, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 65–72. IEEE Press, 2006.

N. M. Ball and R. J. Brunner. Data mining and machine learning in astronomy. *International Journal of Modern Physics D*, 19(07):1049–1106, 2010.

S. Ben-David and R. Urner. On the hardness of domain adaptation and the utility of unlabeled target samples. In N. H. Bshouty, G. Stoltz, N. Vayatis, and T. Zeugmann, editors, *Algorithmic Learning Theory*, pages 139–153. Springer, 2012.

S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, et al. Analysis of representations for domain adaptation. In B. Schölkopf, J. C. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 19, pages 137–144. Curran Associates, Inc., 2006.

S. Ben-David, T. Lu, T. Luu, and D. Pál. Impossibility theorems for domain adaptation. In Y. W. Teh and D. M. Titterington, editors, *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS): JMLR Workshop and Conference Proceedings*, volume 9, pages 129–136, 2010.

J. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

M. W. Bern and D. Eppstein. Optimization over zonotopes and training support vector machines. In F. K. H. A. Dehne, J. Sack, and R. Tamassia, editors, *Proceedings of the Workshop on Algorithms and Data Structures*, volume 2125, pages 111–121. Springer, 2001.

A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning. In A. P. Danyluk, L. Bottou, and M. L. Littman, editors, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 49–56. ACM Press, 2009.

A. Beygelzimer, J. Langford, Z. Tong, and D. Hsu. Agnostic active learning without constraints. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 23, pages 199–207. Curran Associates, Inc., 2010.

S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning for differing training and test distributions. In Z. Ghahramani, editor, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 81–88. ACM Press, 2007.

S. Bickel, M. Brückner, and T. Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10:2137–2155, 2009.

J. Bootkrajang and A. Kabán. Label-noise robust logistic regression and its applications. In P. A. Flach, T. D. Bie, and N. Cristianini, editors, *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, pages 143–158. Springer, 2012.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, 2005.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Workshop on Computational Learning Theory (COLT)*, pages 144–152. ACM Press, 1992.

K. Brinker. Incorporating diversity in active learning with support vector machines. In T. Fawcett and N. Mishra, editors, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 59–66. AAAI Press, 2003.

K. Brinker. Active learning of label ranking functions. In C. E. Brodley, editor, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 129–136. ACM Press, 2004.

C. Campbell, N. Cristianini, and A. Smola. Query learning with large margin classifiers. In P. Langley, editor, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 111–118. Morgan Kaufmann, 2000.

O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, 2006.

A. A. Collister and O. Lahav. ANNz: estimating photometric redshifts using artificial neural networks. *Publications of the Astronomical Society of the Pacific*, 116(818):345, 2004.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3): 273–297, 1995.

C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh. Sample selection bias correction theory. In N. Bshouty, G. Stoltz, N. Vayatis, and T. Zeugmann, editors, *Algorithmic Learning Theory*, pages 38–53. Springer, 2008.

C. Cortes, Y. Mansour, and M. Mohri. Learning bounds for importance weighting. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 23, pages 442–450, 2010.

S. Dasgupta. Two faces of active learning. *Theoretical Computer Science*, 412 (19):1767–1781, 2011.

S. Dasgupta and D. Hsu. Hierarchical sampling for active learning. In W. W. Cohen, A. McCallum, and S. T. Roweis, editors, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 208–215. ACM Press, 2008.

H. Daume III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, pages 101–126, 2006.

B. Demir and L. Bruzzone. A novel active learning method for support vector regression to estimate biophysical parameters from remotely sensed images. In L. Bruzzone, editor, *Proceedings of SPIE 8537, Image and Signal Processing for Remote Sensing XVIII*, volume 8537, page 85370L. International Society for Optics and Photonics, 2012.

I. S. Dhillon. *A New $O(n^2)$ Algorithm for the Symmetric Tridiagonal Eigenvalue Eigenvector Problem*. PhD thesis, University of California at Berkeley, 1998.

S. Dieleman, K. W. Willett, and J. Dambre. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly Notices of the Royal Astronomical Society*, 450:1441–1459, 2015.

J. Du and Z. Cai. Modelling class noise with symmetric and asymmetric distributions. In B. Bonet and S. Koenig, editors, *AAAI Conference on Artificial Intelligence (AAAI)*, pages 2589–2595, 2015.

B. Frenay and M. Verleysen. Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5): 845–869, 2014.

J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977.

Galaxy Zoo. Image AGZ0008187. `https://talk.galaxyzoo.org/#/subjects/AGZ0008187`, 2012.

R. Ganti and A. Gray. Upal: Unbiased pool based active learning. In N. D. Lawrence and M. A. Girolami, editors, *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS): JMLR Workshop and Conference Proceedings*, volume 12, pages 422–431, 2012.

F. Gieseke, J. Heinermann, C. Oancea, and C. Igel. Buffer k-d trees: Processing massive nearest neighbor queries on GPUs. In E. P. Xing and T. Jebara, editors, *Proceedings of the International Conference on Machine Learning (ICML): JMLR Workshop and Conference Proceedings*, volume 32, pages 172–180, 2014.

T. Glasmachers and C. Igel. Second-order SMO improves SVM online and active learning. *Neural Computation*, 20(2):374–382, 2008.

G. H. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, second edition, 1989.

I. Guyon, G. Cawley, G. Dror, and V. Lemaire. Results of the active learning challenge. In I. Guyon, G. C. Cawley, G. Dror, V. Lemaire, and A. R. Statnikov, editors, *Workshop on Active Learning and Experimental Design: JMLR Workshop and Conference Proceedings*, volume 16, pages 19–45, 2011.

N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

S. Hanneke. *Theoretical foundations of active learning*. ProQuest, 2009.

T. Hastie and R. Tibshirani. Classification by pairwise coupling. *Annals of Statistics*, 26(2):451–471, 1998.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2009.

C.-H. Ho, M.-H. Tsai, and C.-J. Lin. Active learning and experimental design with SVMs. In I. Guyon, G. C. Cawley, G. Dror, V. Lemaire, and A. R. Statnikov, editors, *Workshop on Active Learning and Experimental Design: JMLR Workshop and Conference Proceedings*, volume 16, pages 71–84, 2011.

S. Hoi, R. Jin, J. Zhu, and M. Lyu. Semi-supervised SVM batch mode active learning for image retrieval. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–7. IEEE Press, 2008.

J. Huang, A. J. Smola, A. Gretton, K. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 19, pages 601–608. Curran Associates, Inc., 2007.

S.-J. Huang, R. Jin, and Z.-H. Zhou. Active learning by querying informative and representative examples. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 23, pages 892–900. Curran Associates, Inc., 2010.

R. Izbicki, A. B. Lee, and C. M. Schafer. High-dimensional density ratio estimation with extensions to approximate likelihood computation. In S. Kaski and J. Corander, editors, *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS): JMLR Workshop and Conference Proceedings*, volume 33, pages 420–429, 2014.

R. Izbicki, A. B. Lee, and P. E. Freeman. Photo-z estimation: An example of nonparametric conditional density estimation under selection bias. *arXiv preprint arXiv:1604.01339*, 2016.

Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

J. Jiang. A literature survey on domain adaptation of statistical classifiers. `http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/`, 2008.

T. Kanamori, S. Hido, and M. Sugiyama. A least-squares approach to direct importance estimation. *Journal of Machine Learning Research*, 10:1391–1445, 2009.

T. Kanamori, T. Suzuki, and M. Sugiyama. Statistical analysis of kernel-based least-squares density-ratio estimation. *Machine Learning*, 86(3):335–367, 2012.

M. Kojima, S. Mizuno, and A. Yoshise. A polynomial-time algorithm for a class of linear complementarity problems. *Mathematical Programming*, 44:1–26, 1989.

J. Kremer, K. Steenstrup Pedersen, and C. Igel. Active learning with support vector machines. *Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery*, 4(4):313–326, 2014.

J. Kremer, F. Gieseke, K. Steenstrup Pedersen, and C. Igel. Nearest neighbor density ratio estimation for large-scale applications in astronomy. *Astronomy and Computing*, 12:67–72, 2015.

J. Kremer, F. Sha, and C. Igel. Active label correction for class-conditional noise. Submitted, 2016a.

J. Kremer, K. Stensbo-Smidt, F. Gieseke, K. Steenstrup Pedersen, and C. Igel. Big universe, big data: Machine learning and image analysis for astronomy. Submitted, 2016b.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

D. Lewis and W. Gale. A sequential algorithm for training text classifiers. In W. B. Croft and C. J. van Rijsbergen, editors, *Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR)*, pages 3–12. ACM Press, 1994a.

D. Lewis and W. Gale. Training text classifiers by uncertainty sampling. In W. B. Croft and C. J. van Rijsbergen, editors, *Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR)*, pages 3–12. ACM Press, 1994b.

C.-L. Li, C.-S. Ferng, and H.-T. Lin. Active learning with hinted support vector machine. In S. C. H. Hoi and W. L. Buntine, editors, *Proceedings of the Asian Conference on Machine Learning (ACML): JMLR Workshop and Conference Proceedings*, volume 25, pages 221–235, 2012.

M. Lima, C. E. Cunha, H. Oyaizu, J. Frieman, H. Lin, and E. S. Sheldon. Estimating the redshift distribution of photometric galaxy samples. *Monthly Notices of the Royal Astronomical Society*, 390(1):118–130, 2008.

T. Liu and D. Tao. Classification with noisy labels by importance reweighting. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 38(3): 47–461, 2016.

M. Loog. Nearest neighbor-based importance weighting. In *Proceedings of the International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE Press, 2012.

T. Luo, K. Kramer, D. Goldgof, L. Hall, S. Samson, A. Remsen, and T. Hopkins. Active learning to recognize multiple types of plankton. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, volume 3, pages 478–481. IEEE Press, 2004.

A. Mammone, M. Turchi, and N. Cristianini. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(3):283–289, 2009. doi: 10.1002/wics.49.

A. K. Menon, B. van Rooyen, C. S. Ong, and B. Williamson. Learning from corrupted binary labels via class-probability estimation. In F. R. Bach and D. M. Blei, editors, *Proceedings of the International Conference on Machine Learning (ICML): JMLR Workshop and Conference Proceedings*, volume 37, pages 125–134, 2015.

T. M. Mitchell. Generalization as search. *Artificial intelligence*, 18(2):203–226, 1982.

P. Mitra, C. Murthy, and S. Pal. A probabilistic active support vector learning algorithm. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 26(3):413–418, 2004.

D. J. Mortlock, S. J. Warren, B. P. Venemans, M. Patel, P. C. Hewett, R. G. McMahon, C. Simpson, T. Theuns, E. A. Gonzales-Solares, A. Adamson, S. Dye, N. C. Hambly, P. Hirst, M. J. Irwin, E. Kuiper, A. Lawrence, and H. J. A. Rottgering. A luminous quasar at a redshift of z = 7.085. *Nature*, 474 (7353):616–619, 2011.

N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari. Learning with noisy labels. In C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pages 1196–1204. Curran Associates, Inc., 2013.

F. Olsson. A literature survey of active machine learning in the context of natural language processing. Technical report, Swedish Institute of Computer Science, 2009.

K. S. Pedersen, K. Stensbo-Smidt, A. Zirm, and C. Igel. Shape Index Descriptors Applied to Texture-Based Galaxy Analysis. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2440–2447. IEEE Press, 2013. doi: 10.1109/ICCV.2013.303.

J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999a.

J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 12, pages 185–208. MIT Press, 1999b.

J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 12, pages 547–553. MIT Press, 2000.

K. L. Polsterer, P. Zinn, and F. Gieseke. Finding new high-redshift quasars by asking the neighbours. *Monthly Notices of the Royal Astronomical Society*, 428 (1):226–235, 2013.

K. L. Polsterer, F. Gieseke, and C. Igel. Automatic classification of galaxies via machine learning techniques: Parallelized Rotation/Flipping INvariant Kohonen Maps (PINK). In A. R. Taylor and E. Rosolowsky, editors, *Astronomical Data Analysis Software and Systems XXVI*, volume 495 of *ASP Conference Series*, pages 81–86, San Francisco, 2015. Astronomical Society of the Pacific.

J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence. *Dataset Shift in Machine Learning*. MIT Press, 2009.

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

U. Rebbapragada, C. E. Brodley, D. Sulla-Menashe, and M. A. Friedl. Active label correction. In M. J. Zaki, A. Siebes, J. X. Yu, B. Goethals, G. I. Webb, and X. Wu, editors, *Proceedings of the International Conference on Data Mining (ICDM)*, pages 1080–1085. IEEE, 2012.

S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *InternationaI Conference on Learning Representations (ICLR) Workshop*, 2015. URL `http://arxiv.org/abs/1412.6596`.

J. W. Richards, D. L. Starr, H. Brink, A. A. Miller, J. S. Bloom, N. R. Butler, J. B. James, J. P. Long, and J. Rice. Active learning to overcome sample selection bias: Application to photometric variable star classification. *The Astrophysical Journal*, 744(2), 2012.

R. Rifkin and A. Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

E. Robbrecht and D. Berghmans. Automated recognition of coronal mass ejections (CMEs) in near-real-time data. *Astronomy & Astrophysics*, 425:1097–1106, 2004.

N. Roy and A. McCallum. Toward optimal active learning through sampling estimation of error reduction. In C. E. Brodley and A. P. Danyluk, editors, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 441–448, 2001.

S. Salcedo-Sanz, J. L. Rojo-Álvarez, M. Martínez-Ramón, and G. Camps-Valls. Support vector machines in engineering: An overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(3):234–267, 2014.

G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In P. Langley, editor, *Proceedings of the International Conference on Machine Learning (ICML)*, pages 839–846. Morgan Kaufmann, 2000.

B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

B. Settles. *Active Learning*. Morgan & Claypool, 2012.

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

B. Settles, M. Craven, and S. Ray. Multiple-instance active learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 21, pages 1289–1296. Curran Associates, Inc., 2008.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 614–622. ACM Press, 2008.

X. Shi, W. Fan, and J. Ren. Actively transfer domain knowledge. In W. Daelemans, B. Goethals, and K. Morik, editors, *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, pages 342–357. Springer, 2008.

I. Steinwart and A. Christmann. *Support Vector Machines*. Springer, 2008.

K. Stensbo-Smidt, C. Igel, A. Zirm, and K. S. Pedersen. Nearest Neighbour Regression Outperforms Model-based Prediction of Specific Star Formation Rate. In *IEEE International Conference on Big Data*, pages 141–144. IEEE Press, 2013.

K. Stensbo-Smidt, F. Gieseke, C. Igel, A. Zirm, and K. S. Pedersen. Simple, fast and accurate photometric estimation of specific star formation rate. *Monthly Notices of the Royal Astronomical Society*, 2015. URL `http://arxiv.org/abs/1511.05424`. Accepted subject to "moderate revisions".

M. Sugiyama. Active learning for misspecified models. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 18, pages 1305–1312. Curran Associates, Inc., 2005.

M. Sugiyama and K.-R. Müller. Model selection under covariate shift. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pages 235–240. Springer, 2005.

M. Sugiyama, M. Krauledat, and K.-R. Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:985–1005, 2007.

M. Sugiyama, S. Nakajima, H. Kashima, P. V. Bünau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 20, pages 1433–1440. Curran Associates, Inc., 2008.

M. Sugiyama, T. Suzuki, and T. Kanamori. Density ratio estimation: A comprehensive review. In *Statistical Experiment and Its Related Topics*, volume 1703, pages 10–31, 2010a.

M. Sugiyama, I. Takeuchi, T. Suzuki, T. Kanamori, H. Hachiya, and D. Okanohara. Conditional density estimation via least-squares density ratio estimation. In Y. W. Teh and D. M. Titterington, editors, *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS): JMLR Workshop and Conference Proceedings*, volume 9, pages 781–788, 2010b.

S. Sukhbaatar and R. Fergus. Learning from noisy labels with deep neural networks. In *InternationaI Conference on Learning Representations (ICLR) Workshop*, 2015. URL `http://arxiv.org/abs/1406.2080`.

P. D. Tao. Convex analysis approach to D. C. programming: Theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

S. Tong. *Active learning: Theory and applications*. PhD thesis, Stanford University, 2001.

S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proceedings of the International Conference on Multimedia (MM)*, pages 107–118. ACM Press, 2001.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2002.

R. Urner, S. Ben-David, and O. Shamir. Learning from weak teachers. In N. D. Lawrence and M. A. Girolami, editors, *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS): JMLR Workshop and Conference Proceedings*, volume 22, pages 1252–1260, 2012.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, 1998.

K. W. Willett, C. J. Lintott, S. P. Bamford, K. L. Masters, B. D. Simmons, K. R. Casteels, E. M. Edmondson, L. F. Fortson, S. Kaviraj, W. C. Keel, et al. Galaxy Zoo 2: detailed morphological classifications for 304 122 galaxies from the Sloan Digital Sky Survey. *Monthly Notices of the Royal Astronomical Society*, 435(4):2835–2860, 2013.

T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2691–2699. IEEE Press, 2015.

Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang. Representative sampling for text classification using support vector machines. In F. Sebastiani, editor, *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 393–407. Springer, 2003.

H. Yu. SVM selective sampling for ranking with application to data retrieval. In *Proceedings of the International Conference on Knowledge Discovery in Data Mining (SIGKDD)*, pages 354–363. ACM Press, 2005.

B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proceedings of the International Conference on Data Mining (ICDM)*, pages 435–442. IEEE Press, 2003.

X. Zeng and T. R. Martinez. An algorithm for correcting mislabeled data. *Intelligent Data Analysis*, 5(6):491–502, 2001.

L. Zhao, G. Sukthankar, and R. Sukthankar. Incremental relabeling for active learning with noisy crowdsourced annotations. In *IEEE International Conference on Privacy, Security, Risk and Trust (PASSAT) and IEEE International Conference on Social Computing (SocialCom)*, pages 728–733. IEEE Press, 2011.

L. Zhao, G. Sukthankar, and R. Sukthankar. Importance-weighted label prediction for active learning with noisy annotations. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pages 3476–3479. IEEE Press, 2012.