Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence

Active Learning with Model Selection

Alnur Ali
Machine Learning Department
Carnegie Mellon University
alnurali@cmu.edu

Rich Caruana
Microsoft Research
rcaruana@microsoft.com

Ashish Kapoor
Microsoft Research
akapoor@microsoft.com

Abstract

Most active learning methods avoid model selection by training models of one type (SVMs, boosted trees, etc.) using one pre-defined set of model hyperparameters. We propose an algorithm that actively samples data to simultaneously train a *set* of candidate models (different model types and/or different hyperparameters) *and* also select the best model from this set. The algorithm actively samples points for training that are most likely to improve the accuracy of the more promising candidate models, and also samples points for model selection—*all* samples count against the same labeling budget. This exposes a natural trade-off between the focused active sampling that is most effective for training models, and the unbiased sampling that is better for model selection. We empirically demonstrate on six test problems that this algorithm is nearly as effective as an active learning oracle that knows the optimal model in advance.

1 Introduction

In active learning, the goal is to learn a model that has high accuracy on test data by requesting the labels of as few carefully chosen training points as possible. When applying machine learning to a new problem one usually does not know in advance the model type or model complexity that is best for the problem. Unfortunately, most active learning algorithms side-step the important issue of model selection and instead actively sample labels to train a single predefined model. In active learning you only get one chance so this risks training models not well optimized for the task at hand.

Combining model selection with active learning is nontrivial:

- if multiple candidate models are actively trained at the same time, different models will likely benefit from labeling different training points
- actively sampled data is not representative of the natural distribution and thus would yield biased estimates of model accuracy if used for model selection
- if an unbiased sample will be used for model selection, the labels for this validation set should come from the same labeling budget as the training data

The need to collect both an unbiased validation set for model selection and a biased training set for active learning *from the same labeling budget* exposes a trade-off between sampling more training data to improve model accuracy versus having more data to reliably select the best model.

We present an algorithm for Active Learning and Model Selection (ALMS) that actively trains a *set* of models and selects the best model from this set. The algorithm works by requesting the labels of selected points to efficiently train the models, while also requesting the labels of unbiased points to accurately select among the models, with *all* queries counted against a fixed labeling budget. The allocation of points to the training and validation sets is made dynamically by the algorithm during active learning. The unbiased validation data is also used to guide active learning to preferentially select training data for the better performing models. Furthermore, to make maximum use of the labeling budget, the algorithm also uses the labels collected for the unbiased validation set as additional training data *and* to estimate the value of information of candiate points. We experimentally evaluate ALMS against several baselines on real and synthetic data, and show that it is able to train and select models of higher accuracy than traditional active learning on a single model—and performs almost as well as an oracle that knows the optimal regularization parameters in advance.

The rest of this paper is structured as follows. In Section 2, we review related work, outline several potential solutions to the problem of joint active learning and model selection, and describe why they are insufficient. In Section 3, we present our algorithm. In Section 4, we present an empirical evaluation on six test problems. Finally, in Section 5, we conclude.

2 Related Work and Potential Approaches

Active learning can drastically reduce the number of labeled examples required by machine learning algorithms in a variety of real-world scenarios (Dasgupta, Kalai, and Monteleoni 2009), (Yan et al. 2012), (Greiner, Grove, and Roth 2002), (Mazzoni, Wagstaff, and Burl 2006), (Roy and Mccallum 2001), (Tong and Koller 2000), (Beygelzimer et al. 2010). Most work on active learning, however, side-steps the issue of model selection and actively samples labels to train a single model of one type (SVMs, neural nets, boosted trees, etc.) with a single set of hyperparameters (soft margin

parameter, kernel type/width, number of hidden units/layers, tree size, etc.). Unfortunately, failure to select the best model and optimize the model hyperparameters risks training models that are not well suited to the task at hand and that will under or over-fit to the small, biased samples available in active learning. *In the real world, once the labeling budget is used, there is no budget left to actively train more models.*

Select On Biased Training Data One potential approach to the joint active learning and model selection problem is to use the same biased labeled data queried by the active learning procedure for both training and model selection; unfortunately, (Baram, El-Yaniv, and Luz 2003) empirically demonstrate that leave-one-out cross-validation (LOOCV) estimates of the generalization error computed on data labeled by *Uncertainty Sampling* (Lewis and Gale 1994), are highly biased compared to using a random sample from the underlying data distribution—nonetheless, in our empirical evaluation, we compare against the *Query by Committee* algorithm (Seung, Opper, and Sompolinsky 1992) modified to use K-fold cross-validation on the actively sampled data for both model training and selection.

Sample Bias Correction (Sugiyama and Rubens 2008) propose an algorithm for joint active learning and model selection for regression, which relies on a closed-form expression for the LOOCV estimate of a linear regression model that has been "debiased" by means of *importance weighting (IW)* (Sugiyama, Krauledat, and Müller 2007). In IW, a correction factor of $P(\mathbf{x})/Q(\mathbf{x})$, where $P(\mathbf{x})$ is an estimate of the underlying distribution, $Q(\mathbf{x})$ is the biased distribution, and $\mathbf{x} \in \mathbb{R}^d$ is a biased data point, is used (Cortes et al. 2008). Unfortunately, there are issues with this approach. First, a closed-form expression for the LOOCV estimate may not be available for other model types; theoretical bounds on this quantity can exist (Vapnik and Chapelle 2000), but may be too loose to work well in practice. Second, estimates obtained via IW can have high variance, especially when the number of samples is small (Dudík, Langford, and Li 2011; Cortes, Mansour, and Mohri 2010), as in the early stages of active learning. Third, using IW with K-fold CV can introduce additional variance and additional hyperparameters (Huang et al. 2006) that are difficult to deal with in the active learning setting.

Use Unlabeled Pool of Data Another possibility is to use all of the actively learned models to label an unlabeled pool of examples, and then use this pseudo-labeled pool for model selection; this strategy was originally proposed by (Roy and Mccallum 2001) in the single model setting to estimate the value of querying for the label of a new point. Unfortunately, these estimates of model generalization error are highly biased and not well suited for model selection.

Related Work (Sawade et al. 2010) and (Madani, Lizotte, and Greiner 2004) tackle the related problem of actively querying for labels in order to efficiently estimate the generalization error of *already* trained models; in this paper, we consider the more general problem of actively querying labels to both train *and* select from multiple models. (Kapoor and Horvitz 2009) tackle the related, but distinct, problem of budgeted feature vs. label acquisition.

3 Our Approach: ALMS

At a high-level, our algorithm for joint active learning and model selection (ALMS) works as follows:

1. ALMS takes as inputs a set of candidate models $\mathcal{M} = \{M_1, \ldots, M_m\}$, and a pool of unlabeled examples $\mathcal{P} = \{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^p$; for instance, $\mathcal{M}$ could comprise ℓ_2-regularized logistic regression models with different regularization parameters λ, RBF kernel SVMs with different regularization parameters C and kernel bandwidths γ, neural networks with different numbers of hidden layers/units, or even a set containing all of these models.

2. On each round $t \in \{1, \ldots, T\}$ of active learning, ALMS samples the point and its label $(\mathbf{x}^*, y^*)$ from the pool that will increase the expected accuracy of the *full system* most. System accuracy can be increased either by increasing the expected accuracy of the models in $\mathcal{M}$ by adding the sampled point $(\mathbf{x}^*, y^*)$ to the actively sampled training set[1] $\mathcal{T}_t$, or by increasing the probability of *selecting* the best model $M^* \in \mathcal{M}$ by adding the sampled point to the *unbiased* validation set $\mathcal{V}_t$.

3. At the end of active learning, ALMS outputs the model $M^* \in \mathcal{M}$ it expects will perform best on future test data after re-training that model on *all data* in the union of the train and validation sets.

The heart of ALMS (step 2) computes the *value of information for training* $\mathrm{VOI}_{\mathcal{T},M}(\mathbf{x})$, and the *value of information for model selection* $\mathrm{VOI}_{\mathcal{V},M}(\mathbf{x})$ for each point $\mathbf{x} \in \mathcal{P}$, in order to decide which point $\mathbf{x}^* \in \mathcal{P}$ to sample, and whether to place it in the training set $\mathcal{T}_t$ or the validation set $\mathcal{V}_t$: unlike traditional approaches to active learning where only the value of information for training *a single model* is needed, in active learning with model selection, the value of information for training must be computed separately *for each model* in $\mathcal{M}$, and then combined so that lower accuracy models have less influence over data selection—this introduces a cooperation/competition trade-off, as different models may benefit from training on different points in $\mathcal{P}$.

The overall *value of training* $\mathrm{VOI}_{\mathcal{T}}$ and *value of model selection* $\mathrm{VOI}_{\mathcal{V}}$ are computed by taking the max over all points $\mathbf{x} \in \mathcal{P}$; if $\mathrm{VOI}_{\mathcal{T}}$ is larger than $\mathrm{VOI}_{\mathcal{V}}$ then the maximizer $\mathbf{x}^*$ is sampled and placed in the training set $\mathcal{T}_t$, otherwise a point is sampled *at random* (to maintain the unbiasedness of the validation set[2]) from $\mathcal{P}$ and placed in the validation set $\mathcal{V}_t$. ALMS reduces to the traditional active learning algorithm of (Roy and Mccallum 2001) when $\mathcal{M}$ contains only one model and $\mathcal{P}$ is used in place of $\mathcal{V}_t$.

[1] $\mathcal{T}_t$ and $\mathcal{V}_t$ are indexed by t because they can grow on each round of active learning.

[2] Placing the maximizer $\mathbf{x}^*$ in $\mathcal{V}_t$ would make $\mathcal{V}_t$ biased because $\mathbf{x}^*$ is selectively sampled, something we observed in preliminary experiments.

In symbols:

$$\text{VOI}_{\mathcal{T}}\text{ (value of training)} = \max_{\mathbf{x}\in\mathcal{P}} \mathrm{E}[\text{VOI}_{\mathcal{T},M}(\mathbf{x})] \quad (1)$$

$$\text{VOI}_{\mathcal{V}}\text{ (value of model sel.)} = \max_{\mathbf{x}\in\mathcal{P}} \mathrm{E}[\text{VOI}_{\mathcal{V},M}(\mathbf{x})] \quad (2)$$

$$\text{if } \text{VOI}_{\mathcal{T}} > \text{VOI}_{\mathcal{V}} \quad \mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathcal{P}} \text{VOI}_{\mathcal{T}}(\mathbf{x})$$

$$\text{else} \quad \text{draw } \mathbf{x}^* \sim \mathcal{P}.$$

The expectations in Equations 1 and 2 are taken using a distribution over the model space $\mathcal{M}$: we discuss computing this distribution in Section 3.1. The methods for computing the value of information for training and model selection are described in Sections 3.2 and 3.3.

3.1 Computing a Distribution over Models

Intuitively, the probability that a model $M \in \mathcal{M}$ is the best model is proportional to its accuracy on unseen test data; fortunately, ALMS has access to a validation set $\mathcal{V}_t$ it can use to obtain unbiased estimates of a model's accuracy.

On each round t of active learning, ALMS computes the probability of each model $P_t(M)$ being the best by applying a softmax (Miller and Yan 1999), (Sutton and Barto 1998), to each model's error rate as measured on $\mathcal{V}_t$:

$$P_t(M) = 1 - \frac{\exp(\text{error}(M))}{\sum_{M'\in\mathcal{M}} \exp(\text{error}(M'))}, \quad (3)$$

where $\text{error}(M) = \frac{1}{|\mathcal{V}_t|}\sum_{(\mathbf{x},y)\in\mathcal{V}_t}^{|\mathcal{V}_t|} \text{loss}(M,\mathbf{x},y)$, and $\text{loss}(M,\mathbf{x},y)$ computes the loss between M's output on $\mathbf{x}$ and its true label y (e.g., the 0/1 loss, log loss, or squared loss)[3]; here, M is trained on $\mathcal{T}_t$.

3.2 Computing VOI for Training

The value of information for training $\text{VOI}_{\mathcal{T},M}(\mathbf{x})$ measures the improvement in model M's accuracy after training on a point $\mathbf{x} \in \mathcal{P}$. Because ALMS does not have access to $\mathbf{x}$'s true label y before sampling it, ALMS estimates $\tilde{y}$ by averaging the output of each model $M \in \mathcal{M}$ at $\mathbf{x}$ and thresholding:

$$\tilde{y} = I(\bar{y} > t) \qquad \bar{y} = \mathrm{E}[\mathrm{E}[y|\mathbf{x}, M]] \quad (4)$$

where t is a threshold[4], the outer expectation is taken w.r.t. M (Section 3.1), and the inner expectation is taken w.r.t. the output space $\mathcal{Y}$.

ALMS could then estimate the improvement in the entire system's accuracy by training each model on $\mathcal{T}_t \cup (\mathbf{x}, \tilde{y})$, estimating each model's accuracy on the unbiased validation set $\mathcal{V}_t$, and then taking a weighted average of these estimates using $P_t(M)$. Unfortunately, because ALMS already uses $\mathcal{V}_t$ to compute the model probabilities $P_t(M)$ (Section 3.1), using the same data to estimate the model probabilities and to also estimate the model accuracies would introduce bias.

Instead, we employ leave-2-out cross-validation (L2OCV) to simultaneously estimate $P_t(M)$ and $\text{VOI}_{\mathcal{T},M}(\mathbf{x})$ without bias from $\mathcal{V}_t$, while also enabling ALMS to make maximal use of the small sample sizes that arise in active learning[5]. To do this, ALMS first trains all models on the train set plus the candidate point *plus all validation data except two held-out points:* $\{\mathcal{T}_t \cup (\mathbf{x}, \tilde{y})\} \setminus \{(\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j)\}$. Next, ALMS computes a distribution over models using only the single left-out validation point $(\mathbf{x}_i, y_i)$: denote this $P_t^i(M)$. Then, ALMS computes the loss of all models on the other left-out validation point $(\mathbf{x}_j, y_j)$. ALMS repeats these steps for all $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j) \in \mathcal{V}_t$; the final estimate of $P_t(M)$ is the average $P_t^i(M)$ over all $(\mathbf{x}_i, y_i) \in \mathcal{V}_t$, and the final estimate for $\text{VOI}_{\mathcal{T},M}(\mathbf{x})$ for each M is the average loss of M over all $(\mathbf{x}_j, y_j)$. This process is depicted in Figure 1.

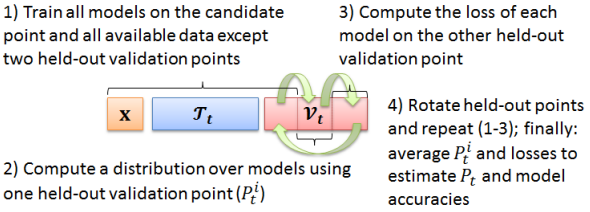

Figure 1: The steps involved in computing the value of information for training $\text{VOI}_{\mathcal{T},M}(\mathbf{x})$.

ALMS computes $\mathrm{E}[\text{VOI}_{\mathcal{T},M}(\mathbf{x})]$ (Equation 1) by taking a weighted average of the estimates of $\text{VOI}_{\mathcal{T},M}(\mathbf{x})$ using the estimates of $P_t(M)$—which can be written concisely as a quadratic form:

$$\mathrm{E}[\text{VOI}_{\mathcal{T},M}(\mathbf{x})] = \mathbf{s}^\top \mathbf{P}\mathbf{L}\mathbf{s}, \quad (5)$$

where $v = |\mathcal{V}_t|, m = |\mathcal{M}|, \mathbf{s} \in \mathbb{R}^v = (1/v, \ldots, 1/v), \mathbf{P} \in \mathbb{R}^{v\times m}$ has P_{ij} set to $P_t^i(M_j)$ the probability of model M_j computed on only the left-out point $(\mathbf{x}_i, y_i) \in \mathcal{V}_t$, and $\mathbf{L} \in \mathbb{R}^{m\times v}$ has L_{ij} set to the loss of model M_i on the other left-out point $(\mathbf{x}_j, y_j) \in \mathcal{V}_t$.

3.3 Computing VOI for Model Selection

Computing the value of information for model selection $\text{VOI}_{\mathcal{V},M}(\mathbf{x})$ proceeds almost exactly as in Section 3.2—the two differences are: (a) models are not trained on the two left-out validation points *or the candidate point* (b) L2OCV is executed with $\mathcal{V}_t \cup (\mathbf{x}, \tilde{y})$ (without using the candidate point for training), making $\mathbf{s} \in \mathbb{R}^{v+1} = (\frac{1}{v+1}, \ldots, \frac{1}{v+1})$, $\mathbf{P} \in \mathbb{R}^{(v+1)\times m}$, and $\mathbf{L} \in \mathbb{R}^{m\times(v+1)}$.

ALMS is specified in Algorithm 1. To summarize informally: a candidate point in the pool has high value for training if it increases the expected accuracy (the entries in $\mathbf{L}$ in Equation 5) of the most promising models (high values in $\mathbf{P}$

[3] In our experiments, we use the lower endpoint of a 95% confidence interval instead of the raw $\text{error}(M)$, which compensates for small sample size early in active learning.

[4] When $\mathcal{Y} = \{0, 1\} : t = 1/2$ and $\bar{y} = \sum_{M\in\mathcal{M}} P_t(M)P(y = 1|\mathbf{x}, M)$.

[5] L2OCV is a form of nested cross validation: hold one point out, cycle through holding out each of the remaining points, then repeat with a new held-out point at the 1st level until all pairs of points have been held out.

Algorithm 1 ALMS.

function ALMS(set of candidate models $\mathcal{M}$, pool of unlabeled examples $\mathcal{P}$)
  initialize $P_0(M)$ (e.g., uniformly)
  for $t = 1, \ldots, T$ rounds **do**
    train all models on $\mathcal{T}_t$, compute $P_t(M)$ (Eq. 3)
    for all $(\mathbf{x}, y) \in \mathcal{P}$ **do**
      compute $\mathbf{x}$'s estimated label $\tilde{y}$ (Eq. 4)
      $\mathrm{E}[\mathrm{VOI}_{\mathcal{T},M}(\mathbf{x})]$ = COMPUTEVOI($\mathcal{T}_t \cup (\mathbf{x}, \tilde{y}), \mathcal{V}_t$) (Eq. 1)
      $\mathrm{E}[\mathrm{VOI}_{\mathcal{V},M}(\mathbf{x})]$ = COMPUTEVOI($\mathcal{T}_t, \mathcal{V}_t \cup (\mathbf{x}, \tilde{y})$) (Eq. 2)
    end for
    $\mathrm{VOI}_{\mathcal{T}} = \max_{\mathbf{x} \in \mathcal{P}} \mathrm{E}[\mathrm{VOI}_{\mathcal{T},M}(\mathbf{x})]$
    $\mathrm{VOI}_{\mathcal{V}} = \max_{\mathbf{x} \in \mathcal{P}} \mathrm{E}[\mathrm{VOI}_{\mathcal{V},M}(\mathbf{x})]$
    if $\mathrm{VOI}_{\mathcal{T}} > \mathrm{VOI}_{\mathcal{V}}$ **then**
      $\mathbf{x}^* = \arg\max_{\mathbf{x} \in \mathcal{P}} \mathrm{VOI}_{\mathcal{T}}(\mathbf{x})$
      $\mathcal{T}_t = \mathcal{T}_t \cup (\mathbf{x}^*, y^*)$
    else
      draw $\mathbf{x}^* \sim \mathcal{P}$
      $\mathcal{V}_t = \mathcal{V}_t \cup (\mathbf{x}^*, y^*)$
    end if
  end for
  select the best model $M^* \in \mathcal{M}$ on $\mathcal{V}_T$
  train M^* on all sampled data $\mathcal{T}_T \cup \mathcal{V}_T$
  Return M^*
end function
function COMPUTEVOI($\mathcal{T}_t, \mathcal{V}_t$)
  for all $(\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j) \in \mathcal{V}_t$ **do**
    train all models on $\{\mathcal{T}_t \cup \mathcal{V}_t\} \setminus \{(\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j)\}$
    compute $P_{i:}$ (i.e., a distribution over models using $(\mathbf{x}_i, y_i)$)
    compute $L_{:j}$ (i.e., $\mathrm{loss}(M, \mathbf{x}_j, y_j),\ \forall M \in \mathcal{M}$)
  end for
  Return $\mathbf{s}^\top \mathbf{P} \mathbf{L} \mathbf{s}$ (Eq. 5)
end function

in Equation 5), while a point has high value for model selection if it increases the probability of selecting models with high expected accuracy. Where computational cost is an issue, ALMS can be sped up without noticeable degradation in accuracy by incrementally retraining models (e.g., logistic regression, SVMs, or naive Bayes) and by subsampling from the $\mathcal{O}(|\mathcal{V}_t|^2)$ L2OCV iterations or from the pool $\mathcal{P}$.

4 Results

We compare ALMS to four baselines on six binary classification problems. Five of the experiments use ℓ_2 regularized logistic regression (regularization parameter $\lambda \in \{2^{-10}, \ldots, 2^{10}\}$), and one of the experiments uses RBF SVMs ($C \in \{10^{-2}, \ldots, 10^2\} \times \gamma \in \{10^{-2}, \ldots, 10^2\}$). All results average over 100 trials.

4.1 Baselines

Passive Learning Passive learning labels a point drawn at random from the pool, and uses K-fold CV on all the labeled data to train and then select the best model in the set of candidate models $\mathcal{M}$ on each round. This strawman illustrates the gap between active and random sampling.

Query by Committee We modified the standard Query by Committee (QBC) (Seung, Opper, and Sompolinsky 1992) algorithm to carry out both model selection and active learning on the set of models $\mathcal{M}$. K-fold CV is done on the biased, actively sampled data to choose the best model. The estimates of model accuracy from K-fold CV are used to weight votes of committee members on which point to sample next; no unbiased validation set $\mathcal{V}_t$ is used.

Uncertainty Sampling We also compare to *Uncertainty Sampling* (Lewis and Gale 1994) applied to a single model, with $\lambda = 1$ for logistic regression, and $C = 1$ and $\gamma = 1$ for SVMs, common default regularization parameters for normalized data; this baseline illustrates the risk of not optimizing model complexity parameters to the task at hand.

Oracle In addition to these three baselines, we also ran an oracle to show the best that could be achieved with Uncertainty Sampling if one knew the optimal regularization parameters in advance. For the oracle we run Uncertainty Sampling to completion with each set of regularization parameters and then pick the model with the best AUC.

4.2 Probabilities and Calibration

As with (Roy and Mccallum 2001), the algorithms we explore use log loss to avoid the coarseness of 0/1 loss measured on small sample sizes. Log loss requires models to predict probabilities. Logistic regression yields well-calibrated probabilities, but SVMs do not. For the experiment with SVMs, we calibrate models with Platt's method (Platt 1999): in the SVM experiment, QBC and Uncertainty Sampling must use the biased actively sampled training set for calibration, whereas ALMS can use its unbiased validation set.

4.3 Empirical Results

We experimented with six data sets. The 1st column in Figure 2 shows learning curves for ALMS and the four baseline methods vs. the number of sampled labels (rounds of active learning). With both logistic regression and RBF SVMs, ALMS dominates passive learning with cross validation, Uncertainty Sampling on a single model, and QBC on the set of models $\mathcal{M}$. ALMS has comparable accuracy to the other methods when there are very few labels (< 10), but pulls away from the other methods when there are 20 or more labels. On most problems ALMS is competitive with the oracle, and, surprisingly, on two problems ALMS outperforms the oracle. It is possible for ALMS to outperform the oracle because (a) Uncertainty Sampling may not always be the best active learning strategy; (b) ALMS has access to an unbiased validation set for calibration; and (c) by maintaining a set of models, ALMS represents a different balance between exploration and exploitation.

The 2nd column in Figure 2 shows the allocation of labels by ALMS to the train and validation sets. ALMS begins by allocating similar numbers of labels to the train and validation sets because train data is needed to train good

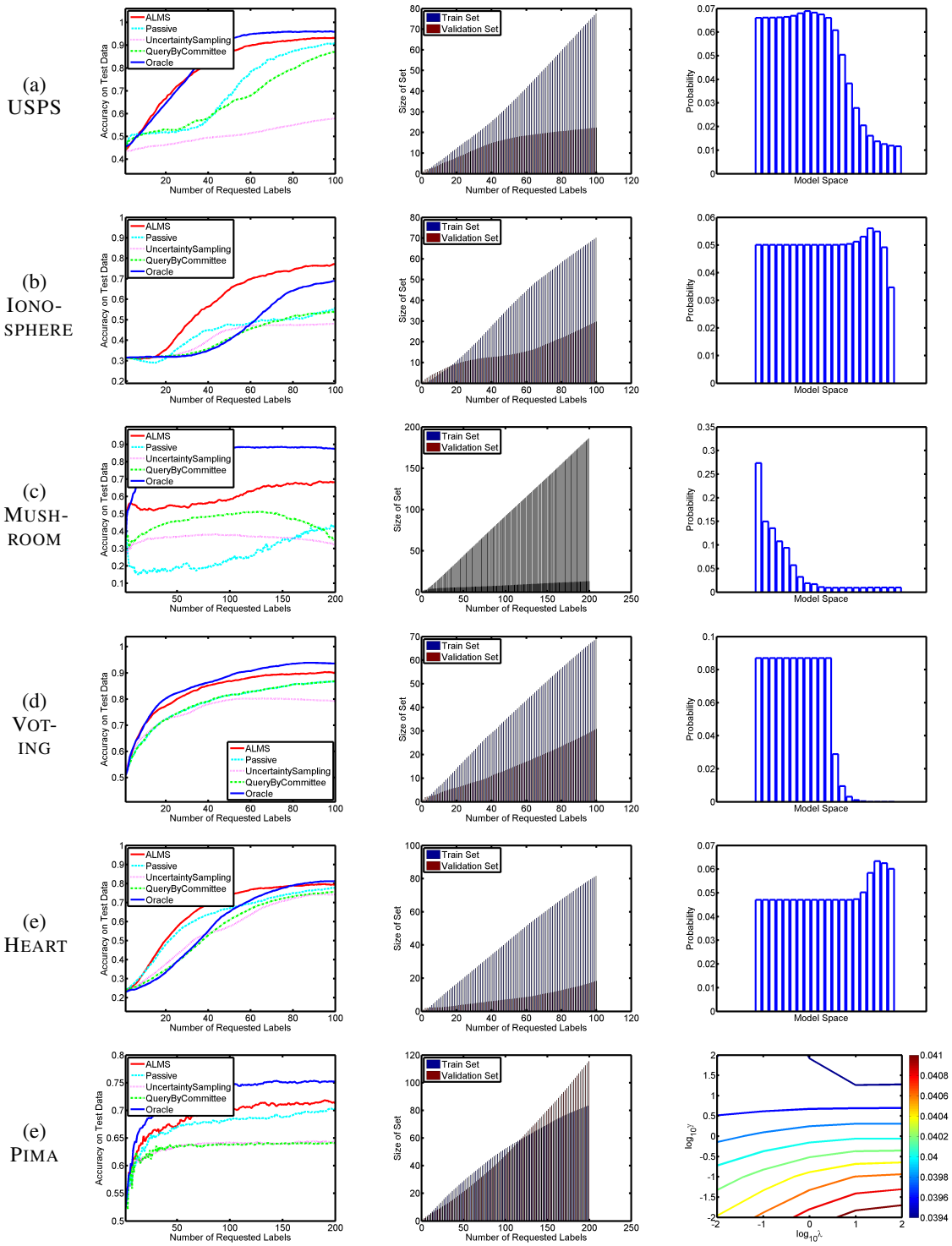

Figure 2: Learning curves (left), allocation of points to the train and validation sets (middle), and final model probabilities (right) for ALMS on six data sets. Error bars omitted to reduce clutter; all results average over 100 trials.

Table 1: Average area under the learning curves for the six data sets. Statistically significant differences between ALMS and all other algorithms (except Oracle) according to a Student's t-test ($\alpha = 0.05$, Bonferroni correction) are indicated in **bold**.

DATA	PASSIVE	UNCERT	QBC	ALMS	*Oracle*
USPS	0.69	0.51	0.66	**0.80**	*0.83*
HEART	0.62	0.56	0.56	**0.66**	*0.59*
IONO	0.44	0.41	0.42	**0.59**	*0.46*
MUSH	0.27	0.36	0.46	**0.60**	*0.84*
VOTE	0.78	0.76	0.78	**0.83**	*0.86*
PIMA	0.67	0.63	0.63	**0.69**	*0.73*
MEAN	0.58	0.54	0.59	**0.70**	*0.72*

models, and validation data is needed to select the better models. This behavior is not hard coded but arises naturally from the algorithm. As active learning progresses, however, ALMS allocates different amounts of data to the train and validation sets on each problem. On VOTING, ALMS allocates roughly equal amounts of data to the train and validation sets, but on MUSHROOM, ALMS allocates far more data to the train set than to the validation set. On average, across the six problems, ALMS allocates about one third of labels to validation and about two thirds of labels to training. Once again, this behavior is not hard coded and emerges naturally from the values of information computed by ALMS for adding labels to the train and validation sets.

On some problems such as USPS and IONOSPHERE, the fraction of data allocated to the validation set changes in interesting ways during learning. On USPS, allocation to the validation set begins to fall off after about 50 rounds of learning. On IONOSPHERE, allocation to the validation set initially is high, then tapers off after about 30 rounds, but then begins to pick up again after 60 rounds. We do not yet fully understand the cause of these changes but suspect it has to do with abrupt changes in the (estimated) quality of different models during learning.

The 3rd column in Figure 2 shows the probabilities assigned to the different models (different hyperparameters) at the end of active learning. Models in the left side (logistic regression) or lower left hand corner (RBF SVMs) of each graph are more complex (less regularized), with regularization increasing (complexity decreasing) as we move to the right (or up). By looking at the graphs across problems it is evident that no one model complexity is appropriate for all problems: IONOSPHERE and HEART favor models with high regularization, USPS and VOTING work well with models of intermediate regularization, and MUSHROOM strongly favors more complex models with little regularization.[6] The RBF SVM results on PIMA are interesting because learning clearly favors models in the lower right-hand corner that have high ℓ_2 regularization, but combined with small kernel width. The results on the six problems clearly demonstrate the value of combining model selection with active learning.

Table 1 shows the area under the curve for three baselines, ALMS, and the Oracle on the six test problems. ALMS has statistically significantly higher area under the curve on all six problems compared to the three baseline methods and yields almost 20% larger AUC than the next best method (QBC). Not only is ALMS the best method on average, it also is the best method on each problem. Compared to the oracle which knows the best regularization parameters prior to active learning, on average ALMS is only 4% worse, and on two of the problems outperforms the oracle. On four of the six test problems ALMS converged to the same models as the Oracle. We do not expect ALMS to always converge to the oracle model with 100 samples because earlier in training it must hedge its bets across multiple models.

4.4 Mushroom Data Set

The results for MUSHROOM are particularly interesting. On MUSHROOM, the most complex model with the least regularization is strongly favored. Presumably because it takes more data to train complex models, on MUSHROOM, ALMS allocates the majority of its labels to the active learning training set and expends relatively few labels on the validation set. The allocation of labels to the train and validation sets on MUSHROOM are the largest skew we have seen.

4.5 Pima Data Set

With RBF SVMs on the PIMA dataset, ALMS (Figure 2(e)) allocates more points to the validation set than to the train set than it does on the other problems. We suspect ALMS selects more validation data on this problem because the SVMs require data for an explicit calibration step and the value of information calculated by ALMS reflects this.

4.6 USPS Data Set

The task in USPS is to distinguish hand-written digits 5 from 8. Figure 3 shows the entropy of the distribution over models. As active learning progresses, entropy continues to decrease indicating that ALMS is converging on a subset of the models.

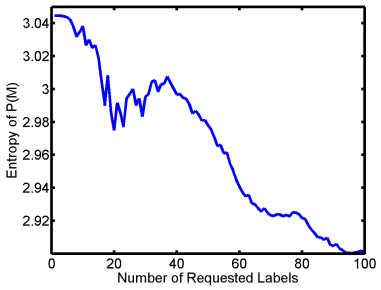

Figure 3: USPS data set. The entropy of the distribution over models, maintained by ALMS, as a function of time.

[6]The graph of model probabilities for MUSHROOM suggests that we probably did not consider a wide enough range of regularization parameters and should have included models with even less regularization.

5 Conclusion

In this paper we proposed an algorithm that requests labels for training points actively selected from an unlabeled pool, and also requests labels for randomly sampled points from the pool for model selection. The algorithm outputs a single model, from a set of candidate models, that has high accuracy on test data, while requesting fewer total labels than several baselines. Moreover, it performs almost as well as an active learning oracle that knows the optimal regularization parameters in advance. An interesting direction for future research is to investigate how to mitigate the downsides of importance weighting with small samples so that the need to sample an unbiased validation set for model selection can be reduced or eliminated.

References

Baram, Y.; El-Yaniv, R.; and Luz, K. 2003. Online choice of active learning algorithms. In *ICML*, 19–26.

Beygelzimer, A.; Hsu, D.; Langford, J.; and Tong, Z. 2010. Agnostic active learning without constraints. In Lafferty, J.; Williams, C. K. I.; Shawe-Taylor, J.; Zemel, R.; and Culotta, A., eds., *Advances in Neural Information Processing Systems 23*. 199–207.

Cortes, C.; Mohri, M.; Riley, M.; and Rostamizadeh, A. 2008. Sample selection bias correction theory. In *Proceedings of the 19th international conference on Algorithmic Learning Theory*, ALT '08, 38–53. Berlin, Heidelberg: Springer-Verlag.

Cortes, C.; Mansour, Y.; and Mohri, M. 2010. Learning bounds for importance weighting. In Lafferty, J.; Williams, C. K. I.; Shawe-Taylor, J.; Zemel, R.; and Culotta, A., eds., *Advances in Neural Information Processing Systems 23*. 442–450.

Dasgupta, S.; Kalai, A. T.; and Monteleoni, C. 2009. Analysis of perceptron-based active learning. *J. Mach. Learn. Res.* 10:281–299.

Dudík, M.; Langford, J.; and Li, L. 2011. Doubly robust policy evaluation and learning. In *ICML*, 1097–1104.

Greiner, R.; Grove, A. J.; and Roth, D. 2002. Learning cost-sensitive active classifiers. *Artif. Intell.* 139(2):137–174.

Huang, J.; Smola, A. J.; Gretton, A.; Borgwardt, K. M.; and Schlkopf, B. 2006. Correcting sample selection bias by unlabeled data. In Schlkopf, B.; Platt, J.; and Hoffman, T., eds., *NIPS*, 601–608. MIT Press.

Kapoor, A., and Horvitz, E. 2009. Breaking boundaries between induction time and diagnosis time active information acquisition. In *NIPS*, 898–906.

Lewis, D. D., and Gale, W. A. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '94, 3–12. New York, NY, USA: Springer-Verlag New York, Inc.

Madani, O.; Lizotte, D. J.; and Greiner, R. 2004. Active model selection. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, UAI '04, 357–365. Arlington, Virginia, United States: AUAI Press.

Mazzoni, D.; Wagstaff, K.; and Burl, M. C. 2006. Active learning with irrelevant examples. In *ECML*, 695–702.

Miller, D., and Yan, L. 1999. Critic-driven ensemble classification. *Trans. Sig. Proc.* 47(10):2833–2844.

Platt, J. C. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, 61–74. MIT Press.

Roy, N., and Mccallum, A. 2001. Toward optimal active learning through sampling estimation of error reduction. In *In Proc. 18th International Conf. on Machine Learning*, 441–448. Morgan Kaufmann.

Sawade, C.; Landwehr, N.; Bickel, S.; and Scheffer, T. 2010. Active risk estimation. In *ICML*, 951–958.

Seung, H. S.; Opper, M.; and Sompolinsky, H. 1992. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, COLT '92, 287–294. New York, NY, USA: ACM.

Sugiyama, M., and Rubens, N. 2008. Active learning with model selection in linear regression.

Sugiyama, M.; Krauledat, M.; and Müller, K.-R. 2007. Covariate shift adaptation by importance weighted cross validation. *J. Mach. Learn. Res.* 8:985–1005.

Sutton, R., and Barto, A. 1998. *Reinforcement Learning: An Introduction*. A Bradford book. Bradford Book.

Tong, S., and Koller, D. 2000. Support vector machine active learning with applications to text classification. In *Journal of Machine Learning Research*, 999–1006.

Vapnik, V., and Chapelle, O. 2000. Bounds on error expectation for support vector machines. *Neural Comput.* 12(9):2013–2036.

Yan, Y.; Rosales, R.; Fung, G.; Farooq, F.; Rao, B.; and Dy, J. G. 2012. Active learning from multiple knowledge sources. In *AISTATS*, 1350–1357.