

Can Active Learning Experience Be Transferred?

Hong-Min Chu

Department of Computer Science and
Information Engineering,
National Taiwan University
E-mail: r04922031@csie.ntu.edu.tw

Hsuan-Tien Lin

Department of Computer Science and
Information Engineering,
National Taiwan University
E-mail: htlin@csie.ntu.edu.tw

Abstract—

Active learning is an important machine learning problem in reducing the human labeling effort. Current active learning strategies are designed from human knowledge, and are applied on each dataset in an immutable manner. In other words, experience about the usefulness of strategies cannot be updated and transferred to improve active learning on other datasets. This paper initiates a pioneering study on whether active learning experience can be transferred. We first propose a novel active learning model that linearly aggregates existing strategies. The linear weights can then be used to represent the active learning experience. We equip the model with the popular linear upper-confidence-bound (LinUCB) algorithm for contextual bandit to update the weights. Finally, we extend our model to transfer the experience across datasets with the technique of biased regularization. Empirical studies demonstrate that the learned experience not only is competitive with existing strategies on most single datasets, but also can be transferred across datasets to improve the performance on future learning tasks.

I. INTRODUCTION

In many machine learning applications, high-quality labels are costly to obtain [1], [2]. Active learning is a machine learning scenario that tries to reduce the labeling cost while still maintaining the performance of learned models by asking key labeling questions [3]. Most current active learning algorithms are based on human knowledge about how to ask questions, and the knowledge is applied immutably on every dataset when conducting active learning. A recent work [4] argued that any single active learning algorithm based on immutable human knowledge is unlikely to perform well on all datasets, and hence proposed to adaptively learn a probabilistic blending of a set of human-designed active learning algorithms. The blending is learned within a single dataset via connecting with multi-armed bandit learning. Given the possibility to learn a decent blending of different pieces of human knowledge within a single dataset, our key thought is: can the learned experience be transferred to other datasets to improve the performance of active learning?

Our thought is related to how human beings learn to ask questions in real life. We do not just learn to ask questions within a single learning task; we instead accumulate experience in question-asking in past and current learning tasks and transfer the experience to future learning tasks. There

are setups in machine learning that study how experience can be transferred to future tasks. The simplest setup is transfer learning [5], or inductive transfer. Transfer learning is about accumulating experience from one or several source tasks and applying the experience to a related target task. Several attempts have been made in previous studies to improve the performance of active learning with transfer learning [6]–[8]. However, all the algorithms proposed in these studies aim to transfer the experience of supervised or semi-supervised learning from the source tasks to the target task, and do not transfer the experience of active learning (question-asking). Furthermore, the algorithms assume a shared feature space between different tasks, while experience transfer between heterogeneous active learning tasks is yet to be studied.

Other related setups include never-ending learning and life-long learning. Never-ending learning is a rather general setup that defines how machines can learn like humans to transfer experience to different tasks in a self-supervised manner, and has been realized in a system for accumulating beliefs by reading continuously from the web [9]. Life-long learning [10], [11], on the other hand, considers feeding the machines with a sequence of tasks with the hope of improving the performance on the next task in the sequence. The setup is similar to our thought but has been realized on only sentiment classification tasks [10].

To the best of our knowledge, neither never-ending nor life-long learning has been carried out on active learning tasks. In fact, allowing the machine to mimic humans in life-long active learning is highly non-trivial, as experience that can be accumulated and transferred between heterogeneous active learning tasks is not well-defined, not to mention applying past experience to future learning tasks.

In this paper, after introducing the cross-dataset (cross-task) active learning problem in Section II, we first propose a notion of machine experience that can be transferred across active learning tasks in Section III. The notion is based on encoding human knowledge of active learning via scoring functions of existing active learning algorithms, and representing machine experience as linear weights that combine the human knowledge. Under the notion, existing active learning algorithms can be simply viewed as taking some special and immutable weights to combine the knowledge.

Then, we improve existing active learning algorithms by designing a novel approach that adaptively update the linear

weights during the active learning process. Inspired by the aforementioned work [4], we connect our problem of updating the linear weights with contextual bandit learning. Based on the connection, we apply a state-of-the-art contextual bandit algorithm, *Linear Upper-Confidence-Bound* (LinUCB) [12], to update the weights. The resulting approach effectively blends existing active learning algorithms towards better performance.

We extend the proposed approach to allow the learned experience (weights) to be transferred across datasets in Section IV. The transferring extension is based on the idea of biased regularization that restricts the adaptive weights to be close to the past experience. The simple formulation of biased regularization can be seamlessly coupled with the LinUCB algorithm to form the transferring extension.

Empirical studies in Section V demonstrate that our approach is competitive to existing active learning algorithms. The results also indicate that the transferring extension effectively improves the learning performance of our approach with the experience learned from both heterogeneous and homogeneous tasks, thus demonstrating the usefulness of the learned experience. Finally, we conclude the possibility of transferring active learning experience in Section VI.

II. BACKGROUND

In this work, we focus on a popular active learning setup called pool-based active learning [3] for binary classification. Under the setup, an active learning algorithm is presented with a labeled pool and an unlabeled pool initially. We denote the labeled pool as $\mathcal{D}_l = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) \dots (\mathbf{x}_{N_l}, y_{N_l})\}$ and the unlabeled pool as $\mathcal{D}_u = \{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{N_u}\}$, where $\mathbf{x}_i, \tilde{\mathbf{x}}_j \in \mathbb{R}^d$, and $y_i \in \{+1, -1\}$. In general, the algorithm can only access a small \mathcal{D}_l in the beginning, while the size of \mathcal{D}_u is relatively large.

With the initial \mathcal{D}_l , the algorithm calls some base model to learn a classifier h_0 . Then, given a budget T , for each iteration $t = 1, 2, \dots, T$, the algorithm is allowed to query the label of an $\tilde{\mathbf{x}}_j \in \mathcal{D}_u$ from some given labeling oracle. The instance-label pair $(\tilde{\mathbf{x}}_j, y_j)$ will then be moved to \mathcal{D}_l , and the base model can be called with the enlarged \mathcal{D}_l to learn a new classifier h_t . The goal of the algorithm is to make the performance of h_1, h_2, \dots, h_T as good as possible, where the performance will be measured with the test accuracy on a separate test set in this work.

We also study how active learning experience can be accumulated across datasets. In the setup of cross-dataset active learning, we present the active learning algorithm with a sequence of datasets $(\mathcal{D}_l^{(1)}, \mathcal{D}_u^{(1)}), (\mathcal{D}_l^{(2)}, \mathcal{D}_u^{(2)}), \dots, (\mathcal{D}_l^{(Q)}, \mathcal{D}_u^{(Q)})$, with the hope of improving the active learning performance along with the sequence like life-long learning [10], [11]. More specifically, we hope that the experience accumulated from $(\mathcal{D}_l^{(1)}, \mathcal{D}_u^{(1)}), \dots, (\mathcal{D}_l^{(q-1)}, \mathcal{D}_u^{(q-1)})$ can be exploited when conducting active learning on $(\mathcal{D}_l^{(q)}, \mathcal{D}_u^{(q)})$ for $q = 2, 3, \dots, Q$.

Many active learning algorithms select $\tilde{\mathbf{x}}_j$ from \mathcal{D}_u in iteration t with a scoring function of instance $\tilde{\mathbf{x}}$ subject to the current classifier h_{t-1} . For an algorithm a , we shall denote

the scoring function as $s_a(\tilde{\mathbf{x}}, h_{t-1})$, and assume that a would query the label of $\tilde{\mathbf{x}}_j = \arg \max_{\tilde{\mathbf{x}} \in \mathcal{D}_u} s_a(\tilde{\mathbf{x}}, h_{t-1})$. The scoring function measures the goodness of each instance, and reflects the strategy taken within the algorithm.

A classic and intuitive strategy is called uncertainty sampling [13], which queries the instance $\tilde{\mathbf{x}}_j$ that the classifier h_{t-1} is most uncertain with. [14] realizes the uncertainty sampling strategy with a scoring function that computes the inverse distance from $\tilde{\mathbf{x}}$ to the hyperplane of h_{t-1} learned from Support Vector Machine (SVM).

Other works argue that uncertainty sampling only works well when h_{t-1} is close enough to the ideal boundary, and may result in unsatisfactory performance when h_{t-1} is not good enough [15]. Representative sampling is a family of strategies, each based on a different scoring function, that tries to improve uncertainty sampling. For example, [16] applies k -means clustering and takes the inverse distance from $\tilde{\mathbf{x}}$ to the cluster center as the scoring function for representativeness, modulated by whether $\tilde{\mathbf{x}}$ resides inside the margin of a SVM classifier h_{t-1} . [17] equips Gaussian distributions on top of k -means clustering to calculate representativeness, and proposes a scoring function that multiplies the uncertainty of $\tilde{\mathbf{x}}$ by its representativeness. [18] optimizes a scoring function based on estimating the label assignments in a min-max view, and argues that the optimized scoring function covers both uncertainty and representativeness.

The strategies above embed our human knowledge of key labeling questions in the scoring functions. Several works [4], [19] also consider selecting the strategies adaptively for better performance, motivated by the fact that human-designed scoring functions cannot always match dataset characteristics and thus adaptive selection may be necessary. The state-of-the-art approach *Active Learning By Learning* [4] performs adaptive strategy selection by connecting the selection problem to bandit learning, and designs a learning-performance-based reward function to guide the bandit learner in selecting reasonable strategies probabilistically. The internal probability that each strategy gets selected reflects the goodness of the strategy, and is updated on the fly within the single dataset.

Recall that we aim to accumulate active learning experience across datasets. Human-designed scoring functions cannot help with so because they are generally immutable and cannot adaptively change with experience. A naïve way of extending current adaptive-selection approaches [4], [19] for accumulating active learning experience is to define the experience as the internal probability distribution for selections, and then transfer the distribution to the next active learning task. Nevertheless, as we shall see in Section V, the unstable nature of probabilistic choices makes the distribution too volatile to serve as robust active learning experience in practice.

III. PROPOSED APPROACH

In this section, we shall first introduce our notion of active learning experience. Then we propose a novel active learning approach, *Linear Strategy Aggregation*, that queries an unlabeled

beled instance and updates the experience simultaneously in each iteration.

A. Notion of active learning experience

As introduced in Section II, the scoring functions of human-designed active learning algorithms represent pieces of human knowledge about key labeling questions. A proper way to combine different pieces of human knowledge, or namely different scoring functions, can then be naturally viewed as experience of active learning.

More specifically, we consider combining, or *blending*, the human-designed scoring functions to a new scoring function for better performance, and define the blending parameters as experience. Note that current adaptive-selection approaches [4] cannot fully match this novel definition, as they blend (via probabilistic selection) the recommended queries of the scoring functions instead of blending the scoring functions directly.

To take an initiative on the definition, we consider the simplest model where the scoring functions are blended linearly, and leave the possibility of using more sophisticated models as future directions. In particular, given a set of scoring functions $\{s_1, s_2, \dots, s_M\}$ from different human-designed strategies, we set the aggregated scoring function to be $\hat{s}(\tilde{\mathbf{x}}, h_{t-1}) = \sum_{m=1}^M w_m s_m(\tilde{\mathbf{x}}, h_{t-1})$. The weight vector $\mathbf{w} = (w_1, w_2, \dots, w_M)$ then contains the blending parameters and serves as the experience that will be transferred.

B. Linear Strategy Aggregation

With the notion of experience established, we now introduce our proposed approach, *Linear Strategy Aggregation* (LSA). LSA solves the task of adaptively updating the experience and querying the unlabeled instance $\tilde{\mathbf{x}}_j$ to maximize the active learning performance. Motivated by previous adaptive selection approaches [4], [19], we design LSA via the connection between the task and a well-known adaptive learning problem of contextual bandit [20]. We will first discuss more details about the contextual bandit problem.

The setup of the contextual bandit problem is as follows [20]: a player is presented with K actions and a budget T . In each iteration $t = 1, \dots, T$, the context vector $\mathbf{z}_{k,t}$ for each action $k \in \{1, 2, \dots, K\}$ is provided, and a player is required to perform an action $k_t \in \{1, 2, \dots, K\}$. Once the action is performed, the corresponding reward $r_{k_t,t}$ is then revealed. The objective of the player is to maximize the cumulative reward. To maximize the cumulative reward, the player is typically required to balance between exploration (choosing actions that improve the estimation of reward) and exploitation (choosing actions with the highest estimated reward).

Many algorithms for the contextual bandit problem have been studied in the literature [12], [21]–[23], and a family of them estimates the reward of an action through a linear model of the corresponding context [12], [22], [23]. A state-of-the-art algorithm of the family is called *Linear Upper-Confidence-Bound* (LinUCB) [12], which not only carries strong theoretical guarantees but also performs well on real-world tasks [24]. Next, we take a closer look at LinUCB, and

then apply it for LSA by connecting the contextual bandit problem back to active learning.

LinUCB maintains the weight vector \mathbf{w}_t of the linear model to be the ridge regression solution from the context vectors to the observed rewards. Specifically, before each iteration t , \mathbf{w}_t is obtained by

$$\mathbf{w}_t = \arg \min_{\mathbf{w}} (\lambda \|\mathbf{w}\|^2 + \|\mathbf{Z}_t \mathbf{w} - \mathbf{r}_t\|^2) \quad , \quad (1)$$

where $\mathbf{Z}_t = (\mathbf{z}_{k_1,1}, \dots, \mathbf{z}_{k_{t-1},t-1})^T$ contains the context vectors that correspond to the chosen actions as rows and $\mathbf{r}_t = (r_{k_1,1}, \dots, r_{k_{t-1},t-1})$ contains the rewards revealed by the chosen actions as elements.

LinUCB runs an online procedure to solve (1) and update \mathbf{w}_t . In particular, LinUCB maintains a matrix $\mathbf{A}_t = \mathbf{Z}_t^T \mathbf{Z}_t + \lambda \mathbf{I}$ and a vector $\mathbf{b}_t = \mathbf{Z}_t^T \mathbf{r}_t$ by

$$\begin{cases} \mathbf{A}_t = \mathbf{A}_{t-1} + \mathbf{z}_{k_{t-1},t-1} \mathbf{z}_{k_{t-1},t-1}^T \\ \mathbf{b}_t = \mathbf{b}_{t-1} + r_{k_{t-1},t-1} \mathbf{z}_{k_{t-1},t-1} \end{cases} \quad , \quad (2)$$

where $\mathbf{A}_0 = \lambda \mathbf{I}$ and $\mathbf{b}_0 = \mathbf{0}$ are initialized before the first iteration. Then, the solution to (1) is simply

$$\mathbf{w}_t = \mathbf{A}_t^{-1} \mathbf{b}_t \quad . \quad (3)$$

To maximize the cumulative reward, LinUCB uses the upper-confidence-bound technique to balance exploration and exploitation. That is, in each iteration t , LinUCB performs the action

$$k_t = \arg \max_k u_{k,t} \quad , \quad (4)$$

where

$$u_{k,t} = \mathbf{w}_t^T \mathbf{z}_{k,t} + \alpha \sqrt{\mathbf{z}_{k,t}^T \mathbf{A}_t^{-1} \mathbf{z}_{k,t}} \quad . \quad (5)$$

The first term corresponds to the estimated reward of action k in iteration t , and the second term represents the uncertainty of action k under its context vector. The parameter α controls the preference between exploration (the second term) and exploitation (the first term).

We follow [19], a pioneer blending approach for active learning, to connect active learning with LSA and contextual bandit with LinUCB. In particular, we treat each $\tilde{\mathbf{x}}_j \in \mathcal{D}_u$ as an action $k \in \{1, 2, \dots, |\mathcal{D}_u|\}$. Then, performing an action k_t in iteration t by LinUCB is equivalent to querying the corresponding $\tilde{\mathbf{x}}_{k_t}$ by LSA. The remaining issues are to specify what the context vectors $\tilde{\mathbf{z}}_{k,t}$ are and how the rewards $r_{k_t,t}$ are calculated. We first discuss our choice of the context vectors to achieve experience updating, and then illustrate our design of the rewards, which represents active learning performance, in Section III-C.

As discussed in Section III-A, our active learning experience \mathbf{w} is defined as the blending parameters of the set of scores $(s_1(\tilde{\mathbf{x}}_{k,t}, h_{t-1}), \dots, s_M(\tilde{\mathbf{x}}_{k,t}, h_{t-1}))$ given an unlabeled instance $\tilde{\mathbf{x}}_{k,t}$. The definition allows a natural connection between LinUCB and LSA by setting

$$\mathbf{z}_{k,t} = (s_1(\tilde{\mathbf{x}}_{k,t}, h_{t-1}), \dots, s_M(\tilde{\mathbf{x}}_{k,t}, h_{t-1})) \quad . \quad (6)$$

Algorithm 1 Linear Strategy Aggregation

Parameters: LinUCB balancing parameter α , ridge regression parameter λ , minimum goodness parameter ϵ , number of iterations T

Input: labeled pool \mathcal{D}_l , unlabeled pool \mathcal{D}_u , scoring functions $\{s_1, s_2, \dots, s_M\}$; a labeling oracle

Begin:

- 1: Initialize $\mathbf{A}_0 = \lambda \mathbf{I}, \mathbf{b}_0 = \mathbf{0}$
 - 2: **for** $t = 1, 2, \dots, T$ **do**
 - 3: Obtain contexts $\mathbf{z}_{1,t}, \mathbf{z}_{2,t}, \dots, \mathbf{z}_{|\mathcal{D}_u|,t}$ with (6) and (7)
 - 4: Obtain $u_{k_t,t}, \mathbf{z}_{k_t,t}$ and $\tilde{\mathbf{x}}_{k_t,t}$ with (4) and (5)
 - 5: Query $\tilde{\mathbf{x}}_{k_t}$ and get \tilde{y}_{k_t} from the oracle
 - 6: Learn h_t with $\mathcal{D}_l \cup \{(\tilde{\mathbf{x}}_{k_t}, \tilde{y}_{k_t})\}$
 - 7: Obtain v_t with (9)
 - 8: Calculate $r_{k_t,t}$ with (8)
 - 9: Update $\mathbf{A}_t, \mathbf{b}_t, \mathbf{w}_t$ by $(\mathbf{z}_{k_t,t}, r_{k_t,t})$ with (2) and (3)
 - 10: $\mathcal{D}_l = \mathcal{D}_l \cup \{(\tilde{\mathbf{x}}_{k_t}, \tilde{y}_{k_t})\}, \mathcal{D}_u = \mathcal{D}_u \setminus \{\tilde{\mathbf{x}}_{k_t}\}$
 - 11: **end for**
-

Then, the vector \mathbf{w}_t in LinUCB corresponds to the evolving experience \mathbf{w} calculated by ridge regression; the inner product $\mathbf{w}_t^T \mathbf{z}_{k,t}$, which is the first term of (5), corresponds to the aggregated scoring function $\hat{s}(\tilde{\mathbf{x}}_{k,t}, h_{t-1})$ that is made from both the current experience \mathbf{w}_t and the human knowledge $\{s_m\}_{m=1}^M$. LSA queries an unlabeled instance with (4) and (5), which contains $\hat{s}(\cdot, \cdot)$ as well as an exploration term introduced by LinUCB, and updates the experience \mathbf{w}_t with (3) and .

Recall that the goal of ridge regression within LinUCB is to provide a good estimate from the context vector to the reward. We apply one trick in $\mathbf{z}_{k,t}$ to improve the quality of the estimate. In particular, we add another element of $\mathbf{z}_{k,t}[0]$, and set the element to a constant value of the previous reward

$$\mathbf{z}_{k,t}[0] = r_{k_{t-1}, t-1}, \quad (7)$$

where the rewards (including the edge case of $\mathbf{z}_{k,1}[0]$) will be defined in Section III-C. According to (5), the added constant does not affect the choice of k_t , but it allows ridge regression to utilize the previous reward for estimating the current reward. In other words, the value provides a shared context on the active learning performance to assist the linear model. Empirically, we observe that the trick indeed improves the quality of the estimate and the stability of LSA.

C. Reward scheme

The only issue left for LSA is a properly designed reward that represents active learning performance, or namely test accuracy in this work. A state-of-the-art reward function proposed is called importance-weighted accuracy (IW-ACC), which is used in the *Active Learning By Learning* (ALBL) approach [4]. IW-ACC weighs each instance in \mathcal{D}_l with the inverse of the probability that the instance is queried, and calculates the weighted accuracy as the reward. The importance weighting allows IW-ACC to be an unbiased estimator of the test accuracy.

More specifically, in each iteration t of ALBL, let $\tilde{\mathbf{x}}_{k_t}$ be the instance queried, y_{k_t} be the obtained label, and $p_{k_t,t}$ be the probability of querying $\tilde{\mathbf{x}}_{k_t}$. Then, with $v_t = p_{k_t,t}^{-1}$, IW-ACC is calculated as

$$r_{k_t, \tau} = \frac{\sum_{t=1}^{\tau} v_t \mathbb{I}[h_{\tau}(\tilde{\mathbf{x}}_{k_t}) = y_{k_t}]}{\sum_{t=1}^{\tau} v_t}, \quad (8)$$

where $\mathbb{I}[\cdot]$ is the indicator function. The probability $p_{k_t,t}$ reflects the *goodness* of $\tilde{\mathbf{x}}_{k_t}$ in iteration t , and the key idea of IW-ACC is to assign v_t as the inverse of the goodness to correct the sampling bias during active learning.

Nevertheless, unlike ALBL, LSA is a deterministic algorithm based on LinUCB. Thus, there is no $p_{k_t,t}$ and IW-ACC cannot be directly taken as the reward. We thus propose a new reward scheme that mimics the key idea of IW-ACC. In our proposed scheme, each instance $\tilde{\mathbf{x}}_{k_t}$ queried in iteration t is weighted with

$$v_t = \left(\max(u_{k_t,t}, \epsilon) \right)^{-1} \quad (9)$$

where $u_{k_t,t}$ is from (5) and $\epsilon > 0$ is a small constant.

Recall that LSA maximizes over $u_{k,t}$ to decide the instance to be queried. That is, $u_{k,t}$ reflects the goodness of the unlabeled instance $\tilde{\mathbf{x}}_{k,t}$. By using the inverse of $u_{k,t}$ as weights, our proposed scheme effectively meets the key idea of importance weighting behind IW-ACC while avoiding the need of probabilistic queries. The small constant $\epsilon > 0$ guards the rare edge cases of $u_{k_t,t} \leq \epsilon$.

In the proposed LSA, the rewards are of another use of serving as $\mathbf{z}_{k,t}[0] = r_{k_{t-1}, t-1}$ in (7). When $t = 1$, there is technically no “previous reward” to use in (7). The simplest choice would be taking $\mathbf{z}_{k,1}[0] = 0.5$ for representing the random-guessing accuracy. In this work, we heuristically take $\mathbf{z}_{k,1}[0]$ to be the training accuracy when learning from the initial \mathcal{D}_l in order to provide a better shared context on the performance.

With the proposed scheme, the final piece of LSA is now complete. In each iteration t , LSA simply runs LinUCB to query an unlabeled instance $\tilde{\mathbf{x}}_{k_t}$ using (4) and updates the experience \mathbf{w}_t with (3) by the context vector $\mathbf{z}_{k_t,t}$ as well as the proposed reward $r_{k_t,t}$ in (8). The details of LSA are listed in Algorithm 1.

IV. ACTIVE LEARNING ACROSS DATASETS

LSA is now able to adaptively update the experience within any single dataset. Our next goal is to achieve experience transfer across datasets, with the hope of improving active learning performance. We thus design an extension of LSA, called *Transfer LSA* (T-LSA), that takes the learned experience as a reference when conducting active learning on the current dataset.

Our design is motivated from an earlier work that focuses on personalized handwriting recognition [25]. The main idea of the work is to first learn a generic handwriting recognizer \mathbf{w}_{gen} by SVM from a large amount of handwriting data of all people. The personalized handwriting recognizer \mathbf{w} is then learned from a small amount of individual data via a

Biased Regularization SVM (BRSVM). BRSVM replaces the ℓ_2 regularization term $\frac{1}{2}\|\mathbf{w}\|^2$ in the objective function of SVM with a biased regularization term $\frac{1}{2}\|\mathbf{w} - \mathbf{w}_{\text{gen}}\|^2$ to enforce the personalized \mathbf{w} to be close to the generic \mathbf{w}_{gen} .

BRSVM for personalized handwriting recognizer allows learning of \mathbf{w} with the prior knowledge of \mathbf{w}_{gen} as a reference point. In our cross-dataset active learning problem, we intend to take \mathbf{w}_{prev} , the experience learned from other datasets, as our reference point. For simplicity, let us first assume that \mathbf{w}_{prev} comes from the experience of active learning from one previous dataset. That is, $\mathbf{w}_{\text{prev}} = \mathbf{w}_T$ learned from $(\mathcal{D}_l^{(1)}, \mathcal{D}_u^{(1)})$. Recall that \mathbf{w}_t in LSA is the ridge-regression solution of (1). Then, we borrow the idea of BRSVM to replace $\frac{1}{2}\|\mathbf{w}\|^2$ with $\frac{1}{2}\|\mathbf{w} - \mathbf{w}_{\text{prev}}\|^2$ as our regularization term. That is, biased regularization can be simply achieved by solving

$$\hat{\mathbf{w}}_t = \arg \min_{\mathbf{w}} \lambda \|\mathbf{w} - \mathbf{w}_{\text{prev}}\|^2 + \|\mathbf{Z}_t \mathbf{w} - \mathbf{r}_t\|^2 \quad (10)$$

instead. The close-form solution is

$$\hat{\mathbf{w}}_t = (\mathbf{Z}_t^T \mathbf{Z}_t + \lambda \mathbf{I})^{-1} (\mathbf{Z}_t^T \mathbf{r}_t + \lambda \mathbf{w}_{\text{prev}}) \quad (11)$$

The parameter λ now represents the trust of previous experience.

To integrate (11) into LSA, we need to update $\hat{\mathbf{w}}_t$ online like (2) and (3). Recall that (2) maintains $\mathbf{A}_t = \mathbf{Z}_t^T \mathbf{Z}_t + \lambda \mathbf{I}$ and $\mathbf{b}_t = \mathbf{Z}_t^T \mathbf{r}_t$. Then, (11) can be re-written as

$$\hat{\mathbf{w}}_t = \mathbf{A}_t^{-1} \underbrace{(\mathbf{b}_t + \lambda \mathbf{w}_{\text{prev}})}_{\mathbf{b}'_t}. \quad (12)$$

Notice that the only difference between (3) and (12) is the term $\lambda \mathbf{w}_{\text{prev}}$ between \mathbf{b}_t and \mathbf{b}'_t . Thus, we can easily achieve biased regularization in T-LSA by replacing $\mathbf{b}_0 = \mathbf{0}$ in LSA with $\mathbf{b}'_0 = \lambda \mathbf{w}_{\text{prev}}$ and maintaining \mathbf{b}'_t instead of \mathbf{b}_t . The weight vector $\hat{\mathbf{w}}_t$ can then be updated online with $\mathbf{A}_t^{-1} \mathbf{b}'_t$ in (12). When $\mathbf{w}_{\text{prev}} = \mathbf{0}$, which means zero experience, biased regularization falls back to usual ℓ_2 regularization and T-LSA falls back to LSA.

We now consider the full setup of cross-dataset active learning, as defined in Section II, where a sequence of datasets, $(\mathcal{D}_l^{(1)}, \mathcal{D}_u^{(1)}), \dots, (\mathcal{D}_l^{(Q)}, \mathcal{D}_u^{(Q)})$, is presented. Let $\hat{\mathbf{w}}^{(1)}$ be the experience learned from $(\mathcal{D}_l^{(1)}, \mathcal{D}_u^{(1)})$. When learning \mathbf{w}_t on $(\mathcal{D}_l^{(2)}, \mathcal{D}_u^{(2)})$ using $\mathbf{w}_{\text{prev}} = \hat{\mathbf{w}}^{(1)}$ as the reference point in (10), the first term $\lambda \|\mathbf{w} - \mathbf{w}_{\text{prev}}\|^2$ allows the information of the earlier experience to be somewhat preserved, and the second term $\|\mathbf{Z}_t \mathbf{w} - \mathbf{r}_t\|^2$ allows new experience to be accumulated. Thus, $\hat{\mathbf{w}}^{(2)}$ learned from $(\mathcal{D}_l^{(2)}, \mathcal{D}_u^{(2)})$ contains experience from both the first and the second datasets. It is then natural to learn $\hat{\mathbf{w}}^{(3)}$ on $(\mathcal{D}_l^{(3)}, \mathcal{D}_u^{(3)})$ with $\mathbf{w}_{\text{prev}} = \hat{\mathbf{w}}^{(2)}$, or more generally learn $\hat{\mathbf{w}}^{(q)}$ on $(\mathcal{D}_l^{(q)}, \mathcal{D}_u^{(q)})$ with $\mathbf{w}_{\text{prev}} = \hat{\mathbf{w}}^{(q-1)}$ for $q = 2, \dots, Q$. The simple use of $\mathbf{w}_{\text{prev}} = \hat{\mathbf{w}}^{(q-1)}$ completes the design of the full T-LSA algorithm, as listed in Algorithm 2. For simplicity, we overload \mathbf{b}_t to denote \mathbf{b}'_t in Algorithm 2.

Algorithm 2 Transfer LSA

Parameters: Same as parameters for Algorithm 1

Input: Datasets sequence $(\mathcal{D}_l^{(1)}, \mathcal{D}_u^{(1)}), \dots, (\mathcal{D}_l^{(Q)}, \mathcal{D}_u^{(Q)})$, scoring functions for Algorithm 1

Begin:

- 1: $\mathbf{w}_{\text{prev}} \leftarrow \mathbf{0}$
 - 2: **for** $q = 1, 2, \dots, Q$ **do**
 - 3: Initialize Algorithm 1 (LSA) with $(\mathbf{A}_0, \mathbf{b}_0) = (\lambda \mathbf{I}, \lambda \mathbf{w}_{\text{prev}})$ instead
 - 4: Run the initialized LSA on $(\mathcal{D}_l^{(q)}, \mathcal{D}_u^{(q)})$ and obtain experience $\hat{\mathbf{w}}^{(q)}$
 - 5: $\mathbf{w}_{\text{prev}} \leftarrow \hat{\mathbf{w}}^{(q)}$
 - 6: **end for**
-

With the help of biased regularization, T-LSA achieves cross-dataset active learning. When the experience is helpful, which possibly happens when transferring experience from more related datasets, T-LSA utilizes the experience to speedup exploration in the wild. When the experience is not so helpful, which can mean a negative transfer in the terminology of transfer learning, the second term $\|\mathbf{Z}_t \mathbf{w} - \mathbf{r}_t\|^2$ in (10) allows new experience to be adaptively accumulated. In Section V-B, we will empirically study how different kinds of experience affect the performance of T-LSA.

V. EXPERIMENT

We couple the following key active learning algorithms with our proposed approaches, LSA and T-LSA, to validate their empirical performance. The algorithms, as illustrated in Section II, are

- 1) UNCERTAIN: uncertainty sampling with SVM [14].
- 2) REPRESENT: representative sampling based on k -mean clustering [16]. Because the uncertainty part is essentially the same as UNCERTAIN, we only take the scoring function for representativeness for blending.
- 3) DUAL: another representative sampling approach using mixture-of-Gaussian weighted uncertainty as scoring function [17].
- 4) QUIRE: another representative sampling approach using the min-max view of label-assignment to optimize the scoring function [18].

We take logistic regression as our base classification model, and use the ℓ_2 -regularized logistic regression solver of LIBLINEAR [26] with default parameters to learn a classifier from the model.

We conduct experiments on two sets of benchmark datasets. The first set is commonly used to validate pool-based active learning approaches for binary classification, and is taken to validate not only the competitiveness of LSA versus other approaches, but also to examine the potential of T-LSA for cross-dataset active learning with heterogeneous datasets. The first benchmark set include the following eight datasets from the UCI repository [27]: *austra*, *breast*, *diabetes*, *german*, *heart*, *letterMvsN*, *liver*, and *wdbc*, where the dataset *letterMvsN* is constructed from a multi-class dataset *letter*.

TABLE I: LSA versus underlying algorithms based on t -test at 90% confidence level (#win/#tie/#loss)

rank	percentage of queried instances							total
	5%	10%	15%	20%	30%	40%	50%	
1st	0/6/2	0/7/1	0/7/1	0/8/0	0/8/0	0/8/0	1/6/1	1/50/5
2nd	0/8/0	0/8/0	1/7/0	0/8/0	0/8/0	1/7/0	1/7/0	3/53/0
3rd	1/7/0	4/4/0	4/4/0	6/2/0	5/3/0	4/4/0	3/5/0	27/29/0
4th	4/4/0	7/1/0	8/0/0	8/0/0	8/0/0	7/1/0	6/2/0	48/8/0
total	5/25/2	11/20/1	13/18/1	14/18/0	13/19/0	12/20/0	11/20/1	79/140/5

TABLE II: LSA versus ALBL based on t -test at 90% confidence level (#win/#tie/#loss)

	percentage of queried instances							total
	5%	10%	15%	20%	30%	40%	50%	
ALBL	0/8/0	2/6/0	2/6/0	2/6/0	2/6/0	2/5/1	3/5/0	13/42/1

The second set, which contains two datasets of handwritten digit recognition, *USPS* and *MNIST*, is used in several previous studies of multi-task learning [28], [29]. We take the second set to examine the potential of T-LSA for cross-dataset active learning with homogeneous datasets. We follow [28] to reduce the feature dimensions of *USPS* and *MNIST* to 87 and 62 respectively with principal component analysis.

For the larger datasets *letterMvsN*, *USPS* and *MNIST*, we randomly keep only 2000 examples to make experiments sufficiently efficient. Then, we split each dataset randomly with 50% for training and 50% for testing. We take the training set as our unlabeled pool \mathcal{D}_u , and the test set for reporting active learning performance. We randomly select 4 instances from the unlabeled pool \mathcal{D}_u as our initial labeled pool \mathcal{D}_l . Experiments on each dataset are averaged over 10 times.

We will first compare LSA with the four underlying active learning algorithms and the state-of-the-art ALBL approach [4] for blending those algorithms on single datasets. Then, we will compare T-LSA with LSA and ALBL under the cross-dataset setting to understand the effectiveness of experience transfer. For fairness, we will also naively extend ALBL to T-ALBL as illustrated in Section II, and take T-ALBL for comparison. In particular, T-ALBL initializes the internal probability distribution with the previously learned distribution to achieve experience transfer.

Parameter tuning of active learning is known to be hard [4]. In our experiments, we run the approaches on several parameter combinations, and report the result of the best combination. Practically, existing blending approaches like ALBL [4] or COMB [19] can then be run on top of the combinations to adaptively approximate the best result. Specifically, for the experiments on single datasets, we run LSA with $\lambda = 1$ and $\alpha \in \{1.5, 2.0, 2.5\}$. For the experiments of cross-dataset active learning, we fix $\alpha = 1.5$ and run LSA and T-LSA with $\lambda = 1$ and $\lambda \in \{1, 5, 10\}$ respectively. For the parameters of other algorithms, we follow the recommended parameters provided in the paper/codes from the authors.

We do not include another adaptive blending approach of COMB [19] for two reasons:

- 1) ALBL is known to outperform COMB on single

datasets [4].

- 2) Unlike ALBL, which maintains an internal probabilistic distribution on the active learning algorithms, COMB maintains the distribution on the unlabeled instances. It is non-trivial to transfer the distribution as experience to other datasets with different number of instances.

A. Experiments on Single Datasets

We first compare LSA with the four underlying active learning algorithms on the first set of eight benchmark datasets, and plot the test accuracy under different percentages of queries in Fig. 1. From the results, we can observe that LSA is usually close to the best curves of the four algorithms after querying 10% of unlabeled instances. The results demonstrate that LSA is effective in terms of blending human knowledge towards decent query decisions. The less-strong performance of LSA in the first 10% of queries hints the need of using experience to guide exploration instead of starting from zero experience.

The results in Fig. 1 is further supported by Table I with t -tests at 90% significance level. The tests compare LSA with the underlying algorithms at different ranks. Table I indicates that LSA often yields competitive performance with the best underlying algorithm, and is always no-worse than the second best. The results in Fig. 1 and Table I confirm that LSA to be a decent adaptive blending approach for active learning, just like its ancestors of ALBL [4] and COMB [19]. Note that LSA is a deterministic approach while ALBL and COMB are both probabilistic.

To understand the effectiveness of LSA as a blending approach, we compare LSA with ALBL. Because of space limits, we plot the test accuracy along with the standard deviation on only four of the datasets, *austra*, *breast*, *heart* and *wdbc* in Fig. 2. We also compare LSA with ALBL with t -tests at 90% confidence level on all datasets, and summarize the results in Table II. The results of both Fig. 2 and Table II indicate that LSA is competitive to and sometimes even slightly better than ALBL. Furthermore, according to Fig. 2, we can observe that the variation (standard deviation) of the LSA curve not only decreases more rapidly than that of the ALBL curve, but is also generally smaller after the first 10% of the exploration queries. The observation indicates that ALBL, being a probabilistic blending approach, is generally less stable than LSA, and matches our conjecture in Section II that the distribution in ALBL may be too volatile to serve as robust active learning experience in practice.

B. Experiments on Active Learning Across Datasets

Next, we move to the experiments of cross-dataset active learning. We first introduce the experiment setting before we proceed to discuss the details of the experiment results. The experiment setting is as follows: A target dataset is first picked, and a random sequence that consists of other datasets is generated. Transferring algorithms, including T-LSA and T-ALBL, are then run on first q datasets of the sequence to accumulate experience. With the previous experience, the active learning performance of the transferring algorithms is

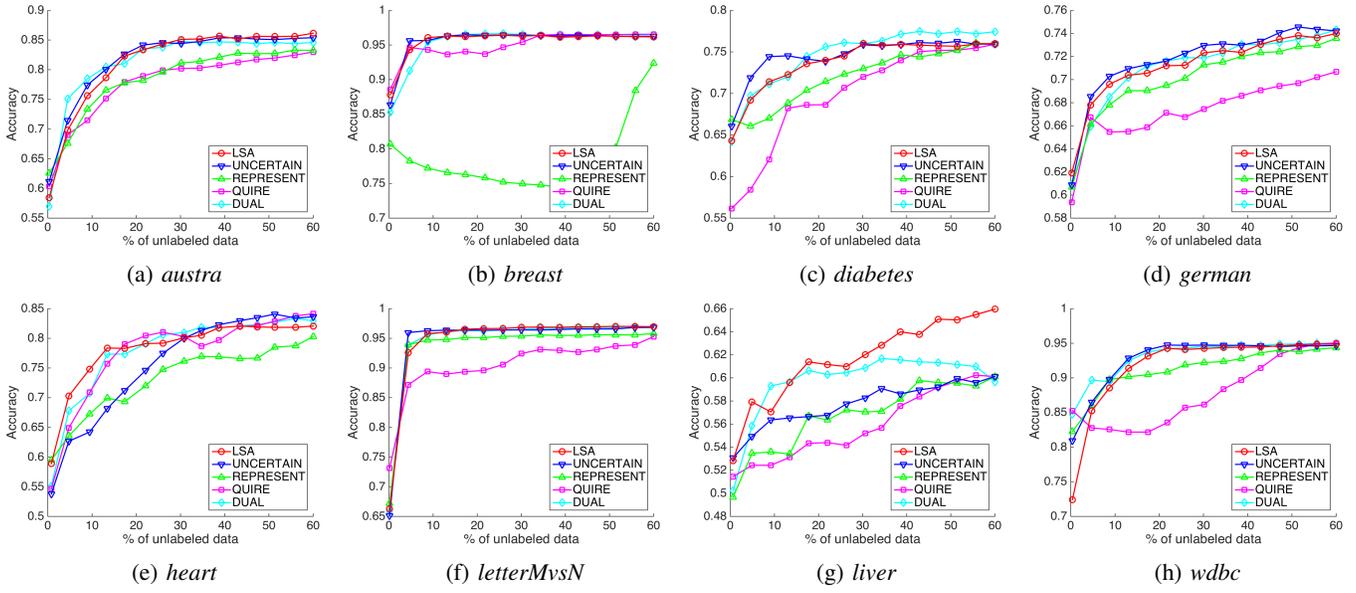


Fig. 1: Test Accuracy of LSA versus underlying strategies

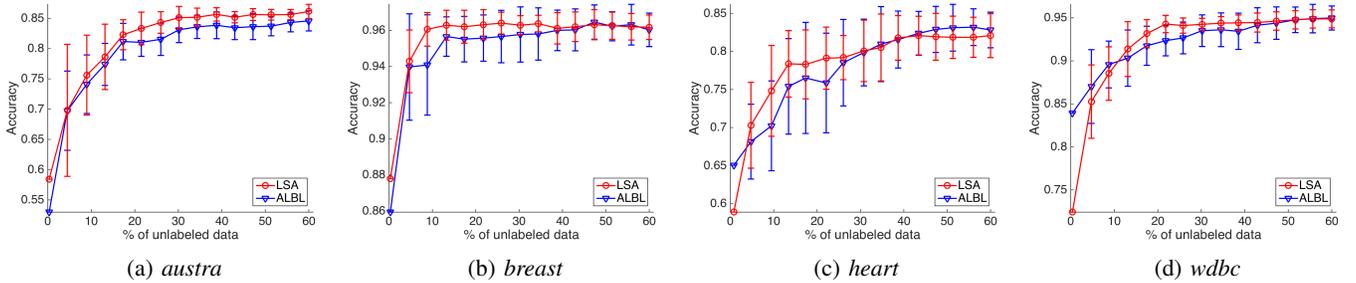


Fig. 2: Test Accuracy of LSA versus ALBL

evaluated on the target dataset. Each result is averaged over 10 different random sequences.

The experiments of active learning across datasets are conducted in two different scenarios, where homogeneous and heterogeneous tasks are considered respectively. Specifically, a set of homogeneous tasks consists of datasets that share similar learning targets and the same feature space, and is constructed from the two benchmark datasets of multi-task learning. A set of heterogeneous tasks, on the other hand, involves datasets having different learning targets and feature space, and is simulated by the eight benchmark datasets of active learning. We will first discuss the experiments on homogeneous tasks, where algorithms that exploit the transferred experience are expected to perform better. The experiments on heterogeneous tasks, which is a more general but more challenging scenario, will then be discussed.

For the experiments in each scenario, we first compare T-LSA and T-ALBL using experience from different number of previous datasets (i.e. different q) with their non-transferring predecessors, namely LSA and ALBL, to evaluate the effective-

ness of experience transfer of active learning. Then, we will directly compare T-LSA with T-ALBL, LSA and ALBL using a specific q to understand the absolute performance difference between T-LSA and other competitors.

We choose to not include QUIRE in the cross-dataset experiments because QUIRE is considerably more time-consuming given its label-assignment estimation steps.

a) Experiments on Homogeneous Tasks: The experiments of learning across homogeneous tasks are conducted on two benchmark datasets of hand-written digit recognition, *USPS* and *MNIST*, for multi-task learning. We split both *USPS* and *MNIST* into 5 binary classification datasets, namely *0vs1*, *2vs3*, *4vs5*, *6vs7* and *8vs9* to construct the set of homogeneous learning tasks. Since the active learning performance on *USPS* and *MNIST* converges quickly, we only compare the results with respect to the queries in first 10% of unlabeled data to better illustrate the difference.

We compare T-LSA with LSA and T-ALBL with ALBL, and present the results of test accuracy in Fig. 3 and Fig. 4 respectively. Owing to the readability, only results of two tasks

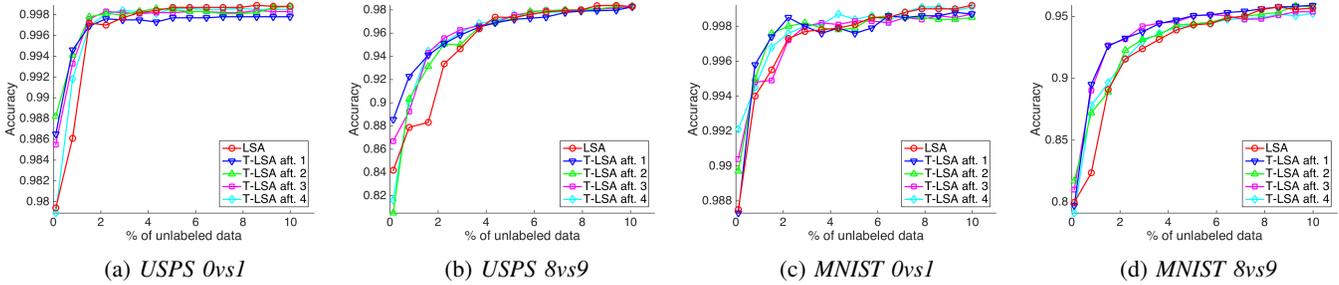


Fig. 3: Test Accuracy of LSA versus Transfer LSA on *MNIST* and *USPS*

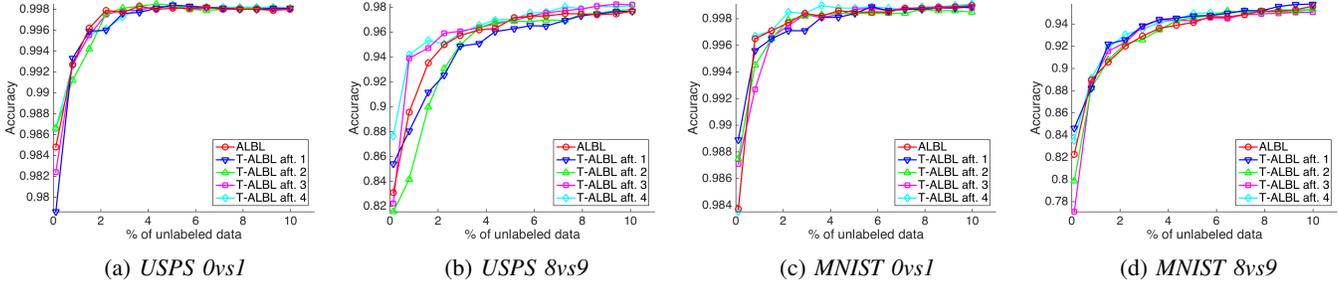


Fig. 4: Test Accuracy of ALBL versus Transfer ALBL on *USPS* and *MNIST*

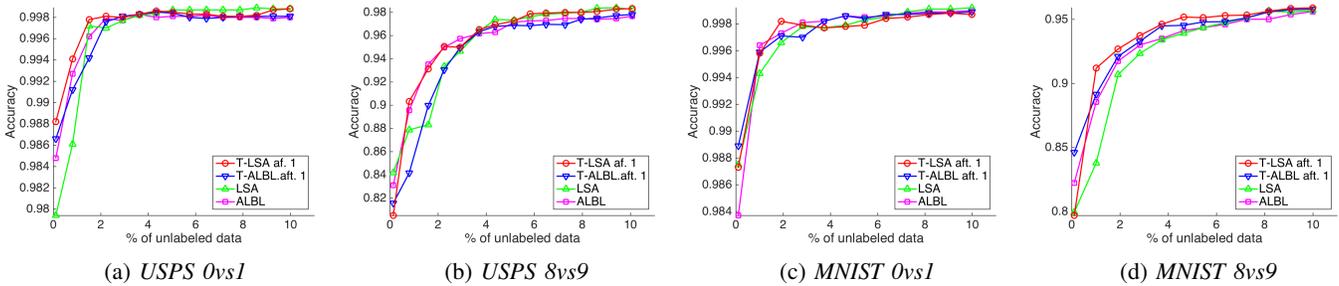


Fig. 5: Test Accuracy of Transfer LSA versus other competitors on *USPS* and *MNIST*

on each dataset are presented here. From Fig. 3, we observe that T-LSA generally outperforms LSA on tasks *0vs1* and *8vs9* of both *USPS* and *MNIST*. On the other hand, Fig. 4 indicates that T-ALBL performs similarly or even worse than ALBL on task *0vs1* of both *USPS* and *MNIST*. For task *8vs9*, the improvement of T-ALBL with regard to ALBL is rather minor on *MNIST*, while obvious negative transfer can be observed on *USPS*.

We then compare T-LSA with T-ALBL, LSA, and ALBL directly using experience from one previous dataset, to examine the absolute performance difference between T-LSA and other competitors. The results are illustrated in Fig. 5. These competitors are further compared on all five tasks of both datasets based on *t*-test at 90% confidence level, and the results are summarized in Table III. From Fig. 5, we can observe that performance of LSA is again inferior especially in the first 4% of queries. T-LSA, on the other hand, often performs the best

among all four competitors. The results of Table III indicates a slight improvement of T-LSA over LSA in the initial stage of learning, and shows the competitive performance of T-LSA over other competitors.

The observations on both *USPS* and *MNIST* demonstrate that T-LSA successfully improves the active learning performance of LSA by transferring the active learning experience via the proposed linear weights, which is as expected in the scenario of active learning across homogeneous tasks. T-ALBL, however, often performs inferior than ALBL, confirming that experience transfer via the probability distribution of ALBL can lead to negative impact.

b) Experiments on Heterogeneous Tasks: Next, we shall discuss the experiments on learning across heterogeneous tasks. The experiments are conducted on the eight benchmark datasets of active learning. The feature spaces and the learning targets vary from each others between different active learning

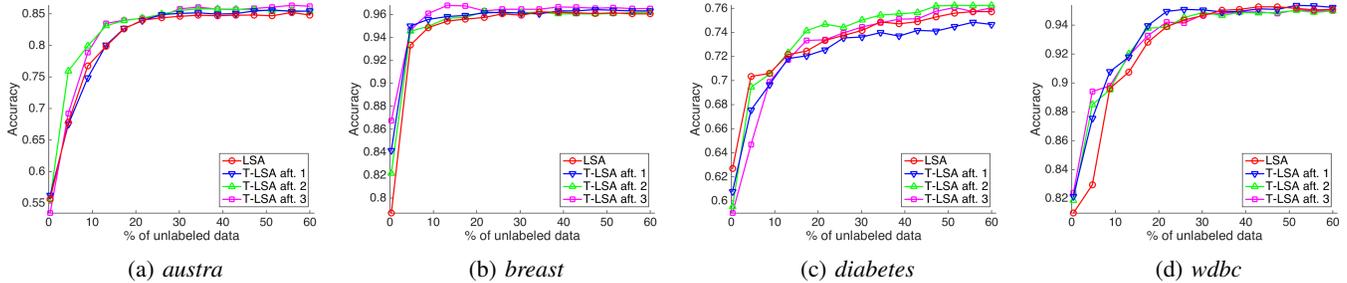


Fig. 6: Test Accuracy of LSA versus Transfer LSA

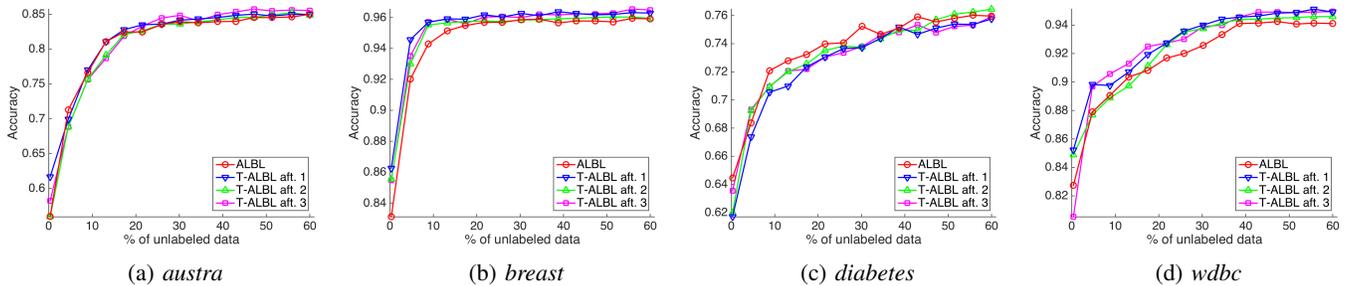


Fig. 7: Test Accuracy of ALBL versus Transfer ALBL

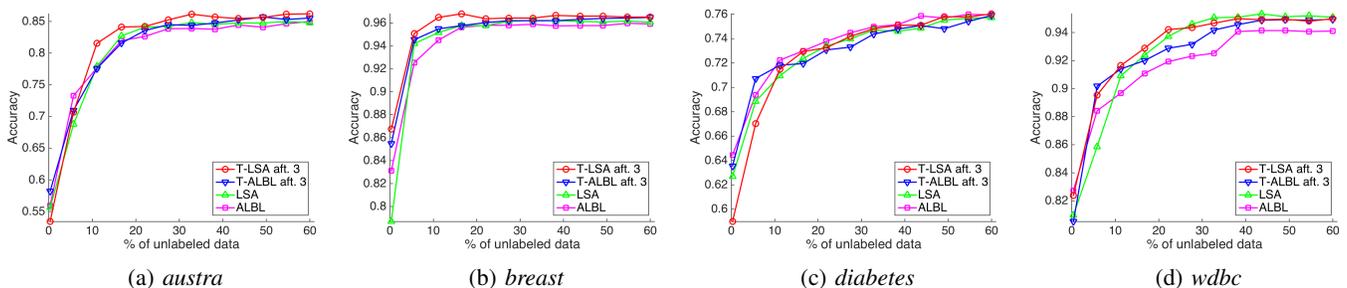


Fig. 8: Test Accuracy of Transfer LSA versus Transfer ALBL

TABLE III: Transfer LSA versus other competitors on USPS and MNIST based on t -test at 90% confidence level (#win/#tie/#loss)

	percentage of queried instances					total
	2%	4%	6%	8%	10%	
LSA	2/8/0	0/10/0	0/10/0	0/10/0	0/10/0	2/48/0
ALBL	0/9/1	0/10/0	2/8/0	0/10/0	1/9/0	3/46/1
T-ALBL aft. 1	0/10/0	0/9/1	1/8/1	1/8/1	1/9/0	3/44/3
total	2/27/1	0/29/1	3/26/1	1/28/1	2/28/0	8/138/4

datasets. We set $q = 3$ in the experiments.

We first compare T-LSA with LSA and T-ALBL with ALBL, and present the results in Fig. 6 and Fig. 7 respectively. Owing to the space limits and readability, only selected results on *austra*, *breast*, *diabetes* and *wdbc* are presented. According to Fig. 6, T-LSA improves the performance of LSA on datasets *austra*, *breast* and *wdbc*. For dataset *diabetes*, T-LSA is

TABLE IV: Transfer LSA versus other competitors based on t -test at 90% confidence level (#win/#tie/#loss)

	percentage of queried instances							total
	5%	10%	15%	20%	30%	40%	50%	
LSA	1/7/0	1/7/0	1/7/0	1/7/0	1/7/0	2/6/0	1/6/1	8/47/1
ALBL	0/7/1	2/6/0	2/6/0	3/5/0	3/5/0	3/5/0	2/6/0	15/40/1
T-ALBL aft. 3	0/7/1	0/8/0	2/6/0	3/5/0	1/7/0	1/7/0	1/7/0	8/47/1
total	1/21/2	3/21/0	5/19/0	7/17/0	5/19/0	6/18/0	4/19/1	31/134/3

inferior in the initial stage, but can quickly catch up and even outperform LSA. On the other hand, we can observe from Fig. 7 that T-ALBL improves over ALBL on datasets *breast* and *wdbc*, but is inferior on datasets *austra* and *diabetes*.

We then compare T-LSA directly with T-ALBL, LSA and ALBL, where the transferring algorithms can exploit the experience from previous 3 datasets (i.e. $q = 3$). We illustrate the results in Fig. 8. We also compare these algorithms based on t -test at 90% confidence level, and summarize the results

in Table IV. From Fig. 8, T-LSA reaches the best performance among all four competitors on datasets *austra*, *breast* and *wdbc*, and can catch up with the best competitor after querying 10% of unlabeled data on dataset *diabetes*. The results of Table IV further confirm that T-LSA can often outperform other competitors.

The aforementioned observations demonstrate that experience transfer via our proposed linear weights is superior to that via the probabilistic distribution of ALBL with the following two advantages: (1) better improvement from experience transfer and (2) ability to recover more quickly when the transferred experience is negative to performance. In addition, T-LSA is shown to improve over LSA by providing a better starting point for exploration in the initial stage of active learning.

The success of T-LSA in both scenarios positively answers the question in our title, where active learning experience can indeed be transferred to improve the active learning performance.

VI. CONCLUSION

We propose a novel approach that accomplishes the mission of transferring active learning experience across datasets. The approach is based on a unified representation of human knowledge and environment status about active learning, and a linear model on the representation. The model allows taking the linear weights as experience, and can be updated by the LinUCB algorithm for contextual bandit learning through a novel reward function. The experience learned from the model can be transferred to other active learning tasks through biased regularization. Empirical studies not only confirm the competitiveness of the proposed approach, but also confirm that it can be beneficial to transfer the experience across active learning tasks that are either homogeneous or heterogeneous for better performance.

REFERENCES

- [1] Y. Liu, "Active learning with support vector machine applied to gene expression data for cancer classification," *Journal of Chemical Information and Modeling*, vol. 44, no. 6, pp. 1936–1941, 2004.
- [2] C. Zhang and T. Chen, "An active learning framework for content-based information retrieval," *IEEE Transactions on Multimedia*, vol. 4, no. 2, pp. 260–268, 2002.
- [3] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2009.
- [4] W. Hsu and H. Lin, "Active learning by learning," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, (AAAI) 2015*, 2015, pp. 2659–2665.
- [5] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [6] W. Daelemans, B. Goethals, and K. Morik, "Actively transfer domain knowledge," in *Machine Learning and Knowledge Discovery in Databases, European Conference, (ECML/PKDD) 2008*, vol. 5212. Springer, 2008.
- [7] D. C. Kale and Y. Liu, "Accelerating active learning with transfer learning," in *Proceedings of the 2013 IEEE 13th International Conference on Data Mining*, 2013, pp. 1085–1090.
- [8] D. C. Kale, M. Ghazvininejad, A. Ramakrishna, J. He, and Y. Liu, "Hierarchical active transfer learning," in *Proceedings of the 2015 SIAM International Conference on Data Mining*, 2015, pp. 514–522.
- [9] T. M. Mitchell, W. W. Cohen, E. R. H. Jr., P. P. Talukdar, J. Betteridge, A. Carlson, B. D. Mishra, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. A. Platanios, A. Ritter, M. Samadi, B. Settles, R. C. Wang, D. T. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling, "Never-ending learning," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, (AAAI) 2015*, 2015, pp. 2302–2310.
- [10] Z. Chen, N. Ma, and B. Liu, "Lifelong learning for sentiment classification," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, (ACL) 2015*, 2015, pp. 750–756.
- [11] P. Ruvolo and E. Eaton, "Ella: An efficient lifelong learning algorithm," in *Proceedings of the 30th International Conference on Machine Learning, (ICML) 2013*, 2013, pp. 507–515.
- [12] W. Chu, L. Li, L. Reyzin, and R. E. Schapire, "Contextual bandits with linear payoff functions," in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, (AISTATS) 2011*, 2011, pp. 208–214.
- [13] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval.*, 1994, pp. 3–12.
- [14] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2001.
- [15] C.-L. Li, C.-S. Ferng, and H.-T. Lin, "Active learning using hint information," *Neural Computation*, vol. 27, no. 8, pp. 1738–1765, August 2015.
- [16] Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang, "Representative sampling for text classification using support vector machines," in *Proceedings of the 25th European Conference on Information Retrieval Research*, vol. 2633, 2003, pp. 393–406.
- [17] P. Donmez, J. G. Carbonell, and P. N. Bennett, "Dual strategy active learning," in *Proceedings of the 18th European Conference on Machine Learning, (ECML) 2007*, 2007, pp. 116–127.
- [18] S. Huang, R. Jin, and Z. Zhou, "Active learning by querying informative and representative examples," in *Advances in Neural Information Processing Systems, (NIPS) 2010*, 2010, pp. 892–900.
- [19] Y. Baram, R. El-Yaniv, and K. Luz, "Online choice of active learning algorithms," in *Proceedings of the 20th International Conference on Machine Learning, (ICML) 2003*, 2003, pp. 19–26.
- [20] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The non-stochastic multiarmed bandit problem," *Society of Industrial and Applied Mathematics J. Comput.*, vol. 32, no. 1, pp. 48–77, 2002.
- [21] A. Beygelzimer, J. Langford, L. Li, L. Reyzin, and R. E. Schapire, "Contextual bandit algorithms with supervised learning guarantees," in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, (AISTATS) 2011*, 2011, pp. 19–26.
- [22] P. Auer, "Using confidence bounds for exploitation-exploration trade-offs," *Journal of Machine Learning Research*, vol. 3, pp. 397–422, 2002.
- [23] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *Proceedings of the 30th International Conference on Machine Learning, (ICML) 2013*, 2013, pp. 127–135.
- [24] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th International Conference on World Wide Web, (WWW) 2010*, 2010, pp. 661–670.
- [25] W. Kienzle and K. Chellapilla, "Personalized handwriting recognition via biased regularization," in *Proceedings of 23rd International Conference on Machine Learning, (ICML) 2006*, 2006, pp. 457–464.
- [26] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.
- [27] M. Lichman, "UCI machine learning repository," 2013.
- [28] Z. Kang, K. Grauman, and F. Sha, "Learning with whom to share in multi-task feature learning," in *Proceedings of the 28th International Conference on Machine Learning, (ICML) 2011*, 2011, pp. 521–528.
- [29] A. Kumar and H. D. III, "Learning task grouping and overlap in multi-task learning," in *Proceedings of the 29th International Conference on Machine Learning, (ICML) 2012*, 2012.