Lazy Transfer Learning

Song Liu
liu@ism.ac.jp
Institute of Statistical Mathematics
10-3 Midori-cho, Tachikawa, Tokyo, 190-8562, Japan.

Kenji Fukumizu
fukumizu@ism.ac.jp
Institute of Statistical Mathematics
10-3 Midori-cho, Tachikawa, Tokyo, 190-8562, Japan

Abstract

The concept of Transfer Learning frequently rises when one has already acquired a general-purpose classifier (e.g. human hand-writing recognizer) and want to enhance it on a similar but slightly different task (John’s hand-writing recognition). In this paper, we focus on a “lazy” setting where an accurate general purpose probabilistic classifier is already given and the transfer algorithm is executed afterwards. We propose a novel methodology to estimate the class-posterior ratio using a sparse parametric model. The new classifier therefore is obtained by multiplying the existing classifier with the approximated posterior ratio. We show the proposed method can achieve promising results through numerical and real-world experiments of text corpus and hand-writing digits classification.

1 Introduction

Supervised learning is to learn a classifier to predict the label information $Y$ from the input feature $X$. In many supervised learning applications, it is expected that the testing samples have slightly different patterns from the training samples. For example, a hand-writing recognition system is trained using dataset generated from a number of volunteers who have “regular writing style”, while the final product is deployed to recognize a specific user who may have special personal writing habit due to a different cultural background.

Specifically, we have a general-purpose task $T_Q$ and a large scale database $D_Q$ that is drawn from distribution $Q$ while the target task $T_P$ and dataset $D_P$ is drawn from another distribution $P$. Target-specific information in $P$ is not reflected in the classifier trained from $D_Q$. The transfer learning [9] [11] [4] usually refers to procedures that make use of the similarity between two learning tasks to build a superior classifier using both datasets $D_P$.
and $\mathcal{D}_Q$. In discriminative learning, such as logistic regression [2], the classifier is the class posterior $p(y|x)$, which is the conditional probability of class label given input feature. In this work, we focus on such probabilistic classification setting.

Since two learning tasks are similar and we may not have enough labelled samples in $\mathcal{D}_P$, the key focus of transfer learning is to “reuse” information as much as possible from the general-purpose learning task. A natural idea is to “borrow” informative samples from the $\mathcal{D}_Q$, while get rid of samples that are harmful. TrAdaBoost [3] follows this exact learning strategy to assign weights to samples from both $\mathcal{D}_P$ and $\mathcal{D}_Q$. By assigning high weights to samples contributes to the performance in the target task, and penalizing samples that “misleads” the classifier, TrAdaBoost reuses the knowledges from both datasets to construct an accurate classifier on the target task. The idea of importance sampling also gives rise to another set of methods to learn weights of samples by using density ratio estimation [13, 7]. Using unlabelled samples from both datasets, an importance weighting function can be learned. By plugging such function into the empirical risk minimization criterion [14], we can use samples from the $\mathcal{D}_Q$ as if they were samples from $\mathcal{D}_P$.

Another line of research follows the strategy of reusing the “model”, since if two tasks are similar, they must have similar parametric forms. As a result of this assumption, the learning algorithm must learn two models from two sets of data simultaneously. Regularization has been utilized to enforce the closeness between learned parameters [4]. More complicated structures, such as dependencies between task parameters are also used to construct a good classifier [11].

However, such setting faces a challenge in practice: the need of transfer may happen after the general-purpose classifier has already been built on a massive dataset using complex models (such as deep neural networks [6]) and one would like to avoid re-training a model on the general-purpose data since it may be costly. For example, one may have trained a hand-writing recognition system and would like to adapt such system to a small group of customers who have different writing habit. It would be ideal that we can come up with a rapid transfer learning algorithm directly built on the legacy classifier as a “patch”. We refer such strategy as the “Lazy Transfer Learning”.

More specifically, the lazy transfer learning consists of two separated stages: the general-purpose classifier learning and the target classifier transferring. The first stage provides an accurate classifier while the second stage is a light-weight algorithm “translates” the learned model on general-purpose dataset to a new model on target datasets. Obviously, the first learning stage is computationally intensive, since it involves a large (general-purpose) dataset and sophisticated modeling. The key to the success of such lazy approach is we only model the “difference” between two models in the second stage, and such “difference” may be considered as minor since we assume two tasks are similar. Therefore, rather than handling two complicated models, (see Figure 1(a)), we may handle a much simpler and easy-to-learn model in the second stage of learning, as long as it captures the “change of patterns”. Especially if such model contains less number of parameters, we may just need a few samples to train. (see Figure 1(b)). This is crucial since we can afford spending time on building a sophisticated model while developing a Machine Learning software, however, once
the software has been shipped to end users, we hope the learning software can be adapted very efficiently on any learning-enabled devices, such as mobile phones or tablets.

The key idea is simple: if a general-purpose probabilistic classifier $H$ has already been trained and available, we should turn to learn the posterior ratio, i.e., the ratio of conditional class posterior probabilities between $P$ and $Q$ then multiply it with the classifier $H$ to obtain a new “composite classifier”. Such transferring step is completely independent from the learning of $H$, i.e., the classifier $H$ is totally “invisible” to us, but still makes contribution in the final classification. One may keep updating $H$ at any time of need independently, and we later show that reducing the error of $H$ lowers the upper-bound of error given by such composite classifier.

The main contribution of this paper is that, we propose an algorithm for the later stage of transferring, and show it can be computed very efficiently using simple parametric models and convex optimization. We also conduct theoretical analysis showing this method is consistent.

This paper is organized as follows: In Section 2, we introduce the problem setting, followed by the motivation and the methodology in Section 3. The algorithm of learning the class-posterior ratio is given in Section 4. We prove the consistency of our posterior estimator in Section 5. Experimental results are shown in Section 6 and finally we conclude our work in Section 7.

2 Problem Setting

In this paper, we handle two sets of data: One is called general-purpose dataset $\mathcal{D}_Q = \left\{ \left(y_q^{(j)}, x_q^{(j)} \right) \right\}_{j=1}^{n'} \overset{i.i.d.}{\sim} Q$, where $Q$ is a joint distribution, $y \in \{1, \cdots, k\}$, and $x \in \mathbb{R}^d$. Another target dataset is also provided: $\mathcal{D}_P = \left\{ \left(y_p^{(i)}, x_p^{(i)} \right) \right\}_{i=1}^{n} \overset{i.i.d.}{\sim} P$ where we believe that the posterior function $p(y|x)$ in $P$ is “similar” to $q(y|x)$ in $Q$. However, $n \ll n'$.

\textsuperscript{1}From now on, single capital letter $P$ or $Q$ refer to the joint distributions of $(X, Y)$. 

\(\text{(a) Traditional Transfer Learning Framework: The learning of two tasks are coupled.}\)

\(\text{(b) Lazy Transfer Learning Framework: The learning of two tasks are de-coupled, and the second stage only models the incremental model.}\)
Our target is to learn an estimate $\hat{p}(y|x)$ of class-posterior and the predicted class is given by $\hat{y} = \arg\max_y \hat{p}(y|x)$.

Clearly, if $n$ is large enough, one may apply conventional Machine Learning algorithms to obtain a good estimate. In this paper, we focus on a scenario where $n$ is relatively small and $n'$ is sufficiently large. Thus, it is desirable if we can transfer information from a related task ($T_Q$) to boost the performance of our target classifier.

Different from previous settings, we focus on a lazy transfer setting: We wish to prepare a pre-trained classifier and defer the transferring to a later stage so that the transferring algorithm does not have to handle complex models and large amount of data. As it is mentioned in Section 1, such setting may find many applications which require rapid transfer after seeing data for $T_P$. In the next section, we illustrate the detailed methodology.

3 Lazy Transfer Modeling

At a glance, this problem setting is absurd, since if the target is to learn a posterior $p(y|x)$, we should directly go for it, rather than using a two-step procedure, as it is against the well-know Clearly, one should learn a classifier for $T_P$ directly if enough samples are available, since Vapnik’s principle [14] states:

When solving a problem of interest, do not solve a more general problem as an intermediate step.

However, we show that in the case of transfer learning, violating this principle may allow us use extra information provided by another dataset $D_Q$. The key idea lies with posterior decomposition.

3.1 Posterior Decomposition

Recall, the discriminative learning aims to learn a posterior $p(y|x)$ of a joint distribution $P$. Note that the posterior can be decomposed into

$$p(y|x) = \frac{p(y|x)}{q(y|x)} \cdot q(y|x),$$

where $\frac{p(y|x)}{q(y|x)}$ is the class posterior ratio, and the $q(y|x)$ is a general-purpose classifier.

This problem formulation leads to a simple transfer learning methodology: Model and learn the posterior ratio and general-purpose classifier separately, then later multiply them together as an estimate of the posterior.

Specifically, we use the parametric models $g(y, x; \theta)$ for the class posterior ratio and $h(y, x; \beta)$ for the posterior $q(y|x)$. The main interest of this paper is to propose an efficient way of learning such composite model using samples from $D_P$ and $D_Q$. 
### 3.2 Kullback-Leibler Divergence Minimization

A natural way of learning such a model is to minimize the Kullback-leibler (KL) divergence between the true posterior and our composite model. Here we first review the definition of conditional KL divergence:

**Definition 1 (Conditional KL Divergence).**

\[
\text{KL}[p||q] = P \log \frac{p(y|x)}{q(y|x)},
\]

We denote \( Pf \) as the short hand of the integral/sum of a function \( f \) over a probability distribution \( P \) on its domain.

Now, we proceed to obtain the following upper-bound of KL divergence from \( p \) to the composite model:

**Proposition 1 (Transfer Learning Upper-bound).** if

\[
K \geq \sup_{x,y} \left| \frac{p(y,x)}{q(y,x)} - 1 \right|
\]

then we have

\[
\text{KL}[p||g_\theta \cdot h_\beta] \leq \text{KL}[p||g_\theta q] + (1 + K)\text{KL}[q||h_\beta],
\]

**Proof.**

\[
\text{KL}[p||g_\theta \cdot h_\beta] = \text{KL}\left[p||g_\theta \cdot \frac{h_\beta}{q}\right] = \text{KL}[p||g_\theta \cdot q] + P \log q - P \log h_\beta
\]

\[
\leq \text{KL}[p||g_\theta \cdot q] + (1 + K)Q(\log q - \log h_\beta)
\]

\[
= \text{KL}[p||g_\theta \cdot q] + (1 + K)\text{KL}[q||h_\beta]
\]

Further,

\[
\text{KL}[p||g_\theta \cdot q] + (1 + K)\text{KL}[q||h_\beta]
\]

\[
\approx -\frac{1}{n} \sum_{i=1}^{n} \log g\left(y^{(i)}_p, x^{(i)}_p; \theta\right) - (1 + K)\frac{1}{n'} \sum_{j=1}^{n'} \log h\left(y^{(j)}_q, x^{(j)}_q; \beta\right) + C
\]

where \( C \) is a constant that is irrelevant to \( \theta \) or \( \beta \).

We may minimize the empirical upper-bound of KL divergence in order to obtain estimates of \( \theta \) and \( \beta \). \( K \) is an unknown constant (introduced in (2)) that illustrates the how dissimilar these two tasks are. Such upper-bound in (1) formalizes the common intuition that “if two tasks are similar, transfer learning should be easy,” since the more similar two tasks are, the smaller the \( K \) is, and the tighter the bound is.
Note that the minimizing (3) leads to two separate maximum likelihood estimation (MLE). The MLE of the second likelihood term of bound (3)
\[
\hat{\beta} = \arg \max_{\beta} \frac{1}{n'} \sum_{j=1}^{n'} \log h(y_q^{(j)}, x_q^{(j)}; \beta)
\]
leads to a conventional MLE of a posterior model, and has been well studied. $h$ can be efficiently modeled and trained using techniques such as logistic regression [2, 16]. As state before, we consider it is already given. However, maximizing the first likelihood term, a posterior ratio
\[
\hat{\theta} = \arg \max_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log g(y_p^{(i)}, x_p^{(i)}; \theta)
\]
is novel and is the key focus of this paper. In the next section, we show the modelling and learning of the posterior ratio is feasible and computationally efficient.

### 3.3 Posterior Ratio Model

Generally speaking, there is no unique way of modelling the posterior ratio, for example, we can either use the log-linear model: $g(y, x; \theta) \propto \exp(\theta^T f(y, x))$ or the linear-in-parameter model: $g(y, x; \theta) \propto \theta^T f(y, x)$, where $f$ is a vector-valued feature transform function. However, one thing that we should note is, since we are minimizing the KL divergence between $p(y|x)$ and $g(y, x; \theta)q(y|x)$, we need to make sure that $g(y, x; \theta)q(y|x)$ is a valid conditional probability, i.e., $\sum_y q(y|x) \frac{g(y, x; \theta)}{N(x; \theta)} = 1$.

For example, a self-normalizing log-linear model, should be written as the following form:
\[
g(y, x; \theta) := \frac{\exp(\theta^T f(y, x))}{N(x; \theta)},
\]
where $N(x; \theta)$ is defined as
\[
N(x; \theta) = \sum_{y \in \{1, \ldots, k\}} q(y|x) \exp(\theta^T f(y, x))
\]

However, we cannot directly evaluate the output value of this model, since we do not have access to $q(y|x)$. Therefore, we can only use samples from $D_Q$ to approximate the normalization term. The detailed technique will be introduced in the next section.

### 4 Estimating Posterior Ratio

Now we introduce the estimator of the class-posterior ratio $p(y|x)/q(y|x)$. In this section, we focus on the log-linear model introduced in (3). Let us substitute the model of (5) into
Figure 1: Approximate the $N(\theta, x)$ using nearest neighbours.

\[ \argmax_{\theta} \frac{1}{n} \sum_{i=1}^{n} \log g(y_p^{(i)}, x_p^{(i)}; \theta) = \frac{1}{n} \sum_{i=1}^{n} \theta^\top f(y_p^{(i)}, x_p^{(i)}) - \frac{1}{n} \sum_{i=1}^{n} \log N(\theta, x_p^{(i)}). \]

The normalization term needs to be evaluated in a pointwise fashion $N(\theta, x_p^{(i)}), \forall x_p^{(i)} \in D_P$. Note that if we have sufficient observations paired with each $x_p^{(i)}$, i.e. \( \{(x_p^{(i)}, y_q^{(j)})\}_{j=1}^{k} \sim Q \), such normalization can be approximated efficiently via sample average:

\[ N(\theta, x_p^{(i)}) \approx \frac{1}{k} \sum_{j=1}^{k} \exp (\theta^\top f(y_q^{(j)}, x_p^{(i)})), \]

However, in practice not many observed samples may be paired with $x_p^{(i)}$. Especially when $x$ is in a continuous domain, we may not observe any paired samples $y_q \sim Q_{Y|X=x_p^{(i)}}$ for any $x_p^{(i)} \in D_P$ (see Figure 1). We may consider using the neighbouring pairs $\{(x_p^{(i)}, y_q^{(j)})\}_{j=1}^{k} \sim Q$, where $x_q^{(j)}$ is a neighbour of $x_p^{(i)}$ to approximate $N(\theta, x_p^{(i)})$, which naturally leads to the idea of $k$-nearest neighbours ($k$-NN) estimation of such quantity:

\[ N(\theta, x_p^{(i)}) \approx \tilde{N}_{n', k}(\theta, x_p^{(i)}) = \frac{1}{k} \sum_{j \in N_{n'}(x_p^{(i)}, k)} \exp (\theta^\top f(y_q^{(j)}, x_p^{(i)})), \]

where $N_{n'}(x_p^{(i)}, k) = \{j \mid x_q^{(j)} \text{ is one of the } k \text{-NNs of } x_p^{(i)}\}$. The resulting optimization is

\[ \hat{\theta} = \argmin_{\theta} -\frac{1}{n} \sum_{i=1}^{n} \theta^\top f(y_p^{(i)}, x_p^{(i)}) + \frac{1}{n} \sum_{i=1}^{n} \log \frac{1}{k} \sum_{j \in N_{n'}(x_p^{(i)}, k)} \exp (\theta^\top f(y_q^{(j)}, x_q^{(j)})), \quad (6) \]
which is convex.
Moreover, in this work, we assume that posteriors are similar, so they may share many feature functions. The resulting posterior ratio may be sparse in classification features. Therefore, in this paper, we use a sparsity inducing norm together with the main objective function to obtain a sparse solution:

\[
\argmin_{\theta} -\frac{1}{n} \sum_{i=1}^{n} \theta^\top f(y_p^{(i)}, x_p^{(i)}) + \frac{1}{n} \sum_{i} \log \frac{1}{k} \sum_{j \in N_q(x_p^{(i)}, k)} \exp (\theta^\top f(y_q^{(j)}, x_q^{(j)})) + \lambda \|\theta\|_1, \tag{7}
\]

where the \(\lambda\) is a regularization term and can be chosen via likelihood cross-validation.

5 Consistency of the Estimator

First, let’s review and define a few notations. The posterior ratio model and its approximation using \(k\)-NN is defined as:

\[
g(y, x; \theta) := \frac{\exp (\theta^\top f(y, x))}{N(x; \theta)}, \quad g_{n'}(y, x; \theta) := \frac{\exp (\theta^\top f(y, x))}{N_{n', k}(x; \theta)},
\]

where \(N_{n', k}(x; \theta)\) is the \(k\)-NN estimation of \(N(x; \theta)\). Moreover, we define the estimated and true parameter as:

\[
\hat{\theta} = \arg\max_{\theta} \ell(\hat{\theta}; D_P, D_Q) = P_n \log g_{n'}(y, x; \theta), \quad \theta^* = \arg\max_{\theta} P \log g(y, x; \theta),
\]

where \(P_n\) is the empirical measure of distribution \(P\).

Assumption 1 (Bounded Posterior Ratio Model). There exists \(M_{\text{max}} > 1\), so that \(|\theta^\top f(y, x)| \leq \log(M_{\text{max}})\). Moreover, \(\theta\) is in a totally bounded metric space and \(\|f\|_{\infty} \leq C\) where \(0 < C < \infty\).

As an important consequence, \(\exp (\theta^\top f(y, x))\), \(N(x; \theta)\) and \(N_{n', k}(x; \theta) \in \left[\frac{1}{M_{\text{max}}}, M_{\text{max}}\right]\), and therefore, the posterior ratio model is always bounded by constants. It is a reasonable assumption as the posterior ratio measures the “differences” between two tasks, the true posterior ratio must be close to one if two tasks are similar.

Assumption 2 (Bounded Covariate Shift). \(\frac{p(x)}{q(x)} \leq R_{\text{max}}\).

The support between \(P\) and \(Q\) must overlap. If samples in \(D_P\) distribute completely differently from those in \(D_Q\), it does not make sense to expect the transfer learning method would work well.

Assumption 3 (Identifiability). \(\theta^*\) is the unique global maximizer of the population objective function \(P \log g(y, x; \theta)\), i.e. for all \(\epsilon > 0\),

\[
\sup_{\theta, \|\theta - \theta^*\| \geq \epsilon} P \log g(y, x; \theta) < P \log g(y, x; \theta^*).
\]
Then we have the following theorem that states our posterior ratio estimator is consistent.

**Theorem 1.** Suppose for each \( x \), the random variable \( \| X - x \| \) is absolutely continuous. If \( n \to \infty, n' \to \infty, k_n'/\log n' \to \infty \) and \( k_n/n' \to 0 \), where \( k_n' \) is the sample dependent version of \( k \), the number of nearest neighbors used in \( k \)-NN approximation. Then under above assumptions, \( \hat{\theta} \overset{p}{\to} \theta^* \). Further \( \ell(\hat{\theta}; D_P, D_Q) \overset{p}{\to} KL[p\|q] \).

The proof relies on the following lemma:

**Lemma 1.** Under all assumptions stated above, if \( n \to \infty, n' \to \infty, k_n'/\log n' \to \infty \) and \( k_n/n' \to 0 \). Then \( \sup_\theta \left[ P_n \log g_n(y, x; \theta) - P \log g(y, x; \theta) \right] \overset{p}{\to} 0 \), i.e. the error caused by approximating objective using samples converges to \( \theta \) uniformly in probability.

One of the key steps in the proof is to decompose the above empirical approximation error of the objective function into: approximation error caused by using samples from \( P + \) modelling error caused by \( k \)-NN using samples from \( Q \). It can be observed that the bound of density ratio \( R_{\max} \) also contributes to the error. The complete proof is included in the appendix.

### 6 Experiments

After obtaining the \( \hat{\theta} \) for the posterior ratio, the class posterior probability of the target task \( T_P \) can estimated via a simple multiplication: \( \hat{p}(y|x) = g(y, x; \hat{\theta}) \cdot h(y, x; \hat{\beta}) \) where \( \hat{\beta} \) is th parameter of a probabilistic classifier. In the following experiments, we use logistic regression to obtain \( \hat{\beta} \), though in practice, one may use many other alternatives to obtain a probabilistic classifier, such as [12].

#### 6.1 Numerical Experiments

The first experiment examines the convergence of our estimator of conditional KL divergence i.e. the estimated parameter \( \hat{\theta} \to \theta^* \) when number of samples increases. If the model is identifiable, then

\[
\ell(\hat{\theta}; D_P, D_Q) - KL[p\|q] \to 0.
\]

The first experiment is designed to illustrate such property with a parametric model. We draw two classes of samples from two Gaussian distributions with different means for \( P \) and \( Q \). Specifically, for \( y = \{-1, 1\} \), we construct \( P \) and \( Q \) as follows:

\[
q(x|1) = N(2, 1), \quad q(x|-1) = N(-2, 1), \\
p(x|1) = N(1.5, 1), \quad p(x|-1) = N(-1.5, 1), \quad y(1) = .5, \quad y(-1) = .5
\]

We draw 5k samples from distribution \( Q \) as \( D_Q \). Also we draw samples from \( P \) as \( D_P \), and train a posterior ratio \( g(y, x; \hat{\theta}) \), where the feature function \( f \) is defined as \( f(x, y) := \)
(a) Dataset for Convergence Test

(b) Estimated conditional KL divergence: \( \ell(\theta; D_P, D_Q) \)

(c) Dataset for Probabalistic Classification.

(d) Lazy Transfer Learning versus simple logistic regression.

(e) True posterior ratio and the estimated ratio. Non-zero Gaussian centers are also plotted in and simple logistic regression. Green, weighted by estimated parameter values.

(f) Hold-out likelihood of Lazy Transfer Learning and simple logistic regression.
By varying $n$, the number of samples in $\mathcal{D}_P$, we may create a plot for $\ell(\hat{\theta}; \mathcal{D}_P, \mathcal{D}_Q)$, and compare it with the true KL divergence from posterior $p$ to $q$ (blue horizontal line). The averaged plot with standard error computed from 25 runs are shown in Figure 2(b) (blue). As a comparison, we draw 50k samples from $Q$, and rerun the estimation for posterior ratio for different $n$ once more. The averaged plot is shown in Figure 2(b) in red.

The result shows, our estimator does converge to the true KL divergence, and the estimation error shrinks as $n \to \infty$. Increasing $n'$ also helps in improving the convergence (comparing the blue error bar with the red error bar). However, such improvement is not as significant as increasing $n$.

Now we test the performance of Lazy Transfer Learning in probabilistic classification setting. From now on, we introduce two baseline methods: LogiP logistic regression on $\mathcal{D}_P$. LogiQ logistic regression on $\mathcal{D}_Q$. We construct $P$ and $Q$ as follows:

$$q(x | 1) = 0.5 \mathcal{N}(-4, 1) + 0.5 \mathcal{N}(4, 1), \quad q(x | -1) = 0.5 \mathcal{N}(0, 1) + 0.5 \mathcal{N}(8, 1), y(-1) = .5$$

$$p(x | 1) = 0.5 \mathcal{N}(-4, 1) + 0.5 \mathcal{N}(4, 1), \quad p(x | -1) = 0.5 \mathcal{N}(0, 1) + 0.5 \mathcal{N}(6, 1), y(1) = .5.$$  

We draw 10k samples from $Q$ as $\mathcal{D}_Q$ and $k$ is chosen to minimize the 5-fold cross-validated MSE over $\mathcal{D}_Q$. In this experiment, the samples from each class are drawn from a two-modal Gaussian mixture, and therefore, the previous feature function $f$ would not work. For all three methods, we use a more flexible feature function: $f_i(x, y) = \delta(y, y_i) \exp\left(\frac{\|x - x_i\|^2}{2\sigma^2}\right)$, where $f_i$ is the $i-$th output of the vector-valued feature function $f$, $x_i$ is a preselected kernel center, $\sigma$ is called the bandwidth of this kernel and $\delta$ is the indicator function. In this experiment, we use all $x_p \in \mathcal{D}_P$ as kernel centers for our posterior ratio estimator and LogiP. For LogiQ, we randomly select 100 $x_q \in \mathcal{D}_Q$ as kernel centers.

First, we show an example where $n = 21$, on Figure 2(d). The predicted posterior $p(1|x)$ via both Lazy Transfer Learning (red dotted line) and LogiP (gray dotted line) on dataset $\mathcal{D}_P$ (without using information from $\mathcal{D}_Q$) is illustrated. The learned $\hat{h}(1, x)$ is shown in pink and the ground truth of $p(y|x)$ is marked in black. It can be seen that the proposed Lazy Transfer Learning has generally learned a posterior function that is pretty close to the true function. In comparison, the posterior learned by logistic regression on $\mathcal{D}_P$ fits poorly to the true posterior, due to insufficient number of samples. The sharp changes in posterior (see, e.g. from -2 to 2) indicate that the logistic regression was “tricked” to believe that there exists a sharp decision boundary between positive and negative samples. In fact, if plenty samples are observed, the posterior around that region should be smooth. Thanks to the information provided in $\mathcal{D}_Q$, the proposed method successfully recovers the correct shape of posterior function via Lazy Transfer Learning.

The learned posterior ratio of the positive class was shown in Figure 2(e). The learned ratio is very close to 1 in most regions along the axis, since the “transfer” happens only within a very restricted region (i.e. two posteriors are similar). This explains the superiority of the proposed method: For the regions that two posteriors do not differ, insufficient number of samples does not harm the performance of the proposed method, since regularization tends to

---

2We introduce our parameter tuning approach in the appendix.
Finally, we compare the hold-out likelihood (HOLL) between the proposed method and the logistic regression in Figure 2(f). The hold-out likelihood is computed over 5k testing samples drawn from \( P \). Again, we vary the size of \( D_P \) and plot the averaged HOLL over 25 runs. The result shows that the proposed method can always achieve a larger HOLL however the LogiP gradually catches up after more data \( D_P \) is provided. This result also shows that the proposed method has very good performance even over small number of samples.

6.2 Real-world Data Experiments

In this section, we compare our method with two representative methods of different strategies: TrAdaBoost [3] is based on re-sampling technique, while Reg [4] assumes parametric similarity between tasks. The proposed method requires an estimator of \( q(y|x) \) which is again provided by LogiQ. We conduct experiments on two datasets:

**20 Newsgroup:** The first experiment is run on a text corpus collected from 20 news groups, where the task is classification. Features are built from bag-of-words data with tf-idf preprocessing. In order to create transferring task, we construct \( D_Q \) by using “rec” category (including 5 sub categories) as negative samples, while choosing “sci.electronics”, one of the sub-categories in “sci” as positive samples. Then we construct \( D_P \) using another sub-category in “sci” as positive samples, and again sample the negative points from “rec” category. We set \( n' = 1000 \) and plot the error rate against \( n \). Number of positive and negative samples are balanced by construction in both datasets.

**MNIST Handwriting Digits:** MNIST is a hand-writing digits dataset that includes vectorized features from grey-scale images where the task is to classify digits. We reduce dimensions of data by averaging the 16-adjacent non-overlapping elements in the vector resulting 49 dimensional vector. We construct \( D_Q \) by using digit 0 as negative class, while using digit \( T_1 \) as positive class. For \( D_P \), we again use digit 0 as negative class, and a digit \( T_2 \) as positive class. In other words, \( D_P \) and \( D_Q \) are different only in positive samples, so we can consider two tasks are “similar” to each other. We set \( n' = 10,000 \) and plot the error rate against \( n \).

Error rates with standard errors of 20 runs are reported in Figure 2. All error rates drop as the size of \( D_P \) increases. As to 20 newsgroup dataset (see Figure 2(a), 2(b) and 2(c)), we can see that the proposed method always achieves the lower error rate than TrAdaboost and Reg, and significant improvement from LogiQ baseline. For the experiments of MNIST, we consider two settings. First, we set \( T_1 = 6 \) and \( T_2 = 8 \). Error rates are shown in Figure 2(d). Again, the proposed method can achieve the lowest error rate as well as Reg. However, when we set \( T_1 = 3 \) and \( T_2 = 5 \) in Figure 2(e) LogiP can achieve very low error rate. Noting that the simple logistic regression on \( D_Q \) cannot provide an accurate estimate on task \( P \) (black dotted line), therefore, the power of “transferring” is very limited. Nonetheless, the proposed method can still achieve the similar performance as LogiP, while the performance of TrAdaBoost cannot improve much from the baseline of LogiQ.

We increase the difficulty of tasks by classifying multiple digits (Figure 2(f)). We use the Gaussian kernel model for \( g \) (as described in Section 6.1), while still using the linear
LogiQ for learning \( h \). Though it is unfair and very time-consuming to train, we allow Reg to use Gaussian kernel model on both tasks, since its regularization requires both posteriors \( p(y|x) \) and \( q(y|x) \) to have the same model. Result shows, our method can achieve the same performance as Reg where a more flexible model is used.

## 7 Conclusion and Future Works

In this paper, we proposed a novel problem setting of transfer learning, namely “Lazy Transfer Learning”, where a general-purpose probabilistic classifier has already been provided. An efficient method for learning the posterior ratio was illustrated and tested over multiple experiments. The result shows the proposed method can utilize the information from the general-purpose learning task to improve its performance on the target task.

This work also opens room for future studies. For example, we have only considered the log-linear model under KL distance. In contrast, linear model with \( \chi^2 \)-divergence [10] may be a more computationally attractive alternative. Also, except \( k-\text{NN} \), other regression methods can also be used to approximate the normalization term. It is interesting to compare the performance between different choices.
[1] D. WK Andrews. Generic uniform convergence. *Econometric theory*, 8(02):241–257, 1992.

[2] D. R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 215–242, 1958.

[3] W. Dai, Q. Yang, G. R. Xue, and Y. Yu. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning*, pages 193–200. ACM, 2007.

[4] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 109–117, New York, NY, USA, 2004. ACM.

[5] L. Györfi. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2002.

[6] G. Hinton, S. Osindero, and YW Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

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

[8] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22:79–86, 1951.

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

[10] K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine*, 50:157–175, 1900.

[11] R. Raina, A. Y. Ng, and D. Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 713–720. ACM, 2006.

[12] M. Sugiyama. Superfast-trainable multi-class probabilistic classifier by least-squares posterior fitting. *IEICE Transactions on Information and Systems*, 93(10):2690–2701, 2010.

[13] M. Sugiyama, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*. Curran Associates, Inc., 2008.
[14] V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, NY, USA, 1998.

[15] L. Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer Publishing Company, Incorporated, 2010.

[16] J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. In *Advances in neural information processing systems*, pages 1081–1088, 2001.
Appendix, Proof for Lemma 1

Proof. First, we decompose supremum of the error:

$$\sup_\theta P_n \log g_n'(y, x; \theta) - P \log g(y, x; \theta)$$

$$= \sup_\theta P_n \theta^\top f(y, x) - P \log N_n'(\theta; x) - \left[ P \theta^\top f(y, x) - P \log N(\theta; x) \right]$$

$$\leq \sup_\theta (P_n - P) \theta^\top f(y, x) + \sup_\theta P_n \log N_n' - P \log N(\theta; x)$$

$$\leq \sup_\theta (P_n - P) \theta^\top f(y, x) + \sup_\theta \left[ (P_n - P) \log N_n' - P \log N(\theta; x) \right]$$

The first two terms are relatively easy to bound. The uniform law of large numbers can be applied to show the first two terms converges to 0 in probability, since i. \( \theta \) is compact, ii. both \( \theta^\top f(y, x) \) and \( \log N_n' \) are continuous over \( \theta \), iii. both above functions are Lipschitz continuous as we will show later. As to the third term, we need to use the following inequality:

$$\log a - \log b \leq \frac{a-b}{b}.$$
for all distributions \((Z, X)\), where \(\mu(x)\) is the probability measure of \(x\).

From Jensen’s inequality, we have

\[
\left\{ \int \left| \frac{1}{k} \sum_{j \in N_{n',k}(x)} z_j - E[Z|X = x] \right| d\mu(x) \right\}^2 \leq \int \left| \frac{1}{k} \sum_{j \in N_{n',k}(x)} z_j - E[Z|X = x] \right|^2 d\mu(x),
\]

and it can be seen that the right hand side also converges to 0 in probability. By using the continuous mapping theorem, we can finally show that \(\int \left| \frac{1}{k} \sum_{j \in N_{n',k}(x)} z_j - E[Z|X = x] \right| d\mu(x)\) converges to 0 in probability.

We let \(Z_\theta = \exp(\theta^T f(Y, X))\) be a new random variable and thus we have samples \(\{z_{\theta,j}, x_i\}_{i=1}^n \sim (Z_\theta, X)\), and

\[
|Q (N_{n'}(\theta; x) - N(\theta; x))| \leq \int \mu(x) \left| \frac{1}{k} \sum_{j \in N_{n',k}(x)} z_{\theta,j} - E[Z_\theta|X = x] \right| d\mu(x).
\]

By applying the Theorem 2 we can conclude, such \(Q \ N_{n'}(\theta; x) - N(\theta; x)\) converges 0 in probability for all distribution \((Z_\theta, X)\) indexed by parameter \(\theta\). Next, we verify the SE of \(Q \ N_{n'}(\theta; x) - N(\theta; x)\). Given Assumption 1, we have

\[
Q \ {N_{n'}(\theta; x) - N(\theta; x)} = Q \ {N_{n'}(\theta; x) - N_{n'}(\theta'; x) + N(\theta'; x) - N(\theta; x)} \\
\leq 2M_{\max}\|f\|\infty\|\theta - \theta'\|2. \tag{8}
\]

The last line is due to Mean-value Theorem:

\[
\exp(\theta^T f(y, x)) - \exp(\theta'^T f(y, x)) \leq \|\theta - \theta'|\|f(y, x)\exp(\bar{\theta}^T f(y, x))\| \\
\leq \|\theta - \theta'|\|f\|\infty M_{\max}
\]

In fact, (8) shows the function \(Q \ (N_{n'}(\theta; x) - N(\theta; x))\) is Lipschitz continuous with respect to \(\theta\), and according to Lemma 2 in [1], it implies SE.

Now we can utilize the property of i. boundedness of \(\theta\), ii. SE and iii. universal consistency to show that

\[
\sup_\theta Q \ (N_{n'}(\theta; x) - N(\theta; x)) \xrightarrow{p} 0.
\]

After obtaining Lemma 1, the rest is similar to the proof of Theorem 9.13 in [15]. Let \(M(\theta) := P \log g(y, x; \theta)\) and \(M_{n,n'}(\theta) := P_n \log g_{n'}(y, x; \theta)\) and

\[
M(\theta^*) - M(\hat{\theta}) = M(\theta^*) - M_{n,n'}(\hat{\theta}) + M_{n,n'}(\hat{\theta}) - M(\hat{\theta}) \\
\leq M(\theta^*) - M_{n,n'}(\theta^*) + M_{n,n'}(\hat{\theta}) - M(\hat{\theta}) \\
\leq M(\theta^*) - M_{n,n'}(\theta^*) + \sup_\theta M_{n,n'}(\theta) - M(\theta).
\]
The last line converges to 0 in probability is proved in Lemma 1. Therefore, we can write:

$$\forall \epsilon > 0, P(M(\theta^*) - M(\hat{\theta}) > \epsilon) \to 0.$$ 

Due to Assumption 3 for an arbitrary choice of $\epsilon_0 > 0$, if $\|\theta^* - \hat{\theta}\| \geq \epsilon_0$, there must be an $\epsilon > 0$, so that $M(\theta^*) - M(\hat{\theta}) > \epsilon$. Therefore, we conclude

$$\forall \epsilon_0 > 0, P(\|\theta^* - \hat{\theta}\| \geq \epsilon_0) \leq P(M(\theta^*) - M(\hat{\theta}) \geq \epsilon) \to 0.$$ 

Also, $M_n(\hat{\theta}) - M(\theta^*) = M_n(\hat{\theta}) - M(\hat{\theta}) + M(\hat{\theta}) - M(\theta^*)$. Due to Lemma 1 it converges to 0 in probability. Therefore, we have $\ell(\hat{\theta}; D_F, D_Q) \overset{P}{\to} KL[p\|q].$

**Tuning Parameters in Posterior Ratio Estimation**

**k in k-NN:** As it is mentioned in Section 6.1, $k$ is tuned via 5-fold cross validation, and is based on the testing criterion:

$$\text{MSE}(\theta) = \frac{1}{|D_{\text{Hold-Out}}|} \sum_{j \in D_{\text{Hold-Out}}} \left( \exp(\theta^T f(y_q^{(j)}, x_q^{(j)})) - \frac{1}{k} \sum_{jj \in N(x_q^{(j)})} \exp(\theta^T f(y_q^{(jj)}, x_q^{(jj)})) \right)^2,$$

where $D_{\text{Hold-Out}}$ is a holdout dataset. However, such value depends on $\theta$ and it changes every iteration during the gradient decent. Instead of tuning $k$ after each iteration, we follow a simple heuristics: 1) Fix $k$ and run gradient descent. 2) choose a suitable $k$ that minimizes (9). 1) and 2) are repeatedly carried out until converge. Such heuristics have good performance in experiments.