Privacy-preserving Transfer Learning for Knowledge Sharing

Xiawei Guo[1] **Quanming Yao**[1] **WeiWei Tu**[1] **Yuqiang Chen**[1] **Wenyuan Dai**[1] **Qiang Yang**[1]
[1]4Paradigm Inc, Beijing, China

Abstract

In many practical machine-learning applications, it is critical to allow knowledge to be transferred from external domains while preserving user privacy. Unfortunately, existing transfer-learning works do not have a privacy guarantee. In this paper, for the first time, we propose a method that can simultaneously transfer knowledge from external datasets while offering an ϵ-differential privacy guarantee. First, we show that a simple combination of the hypothesis transfer learning and the privacy preserving logistic regression can address the problem. However, the performance of this approach can be poor as the sample size in the target domain may be small. To address this problem, we propose a new method which splits the feature set in source and target data into several subsets, and trains models on these subsets before finally aggregating the predictions by a stacked generalization. Feature importance can also be incorporated into the proposed method to further improve performance. We prove that the proposed method has an ϵ-differential privacy guarantee, and further analysis shows that its performance is better than above simple combination given the same privacy budget. Finally, experiments on MINST and real-world RUIJIN datasets show that our proposed method achieves the start-of-the-art performance.

1 Introduction

Machine learning today relies on large quantities of high quality data. However, due to the expense and difficulties in data acquisition, often, one cannot have a sufficient amount of training data. Thus, one may want to make use of the external datasets that are provided by other data-rich organizations. Such scenario frequently appears in many real-world applications. For example, in Figure 1, small or newly established hospitals may not have sufficient labeled medical records, and they may wish to borrow some knowledge from large and public hospitals to boost the performance of their prediction models. Transfer learning (Pan and Yang 2010) is a proven tool for this purpose, as an established hospital may already have much experience in the form of labeled medical data, which can be transferred to a newly established medical service.

When knowledge transfer is conducted, data privacy becomes a serious concern. Recently, there even are several laws for the privacy issue. One of the most famous is Europe's General Data Protection Regulation (GDPR) [1], which is a regulation to regulate the protection on private data and on data transmission between organizations. Such regulations and public requirements place raise challenges for cross-organizational transfer learning. When applying transfer learning, the source and target datasets may belong to different institutes or companies. As a result, the privacy of users in a source organization may be compromised at the target organization. As far as we know, there is no existing transfer learning algorithm designed for solving this problem. We believe that it is time for machine learning researchers to design new methods to tackle this problem and make transfer learning compliant with the laws.

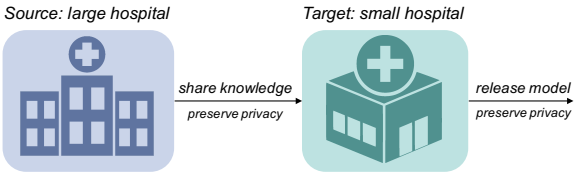


Figure 1: The knowledge sharing problem from large hospitals (source) to the small ones (target). Privacy needs to be protected for both the source and target.

In the past, researchers have designed various ways to ensure protection of privacy in data publishing. The theory of differential privacy (Dwork et al. 2006b; Dwork 2008) has been developed as a standard for ensure the privacy of data when data is exchanged between organizations. To design an algorithm with a privacy-preserving guarantee, carefully designed noise is often added to the original data to disambiguate the analytic algorithms. However, for machine learning, such injection of noise often incurs a major degradation of learning performance. As a result, many machine learning algorithms have been modified to achieve the differential privacy while preserving the performance of machine learning, including logistic regression (Chaudhuri, Monteleoni, and Sarwate 2011), tree models (Emekçi et al. 2007; Jagannathan, Pillaipakkamnatt, and Wright 2009; Fong and Weber-Jahnke 2012), and deep neural networks (Shokri and Shmatikov 2015; Abadi et al. 2016), etc. In

The first two authors make equal contributions. Correspondence to Q. Yao: yaoquanming@4paradigm.com

[1] https://en.wikipedia.org/wiki/General_Data_Protection_Regulation

our scenario, we do not consider tree models, as they are usually difficult to transferred. Deep neural networks, is also not a good choice, as it is hard to interpreted and can be a problem when facing the law (e.g., GDPR). However, linear models is simple and easy to understand, and its differentially private variants also have rich and rigorous theoretical guarantee (Chaudhuri, Monteleoni, and Sarwate 2011; Bassily, Smith, and Thakurta 2014; Hamm, Cao, and Belkin 2016; Kasiviswanathan and Jin 2016). These considerations motivate us in using linear models for privacy-preserving transfer learning.

In this paper, we start from the hypothesis transfer learning method of (Kuzborskij and Orabona 2013), which we use to combine with private-preserving logistic regression (Chaudhuri, Monteleoni, and Sarwate 2011). We show how to innovatively integrate these methods to solve the knowledge sharing problem with privacy preserving guarantee. However, the simple-minded combination of transfer learning with privacy preserving constraints can suffer from poor learning performance. Specifically, to preserve the privacy, the noise added to the source learning models can be too high when the target samples sizes are small or when the feature dimensions are high. The challenge is how to make the best balance. This is a serious problem for transfer learning to the target domain. As a summary, contributions are as follows:

- We propose a novel method, which is based on HTL, to solve the knowledge sharing problem from the source to the target. This method draws a best balance between privacy-preserving concern, transfer learning performance and target-domain data sizes;
- We prove the algorithm has an ϵ-differential privacy guarantee for both the source and target. We also analyze the generalization performance of the proposed method, which shows it is indeed better than the simple combination when privacy budgets are the same; and
- Except for the popular MNIST dataset, we also test our method on a real-world RUIJIN dataset. This dataset contains medical records from thousands of people, and is collected from hospitals across the whole China. The results show, given the same privacy budget, our method can achieve much better performance than other state-of-the-arts.

The rest of the paper is organized as follows. Section 2 gives related works; the proposed algorithm is in Section 3; experiments are performed in Section 4; and concluding remarks are given in Section 5.

Notation In this paper, vectors are denoted by lowercase boldface. We use subscript $(\cdot)_s$ and $(\cdot)_t$, e.g., $\mathbf{w}_s$ and $\mathbf{w}_t$, to denote variables in the source and target domain. Then, we use the superscript $(\cdot)^{(k)}$ to denote the variables associated with the kth split of features, e.g., $q^{(k)}$. Parameters with a privacy guarantee is indicated by a "bar" above, e.g., $\bar{\mathbf{w}}$.

2 Related Works

1. Differential Privacy. Differential privacy (Dwork et al. 2006b; Dwork and Roth 2014) is established as a rigorous standard to guarantee privacy for algorithms that access to the private data. We say that the algorithm preserves ϵ-differential privacy if one entry difference in the dataset dose not affect the likelihood of a specific output of the algorithm by more than ϵ. A formal definition is defined as follows.

Definition 1. (Dwork et al. 2006b) *A randomized mechanism M is ϵ-differentially private, if for all output t of M, and for all databases $\mathcal{D}_1$ and $\mathcal{D}_2$ which differ by at most one element, we have $Pr(M(\mathcal{D}_1) = t) \leq e^{\epsilon} Pr(M(\mathcal{D}_2) = t)$.*

The parameter ϵ is called privacy budget. To meet an ϵ-differential privacy guarantee, usually, carefully generated noise needs to be added into the learning algorithm. Smaller ϵ provides stricter privacy guarantee but usually requires adding more noise, and leads to larger decreasing on the learning performance.

2. Privacy-Preserving Logistic Regression (PLR). The state-of-the-art privacy-preserving learning with logistic regression (LR) is proposed in (Chaudhuri, Monteleoni, and Sarwate 2011). Given dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$ where $\mathbf{x}_i$s are samples and y_is are corresponding labels, they design an ϵ-differentially private variant for

$$\min_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{w}^\top \mathbf{x}_i, y_i) + \lambda g(\mathbf{w}), \tag{1}$$

where $f(\alpha, \beta) = \log(1 + e^{-\alpha\beta})$ is the logistic loss, $\mathbf{w}$ is the learning parameter and $\lambda > 0$ is the hyper-parameter. The following assumption is made on (1).

Assumption 1. *The regularizer g is 1-strongly convex* [2].

However, directly obtaining $\mathbf{w}$ from (1) fails to reach this goal (Dwork et al. 2006b). Thus, Chaudhuri et.al. 2011 proposed to change the objective as

$$\min_{\mathbf{w}} \bar{F}(\mathbf{w}; \mathcal{D}, \mathbf{b}, \Delta) + \lambda g(\mathbf{w}), \tag{2}$$

where $\bar{F}(\mathbf{w}; \mathcal{D}, \mathbf{b}, \Delta) = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{w}^\top \mathbf{x}_i, y_i) + \frac{\mathbf{b}^\top \mathbf{w}}{n} + \frac{1}{2}\Delta\|\mathbf{w}\|_2^2$. Thus, two extra terms, i.e., random vector $\mathbf{b}$ and quadratic term $\frac{1}{2}\Delta\|\mathbf{w}\|_2^2$, are added to (1). Specifically, Δ is a constant depending on λ, n and ϵ, which is a requirement of the proof; $\mathbf{b}$ is drawn from the [3] distribution $h(\mathbf{b}) \propto \exp(\frac{\epsilon'}{2}\|\mathbf{b}\|_2)$ and the noise level is determined by ϵ. The complete mechanism based on (2) is in Algorithm 1, and its privacy guarantee is in Proposition 1.

Proposition 1. (Chaudhuri, Monteleoni, and Sarwate 2011) *Algorithm 1 has the ϵ-differential privacy guarantee if Assumption 1 is satisfied.*

However, in a machine learning model, we are also concerned with model performance. Having a good privacy guarantee does not mean that the model has a good learning performance; in fact, in many practices, the performance

[2] If a function f is μ-strongly convex, then $f(\alpha\mathbf{w}+(1-\alpha)\mathbf{u}) \leq \alpha f(\mathbf{w}) + (1-\alpha)f(\mathbf{u}) - \frac{\mu}{2}\alpha(1-\alpha)\|\mathbf{w}-\mathbf{u}\|_2^2$ for any $\alpha \in (0, 1)$

[3] To draw $\mathbf{b}$, one can first pick $\|\mathbf{b}\|_2$ from the Gamma distribution $\Gamma(d, \frac{2}{\epsilon'})$, and pick a unit vector $\mathbf{a}$ uniformly, then $\mathbf{b} = \|\mathbf{b}\|_2\mathbf{a}$ (Chaudhuri, Monteleoni, and Sarwate 2011).

Algorithm 1 PLR($\mathcal{D}, \epsilon, \lambda, g$): Privacy-preserved learning with logistic regression.

Require: privacy budget ϵ, a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, hyper-parameter λ and regularizer g.

$\epsilon' = \epsilon - \log(1 + \frac{1}{2n\lambda} + \frac{1}{16n^2\lambda^2})$;
if $\epsilon' > 0$ **then**
  $\Delta = 0$
else
  $\Delta = (4n(\exp(\frac{\epsilon}{4}) - 1))^{-1} - \lambda$ and $\epsilon' = \frac{\epsilon}{2}$;
end if
scale $\mathcal{D}$ so that for all i, $\|\mathbf{x}_i\|_2 \leq 1$;
pick a random vector $\mathbf{b}$ from $h(\mathbf{b}) \propto \exp(\frac{\epsilon'}{2}\|\mathbf{b}\|_2)$;
obtain $\bar{\mathbf{w}}$ by solving (2);
return $\bar{\mathbf{w}}$ (ϵ-differential privacy guarantee).

dramatically degrades. Thus, given the same privacy budget, algorithms with a better learning guarantee are desired. Proposition 2 is offered for such guarantee of Algorithm 1.

Proposition 2. (Chaudhuri, Monteleoni, and Sarwate 2011) *Let $g(\cdot) = \frac{1}{2}\|\cdot\|_2$ and $\hat{\mathbf{w}}$ be a reference predictor and P be the data distribution. Then, there exists a constant C such that for $\delta > 0$, if the training examples from $\mathcal{D}$ are i.i.d. drawn according to P, and*

$$n > C_1 \max\left(\frac{\|\hat{\mathbf{w}}\|_2^2 \log(\frac{1}{\delta})}{\epsilon_g^2}, \frac{d\log(\frac{d}{\delta})\|\hat{\mathbf{w}}\|_2}{\epsilon_g \epsilon}, \frac{\|\hat{\mathbf{w}}\|_2^2}{\epsilon_g \epsilon}\right), \quad (3)$$

then the output $\bar{\mathbf{w}}$ of Algorithm 1 satisfies $Pr[L(\bar{\mathbf{w}}, P) \leq L(\hat{\mathbf{w}}, P) + \epsilon_g] \geq 1 - \delta$ where $L(\mathbf{w}; P) = \mathbb{E}_{(\mathbf{x},y)\sim P}\left[\ell(\mathbf{w}^\top \mathbf{x}, y)\right]$ is the expected loss of predictor $\mathbf{w}$ over distribution P.

Proposition 2 shows how much samples are needed in order to reach ϵ_g error as compared with the desired classifier. Basically, the smaller the R.H.S. in (3) the better the learning performance. Comparing with non-privacy model (1), which only needs $n > C_1\|\hat{\mathbf{w}}\|^2 \log(\frac{1}{\delta})/\epsilon_g^2$ samples to reach the same error ϵ_g (Shalev-Shwartz and Srebro 2008), we can see there are two extra terms that depend on ϵ in the $\max$ operator in Proposition 2. Thus, the learning performance of Algorithm 1 is indeed deteriorated.

3. Hypothesis Transfer Learning (HTL). Transfer learning (Pan and Yang 2010) is a powerful and promising method to extract useful knowledge from a source domain to a target domain. In this paper, we consider hypothesis transfer learning (HTL) (Orabona et al. 2009; Kuzborskij and Orabona 2013; Kuzborskij and Orabona 2017). Using (1) as an example, HTL improves the performance on target domain via adding an extra regularization, i.e.,

$$\mathbf{w}_t = \arg\min_{\mathbf{w}} \frac{1}{n_t}\sum_{i=1}^{n_t} f(\mathbf{w}^\top \mathbf{x}_i, y_i) + \lambda g(\mathbf{w}) + \frac{\eta}{2}\|\mathbf{w} - \mathbf{w}_s\|_2^2, \quad (4)$$

where $f(\alpha, \beta) = \log(1 + e^{-\alpha\beta})$ is the logistic loss, η is a hyper-parameter, $\mathcal{D}_t$ is the data in the target domain and $\mathbf{w}_s$ is the predictor obtained from the source domain.

Although there are other techniques for transfer learning such as domain adaptation and feature representation transfer (Pan and Yang 2010), as we will show in Section 3, HTL is a very good choice because it is a general method that can be incorporated with the privacy guarantee very well.

4. Learning from Multi-parties. Finally, we discuss about learning privately from multi-party data (Pathak, Rane, and Raj 2010), which is the most related scenario to ours. In this case, each party has its own private data, and the task here is to obtain final predictions using data from all parties. There are two lines of researches. The first one is to train classifiers in each party locally, then the problem is how to privately combine these predictions. A carefully designed protocol is proposed in (Pathak, Rane, and Raj 2010) to solve this problem. The performance is surpassed by (Hamm, Cao, and Belkin 2016), which used a second level classifier to combine different predictions. Later, public un-labeled nonsensitive data are introduced to improve the second level classifier in (Papernot et al. 2017). However, our goal here is to boost learning performance in the target domain, and these works cannot achieve this. The second line of researches lie on simultaneously training a model among all parties. A stochastic gradient descent among parties are used in (Rajkumar and Agarwal 2012), and then a multi-task learning method is proposed in (Xie et al. 2017). While these works improve performance of previous ones based on aggregation, they gradually lose privacy guarantee during the iterations of algorithms. Thus, they do not apply here either.

3 The Proposed Approach

First in Section 3.1, we show how a simple combination can be done based on privacy-preserving LR and HTL. As such combination can suffer from small samples in the target, we improve privacy-preserving LR in Section 3.2, and use it to solve the knowledge sharing problem in Section 3.3.

3.1 First Attempt: A Simple Combination

Basically, HTL can already solve the knowledge sharing problem as described in Section 1. We can obtain $\mathbf{w}_s$ using the source data from (1), and then obtain the desired predictor $\mathbf{w}_t$ on target data from (4). The problem is that there is no privacy guarantee yet. However, Algorithm 1 can be used to solve this issue. The key idea is to perturb the objective by $\mathbf{b}$ and Δ. Thus,we solve

$$\bar{\mathbf{w}}_s = \arg\min_{\mathbf{w}} \bar{F}(\mathbf{w}; \mathcal{D}_s, \mathbf{b}, \Delta) + \lambda_s g_s(\mathbf{w}), \quad (5)$$

for the source, where $g_s(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_2^2$. Then, we solve

$$\bar{\mathbf{w}}_t = \arg\min_{\mathbf{w}} \bar{F}(\mathbf{w}; \mathcal{D}_t, \mathbf{b}, \Delta) + \lambda_t g_t(\mathbf{w}), \quad (6)$$

for the target, where $g_t(\mathbf{w}) = \frac{\eta}{2}\|\mathbf{w}\|_2^2 + \frac{1-\eta}{2}\|\mathbf{w} - \bar{\mathbf{w}}_s\|_2^2$ and $\bar{\mathbf{w}}_s$ is obtained from (5). The complete procedure for the simple approach is given in Algorithm 2, and it can be easily seen ϵ_s and ϵ_t-differential privacy are guaranteed for the source and target.

Algorithm 2 Simple Combination (SimComb)

Require: dataset $\mathcal{D}_s$ and $\mathcal{D}_t$, privacy budgets ϵ_s and ϵ_t, hyper-parameters $\lambda_s, \lambda_t, \eta_t$.
in source domain
1: $\bar{\mathbf{w}}_s = \text{PLR}(\mathcal{D}_s, \epsilon_s, \lambda_s, g_s)$ where $g_s(\cdot) = \frac{1}{2}\|\cdot\|_2^2$;
in target domain
2: $\bar{\mathbf{w}}_t = \text{PLR}(\mathcal{D}_t, \epsilon_t, \lambda_t, g_t)$ where $g_t(\cdot) = \frac{\eta}{2}\|\cdot\|_2^2 + \frac{1-\eta}{2}\|\cdot - \bar{\mathbf{w}}_s\|_2^2$;
3: **return** $\bar{\mathbf{w}}_s$ for the source (ϵ_s-privacy guarantee), and $\bar{\mathbf{w}}_t$ for the target (ϵ_t-privacy guarantee).

Simple combination issues. However, such a direct combination has two main problems. First,

(A). Algorithm 2 suffers from poor performance when the dimension is high.

This problem comes from the fact that Algorithm 1 is sensitive to the dimension d, which determines the noise level. The theoretical analysis in Proposition 2 also shows that with limited privacy budget, more samples are needed to preserve the generalization performance when d increases. The problem is even worse in our setting, as the number of samples of target is usually small.

(B). Algorithm 2 treats all features equally, thus does not consider feature importance.

Note that, the noise is equally added to all features in Algorithm 2. However, if we can add less noise into features that are more importance while keeping the same privacy guarantee, we are likely to get a better learning performance.

Overview of proposed method. To address the above two problems (A) and (B), we propose to split features of the source data into disjoint subsets and transfer the learned model from each subset to the target domain separately. For models trained on these subsets, we can hope that each of them will suffer a smaller impact from the noises while preserving privacy. Besides, when the feature importance is available, less noise can be added into subset with larger importance. These are the intuitions of our work. The framework is in Figure 2.

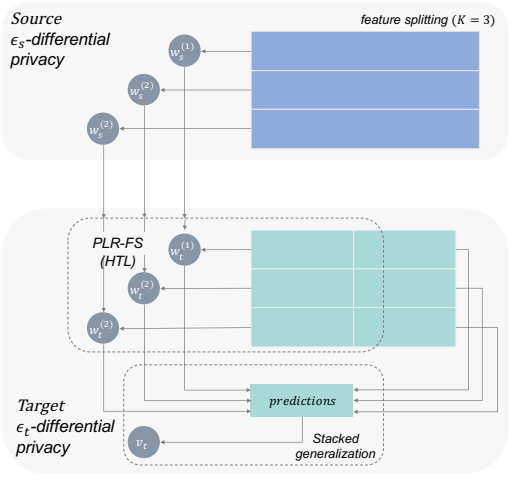


Figure 2: The framework of the proposed method.

3.2 Building Block: PLR with Feature Splitting

In order to accomplish the above procedure, we first propose a new algorithm, which incorporates feature splitting into Algorithm 1. This algorithm will also acts as a key building block for our approach.

The proposed method is presented in Algorithm 3. In the sequel, let $S_G(\mathcal{D}) = \{G^{(1)}(\mathcal{D}), \cdots, G^{(K)}(\mathcal{D})\}$ be a split of all features into K subsets, and the importance of each split is $q^{(k)}$ with $q^{(k)} \geq 0$ and $\sum_{k=1}^K q^{(k)} = 1$. First, features in dataset $\mathcal{D}$ is split into K subsets based on S_G in step 1. Then, same as Algorithm 1, we refine ϵ as ϵ' in step 2. However, to ensure privacy, the features in each subset are also re-scaled (step 4) and extra terms involving with $\mathbf{b}^{(k)}$ and $\Delta^{(k)}$ are added (step 5-10). Finally, we train a logistic regression within each subset of the dataset (step 11).

Algorithm 3 PLR-FS($\mathcal{D}, \epsilon, S_G, \{\lambda^{(k)}, g^{(k)}, q^{(k)}\}_{k=1}^K$): Privacy-Preserving LR with feature splitting

Require: privacy budget ϵ, dataset $\mathcal{D}$, hyper-parameters $\{\lambda^{(k)}\}_{k=1}^K$, regularizers $\{g^{(k)}\}_{k=1}^K$, feature importance $\{q^{(k)}\}_{k=1}^K$ where $q^{(k)} \geq 0$ and $\sum_{k=1}^K q^{(k)} = 1$;
1: split features of $\mathcal{D}$ based on S_G in K partitions as $\{G^{(1)}(\mathcal{D}), \cdots, G^{(K)}(\mathcal{D})\}$;
2: $\epsilon' = \epsilon - \sum_{k=1}^K \log(1 + \frac{(q^{(k)})^2}{2n\lambda^{(k)}} + \frac{(q^{(k)})^4}{16n^2(\lambda^{(k)})^2})$;
3: **for all** each $G^{(k)}(\mathcal{D}) \in \mathcal{D}$ **do**
4: scale $G^{(k)}(\mathcal{D})$ so that for all i, $\|\mathbf{x}_i^{(k)}\|_2 \leq q^{(k)}$;
5: **if** $\epsilon' > 0$ **then**
6: $\Delta^{(k)} = 0$ and $\epsilon^{(k)} = \epsilon$;
7: **else**
8: $\Delta^{(k)} = \frac{(q^{(k)})^2}{4n(\exp(q^{(k)}\epsilon/4)-1)} - \lambda^{(k)}$ and $\epsilon^{(k)} = \frac{\epsilon}{2}$;
9: **end if**
10: pick a random vector $\mathbf{b}^{(k)}$ from $h(\mathbf{b}) \propto \exp(\frac{\epsilon^{(k)}\|\mathbf{b}\|_2}{2})$;
11: obtain $\bar{\mathbf{w}}^{(k)}$ by solving $\min_{\mathbf{w}^{(k)}} \bar{F}(\mathbf{w}^{(k)}; G^{(k)}(\mathcal{D}), \mathbf{b}^{(k)}, \Delta^{(k)}) + \lambda g^{(k)}(\mathbf{w}^{(k)})$;
12: **end for**
13: **return** $\{\bar{\mathbf{w}}^{(1)}, \cdots, \bar{\mathbf{w}}^{(K)}\}$ (ϵ-differential privacy);

The privacy guarantee is in Proposition 3. This proposition extends previous one (Proposition 1) for Algorithm 1. However, the extension is not trivial. First, we have multiple learning problems in Algorithm 3 and they are not independent with each other due to shared labels. Finally, different importance $q^{(k)}$s exist for each subset of features.

Proposition 3. *If each $g^{(k)}$ is 1-strongly convex, then Algorithm 3 has a ϵ-differential privacy guarantee.*

As multiple predictors are generated from Algorithm 3, unlike Proposition 2, which gives the learning performance for the whole Algorithm 1, we can only give the learning performance for each predictor generated from Algorithm 3, which is in Proposition 4.

Proposition 4. *Let $g^{(k)} = \frac{1}{2}\|\cdot - \mathbf{w}_0^{(k)}\|_2^2$ where $\mathbf{w}_0^{(k)}$ is any constant vector. There exists a constant C_2 such that for $\delta > 0$, if the training samples in $\mathcal{D}$ are drawn i.i.d. according to*

P, and if

$$n > C_2 \max\left(\frac{(a^{(k)})^2 \log(\frac{1}{\delta})}{\epsilon_g^2}, \frac{d\log(\frac{d}{K\delta})a^{(k)}}{q^{(k)}K\epsilon_g\epsilon}, \frac{(a^{(k)})^2}{\epsilon_g\epsilon}\right), \quad (7)$$

where $a^{(k)} = q^{(k)}\|\hat{\mathbf{w}}^{(k)}\|_2$ *and* $\hat{\mathbf{w}}^{(k)}$ *is a reference predictor. Then,* $\bar{\mathbf{w}}^{(k)}$ *in Algorithm 3 satisfies* $Pr[L(\bar{\mathbf{w}}^{(k)}, P) \leq L(\hat{\mathbf{w}}^{(k)}, P) + \epsilon_g] \geq 1 - \delta$.

Comparing with Proposition 2, the main difference is that $\|\hat{\mathbf{w}}\|_2$ is replaced with $a^{(k)}$ in Proposition 4. Note that as feature vectors in Algorithm 3 is scaled as $\|\mathbf{x}_i^{(k)}\|_2 \leq q^{(k)}$, we have $a^{(k)} \simeq \|\hat{\mathbf{w}}\|_2$. When ϵ is small and dimension is large, the second term in (3) and (7) will be dominated. In this case, the ratio of sample complexities in Proposition 4 divided by Proposition 2 is $r^{(k)} = \frac{\log(\frac{d}{K\delta})}{q^{(k)}K\log(\frac{d}{\delta})}$. If features are randomly split and their importance is treated equally, then $q^{(k)} = 1/K$, and $r^{(k)} < 1$ as $\log(d/(K\delta)) < \log(d/\delta)$. This leads to a strict improvement over Proposition 2. Besides, if feature importance is known, for the important feature group k, $q^{(k)} > 1/K$ and $r^{(k)}$ will be smaller. While the bound for $\bar{\mathbf{w}}^{(k)}$ from less important features are worse, as important features are more informative, we will see splitting based on features' importance can have a significant improvement over random splitting in a dataset where the variance of feature importance is large.

3.3 Proposed PPTL-FS Approach

Now, we are ready to present the proposed algorithm to solve the privacy-preserving transfer learning problem in Figure 1. The complete procedure is given in Algorithm 4, and illustrated in Figure 2. The detailed steps are as follows:

Algorithm 4 Privacy-Preserving Transfer Learning based on feature splitting and stacked generalization (PPTL-FS)

Require: splitting S_G, feature importance $\{q^{(k)}\}_{k=1}^K$, dataset $\mathcal{D}_s$ and $\mathcal{D}_t$, privacy budgets ϵ_s and ϵ_t, hyper-parameters $\{\lambda_s^{(k)}\}_{k=1}^K$, $\{\lambda_t^{(k)}, \eta_t^{(k)}\}_{k=1}^K$ and μ_t.

in the source domain

1: $\{\bar{\mathbf{w}}_s^{(1)}, \cdots, \bar{\mathbf{w}}_s^{(K)}\}$ = PLR-FS$(\mathcal{D}_s, \epsilon_s, S_G, \{\lambda_s^{(k)}, g_s, q^{(k)}\}_{k=1}^K)$ where $g_s(\cdot) = \frac{1}{2}\|\cdot\|_2^2$;

in the target domain

2: randomly split $\mathcal{D}_t$ into 2 subsets $\mathcal{D}_{t0}$ and $\mathcal{D}_{t1}$ with approximately equal sizes;

3: level-0 model: $\{\bar{\mathbf{w}}_t^{(1)}, \cdots, \bar{\mathbf{w}}_t^{(K)}\}$ = PLR-FS$(\mathcal{D}_{t0}, \epsilon_t, S_G, \{\lambda_t^{(k)}, g_t^{(k)}, q^{(k)}\}_{k=1}^K)$ where $g_t^{(k)}(\cdot) = \frac{\eta^{(k)}}{2}\|\cdot\|_2^2 + \frac{1-\eta^{(k)}}{2}\|\cdot - \bar{\mathbf{w}}_s^{(k)}\|_2$;

4: level-1 model: $\bar{\mathbf{v}}_t$ = PLR$(\tilde{\mathcal{D}}_{t1}, \epsilon_t, \mu_t, g)$ with $g(\cdot) = \frac{1}{2}\|\cdot\|_2^2$;

5: **return** $\{\bar{\mathbf{w}}_s^{(1)}, \cdots, \bar{\mathbf{w}}_s^{(K)}\}$ for the source (ϵ_s-privacy guarantee), $\{\bar{\mathbf{w}}_t^{(1)}, \cdots, \bar{\mathbf{w}}_t^{(K)}\}$ and $\bar{\mathbf{v}}_t$ for the target (ϵ_t-privacy guarantee).

Feature Splitting. First, we require both source and target having the same splitting S_G on features. This can be done either by a random splitting or based on features' importance.

Source (step 1). Similar to Section 3.1, we take $g_s(\cdot) = \frac{1}{2}\|\cdot\|_2^2$ as the regularization on the source. Then, we directly use Algorithm 3 to obtain predictors $\{\bar{\mathbf{w}}_s^{(1)}, \cdots, \bar{\mathbf{w}}_s^K\}$ from the source based on the features splitting. These predictors will be later transferred to the target.

Target (step 2-4). With $\{\bar{\mathbf{w}}_s^{(1)}, \cdots, \bar{\mathbf{w}}_s^{(K)}\}$ from the source, we then try to learn a high-quality model in the target domain. Following Section 3.1,we adapt HTL by taking $g_t^{(k)}(\cdot) = \frac{\eta}{2}\|\cdot\|_2^2 + \frac{1-\eta}{2}\|\cdot - \bar{\mathbf{w}}_s^{(k)}\|_2^2$ as the regularizer. This again can be done with Algorithm 3. However, the learning process leads to K predictors in the target, and we do not know which predictor should be used nor how to combine them. To solve this problem, we next propose to use a variant of stacked generalization (Ting and Witten 1997).

We randomly split $\mathcal{D}_t$ into two disjoint parts $\mathcal{D}_{t0}, \mathcal{D}_{t1}$, each has half of the samples of $\mathcal{D}_t$ (step 2). Then, Algorithm 3 is trained based on $\mathcal{D}_{t0}$ (level-0 model), and we obtain $\{\bar{\mathbf{w}}_t^{(1)}, \cdots, \bar{\mathbf{w}}_t^{(K)}\}$ as the predictors from the target (step 3). After that,we take the predicted labels from $\mathcal{D}_{t1}$ by these predictors as features, and train a privacy-preserving logistic regression with parameter $\mathbf{v}_t$ (level-1 model), which can be done by Algorithm 1 (step 4 in Algorithm 4). The final prediction is then given by the output of level-1 model. In this way, the level-1 model acts as an aggregator to combine $\{\bar{\mathbf{w}}_t^{(1)}, \bar{\mathbf{w}}_t^2, \cdots, \bar{\mathbf{w}}_t^{(K)}\}$ in a data-driven manner.

Algorithm Analysis. First, we show the privacy guarantee of Algorithm 4 in Theorem 5.

Theorem 5. *The output* $\{\bar{\mathbf{w}}_s^{(1)}, \cdots, \bar{\mathbf{w}}_s^{(K)}\}$ *is* ϵ_s*-differentially private with respect to* $\mathcal{D}_s$, $\{\bar{\mathbf{w}}_t^{(1)}, \cdots, \bar{\mathbf{w}}_t^{(K)}\}$ *and* $\bar{\mathbf{v}}_t$ *are* ϵ_t*-differentially private to* $\mathcal{D}_t$.

As the learning guarantee of stacked generalization is still an open issue (Ting and Witten 1999), we cannot offer the guarantee of generalization performance for Algorithm 4 here. However, we can still expect Algorithm 4 to be much better than Algorithm 2. The reasons are as follows. First, when less noise is introduced, better generalization performance can be expected in the target domain, which is also verified by Proposition 4. This means that the prediction performance of $\{\bar{\mathbf{w}}_s^k\}_{k=1}^K$ and $\{\bar{\mathbf{w}}_t^k\}_{k=1}^K$ from Algorithm 4 is better than that of $\mathbf{w}_s$ and $\mathbf{w}_t$ from Algorithm 2. Second,while feature interactions may slightly drop by the feature splitting, the level-1 model can alleviate this problem by aggregating predictions from different splits in level-0 model in a supervised way. Finally, feature importance can be incooperated in Algorithm 2, which can further boost the learning performance by adding less noise on more important features.

4 Experiment

In the sequel, experiments are performed on a Server with Intel(R) Xeon(R) E5 CPU and 250G Memory. All the codes are implemented in Python.

Table 1: Testing AUC for MNIST dataset with different ϵ.

ϵ	0.5	1.0	2.0	4.0	8.0
PPTL-FS(W)	**0.9007 ± 0.0391**	**0.9500 ± 0.0204**	**0.9825 ± 0.0051**	**0.9895 ± 0.0019**	0.9921 ± 0.0017
PPTL-FS(R)	0.7687 ± 0.0998	0.8459 ± 0.0701	0.9361 ± 0.0214	0.9761 ± 0.0095	**0.9964 ± 0.0004**
SimComb	0.7005 ± 0.0613	0.8088 ± 0.0812	0.9642 ± 0.0073	**0.9906 ± 0.0018**	0.9943 ± 0.0009
SourceD	0.6527 ± 0.0927	0.7510 ± 0.0854	0.8806 ± 0.0431	0.9401 ± 0.0180	0.9523 ± 0.0088
Direct	0.6467 ± 0.0787	0.6978 ± 0.0651	0.8657 ± 0.0429	0.9632 ± 0.0117	0.9877 ± 0.0039

Table 2: Testing AUC for RUIJIN dataset with different ϵ.

ϵ	0.5	1.0	2.0	4.0	8.0
PPTL-FS(W)	**0.7108 ± 0.0611**	**0.7469 ± 0.0319**	**0.7649 ± 0.0255**	**0.7581 ± 0.0392**	**0.7564 ± 0.0362**
PPTL-FS(R)	0.6235 ± 0.0732	0.6778 ± 0.0495	0.7107 ± 0.0474	0.7372 ± 0.0337	0.7226 ± 0.0397
SimComb	0.5570 ± 0.1134	0.6023 ± 0.0854	0.6255 ± 0.0763	0.6416 ± 0.0706	0.7386 ± 0.0311
SourceD	0.6067 ± 0.0678	0.6298 ± 0.0420	0.6546 ± 0.0330	0.6489 ± 0.0462	0.7349 ± 0.0320
Direct	0.5112 ± 0.0733	0.5476 ± 0.0882	0.6216 ± 0.0815	0.6649 ± 0.0552	0.7231 ± 0.0381

4.1 Data Description

Two datasets are used in our experiments. The first one is MNIST, which are small images of digital numbers and popularly used for handwritten recognition (LeCun et al. 1998). The second one is a private dataset provided by ShangHai RUIJIN hospital used to train and test early signs of chronical diseases that cover a large sector of the Chinese population, such as Diabetes. Our problem setup (Figure 1) also comes from application background of such data set. The details of the data are as follows [4] :

MNIST. We generate a toy dataset of two classification tasks here. Digits 0 and 8 are taken as the source, 0 and 9 as the target. We apply PCA on the whole dataset and select the top 100 features with the largest variance. To simulate the case where the size of the task dataset is not large, we randomly picked 2000 samples for the source and 1000 samples for the target.

RUIJIN. This is a real dataset where user privacy is a big concern as the data contains patients' personal medical records. The dataset is collected from two medical investigations over 22 medical centres distributed in different locations in China conducted by Shanghai RUIJIN Hospital in 2010 and 2013, respectively. There are in total 105,763 participants who participated in both investigations. The first investigation consists questionnaires and laboratory test records collecting demographic information, disease information, life-style information and physical examination results. The second investigation includes diagnosis of diabetes, which are used as the ground truth (labels) of diabetes prediction. Suggested by the hospital, in this dataset we choose 18 centres with 52 important features (Table 3) and the first two centres are combined as the source and remained centres are target. When we make predictions for patients form a new center, we wish to rely on previous built models for other related but different centers; otherwise we would have to do a lot of new manual labor at each new center. When doing this transfer learning, user privacy is of key concern. Thus, we wish to apply the privacy-preserving transfer learning algorithms to solve the problems.

[4] Due to the noise, the dimension in private applications is much smaller than non-private applications. In our experiments, the dimension of datasets is as large as previous works (Chaudhuri, Monteleoni, and Sarwate 2011; Abadi et al. 2016)

Table 3: Sample sizes of different centres in RUIJIN dataset.

centre id	1	2	3	4	5	6
#sample	7882	4820	4334	4739	6121	2327
centre id	7	8	9	10	11	12
#sample	5619	6360	4966	5793	6215	3659
centre id	13	14	15	16	17	18
#sample	5579	2316	4285	6017	6482	4493

4.2 Compared Methods

We compare the following methods: (i). Direct: Directly training an ϵ-differentially private logistic regression model on the target dataset; (ii). SourceD: Directly using the ϵ-differentially private source classifier to predict on the target dataset; (iii). SimComb: The Simple approach (Algorithm 2); (iv). Two variant of the proposed method (Algorithm 4). PPTL-FS(R): features are uniformly random split into K groups; and PPTL-FS(W): features are split into K groups with approximately equal size based on their importance. For MINST, the feature importance is given by the variance obtained from PCA; and for RUIJIN, feature importance is given by doctors in RUIJIN hospital.

For all experiments, we randomly select 80% as training set and 20% as testing set for both source and target datasets. All other parameters (all methods) are tuned by a 3-fold cross-validation. Moreover, we use $K = 5$ for our proposed method. In the transfer phase of our proposed method, we use 50% of the target training set as $\mathcal{D}_{t0}$ and the remaining as $\mathcal{D}_{t1}$. For performance evaluation, we use the area-under-the-curve (AUC) (Hanley and McNeil 1983) as the measurement, which is most often used for classification problems, and a higher AUC is desired. For each dataset, we repeat the experiment 10 times.

4.3 Performance

MNIST. We vary privacy budget as $\epsilon_s = \epsilon_t = \epsilon \in \{0.5, 1.0, 2.0, 4.0, 8.0\}$. The result is shown in [5] Table 1 and [6] Figure 3(a). First, we can see that when ϵ gets smaller, AUC of all methods decrease. This is because more noise needs to be introduced for smaller ϵ. Then, SimComb consistently gets better performance than SourceD and Direct, which verifies the benefits of hypothesis transfer learning. Besides, when ϵ gets smaller, PPTL-FS(R) can

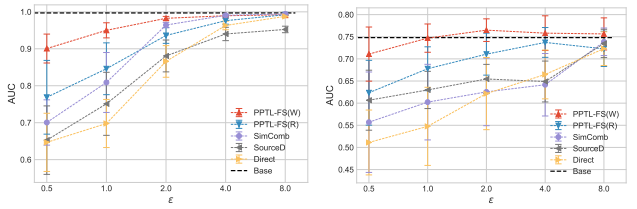
[5] In the sequel, the highest and comparable AUCs according to the pairwise t-test with 95% confidence are highlighted.

[6] Due to lack of space, figures in large size are in Appendix **??**.

Table 4: Testing AUC on centres 3-18 in RUIJIN dataset.

centre	3	4	5	6	7	8
PPTL-FS(W)	**0.7469 ± 0.0321**	**0.7360 ± 0.0317**	**0.7395 ± 0.0400**	**0.7143 ± 0.0397**	**0.7664 ± 0.0388**	**0.7072 ± 0.0169**
PPTL-FS(R)	0.6778 ± 0.0495	**0.7237 ± 0.0373**	0.6517 ± 0.1026	**0.7081 ± 0.0329**	0.6529 ± 0.0697	0.6627 ± 0.0365
SimComb	0.6023 ± 0.0853	0.6080 ± 0.0782	0.5279 ± 0.0623	0.5632 ± 0.0670	0.5776 ± 0.0750	0.6012 ± 0.0307
SourceD	0.6298 ± 0.0420	0.6151 ± 0.0661	0.5270 ± 0.0565	0.5743 ± 0.0631	0.5592 ± 0.0690	0.6017 ± 0.0308
Direct	0.5476 ± 0.0882	0.6203 ± 0.0556	0.6355 ± 0.0458	0.5788 ± 0.0749	0.5327 ± 0.0575	0.6132 ± 0.0350
centre	9	10	11	12	13	14
PPTL-FS(W)	**0.7205 ± 0.0464**	**0.7529 ± 0.0417**	**0.7010 ± 0.0233**	**0.6983 ± 0.0361**	**0.7357 ± 0.0462**	**0.7384 ± 0.0451**
PPTL-FS(R)	**0.6817 ± 0.0336**	0.6916 ± 0.0444	0.6346 ± 0.0267	0.6437 ± 0.0499	0.6350 ± 0.0536	0.6452 ± 0.0610
SimComb	0.5804 ± 0.0708	0.5827 ± 0.0566	0.5468 ± 0.0665	0.5167 ± 0.0747	0.5652 ± 0.0587	0.5469 ± 0.0893
SourceD	0.5757 ± 0.0769	0.5362 ± 0.0768	0.5465 ± 0.0476	0.5351 ± 0.0585	0.5625 ± 0.0582	0.5463 ± 0.0712
Direct	0.5609 ± 0.0764	0.5842 ± 0.0450	0.5151 ± 0.0652	0.5546 ± 0.0608	0.5531 ± 0.0659	0.5203 ± 0.0889
centre	15	16	17	18	averaged ranking	
PPTL-FS(W)	**0.7459 ± 0.0520**	**0.6614 ± 0.0942**	**0.6968 ± 0.0233**	**0.6039 ± 0.0124**	**1**	
PPTL-FS(R)	**0.7177 ± 0.0647**	**0.6435 ± 0.0439**	0.6470 ± 0.0619	0.5667 ± 0.0362	2	
SimComb	0.5917 ± 0.0806	0.6152 ± 0.0708	0.5583 ± 0.0652	0.5239 ± 0.0269	3.81	
SourceD	0.5554 ± 0.0830	0.6068 ± 0.0916	0.5515 ± 0.0620	0.5274 ± 0.0246	3.93	
Direct	0.6193 ± 0.0701	0.5634 ± 0.0259	0.5578 ± 0.0602	0.5169 ± 0.0527	4.18	

get better performance than SimComb. This observation is consistent with our Proposition 4, which show better learning performance can be achieved when ϵ is small and d is large. Finally, PPTL-FS(W) is the best method, as it further improves over PPTL-FS(R) by adding less noise to more important features.



(a) MNIST. (b) RUIJIN.

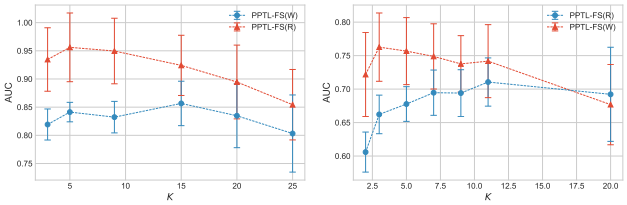
Figure 3: Testing AUC with different ϵ. The black and horizon line is the performance obtained from non-private logistic regression.

RUIJIN. Here, we set $\epsilon_s = \epsilon_t = 1.0$. The result is shown in Table 4. However, unlike the previous case, SimComb may not get better performance than SourceD and Direct, which is perhaps due to the noise introduced in features. However, EtransR improves over SimComb by feature splitting, and consistently gets better performance than SourceD, Direct and SimComb. Finally, PPTL-FS(W), which considers features importance, is the best method.

We then vary $\epsilon_s = \epsilon_t = \epsilon \in \{0.5, 1.0, 2.0, 4.0, 8.0\}$, and then plot the testing AUC on the centre 3 in Figure 3(b). The observations are the same as previous one for MNIST dataset. Testing AUC of all methods get lower with smaller ϵ, and PPTL-FS(W) is the best (compared with methods having the privacy guarantee).

Influence of the splitting size K. In previous experiments we fix the split size $K = 5$. Here we will demonstrate the influence of K. In RUIJIN and MNIST dataset, we test the performance of PPTL-FS(W) and PPTL-FS(R) for different K's. We set $\epsilon_s = \epsilon_t = 1.0$, and other parameters are tuned as previous experiments. The experiment is repeat for 30 times. The result is shown in Figure 4.

As we can see, there is a peak of K for our proposed method in both datasets. When K increases, in each subset



(a) MNIST Dataset (b) RUIJIN Dataset

Figure 4: Testing AUC with different K.

the dimension decreases, thus the noise level decreases, which leads to the increasing of the performance. This is consistent with Proposition 4. When K is too large, the interactions among features cannot be learned well, so the performance decreases. Another observation is that the peak of K of PPTL-FS(W) is smaller than that of PPTL-FS(R). This is because PPTL-FS(W) tends to put important features in subsets that can be learned well enough with a relatively small K and get good performance. Note that for all K's in both datasets, the performance of PPTL-FS(R) is not better than PPTL-FS(W) with a significance level 0.05

5 Conclusion and Future Works

In this paper we presented a novel privacy-preserving transfer learning for the knowledge sharing problem. Unlike previous privacy-preserving methods, we take particular attention to the learning performance in the target domains even after much noise is added to the source domain for privacy preservation. This is done by using a stacked learning model taking components from the split source domains by features. We show that this approach addressed both the privacy preserving issue and the model performance issue, which is a critical issue facing machine learning under privacy laws in many parts of the world.

In the future, first we plan to explore (ϵ, δ)-differential privacy, which is a relax version of Definition 1 (Dwork et al. 2006a). This usually leads to weaker privacy guarantee but better predicting performance. We will also consider extending by transferring other types of learning algorithms, such as deep learning, kernel learning and Bayesian learning.

References

[Abadi et al. 2016] Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H. B.; Mironov, I.; Talwar, K.; and Zhang, L. 2016. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security*, 308–318. ACM.

[Bassily, Smith, and Thakurta 2014] Bassily, R.; Smith, A.; and Thakurta, A. 2014. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Annual Symposium on Foundations of Computer Science*, 464–473. IEEE.

[Chaudhuri, Monteleoni, and Sarwate 2011] Chaudhuri, K.; Monteleoni, C.; and Sarwate, A. D. 2011. Differentially private empirical risk minimization. *Journal of Machine Learning Research* 12(Mar):1069–1109.

[Dwork and Roth 2014] Dwork, C., and Roth, A. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9(3–4):211–407.

[Dwork et al. 2006a] Dwork, C.; Kenthapadi, K.; McSherry, F.; Mironov, I.; and Naor, M. 2006a. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 486–503.

[Dwork et al. 2006b] Dwork, C.; McSherry, F.; Nissim, K.; and Smith, A. 2006b. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, 265–284. Springer.

[Dwork 2008] Dwork, C. 2008. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, 1–19. Springer.

[Emekçi et al. 2007] Emekçi, F.; Sahin, O. D.; Agrawal, D.; and El Abbadi, A. 2007. Privacy preserving decision tree learning over multiple parties. *Data & Knowledge Engineering* 63(2):348–361.

[Fong and Weber-Jahnke 2012] Fong, P. K., and Weber-Jahnke, J. H. 2012. Privacy preserving decision tree learning using unrealized data sets. *IEEE Transactions on knowledge and Data Engineering* 24(2):353–364.

[Hamm, Cao, and Belkin 2016] Hamm, J.; Cao, Y.; and Belkin, M. 2016. Learning privately from multiparty data. In *International Conference on Machine Learning*, 555–563.

[Hanley and McNeil 1983] Hanley, J. A., and McNeil, B. J. 1983. A method of comparing the areas under receiver operating characteristic curves derived from the same cases. *Radiology* 148(3):839–843.

[Jagannathan, Pillaipakkamnatt, and Wright 2009] Jagannathan, G.; Pillaipakkamnatt, K.; and Wright, R. N. 2009. A practical differentially private random decision tree classifier. In *IEEE International Conference on Data Mining Workshops*, 114–121. IEEE.

[Kasiviswanathan and Jin 2016] Kasiviswanathan, S. P., and Jin, H. 2016. Efficient private empirical risk minimization for high-dimensional learning. In *International Conference on Machine Learning*, 488–497.

[Kuzborskij and Orabona 2013] Kuzborskij, I., and Orabona, F. 2013. Stability and hypothesis transfer learning. In *International Conference on Machine Learning*, 942–950.

[Kuzborskij and Orabona 2017] Kuzborskij, I., and Orabona, F. 2017. Fast rates by transferring from auxiliary hypotheses. *Machine Learning* 106(2):171–195.

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

[Orabona et al. 2009] Orabona, F.; Castellini, C.; Caputo, B.; Fiorilla, A. E.; and Sandini, G. 2009. Model adaptation with least-squares SVM for adaptive hand prosthetics. In *IEEE International Conference on Robotics and Automation*, 2897–2903.

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

[Papernot et al. 2017] Papernot, N.; Abadi, M.; Erlingsson, U.; Goodfellow, I.; and Talwar, K. 2017. Semi-supervised knowledge transfer for deep learning from private training data. In *International Conference on Learning Representations*.

[Pathak, Rane, and Raj 2010] Pathak, M.; Rane, S.; and Raj, B. 2010. Multiparty differential privacy via aggregation of locally trained classifiers. In *Advances in Neural Information Processing Systems*, 1876–1884.

[Rajkumar and Agarwal 2012] Rajkumar, A., and Agarwal, S. 2012. A differentially private stochastic gradient descent algorithm for multiparty classification. In *Artificial Intelligence and Statistics*, 933–941.

[Shalev-Shwartz and Srebro 2008] Shalev-Shwartz, S., and Srebro, N. 2008. SVM optimization: inverse dependence on training set size. In *International Conference on Machine Learning*, 928–935. ACM.

[Shokri and Shmatikov 2015] Shokri, R., and Shmatikov, V. 2015. Privacy-preserving deep learning. In *ACM SIGSAC conference on computer and communications security*, 1310–1321.

[Ting and Witten 1997] Ting, K. M., and Witten, I. H. 1997. Stacked generalization: when does it work? In *International Joint Conference on Artifical Intelligence*.

[Ting and Witten 1999] Ting, K. M., and Witten, I. H. 1999. Issues in stacked generalization. *Journal of Artificial Intelligence Research* 10:271–289.

[Xie et al. 2017] Xie, L.; Baytas, I.; Lin, K.; and Zhou, J. 2017. Privacy-preserving distributed multi-task learning with asynchronous updates. In *International Conference on Knowledge Discovery and Data Mining*, 1195–1204.