Differentially Private Transfer Learning with Conditionally Deep Autoencoders

Mohit Kumar mohit.kumar@uni-rostock.de
Faculty of Computer Science and Electrical Engineering
University of Rostock, Germany
Software Competence Center Hagenberg GmbH
A-4232 Hagenberg, Austria

Bernhard A. Moser bernhard.moser@scch.at
Software Competence Center Hagenberg GmbH
A-4232 Hagenberg, Austria

Editor:

Abstract

This paper considers the problem of differentially private semi-supervised transfer learning. The notion of *membership-mapping* is developed using measure theory basis to learn data representation via a fuzzy membership function. An alternative conception of deep autoencoder, referred to as *Conditionally Deep Membership-Mapping Autoencoder (CDMMA)* (that consists of a nested compositions of membership-mappings), is introduced. Under practice-oriented settings, an analytical solution for the learning of CDMFA can be derived by means of variational optimization. The paper proposes a transfer learning approach that combines CDMMA with a tailored noise adding mechanism to achieve a given level of privacy-loss bound with the minimum perturbation of the data. Numerous experiments were carried out using MNIST, USPS, Office, and Caltech256 datasets to verify the competitive robust performance of the proposed methodology.

Keywords: Privacy, Transfer Learning, Membership Function, Variational Optimization.

1. Introduction

The availability of high quality labelled data is crucial for the success of machine learning methods. While a single entity may not own massive amount of data, a collaboration among data-owners regarding sharing of knowledge extracted locally from their private data can be beneficial. The data privacy concerns and the legal requirements may not allow a centralization of the data from multiple sources. Thus, an interest in privacy-preserving machine learning with distributed training datasets arises. We consider the privacy-preserving distributed machine learning problem under a scenario that the knowledge extracted from a labelled training dataset (referred to as *source domain*) is intended to improve the learning of a classifier trained using a dataset with both unlabelled and very few labelled samples (referred to as *target domain*) such that source and target domains are allowed to be heterogeneous. That is, source and target data samples are allowed to differ in their dimensions and no assumptions are made regarding statistical distributions of source and target data. The problem of *privacy-preserving semi-supervised transfer learning* has previously been ad-

dressed in the literature from different prospects. We focus on the development of a method able to simultaneously deal with high-dimensional data and heterogeneous domains.

State-of-the-art: A lot of research has been carried out in the area of transfer learning. The heterogeneous data from source and target domain (i.e. source and target domains have different feature space and dimensions) can be transformed to a common subspace by using two different projection matrices. Existing supervised learning methods (e.g., SVM) can be then employed to learn the projection matrices and the target domain classifier (Li et al., 2014). It is possible to learn a transformation that maps feature points from one domain to another using cross-domain constraints formed by requiring that the transformation maps points from the same category (but different domain) near each other (Hoffman et al., 2014). A study (Herath et al., 2017) learns projections from each domain to a latent space via simultaneously minimizing a notion of domain variance while maximizing a measure of discriminatory power where Riemannian optimization techniques are used to match statistical properties between samples projected into the latent space from different domains. Another study (Courty et al., 2017) proposes a regularized unsupervised optimal transportation model to perform the alignment of the representations in the source and target domains. The method in (Gong et al., 2012) uses geodesic flow to construct an infinite-dimensional feature space that assembles information on the source domain, on the target domain, and on *phantom* domains interpolating between source and target domains. Inner products in infinite-dimensional feature space give rise to a kernel function facilitating the construction of any kernelized classifiers. Another approach is of an adaptation of source model to the target domain via iteratively deleting source-domain samples and adapting the model gradually to the target-domain instances (Bruzzone and Marconcini, 2010). Boosting-based learning algorithms can be also used to adaptively assign the training weights to source and target samples based on their relevance in the training of the classifier (Dai et al., 2007). Bayesian learning can be a framework to study transfer learning through modeling of a joint prior probability density function for feature-label distributions of the source and target domains (Karbalayghareh et al., 2018). Deep learning framework is another promising research direction explored for transfer learning (Long et al., 2015, 2016; Ganin et al., 2016).

The datasets may contain sensitive information that need to be protected from *model inversion* attack (Fredrikson et al., 2015) and from adversaries with an access to model parameters and knowledge of the training procedure. This goal has been addressed within the framework of differential privacy (Abadi et al., 2016; Phan et al., 2016). Differential Privacy (Dwork et al., 2006; Dwork and Roth, 2014) is a formalism to quantify the degree to which the privacy for each individual in the dataset is preserved while releasing the output of a data analysis algorithm. Differential privacy provides a guarantee that an adversary, by virtue of presence or absence of an individual's data in the dataset, would not be able to draw any conclusions about an individual from the released output of the analysis algorithm. This guarantee is achieved by means of a randomization of the data analysis process. In the context of machine learning, randomization is carried out via either adding random noise to the input or output of the machine learning algorithm or modifying the machine learning algorithm itself. A limited number of studies exist on differentially private semi-supervised transfer learning. The authors in (Ji and Elkan, 2013) suggest an

importance weighting mechanism to preserve the differential privacy of a private dataset via computing and releasing a weight for each record in an existing public dataset such that computations on public dataset with weights is approximately equivalent to computations on private dataset. The importance weighting mechanism is adapted in (Wang et al., 2018) to determine the weight of a source hypothesis in the process of constructing informative Bayesian prior for logistic regression based target model. (Papernot et al., 2017) introduces *private aggregation of teacher ensembles* approach where an ensemble of *teacher* models is trained on disjoint subsets of the sensitive data and a *student* model learns to predict an output chosen by noisy voting among all of the teachers. Another approach (Acs et al., 2017; Xie et al., 2018; Zhang et al., 2017) is to construct a differentially private unsupervised generative model for generating a synthetic version of the private data, and then releases the synthetic data for a non-private learning. This technique is capable of effectively handling high-dimensional data in differential privacy setting. The study in (Niinimäki et al., 2019) uses a large public dataset to learn a dimension-reducing representation mapping which is then applied on private data to obtain a low-dimensional representation of the private data followed by the learning of a differentially private predictor. However, these methods don't consider the heterogeneous domains.

Differential privacy preserves the privacy of the training dataset via adding random noise to ensure that an adversary can not infer any single data instance by observing model parameters or model outputs. We follow the *input perturbation* method where noise is added to original data to achieve $(\epsilon, \delta)-$differential privacy of any subsequent computational algorithm processing the perturbed data. However, the injection of noise into data would in general result in a loss of algorithm's accuracy. Therefore, design of a noise injection mechanism achieving a good trade-off between privacy and accuracy is a topic of interest (Geng et al., 2018; Balle and Wang, 2018; Ghosh et al., 2012; Gupte and Sundararajan, 2010; Geng and Viswanath, 2016a; Geng et al., 2015; Geng and Viswanath, 2016b). The authors in (Kumar et al., 2019) derive the probability density function of noise that minimizes the expected noise magnitude together with satisfying the sufficient conditions for $(\epsilon, \delta)-$differential privacy. This noise adding mechanism was applied for differentially private distributed deep learning in (Kumar et al., 2020, 2021). In this study, the optimal noise adding mechanism of (Kumar et al., 2019) is applied for differentially private semi-supervised transfer learning.

Deep neural networks outperform classical machine learning techniques in a wide range of applications but their training requires a large amount of data. The issues, such as determining the optimal model structure, smaller training dataset, and iterative time-consuming nature of numerical learning algorithms, are inherent to the neural networks based parametric deep models. The nonparametric approach on the other hand can be promising to address the issue of optimal choice of model structure. However, an analytical solution instead of iterative gradient-based numerical algorithms will be still desired for the learning of deep models. These motivations have led to the development of a nonparametric deep model (Kumar and Freudenthaler, 2019; Kumar et al., 2020) that is learned analytically for representing data points. The study in (Kumar and Freudenthaler, 2019; Kumar et al., 2020) introduces the concept of *Student-t fuzzy-mapping* which is about representing mappings through a fuzzy set with Student-t type membership function such that the dimension of membership function increases with an increasing data size. A relevant result is

that a deep autoencoder model formed via a composition of finite number of nonparametric fuzzy-mappings can be learned analytically. However, (Kumar and Freudenthaler, 2019; Kumar et al., 2020) didn't provide a formal mathematical framework for the conceptualization of so-called fuzzy-mapping. This study provides to fuzzy-mapping a *measure-theoretic* conceptualization and refers it to as *membership-mapping*.

Motivation: The motivation of this study is derived from the ambition of developing a differentially private semi-supervised transfer learning framework that

R1: is capable of handling high-dimensional data and heterogeneity of domains;

R2: optimizes the differential private noise adding mechanism such that for a given level of privacy, the perturbation in the data is as small as possible;

R3: allows learning of the target domain model without requiring an access to source domain private training data;

R4: allows employing deep models in source and target domains so that data features at different abstraction levels can be used to transfer knowledge across domains;

R5: provides a conceptualization of an analytical approach to the learning of deep models while addressing the issues related to optimal choice of model structure and small sized training data.

To the best knowledge of authors, there does not exist any study in the literature addressing sufficiently simultaneously all of the aforementioned five requirements (i.e. R1-R5). Thus, we present in this study a novel approach to differentially private semi-supervised transfer learning that fulfills all of the requirements.

Proposed method: The basic idea of the proposed approach is stated in Fig. 1 while a more specific method description will be provided in Fig. 7. The method is as follows:

- An optimal differentially private noise adding mechanism is used to perturb the source dataset for preserving its privacy. The perturbed source data is used for the learning of classifier and for the computation of other parameters required for transferring knowledge from source to target domain.
- Both source and target classifiers consist of *Conditionally Deep Membership-Mapping Autoencoder (CDMMA)* based compositions. A multi-class classifier is presented that employs a parallel composition of CDMMAs to learn data representation for each class. An analytical approach is presented for the learning of the CDMMA.
- Since differential privacy will remain immune to any post-processing of noise added data samples, the perturbed source dataset is used to
  - build a differentially private source domain classifier,
  - compute a differentially private source domain latent subspace transformation-matrix,
  - define differentially private class-centers in latent subspace of source domain.

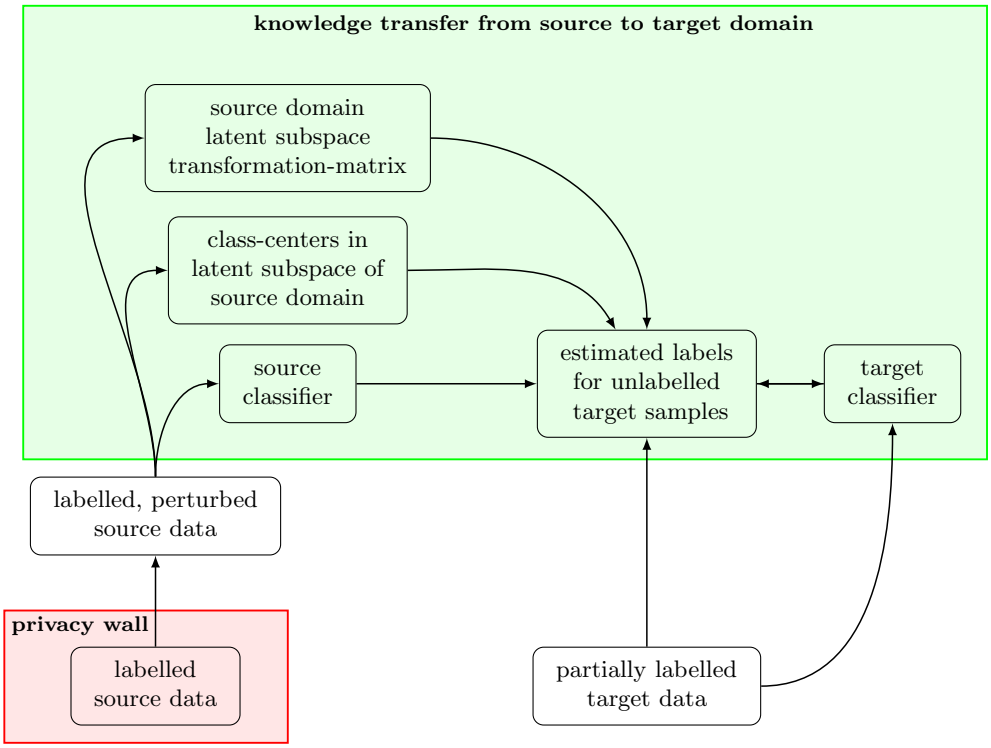


Figure 1: The proposed approach to privacy-preserving semi-supervised transfer learning.

- The class-labels for unlabelled target data samples are estimated via
  - representing target data samples in source-data-space using a transformation that maps a target sample close to center of $c-$th labelled target data samples to a point in source-data-space that is close to center of $c-$th labelled source data samples,
  - combining source and target domain classifiers for predicting class-labels.
- The target domain classifier is learned adaptively in a manner that higher-level data features are used during initial iterations for updating the classifier parameters and as the number of iterations increases more and more lower-level data features are intended to be included in the process of updating the classifier parameters.
- Since no flow of data/information occurs from target to source, no noise is added to target data.

Novelty & Contributions: The source and target models in our methodology employ CDMMAs. The classical deep autoencoder consists of two symmetrical networks of multiple layers such that first network represents the encoding and second network represents the decoding. However, the CDMMA considered in this study is composed of layers such that each layer learns data representation at certain abstraction level through a *membership-mapping autoencoder*. This study will provide a conceptualization of membership-mapping

(which is a class of mappings being characterized by the fuzzy membership functions) with measure theory basis. Our approach is to use membership-mapping as the building block of deep models. The motivation behind this approach is derived from the facts that an analytical learning solution could be derived for membership-mappings via variational optimization methodology and thus the typical issues associated to parametric deep models (such as determining the optimal model structure, smaller training dataset, and iterative time-consuming nature of numerical learning algorithms) will be automatically addressed. Thus, a formal mathematical framework will be provided for the analytical solution of the learning problem associated to deep models. The analytical approach to the learning of CDMMA, facilitating high-dimensional data representation learning at varying abstraction levels across CDMMA's different layers, will be applied for building source and target domain classifiers for differentially private transfer learning. This is the novelty of this study and contribution is to address all of the posed requirements: R1-R5. Sufficient experimentation has been provided on benchmark problems to compare the proposed approach with the state-of-art and a remarkable improvement in the performance over the state-of-art is observed.

Organization: Section 2 introduces membership-mappings using measure theory basis and presents an analytical approach to the learning of membership-mappings based on variational optimization. The CDMMA, with membership-mappings as its building blocks, is presented in section 3. Section 4 describes a differentially private semi-supervised transfer learning method implementing an optimal noise adding mechanism. The experiments in section 5 provide demonstrative examples (on MNIST and USPS datasets) and the comparison with standard techniques (on Office and Caltech256 datasets). Finally, concluding remarks are given in section 6.

2. Membership-Mappings: Conceptualization and Learning

This section introduces the concept of membership-mapping formulated using measure theory basis followed by the problem of variational learning of membership-mappings.

2.1 Measure Theoretic Conceptualization

2.1.1 Notations

- Let $n, N, p, M \in \mathbb{N}$.
- Let $\mathcal{B}(\mathbb{R}^N)$ denote the *Borel* $\sigma-$*algebra* on $\mathbb{R}^N$, and let λ^N denote the *Lebesgue measure* on $\mathcal{B}(\mathbb{R}^N)$.
- Let $(\mathcal{X}, \mathcal{A}, \rho)$ be a probability space with unknown probability measure ρ.
- Let us denote by $\mathcal{S}$ the set of finite samples of data points drawn i.i.d. from ρ, i.e.,

$$\mathcal{S} := \{(x^i \sim \rho)_{i=1}^N \mid N \in \mathbb{N}\}. \tag{1}$$

- For a sequence $\mathrm{x} = (x^1, \cdots, x^N) \in \mathcal{S}$, let $|\mathrm{x}|$ denote the cardinality i.e. $|\mathrm{x}| = N$.

- If $\mathrm{x} = (x^1, \cdots, x^N)$, $\mathrm{a} = (a^1, \cdots, a^M) \in \mathcal{S}$, then $\mathrm{x} \wedge \mathrm{a}$ denotes the concatenation of the sequences x and a, i.e., $\mathrm{x} \wedge \mathrm{a} = (x^1, \ldots, x^N, a^1, \ldots, a^M)$.
- Let us denote by $\mathbb{F}(\mathcal{X})$ the set of $\mathcal{A}$-$\mathcal{B}(\mathbb{R})$ measurable functions $f : \mathcal{X} \to \mathbb{R}$, i.e.,

$$\mathbb{F}(\mathcal{X}) := \{f : \mathcal{X} \to \mathbb{R} \mid f \text{ is } \mathcal{A}\text{-}\mathcal{B}(\mathbb{R}) \text{ measurable}\}. \tag{2}$$

- For convenience, the values of a function $f \in \mathbb{F}(\mathcal{X})$ at points in the collection $\mathrm{x} = (x^1, \cdots, x^N)$ are represented as $f(\mathrm{x}) = (f(x^1), \cdots, f(x^N))$.
- For a given $\mathrm{x} \in \mathcal{S}$ and $A \in \mathcal{B}(\mathbb{R}^{|\mathrm{x}|})$, the cylinder set $\mathcal{T}_{\mathrm{x}}(A)$ in $\mathbb{F}(\mathcal{X})$ is defined as

$$\mathcal{T}_{\mathrm{x}}(A) := \{f \in \mathbb{F}(\mathcal{X}) \mid f(\mathrm{x}) \in A\}. \tag{3}$$

- Let $\mathcal{T}$ be the family of cylinder sets defined as

$$\mathcal{T} := \left\{\mathcal{T}_{\mathrm{x}}(A) \mid A \in \mathcal{B}(\mathbb{R}^{|\mathrm{x}|}),\ \mathrm{x} \in \mathcal{S}\right\}. \tag{4}$$

- Let $\sigma(\mathcal{T})$ be the σ-algebra generated by $\mathcal{T}$.
- Given two $\mathcal{B}(\mathbb{R}^N) - \mathcal{B}(\mathbb{R})$ measurable mappings, $g : \mathbb{R}^N \to \mathbb{R}$ and $\mu : \mathbb{R}^N \to \mathbb{R}$, the weighted average of $g(\mathrm{y})$ over all $\mathrm{y} \in \mathbb{R}^N$, with $\mu(\mathrm{y})$ as the weighting function, is computed as

$$\langle g \rangle_\mu := \frac{1}{\int_{\mathbb{R}^N} \mu(\mathrm{y})\ \mathrm{d}\lambda^N(\mathrm{y})} \int_{\mathbb{R}^N} g(\mathrm{y})\mu(\mathrm{y})\ \mathrm{d}\lambda^N(\mathrm{y}). \tag{5}$$

2.1.2 Representation of Samples via Attribute Values

Let us consider a given observation $x \in \mathcal{X}$, a data point $\tilde{x} \in \mathcal{X}$, and a mapping $\mathbf{A}_x(\tilde{x}) : \tilde{x} \mapsto \mathbf{A}_x(\tilde{x}) \in [0, 1]$ such that $\mathbf{A}_x(\tilde{x})$ can be interpreted as evaluation of the degree to which the data point $\tilde{x}$ matches a given attribute induced by the observation x. $\mathbf{A}_x(\cdot)$ is called a membership function and is thought to be constructed based on the observed data x. This interpretation is motivated by Fuzzy Logics, where so-called linguistic variables represent attributes and the presumed degree to which an observation matches the attribute is called the membership value. In our approach we consider $\mathbf{A}_{x,f}(\tilde{x}) = (\zeta_x \circ f)(\tilde{x})$ to be composed of two mappings $f : \mathcal{X} \to \mathbb{R}$ and $\zeta_x : \mathbb{R} \to [0, 1]$. $f \in \mathbb{F}(\mathcal{X})$ can be interpreted as physical measurement (e.g., temperature), and $\zeta_x(f(\tilde{x}))$ as degree to which $\tilde{x}$ matches the attribute under consideration, e.g. "*hot*" where e.g. x is a representative sample of "*hot*".

Next, we extend this concept to sequences of data points in order to evaluate how much a sequence $\tilde{\mathrm{x}} = (\tilde{x}^1, \ldots, \tilde{x}^N) \in \mathcal{S}$ matches to the attribute induced by observed sequence $\mathrm{x} = (x^1, \ldots, x^N) \in \mathcal{S}$ w.r.t. the feature f via defining

$$\mathbf{A}_{\mathrm{x},f}(\tilde{\mathrm{x}}) = (\zeta_{\mathrm{x}} \circ f)(\tilde{\mathrm{x}}) \tag{6}$$

$$= \zeta_{\mathrm{x}}(f(\tilde{x}^1), \ldots, f(\tilde{x}^N)), \tag{7}$$

where the membership functions $\zeta_{\mathrm{x}} : \mathbb{R}^{|\mathrm{x}|} \to [0, 1]$, $\mathrm{x} \in \mathcal{S}$, satisfy the following properties:

Nowhere Vanishing: $\zeta_{\mathrm{x}}(\mathrm{y}) > 0$ for all $\mathrm{y} \in \mathbb{R}^{|\mathrm{x}|}$, i.e.,

$$\operatorname{supp}[\zeta_{\mathrm{x}}] = \mathbb{R}^{|\mathrm{x}|}. \tag{8}$$

Positive and Bounded Integrals: the functions ζ_{x} are absolutely continuous and Lebesgue integrable over the whole domain such that for all $\mathrm{x} \in \mathcal{S}$ we have

$$0 < \int_{\mathbb{R}^{|\mathrm{x}|}} \zeta_{\mathrm{x}} \, \mathrm{d}\lambda^{|\mathrm{x}|} < \infty. \tag{9}$$

Consistency of Induced Probability Measure: the membership function induced probability measures $\mathbb{P}_{\zeta_{\mathrm{x}}}$, defined on any $A \in \mathcal{B}(\mathbb{R}^{|\mathrm{x}|})$, as

$$\mathbb{P}_{\zeta_{\mathrm{x}}}(A) := \frac{1}{\int_{\mathbb{R}^{|\mathrm{x}|}} \zeta_{\mathrm{x}} \, \mathrm{d}\lambda^{|\mathrm{x}|}} \int_A \zeta_{\mathrm{x}} \, \mathrm{d}\lambda^{|\mathrm{x}|} \tag{10}$$

are consistent in the sense that for all $\mathrm{x}, \ \mathrm{a} \in \mathcal{S}$:

$$\mathbb{P}_{\zeta_{\mathrm{x}\wedge\mathrm{a}}}(A \times \mathbb{R}^{|\mathrm{a}|}) = \mathbb{P}_{\zeta_{\mathrm{x}}}(A). \tag{11}$$

For convenience, let us denote the collection of membership functions satisfying aforementioned assumptions by

$$\Theta := \{\zeta_{\mathrm{x}} : \mathbb{R}^{|\mathrm{x}|} \to [0,1] \mid (8), (9), (11), \ \mathrm{x} \in \mathcal{S}\}. \tag{12}$$

2.1.3 $\mathbb{F}(\mathcal{X})$ as Probability Space

It is shown in Appendix A that $(\mathbb{F}(\mathcal{X}), \sigma(\mathcal{T}), \mathbf{p})$ is a measure space and the probability measure $\mathbf{p}$, that was guaranteed by Kolmogorov extension theorem, is defined as

$$\mathbf{p}(\mathcal{T}_{\mathrm{x}}(A)) := \mathbb{P}_{\zeta_{\mathrm{x}}}(A) \tag{13}$$

where $\zeta_{\mathrm{x}} \in \Theta$, $\mathrm{x} \in \mathcal{S}$, $A \in \mathcal{B}(\mathbb{R}^{|\mathrm{x}|})$, and $\mathcal{T}_{\mathrm{x}}(A) \in \mathcal{T}$.

2.1.4 Expectations over Measure Space $(\mathbb{F}(\mathcal{X}), \sigma(\mathcal{T}), \mathbf{p})$ as Weighted Averages

It is shown in Appendix B that for a given $\mathcal{B}(\mathbb{R}^{|\mathrm{x}|}) - \mathcal{B}(\mathbb{R})$ measurable mapping $g : \mathbb{R}^{|\mathrm{x}|} \to \mathbb{R}$, expectation of $(g \circ f)(\mathrm{x})$ over $f \in \mathbb{F}(\mathcal{X})$ w.r.t. probability measure $\mathbf{p}$ is given as

$$\mathbb{E}_{\mathbf{p}}[(g \circ \cdot)(\mathrm{x})] = \mathbb{E}_{\mathbb{P}_{\zeta_{\mathrm{x}}}}[g]. \tag{14}$$

The weighted average of $g(\mathrm{y})$ over all $\mathrm{y} \in \mathbb{R}^{|\mathrm{x}|}$ with $\zeta_{\mathrm{x}}(\mathrm{y})$ as the weighting function is given as

$$\langle g \rangle_{\zeta_{\mathrm{x}}} := \frac{1}{\int_{\mathbb{R}^{|\mathrm{x}|}} \zeta_{\mathrm{x}}(\mathrm{y}) \, \mathrm{d}\lambda^{|\mathrm{x}|}(\mathrm{y})} \int_{\mathbb{R}^{|\mathrm{x}|}} g(\mathrm{y}) \zeta_{\mathrm{x}}(\mathrm{y}) \, \mathrm{d}\lambda^{|\mathrm{x}|}(\mathrm{y}). \tag{15}$$

It follows immediately that

$$\mathbb{E}_{\mathbf{p}}[(g \circ \cdot)(\mathrm{x})] = \langle g \rangle_{\zeta_{\mathrm{x}}}. \tag{16}$$

The significance of equality (16) is to allow calculating averages over all real valued functions belonging to $\mathbb{F}(\mathcal{X})$ via simply computing a weighted average.

2.1.5 A Membership-Mapping Example

Definition 1 (Student-t Membership-Mapping) *A Student-t membership-mapping, $\mathcal{F} \in \mathbb{F}(\mathcal{X})$, is a mapping with input space $\mathcal{X} = \mathbb{R}^n$ and a membership function $\zeta_{\mathrm{x}} \in \Theta$ that is Student-t like:*

$$\zeta_{\mathrm{x}}(\mathrm{y}) = \left(1 + 1/(\nu - 2)\,(\mathrm{y} - \mathrm{m_y})^T K_{\mathrm{xx}}^{-1}\,(\mathrm{y} - \mathrm{m_y})\right)^{-\frac{\nu+|\mathrm{x}|}{2}} \tag{17}$$

where $\mathrm{x} \in \mathcal{S}$, $\mathrm{y} \in \mathbb{R}^{|\mathrm{x}|}$, $\nu \in \mathbb{R}_+ \setminus [0, 2]$ is the degrees of freedom, $\mathrm{m_y} \in \mathbb{R}^{|\mathrm{x}|}$ is the mean vector, and $K_{\mathrm{xx}} \in \mathbb{R}^{|\mathrm{x}|\times|\mathrm{x}|}$ is the covariance matrix with its $(i,j)-th$ element given as

$$(K_{\mathrm{xx}})_{i,j} = kr(x^i, x^j) \tag{18}$$

where $kr : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a positive definite kernel function defined as

$$kr(x^i, x^j) = \sigma^2 \exp\left(-0.5 \sum_{k=1}^{n} w_k \left|x_k^i - x_k^j\right|^2\right) \tag{19}$$

where x_k^i is the $k-th$ element of x^i, σ^2 is the variance parameter, and $w = (w_1, \cdots, w_n)$ with $w_k \geq 0$. It is shown in Appendix C that membership function as defined in (17) satisfies the consistency condition (11).

Remark 2 (Membership-Mapping as *Fuzzy-Mapping*) *Assuming that there exists of a fuzzy subset of $\mathbb{R}^{|\mathrm{x}|}$ with its membership function as ζ_{x}, the membership-mapping as per the terminology of (Kumar and Freudenthaler, 2019; Kumar et al., 2020) is a fuzzy-mapping.*

Remark 3 (The Effect of Degree of freedom ν) *Fig. 2 shows the effect of degree of freedom ν on the logarithmic membership value (i.e. $\log(\zeta_{\mathrm{x}}(\mathrm{y}))$) associated to a Student-t membership-mapping. A higher value of ν leads to an increased* spread *of the membership function. Thus, a relatively lower value of ν could alleviate the effect of outliers on the spread of membership function.*

2.1.6 Interpolation by Student-t Membership-Mapping

Let $\mathcal{F} \in \mathbb{F}(\mathbb{R}^n)$ be a zero-mean Student-t membership-mapping. Let $\mathrm{x} = \{x^i \in \mathbb{R}^n \mid i \in \{1, \cdots, N\}\}$ be a given set of input points. The corresponding mapping outputs, represented by the vector $\mathrm{f} := (\mathcal{F}(x^1), \cdots, \mathcal{F}(x^N))$, follow

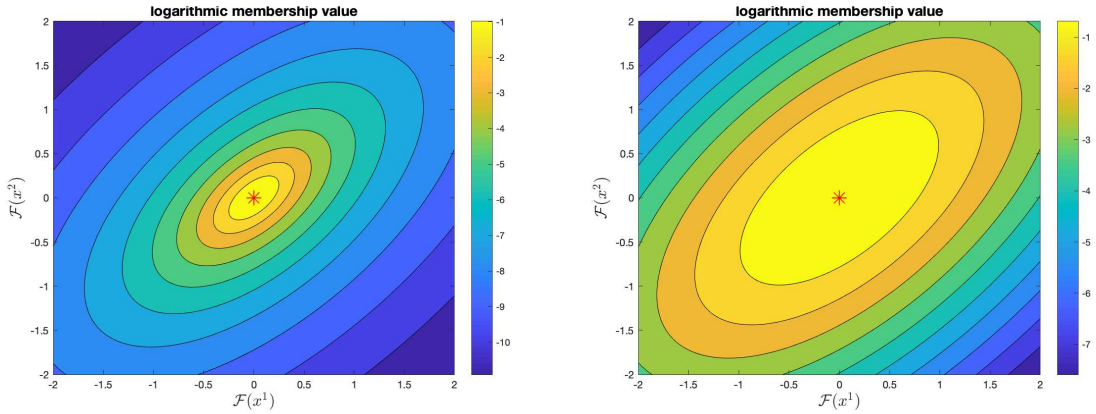
$$\zeta_{\mathrm{x}}(\mathrm{f}) = \left(1 + (1/(\nu - 2))\mathrm{f}^T K_{\mathrm{xx}}^{-1}\mathrm{f}\right)^{-\frac{\nu+N}{2}}. \tag{20}$$

Let $\mathrm{a} = \{a^m \mid a^m \in \mathbb{R}^n,\ m \in \{1, \cdots, M\}\}$ be the set of auxiliary inducing points. The mapping outputs corresponding to auxiliary inducing inputs, represented by the vector $\mathrm{u} := (\mathcal{F}(a^1), \cdots, \mathcal{F}(a^M))$, follow

$$\zeta_{\mathrm{a}}(\mathrm{u}) = \left(1 + (1/(\nu - 2))\mathrm{u}^T K_{\mathrm{aa}}^{-1}\mathrm{u}\right)^{-\frac{\nu+M}{2}} \tag{21}$$

where $K_{\mathrm{aa}} \in \mathbb{R}^{M\times M}$ is positive definite matrix with its $(i,j)-$th element given as

$$(K_{\mathrm{aa}})_{i,j} = kr(a^i, a^j) \tag{22}$$



(a) $\nu = 2.1$ (b) $\nu = 10$

Figure 2: The effect of degree of freedom ν on the logarithmic membership value (i.e. $\log(\zeta_{\mathrm{x}}(\mathrm{y}))$) associated to a Student-t membership-mapping is illustrated. A mapping $\mathcal{F} \in \mathbb{F}(\mathbb{R})$, for a given set of two input samples ($x^1 = -1$ and $x^2 = 1$), is considered with $\mathrm{m_y} = [0\ 0]^T$, $\sigma^2 = 1$, $w_1 = 0.25$, and taking two different values of ν. The figure displays the logarithmic membership value using a color map where the mean point $\mathrm{m_y}$ has been marked as *.

where $kr : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a positive definite kernel function defined as in (19). Similarly, the combined mapping outputs (f, u) follow

$$\zeta_{\mathrm{x}\wedge\mathrm{a}}((\mathrm{f},\mathrm{u})) = \left(1 + \frac{1}{\nu - 2}\left(\begin{bmatrix}\mathrm{f}\\ \mathrm{u}\end{bmatrix}\right)^T \begin{bmatrix}K_{\mathrm{xx}} & K_{\mathrm{xa}}\\ K_{\mathrm{ax}} & K_{\mathrm{aa}}\end{bmatrix}^{-1} \begin{bmatrix}\mathrm{f}\\ \mathrm{u}\end{bmatrix}\right)^{-\frac{\nu+N+M}{2}}. \tag{23}$$

It can be verified using a standard result regarding the inverse of a partitioned symmetric matrix that

$$\frac{\zeta_{\mathrm{x}\wedge\mathrm{a}}((\mathrm{f},\mathrm{u}))}{|\zeta_{\mathrm{a}}(\mathrm{u})|^{(\nu+N+M)/(\nu+M)}} = \left(1 + \frac{1}{\nu + M - 2}(\mathrm{f} - \bar{m}_{\mathrm{f}})^T \left(\frac{\nu + (\mathrm{u})^T (K_{\mathrm{aa}})^{-1}\mathrm{u} - 2}{\nu + M - 2}\bar{K}_{\mathrm{xx}}\right)^{-1} (\mathrm{f} - \bar{m}_{\mathrm{f}})\right)^{-\frac{\nu+M+N}{2}} \tag{24}$$

$$\bar{m}_{\mathrm{f}} = K_{\mathrm{xa}}(K_{\mathrm{aa}})^{-1}\mathrm{u} \tag{25}$$

$$\bar{K}_{\mathrm{xx}} = K_{\mathrm{xx}} - K_{\mathrm{xa}}(K_{\mathrm{aa}})^{-1}K_{\mathrm{xa}}^T. \tag{26}$$

The expression on the right hand side of equality (24) define a Student-t membership function with the mean $\bar{m}_{\mathrm{f}}$. It is observed from (25) that $\bar{m}_{\mathrm{f}}$ is an interpolation on the elements of u based on the closeness of points in x with that of a. Hence, f, based upon the interpolation on elements of u, could be represented by means of a membership function, $\mu_{\mathrm{f;u}} : \mathbb{R}^N \to [0, 1]$, defined as r.h.s. of (24):

$$\mu_{\mathrm{f;u}}(\tilde{\mathrm{f}}) := \left(1 + \frac{1}{\nu + M - 2}(\tilde{\mathrm{f}} - \bar{m}_{\mathrm{f}})^T \left(\frac{\nu + (\mathrm{u})^T (K_{\mathrm{aa}})^{-1}\mathrm{u} - 2}{\nu + M - 2}\bar{K}_{\mathrm{xx}}\right)^{-1} (\tilde{\mathrm{f}} - \bar{m}_{\mathrm{f}})\right)^{-\frac{\nu+M+N}{2}} \tag{27}$$

Here, the pair $(\mathbb{R}^N, \mu_{\mathrm{f;u}})$ constitutes a fuzzy set and $\mu_{\mathrm{f;u}}(\tilde{\mathrm{f}})$ is interpreted as the degree to which $\tilde{\mathrm{f}}$ matches an attribute induced by f for a given u.

2.2 Variational Learning

2.2.1 A Modeling Scenario

Given a dataset $\{(x^i, y^i) \mid x^i \in \mathbb{R}^n,\ y^i \in \mathbb{R}^p,\ i \in \{1, \cdots, N\}\}$, it is assumed that there exist zero-mean Student-t membership-mappings $\mathcal{F}_1, \cdots, \mathcal{F}_p \in \mathbb{F}(\mathbb{R}^n)$ such that

$$y^i \approx \big[\, \mathcal{F}_1(x^i)\ \cdots\ \mathcal{F}_p(x^i) \,\big]^T. \tag{28}$$

2.2.2 Disturbances and Auxiliary Inducing Points

For $j \in \{1, 2, \cdots, p\}$, define

$$\mathrm{y}_j = \big[\, y_j^1\ \cdots\ y_j^N \,\big]^T \in \mathbb{R}^N \tag{29}$$

$$\mathrm{f}_j = \big[\, \mathcal{F}_j(x^1)\ \cdots\ \mathcal{F}_j(x^N) \,\big]^T \in \mathbb{R}^N \tag{30}$$

where y_j^i denotes the $j-$th element of y^i. The vectors y_j and f_j will be subsequently referred to as *data* and *output* of membership-mappings, respectively. The difference between data and membership-mappings' outputs will be referred to as *disturbance* and denoted by v_j, i.e.,

$$\mathrm{v}_j = \mathrm{y}_j - \mathrm{f}_j. \tag{31}$$

A set of auxiliary inducing points, $\mathrm{a} = \{a^m \in \mathbb{R}^n \mid m \in \{1, \cdots, M\}\}$, is introduced. The membership-mappings' output values at auxiliary inducing input points are collected in a vector defined as

$$\mathrm{u}_j = \big[\, \mathcal{F}_j(a^1)\ \cdots\ \mathcal{F}_j(a^M) \,\big]^T \in \mathbb{R}^M. \tag{32}$$

2.2.3 Membership Functional Representation Approach

Definition 4 (Membership Functional Representation of Variables) *A variable $y \in \mathbf{Y}$ is represented by means of a membership function $\mu_y : \mathbf{Y} \to [0, 1]$, where the pair $(\mathbf{Y}, \mu_y)$ constitutes a fuzzy set and $\mu_y(\tilde{y})$ is interpreted as the degree to which a point $\tilde{y} \in \mathbf{Y}$ matches an attribute induced by $y \in \mathbf{Y}$.*

- *To represent a variable $x \in \mathbb{R}^n$, a Gaussian membership function, $\mu_x : \mathbb{R}^n \to [0, 1]$ could be considered:*

$$\mu_x(\tilde{x}) = \exp\big(-0.5(\tilde{x} - m_x)^T K_x^{-1}(\tilde{x} - m_x)\big) \tag{33}$$

  where $m_x \in \mathbb{R}^n$ is the mean vector and $K_x \in \mathbb{R}^{n\times n}$ is a positive definite matrix referred to as covariance matrix.

- *To represent a positive scalar $\tau > 0$, a Gamma like membership function, $\mu_\tau : \mathbb{R}_{>0} \to [0, 1]$, could be considered:*

$$\mu_\tau(\tilde{\tau}) = (b_\tau/(a_\tau - 1))^{a_\tau - 1} \exp(a_\tau - 1)(\tilde{\tau})^{a_\tau - 1} \exp(-b_\tau \tilde{\tau}) \tag{34}$$

  where $a_\tau \geq 1$ and $b_\tau > 0$.

- *Another form of Gamma like membership function for representing a positive scalar $z > 0$, $\mu_z : \mathbb{R}_{>0} \to [0, 1]$, could be as*

$$\mu_z(\tilde{z}) = (s)^r \exp(r)(\tilde{z})^r \exp(-rs\tilde{z}) \tag{35}$$

where $r \geq 0$ and $s > 0$.

Definition 5 (Representation of input variable x^i) *x^i is represented by means of a Gaussian membership function, $\mu_{x^i} : \mathbb{R}^n \to [0, 1]$, as*

$$\mu_{x^i}(\tilde{x}^i) = \exp\left(-0.5(1/\sigma_x^2)\|\tilde{x}^i - x^i\|^2\right) \tag{36}$$

where $\sigma_x^2 I_n$ is the covariance matrix.

Definition 6 (Disturbance-Model) *Disturbance vector v_j is represented by means of a zero-mean Gaussian membership function with scaled precision, $\mu_{\mathrm{v}_j} : \mathbb{R}^N \to [0, 1]$, as*

$$\mu_{\mathrm{v}_j}(\tilde{\mathrm{v}}_j) = \exp\left(-0.5\tau z\|\tilde{\mathrm{v}}_j\|^2\right) \tag{37}$$

where $\tau > 0$ is the precision which is scaled by a factor $z > 0$. The precision τ is represented by means of a Gamma membership function, $\mu_\tau : \mathbb{R}_{>0} \to [0, 1]$, as

$$\mu_\tau(\tilde{\tau}) = (b_\tau/(a_\tau - 1))^{a_\tau - 1} \exp(a_\tau - 1)(\tilde{\tau})^{a_\tau - 1} \exp(-b_\tau\tilde{\tau}) \tag{38}$$

where $a_\tau \geq 1$ and $b_\tau > 0$. The scaling factor z is represented by means of a Gamma membership function, $\mu_z : \mathbb{R}_{>0} \to [0, 1]$, as

$$\mu_z(\tilde{z}) = (s)^r \exp(r)(\tilde{z})^r \exp(-rs\tilde{z}). \tag{39}$$

Here, $r > 0$ and $s > 0$ are represented by means of Gamma membership functions, $\mu_r : \mathbb{R}_{>0} \to [0, 1]$ and $\mu_s : \mathbb{R}_{>0} \to [0, 1]$ respectively, as

$$\mu_r(\tilde{r}) = (b_r/(a_r - 1))^{a_r - 1} \exp(a_r - 1)(\tilde{r})^{a_r - 1} \exp(-b_r\tilde{r}) \tag{40}$$

$$\mu_s(\tilde{s}) = (b_s/(a_s - 1))^{a_s - 1} \exp(a_s - 1)(\tilde{s})^{a_s - 1} \exp(-b_s\tilde{s}) \tag{41}$$

where $a_r, a_s \geq 1$ and $b_r, b_r > 0$.

Remark 7 (Disturbance Model Precision) *It follows from (37) that precision matrix (the inverse of spread matrix) associated to membership function μ_{v_j} is given as*

$$H_{\mathrm{v}_j} = \tau z I_N. \tag{42}$$

where I_N is the identity matrix of size N.

Definition 8 (Representation of Data y_j for Given Mappings Output f_j) *Since $\mathrm{y}_j = \mathrm{f}_j + \mathrm{v}_j$, it follows from (37) that y_j, for given f_j, is represented by means of a membership function, $\mu_{\mathrm{y}_j;\mathrm{f}_j} : \mathbb{R}^N \to [0, 1]$, as*

$$\mu_{\mathrm{y}_j;\mathrm{f}_j}(\tilde{\mathrm{y}}_j) = \exp\left(-0.5\tau z\|\tilde{\mathrm{y}}_j - \mathrm{f}_j\|^2\right). \tag{43}$$

Definition 9 (Representation of Mappings Output f_j Based on Interpolation) *f_j, based upon an interpolation on the auxiliary-outputs u_j, is represented by means of a membership function, $\mu_{\mathrm{f}_j;\mathrm{u}_j} : \mathbb{R}^N \to [0,1]$, as*

$$\mu_{\mathrm{f}_j;\mathrm{u}_j}(\tilde{\mathrm{f}}_j) = \left(1 + \frac{1}{\nu + M - 2}(\tilde{\mathrm{f}}_j - \bar{m}_{\mathrm{f}_j})^T \left(\frac{\nu + (\mathrm{u}_j)^T (K_{\mathrm{aa}})^{-1}\mathrm{u}_j - 2}{\nu + M - 2}\bar{K}_{\mathrm{xx}}\right)^{-1} (\tilde{\mathrm{f}}_j - \bar{m}_{\mathrm{f}_j})\right)^{-\frac{\nu+M+N}{2}} \quad (44)$$

$$\bar{m}_{\mathrm{f}_j} = K_{\mathrm{xa}}(K_{\mathrm{aa}})^{-1}\mathrm{u}_j \quad (45)$$

$$\bar{K}_{\mathrm{xx}} = K_{\mathrm{xx}} - K_{\mathrm{xa}}(K_{\mathrm{aa}})^{-1}K_{\mathrm{xa}}^T. \quad (46)$$

Definition 10 (Representation of Data y_j for Fixed Auxiliary-Outputs u_j) *y_j, for given u_j, is represented by means of a membership function, $\mu_{\mathrm{y}_j;\mathrm{u}_j} : \mathbb{R}^N \to [0,1]$, as*

$$\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j) \propto \exp\left(\left\langle \cdots \left\langle \left\langle \log(\mu_{\mathrm{y}_j;\mathrm{f}_j}(\tilde{\mathrm{y}}_j))\right\rangle_{\mu_{\mathrm{f}_j;\mathrm{u}_j}} \right\rangle_{\mu_{x^1}} \cdots \right\rangle_{\mu_{x^N}}\right) \quad (47)$$

where $\mu_{\mathrm{y}_j;\mathrm{f}_j}$ is given by (43), $\mu_{\mathrm{f}_j;\mathrm{u}_j}$ is defined as in (44), and μ_{x^i} is defined as in (36), and $<\cdot>$. is the averaging operation as defined in (5). Thus, $\mu_{\mathrm{y}_j;\mathrm{u}_j}$ is obtained from $\log(\mu_{\mathrm{y}_j;\mathrm{f}_j})$ after averaging out the variables f_j and $(x^1, \cdots, x^N)$ using their respective membership functions. It is shown in Appendix D that

$$\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j) \propto \exp\Big(-0.5\tau z\|\tilde{\mathrm{y}}_j\|^2 + (\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\hat{m}_{\mathrm{u}_j}(\tilde{\mathrm{y}}_j) - 0.5(\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\mathrm{u}_j + 0.5(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j + \{/(\tilde{\mathrm{y}}_j, \mathrm{u}_j)\}\Big) \quad (48)$$

where $\hat{K}_{\mathrm{u}_j}$, $\hat{m}_{\mathrm{u}_j}(\tilde{\mathrm{y}}_j)$ are given by (156), (157) respectively, and $\{/(\tilde{\mathrm{y}}_j, \mathrm{u}_j)\}$ represents all those terms which are independent of both $\tilde{\mathrm{y}}_j$ and u_j. The constant of proportionality in (48) is chosen to exclude $(\tilde{\mathrm{y}}_j, \mathrm{u}_j)-$independent terms in the expression for $\mu_{\mathrm{y}_j;\mathrm{u}_j}$, i.e.,

$$\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j) = \exp\Big(-0.5\tau z\|\tilde{\mathrm{y}}_j\|^2 + (\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\hat{m}_{\mathrm{u}_j}(\tilde{\mathrm{y}}_j) - 0.5(\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\mathrm{u}_j + 0.5(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j\Big). \quad (49)$$

Definition 11 (Data-Model) *y_j is represented by means of a membership function, $\mu_{\mathrm{y}_j} : \mathbb{R}^N \to [0,1]$, as*

$$\mu_{\mathrm{y}_j}(\tilde{\mathrm{y}}_j) \propto \exp\left(\left\langle \log(\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j))\right\rangle_{\mu_{\mathrm{u}_j}}\right) \quad (50)$$

where $\mu_{\mathrm{y}_j;\mathrm{u}_j}$ is given by (49) and $\mu_{\mathrm{u}_j} : \mathbb{R}^M \to [0,1]$ is a membership function representing u_j. Thus, μ_{y_j} is obtained from $\log(\mu_{\mathrm{y}_j;\mathrm{u}_j})$ after averaging out the auxiliary-outputs u_j using membership function μ_{u_j}.

2.2.4 Variational Optimization of Data-Model

To determine μ_{u_j} for a given y_j, $\log(\mu_{\mathrm{y}_j}(\mathrm{y}_j))$ is maximized w.r.t. μ_{u_j} around an initial guess. It follows from (50) that maximization of $\log(\mu_{\mathrm{y}_j}(\mathrm{y}_j))$ is equivalent to the maximization of $\left\langle \log(\mu_{\mathrm{y}_j;\mathrm{u}_j}(\mathrm{y}_j))\right\rangle_{\mu_{\mathrm{u}_j}}$. The zero-mean Gaussian membership function with covariance as equal to K_{aa} is taken as the initial guess towards which the optimization problem is regularized.

Result 1 *The solution of following maximization problem:*

$$\mu^*_{\mathrm{u}_j} = \arg\max_{\mu_{\mathrm{u}_j}} \left[\left\langle \log(\mu_{\mathrm{y}_j;\mathrm{u}_j}(\mathrm{y}_j)) \right\rangle_{\mu_{\mathrm{u}_j}} - \left\langle \log(\frac{\mu_{\mathrm{u}_j}(\mathrm{u}_j)}{\exp\left(-0.5(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j\right)}) \right\rangle_{\mu_{\mathrm{u}_j}} \right] \tag{51}$$

under the fixed integral constraint:

$$\int_{\mathbb{R}^M} \mu_{\mathrm{u}_j} \, \mathrm{d}\lambda^M = C_{\mathrm{u}_j} > 0 \tag{52}$$

where the value of C_{u_j} is so chosen such that the maximum possible values of $\mu^*_{\mathrm{u}_j}$ remain as equal to unity, is given as

$$\mu^*_{\mathrm{u}_j}(\mathrm{u}_j) = \exp\left(-0.5\left(\mathrm{u}_j - \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T \hat{K}^{-1}_{\mathrm{u}_j}\left(\mathrm{u}_j - \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)\right) \tag{53}$$

where $\hat{K}_{\mathrm{u}_j}$ and $\hat{m}_{\mathrm{u}_j}$ are given by (156) and (157) respectively. This results in

$$\langle \mathrm{u}_j \rangle_{\mu^*_{\mathrm{u}_j}} = \tau z \hat{K}_{\mathrm{u}_j}(K_{\mathrm{aa}})^{-1}(\Psi)^T \mathrm{y}_j \tag{54}$$

$$\begin{aligned}\left\langle \log(\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j)) \right\rangle_{\mu^*_{\mathrm{u}_j}} &= -0.5\tau z\|\tilde{\mathrm{y}}_j\|^2 + \tau z\left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T(K_{\mathrm{aa}})^{-1}(\Psi)^T\tilde{\mathrm{y}}_j - 0.5\tau z\left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T\Big\{(K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1} \\ &\quad + \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2}(K_{\mathrm{aa}})^{-1}\Big\}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) - 0.5\tau z Tr\left((K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1}\hat{K}_{\mathrm{u}_j}\right. \\ &\quad \left. + \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2}(K_{\mathrm{aa}})^{-1}\hat{K}_{\mathrm{u}_j}\right)\end{aligned} \tag{55}$$

where ξ, Ψ, and Φ are given by (153), (154), and (155) respectively.

Proof: The proof is provided in Appendix E. ∎

The data-model (50) using (55) becomes as

$$\begin{aligned}\mu_{\mathrm{y}_j}(\tilde{\mathrm{y}}_j) \propto \exp\Big(&-0.5\tau z\|\tilde{\mathrm{y}}_j\|^2 + \tau z\left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T(K_{\mathrm{aa}})^{-1}(\Psi)^T\tilde{\mathrm{y}}_j \qquad (56)\\ &- 0.5\tau z\left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T\left\{(K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1} + \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2}(K_{\mathrm{aa}})^{-1}\right\}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) \\ &- 0.5\tau z Tr\left((K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1}\hat{K}_{\mathrm{u}_j} + \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2}(K_{\mathrm{aa}})^{-1}\hat{K}_{\mathrm{u}_j}\right)\Big). \qquad (57)\end{aligned}$$

That is,

$$\begin{aligned}\mu_{\mathrm{y}_j}(\tilde{\mathrm{y}}_j) \propto \exp\Big(&-0.5\tau z\Big[\|\tilde{\mathrm{y}}_j\|^2 - 2\left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T(K_{\mathrm{aa}})^{-1}(\Psi)^T\tilde{\mathrm{y}}_j + \left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T(K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) \\ &+ \left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T\frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2}(K_{\mathrm{aa}})^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\Big] + \{/(\mathrm{y}_j, \tilde{\mathrm{y}}_j)\}\Big)\end{aligned} \tag{58}$$

where $\{/(\mathrm{y}_j, \tilde{\mathrm{y}}_j)\}$ represents all $(\mathrm{y}_j, \tilde{\mathrm{y}}_j)-$independent terms. The constant of proportionality in (58) is chosen to exclude $(\mathrm{y}_j, \tilde{\mathrm{y}}_j)-$independent terms resulting in

$$\begin{aligned}\mu_{\mathrm{y}_j}(\tilde{\mathrm{y}}_j) = \exp\Big(&-0.5\tau z\Big[\|\tilde{\mathrm{y}}_j\|^2 - 2\left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T(K_{\mathrm{aa}})^{-1}(\Psi)^T\tilde{\mathrm{y}}_j + \left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T(K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) \\ &+ \left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T\frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2}(K_{\mathrm{aa}})^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\Big]\Big).\end{aligned} \tag{59}$$

2.2.5 Variational Optimization of Disturbance-Model

The data-model (59) involves disturbance parameters (τ, z) whose estimation would require an optimization of disturbance-model (Definition 6). Variational optimization is used to determine the membership functions representing τ and z for given $\{\mathrm{y}_1, \cdots, \mathrm{y}_p\}$ such that the averaged membership of data y_j to membership function μ_{y_j} (that serves as data-model) is maximized. Let $q_\tau : \mathbb{R}_{>0} \to [0,1]$, $q_z : \mathbb{R}_{>0} \to [0,1]$, $q_r : \mathbb{R}_{>0} \to [0,1]$, and $q_s : \mathbb{R}_{>0} \to [0,1]$ be arbitrary membership functions. For simplicity, define

$$\begin{aligned}\Omega &:= (\ \tau\ \ z\ \ r\ \ s\ )\\ q_\Omega\left((\ \tilde{\tau}\ \ \tilde{z}\ \ \tilde{r}\ \ \tilde{s}\ )\right) &:= q_\tau(\tilde{\tau}) q_z(\tilde{z}) q_r(\tilde{r}) q_s(\tilde{s})\\ \mu_\Omega\left((\ \tilde{\tau}\ \ \tilde{z}\ \ \tilde{r}\ \ \tilde{s}\ )\right) &:= \mu_\tau(\tilde{\tau}) \mu_z(\tilde{z}) \mu_r(\tilde{r}) \mu_s(\tilde{s})\end{aligned}$$

where $\mu_\tau(\tilde{\tau})$, $\mu_z(\tilde{z})$, $\mu_r(\tilde{r})$, and $\mu_s(\tilde{s})$ are defined by (38), (39), (40), and (41) respectively. We seek to maximize over q_Ω an objective functional defined as

$$J = \sum_{j=1}^{p} \left\langle \log\left(\mu_{\mathrm{y}_j}(\mathrm{y}_j)\right)\right\rangle_{q_\Omega} + 0.5 \sum_{j=1}^{p} \left\langle \log(|H_{\mathrm{v}_j}|)\right\rangle_{q_\Omega} - \left\langle \log\left(q_\Omega(\Omega)/\mu_\Omega(\Omega)\right)\right\rangle_{q_\Omega}. \tag{60}$$

Remark 12 (Interpretation of Objective Functional (60)) *The first term in the expression of J indicates that the optimization problem would maximize the sum (over number of variables) of averaged membership of data y_j to the membership function μ_{y_j} (data-model). The second term in the expression of J would derive the solution of optimization problem towards maximization of averaged value of logarithmic determinant of the precision matrix associated to the disturbance model. The third term in the expression of J would regularizes the optimization problem towards initial guess μ_Ω.*

Problem 1 (Variational Optimization of Disturbance-Model) *Solve*

$$\{q_\tau^*, q_z^*, q_r^*, q_s^*\} = \arg \max_{\{q_\tau, q_z, q_r, q_s\}} J \tag{61}$$

under the following fixed integral constraints:

$$\int_{\mathbb{R}_{>0}} q_\tau\ \mathrm{d}\lambda^1 = C_\tau > 0, \tag{62}$$

$$\int_{\mathbb{R}_{>0}} q_z\ \mathrm{d}\lambda^1 = C_z > 0, \tag{63}$$

$$\int_{\mathbb{R}_{>0}} q_r\ \mathrm{d}\lambda^1 = C_r > 0, \tag{64}$$

$$\int_{\mathbb{R}_{>0}} q_s\ \mathrm{d}\lambda^1 = C_s > 0 \tag{65}$$

where the values of C_τ, C_z, C_r, and C_s are so chosen such that the maximum possible values of q_τ^*, q_z^*, q_r^*, and q_s^* remain as equal to unity.

Result 2 *The analytical expressions for variational membership functions, that solve Problem 1, are*

$$q_\tau^*(\tilde{\tau}) = \left(\hat{b}_\tau/(\hat{a}_\tau - 1)\right)^{\hat{a}_\tau - 1} \exp(\hat{a}_\tau - 1)(\tilde{\tau})^{\hat{a}_\tau - 1} \exp\left(-\hat{b}_\tau \tilde{\tau}\right) \tag{66}$$

$$\hat{a}_\tau = a_\tau + 0.5Np \tag{67}$$

$$\hat{b}_\tau(O) = b_\tau + \frac{\hat{a}_z}{2\hat{b}_z} O \tag{68}$$

$$q_z^*(\tilde{z}) = \left(\hat{b}_z/(\hat{a}_z - 1)\right)^{\hat{a}_z - 1} \exp(\hat{a}_z - 1)(\tilde{z})^{\hat{a}_z - 1} \exp\left(-\hat{b}_z \tilde{z}\right) \tag{69}$$

$$\hat{a}_z = 1 + 0.5Np + \hat{a}_r/\hat{b}_r \tag{70}$$

$$\hat{b}_z(O) = \frac{\hat{a}_r}{\hat{b}_r}\frac{\hat{a}_s}{\hat{b}_s} + \frac{\hat{a}_\tau}{2\hat{b}_\tau} O \tag{71}$$

$$q_r^*(\tilde{r}) = \left(\hat{b}_r/(\hat{a}_r - 1)\right)^{\hat{a}_r - 1} \exp(\hat{a}_r - 1)(\tilde{r})^{\hat{a}_r - 1} \exp\left(-\hat{b}_r \tilde{r}\right) \tag{72}$$

$$\hat{a}_r = a_r \tag{73}$$

$$\hat{b}_r = b_r + (\hat{a}_s/\hat{b}_s)(\hat{a}_z/\hat{b}_z) - \psi\left(\hat{a}_s\right) + \log\left(\hat{b}_s\right) - 1 - \psi\left(\hat{a}_z\right) + \log\left(\hat{b}_z\right) \tag{74}$$

$$q_s^*(\tilde{s}) = \left(\hat{b}_s/(\hat{a}_s - 1)\right)^{\hat{a}_s - 1} \exp(\hat{a}_s - 1)(\tilde{s})^{\hat{a}_s - 1} \exp\left(-\hat{b}_s \tilde{s}\right) \tag{75}$$

$$\hat{a}_s = a_s + (\hat{a}_r/\hat{b}_r) \tag{76}$$

$$\hat{b}_s = b_s + (\hat{a}_r/\hat{b}_r)(\hat{a}_z/\hat{b}_z) \tag{77}$$

where

$$O = \sum_{j=1}^{p} \Big(\|\mathrm{y}_j\|^2 - 2\left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T (K_{\mathrm{aa}})^{-1} (\Psi)^T \mathrm{y}_j + \left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T (K_{\mathrm{aa}})^{-1} \Phi (K_{\mathrm{aa}})^{-1} \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) + \left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2} (K_{\mathrm{aa}})^{-1} \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) \Big). \tag{78}$$

Proof: The proof is provided in Appendix F. ■

2.2.6 Estimation of Membership-Mapping Output

Definition 13 (Averaged Estimation of Membership-Mapping Output) *$\mathcal{F}_j(x^i)$ (which is the $i-$th element of vector f_j (30)) can be estimated as*

$$\widehat{\mathcal{F}_j(x^i)} := \left\langle \left\langle \langle (\mathrm{f}_j)_i \rangle_{\mu_{\mathrm{f}_j;\mathrm{u}_j}} \right\rangle_{\mu_{x^i}} \right\rangle_{\mu_{\mathrm{u}_j}} \tag{79}$$

where $(\mathrm{f}_j)_i$ denotes the $i-$th element of f_j, $\mu_{\mathrm{f}_j;\mathrm{u}_j}$ is defined as in (44), μ_{x^i} is defined as in (36), and $\mu_{\mathrm{u}_j} : \mathbb{R}^M \to [0,1]$ is a membership function representing u_j. That is, $\mathcal{F}_j(x^i)$, being a function of x^i and u_j, is averaged over x^i and u_j for an averaged estimation.

Using (44) and (45), we have

$$\langle(\mathrm{f}_j)_i\rangle_{\mu_{\mathrm{f}_j;\mathrm{u}_j}} = (K_{\mathrm{xa}}(K_{\mathrm{aa}})^{-1}\mathrm{u}_j)_i \tag{80}$$

$$= \left[\, kr(x^i,a^1)\ \cdots\ kr(x^i,a^M)\,\right](K_{\mathrm{aa}})^{-1}\mathrm{u}_j. \tag{81}$$

Thus,

$$\widehat{\mathcal{F}_j(x^i)} = \left[\,\left\langle kr(x^i,a^1)\right\rangle_{\mu_{x^i}}\ \cdots\ \left\langle kr(x^i,a^M)\right\rangle_{\mu_{x^i}}\,\right](K_{\mathrm{aa}})^{-1}\left\langle \mathrm{u}_j\right\rangle_{\mu_{\mathrm{u}_j}}. \tag{82}$$

For given set of parameters: $w = (w_1,\cdots,w_n)$ with $w_k > 0$, $\mathrm{a} = \{a^m \in \mathbb{R}^n \mid m \in \{1,\cdots,M\}\}$, $\sigma^2 > 0$, and $\sigma_x^2 > 0$; let $G(x) \in \mathbb{R}^{1\times M}$ be a vector-valued function whose $m-$th element for any $x \in \mathbb{R}^n$ is defined as

$$G_m(x) := \frac{\sigma^2}{\prod_{k=1}^n\left(\sqrt{1+w_k\sigma_x^2}\right)}\exp\left(-\frac{1}{2}\sum_{k=1}^n\frac{w_k|a_k^m - x_k|^2}{1+w_k\sigma_x^2}\right) \tag{83}$$

where a_k^m and x_k are the $k-$th elements of x and a^m respectively. It follows from (83), (154), and (151) that

$$\left\langle kr(x^i,a^m)\right\rangle_{\mu_{x^i}} = G_m(x^i). \tag{84}$$

Using (84) and (54) in (82), we have

$$\widehat{\mathcal{F}_j(x^i)} = \tau z\left(G(x^i)\right)(K_{\mathrm{aa}})^{-1}\hat{K}_{\mathrm{u}_j}(K_{\mathrm{aa}})^{-1}(\Psi)^T\mathrm{y}_j. \tag{85}$$

Substituting $\hat{K}_{\mathrm{u}_j}$ from (156) in (85), we get

$$\widehat{\mathcal{F}_j(x^i)} = \left(G(x^i)\right)\left(\Phi + \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu+M-2}K_{\mathrm{aa}} + \frac{K_{\mathrm{aa}}}{\tau z}\right)^{-1}(\Psi)^T\mathrm{y}_j. \tag{86}$$

It follows from (66) and (69) that the average value of product τz is given as

$$\left\langle\left\langle\tau z\right\rangle_{q_\tau^*}\right\rangle_{q_z^*} = (\hat{a}_\tau/\hat{b}_\tau)(\hat{a}_z/\hat{b}_z). \tag{87}$$

The value of $\widehat{\mathcal{F}_j(x^i)}$ can be estimated via computing average value of the product τz from (87) and then using (86). Thus, an estimated value of $\widehat{\mathcal{F}_j(x^i)}$, denoted as $\mathcal{E}(\widehat{\mathcal{F}_j(x^i)})$, is defined as

$$\mathcal{E}(\widehat{\mathcal{F}_j(x^i)}) := \left(G(x^i)\right)\left(\Phi + \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu+M-2}K_{\mathrm{aa}} + \frac{\hat{b}_\tau\hat{b}_z}{\hat{a}_\tau\hat{a}_z}K_{\mathrm{aa}}\right)^{-1}(\Psi)^T\mathrm{y}_j. \tag{88}$$

Let $\alpha = [\,\alpha_1\ \cdots\ \alpha_p\,] \in \mathbb{R}^{M\times p}$ be a matrix with its $j-$th column defined as

$$\alpha_j := \left(\Phi + \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu+M-2}K_{\mathrm{aa}} + \frac{\hat{b}_\tau}{\hat{a}_\tau}\frac{\hat{b}_z}{\hat{a}_z}K_{\mathrm{aa}}\right)^{-1}(\Psi)^T\mathrm{y}_j \tag{89}$$

so that $\mathcal{E}(\widehat{\mathcal{F}_j(x^i)})$ could be expressed as

$$\mathcal{E}(\widehat{\mathcal{F}_j(x^i)}) = \big(G(x^i)\big)\,\alpha_j. \tag{90}$$

Further, define a matrix $B \in \mathbb{R}^{M\times N}$ as

$$B := \left(\Phi + \frac{\xi - Tr((K_{\rm aa})^{-1}\Phi)}{\nu + M - 2}K_{\rm aa} + \frac{\hat{b}_\tau}{\hat{a}_\tau}\frac{\hat{b}_z}{\hat{a}_z}K_{\rm aa}\right)^{-1}(\Psi)^T. \tag{91}$$

Remark 14 (Estimation of $\hat{m}_{\mathrm{u}_j}$) *It follows from (156) and (157) that $\hat{m}_{\mathrm{u}_j}$ is given as*

$$\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) = K_{\rm aa}\left(\Phi + \frac{\xi - Tr((K_{\rm aa})^{-1}\Phi)}{\nu + M - 2}K_{\rm aa} + \frac{K_{\rm aa}}{\tau z}\right)^{-1}(\Psi)^T\mathrm{y}_j. \tag{92}$$

The value of $\hat{m}_{\mathrm{u}_j}$ can be estimated via computing average value of the product τz from (87) and then using (92). Thus, an estimated value of $\hat{m}_{\mathrm{u}_j}$, denoted as $\mathcal{E}(\hat{m}_{\mathrm{u}_j})$, is defined as

$$\mathcal{E}(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)) := K_{\rm aa}\left(\Phi + \frac{\xi - Tr((K_{\rm aa})^{-1}\Phi)}{\nu + M - 2}K_{\rm aa} + \frac{\hat{b}_\tau\hat{b}_z}{\hat{a}_\tau\hat{a}_z}K_{\rm aa}\right)^{-1}(\Psi)^T\mathrm{y}_j. \tag{93}$$

Remark 15 (Precision of the Disturbance Model) *A measure of the precision of disturbance model is defined from the logarithmic determinant of the precision matrix as*

$$\beta := (1/N)\mathcal{E}(\log(|H_{\mathrm{v}_j}|)) \tag{94}$$
$$= (\hat{a}_\tau/\hat{b}_\tau)(\hat{a}_z/\hat{b}_z). \tag{95}$$

2.2.7 Learning Algorithm and Predictions

Algorithm 1 is suggested for the variational learning of membership-mappings. Given the parameters set $\mathbb{M} = \{\alpha, w, \mathrm{a}, \sigma^2, \sigma_x^2, B\}$ returned by Algorithm 1, the learned membership-mappings could be used to predict output corresponding to any arbitrary input data point $x^* \in \mathbb{R}^n$ as

$$\hat{y}(x^*;\mathbb{M}) = \Big[\,\mathcal{E}(\widehat{\mathcal{F}_1(x^*)})\ \cdots\ \mathcal{E}(\widehat{\mathcal{F}_p(x^*)})\,\Big]^T \tag{97}$$

where $\mathcal{E}(\widehat{\mathcal{F}_j(x^*)})$, defined as in (90), is the estimated averaged output of $j-$th membership-mapping. It follows from (90) that

$$\hat{y}(x^*;\mathbb{M}) = \alpha^T(G(x^*))^T \tag{98}$$

where $G(\cdot) \in \mathbb{R}^{1\times M}$ is a vector-valued function whose elements are defined as in (83).

Algorithm 1 Variational learning of the membership-mappings

Require: Dataset $\big\{(x^i, y^i) \mid x^i \in \mathbb{R}^n,\ y^i \in \mathbb{R}^p,\ i \in \{1, \cdots, N\}\big\}$; number of auxiliary points $M \in \{1, 2, \cdots, N\}$; the degrees of freedom associated to the Student-t membership-mapping $\nu \in \mathbb{R}_+ \setminus [0, 2]$.

Choose free parameters as $\sigma^2 = 1$ and $\sigma_x^2 = 0.01$.

The auxiliary inducing points are suggested to be chosen as the cluster centroids:

$$\mathrm{a} = \{a^m\}_{m=1}^M = \mathit{cluster\_centroid}(\{x^i\}_{i=1}^N, M)$$

where $\mathit{cluster\_centroid}(\{x^i\}_{i=1}^N, M)$ represents the k-means clustering on $\{x^i\}_{i=1}^N$.

Define $w = (w_1, w_2, \cdots, w_n)$ such that w_k (for $k \in \{1, 2, \cdots, n\}$) is equal to the inverse of squared-distance between two most-distant points in the set: $\{x_k^1, x_k^2, \cdots, x_k^N\}$.

Compute K_{aa}, ξ, Ψ, and Φ using (22), (153), (154), and (155) respectively.

Choose $a_\tau = b_\tau = a_r = b_r = a_s = b_s = 1$.

Initialise $\hat{a}_\tau = \hat{b}_\tau = \hat{a}_z = \hat{b}_z = \hat{a}_r = \hat{b}_r = 1$.

Initialize $\hat{a}_s$ and $\hat{b}_s$ using (76) and (77).

repeat

  Update $\mathcal{E}(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j))$ using (93).

  Estimate O using (78) as

$$\mathcal{E}(O) = \sum_{j=1}^{p} \Big(\|\mathrm{y}_j\|^2 - 2 \left(\mathcal{E}(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j))\right)^T (K_{\mathrm{aa}})^{-1} (\Psi)^T \mathrm{y}_j + \left(\mathcal{E}(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j))\right)^T (K_{\mathrm{aa}})^{-1} \Phi (K_{\mathrm{aa}})^{-1} \mathcal{E}(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)) + \left(\mathcal{E}(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j))\right)^T \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2} (K_{\mathrm{aa}})^{-1} \mathcal{E}(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)) \Big). \tag{96}$$

  Update $\hat{a}_\tau, \hat{b}_\tau(\mathcal{E}(O)), \hat{a}_z, \hat{b}_z(\mathcal{E}(O)), \hat{a}_r, \hat{b}_r, \hat{a}_s, \hat{b}_s$ using (67), (68), (70), (71), (73), (74), (76), (77) respectively.

  Estimate the precision of the disturbance model, β, using (95).

until (β nearly converges)

Compute matrix B using (91) and matrix $\alpha = [\,\alpha_1 \ \cdots \ \alpha_p\,]$ using (89).

return The parameters set $\mathbb{M} = \{\alpha, w, \mathrm{a}, \sigma^2, \sigma_x^2, B\}$.

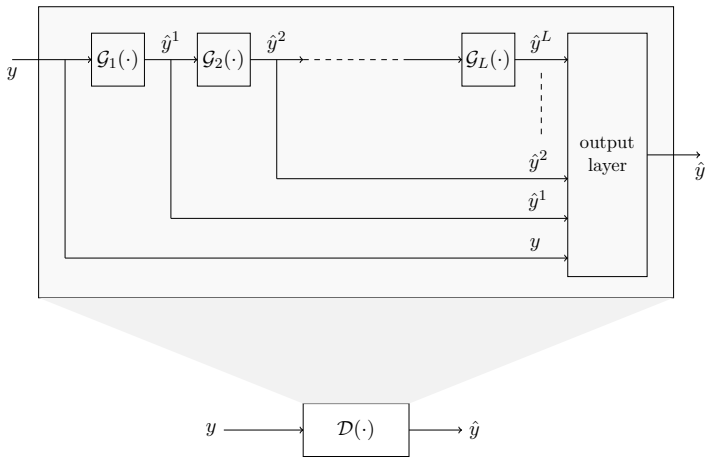


Figure 3: The structure of an $L-$layered conditionally deep autoencoder consisting of a nested compositions of membership-mapping autoencoders.

3. Conditionally Deep Membership-Mapping Autoencoders

3.1 Architecture and Learning

Definition 16 (Membership-Mapping Autoencoder) *A membership-mapping autoencoder, $\mathcal{G} : \mathbb{R}^p \to \mathbb{R}^p$, maps an input vector $y \in \mathbb{R}^p$ to $\mathcal{G}(y) \in \mathbb{R}^p$ such that*

$$\mathcal{G}(y) \stackrel{\text{def}}{=} [\, \mathcal{F}_1(Py) \ \cdots \ \mathcal{F}_p(Py) \,]^T, \tag{99}$$

where $\mathcal{F}_j$ ($j \in \{1, 2, \cdots, p\}$) is a Student-t membership-mapping (Definition 1), $P \in \mathbb{R}^{n \times p} (n \leq p)$ is a matrix such that the product Py is a lower-dimensional encoding for y. That is, membership-mapping autoencoder first projects the input vector onto a lower dimensional subspace and then constructs the output vector through Student-t membership-mappings.

Definition 17 (Conditionally Deep Membership-Mapping Autoencoder (CDMMA)) *A conditionally deep membership-mapping autoencoder, $\mathcal{D} : \mathbb{R}^p \to \mathbb{R}^p$, maps a vector $y \in \mathbb{R}^p$ to $\mathcal{D}(y) \in \mathbb{R}^p$ through a nested composition of finite number of membership-mapping autoencoders such that*

$$\hat{y}^l = (\mathcal{G}_l \circ \cdots \circ \mathcal{G}_2 \circ \mathcal{G}_1)(y), \ \forall l \in \{1, 2, \cdots, L\} \tag{100}$$

$$l^* = \arg \min_{l \in \{1,2,\cdots,L\}} \|y - \hat{y}^l\|^2 \tag{101}$$

$$\mathcal{D}(y) = \hat{y}^{l^*}, \tag{102}$$

where $\mathcal{G}_l(\cdot)$ is a membership-mapping autoencoder (Definition 16); $\hat{y}^l$ is the output of $l-$th layer representing input vector y at certain abstraction level such that $\hat{y}^1$ is least abstract representation and $\hat{y}^L$ is most abstract representation of the input vector; and the autoencoder output $\mathcal{D}(y)$ is equal to the output of the layer re-constructing the given input vector

as good as possible where re-construction error is measured in-terms of squared Euclidean distance. The structure of deep autoencoder (as displayed in Fig. 3) is such that

$$\begin{aligned}\hat{y}^l &= \mathcal{G}_l(\hat{y}^{l-1}),\\ &= \left[\, \mathcal{F}_1^l(P^l\hat{y}^{l-1}) \;\cdots\; \mathcal{F}_p^l(P^l\hat{y}^{l-1}) \,\right]^T\end{aligned}$$

where $\hat{y}^0 = y$, $P^l \in \mathbb{R}^{n_l \times p}$ *is a matrix with* $n_l \in \{1, \cdots, p\}$ *such that* $n_1 \geq n_2 \geq \cdots \geq n_L$, *and* $\mathcal{F}_j^l(\cdot)$ *is a Student-t membership-mapping.*

Given a set of N samples $\{y^1, \cdots, y^N\}$, the learning problem is of deriving an expression for the output of each layer of CDMMA under some optimality criterion. Since CDMMA consists of layers of membership-mappings, Algorithm 1 could be directly applied for the variational learning of individual layers. Thus, Algorithm 2 is suggested for the variational learning of CDMMA. The salient features of Algorithm 2 are as follow:

Algorithm 2 Variational learning of CDMMA

Require: Data set $\mathbf{Y} = \left\{y^i \in \mathbb{R}^p \mid i \in \{1, \cdots, N\}\right\}$; the subspace dimension $n \in \{1, 2, \cdots, p\}$; number of auxiliary points $M \in \{1, 2, \cdots, N\}$; the number of layers $L \in \mathbb{Z}_+$.

1: Choose free parameters as $\nu^1 = 2.1, \nu^2 = \infty, \cdots, \nu^L = \infty$.

2: **for** $l = 1$ to L **do**

3: Set subspace dimension associated to $l-$th layer as $n_l = \max(n - l + 1, 1)$.

4: Define $P^l \in \mathbb{R}^{n_l \times p}$ such that $i-$th row of P^l is equal to transpose of eigenvector corresponding to $i-$th largest eigenvalue of sample covariance matrix of data set $\mathbf{Y}$.

5: Define a latent variable $x^{l,i} \in \mathbb{R}^{n_l}$, for $i \in \{1, \cdots, N\}$, as

$$x^{l,i} = \begin{cases} P^l y^i & \text{if } l = 1,\\ P^l \hat{y}^{l-1}(x^{l-1,i}; \mathbb{M}^{l-1}) & \text{if } l > 1 \end{cases} \tag{103}$$

where $\hat{y}^{l-1}$ is the estimated output of the $(l-1)-$th layer computed using (98) for the parameters set $\mathbb{M}^{l-1} = \{\alpha^{l-1}, w^{l-1}, \mathrm{a}^{l-1}, \sigma^2, \sigma_x^2, B^{l-1}\}$.

6: Compute parameters set $\mathbb{M}^l$ characterizing the membership-mappings associated to $l-$th layer by applying Algorithm 1 on data set $\left\{(x^{l,i}, y^i) \mid i \in \{1, \cdots, N\}\right\}$ with number of auxiliary points M and degrees of freedom as ν^l.

7: **end for**

8: **return** The parameters set $\mathcal{M} = \{\{\mathbb{M}^1, \cdots, \mathbb{M}^L\}, \{P^1, \cdots, P^L\}\}$.

- Following Kumar and Freudenthaler (2019), the degree of freedom $\nu^l \in \mathbb{R}_+ \setminus [0, 2]$ for $l = 1$ (i.e. for the first layer) is sufficiently low for a robust filtering of high-dimensional data. As the uncertainties on input data have been filtered out by the first layer, ν^l for subsequent layers (i.e. for $l > 1$) is increased to ∞ so that the precision of disturbance model increases as high as possible.

- CDMMA discovers layers of increasingly abstract data representation as a result of letting $\{n_1, \cdots, n_L\}$ a monotonically decreasing sequence at step 3 of Algorithm 2. This will be illustrated in Fig. 4. As observed in Fig. 4, the first layer of CDMMA models the lowest level data-features while moving deep across the layer the higher level data-features are modeled.

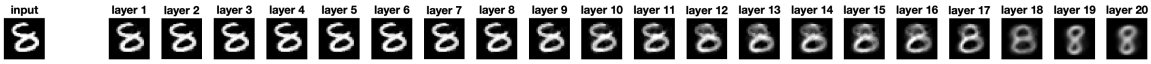


Figure 4: A CDMMA was built using Algorithm 2 (taking $n = 20$; $M = 500$; $L = 20$) on a dataset consisting of 1000 randomly chosen samples of digit 8 from MNIST digits dataset. Corresponding to the input sample (shown at the extreme left of the figure), the estimated outputs of different layers of deep autoencoder are displayed. It is observed that CDMMA, as a result of letting $\{n_1, \cdots, n_L\}$ a monotonically decreasing sequence at step 3 of Algorithm 2, discover layers of increasingly abstract data representation with lowest-level data features being modeled by first layer and the highest-level by end layer.

Definition 18 (Filtering by CDMMA) *Given a CDMMA with its parameters being represented by a set $\mathcal{M} = \{\{\mathbb{M}^1, \cdots, \mathbb{M}^L\}, \{P^1, \cdots, P^L\}\}$, the autoencoder can be applied for filtering a given input vector $y \in \mathbb{R}^p$ as follows:*

$$x^l(y; \mathcal{M}) = \begin{cases} P^l y, & l = 1 \\ P^l \hat{y}^{l-1}(x^{l-1}; \mathbb{M}^{l-1}) & l \geq 2 \end{cases} \tag{104}$$

Here, $\hat{y}^{l-1}$ is the output of the $(l-1)-$th layer estimated using (98). Finally, CDMMA's output, $\mathcal{D}(y; \mathcal{M})$, is estimated as

$$\mathcal{E}(\mathcal{D}(y; \mathcal{M})) = \hat{y}^{l^*}(x^{l^*}; \mathbb{M}^{l^*}) \tag{105}$$

$$l^* = \arg \min_{l \in \{1, \cdots, L\}} \|y - \hat{y}^l(x^l; \mathbb{M}^l)\|^2. \tag{106}$$

For a big dataset i.e. N is large, Algorithm 2 may require a larger M. A higher value M would increase the computational time required by Algorithm 2 for learning. To circumvent the problem of large computation time for processing big data, it is suggested that data be partitioned into subsets and corresponding to each data-subset a separate CDMMA is learned. This motivates defining of a wide conditionally deep autoencoder as in Definition 19.

Definition 19 (A Wide CDMMA) *A wide CDMMA, $\mathcal{WD} : \mathbb{R}^p \to \mathbb{R}^p$, maps a vector $y \in \mathbb{R}^p$ to $\mathcal{WD}(y) \in \mathbb{R}^p$ through a parallel composition of S ($S \in \mathcal{Z}_+$) number of CDMMAs such that*

$$\mathcal{WD}(y; \mathcal{P} = \{\mathcal{M}^s\}_{s=1}^S) = \mathcal{D}(y; \mathcal{M}^{s^*}) \tag{107}$$

$$s^* = \arg \min_{s \in \{1, 2, \cdots, S\}} \|y - \mathcal{D}(y; \mathcal{M}^s)\|^2, \tag{108}$$

where $\mathcal{D}(y; \mathcal{M}^s)$ is the output of $s-$th CDMMA being characterized by parameters set $\mathcal{M}^s$. The estimated output of wide CDMMA is as

$$\mathcal{E}(\mathcal{WD}(y; \mathcal{P})) = \mathcal{E}(\mathcal{D}(y; \mathcal{M}^{s^*})), \tag{109}$$

where $\mathcal{E}(\mathcal{D}(y; \mathcal{M}^s))$, computed using (105), is estimated output of $s-$th CDMMA.

Algorithm 3 is suggested for the variational learning of wide CDMMA.

Algorithm 3 Variational learning of wide CDMMA

Require: Data set $\mathbf{Y} = \{y^i \in \mathbb{R}^p \mid i \in \{1, \cdots, N\}\}$; the subspace dimension $n \in \{1, 2, \cdots, p\}$; ratio $M/N \in (0, 1]$; the number of layers $L \in \mathbb{Z}_+$.

1: Apply k-means clustering to partition $\mathbf{Y}$ into S subsets, $\{\mathbf{Y}^1, \cdots, \mathbf{Y}^S\}$, where $S = \lceil N/1000 \rceil$.
2: **for** $s = 1$ to S **do**
3: Build a CDMMA, $\mathcal{M}^s$, by applying Algorithm 2 on $\mathbf{Y}^s$ taking n as the subspace dimension; the number of auxiliary points as equal to $(M/N) \times \#\mathbf{Y}^s$ (where $\#\mathbf{Y}^s$ is the number of data points in $\mathbf{Y}^s$); and L as the number of layers.
4: **end for**
5: **return** the set of parameters sets: $\mathcal{P} = \{\mathcal{M}^s\}_{s=1}^S$.

3.2 Demonstrative Examples

Algorithm 3 requires choosing a value of n and M/N. Fig. 5 and Fig. 6 illustrate the effect of n and M/N respectively.

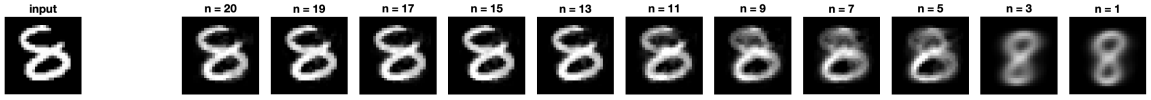


Figure 5: On a dataset consisting of 1000 randomly chosen samples of digit 8 from MNIST digits dataset, different wide CDMMAs were built using Algorithm 3 choosing $M/N = 0.25$; $L = 5$; and n from $\{20, 19, 17, 15, 13, 11, 9, 7, 5, 3, 1\}$. Corresponding to the input sample (shown at the extreme left of the figure), the estimated outputs of different wide CDMMAs (built using different values of n) are displayed. It is observed that as n keeps on decreasing, the autoencoder learns increasingly abstract data representation.

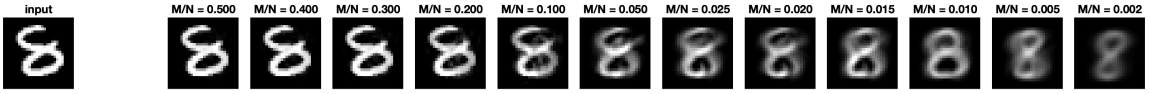


Figure 6: On a dataset consisting of 1000 randomly chosen samples of digit 8 from MNIST digits dataset, different wide CDMMAs were built using Algorithm 3 choosing M/N from $\{0.5, 0.4, 0.3, 0.2, 0.1, 0.05, 0.025, 0.02, 0.015, 0.01, 0.005, 0.002\}$; $L = 5$; $n = 20$. Corresponding to the input sample (shown at the extreme left of the figure), the estimated outputs of different wide CDMMAs (built using different M/N values) are displayed. It is observed that as the M/N value keeps on decreasing, the autoencoder learns increasingly abstract data representation.

3.3 Classification Applications

An application of deep autoencoder to classification follows via learning data representation for each class through a wide CDMMA. This motivates the defining of a classifier as in Definition 20.

Definition 20 (A Classifier) *A classifier,* $\mathcal{C} : \mathbb{R}^p \to \{1, 2, \cdots, C\}$*, maps a vector* $y \in \mathbb{R}^p$ *to* $\mathcal{C}(y) \in \{1, 2, \cdots, C\}$ *such that*

$$\mathcal{C}(y; \{\mathcal{P}_c\}_{c=1}^{C}) = \arg \min_{c \in \{1,2,\cdots,C\}} \|y - \mathcal{E}(\mathcal{WD}(y; \mathcal{P}_c))\|^2 \tag{110}$$

where $\mathcal{E}(\mathcal{WD}(y; \mathcal{P}_c))$*, computed using (109), is the estimated output of* $c-$*th wide CDMMA. The classifier assigns to an input vector the label of that class whose associated autoencoder best reconstructs the input vector.*

Finally, Algorithm 4 is provided for the learning of the classifier.

Algorithm 4 Variational learning of the classifier

Require: Labeled data set $\mathbf{Y} = \left\{\mathbf{Y}_c \mid \mathbf{Y}_c = \left\{y^{i,c} \in \mathbb{R}^p \mid i \in \{1, \cdots, N_c\}\right\}, c \in \{1, \cdots, C\}\right\}$; the subspace dimension $n \in \{1, \cdots, p\}$; ratio $M/N \in (0, 1]$; the number of layers $L \in \mathbb{Z}_+$.

1: **for** $c = 1$ to C **do**
2: Build a wide CDMMA, $\mathcal{P}_c = \{\mathcal{M}_c^s\}_{s=1}^{S_c}$, by applying Algorithm 3 on $\mathbf{Y}_c$ for given n, M/N, and L.
3: **end for**
4: **return** the set of parameters sets $\{\mathcal{P}_c\}_{c=1}^{C}$.

4. Privacy-Preserving Transfer Learning

4.1 An Optimal $(\epsilon, \delta)-$Differentially Private Noise Adding Mechanism

This subsection reviews an optimal noise adding mechanism that was derived using an information theoretic approach in (Kumar et al., 2019). We consider a training dataset consisting of N number of samples with each sample having p number of attributes. Assuming the data as numeric, the dataset can be represented by a matrix, say $\mathrm{Y} \in \mathbb{R}^{p \times N}$. The machine learning algorithms typically train a model using available dataset. A given machine learning algorithm, training a model using data matrix Y, can be represented by a mapping, $\mathcal{A} : \mathbb{R}^{p \times N} \to \mathbf{M}$, where $\mathbf{M}$ is the model space. That is, for a given dataset Y, the algorithm builds a model $\mathcal{M} \in \mathbf{M}$ such that $\mathcal{M} = \mathcal{A}(\mathrm{Y})$. The privacy of data can be preserved via adding a suitable random noise to data matrix before the application of algorithm $\mathcal{A}$ on the dataset. This will result in a private version of algorithm $\mathcal{A}$ which is formally defined by Definition 21.

Definition 21 (A Private Algorithm on Data Matrix) *Let* $\mathcal{A}^+ : \mathbb{R}^{p \times N} \to Range(\mathcal{A}^+)$ *be a mapping defined as*

$$\mathcal{A}^+(\mathrm{Y}) = \mathcal{A}(\mathrm{Y} + \mathrm{V}), \ \mathrm{V} \in \mathbb{R}^{p \times N} \tag{111}$$

where V *is a random noise matrix with* $f_{\mathrm{v}_j^i}(v)$ *being the probability density function of its* $(j, i)-$*th element* v_j^i*;* v_j^i *and* $\mathrm{v}_j^{i'}$ *are independent from each other for* $i \neq i'$*; and* $\mathcal{A} : \mathbb{R}^{p \times N} \to \mathbf{M}$ *(where* $\mathbf{M}$ *is the model space) is a given mapping representing a machine learning algorithm. The range of* $\mathcal{A}^+$ *is as*

$$Range(\mathcal{A}^+) = \left\{\mathcal{A}(\mathrm{Y} + \mathrm{V}) \mid \mathrm{Y} \in \mathbb{R}^{p \times N}, \mathrm{V} \in \mathbb{R}^{p \times N}\right\}. \tag{112}$$

We intend to protect the algorithm $\mathcal{A}^+$ from an adversary who seeks to gain an information about the data from algorithm's output by perturbing the values in a sample of the dataset. We seek to attain differential privacy for algorithm $\mathcal{A}^+$ against the perturbation in an element of Y, say $(j_0, i_0)-$th element, such that magnitude of the perturbation is upper bounded by a scalar d. Following (He et al., 2020), the $d-$adjacency and $(\epsilon, \delta)-$differential privacy definitions are provided in Definition 22 and Definition 23 respectively.

Definition 22 ($d-$Adjacency for Data Matrices) *Two matrices $\mathrm{Y}, \mathrm{Y}' \in \mathbb{R}^{p\times N}$ are $d-$adjacent if for a given $d \in \mathbb{R}_+$, there exist $i_0 \in \{1, 2, \cdots, N\}$ and $j_0 \in \{1, 2, \cdots, p\}$ such that $\forall i \in \{1, 2, \cdots, N\}, j \in \{1, 2, \cdots, p\}$,*

$$\left|\mathrm{y}_j^i - \mathrm{y}_j'^i\right| \leq \begin{cases} d, & \text{if } i = i_0, j = j_0 \\ 0, & \text{otherwise} \end{cases}$$

where y_j^i and $\mathrm{y}_j'^i$ denote the $(j, i)-$th element of Y and Y' respectively. Thus, Y and Y' differ by only one element and the magnitude of the difference is upper bounded by d.

Definition 23 ($(\epsilon, \delta)-$Differential Privacy for $\mathcal{A}^+$ (Kumar et al., 2019)) *The algorithm $\mathcal{A}^+(\mathrm{Y})$ is $(\epsilon, \delta)-$differentially private if*

$$Pr\{\mathcal{A}^+(\mathrm{Y}) \in \mathcal{O}\} \leq \exp(\epsilon) Pr\{\mathcal{A}^+\left(\mathrm{Y}'\right)) \in \mathcal{O}\} + \delta \tag{113}$$

for any measurable set $\mathcal{O} \subseteq Range(\mathcal{A}^+)$ and for $d-$adjacent matrices pair $(\mathrm{Y}, \mathrm{Y}')$.

Intuitively, Definition 23 means that changing the value of an element in the training data matrix by an amount upper bounded by d can change the distribution of output of the algorithm $\mathcal{A}^+$ only by a factor of $\exp(\epsilon)$ with probability at least $1 - \delta$. Thus, the lower value of ϵ and δ lead to a higher amount of privacy.

Result 3 (An Optimal $(\epsilon, \delta)-$Differentially Private Noise (Kumar et al., 2019)) *The probability density function of noise, that minimizes the expected noise magnitude together with satisfying the sufficient conditions for $(\epsilon, \delta)-$differential privacy for $\mathcal{A}^+$, is given as*

$$f^*_{\mathrm{v}_j^i}(v; \epsilon, \delta, d) = \begin{cases} \delta\, Dirac\delta(v), & v = 0 \\ (1-\delta)\frac{\epsilon}{2d}\exp(-\frac{\epsilon}{d}|v|), & v \in \mathbb{R} \setminus \{0\} \end{cases} \tag{114}$$

where $Dirac\delta(v)$ is Dirac delta function satisfying $\int_{-\infty}^{\infty} Dirac\delta(v)\ \mathrm{d}v = 1$.

Proof: The proof follows from (Kumar et al., 2019). ■

Remark 24 (Generating Random Samples from $f^*_{\mathrm{v}_j^i}$) *The method of "inverse transform sampling" can be used to generate random samples from cumulative distribution function. The cumulative distribution function of $f^*_{\mathrm{v}_j^i}$ is given as*

$$F_{\mathrm{v}_j^i}(v; \epsilon, \delta, d) = \begin{cases} \frac{1-\delta}{2}\exp(\frac{\epsilon}{d}v), & v < 0 \\ \frac{1+\delta}{2}, & v = 0 \\ 1 - \frac{1-\delta}{2}\exp(-\frac{\epsilon}{d}v), & v > 0 \end{cases} \tag{115}$$

The inverse cumulative distribution function is given as

$$F^{-1}_{\mathrm{v}^i_j}(t^i_j;\epsilon,\delta,d)=\begin{cases}\frac{d}{\epsilon}\log(\frac{2t^i_j}{1-\delta}), & t^i_j<\frac{1-\delta}{2}\\ 0, & t^i_j\in[\frac{1-\delta}{2},\frac{1+\delta}{2}]\\ -\frac{d}{\epsilon}\log(\frac{2(1-t^i_j)}{1-\delta}), & t^i_j>\frac{1+\delta}{2}\end{cases},\ t^i_j\in(0,1). \tag{116}$$

Thus, via generating random samples from the uniform distribution on $(0,1)$ *and using (116), the noise additive mechanism can be implemented.*

Algorithm 5 Differentially private approximation of data samples

Require: Data set $\mathbf{Y}=\{y^i\in\mathbb{R}^p\mid i\in\{1,\cdots,N\}\}$; differential privacy parameters: $d\in\mathbb{R}_+$, $\epsilon\in\mathbb{R}_+$, $\delta\in(0,1)$.

1: A differentially private approximation of data samples is provided as

$$y^{+i}_j=y^i_j+F^{-1}_{\mathrm{v}^i_j}(t^i_j;\epsilon,\delta,d),\ t^i_j\in(0,1) \tag{117}$$

where $F^{-1}_{\mathrm{v}^i_j}$ is given by (116) and y^{+i}_j is $j-$th element of $y^{+i}\in\mathbb{R}^p$.

2: **return** $\mathbf{Y}^+=\{y^{+i}\in\mathbb{R}^p\mid i\in\{1,\cdots,N\}\}$.

For a given value of (ϵ,δ,d), Algorithm 5 is formally stated for a differentially private approximation of a data samples.

Since the noise adding mechanism (i.e. Result 3) is independent of the choice of algorithm operating on training data matrix, therefore any algorithm operating on noise added data samples will remain $(\epsilon,\delta)-$differentially private. That is, differential privacy remains invariant to any post-processing of noise added data samples. This allows us to build a differentially private classifier as stated in Algorithm 6.

Algorithm 6 Variational learning of a differentially private classifier

Require: Differentially private approximated dataset: $\mathbf{Y}^+=\{\mathbf{Y}^+_c\mid c\in\{1,\cdots,C\}\}$; the subspace dimension $n\in\{1,\cdots,p\}$; ratio $M/N\in(0,1]$; the number of layers $L\in\mathbb{Z}_+$.

1: Run Algorithm 4 on $\mathbf{Y}^+$ to build a classifier characterized by parameters sets $\{\mathcal{P}^+_c\}^C_{c=1}$.

2: **return** $\{\mathcal{P}^+_c\}^C_{c=1}$.

4.2 Differentially Private Semi-Supervised Transfer Learning

We consider a scenario of knowledge transfer from a dataset consisting of labeled samples from a domain (referred to as source domain) to another dataset consisting of mostly unlabelled samples and only a few labelled samples from another domain (referred to as target domain) such that both source and target datasets have been sampled from the same set of classes but in their respective domains. The aim is to transfer the knowledge extracted by a classifier trained using source dataset to the classifier of target domain such that privacy of source dataset is preserved. Let $\{\mathbf{Y}^{sr}_c\}^C_{c=1}$ be the labelled source dataset where $\mathbf{Y}^{sr}_c=\{y^{i,c}_{sr}\in\mathbb{R}^{p_{sr}}\mid i\in\{1,\cdots,N^{sr}_c\}\}$ represents $c-$th labelled samples. The target dataset consist of a few labelled samples $\{\mathbf{Y}^{tg}_c\}^C_{c=1}$ (with $\mathbf{Y}^{tg}_c=\{y^{i,c}_{tg}\in\mathbb{R}^{p_{tg}}\mid i\in\{1,\cdots,N^{tg}_c\}\}$)

and another set of unlabelled samples $\mathbf{Y}_*^{tg} = \{y_{tg}^{i,*} \in \mathbb{R}^{p_{tg}} \mid i \in \{1, \cdots, N_*^{tg}\}\}$. A generalized setting is considered where source and target data dimensions could be different, i.e., $p_{sr} \neq p_{tg}$. Our approach to semi-supervised transfer learning consists of following steps:

Differentially private source domain classifier: For a given differential privacy parameters: d, ϵ, δ; Algorithm 5 is applied on $\mathbf{Y}_c^{sr}$ to obtain the differentially private approximated data samples, $\mathbf{Y}_c^{+sr} = \{y_{sr}^{+i,c} \in \mathbb{R}^{p_{sr}} \mid i \in \{1, \cdots, N_c^{sr}\}\}$, for all $c \in \{1, \cdots, C\}$. Algorithm 6 is applied on $\{\mathbf{Y}_c^{+sr}\}_{c=1}^C$ to build a differentially private source domain classifier characterized by parameters sets $\{\mathcal{P}_c^{+sr}\}_{c=1}^C$.

Differentially private source domain latent subspace transformation-matrix: For a lower-dimensional representation of both source and target samples, a subspace dimension, $n_{st} \in \{1, 2, \cdots, \min(p_{sr}, p_{tg})\}$, is chosen. Let $V^{+sr} \in \mathbb{R}^{n_{st} \times p_{sr}}$ be the transformation-matrix with its $i-$th row equal to transpose of eigenvector corresponding to $i-$th largest eigenvalue of sample covariance matrix computed on $\{\mathbf{Y}_c^{+sr}\}_{c=1}^C$.

Differentially private class-centers in latent subspace of source domain: Let $\bar{m}_c^{+sr} \in \mathbb{R}^{n_{st}}$ be the vector defined as

$$\bar{m}_c^{+sr} = \text{median}\left(\left\{V^{+sr} y_{sr}^{+i,c} \mid i \in \{1, \cdots, N_c^{sr}\}\right\}\right). \tag{118}$$

That is, $\bar{m}_c^{+sr}$ is the center of the $c-$th labelled noise added source data samples in latent subspace.

Target domain latent subspace transformation-matrix: Let $V^{tg} \in \mathbb{R}^{n_{st} \times p_{tg}}$ be the transformation-matrix with its $i-$th row equal to transpose of eigenvector corresponding to $i-$th largest eigenvalue of sample covariance matrix computed on $\{\mathbf{Y}_c^{tg}\}_{c=1}^C \cup \mathbf{Y}_*^{tg}$. For the case of homogeneous source and target domains (i.e. $p_{sr} = p_{tg}$), one could choose V^{tg} as equal to V^{+sr}.

Class-centers in latent subspace of target domain: For a given classifier function, $\hat{c} : \mathbb{R}^{tg} \to \{1, 2, \cdots, C\}$, let $\bar{m}_c^{tg} \in \mathbb{R}^{n_{st}}$ be a vector defined as

$$\bar{m}_c^{tg} = \text{median}\left(\left\{V^{tg} y_{tg}^{i,c} \mid i \in \{1, \cdots, N_c^{tg}\}\right\} \cup \left\{V^{tg} y_{tg}^{i,*} \mid \hat{c}(y_{tg}^{i,*}) = c, i \in \{1, \cdots, N_*^{tg}\}\right\}\right) \tag{119}$$

That is, $\bar{m}_c^{tg}$ is the center of the $c-$th labelled target data samples in latent subspace.

A transformation for representing target data samples in source-data-space: To represent a target sample in source-data-space, we follow *subspace alignment* approach where a target sample is first aligned to source data in subspace followed by a linear transformation to source-data-space. An unlabelled target sample $y_{tg}^{i,*}$ can be mapped to source-data-space via a linear transformation: $f_c^{tg \to sr}(y_{tg}^{i,*}) = (V^{+sr})^T (V^{tg} y_{tg}^{i,*} + \Delta\bar{m}_c)$, where $\Delta\bar{m}_c \in \mathbb{R}^{n_{st}}$ is chosen such that estimated center of $c-$th labelled target data samples is mapped to the estimated center of $c-$th labelled source data samples. Since V^{+sr} is orthonormal (i.e. $V^{+sr}(V^{+sr})^T = I$), $\Delta\bar{m}_c$ is given as $\Delta\bar{m}_c = \bar{m}_c^{+sr} - \bar{m}_c^{tg}$. Thus, for a given set, $\theta = \{\bar{m}_c^{tg}, V^{tg}, \bar{m}_c^{+sr}, V^{+sr}\}$, a transformation, $f_c^{tg \to sr} : \mathbb{R}^{p_{tg}} \to \mathbb{R}^{p_{sr}}$, is defined as

$$f_c^{tg \to sr}(y_{tg}^{i,*}; \theta = \{\bar{m}_c^{tg}, V^{tg}, \bar{m}_c^{+sr}, V^{+sr}\}) \stackrel{\text{def}}{=} (V^{+sr})^T \left(V^{tg} y_{tg}^{i,*} - \bar{m}_c^{tg} + \bar{m}_c^{+sr}\right). \tag{120}$$

That is, $f_c^{tg \to sr}$ maps a target sample close to center of $c-$th labelled target data samples to a point in source-data-space that is close to center of $c-$th labelled source data samples.

A combination of source and target domain classifiers: The label associated to $y_{tg}^{i,*}$ is predicted via combining both source and target domain classifiers as

$$\hat{c}(y_{tg}^{i,*};\{\mathcal{P}_c^{tg}\}_{c=1}^{C},\{\mathcal{P}_c^{+sr}\}_{c=1}^{C},\theta) = \arg \min_{c\,\in\,\{1,2,\cdots,C\}} \left\{ \min\left(\left\| y_{tg}^{i,*} - \mathcal{WD}(y_{tg}^{i,*};\mathcal{P}_c^{tg}) \right\|^2, \left\| f_c^{tg\rightarrow sr}(y_{tg}^{i,*};\theta) - \mathcal{WD}(f_c^{tg\rightarrow sr}(y_{tg}^{i,*};\theta);\mathcal{P}_c^{+sr}) \right\|^2 \right) \right\}. \quad (121)$$

That is, $y_{tg}^{i,*}$ is assigned the $c-$th class label, if

- either the wide-deep autoencoder associated to $c-$th class of target data space (which is characterized by set of parameters $\mathcal{P}_c^{tg}$) could best reconstruct $y_{tg}^{i,*}$, or
- the differentially private wide-deep autoencoder associated to $c-$th class of source data space (which is characterized by set of parameters $\mathcal{P}_c^{+sr}$) could best reconstruct $f_c^{tg\rightarrow sr}(y_{tg}^{i,*};\theta)$ (which is corresponding to $y_{tg}^{i,*}$ the transformed point in source data space close to the center of $c-$th labelled source data samples).

Building of target domain classifier: Our idea is to iteratively use (121) for predicting the labels of unlabelled target data samples followed by Algorithm 4 for building target domain classifier. The $k-$th iteration consists of following updates:

$$\{\mathcal{P}_c^{tg}|_k\}_{c=1}^{C} = \text{Algorithm 4}\left(\left\{\mathbf{Y}_c^{tg} \cup \mathbf{Y}_{*,c}^{tg}|_{k-1}\right\}_{c=1}^{C}, n|_k, (M/N)|_k, L\right) \quad (122)$$

$$\theta|_k = \{\bar{m}_c^{tg}|_{k-1}, V^{tg}, \bar{m}_c^{+sr}, V^{+sr}\} \quad (123)$$

$$\mathbf{Y}_{*,c}^{tg}|_k = \left\{ y_{tg}^{i,*} \mid \hat{c}(y_{tg}^{i,*};\{\mathcal{P}_c^{tg}|_k\}_{c=1}^{C},\{\mathcal{P}_c^{+sr}\}_{c=1}^{C},\theta|_k) = c,\ i \in \{1,\cdots,N_*^{tg}\} \right\} \quad (124)$$

$$\bar{m}_c^{tg}|_k = \text{median}\left(\left\{ V^{tg} y_{tg}^{i,c} \mid i \in \{1,\cdots,N_c^{tg}\} \right\} \cup \left\{ V^{tg} y_{tg}^{i,*} \mid \hat{c}(y_{tg}^{i,*};\{\mathcal{P}_c^{tg}|_k\}_{c=1}^{C},\{\mathcal{P}_c^{+sr}\}_{c=1}^{C},\theta|_k) = c,\ i \in \{1,\cdots,N_*^{tg}\} \right\} \right) \quad (125)$$

where $n|_k$ and $M/N|_k$ are parameters of Algorithm 4 in $k-$th iteration such that $\{n|_1, n|_2, \cdots\}$ and $\{(M/N)|_1, (M/N)|_2, \cdots\}$ are monotonically non-decreasing sequences. The reason for subspace dimension n and M/N ratio to follow a monotonically non-decreasing curve during the iterations is following:

- We intend to use higher-level data features during initial iterations for updating the predicted-labels of unlabeled target data samples and as the number of iterations increases more and more lower-level data features are intended to be included in the process of updating the predicted-labels. Since the lower values of n and M/N lead to modeling of higher-level data features and higher values lead to modeling of lower-level data features (as illustrated in Fig. 5 and Fig. 6), n and M/N values are chosen as to form a monotonically non-decreasing sequence.

The method can be systematically implemented using Algorithm 7. The proposed approach (stated in Fig. 1 and implemented by Algorithm 7) is further illustrated in Fig. 7.

Algorithm 7 Differentially private semi-supervised transfer learning

Require: A differentially private source domain classifier characterized by parameters sets $\{\mathcal{P}_c^{+sr}\}_{c=1}^C$; a differentially private source domain latent subspace transformation-matrix: $V^{+sr} \in \mathbb{R}^{n_{st} \times p_{sr}}$; the differentially private class-centers in latent subspace of source domain: $\{\bar{m}_c^{+sr} \in \mathbb{R}^{n_{st}}\}_{c=1}^C$; the set of a few labelled target samples: $\{\mathbf{Y}_c^{tg}\}_{c=1}^C$ (where $\mathbf{Y}_c^{tg} = \{y_{tg}^{i,c} \in \mathbb{R}^{p_{tg}} \mid i \in \{1, \cdots, N_c^{tg}\}\}$ is the set of $c-$th labelled target samples); the set of unlabelled target samples: $\mathbf{Y}_*^{tg} = \{y_{tg}^{i,*} \in \mathbb{R}^{p_{tg}} \mid i \in \{1, \cdots, N_*^{tg}\}\}$; a target domain initial classifier built using labelled target data samples and characterized by parameters sets: $\{\mathcal{P}_c^{tg}|_0\}_{c=1}^C$; the number of iterations: N_{it}; a monotonically non-decreasing subspace dimension sequence: $\{n|_1, n|_2, \cdots, n|_{N_{it}}\}$; a monotonically non-decreasing M/N ratio sequence: $\{(M/N)|_1, (M/N)|_2, \cdots, (M/N)|_{N_{it}}\}$; the number of layers in the deep model associated to target domain classifier: L.

1: Use initial target classifier (i.e. $\{\mathcal{P}_c^{tg}|_0\}_{c=1}^C$) to predict class-labels of unlabelled target data samples and partition the total unlabelled samples into C different subsets such that each subset consists of samples with same predicted class label. That is, $\forall c \in \{1, 2, \cdots, C\}$, build a subset:

$$\mathbf{Y}_{*,c}^{tg}|_0 = \left\{ y_{tg}^{i,*} \mid \mathcal{C}(y_{tg}^{i,*}; \{\mathcal{P}_c^{tg}|_0\}_{c=1}^C) = c,\ i \in \{1, \cdots, N_*^{tg}\} \right\} \tag{126}$$

where $\mathcal{C}(\cdot)$ is defined by (110).

2: Define the target domain latent subspace transformation-matrix: $V^{tg} \in \mathbb{R}^{n_{st} \times p_{tg}}$ such that $i-$th row of V^{tg} is equal to transpose of eigenvector corresponding to $i-$th largest eigenvalue of sample covariance matrix computed on $\{\mathbf{Y}_c^{tg}\}_{c=1}^C \cup \mathbf{Y}_*^{tg}$. In the special case of $p_{sr} = p_{tg}$, V^{tg} could be defined as $V^{tg} = V^{+sr}$.

3: Initialize the center of the $c-$th labelled target data samples in latent subspace as

$$\begin{aligned} \bar{m}_c^{tg}|_0 = \text{median} \Big(& \left\{ V^{tg} y_{tg}^{i,c} \mid i \in \{1, \cdots, N_c^{tg}\} \right\} \\ & \cup \left\{ V^{tg} y_{tg}^{i,*} \mid \mathcal{C}(y_{tg}^{i,*}; \{\mathcal{P}_c^{tg}|_0\}_{c=1}^C) = c,\ i \in \{1, \cdots, N_*^{tg}\} \right\} \Big) \end{aligned} \tag{127}$$

where $\mathcal{C}(\cdot)$ is defined by (110).

4: **for** $k = 1$ to N_{it} **do**

5: Set $L|_k = L$, if $k = N_{it}$, otherwise set $L|_k = 1$.

6: Update the target domain classifier as

$$\{\mathcal{P}_c^{tg}|_k\}_{c=1}^C = \text{Algorithm 4} \left(\left\{ \mathbf{Y}_c^{tg} \cup \mathbf{Y}_{*,c}^{tg}|_{k-1} \right\}_{c=1}^C, n|_k, (M/N)|_k, L|_k \right). \tag{128}$$

7: Estimate the class-labels for unlabelled target data samples using following:

$$\theta|_k = \{\bar{m}_c^{tg}|_{k-1}, V^{tg}, \bar{m}_c^{+sr}, V^{+sr}\} \tag{129}$$

$$\mathbf{Y}_{*,c}^{tg}|_k = \left\{ y_{tg}^{i,*} \mid \hat{c}(y_{tg}^{i,*}; \{\mathcal{P}_c^{tg}|_k\}_{c=1}^C, \{\mathcal{P}_c^{+sr}\}_{c=1}^C, \theta|_k) = c,\ i \in \{1, \cdots, N_*^{tg}\} \right\} \tag{130}$$

where $\hat{c}(\cdot)$ is defined by (121).

8: Update the center of the $c-$th labelled target data samples in latent subspace as

$$\begin{aligned} \bar{m}_c^{tg}|_k = \text{median} \Big(& \left\{ V^{tg} y_{tg}^{i,c} \mid i \in \{1, \cdots, N_c^{tg}\} \right\} \\ & \cup \left\{ V^{tg} y_{tg}^{i,*} \mid \hat{c}(y_{tg}^{i,*}; \{\mathcal{P}_c^{tg}|_k\}_{c=1}^C, \{\mathcal{P}_c^{+sr}\}_{c=1}^C, \theta|_k) = c,\ i \in \{1, \cdots, N_*^{tg}\} \right\} \Big) \end{aligned} \tag{131}$$

where $\hat{c}(\cdot)$ is defined by (121).

9: **end for**

10: **return** $\{\mathcal{P}_c^{tg}\}_{c=1}^C$, where $\mathcal{P}_c^{tg} = \mathcal{P}_c^{tg}|_{N_{it}}$.

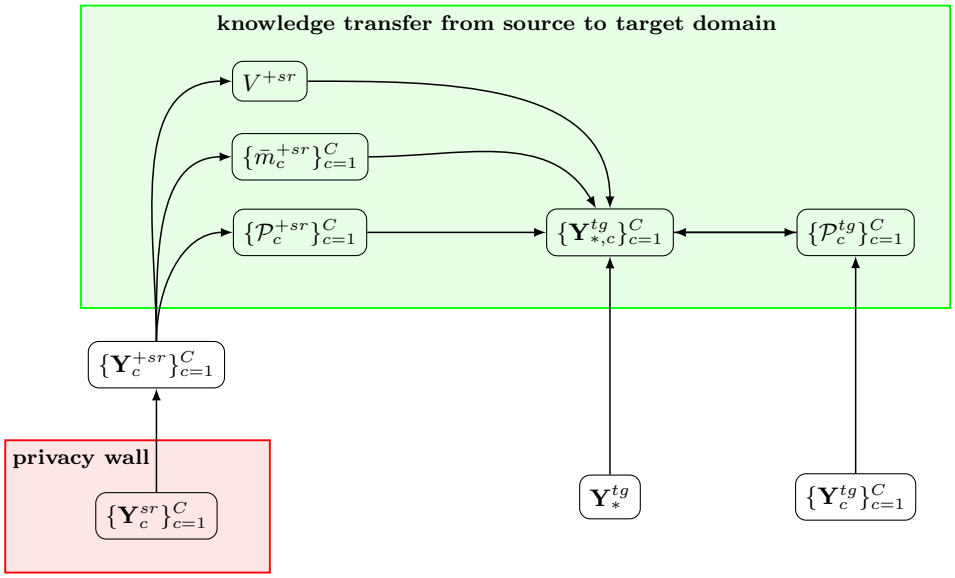


Figure 7: Algorithm 7 implements the proposed privacy-preserving semi-supervised transfer learning approach (stated in Fig. 1).

5. Experiments

Algorithms 2-7 were implemented using MATLAB R2017b. The experiments have been made on a MacBook Pro machine with a 2.2 GHz Intel Core i7 processor and 16 GB of memory.

We study experimentally the differential privacy (Definition 23) of the source domain training data such that for all $1-$adjacent training data matrices, the absolute value of privacy-loss incurred by observing the output of any computation algorithm will be bounded by ϵ with probability at least $1-\delta$. That is, d is taken as equal to 1 for defining adjacent matrices in Definition 23.

Algorithm 7 requires several inputs which are obtained in an experimental setting described as below:

1. Algorithm 5, for a given d and (ϵ, δ), is applied to obtain a differentially private approximation of source dataset.

2. Differentially private source domain classifier is built using Algorithm 6 taking subspace dimension as equal to $\min(20, p_{sr})$ (where p_{sr} is the dimension of source data samples), ratio M/N as equal to 0.5, and number of layers as equal to 5.

3. Differentially private source domain latent subspace transformation-matrix is computed with $n_{st} = \min(\lceil p_{sr}/2 \rceil, p_{tg})$, where p_{tg} is the dimension of target data samples.

4. Differentially private class-centers in latent subspace of source domain are computed using (118).

5. Initial target domain classifier is built using Algorithm 4 on labelled target samples taking subspace dimension as equal to $\min(20, \min_{1\leq c\leq C}\{N_c^{tg}\}-1)$ (where N_c^{tg} is the number of $c-th$ class labelled target samples), ratio M/N as equal to 1, and number of layers as equal to 1.

6. The number of iterations is set equal to 4; the monotonically non-decreasing subspace dimension sequence is chosen as $\{4, 6, 8, 10\}$; the monotonically non-decreasing M/N ratio sequence is chosen as $\{\frac{1}{10}, \frac{1}{8}, \frac{1}{6}, \frac{1}{4}\}$.

7. The number of layers in the deep model associated to target domain classifier is set as equal to 5.

5.1 Demonstrative Examples Using MNIST and USPS Datasets

Our first experiment is on the widely used MNIST digits dataset containing 28×28 sized images divided into training set of 60000 images and testing set of 10000 images. The images' pixel values were divided by 255 to normalize the values in the range from 0 to 1. The 28×28 normalized values of each image were flattened to an equivalent $784-$dimensional data vector. The transfer learning experiment was carried in the same setting as in (Papernot et al., 2017) where 60000 training samples constituted the source dataset; a set of 9000 test samples constituted target dataset, and the performance was evaluated on the remaining 1000 test samples. Out of 9000 target samples, only 10 samples per class were labelled and rest 8900 target samples remained as unlabelled. The experiments were carried out at different values of privacy-loss bound $\epsilon \in \{0.1, 0.5, 1, 1.5, 1.9, 2, 3, 5, 8\}$ while keeping failure probability fixed at $\delta = 1e{-}5$.

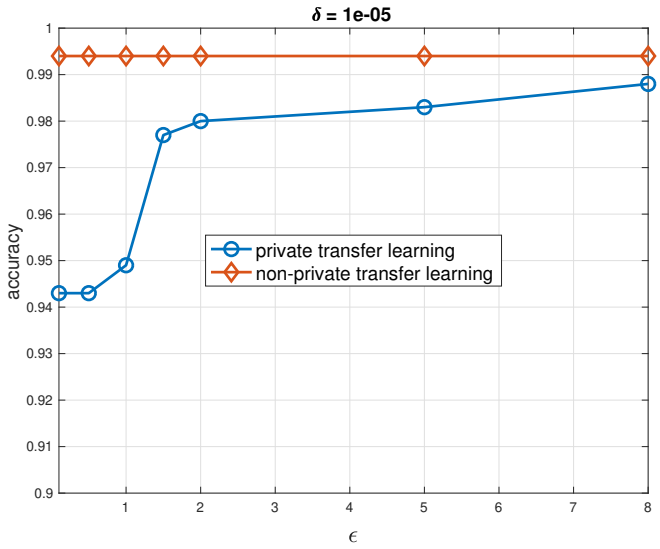


Figure 8: The effect of privacy-loss bound ϵ on accuracy for MNIST dataset.

Fig. 8 shows the evolution of testing accuracy as a function of ϵ for a fixed δ. Fig. 8 also displays the accuracy for the non-private case that corresponds to $\epsilon = \infty$ i.e. no noise

Table 1: Privacy and utility results on MNIST dataset. The second column reports the privacy-loss bound ϵ and failure probability δ of $(\epsilon, \delta)-$differential privacy guarantee.

Method	(ϵ, δ)	**classification accuracy**
proposed	$(8, 1e{-}5)$	**98.80%**
(Papernot et al., 2017)	$(8.03, 1e{-}5)$	98.10%
(Abadi et al., 2016)	$(8, 1e{-}5)$	97.00%
proposed	$(2, 1e{-}5)$	**98.00%**
(Papernot et al., 2017)	$(2.04, 1e{-}5)$	**98.00%**
(Abadi et al., 2016)	$(2, 1e{-}5)$	95.00%
proposed	non-private	**99.40%**
(Papernot et al., 2017)	non-private	99.18%

is added. A lower privacy-loss bound bound implies a larger amount of noise being added

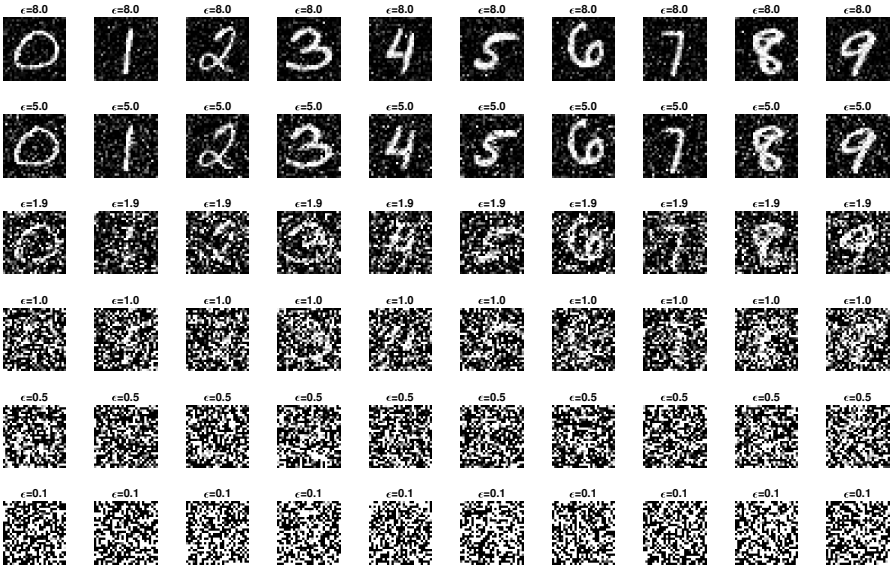


Figure 9: An example of noise added to a randomly selected sample from each class of MNIST source data for different values of ϵ. For each sample, the corresponding ϵ value has been stated at the top of the image.

to data samples as illustrated through an example in Fig. 9. For a comparison, Table 1 reports the values of $(\epsilon, \delta)-$differential privacy guarantees and corresponding classification

accuracies. The proposed method is able to learn 98.80% accurate model together with providing a strict bound of $\epsilon = 8$, which is a better result than that of previous studies.

We considered a problem of heterogeneous transfer learning from MNIST to USPS dataset. The USPS is another dataset that has 7291 training and 2007 test images of digits where each image has 16×16 (=256) grayscale pixels. The aim of this experiment was to study how privacy-preservation affect transferring knowledge from a higher resolution and more varied MNIST dataset to USPS dataset. The MNIST→USPS semi-supervised transfer learning problem was previously studied in (Belhaj et al., 2018). For a comparison, MNIST→USPS problem was considered in the same experimental setting as in (Belhaj et al., 2018) where only 100 target samples were labelled and remaining 7191 samples remained as unlabelled. The experiments were carried out at different values of privacy-loss bound $\epsilon \in \{0.1, 0.5, 1, 2, 5\}$ while keeping failure probability fixed at $\delta = 1e{-}5$. The performance was evaluated on target domain testing dataset in-terms of classification accuracy.

Table 2 reports the results of 10 independent MNIST→USPS experiments. As observed in Table 2, the proposed method, despite being privacy-preserving and having not required an access to source data samples, performs comparable to the Deep Variational Transfer (a variational autoencoder that transfers knowledge across domains using a shared latent Gaussian mixture model) proposed in (Belhaj et al., 2018). Further, the proposed method's consistent performance over a wide range of privacy-loss bound ϵ verifies the robustness of the target model towards the perturbations in source data caused by the privacy requirements demanded by source data owner.

Table 2: Results of 10 independent MNIST→USPS transfer learning experiments expressed in average accuracy.

method	**average classification accuracy (in %)**
$(0.1, 1e{-}5)-$differentially private proposed	93.01
$(0.5, 1e{-}5)-$differentially private proposed	93.01
$(1, 1e{-}5)-$differentially private proposed	93.02
$(2, 1e{-}5)-$differentially private proposed	93.03
$(5, 1e{-}5)-$differentially private proposed	**93.22**
Deep Variational Transfer (Belhaj et al., 2018)	92.03

5.2 Comparisons Using Office and Caltech256 Datasets

"Office+Caltech256" dataset has 10 common categories of both Office and Caltech256 datasets. This dataset has been widely used (Hoffman et al., 2013; Herath et al., 2017; Karbalayghareh et al., 2018; Hoffman et al., 2014) for evaluating multi-class accuracy performance in a standard domain adaptation setting with a small number of labelled target samples. The dataset has fours domains: *amazon*, *webcam*, *dslr*, and *caltech256*. We follow the experimental setup of (Hoffman et al., 2013; Herath et al., 2017; Karbalayghareh et al., 2018; Hoffman et al., 2014):

1. the number of training samples per class in the source domain is 20 for *amazon* and is 8 for other three domains;

2. the number of labelled samples per class in the target domain is 3 for all the four domains;

3. 20 random train/test splits are created and the performance on target domain test samples is averaged over 20 experiments.

Following (Herath et al., 2017), the deep-net VGG-FC6 features are extracted from the images and the proposed method is compared with

1. SVM-t: A base-line is created using a linear SVM classifier trained using only the labelled target samples without transfer learning.

2. ILS (1-NN) (Herath et al., 2017): This method learns an Invariant Latent Space (ILS) to reduce the discrepancy between domains and uses Riemannian optimization techniques to match statistical properties between samples projected into the latent space from different domains.

3. CDLS (Tsai et al., 2016): The Cross-Domain Landmark Selection (CDLS) method derives a domain-invariant feature subspace for heterogeneous domain adaptation.

4. MMDT (Hoffman et al., 2014): The Maximum Margin Domain Transform (MMDT) method adapts max-margin classifiers in a multi-class manner by learning a shared component of the domain shift as captured by the feature transformation.

5. HFA (Li et al., 2014): The Heterogeneous Feature Augmentation (HFA) method learns common latent subspace and a classifier under max-margin framework.

6. OBTL (Karbalayghareh et al., 2018): The Optimal Bayesian Transfer Learning (OBTL) method employs Bayesian framework to transfer learning through modeling of a joint prior probability density function for feature-label distributions of the source and target domains.

The "Office+Caltech256" dataset has been previously studied in (Hoffman et al., 2013; Herath et al., 2017; Karbalayghareh et al., 2018; Hoffman et al., 2014) using SURF features. Therefore, the state-of-art results on this dataset using SURF features are additionally considered for a comparison. There are in total 4 domains associated to "Office+Caltech256" dataset. Taking a domain as source and other domain as target, 12 different transfer learning experiments can be performed on these 4 domains. Table 4, Table 5, Table 6, Table 7, Table 8, Table 9, Table 10, Table 11, Table 12, Table 13, Table 14, and Table 15 in Appendix G report the results of experiments. The first two best performances have been marked in Table 4, Table 5, Table 6, Table 7, Table 8, Table 9, Table 10, Table 11, Table 12, Table 13, Table 14, and Table 15.

Finally, Table 3 lists the average performance achieved by different methods during the experiments. The methods have been also ranked in Table 3 based on their performances. Following inferences are drawn from the experimental results:

Table 3: Accuracy (in %, averaged over the results of 12 independent transfer learning experiments) obtained by different methods on "Office+Caltech256" dataset.

method	**feature type**	**accuracy (%)**	**rank**
non-private proposed	VGG-FC6	90.4	1
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	89.5	2
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	89.4	3
non-private ILS (1-NN)	VGG-FC6	88.5	4
non-private CDLS	VGG-FC6	85.9	5
SVM-t (without knowledge transfer)	VGG-FC6	84.3	6
non-private HFA	VGG-FC6	83.7	7
non-private MMDT	VGG-FC6	80.8	8
non-private OBTL	SURF	58.9	9
non-private ILS (1-NN)	SURF	55.6	10
non-private CDLS	SURF	53.5	11
non-private MMDT	SURF	52.5	12
non-private HFA	SURF	48.1	13

1. Out of 12 experiments, the proposed approach achieved best performance in 9 experiments.

2. As observed in Table 3, the non-private version of the proposed method performs better than the state-of-art non-private methods' results.

3. It is observed in Table 3 that as the privacy-loss bound is decreased from $\epsilon = \infty$ (i.e. non-private case) to $\epsilon = 0.1$, the accuracy of the suggested method decreases only slightly from 90.4% to 89.4%. Thus, the proposed method is robust towards the noise added to source domain training data for preserving privacy.

4. The most remarkable result observed in Table 3 is that the proposed method, despite ensuring privacy-loss bound to be as low as 0.1, performs better than even the non-private state-of-art methods' results on this dataset.

6. Concluding Remarks

This study has outlined a novel approach to differentially private semi-supervised transfer learning that exploits the variational deep mappings and an optimal noise adding mechanism for achieving a robustness of target model towards the perturbations in source data caused by the privacy requirements demanded by source data owner. A variational-deep-mapping based approach was introduced that sufficiently addresses all of the five requirements identified regarding the privacy-preserving semi-supervised transfer learning problem. Numerous experiments were carried out using MNIST, USPS, Office, and Caltech256 datasets to verify the competitive performance of the proposed method. The experimental studies further

verify that our approach is capable of achieving a low privacy-loss bound without letting the accuracy be much affected.

The future work is on information theoretic study of transfer learning where the transferability of knowledge from source to target domain will be quantified in-terms of mutual information between the source data vector and target data vector. This will be done via maximizing a lower bound on mutual information over a stochastic model describing the relationship between source and target data.

Acknowledgments

The research reported in this paper has been partly supported by EU Horizon 2020 Grant 826278 "Securing Medical Data in Smart Patient-Centric Healthcare Systems" (Serums), Austrian Research Promotion Agency (FFG) Grant 873979 "Privacy Preserving Machine Learning for Industrial Applications" (PRIMAL), and the Austrian Ministry for Transport, Innovation and Technology, the Federal Ministry for Digital and Economic Affairs, and the Province of Upper Austria in the frame of the COMET center SCCH.

Appendix A. A Probability Measure on $\mathbb{F}(\mathcal{X})$

Given a sequence of samples $(x^i)_{i=1}^{\mathbb{N}}$, define $\mathrm{S}(N) := (x^1, \cdots, x^N)$ i.e. $\mathrm{S}(N+1) = \mathrm{S}(N) \wedge (x^{N+1})$, $N \in \mathbb{N}$. For each $N \in \mathbb{N}$, let $\mathbb{P}_{\zeta_{\mathrm{S}(N)}}$ be a probability measure induced by a membership function $\zeta_{\mathrm{S}(N)} \in \Theta$. As per assumption (11), the measures, $(\mathbb{P}_{\zeta_{\mathrm{S}(N)}})_{N=1}^{\mathbb{N}}$, are consistent in the sense that $\mathbb{P}_{\zeta_{\mathrm{S}(N+1)}}(A \times \mathbb{R}) = \mathbb{P}_{\zeta_{\mathrm{S}(N)}}(A)$, for any $A \in \mathcal{B}(\mathbb{R}^N)$ and $N \in \mathbb{N}$. Then Kolmogorov extension theorem guarantees the existence of a probability measure $\mathbf{p}$ on $\mathbb{R}^{\mathbb{N}}$ satisfying $\mathbf{p}(A \times \mathbb{R}^{\mathbb{N}}) = \mathbb{P}_{\zeta_{\mathrm{S}(N)}}(A)$, for any $A \in \mathcal{B}(\mathbb{R}^N)$.

It can be observed that $\mathcal{T}$ forms an algebra of subsets of $\mathbb{F}(\mathcal{X})$. To see this, consider $\mathrm{x} \in \mathcal{S}$, $A \in \mathcal{B}(\mathbb{R}^{|\mathrm{x}|})$, $\mathrm{a} \in \mathcal{S}$, and $B \in \mathcal{B}(\mathbb{R}^{|\mathrm{a}|})$. Now, we have

$$\mathbb{F}(\mathcal{X}) = \mathcal{T}_{\mathrm{x}}(\mathbb{R}^{|\mathrm{x}|}) \in \mathcal{T} \tag{132}$$

$$(\mathcal{T}_{\mathrm{x}}(A))^{\complement} = \mathcal{T}_{\mathrm{x}}(\mathbb{R}^{|\mathrm{x}|} \setminus A) \in \mathcal{T} \tag{133}$$

$$\mathcal{T}_{\mathrm{x}}(A) \cap \mathcal{T}_{\mathrm{a}}(B) = \mathcal{T}_{\mathrm{x}\wedge\mathrm{a}}(A \times B) \in \mathcal{T}. \tag{134}$$

Thus, $\mathcal{T}$ is an algebra of subsets of $\mathbb{F}(\mathcal{X})$. Let $\tilde{\mathbf{p}} : \mathcal{T} \to [0,1]$ be a function defined as

$$\tilde{\mathbf{p}}(\mathcal{T}_{\mathrm{x}}(A)) := \mathbb{P}_{\zeta_{\mathrm{x}}}(A). \tag{135}$$

As $\zeta_{\mathrm{x}} \in \Theta$, (11) holds, and therefore (135) uniquely defines $\tilde{\mathbf{p}}$ over $\mathcal{T}$ without depending on the special representation of cylinder set $\mathcal{T}_{\mathrm{x}}(A)$. It follows from (135) that $\tilde{\mathbf{p}}$ is a $\sigma-$finite *pre-measure* (i.e. $\sigma-$additive) on algebra $\mathcal{T}$ of cylinder sets. Thus, according to *Carathéodory's extension theorem*, $\tilde{\mathbf{p}}$ can be extended in a unique way to a measure $\mathbf{p} : \sigma(\mathcal{T}) \to \mathbb{R}_{\geq 0}$ on the $\sigma-$algebra generated by $\mathcal{T}$. Hence, $(\mathbb{F}(\mathcal{X}), \sigma(\mathcal{T}), \mathbf{p})$ is measure space and a probabilistic measure $\mathbf{p}$, for a set $\mathcal{T}_{\mathrm{x}}(A) \in \mathcal{T}$, is defined as in (13).

Appendix B. Expectations Over $\mathbb{F}(\mathcal{X})$

Given $\mathrm{x} \in \mathcal{S}$, define a projection from $\mathbb{F}(\mathcal{X})$ to $\mathbb{R}^{|\mathrm{x}|}$ as

$$\Pi_{\mathrm{x}}(f) := f(\mathrm{x}) \tag{136}$$

where $f \in \mathbb{F}(\mathcal{X})$. For any $A \in \mathcal{B}(\mathbb{R}^{|\mathrm{x}|})$,

$$\Pi_{\mathrm{x}}^{-1}(A) = \mathcal{T}_{\mathrm{x}}(A). \tag{137}$$

It follows from (13) and (137) that

$$\mathbb{P}_{\zeta_{\mathrm{x}}} = \mathbf{p} \circ \Pi_{\mathrm{x}}^{-1}. \tag{138}$$

For a $\mathcal{B}(\mathbb{R}^{|\mathrm{x}|}) - \mathcal{B}(\mathbb{R})$ measurable mapping $g : \mathbb{R}^{|\mathrm{x}|} \to \mathbb{R}$, the average value of $g(f(\mathrm{x}))$ over all real valued functions $f \in \mathbb{F}(\mathcal{X})$ can be calculated via taking expectation of $g(\Pi_{\mathrm{x}}(f))$ w.r.t. probabilistic measure $\mathbf{p}$. That is,

$$\begin{aligned}
\mathbb{E}_{\mathbf{p}}\left[g(f(\mathrm{x}))\right] &= \mathrm{E}_{\mathbf{p}}\left[g(\Pi_{\mathrm{x}}(f))\right] && (139)\\
&= \int_{\mathbb{F}(\mathcal{X})} g \circ \Pi_{\mathrm{x}} \ \mathrm{d}\mathbf{p} && (140)\\
&= \int_{\mathbb{R}^{|\mathrm{x}|}} g \ \mathrm{d}\mathbb{P}_{\zeta_{\mathrm{x}}} && (141)\\
&= \mathbb{E}_{\mathbb{P}_{\zeta_{\mathrm{x}}}}[g]. && (142)
\end{aligned}$$

Appendix C. Membership function (17) satisfies (11)

It follows from (17) that

$$\int_{\mathbb{R}^{|\mathrm{x}|}} \zeta_{\mathrm{x}}(\mathrm{y}) \ \mathrm{d}\lambda^{|\mathrm{x}|}(\mathrm{y}) = \frac{\Gamma(\nu/2)}{\Gamma((\nu+|\mathrm{x}|)/2)} (\mathrm{pi})^{|\mathrm{x}|/2} (\nu)^{|\mathrm{x}|/2} \left(\frac{\nu-2}{\nu}\right)^{1/2} |K_{\mathrm{xx}}|^{1/2}, \tag{143}$$

$$\frac{\zeta_{\mathrm{x}}(\mathrm{y})}{\int_{\mathbb{R}^{|\mathrm{x}|}} \zeta_{\mathrm{x}}(\mathrm{y}) \ \mathrm{d}\lambda^{|\mathrm{x}|}(\mathrm{y})} = p_{\mathbf{y}}\left(\mathrm{y}; \mathrm{m}_{\mathrm{y}}, K_{\mathrm{xx}}, \nu\right), \tag{144}$$

where $p_{\mathbf{y}}\left(\mathrm{y}; \mathrm{m}_{\mathrm{y}}, K_{\mathrm{xx}}, \nu\right)$ is the density function of multivariate $t-$distribution with mean m_{y}, covariance K_{xx} (and scale matrix as equal to $((\nu-2)/\nu)K_{\mathrm{xx}}$), and degrees of freedom ν. Further, we have

$$\frac{\zeta_{\mathrm{x}\wedge\mathrm{a}}((\mathrm{y},\mathrm{u}))}{\int_{\mathbb{R}^{|\mathrm{x}|+|\mathrm{a}|}} \zeta_{\mathrm{x}\wedge\mathrm{a}}((\mathrm{y},\mathrm{u})) \ \mathrm{d}\lambda^{|\mathrm{x}|+|\mathrm{a}|}((\mathrm{y},\mathrm{u}))} = p_{(\mathbf{y},\mathbf{u})}\left((\mathrm{y},\mathrm{u}); (\mathrm{m}_{\mathrm{y}}, \mathrm{m}_{\mathrm{u}}), \begin{bmatrix} K_{\mathrm{xx}} & K_{\mathrm{xa}} \\ K_{\mathrm{ax}} & K_{\mathrm{aa}} \end{bmatrix}, \nu\right). \tag{145}$$

As the marginal distributions of multivariate $t-$distribution are also $t-$distributions (Nadarajah and Kotz, 2005) i.e.

$$\int_{\mathbb{R}^{|\mathrm{a}|}} p_{(\mathbf{y},\mathbf{u})}\left((\mathrm{y},\mathrm{u}); (\mathrm{m}_{\mathrm{y}}, \mathrm{m}_{\mathrm{u}}), \begin{bmatrix} K_{\mathrm{xx}} & K_{\mathrm{xa}} \\ K_{\mathrm{ax}} & K_{\mathrm{aa}} \end{bmatrix}, \nu\right) \ \mathrm{d}\lambda^{|\mathrm{a}|}(\mathrm{u}) = p_{\mathbf{y}}\left(\mathrm{y}; \mathrm{m}_{\mathrm{y}}, K_{\mathrm{xx}}, \nu\right), \tag{146}$$

we have

$$\frac{\int_{\mathbb{R}^{|\mathrm{a}|}} \zeta_{\mathrm{x}\wedge\mathrm{a}}((\mathrm{y},\mathrm{u})) \ \mathrm{d}\lambda^{|\mathrm{a}|}(\mathrm{u})}{\int_{\mathbb{R}^{|\mathrm{x}|+|\mathrm{a}|}} \zeta_{\mathrm{x}\wedge\mathrm{a}}((\mathrm{y},\mathrm{u})) \ \mathrm{d}\lambda^{|\mathrm{x}|+|\mathrm{a}|}((\mathrm{y},\mathrm{u}))} = \frac{\zeta_{\mathrm{x}}(\mathrm{y})}{\int_{\mathbb{R}^{|\mathrm{x}|}} \zeta_{\mathrm{x}}(\mathrm{y}) \ \mathrm{d}\lambda^{|\mathrm{x}|}(\mathrm{y})}. \tag{147}$$

For any $A \in \mathcal{B}(\mathbb{R}^{|\mathrm{x}|})$,

$$\frac{\int_{A\times\mathbb{R}^{|\mathrm{a}|}} \zeta_{\mathrm{x}\wedge\mathrm{a}}((\mathrm{y},\mathrm{u}))\ \mathrm{d}\lambda^{|\mathrm{x}|+|\mathrm{a}|}((\mathrm{y},\mathrm{u}))}{\int_{\mathbb{R}^{|\mathrm{x}|+|\mathrm{a}|}} \zeta_{\mathrm{x}\wedge\mathrm{a}}((\mathrm{y},\mathrm{u}))\ \mathrm{d}\lambda^{|\mathrm{x}|+|\mathrm{a}|}((\mathrm{y},\mathrm{u}))} = \frac{\int_A \zeta_{\mathrm{x}}(\mathrm{y})\ \mathrm{d}\lambda^{|\mathrm{x}|}(\mathrm{y})}{\int_{\mathbb{R}^{|\mathrm{x}|}} \zeta_{\mathrm{x}}(\mathrm{y})\ \mathrm{d}\lambda^{|\mathrm{x}|}(\mathrm{y})}. \tag{148}$$

Thus, (11) is satisfied.

Appendix D. Evaluation of $\mu_{\mathrm{y}_j;\mathrm{u}_j}$

Using (44), we have

$$\left\langle \log(\mu_{\mathrm{y}_j;\mathrm{f}_j}(\tilde{\mathrm{y}}_j))\right\rangle_{\mu_{\mathrm{f}_j;\mathrm{u}_j}} = -0.5\tau z\|\tilde{\mathrm{y}}_j - \bar{m}_{\mathrm{f}_j}\|^2 - 0.5\tau z\frac{\nu + (\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j - 2}{\nu + M - 2}Tr(\bar{K}_{\mathrm{xx}})$$

where $Tr(\cdot)$ denotes the trace operator. Using (45) and (46),

$$\begin{aligned}\left\langle \log(\mu_{\mathrm{y}_j;\mathrm{f}_j}(\tilde{\mathrm{y}}_j))\right\rangle_{\mu_{\mathrm{f}_j;\mathrm{u}_j}} &= -0.5\tau z\|\tilde{\mathrm{y}}_j\|^2 + \tau z(\tilde{\mathrm{y}}_j)^T K_{\mathrm{xa}}(K_{\mathrm{aa}})^{-1}\mathrm{u}_j - 0.5\tau z(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}K_{\mathrm{ax}}K_{\mathrm{xa}}(K_{\mathrm{aa}})^{-1}\mathrm{u}_j \\ &\quad -0.5\tau z\frac{\nu + (\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j - 2}{\nu + M - 2}\left(Tr(K_{\mathrm{xx}}) - Tr((K_{\mathrm{aa}})^{-1}K_{\mathrm{ax}}K_{\mathrm{xa}})\right)\end{aligned} \tag{149}$$

Define

$$\xi = \sum_{i=1}^{N}\left\langle kr(x^i,x^i)\right\rangle_{\mu_{x^i}}. \tag{150}$$

Define a matrix $\Psi \in \mathbb{R}^{N\times M}$ with $(i,m)-$th element as

$$\Psi_{i,m} = \left\langle kr(x^i,a^m)\right\rangle_{\mu_{x^i}}. \tag{151}$$

Define a matrix $\Phi \in \mathbb{R}^{M\times M}$ with $(m,m')-$th element as

$$\Phi_{m,m'} = \sum_{i=1}^{N}\left\langle kr(a^m,x^i)kr(x^i,a^{m'})\right\rangle_{\mu_{x^i}}. \tag{152}$$

Using (19) and (36) in (150), (151), and (152),

$$\xi = N\sigma^2 \tag{153}$$

$$\Psi_{i,m} = \frac{\sigma^2}{\prod_{k=1}^{n}\left(\sqrt{1+w_k\sigma_x^2}\right)}\exp\left(-\frac{1}{2}\sum_{k=1}^{n}\frac{w_k|a_k^m - x_k^i|^2}{1+w_k\sigma_x^2}\right) \tag{154}$$

$$\Phi_{m,m'} = \frac{\sigma^4}{\prod_{k=1}^{n}\left(\sqrt{1+2w_k\sigma_x^2}\right)}\sum_{i=1}^{N}\exp\left(-\frac{1}{4}\sum_{k=1}^{n}w_k(a_k^m - a_k^{m'})^2 - \sum_{k=1}^{n}\frac{w_k|0.5(a_j^m + a_k^{m'}) - x_k^i|^2}{1+2w_k\sigma_x^2}\right) \tag{155}$$

where a_k^m and x_k^i denotes the $k-$th element of a^m and x^i respectively. Using (149),

$$\begin{aligned}&\left\langle \cdots\left\langle\left\langle \log(\mu_{\mathrm{y}_j;\mathrm{f}_j}(\tilde{\mathrm{y}}_j))\right\rangle_{\mu_{\mathrm{f}_j;\mathrm{u}_j}}\right\rangle_{\mu_{x^1}}\cdots\right\rangle_{\mu_{x^N}} \\ &= -0.5\tau z\|\tilde{\mathrm{y}}_j\|^2 + \tau z(\tilde{\mathrm{y}}_j)^T\Psi(K_{\mathrm{aa}})^{-1}\mathrm{u}_j - 0.5\tau z(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1}\mathrm{u}_j \\ &\quad -\frac{\tau z}{2}\frac{\nu + (\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j - 2}{\nu + M - 2}\left(\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)\right).\end{aligned}$$

Using (47),

$$\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j) \propto \exp\Big(-0.5\tau z\|\tilde{\mathrm{y}}_j\|^2 + \tau z(\tilde{\mathrm{y}}_j)^T\Psi(K_{\mathrm{aa}})^{-1}\mathrm{u}_j - 0.5\tau z(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1}\mathrm{u}_j \\ -\frac{\tau z}{2}\frac{(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j}{\nu+M-2}\left(\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)\right) + \{/(\tilde{\mathrm{y}}_j,\mathrm{u}_j)\}\Big)$$

where $\{/(\tilde{\mathrm{y}}_j,\mathrm{u}_j)\}$ represents all those terms which are independent of both $\tilde{\mathrm{y}}_j$ and u_j. Define

$$\hat{K}_{\mathrm{u}_j} = \left((K_{\mathrm{aa}})^{-1} + \tau z(K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1} + \tau z\frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu+M-2}(K_{\mathrm{aa}})^{-1}\right)^{-1} \tag{156}$$

$$\hat{m}_{\mathrm{u}_j}(\tilde{\mathrm{y}}_j) = \tau z\hat{K}_{\mathrm{u}_j}(K_{\mathrm{aa}})^{-1}(\Psi)^T\tilde{\mathrm{y}}_j \tag{157}$$

to express $\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j)$ as in (48).

Appendix E. Proof of Result 1

A new objective functional is defined after excluding u_j-independent terms and taking into account the integral constraint through a Lagrange multiplier γ:

$$\mathcal{J} = \Big\langle (\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) - 0.5(\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\mathrm{u}_j + 0.5(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j - \log(\mu_{\mathrm{u}_j}(\mathrm{u}_j)) - 0.5(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}\mathrm{u}_j\Big\rangle_{\mu_{\mathrm{u}_j}}$$
$$+\gamma\left\{\int_{\mathbb{R}^M}\mu_{\mathrm{u}_j}(\mathrm{u}_j)\,\mathrm{d}\lambda^M(\mathrm{u}_j) - C_{\mathrm{u}_j}\right\} \tag{158}$$
$$= \frac{1}{C_{\mathrm{u}_j}}\int_{\mathbb{R}^M}\mathrm{d}\lambda^M(\mathrm{u}_j)\,\mu_{\mathrm{u}_j}(\mathrm{u}_j)\left\{(\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) - 0.5(\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\mathrm{u}_j - \log(\mu_{\mathrm{u}_j}(\mathrm{u}_j))\right\}$$
$$+\gamma\left\{\int_{\mathbb{R}^M}\mu_{\mathrm{u}_j}(\mathrm{u}_j)\,\mathrm{d}\lambda^M(\mathrm{u}_j) - C_{\mathrm{u}_j}\right\} \tag{159}$$

Setting the functional derivative of $\mathcal{J}$ w.r.t. μ_{u_j} equal to zero,

$$0 = \gamma + (1/C_{\mathrm{u}_j})\left[-1 - 0.5(\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\mathrm{u}_j + (\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) - \log(\mu_{\mathrm{u}_j}(\mathrm{u}_j))\right]. \tag{160}$$

That is,

$$\mu_{\mathrm{u}_j}(\mathrm{u}_j) = \exp(\gamma C_{\mathrm{u}_j} - 1)\exp\left(-0.5(\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\mathrm{u}_j + (\mathrm{u}_j)^T\hat{K}_{\mathrm{u}_j}^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right). \tag{161}$$

The optimal value of γ is obtained by solving $\int_{\mathbb{R}^M}\mu_{\mathrm{u}_j}\,\mathrm{d}\lambda^M = C_{\mathrm{u}_j}$. This leads to

$$\exp(\gamma C_{\mathrm{u}_j} - 1)\sqrt{(2\pi)^M/|\hat{K}_{\mathrm{u}_j}^{-1}|}\exp\left(0.5\left(\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right)^T\hat{K}_{\mathrm{u}_j}^{-1}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)\right) = C_{\mathrm{u}_j}. \tag{162}$$

Thus, the optimal expression for μ_{u_j} is given as

$$\mu^*_{\mathrm{u}_j}(\mathrm{u}_j) = C_{\mathrm{u}_j}\sqrt{|\hat{K}_{\mathrm{u}_j}^{-1}|/(2\pi)^M}\exp\left(-0.5(\mathrm{u}_j - \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j))^T\hat{K}_{\mathrm{u}_j}^{-1}(\mathrm{u}_j - \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j))\right). \tag{163}$$

Finally, C_{u_j} is chosen such that $\max_{\mathrm{u}_j}\mu^*_{\mathrm{u}_j}(\mathrm{u}_j) = 1$. This results in

$$\mu^*_{\mathrm{u}_j}(\mathrm{u}_j) = \exp\left(-0.5(\mathrm{u}_j - \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j))^T\hat{K}_{\mathrm{u}_j}^{-1}(\mathrm{u}_j - \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j))\right). \tag{164}$$

Thus, $\langle \mathrm{u}_j \rangle_{\mu^*_{\mathrm{u}_j}} = \hat{m}_{\mathrm{u}_j}(\mathrm{y}_j)$, and using (157), we get (54).

Using

$$\hat{K}^{-1}_{\mathrm{u}_j} - (K_{\mathrm{aa}})^{-1} = \tau z (K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1} + \tau z \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2}(K_{\mathrm{aa}})^{-1} \tag{165}$$

$$\hat{K}^{-1}_{\mathrm{u}_j}\hat{m}_{\mathrm{u}_j}(\mathrm{y}_j) = \tau z (K_{\mathrm{aa}})^{-1}(\Psi)^T \mathrm{y}_j, \tag{166}$$

we have

$$\begin{aligned}\log(\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j)) = &-0.5\tau z\|\tilde{\mathrm{y}}_j\|^2 + \tau z(\mathrm{u}_j)^T(K_{\mathrm{aa}})^{-1}(\Psi)^T\tilde{\mathrm{y}}_j \\ &-0.5\tau z(\mathrm{u}_j)^T\left\{(K_{\mathrm{aa}})^{-1}\Phi(K_{\mathrm{aa}})^{-1} + \frac{\xi - Tr((K_{\mathrm{aa}})^{-1}\Phi)}{\nu + M - 2}(K_{\mathrm{aa}})^{-1}\right\}\mathrm{u}_j.\end{aligned}$$

Thus, $\left\langle \log(\mu_{\mathrm{y}_j;\mathrm{u}_j}(\tilde{\mathrm{y}}_j))\right\rangle_{\mu^*_{\mathrm{u}_j}}$ is given as in (55).

Appendix F. Proof of Result 2

J can be rewritten using (60), (59), (78), and (42) as

$$J = -0.5\left\langle \tau \right\rangle_{q_\tau}\left\langle z \right\rangle_{q_z} O + 0.5Np\left\langle \log(\tau)\right\rangle_{q_\tau} + 0.5Np\left\langle \log(z)\right\rangle_{q_z} - \left\langle \log\left(q_\Omega(\Omega)/\mu_\Omega(\Omega)\right)\right\rangle_{q_\Omega}.$$

F.1 Optimization of q_τ

J can be separated into $\tau-$dependent and $\tau-$independent terms as follows:

$$J = -0.5\left\langle \tau \right\rangle_{q_\tau}\left\langle z \right\rangle_{q_z} O + 0.5Np\left\langle \log(\tau)\right\rangle_{q_\tau} - \left\langle \log\left(q_\tau(\tau)/\mu_\tau(\tau)\right)\right\rangle_{q_\tau} + \{/\tau\}$$

where $\{/\tau\}$ represents all $\tau-$independent terms. After substituting the value of μ_τ from (38), J becomes

$$J = -0.5\left\langle \tau \right\rangle_{q_\tau}\left\langle z \right\rangle_{q_z} O + (a_\tau + 0.5Np - 1)\left\langle \log(\tau)\right\rangle_{q_\tau} - b_\tau\left\langle \tau \right\rangle_{q_\tau} - \left\langle \log\left(q_\tau(\tau)\right)\right\rangle_{q_\tau} + \{/\tau\}.$$

Define

$$\begin{aligned}\hat{a}_\tau &= a_\tau + 0.5Np \\ \hat{b}_\tau &= b_\tau + 0.5\left\langle z \right\rangle_{q_z} O\end{aligned}$$

to express J as

$$\begin{aligned}J &= \left\langle (\hat{a}_\tau - 1)\log(\tau) - \hat{b}_\tau\tau - \log\left(q_\tau(\tau)\right)\right\rangle_{q_\tau} + \{/\tau\} \\ &= \frac{1}{C_\tau}\int_{\mathbb{R}_{>0}} \mathrm{d}\lambda^1(\tau)\, q_\tau(\tau)\left\{(\hat{a}_\tau - 1)\log(\tau) - \hat{b}_\tau\tau - \log\left(q_\tau(\tau)\right)\right\} + \{/\tau\}.\end{aligned}$$

The constraint on q_τ can be incorporated in the optimization problem by the use of Lagrange multiplier γ to define the new functional as

$$\mathcal{J} = (1/C_\tau)\int_{\mathbb{R}_{>0}} \mathrm{d}\lambda^1(\tau)\, q_\tau(\tau)\left\{(\hat{a}_\tau - 1)\log(\tau) - \hat{b}_\tau\tau - \log\left(q_\tau(\tau)\right)\right\} + \gamma\left\{\int_{\mathbb{R}_{>0}} \mathrm{d}\lambda^1(\tau)\, q_\tau(\tau) - C_\tau\right\}.$$

Setting the functional derivative of $\mathcal{J}$ w.r.t. $q_\tau(\tau)$ equal to zero,

$$(1/C_\tau)\left\{(\hat{a}_\tau - 1)\log(\tau) - \hat{b}_\tau\tau - 1 - \log(q_\tau(\tau))\right\} + \gamma = 0.$$

That is,

$$q_\tau(\tau) = \exp(\gamma C_\tau - 1)\,(\tau)^{\hat{a}_\tau - 1}\exp\left(-\hat{b}_\tau\tau\right).$$

The optimal value of γ is obtained by solving $\int_{\mathbb{R}_{>0}} \mathrm{d}\lambda^1(\tau)\, q_\tau(\tau) = C_\tau$. This leads to

$$\exp(\gamma C_\tau - 1)\,(\Gamma(\hat{a}_\tau)/(\hat{b}_\tau)^{\hat{a}_\tau}) = C_\tau$$

where $\Gamma(\cdot)$ is the Gamma function. Thus, the optimal expression for $q(\tau)$ is given as

$$q^*_\tau(\tau; C_\tau) = C_\tau\left((\hat{b}_\tau)^{\hat{a}_\tau}/\Gamma(\hat{a}_\tau)\right)(\tau)^{\hat{a}_\tau - 1}\exp\left(-\hat{b}_\tau\tau\right).$$

Finally, C_τ is chosen to make $\max_\tau q^*_\tau(\tau; C_\tau) = 1$. This consideration results in

$$q^*_\tau(\tau) = \left(\hat{b}_\tau/(\hat{a}_\tau - 1)\right)^{\hat{a}_\tau - 1}\exp(\hat{a}_\tau - 1)(\tau)^{\hat{a}_\tau - 1}\exp\left(-\hat{b}_\tau\tau\right).$$

F.2 Optimization of q_z

J can be separated into $z-$dependent and $z-$independent terms as follows:

$$J = -0.5\left\langle\tau\right\rangle_{q_\tau}\left\langle z\right\rangle_{q_z} O + 0.5Np\left\langle\log(z)\right\rangle_{q_z} - \left\langle\left\langle\left\langle\log\left(q_z(z)/\mu_z(z)\right)\right\rangle_{q_z}\right\rangle_{q_r}\right\rangle_{q_s} + \{/z\}.$$

Substituting the value of μ_z from (39), J becomes

$$J = -0.5\left\langle\tau\right\rangle_{q_\tau}\left\langle z\right\rangle_{q_z} O + \left(0.5Np + \left\langle r\right\rangle_{q_r}\right)\left\langle\log(z)\right\rangle_{q_z} - \left\langle r\right\rangle_{q_r}\left\langle s\right\rangle_{q_s}\left\langle z\right\rangle_{q_z} - \left\langle\log\left(q_z(z)\right)\right\rangle_{q_z} + \{/z\}.$$

Define

$$\begin{aligned}\hat{a}_z &= 1 + 0.5Np + \left\langle r\right\rangle_{q_r}\\ \hat{b}_z &= \left\langle r\right\rangle_{q_r}\left\langle s\right\rangle_{q_s} + 0.5\left\langle\tau\right\rangle_{q_\tau} O\end{aligned}$$

to express J as

$$\begin{aligned}J &= \left\langle(\hat{a}_z - 1)\log(z) - \hat{b}_z z - \log\left(q_z(z)\right)\right\rangle_{q_z} + \{/z\}\\ &= \frac{1}{C_z}\int_{\mathbb{R}_{>0}} \mathrm{d}\lambda^1(z)\, q_z(z)\left\{(\hat{a}_z - 1)\log(z) - \hat{b}_z z - \log\left(q_z(z)\right)\right\} + \{/z\}.\end{aligned}$$

Maximizing J w.r.t. $q_z(z)$ under the constraint: $\int_{\mathbb{R}_{>0}} \mathrm{d}\lambda^1(z)\, q_z(z) = C_z$, and then choosing C_z such that $\max_z q_z(z) = 1$, results in

$$q^*_z(z) = \left(\hat{b}_z/(\hat{a}_z - 1)\right)^{\hat{a}_z - 1}\exp(\hat{a}_z - 1)(z)^{\hat{a}_z - 1}\exp\left(-\hat{b}_z z\right).$$

F.3 Optimization of q_r

J can be partitioned into $r-$dependent and $r-$independent terms as follows

$$\begin{aligned}J &= -\left\langle \left\langle \langle \log\left(q_r(r)/\left(\mu_z(z)\mu_r(r)\right)\right)\rangle_{q_z} \right\rangle_{q_r} \right\rangle_{q_s} + \{/r\} \\ &= \langle r\rangle_{q_r} \langle \log(s)\rangle_{q_s} + \langle r\rangle_{q_r} + \langle r\rangle_{q_r} \langle \log(z)\rangle_{q_z} - \langle r\rangle_{q_r} \langle s\rangle_{q_s} \langle z\rangle_{q_z} + (a_r - 1)\langle \log(r)\rangle_{q_r} \\ &\quad -b_r \langle r\rangle_{q_r} - \langle \log(q_r(r))\rangle_{q_r} + \{/r\}.\end{aligned}$$

Define

$$\begin{aligned}\hat{a}_r &= a_r \\ \hat{b}_r &= b_r + \langle s\rangle_{q_s} \langle z\rangle_{q_z} - \langle \log(s)\rangle_{q_s} - 1 - \langle \log(z)\rangle_{q_z}\end{aligned}$$

to express J as

$$J = \left\langle (\hat{a}_r - 1)\log(r) - \hat{b}_r r - \log(q_r(r)) \right\rangle_{q_r} + \{/r\}.$$

Maximizing J w.r.t. $q_r(r)$ under the constraint: $\int_{\mathbb{R}_{>0}} \mathrm{d}\lambda^1(r)\, q_r(r) = C_r$, and then choosing C_r such that $\max_r q_r(r) = 1$, results in

$$q_r^*(r) = \left(\hat{b}_r/(\hat{a}_r - 1)\right)^{\hat{a}_r - 1} \exp(\hat{a}_r - 1)(r)^{\hat{a}_r - 1} \exp\left(-\hat{b}_r r\right).$$

F.4 Optimization of q_s

J can be partitioned into $s-$dependent and $s-$independent terms as follows

$$\begin{aligned}J &= -\left\langle \left\langle \langle \log\left(q_s(s)/\left(\mu_z(z)\mu_s(s)\right)\right)\rangle_{q_z} \right\rangle_{q_r} \right\rangle_{q_s} + \{/s\} \\ &= \langle r\rangle_{q_r} \langle \log(s)\rangle_{q_s} - \langle r\rangle_{q_r} \langle s\rangle_{q_s} \langle z\rangle_{q_z} + (a_s - 1)\langle \log(s)\rangle_{q_s} - b_s \langle s\rangle_{q_s} - \langle \log(q_s(s))\rangle_{q_s} + \{/s\}.\end{aligned}$$

Define

$$\begin{aligned}\hat{a}_s &= a_s + \langle r\rangle_{q_r} \\ \hat{b}_s &= b_s + \langle r\rangle_{q_r} \langle z\rangle_{q_z}\end{aligned}$$

to express J as

$$J = \left\langle (\hat{a}_s - 1)\log(s) - \hat{b}_s s - \log(q_s(s)) \right\rangle_{q_s} + \{/s\}.$$

Maximizing J w.r.t. $q_s(s)$ under the constraint: $\int_{\mathbb{R}_{>0}} \mathrm{d}\lambda^1(s)\, q_s(s) = C_s$, and then choosing C_s such that $\max_s q_s(s) = 1$, results in

$$q_s^*(s) = \left(\hat{b}_s/(\hat{a}_s - 1)\right)^{\hat{a}_s - 1} \exp(\hat{a}_s - 1)(s)^{\hat{a}_s - 1} \exp\left(-\hat{b}_s s\right).$$

F.5 Evaluation of Weighted Averages

Having derived the expression for membership functions, the averages can be evaluated as $\langle \tau\rangle_{q_\tau^*} = \hat{a}_\tau/\hat{b}_\tau$, $\langle z\rangle_{q_z^*} = \hat{a}_z/\hat{b}_z$, $\langle \log(z)\rangle_{q_z^*} = \psi(\hat{a}_z) - \log(\hat{b}_z)$, $\langle r\rangle_{q_r^*} = \hat{a}_r/\hat{b}_r$, $\langle s\rangle_{q_s^*} = \hat{a}_s/\hat{b}_s$, $\langle \log(s)\rangle_{q_s^*} = \psi(\hat{a}_s) - \log(\hat{b}_s)$, where $\psi(\cdot)$ is the digamma function.

Table 4: Accuracy (in %, averaged over 20 experiments) obtained in *amazon→caltech256* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	82.4
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	**84.2**
non-private proposed	VGG-FC6	84.0
SVM-t (without knowledge transfer)	VGG-FC6	73.9
non-private ILS (1-NN)	VGG-FC6	83.3
non-private CDLS	VGG-FC6	78.1
non-private MMDT	VGG-FC6	78.7
non-private HFA	VGG-FC6	75.5
non-private OBTL	SURF	41.5
non-private ILS (1-NN)	SURF	43.6
non-private CDLS	SURF	35.3
non-private MMDT	SURF	36.4
non-private HFA	SURF	31.0

F.6 Estimation of Membership Functions' Parameters

The set of equations for estimating optimal membership functions' parameter, obtained after evaluating the integrals, are summarized as (67, 68, 70, 71, 73, 74, 76, 77) in Result 2.

Appendix G. Results of Experiments on "Office+Caltech256" Dataset

Table 5: Accuracy (in %, averaged over 20 experiments) obtained in *amazon*→*dslr* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	**91.8**
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	91.5
non-private proposed	VGG-FC6	91.5
SVM-t (without knowledge transfer)	VGG-FC6	89.6
non-private ILS (1-NN)	VGG-FC6	87.7
non-private CDLS	VGG-FC6	86.9
non-private MMDT	VGG-FC6	77.1
non-private HFA	VGG-FC6	87.1
non-private OBTL	SURF	60.2
non-private ILS (1-NN)	SURF	49.8
non-private CDLS	SURF	60.4
non-private MMDT	SURF	56.7
non-private HFA	SURF	55.1

Table 6: Accuracy (in %, averaged over 20 experiments) obtained in *amazon*→*webcam* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	91.4
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	91.7
non-private proposed	VGG-FC6	**92.2**
SVM-t (without knowledge transfer)	VGG-FC6	88.8
non-private ILS (1-NN)	VGG-FC6	90.7
non-private CDLS	VGG-FC6	91.2
non-private MMDT	VGG-FC6	82.5
non-private HFA	VGG-FC6	87.9
non-private OBTL	SURF	72.4
non-private ILS (1-NN)	SURF	59.7
non-private CDLS	SURF	68.7
non-private MMDT	SURF	64.6
non-private HFA	SURF	57.4

Table 7: Accuracy (in %, averaged over 20 experiments) obtained in *caltech256→amazon* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	92.4
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	**92.8**
non-private proposed	VGG-FC6	**92.8**
SVM-t (without knowledge transfer)	VGG-FC6	85.1
non-private ILS (1-NN)	VGG-FC6	89.7
non-private CDLS	VGG-FC6	88.0
non-private MMDT	VGG-FC6	85.9
non-private HFA	VGG-FC6	86.2
non-private OBTL	SURF	54.8
non-private ILS (1-NN)	SURF	55.1
non-private CDLS	SURF	50.9
non-private MMDT	SURF	49.4
non-private HFA	SURF	43.8

Table 8: Accuracy (in %, averaged over 20 experiments) obtained in *caltech256→dslr* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	**92.2**
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	89.1
non-private proposed	VGG-FC6	91.0
SVM-t (without knowledge transfer)	VGG-FC6	89.3
non-private ILS (1-NN)	VGG-FC6	86.9
non-private CDLS	VGG-FC6	86.3
non-private MMDT	VGG-FC6	77.9
non-private HFA	VGG-FC6	87.0
non-private OBTL	SURF	61.5
non-private ILS (1-NN)	SURF	56.2
non-private CDLS	SURF	59.8
non-private MMDT	SURF	56.5
non-private HFA	SURF	55.6

Table 9: Accuracy (in %, averaged over 20 experiments) obtained in *caltech256→webcam* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	90.2
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	90.6
non-private proposed	VGG-FC6	91.2
SVM-t (without knowledge transfer)	VGG-FC6	87.1
non-private ILS (1-NN)	VGG-FC6	**91.4**
non-private CDLS	VGG-FC6	89.7
non-private MMDT	VGG-FC6	82.8
non-private HFA	VGG-FC6	86.0
non-private OBTL	SURF	71.1
non-private ILS (1-NN)	SURF	62.9
non-private CDLS	SURF	66.3
non-private MMDT	SURF	63.8
non-private HFA	SURF	58.1

Table 10: Accuracy (in %, averaged over 20 experiments) obtained in *dslr→amazon* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	91.1
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	91.6
non-private proposed	VGG-FC6	**91.8**
SVM-t (without knowledge transfer)	VGG-FC6	84.6
non-private ILS (1-NN)	VGG-FC6	88.7
non-private CDLS	VGG-FC6	88.1
non-private MMDT	VGG-FC6	83.6
non-private HFA	VGG-FC6	85.9
non-private OBTL	SURF	54.4
non-private ILS (1-NN)	SURF	55.0
non-private CDLS	SURF	50.7
non-private MMDT	SURF	46.9
non-private HFA	SURF	42.9

Table 11: Accuracy (in %, averaged over 20 experiments) obtained in *dslr→caltech256* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	82.6
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	83.5
non-private proposed	VGG-FC6	**85.1**
SVM-t (without knowledge transfer)	VGG-FC6	76.0
non-private ILS (1-NN)	VGG-FC6	81.4
non-private CDLS	VGG-FC6	77.9
non-private MMDT	VGG-FC6	71.8
non-private HFA	VGG-FC6	74.8
non-private OBTL	SURF	40.3
non-private ILS (1-NN)	SURF	41.0
non-private CDLS	SURF	34.9
non-private MMDT	SURF	34.1
non-private HFA	SURF	30.9

Table 12: Accuracy (in %, averaged over 20 experiments) obtained in *dslr→webcam* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	92.7
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	93.3
non-private proposed	VGG-FC6	94.8
SVM-t (without knowledge transfer)	VGG-FC6	88.6
non-private ILS (1-NN)	VGG-FC6	**95.5**
non-private CDLS	VGG-FC6	90.7
non-private MMDT	VGG-FC6	86.1
non-private HFA	VGG-FC6	86.9
non-private OBTL	SURF	83.2
non-private ILS (1-NN)	SURF	80.1
non-private CDLS	SURF	68.5
non-private MMDT	SURF	74.1
non-private HFA	SURF	60.5

Table 13: Accuracy (in %, averaged over 20 experiments) obtained in *webcam*→*amazon* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	92.5
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	92.7
non-private proposed	VGG-FC6	**92.8**
SVM-t (without knowledge transfer)	VGG-FC6	85.7
non-private ILS (1-NN)	VGG-FC6	88.8
non-private CDLS	VGG-FC6	87.4
non-private MMDT	VGG-FC6	84.7
non-private HFA	VGG-FC6	85.1
non-private OBTL	SURF	55.0
non-private ILS (1-NN)	SURF	54.3
non-private CDLS	SURF	51.8
non-private MMDT	SURF	47.7
non-private HFA	SURF	56.5

Table 14: Accuracy (in %, averaged over 20 experiments) obtained in *webcam*→*caltech256* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1e{-}5)-$differentially private proposed	VGG-FC6	80.0
$(1, 1e{-}5)-$differentially private proposed	VGG-FC6	81.3
non-private proposed	VGG-FC6	**82.9**
SVM-t (without knowledge transfer)	VGG-FC6	73.6
non-private ILS (1-NN)	VGG-FC6	82.8
non-private CDLS	VGG-FC6	78.2
non-private MMDT	VGG-FC6	73.6
non-private HFA	VGG-FC6	74.4
non-private OBTL	SURF	37.4
non-private ILS (1-NN)	SURF	38.6
non-private CDLS	SURF	33.5
non-private MMDT	SURF	32.2
non-private HFA	SURF	29.0

Table 15: Accuracy (in %, averaged over 20 experiments) obtained in *webcam*→*dslr* semi-supervised transfer learning experiments.

method	**feature type**	**accuracy (%)**
$(0.1, 1\text{e}{-5})-$differentially private proposed	VGG-FC6	93.6
$(1, 1\text{e}{-5})-$differentially private proposed	VGG-FC6	91.5
non-private proposed	VGG-FC6	94.4
SVM-t (without knowledge transfer)	VGG-FC6	89.1
non-private ILS (1-NN)	VGG-FC6	**94.5**
non-private CDLS	VGG-FC6	88.5
non-private MMDT	VGG-FC6	85.1
non-private HFA	VGG-FC6	87.3
non-private OBTL	SURF	75.0
non-private ILS (1-NN)	SURF	70.8
non-private CDLS	SURF	60.7
non-private MMDT	SURF	67.0
non-private HFA	SURF	56.5

References

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, page 308–318, New York, NY, USA, 2016. Association for Computing Machinery. doi: 10.1145/2976749.2978318.

G. Acs, L. Melis, C. Castelluccia, and E. De Cristofaro. Differentially private mixture of generative neural networks. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 715–720, 2017.

Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. *CoRR*, abs/1805.06530, 2018.

Marouan Belhaj, Pavlos Protopapas, and Weiwei Pan. Deep variational transfer: Transfer learning through semi-supervised deep generative models. *ArXiv*, abs/1812.03123, 2018.

Lorenzo Bruzzone and Mattia Marconcini. Domain adaptation problems: A dasvm classification technique and a circular validation strategy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(5):770–787, May 2010. ISSN 0162-8828.

N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9): 1853–1865, 2017.

Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 193–200, New York, NY, USA, 2007. Association for Computing Machinery. ISBN 9781595937933.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014. doi: 10.1561/0400000042. URL `https://doi.org/10.1561/0400000042`.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, pages 486–503, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-34547-3.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 1322–1333, New York, NY, USA, 2015. ACM. doi: 10.1145/2810103.2813677. URL `http://doi.acm.org/10.1145/2810103.2813677`.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17(1):2096–2030, January 2016. ISSN 1532-4435.

Q. Geng and P. Viswanath. The optimal noise-adding mechanism in differential privacy. *IEEE Transactions on Information Theory*, 62(2):925–951, Feb 2016a. doi: 10.1109/TIT.2015.2504967.

Q. Geng and P. Viswanath. Optimal noise adding mechanisms for approximate differential privacy. *IEEE Transactions on Information Theory*, 62(2):952–969, Feb 2016b. doi: 10.1109/TIT.2015.2504972.

Q. Geng, P. Kairouz, S. Oh, and P. Viswanath. The staircase mechanism in differential privacy. *IEEE Journal of Selected Topics in Signal Processing*, 9(7):1176–1184, Oct 2015. doi: 10.1109/JSTSP.2015.2425831.

Quan Geng, Wei Ding, Ruiqi Guo, and Sanjiv Kumar. Optimal noise-adding mechanism in additive differential privacy. *CoRR*, abs/1809.10224, 2018.

A. Ghosh, T. Roughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. *SIAM Journal on Computing*, 41(6):1673–1693, 2012. doi: 10.1137/09076828X.

B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2066–2073, 2012.

Mangesh Gupte and Mukund Sundararajan. Universally optimal privacy mechanisms for minimax agents. In *Proceedings of the Twenty-ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '10, pages 135–146, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0033-9. doi: 10.1145/1807085.1807105.

J. He, L. Cai, and X. Guan. Differential private noise adding mechanism and its application on consensus algorithm. *IEEE Transactions on Signal Processing*, 68:4069–4082, 2020. doi: 10.1109/TSP.2020.3006760.

Samitha Herath, Mehrtash Harandi, and Fatih Porikli. Learning an invariant hilbert space for domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Judy Hoffman, Erik Rodner, Jeff Donahue, Kate Saenko, and Trevor Darrell. Efficient learning of domain-invariant image representations. *CoRR*, abs/1301.3224, 2013.

Judy Hoffman, Erik Rodner, Jeff Donahue, Brian Kulis, and Kate Saenko. Asymmetric and category invariant feature transformations for domain adaptation. *International Journal of Computer Vision*, 109(1):28–41, 2014. doi: 10.1007/s11263-014-0719-3.

Zhanglong Ji and Charles Elkan. Differential privacy based on importance weighting. *Machine Learning*, 93(1):163–183, 2013.

A. Karbalayghareh, X. Qian, and E. R. Dougherty. Optimal bayesian transfer learning. *IEEE Transactions on Signal Processing*, 66(14):3724–3739, 2018.

M. Kumar and B. Freudenthaler. Fuzzy membership functional analysis for nonparametric deep models of image features. *IEEE Transactions on Fuzzy Systems*, pages 1–1, 2019. doi: 10.1109/TFUZZ.2019.2950636.

M. Kumar, M. Rossbory, B. A. Moser, and B. Freudenthaler. Deriving an optimal noise adding mechanism for privacy-preserving machine learning. In Gabriele Anderst-Kotsis, A Min Tjoa, Ismail Khalil, Mourad Elloumi, Atif Mashkoor, Johannes Sametinger, Xabier Larrucea, Anna Fensel, Jorge Martinez-Gil, Bernhard Moser, Christin Seifert, Benno Stein, and Michael Granitzer, editors, *Proceedings of the 3rd International Workshop on Cyber-Security and Functional Safety in Cyber-Physical (IWCFS 2019), August 26-29, 2019, Linz, Austria*, pages 108–118, Cham, 2019. Springer International Publishing.

M. Kumar, W. Zhang, M. Weippert, and B. Freudenthaler. An explainable fuzzy theoretic nonparametric deep model for stress assessment using heartbeat intervals analysis. *IEEE Transactions on Fuzzy Systems*, pages 1–1, 2020. doi: 10.1109/TFUZZ.2020.3029284.

M. Kumar, M. Rossbory, B. A. Moser, and B. Freudenthaler. An optimal $(\epsilon, \delta)-$differentially private learning of distributed deep fuzzy models. *Information Sciences*, 546:87–120, 2021. doi: https://doi.org/10.1016/j.ins.2020.07.044.

Mohit Kumar, Michael Rossbory, Bernhard A. Moser, and Bernhard Freudenthaler. Differentially private learning of distributed deep models. In *Adjunct Publication of the 28th ACM Conference on User Modeling, Adaptation and Personalization*, UMAP '20 Adjunct, pages 193–200, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379502. doi: 10.1145/3386392.3399562.

W. Li, L. Duan, D. Xu, and I. W. Tsang. Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1134–1148, 2014.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 97–105, Lille, France, 07–09 Jul 2015. PMLR.

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 136–144, Red Hook, NY, USA, 2016. Curran Associates Inc.

Saralees Nadarajah and Samuel Kotz. Mathematical properties of the multivariate t distribution. *Acta Applicandae Mathematica*, 89(1):53–84, Dec 2005. ISSN 1572-9036. doi: 10.1007/s10440-005-9003-4.

Teppo Niinimäki, Mikko A Heikkilä, Antti Honkela, and Samuel Kaski. Representation transfer for differentially private drug sensitivity prediction. *Bioinformatics*, 35(14):i218–i224, 07 2019.

Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *ICLR*. OpenReview.net, 2017. URL `http://dblp.uni-trier.de/db/conf/iclr/iclr2017.html#PapernotAEGT17`.

NhatHai Phan, Yue Wang, Xintao Wu, and Dejing Dou. Differential privacy preservation for deep auto-encoders: An application of human behavior prediction. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 1309–1316. AAAI Press, 2016. URL `http://dl.acm.org/citation.cfm?id=3015812.3016005`.

Y. H. Tsai, Y. Yeh, and Y. F. Wang. Learning cross-domain landmarks for heterogeneous domain adaptation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5081–5090, 2016.

Yang Wang, Quanquan Gu, and Donald E. Brown. Differentially private hypothesis transfer learning. In Michele Berlingerio, Francesco Bonchi, Thomas Gärtner, Neil Hurley, and Georgiana Ifrim, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Dublin, Ireland, September 10-14, 2018, Proceedings, Part II*, volume 11052 of *Lecture Notes in Computer Science*, pages 811–826. Springer, 2018.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *ArXiv*, abs/1802.06739, 2018.

Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Trans. Database Syst.*, 42 (4), October 2017. doi: 10.1145/3134428.