Machine Learning (2023) 112:311–351
https://doi.org/10.1007/s10994-022-06279-5

Reconciling privacy and utility: an unscented Kalman filter-based framework for differentially private machine learning

Kunsheng Tang[1,2] · **Ping Li**[1] · **Yide Song**[1] · **Tian Luo**[1]

Received: 31 March 2022 / Revised: 8 August 2022 / Accepted: 9 November 2022 /
Published online: 7 December 2022
© The Author(s), under exclusive licence to Springer Science+Business Media LLC, part of Springer Nature 2022

Abstract
Machine learning (ML) is extensively used in fields involving sensitive data, data holders are seeking to protect the privacy of data and build the ML models with high-quality utility. Differential privacy provides a feasible solution for them. However, there is a mutual constraint on the privacy and utility of the models under this solution. Therefore, how to improve (or even maximize) the utility of the model while preserving privacy becomes an urgent problem. To resolve this problem, we apply unscented Kalman filter (UKF) to various implementations of differential privacy (DP)-enabled ML (DPML). We propose a UKF-based DP-enabled ML (UKF-DPML) framework that achieves higher model utility with the given privacy budget ϵ. An evaluation module is included in the framework to ensure a fair estimation of DPML models. We validate the effectiveness of this framework through mathematical reasoning, followed by empirical evaluation of various implementations of UKF-DPML and DPML respectively. In the evaluation, we measure the ability of withstanding real-world privacy attacks and providing accurate classification, thus assessing the privacy and utility of the model. We conduct a range of privacy budgets and implementations on three datasets, each of which provides the same mathematical privacy guarantees. By measuring the resistance of UKF-DPML and DPML models to membership and attribute inference attacks and their classification accuracy, we obtain that applying UKF to the aggregates perturbed by DP noises results in higher utility with the same privacy budget and the effect of improved utility is related to the stage where UKF is applied.

Keywords Differential privacy · Machine learning · Unscented Kalman filter · Privacy-utility reconcilement · Inference attacks

Editors: João Gama, Alípio Jorge, Salvador García.

✉ Ping Li
liping26@mail2.sysu.edu.cn

Extended author information available on the last page of the article

1 Introduction

As the pioneer of data analysis, machine learning (ML) has grown rapidly in the past few years and is widely used in data mining (Fu et al., 2018; Lee et al., 2022), computer vision (Li & Wu, 2022; Chen et al., 2016), electronic mail filtering (Wittel & Wu, 2004; Launchbury et al., 2014), credit card fraud detection (Fu et al., 2016; Roy et al., 2018), natural language processing (Makridis et al., 2022), etc. In these scenarios, users tend to generate huge amounts of data every minute. Data holders can send these data to a cloud service provider (CSP) to identify potential data models. These models may help to support decision making, improve business, provide value-added services (Ducange et al., 2018), predictive services, and recommendation services (Yin et al., 2019) to users, etc. In this context, many CSPs have launched Machine Learning as a Service (MLaaS) (Ribeiro et al., 2015), such as Google Prediction API (Mining & Fakir, 2016), Amazon ML (Herbrich, 2017), Microsoft Azure ML (Barnes, 2015), and BigML (Donaldson & Donaldson, 2012). The MLaaS provides data holders with automated solutions for ML-based data processing, model training, prediction services. ML practitioners can deploy applications on cloud platforms without building their own large-scale infrastructure and computing resources. Although it is simple and straightforward for MLaaS to collect data directly from the client, the increasing number of privacy attacks deplete ML models as well as violate the requirements of national regulations and laws (e.g., EU and USA regulations such as Regulation (2018) and (California, 2020), and network security law of CHN (Network, 2016)). This centralized training approach has gradually increased concerns about the privacy and security of personal data.

To protect the privacy of their models and improve their ability to generate ML and deep learning (DL) models with highly utility, CSPs are turning to theoretical frameworks that leverage privacy-preserving ML (PPML) techniques. There are numerous techniques such as differential privacy (Dwork et al., 2006; Dwork & Roth, 2014) (DP), homomorphic encryption (Gentry, 2009) and multi-party security (Fan et al., 2021). The combination of DP and ML yields the DP-enabled ML (DPML) models. However, DPML models usually face a tradeoff between utility and privacy. The loss of utility caused by the addition of DP noise, and the weakness of non-semantic security make many difficulties in the deployment of DPML: *how to* obtain higher utility while ensuring the privacy of the model, and *how to* fairly evaluate the overall performance of the model are pressing issues to be addressed.

Related work The study in Jayaraman and Evans (2019) combined DP and ML, and explored the impact of various differential privacy mechanisms on machine learning. However, they did not consider the improvement on the utility. A recent preliminary study in Zhao et al. (2020) investigated a method for evaluating the privacy and utility tradeoff of DPML models. Zhao et al. (2020) proposed a framework that can effectively evaluate the performance of DPML. In their work, they achieved differential privacy by injecting noise at each of the three basic stages of machine learning. And they applied the scheme to Neural Networks (NN) and Naïve Bayes (NB) methods for training. The results illustrated that there is an inflection point for the privacy and utility of DPML models making the two balanced. However, with the evaluation module of their proposed framework, they only verified out the existence of the inflection point and did not perform an optimization scheme for the utility improvement of the model. Kim and Lee (2021) proposed a hybrid scheme that combines machine learning models with unscented Kalman filter (UKF) for solving the problem of wind proximity forecasting in the aviation industry. They combined the ML methods with UKF to improve the fidelity of the trained model. The results showed that the

fidelity was improved after applying UKF. But they did not combine the filter with DPML methods. The application of UKF to the DP mechanism was considered by Wang et al. (2018). They investigated how to improve the accuracy of queries in DP-based data publishing while ensuring privacy. They proposed to improve the accuracy of data by applying UKF to filter the noise-perturbed aggregates. They worked on queries in each of the four datasets from the real world. In their experiment, for each group of queries, they set up two separate control groups with and without the application of UKF to pretreat the aggregates. By the results of the experiment, they found that UKF can improve the accuracy of data queries with the same privacy budget. Although they were not combined with machine learning, from their results we find the possibility of UKF to improve the precision of the aggregates after being processed by noise.

Inspired by Jayaraman and Evans (2019), Zhao et al. (2020), Kim and Lee (2021), Wang et al. (2018), we delve deeper into this issue and in this paper, we set out to investigate different ML/DL methods on various real-world datasets for: 1) *how to* guarantee higher utility with the same privacy budget and 2) *how to* fairly evaluate the overall performance of the final model.

Contributions We propose a UKF-based DP-enabled ML (UKF-DPML) framework that enables researchers to obtain a DPML model with higher utility for the same privacy budget. Our goal is to select the considerable performing method that yields higher prediction accuracy and ensuring reliable privacy preservation by studying the different stages where UKF-filtered DP noise can be applied. Equally important, we provide DPML practitioners with a higher profit-value method by training models with our framework.

We first prove the rationality of the proposed UKF-DPML framework through mathematical reasoning, and then verify its effectiveness through experiments. We apply the framework to various implementations of classical ML and DL methods (e.g., Logistic Regression and Neural Networks). Crucially, a standard evaluation module has been built into the framework to ensure fair evaluation of these DPML implementations. To evaluate the privacy of models, we measure their ability to withstand black-box privacy attacks. Those attacks might actually be launched in the real world. To evaluate the utility, we measure the extent to which the model provides accurate classification.

In particular, we investigate how utility and privacy of ML models are affected by UKF after adding DP noise at different stages of the ML pipeline: Stage (1) before the ML/DL training, by injecting noise to the input data. Stage (2) where DP noise is injected while model updates. Stage (3) by perturbing learned model parameters of the trained model. It is emphasized that for each set of experiments, we set up the DPML implementation without UKF, thus to verify whether the application of UKF guarantees a higher model utility with the same privacy. We evaluate each UKF-DPML and DPML implementation in a series of privacy budgets, with each instance providing the same mathematical privacy guarantees. We measure different metrics to evaluate the model performance described above: model and data privacy (membership inference attacks and attribute inference attacks) and model utility (classification accuracy).

We conduct experiments on a series of datasets from the real world, such as the CIFAR-100 (Krizhevsky & Hinton, 2009), Purchase-100 (Kaggle, 2014), and Netflix Prize-100 datasets (Kaggle, 2006). Through our experiments, we make the following observations. Most notably, applying UKF to the aggregates after DP perturbing results in higher model utility with the same privacy budget and the effect of improved utility is related to the stage where UKF is adopted. This observation is consistent across all DPML methods.

Summary of contributions Through our research, we provide insight into *how to* improve utilty while ensuring privacy of DPML models and *how to* fairly evaluate the overall performance of the model. We have summarized our contribution as follows:

- We propose a UKF-DPML framework that enables researchers to obtain a DPML model with higher utility with the same level of privacy protection. The framework also contains modules for model evaluation.
- We first prove the rationality of the proposed framework by mathematical reasoning, and verify its validity by experiments.
- Our experiments utilize three different attacks to measure the privacy of the model, which provides more accurate evaluation results.
- For each set of experiments, we set it up on a range of privacy budgets and various real-world datasets. Additionally, for each DMPL implementation, we set up controlled experiments with and without the application of UKF to observe changes in model performance after applying our proposed UKF-DPML framework.
- Through our experiments, the most notable finding is applying UKF to the aggregates after the DP perturbation yields higher model utility for the same privacy budget, and the utility improvement is related to the stage at which UKF is applied. This observation is consistent across all DPML methods.

Organization of the paper Section 2 describes the methodologies that we applied in our study. Section 3 provides the execution framework of our experiments. Section 4 presents the results of experiments. In Sects. 5 and 6, we present the discussions and directions for future work separately. Lastly, the conclusions are presented in Sect. 7.

2 Methodology

2.1 Overview

Our goal is to allow different ML/DL methods to improve (or even maximize) the utility while ensuring the same privacy budget. We propose a UKF-based DP-enabled (UKF-DPML) framework and verify the effectiveness through mathematics and experiments respectively. An instantiation of our framework is presented in Fig. 5. In our framework, machine learning is divided into three stages: 1) *data collection*, 2) *model training* and 3) *model finalization*. We also consider an evaluation module to measure the performance of each method equally. Additionally, there are two main key components included in the framework: differential privacy (DP) noise and the unscented Kalman filter (UKF), where DP noise is used to protect privacy and UKF is applied to filter the DP-perturbed aggregates. We apply these two components at each stage in this framework. For the first key component, the DP mechanism achieves privacy protection by injecting noise into the aggregates (Sect. 2.2). In our experiments, we choose noise that satisfies the ϵ−differential privacy and (ϵ, δ)-differential privacy. The parameter ϵ is used to manipulate the amount of noise added in different stages. For another key component, the UKF is an extension of the Kalman filter (Sect. 2.3), which is mainly applied to nonlinearly distributed data to improve the accuracy of the data by 1) the unscented transform, 2) predict and 3) correct (Sect. 2.4).

In Sect. 2.5.1, we describe the algorithm for injecting filtered-noise in three stages. In Sect. 2.5.2, we summarize the algorithms mentioned above and prove the rationality of

the framework by mathematical reasoning. Besides, a model evaluation module exists in our framework (Sect. 2.6). For each implementation of the DPML method, we evaluate its privacy and utility. Privacy is evaluated by quantifying the adversary advantage of multiple inferential attacks on the trained model (Sect. 2.6.1). As for utility, we measure it mainly by computing the accuracy loss of the model (Sect. 2.6.2).

Furthermore, we verify the validity of the framework through experiments. We first introduce our experimental framework (Sect. 3.1), including the ML/DL method used (Sect. 3.1.1), the parameter settings (Sects. 3.1.2 and 3.1.3), and the experimental steps (Sect. 3.1.4). We also give a detailed description of the three datasets used in the experiments (Sect. 3.2). In Sect. 4, we show the results of experiments, where Sect. 4.1 is for Logistic Regression and Sect. 4.2 is for Natural Networks. In Sect. 4.3, we give a summary of the findings from experiments and answer the three questions raised in Sect. 3.

Subsequently, we have some discussion in Sect. 5, describe our future work in Sect. 6, and summarize our work in Sect. 7. Next, we present the details of each building block of the proposed UKF-DPML framework in this section.

2.2 Differential privacy

Differential Privacy (DP) is a mechanism tailored to protecting publicly shared information in the dataset. With DP, the dataset can describe the patterns of groups and simultaneously reserve the information of a single datapoint in the dataset. There are numerous applications of DP existed in practice. DP can be used to protect the privacy of machine learning model according to Chaudhuri and Monteleoni (2008), Chaudhuri et al. (2011), Abadi et al. (2016), Shokri and Shmatikov (2015). DP can also be utilized in streaming data release (Dwork et al., 2010; Chan et al., 2011, 2012).

Dwork and Roth (2014) introduced the following denition: (ϵ, δ)-differential privacy:

Definition 2.1 ((ϵ, δ)-Differential Privacy, (ϵ, δ)-DP). Given two adjacent datasets D_1 and D_2 (denote as $\|D_1 \Delta D_2\|$=1), a mechanism f with (ϵ, δ)-DP can be defined if the outputs of the datasets are identical within a certain privacy budget ϵ. Mathematically, the algorithm f preserves (ϵ, δ)-DP if the following equation is satisfied:

$$\Pr[f(D_1) \in S] \leq e^{\epsilon} \times \Pr\left[f\left(D_2\right) \in S\right] + \delta,$$

where ϵ is the privacy budget and δ is the probability of failure, and $S \subseteq Range(f)$. $Range(f)$ represents the value range of the f. When δ=0, it becomes ϵ-differential privacy (ϵ-DP). We can also quantify the privacy loss by:

$$\ln \frac{\Pr[f(D_1) \in S]}{\Pr[f(D_2) \in S]}.$$

In (ϵ, δ)-DP and ϵ-DP mechanisms, the noise satisfies Gaussian (Gauss) and Laplace (Lap) distribution separately.

Composition The (ϵ, δ)-DP satisfies the linear composition property: for the same δ, given (ϵ_1, δ)-DP and (ϵ_2, δ)-DP performed on the same data, the privacy budget becomes $\epsilon_1 + \epsilon_2$ when two DP are applied simultaneously.

Relaxed Definitions As the general relaxation of ϵ-DP, in addition to (ϵ, δ)-DP, there is (α, ϵ)-differential privacy ((α, ϵ)-DP) (Mironov et al., 2019). Both of them are introduced

which allow for a small probability of failure and retain some useful properties like advanced composition and support for Gaussian mechanism. In (α, ϵ)-DP, α is the Rényi divergence of order, used to measure the difference between two probability distributions. Meanwhile, the privacy budget ϵ is used to control the tolerance of DP definition.

In this work, we apply pure ϵ-DP and (α, ϵ)-DP to DPML methods. ϵ-DP can be reduced from (ϵ, δ)-DP (Dwork & Roth, 2014) by taking δ=0 and (α, ϵ)-DP by taking α=∞.

2.3 Kalman filter

In the real world applications, measuring some physical variables is not practicable or not cost-effective and the Kalman filtering is brought up to solve this. Kalman filtering (KF) is an feasible algorithm that provides estimates of unknown variables by a series of measurements observed over time. Due to its compatibility of combining multiple source of measurements, it is likely to be more accurate than those based on a single measurement only. KF is widely adopted and gains notable utility in diverse fields like the signal processing in aerospace and tracking undersea sonar. The methodology and the model is similar to linear models of regression. Figure 1 illustrates the two states of the KF implementation process. Next, we discuss about the detailed information on the working process of KF.

Assuming that u_k depends on impalpable quantity r_k known as the nature state and u_k denotes the observed values of a variable at time k, the relationship can be specified by the observation equation:

$$u_k = H_k r_k + v_k, \tag{1}$$

where H_k is a known quantity and v_k is the observation error which has a normal distribution with a known variance R, denoted as $v_k \sim N(0, R)$.

The nature state (Meinhold & Singpurwalla, 1983) is similar to the regression coefficient in conventional linear model but it may change over time. So we assume that the nature state r_k at time k is transited from the state at previous time $(k-1)$ and the relationship is specified by the equation:

$$r_k = G_k r_{k-1} + w_k, \tag{2}$$

where G_k is a known quantity and the system equation error and w_k is a normal distribution with a known variance Q: $w_k \sim N(0, Q)$. In the real world, the relationship of the state of nature is predefined by scientific principles. In this way, the KF has the ability to containing the known information of the system behavior in the statistical model.

The KF works in a recursive procedure containing **predict step** and **correct step**. Note that in the following formulas, the superscript $'-'$ represents prior estimate and $'\hat{}'$ represents posteriori state estimate.

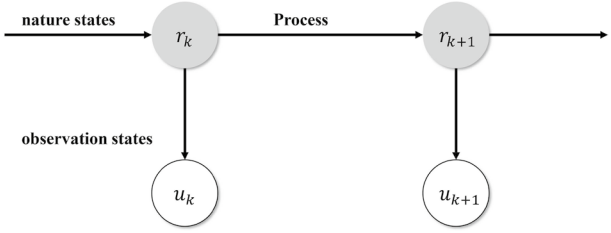


Fig. 1 The illustration of state-space model for KF

(1) **The predict step** occurs before the observation of data u_k. The priori state estimate $\hat{r}_k^-$ and the priori estimate covariance P_k^- can be calculated using the previous state estimate:

$$\hat{r}_k^- = \hat{r}_{k-1}^-,$$
$$P_k^- = P_{k-1}^- + Q.$$

(2) **The correct step** occurs after the observation of data u_k. Based on the previously updated estimate $\hat{r}_k^-$, the optimal Kalman gain K_k, the posteriori state estimate $\hat{r}_k$ and the posteriori estimate covariance P_k can be calculated via:

$$K_k = P_k^- \left(P_k^- + R\right)^{-1}, \tag{3}$$

$$\hat{r}_k = \hat{r}_k^- + K_k\left(u_k - \hat{x}_k^-\right), \tag{4}$$

$$P_k = \left(1 - K_k\right)P_k^-. \tag{5}$$

Based on this regular, KF only utilizes the previous state for calculating the current state so it shows a great performance on the real time problem. However, the linear transformation would not perform well on the nonlinear application system.

2.4 Unscented Kalman filter

The basic KF is useful in linear systems. However, it may not converge in nonlinear systems. Extended Kalman Filter (EKF) is a commonly used method to solve this problem, which applys mathematical techniques to the basic KF to make it usable in the non-linear systems. Nevertheless, the EKF has two demerits which are generally recognized: (1) the mathematical techniques applied can make filter highly unstable and (2) the EKF is difficult to implement due to its mathematical details. Hence, the Unscented Kalman Filter (UKF) was proposed (Julier & Uhlmann, 1997) as a viable solution to issues outlined above and it was proven to be more efficient, accurate and easier to implement (Julier & Uhlmann, 1997) in non-linear system.

In this work, we focus on applying UKF to differentially private machine learning in order to improve the accuracy of model after employing the DP noises. Different from KF, the process of UKF can be divided into three parts: (1) **unscented transform**, (2) **predict** and (3) **correct** according to the work (Julier & Uhlmann, 1997; Wan et al., 2001; Kandepu et al., 2008; Wan & Merwe, 2000). We discuss about how UKF works as follows.

(1) **The unscented transform** The unscented transform is a method for computing the statistical properties of random variables that satisfy a nonlinear function. First, it selects a set of sigma points that are consistent with the original mean and covariance. Then, the nonlinear function is applied to these sigma points to obtain a set of deformation points. The deformation points are used to approximate the original variables. In order to select the set of sigma points, Julier and Uhlmann (2002), Julier and Uhlmann (2004), Julier (2003) introduced a series of methods including the criterion to minimize the cost function when selecting.

In our application, we set up two states: the *initial state* x_k and the *noise state* z_k. The initial state x_k represents the k-th point in the original aggregates, and the noise state z_k

represents the result of adding DP-noise to the original x_k. The two states are described as follows:

$$x_k = F(x_{k-1}) + q_k, \tag{6}$$

$$z_k = x_k + p_k, \tag{7}$$

where the function F represents the nonlinear model fitted by the original aggregates, which is not necessarily continuous. q_k denotes the loss degree of fitting. p_k denotes DP-noise interference. q_k and p_k satisfy the normal distributions $q_k \sim N(0, Q)$, $p_k \sim N(0, R)$.

Consider an aggregates x of initial state is a vector of dimension L and possesses a mean $\bar{x}$ and covariance P_x. We approximate this original aggregates by constructing $2L + 1$ sigma vectors X_i with corresponding weight W_i. Figure 2 summarizes the key steps for sigma points selection,

where $\lambda = \nu^2(L + \kappa) - L$ is a scaling parameter. ν is a small positive value that affects the placement of the sigma points around $\bar{x}$. κ is the secondary scaling parameter. $\left(\sqrt{(L+\lambda)P_x}\right)_i$ is the i-th row of the matrix square root. β is used to integrate prior knowledge of the distribution of x. When i=0, the weights of the mean $W_i^{(m)}$ and covariance $W_i^{(c)}$ are different and when $1 \le i \le 2L$, the weights of them are the same at each point.

(2) **The predict step** In this step, we apply the initial state and the noise state equation to the above sigma point vectors respectively. Consequently, we obtain the corresponding deformation points $X'_{k|k-1}$ and $Z_{k|k-1}$ (including prior estimate mean and covariance). Figure 3 summarizes the key steps of the predict process in UKF,

where $X'_{k|k-1}$ represents the deformation point vector of X_k transforming from X_{k-1} through the function F and q_k. And $X'_{i,k|k-1}$ is the i-th point in $X'_{k|k-1}$ point vector. $Z_{k|k-1}$

Sigma Points Selection in Unscented Transform:

- **Initialize** the initial sigma point X_0 and its weights of mean $W_0^{(m)}$ and covariance $W_0^{(c)}$:

$$X_0 = \bar{x}, \qquad i = 0$$

$$W_0^{(m)} = \lambda/(L+\lambda), \qquad i = 0$$

$$W_0^{(c)} = \lambda/(L+\lambda) + \left(1 - \nu^2 + \beta\right), \quad i = 0.$$

- **Calculate** the remaining sigma point X_i:

$$X_i = \bar{x} + \left(\sqrt{(L+\lambda)P_x}\right)_i, \qquad i = 1, \cdots, L$$

$$X_i = \bar{x} - \left(\sqrt{(L+\lambda)P_x}\right)_i, \qquad i = L+1, \cdots, 2L.$$

- **Calculate** the corresponding weights of mean $W_i^{(m)}$ and convariance $W_i^{(c)}$:

$$W_i^{(m)} = W_i^{(c)} = 1/2(L+\lambda), \qquad i = 1, \cdots, 2L.$$

Fig. 2 The key steps for sigma points selection in unscented transform

Predict Process in UKF:

– **Calculate** deformation points:

$$X'_{k|k-1} = F\left(X_{k-1}\right) + q_k,$$

$$Z_{k|k-1} = X'_{k|k-1} + p_k.$$

– **Produce** the prior estimated mean:

$$\hat{x}_k^- = \sum_{i=0}^{2L} W_i^{(m)} \mathcal{X}'_{i,k|k-1},$$

$$\hat{z}_k^- = \sum_{i=0}^{2L} W_i^{(m)} Z_{i,k|k-1}.$$

– **Produce** the prior estimated covariance:

$$P_k^- = \sum_{i=0}^{2L} W_i^{(c)} \left[\mathcal{X}'_{i,k|k-1} - \hat{x}_k^-\right] \left[\mathcal{X}'_{i,k|k-1} - \hat{x}_k^-\right]^T + Q.$$

Fig. 3 The key steps of the predict process in UKF

represents the vector of applying the noise state equation to $X'_{k|k-1}$. Similarly, $Z_{i,k|k-1}$ is the i-th point in $Z_{k|k-1}$ vector. Additionally, $\hat{x}_k^-$ and P_k^- are the a priori estimated mean and covariance of the initial state respectively. $\hat{z}_k^-$ is the a priori estimated mean of the noise state.

(3) **The correct step**In this step, we combine the mean and covariance of the prior estimates in the initial and noise state to obtain the posterior estimate $\hat{x}_k$ and covariance P_k of

Correct Process in UKF:

– **Calculate** the Kalman gain $\mathcal{K}_k$:

$$P_{\tilde{z}_k \tilde{z}_k} = \sum_{i=0}^{2L} W_i^{(c)} \left[Z_{i,k|k-1} - \hat{z}_k^-\right] \left[Z_{i,k|k-1} - \hat{z}_k^-\right]^T + R,$$

$$P_{x_k z_k} = \sum_{i=0}^{2L} W_i^{(c)} \left[X_{i,k|k-1} - \hat{x}_k^-\right] \left[Z_{i,k|k-1} - \hat{z}_k^-\right]^T,$$

$$\mathcal{K}_k = P_{x_k z_k} P_{\tilde{z}_k \tilde{z}_k}^{-1}.$$

– **Produce** the posterior estimate $\hat{x}_k$:

$$\hat{x}_k = \hat{x}_k^- + \mathcal{K}_k \left(z_k - \hat{z}_k^-\right).$$

– **Produce** the posterior estimated covariance P_k:

$$P_k = P_k^- - \mathcal{K}_k P_{\tilde{z}_k \tilde{z}_k} \mathcal{K}_k^T.$$

Fig. 4 The key steps of the correct process in UKF

the noise-perturbed z_k filtered by UKF. Figure 4 summarizes the key steps of the correct process in UKF.

In Fig. 4, $P_{\tilde{z}_k\tilde{z}_k}$ represents the noise covariance ($\tilde{z}_k = z_k - \hat{z}_k^-$), and $P_{x_k z_k}$ represents the initial-noise cross state covariance. $X_{i,k|k-1}$ is the i-th point in $X_{k|k-1}$ point vector. The Kalman gain K_k is determined by the two covariances.

Hence, it is clear that the UKF-filtered posterior estimate $\hat{x}_k$ is obtained by combining the prior estimated mean ($\hat{x}_k^-$ and $\hat{z}_k^-$), the Kalman gain (K_k), and the noise-perturbed result (z_k).

2.5 UKF-based DP-enabled ML framework

In this section, we propose a UKF-based DP-enabled ML (UKF-DPML) framework to improve the utility of the trained model without depleting privacy. The framework mainly consists of two key components to achieve this goal: DP noise and UKF.

To ensure the privacy of the training data and the trained model, we employ the noise mechanism that satisfies ϵ-DP and (ϵ, δ)-DP respectively. In this paper, we focus on the Laplacian and Guassian noise mechanism, which achieves ϵ-DP and (ϵ, δ)-DP by injecting random noise satisfying the Laplacian and Guassian distribution respectively into the original aggregates. Next, we process the noise-perturbed aggregates with UKF to improve the availability of perturbed data, thereby increasing the usefulness of differentially private ML models.

This UKF-DPML framework is depicted in Fig. 5. We describe this framework in detail below.

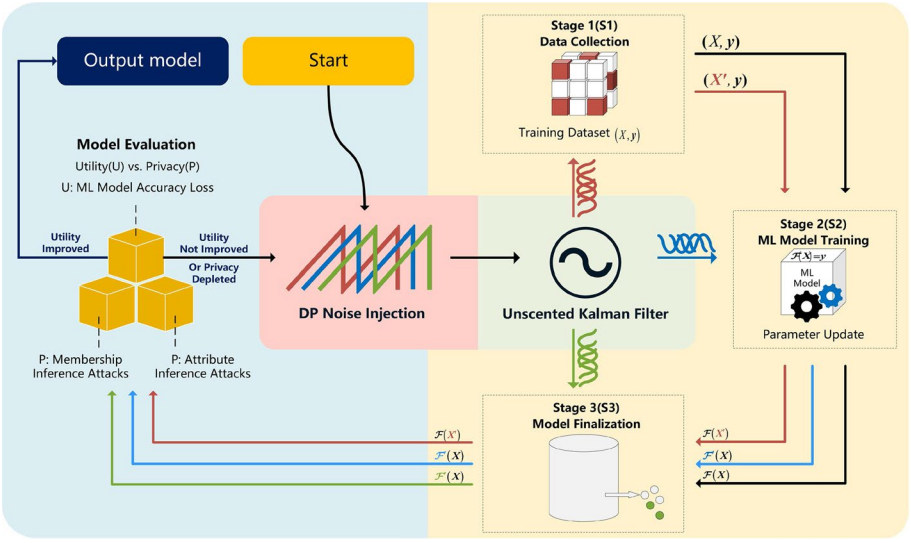


Fig. 5 Our instantiation of the proposed framework, with the three basic stages that DP noise can be introduced in the ML pipeline to guarantee data privacy and UKF can be adopted to improve model utility, and performance metrics used to assess utility-privacy tradeoff

2.5.1 ML pipeline stages for DP noise injection

In our framework, there are three basic stages: (1) data collection, (2) model training and (3) model finalization, where DP noise can be added in machine learning as we are inspired by the work in Zhao et al. (2020), Jayaraman and Evans (2019). These three basic stages are shown in Fig. 5.

The goal of machine learning is to be able to optimally understand the relationship between the data and the labels, and to classify the data into the labels with the highest degree of conformity. To understand our framework more concrete, we let the function F implement the mapping of the training dataset X to the corresponding label Y i.e., $F(X) = Y$. In the following, we discuss these three stages of DP-noise addition.

Stage 1 (S1): Data collection In the data collection stage, we can apply DP noise to this aggregates $data(X)$ before model training. Each record in the original dataset $data(X)$ utilizes the DP noise-perturbed dataset $data\left(X'\right)$ to protect the sensitive information in the original dataset. Therefore, when we need to publicly release data containing sensitive information, we can train a synthetic dataset with the same distribution and meaning as the real dataset. However, DP noise interference can be utilized to protect the privacy of the original data. Applying this synthetic dataset to the machine learning trained model $F(X')$ is said to be DP-enabled.

Stage 2 (S2): Model training In this stage, we achieve privacy preservation by applying DP noise to the parameter updates of the model gradient training process. Note that in this stage, each step of parameter update in each generation of model training is restricted. In this way, the added DP noise ($F'(X)$) is not excessive and changes the model which can compromise the privacy of the training data set for that batch. A typical implementation of DP for machine learning at this stage is the *Tensorflow-Privacy's* library developed by *tensorflow.org* Google (2019). This library implements the DP noise stochastic gradient descent algorithm for ML Bottou (2010).

Stage 3 (S3): Model finalization When completing the machine learning training output model using raw aggregates, we can apply DP noise to the trained model parameter distribution ($F'(X)$). We reduce the correlation between the model and the training dataset to lower the privacy leakage risk of attribute and membership inference attacks on the trained model, thus achieving the purpose of protecting data privacy and DP.

2.5.2 UKF-based DP-enabled machine learning algorithms

To enable DPML methods to improve the utility of the trained model without depleting privacy, we process the noise-perturbed aggregates using UKF. The availability of the perturbed data is improved because the aggregates derive a posteriori estimates of the original aggregates. The processed results are then applied to ML model training. Ultimately, the utility of the model is evaluated by accuracy loss (Zhao et al., 2020) and the privacy of the model by membership inference (MI) (Shokri et al., 2017) and attribute inference (AI) attacks (Fredrikson et al., 2017). If the utility of the model does not improve or the privacy preserving effect decreases, it is returned to noise addition for reprocessing.

According to the description in Sect. 2.4, we can learn that UKF consists of three key components (1) the unscented transform, (2) predict and (3) correct. Based on processing description in Figs. 2, 3, 4, the execution of predict function and correct function is described in Algorithm 1 UKFDPMLPredict and Algorithm 2 UKFDPMLCorrect.

Algorithm 1 UKFDPMLPredict(k)

Input: Point set X_{k-1}

Output: $\left\{\hat{x}_k^-, P_k^-\right\}$, the priori estimate and covariance

$\hat{x}_k^- = \sum_{i=0}^{2L} W_i^{(m)} X'_{i,k|k-1}$

$P_k^- = \sum_{i=0}^{2L} W_i^{(c)} \left[X'_{i,k|k-1} - \hat{x}_k^-\right] \left[X'_{i,k|k-1} - \hat{x}_k^-\right]^T + Q$

return $\left\{\hat{x}_k^-, P_k^-\right\}$

Algorithm 2 UKFDPMLCorrect(k)

Input: Priori estimate $\hat{x}_k^-$ and measurement z_k

Output: Posteriori estimate $\hat{x}_k$

$K_k = P_{x_k z_k} P_{\tilde{z}_k \tilde{z}_k}^{-1}$

$\hat{x}_k = \hat{x}_k^- + K_k \left(z_k - \hat{z}_k^-\right)$

$P_k = P_k^- - K_k P_{\tilde{z}_k \tilde{z}_k} {K_k}^T$

return $\{\hat{x}_k\}$

Subsequently, we apply the UKF to two machine learning algorithms of interest: LR and NN, thereby validating the effectiveness of the framework. Table 1 describes the methods used to learn the data in a supervised environment, the stages of applying DP noise and the use of the UKF filter. In the following, we describe the relevant algorithms involved in these three stages.

(1) ***S1: Directly inject Laplacian noise into aggregates with UKF adopted*** In this stage, we apply Laplacian noise directly to the nonlinearly distributed dataset, then let the noise-perturbed dataset be filtered by UKF, and lastly apply the filtered aggregates to the DPML model for training. Thus, the process is independent of the DPML model used in the subsequent steps of the framework. Due to this independence, we can apply this method to both UKF-DPML and DPML methods so that they generate two different types of datasets: the DP- disturbed dataset and the control set of the original dataset without privacy protection. Specifically, DP in this stage is achieved by adding noise to each vector in the dataset. When applying DP noise, we assume that the features of the dataset are independent of each other, which means that maximum noise must be applied to each feature to ensure the effect of differential privacy is achieved. The Laplacian noise obeying $\text{Lap}\left(0, \beta_i\right)$ is sampled independently for each eigenvalue of each data vector and then processed using UKF filtering, where $\beta_i = \frac{S_i}{\epsilon/n}$ and S_i is the range of values of the i-th eigenvalue (Das et al., 2016) (Algorithm 3)

Table 1 UKFDP-enabled ML methods used in each stage

ML method	Stage where DP noise is applied			UKF
	S1	S2	S3	Adopted
Logistic regression	×		×	✓
Neural network	×	×		✓

Algorithm 3 Direct injection of DP noise to dataset with UKF adopted before training

Input: Training set $\{x_k, k = 1, \cdots, n\}$ with $length = n$
Output: $\{t_k, k{=}1, \cdots, n\}$, the dataset disturbed by UKF-filtered DP noise.

```
for k=1, ..., n do
    y_k = x_k + b; where b: = {b_0, ..., b_i, ..., b_n}; b_i ← Lap(0, β_i) in S1
    priori ← UKFDPMLPredict(k)
    posteriori ← UKFDPMLCorrect(k)
    t_k ←posterior
end for
Proceed with learning task F on {t_k}
```

The inclusion of DP noise in S1 is independent and more flexible, since any ML or deep learning (DL) algorithm can be trained after S1. The application of noise depends on the type of data and the complexity of its dataset, and the application of the filter depends on the distribution characteristics of the dataset. UKF is suitable for use on non-linearly distributed datasets to achieve the effect of ensuring privacy while improving the utility of the trained model.

(2) ***S2: DP-enabled Neural Networks with UKF adopted*** Neural Networks (NN) is at the core of DL, which is inspired by the human brain and mimic the way biological neurons transmit signals (Gurney, 2018). It consists of layers of nodes, mainly three kinds of layers: one input, one or more hidden, and one output. Each node is called an artificial neuron, and they are connected to another node with associated weights. If the output of any individual node exceeds a threshold, then the node is activated and sends data to the next layer of the network. Otherwise, no data is passed to the next layer of the network.

Tensorflow-Privacy (Google, 2019) implements a differentially private neural network algorithm based on stochastic gradient descent (DPSGD) according to the work (Abadi et al., 2016). DPSGD algorithm attempts to find a network parameter θ to learn the function F. This algorithm limits the size of the gradient update during the first clipping to avoid adding excessive noise and changing the model, thus compromising the privacy of the batch training dataset. This algorithm achieves differential privacy by adding noise to the updated parameters based on the privacy parameters and batch sensitivity. DPSGD employs the noise obeying Guass(0, $\phi^2 C^2$) to satisfy (δ, ϵ)-DP, where ϕ is the noise scale ($\phi = \sqrt{2 \log \frac{1.25}{\delta}}/\epsilon$) and C is the gradient norm bound (Abadi et al., 2016). We adapt this algorithm to include a UKF filtering process to filter the parameters after adding noise (Algorithm 4).

Algorithm 4 DP-based Stochastic Gradient Decent with UKF adopted

Input: Training set $\{x_k, k = 1, \cdots, n\}$ with $length = n$ and $batch = T'$
Output: Parameters θ' perturbed by UKF-filtered DP-noise

$\theta' \leftarrow$ RAND, initialize the parameters randomly
for batch $t \in T'$ **do**
  compute gradient $\Delta\theta$, clip gradients $\Delta\theta$, add *DP-noise* b, $b \leftarrow$ Guass$(0, \phi^2 C^2)$
  $\theta' = \theta' + \Delta\theta + b$
  priori $\leftarrow$UKFDPMLPredict(t)
  posteriori $\leftarrow$UKFDPMLCorrect(t)
  $\theta' \leftarrow$ posterior
end for
Complete F' learning task with θ'

(3) S3: DP-enabled Logistic Regression with UKF adopted Logistic Regression (LR) is used to deal with regression problems in which the dependent variable is a categorical variable, commonly dichotomous or binomial distribution problems. LR can also deal with multicategorical problems, which actually belongs to a classification method. It achieves the purpose of dichotomizing the data by applying gradient descent to solve the parameters through the method of maximizing the likelihood function (Chaudhuri & Monteleoni, 2008).

IBM LR (Holohan et al., 2019) implements a logistic regression algorithm that satisfies DP. This method adds noise to the learning distribution associated with the input eigenvalues and output decisions to achieve DP. We have adapted the algorithm to include UKF. Algorithm 6 describes the process of applying UKF to the mean and standard deviation distributions (μ, σ) computed from the training dataset *data*(X).

Algorithm 5 DP-based Logistic Regression provided by IBM with UKF adopted

Input: Training set $\{x_k, k = 1, \cdots, n\}$ with $length = n$
Output: (μ', σ'), the model distribution parameters perturbed by UKF-filtered DP noise

Compute (μ, σ) from $\{x_k, k = 1, \cdots, n\}$
for i in features **do**
  compute scaling factor $S_{(\mu,i)}$ and $S_{(\sigma,i)}$ from feature mean μ_σ, feature STD σ_i, and ϵ
  $\mu_i' = \mu_i + b_i$; where $b_i \leftarrow \text{Lap}\left(0, S_{(\mu,i)}\right)$
  priori $\leftarrow$ UKFDPMLPredict(i)
  posteriori $\leftarrow$ UKFDPMLCorrect(i)
  $\mu_i' \leftarrow$ posterior
  $\sigma_i' = \sigma_i + b_i$; where $b_i \leftarrow \text{Lap}\left(0, S_{(\sigma,i)}\right)$
  priori$'$ $\leftarrow$ UKFDPMLPredict(i)
  posteriori$'$ $\leftarrow$ UKFDPMLCorrect(i)
  $\sigma_i' \leftarrow$ posteriori $'$
end for
Compute output priors $P(y \mid x)$ from (μ', σ')

2.5.3 Scheme overview

We apply UKF to three stages of DPML for improving the usability of the aggregates after being disturbed by noise. In these three stages, we summarize the execution process of this filter as expressed in Algorithm 6 UKFDPML.

Algorithm 6 UKFDPML (k)

Input: Original aggregates $\{x_k, k = 1, \cdots, n\}$, privacy budget ϵ, length of aggregates $= n$
Output: $\{g_k, k = 1, \cdots, n\}$, the aggregates of UKF processed.

```
for k=1, ..., n do
    y_k = x_k + b; where b := {b_0, ..., b_i, ..., b_n};
    b_i ← Lap(0, β_i) in S1 or
    b_i ← Guass(0, φ^2 C^2) in S2 or
    b_i ← Lap(0, S_(u,i) or S_(σ,i)) in S3
    priori ← UKFDPMLPredict(k)
    posteriori ← UKFDPMLCorrect(k)
    g_k ← posterior
end for
return {g_k}
```

The input of the UKFDPML algorithm contains a set of n-dimensional vectors $\{x_k, k =1, \cdots, n\}$, which can represent one of the following three cases according to the three basic positions injected by DP noise: (1) S1: the machine learning training dataset $data(X')$ perturbed by DP noise, (2) S2: the model parameter set $\{\theta\}$ after DP noise injected in the gradient training process or (3) S3: the set of mean and standard deviation distributions (μ, σ) associated with the output decision and perturbed by DP noise. n also represents the length of the dataset. The output aggregates $\{g_k, k=1, \cdots, n\}$ then represents (1) S1: the $data(X'')$ of the machine learning training dataset after being perturbed by noise and filtered with UKF, (2) S2: the set of model parameters $\{\theta'\}$ of the gradient training process with UKF adopted and (3) S3: the set of mean and standard deviation distributions (μ', σ') related to output decision and filtered with UKF after noise interference.

Afterwards, we apply the filtered set $\{g_k, k=1, \cdots, n\}$ to the machine learning model to complete the training. Lastly, evaluate the utility and privacy of the trained model. If the utility does not improve or the privacy preserving effect decreases, it is returned to noise addition and reprocessed. The algorithm 6 summarizes the main execution process of the UKF-DPML framework, which is satisfying ϵ-DP, and we give the following theorem.

Theorem 1 *The UKFDPML algorithm satisfies ϵ-differential privacy.*

Proof The function $f : D \to R^n$, represents the mapping of a aggregate D to an n-dimensional space R, if

$$f(D) = \left(x_1, x_2, \cdots, x_n\right)^T, \tag{8}$$

$$
\begin{aligned}
f\left(D'\right) &= \left(x_1', x_2', \cdots, x_n'\right)^T \\
&= \left(x_1 + \Delta x_1, x_2 + \Delta x_2, \cdots, x_n + \Delta x_n\right)^T,
\end{aligned} \tag{9}
$$

where $\left(\Delta x_1, \Delta x_2, \cdots, \Delta x_n\right)$ denotes the noise interference term with UKF applied. Then exists:

$$
\begin{aligned}
\Delta f &= \max\left(\sum_{i=1}^{n} \left\|x_i - x_i'\right\|\right) \\
&= \max\left(\sum_{i=1}^{n} \left\|\Delta x_i\right\|\right)
\end{aligned}
$$

Let x_i=0, $A(D)$=$(0, 0, \cdots, 0)^T$, $A(D') = (\Delta x_1, \Delta x_2, \cdots, \Delta x_n)^T$, $O = (y_1, y_2, \cdots, y_n)^T$, and we have:

$$
\begin{aligned}
\frac{\Pr[A(D) = O]}{\Pr[A(D') = O]} &= \frac{\prod_{i=1}^{n} \frac{\epsilon_k}{2\Delta f} e^{\frac{-\epsilon_k}{\Delta f}\|y_i\|}}{\prod_{i=1}^{n} \frac{\epsilon_k}{\Delta f} e^{\frac{-\epsilon_k}{\Delta f}\|\Delta x_i - y_i\|}} \\
&= \prod_{i=1}^{n} e^{\frac{-\epsilon_k}{\Delta f}\|y_i\| - \|\Delta x_i - y_i\|} \\
&= e^{\frac{\epsilon_k}{\Delta f}\sum_{i=1}^{n}(-\|y_i\| + \|\Delta x_i - y_i\|)} \\
&\leq e^{\frac{\epsilon_k}{\Delta f}\sum_{i=1}^{n}(\|\Delta x_i\|)} \\
&\leq e^{\epsilon_k}.
\end{aligned}
$$

□

From the derivation, it can be seen that differential privacy protection is satisfied for each x_i vector with a privacy budget of $\epsilon_k = \epsilon/n$. Since differential privacy satisfies the combinatorial property, the UKFDPML algorithm is satisfied with ϵ-differential privacy. Note that the subalgorithms UKFDPMLPredict and UKFDPMLCorrect do not affect each other with the original aggregates, and therefore, do not consume any privacy budget.

Above, we proved the theoretical validity of the framework mathematically, and next we will verify the validity of this framework again experimentally in Sect. 4.

2.6 Model evaluation

There are two facets of measuring the performance of DP-enabled ML models—privacy and utility. In this section, we introduce why utilizing these factors and how these indicators are indirectly measured.

In our framework, the model evaluation module can be extended to other considerations in privacy-utility tradeoff analysis. This could include Resource metrics (e.g., computational resources required to train ML models) and the various dataset features used.

2.6.1 Privacy attacks & privacy metrics

Privacy has been previously measured with metrics like information leakage (Issa et al., 2016) and mutual information (Wang et al., 2016). Therefore, the membership inference (MI) (Salem et al., 2018; Yeom et al., 2018) and the attribute inference (AI) (Yeom et al., 2018; Zhao et al., 2019) is gradually becoming more popular in terms of the measurement of privacy leaks in ML models. Those attacks are performed in the condition that the attackers can only get access to the input and the output of the model. The MI attacks require only one query while the AI attacks need numerous queries. In this section, we survey on several MI and AI attacks and we use Yeom MI, Salem and Yeom AI attacks in the experiment part.

Membership inference attacks The purpose of the membership inference (MI) attack is to infer whether a given record is used to training the model. Thus, this kind of attacks can cause the leakage of sensitive information from training data. The attack target is related to the differential privacy's definition. According to DP, the difference between datasets applied with or without noise needs to be maintained under ϵ. Obviously, it is unrealistic for those who train ML model to pursue absolute confidentiality of dataset used. In this work, we discuss about three MI attack implementations (Salem et al., 2018; Shokri et al., 2017; Yeom et al., 2018).

- *ShokriMI Attack* (Shokri et al., 2017). A kind of membership inference attack which utilises confidence scores as the query results from the target model was proposed by Shokri et al. The method builds up the shadow models on the labelled dataset by postulating the underlying distribution of training set or querying to the target model. Then the shadow models are used to train the attack model to distinguish if the given input is present in the shadow model. The final step is to make Application Programming Interface (API) requests to the target model. Confidence scores grabbed from the requests can be used to infer whether the input is in the training set of the target model for every input record. In this case, the attack model does not recognize the data by the confidence scores. The key task is to distinguish among the training inputs from non-training inputs. Both of them are classified with high confidence scores.
- *SalemMI attack*. Recently, Salem et al. (2018) proposed more generic membership inference attacks by relaxing the assumptions made by Shokri et al. In their works, they proposed two ways of implementing the membership inference attacks. One is the need of building the same structure of model as the target model which can be achieved by using the same Machine Learning as a Service (MLaaS). Another is the dataset used to train the shadow models which should have the same distribution as the target model's training dataset.
- *YeomMI attack*. Yeom et al. (2018) proposed another more efficient membership inference attack method. With less computation it can also retain almost the same performance as previously released attack methods. Their results also shows that overfitting is a sufficient condition for membership vulnerability and it is commonly a satisfied condition in practice. Yeom et al. also proved that a stable training algorithm, which is not overfitting, can be undermined. And it results that the model may have the same membership information provided by the black box behavior.

Attribute inference attacks Compared to the MI attack, the attribute inference (AI) attack is used to infer highly sensitive attributes by the rest of the non-sensitive attributes. Given

the attribute with the dimensionality of n, the adversaries first utilize $n-1$ ground truth attributes and the output of the API requests from the MLaaS. Subsequently, combining the priori probability of features, they can calculate the most feasible value of x_n. However, the result is based on the assumption that there is certain correlation between the output of the model and the sensitive attribute we want to infer.

- *YeomAI Attack* (Yeom et al., 2018). The first method of AI attack was proposed by Yeom et al. (2018). The author characterized the concept of attribute advantage which is the ability to infer the target feature from an incomplete datapoint from training data. They also found that the influence (Wu et al., 2016) has impact on the privacy leaks caused by overfitting. As the influence increases, the attacker's ability to learning training data will decline no matter how large the generalization error is. The main idea of the method is to measure the difference between the accuracy of the training dataset and test dataset. Afterwards, the attacks is enable to capture the attribute advantage maximizing the performance of attack. In the ideal circumstance, the attack fits well on the training data.
- *SalemAI Attack* (Shokri et al., 2017). The second method of attribute inference attack was proposed by Salem et al. It uses possible permutations of feature expression and select the permutation which is the closest to the ground truth attributes.

Other attacks There are other kinds of attacks which are trying to infer specific information from the target model. Carlini et al. (2019) proposed the memorization attacks to manipulate the models with great capacity in order to memorize sensitive information in the traning data. These attacks posed a litte threat facing with some differential privacy mechanisms. Model stealing is another form of privacy attacks aiming to inferring the model parameters inside the black-box model by adversarial learning (Lowd & Meek, 2005) and equation solving attacks (Tramèr et al., 2016). The hyperparameters targeted attack is aiming to recover the hyperparameters that the model used in training process (Yeom et al., 2018). The hyperparameters are the essential part in the machine learning model thus leaking them will bring financial loss to commercial organization. The property inference attack is a method to infer whether the training dataset has specific property through a white-box access. These attack method were successfully performed on HMM, SVM models and the Neural Networks (Ateniese et al., 2015; Ganju et al., 2018).

Measuring privacy leaks We employ adversary advantage to measure privacy leaks of the trained model. The adversary advantage of privacy attacks is the observed improvement gained by obtaining copies in the training dataset.

The true positive rate (TPR) is the rate of input that are actually positive and are predicted positive. The false positive rate (FPR) is the rate of negative input which are predicted positive. Those are used as the metrics to indicate the success rate and failure rate of privacy attack respectively.

Definition 2.2 (*Adversary advantage*). As what is provided in Yeom et al. (2018), the advantage can be mathematically defined as:

$$ADV = TPR - FPR.$$

In the experimental part of the paper, we utilize ADV to measure the impact of MI and AI attacks.

Although the attack methods outlined above may cause privacy leak, these kinds of leak are likely to be available in specific application field and there is no clear and general definition of them. For instance, the property inference attacks will obtain some statistical information from the training data. However, there is no general definition about what statistical properties could be tolerant of being fetched and be sensitive to data leaks. Additionally, the mechanisms mentioned in this section is not closely conforming the threat model of differential privacy. Therefore, we take inference attacks into consideration in the experiment part and also ignoring those attacks which need white-box access to the target model.

2.6.2 ML utility metrics

Machine Learning (ML) is to learn the pattern from training datasets and gives prediction of unseen inputs. In practice, we split the existing dataset which has ground truth labels into two parts, one for training and one for testing. After training the model, we can apply the model to the testing part of the dataset and compare the output from the model with the ground truth labels. The proportion of successful predict is called accuracy (ACC): $ACC = n_{correct}/n_{holdout}$.

We use Accuracy Loss (ACL) to indicate the metrics of performance which is the ratio of performance loss when DP is employed to the three stages in ML process ($S_m, m \in \{1, 2, 3\}$). The ACL compares the model with DP applied to an equivalent ML model trained without DP applied ($\epsilon = \infty$)

$$ACL = 1 - \frac{ACC_{(S_m,\epsilon)}}{ACC_{(S_m,\epsilon=\infty)}}. \tag{10}$$

3 Experimental investigation

In this section, we describe how to instantiate the framework proposed in Sect. 2.5 and how to experimentally verify the effectiveness of the framework. We design a set of experimental protocols to further validate the framework based on the theoretical proof of its reasonableness (Theorem 1). Through our experiments, we would like to resolve the following questions: (1) **Q1:** questions about UKF performance improvement, (2) **Q2:** questions about the tradeoff inflection points between privacy and utility of the model and (3) **Q3:** questions about the UKF thresholds.

Q1: Can UKF-DPML framework improve the utility of the model without compromising privacy compared to a pure differentially private machine learning model? Is the effectiveness of the improved model utility after applying this framework related to the stage of UKF action? Does applying UKF at different stages of the ML training process have different effects on utility improvement?

Q2: Is there a tradeoff inflection point between the utility and privacy of DPML models? Is there a correlation between the effectiveness of UKF and this point?

Q3: Is there a threshold for improving the utility of the model after applying this framework? Will filtering again after this threshold cause a significant change in the privacy of the model?

We attempt to empirically identify the important parameters that affect this privacy-utility relationship of the model during the experiment based on previous studies (Zhao et al., 2020; Jayaraman & Evans, 2019). Through the analysis of our experimental results, we answer the above questions in Sect. 4.3.

Note that we employ MI and AI attacks to measure privacy leakage. The conclusions we draw from such experiments are limited to represent lower bounds on privacy leakage, as they are measuring the effectiveness of a particular attack. Although the privacy attacks in reality also exist with auxiliary information to aid the attack, the results of such experiments cannot be used to make strong judgments about the best possible attacks. Nevertheless, our experimental results do provide clear evidence to validate the effectiveness of our framework.

3.1 Experimental framework

In this section, we describe the implementation of DPML used during the experiments, and the metrics for evaluating the privacy and utility of the model after applying the UKF-DPML framework. We then outline our experimental steps and detail the datasets used in the experiments. Our experiments extend the study of Zhao et al. (2020), to which we add the following new elements.

- Add two key components of our framework: the DP noise and UKF.
- Add new machine learning algorithms applied to our framework.
- Adjust the code to fit the execution flow of our framework.
- Add the implementation of the MI attack proposed by Yeom et al. (2018)
- Add the implementation of two AI attack proposed by Liu et al. (2021) and Yeom et al. (2018)
- Add comparison experiments to compare the evaluation results of whether or not to apply the framework to train models and verify the effectiveness of the framework.

3.1.1 Machine& deep learning methods

We apply this framework to the two ML algorithms mentioned in Sect. 2.5.1: LR (Sect. 4.1) and NN for non-convex learning (Sect. 4.2). The code provided by Tensorflow-Privacy Abadi et al. (2016) can be found in Google (2019). IBM LR Chaudhuri and Monteleoni (2008) provides a differential privacy implementation of the logistic regression algorithm implementation, the code of which can be found in Holohan et al. (2019). The research in Chaudhuri and Monteleoni (2008) describes more complete algorithms for handling binary classification and multi-classification problems through LR method. The hyper-parameters of the NN models were replicated from Jayaraman and Evans (2019). All other models' parameters are consistent with library defaults.

3.1.2 UKF parameters

We adapt the algorithm mentioned in Section 3.1.1 to include the UKF. In our experiments, we used the FilterPy library FilterPy (2020) to implement the functions related to UKF. We set the default parameters of the UKF as shown in Table 2.

Note that the distribution variance R of noise state z is related to the distribution rate of noise in DPML models, so in our experiments we let R be equal to the distribution regular

with added noise, i.e., $R = \beta_i$ and $\phi^2 C^2$ (mentioned in Algorithm 3 and 4 respectively). In the experiments for the first data x_0, we replace it with the noise-perturbed z_0 to protect the privacy of the original data. In addition, the parameter values need to be modified according to the applied datasets and models.

3.1.3 Performance metrics & privacy budget

We employ different ML models to learn on different datasets and measure different performance metrics on the trained models. For the prediction performance of the training models, we evaluate it by Accuracy Loss (ACL) (Sect. 2.6.2). For the privacy of the models, we quantify the degree of privacy leakage by performing three MI and AI attacks, which are YeomMI, SalemAI and YeomAI (Sect. 2.6.1). We vary the noise applied in each stage of the framework and in each ML model by setting different privacy budgets as follows,

$$\epsilon \in \{0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100, 500, 1000\}.$$

3.1.4 Experimental steps

Our experiments are split into four parts: (1) select the dataset, (2) apply the UKF-DPML and DPML model, (3) select the privacy budget ϵ, (4) evaluate the models and compare the evaluation results of the models with and without applying UKF. We first draw two sets of samples from the dataset to form the training set and the test set in our experiments. Both two set are composed of 10,000 samples each. Then, we use the training set to train the ML model.

For each ML algorithm trained on each dataset, we have two sets of control experiments: the DPML model with UKF applied according to this UKF-DPML framework and the DPML model without UKF applied according to the methods in Zhao et al. (2020).

Note that in the case of adding DP noise in S1, we are applying noise to the training set before model training. After training, we evaluate the model performance separately, so as to verify whether the present framework can guarantee the improved utility of the model without depleting privacy.

In particular, we evaluate the degree of privacy compromise of the model by executing YeomMI, SalemAI and YeomAI attacks. In the MI attacks, the training set constitutes the membership set, and the test set is the non-membership test set. In the AI attacks, we randomly select 20 different attributes as protected attributes for each attack. Then repeat the entire training and attack process 10 times for each data set. Consequently, the training and test sets are resampled to reduce the effect of bias generated by data or DP noise. We

Table 2 Experiment default value of UKF parameters

Parameter	Implication	Default value
Q	Distribution variance	1000
R	Distribution variance	100
β	Variable distribution	2
ν	Positive value	0.001
κ	Secondary scale	0.001

randomly choose the attributes to attack the DPML models. For the UKF-DPML models, we also choose the same to ensuring the consistency of the attack experiment.

Remark 1 Since we re-adopt each training/testing set, a different membership and non-membership set is sampled each time, which results in a random outcome for each AI or MI attack. Furthermore, there is inherent randomness in the training of each model, and also, DP noise is resampled each time. There are many sources of randomness in this process. Therefore, in order to reduce the impact of this randomness, the results we report in this paper are repeated on average 10 times. Consequently, we can attempt to capture both high and low performance instances of this attack. We also show the degree of variation in these results (error bars on each data point) in Sects. 4.1 and 4.2. We find that even in our results, it is still different (even moreso when the dominance is low). In ideal world, we would like to perform more replications to obtain more accurate averages.

3.2 Experimental datasets

We conduct machine learning training to verify the effectiveness of the framework using three real datasets which can be accessed from Kaggle (https://www.kaggle.com). The size and attributes of these three datasets are summarized in the Table 3. The distribution of the categories of partial elements within the three data sets are shown in Fig. 6, which also represents the distribution of the overall data. As we can be seen from that, the distributions of the elements are all nonlinear and therefore suitable for processing using the UKF in the framework.

In the following, we give the detailed information about these three datasets.

CIFAR-100 Krizhevsky and Hinton (2009): The dataset is composed of 50,000 images of different objects, which are classified into 100 classes. We preprocess this dataset with Principal Component Analysis to extract 50 key features to represent each image Jayaraman and Evans (2019).

Purchase-100 Kaggle (2014): The Purchase dataset contains the purchases of 200,000 users, each of whom shopped for the same 599 products. Each element in the dataset is represented by a binary value indicating whether or not one of the 599 products was purchased. 1 represents a purchase and 0 represents none. For the preprocessing of this dataset, we use an approach consistent with that of the literature Shokri et al. (2017), where the transaction records of each user in the dataset are coded as a binary vector. The users are then clustered into groups of purchasers by means of clustering. Since our aim is to verify the validity of the proposed framework, we ensure that the category complexity of each dataset is equal. The elements of this Purchase dataset are also clustered into 100 categories.

Netflix Prize-100 Kaggle (2006): Netflix first released the dataset in 2006, which contains viewers' ratings for movies watched on the Netflix platform. It uses a scale of 1 to 5 to indicate high or low viewership where 1 represents the lowest viewership and 5 represents the highest viewership. This dataset is also used in the work Zhao et al. (2020) and Yeom et al. (2018) for ML training. We use the same approach as Zhao et al. (2020) to preprocess this dataset: (1) extracting the user ratings of the top 1000 movies in the dataset, (2) transforming each user's evaluation of a movie into an independent feature vector with unrated movies filled with zero values instead, (3) eliminating users who do not meet the requirements of rating at least one movie and (4) lastly, the records of users who meet the

Table 3 Summary of dataset used in our experiment, with respect to number of instances available classes provided, and attributes available

Dataset	Instances	Classes	Attributes
CIFAR-100 Liu et al. (2021)	50, 000	100	50
Purchase-100 Dwork (2008)	200, 000	100	599
Netflix Prize-100 Krizhevsky and Hinton (2009)	100, 000	100	1000

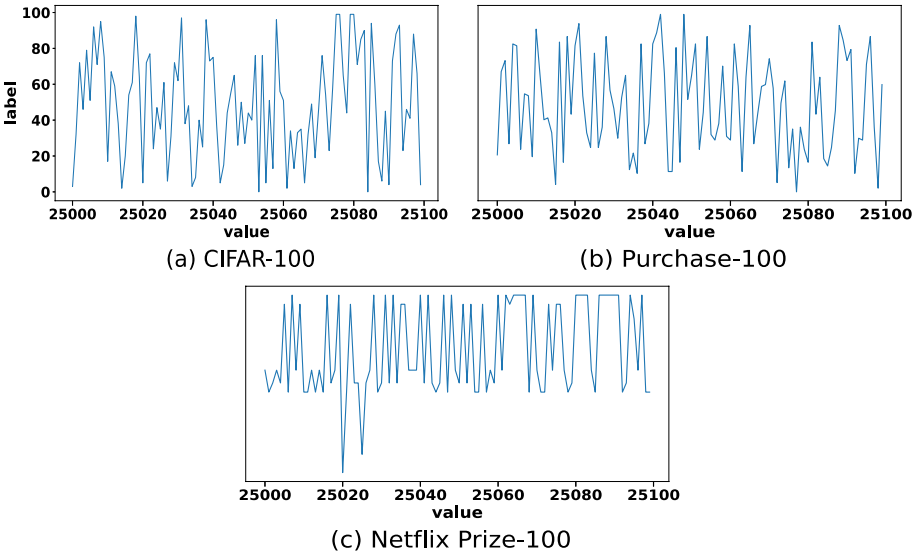


Fig. 6 Partial data distribution illustration of CIFAR-100, Purchase-100 and Netflix Prize-100 datasets

requirements are clustered into 100 categories by k-means clustering in order to control the complexity of the dataset uniformly.

4 Experimental results

In this section, we show the results of applying this framework on the LR and NN methods. In Sect. 4.3, we summarize the findings of the results and answer the questions posed in Sect. 3.1.

4.1 Logistic regression results

For the LR algorithm, we add DP-noise in S1 and S3 according to the proposed UKF-DPML framework, and then apply UKF to filter the noise-perturbed aggregates. As comparison, we also train a DPML model without applying UKF, and finally evaluate the models. *The results suggest that training the model on three different datases with this UKF-DPML framework can improve the utility without loss of privacy.* We describe the experiments on these three datasets in a more detailed manner.

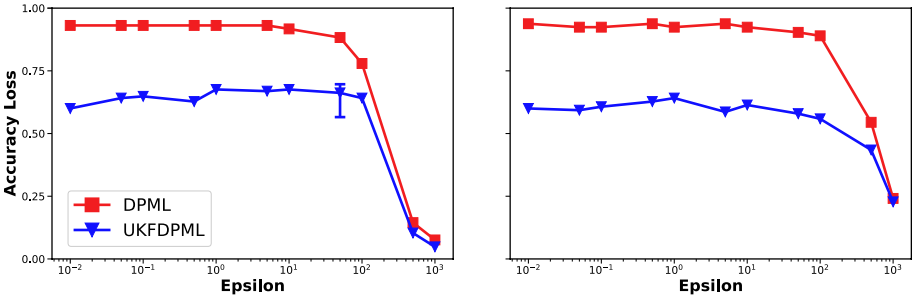
4.1.1 Accuracy loss

The variation in accuracy loss (ACL) of the model after training on the CIFAR-100 dataset is shown in Fig. 7. Similarly, Figs. 8, 9 shows the ACL on Purchase-100 and Netflix Prize-100 dataset separately. In these figures, the line with the DPML label represents the ACL of the DPML model without UKF adopted, and the line with the UKFDPML label represents the results with UKF adopted according to the framework. In our experiments, we reserve the precision of the ACL to 8 decimal places because this is sufficient to indicate the magnitude of its variation.

Figure 7 shows the decrease of ACL after applying UKF when LR model is trained on the CIFAR-100 dataset whether noise is added at S1 and S3. In S2, the maximum difference in ACL metric between the DP-enabled LR model without UKF and with UKF applied is 0.2488263 (for ϵ=0.01) and the minimum difference is 0.0066293 (for ϵ=1000). In S3, the maximum and minimum differences are 0.3379313 (for ϵ=0.01) and 0.0137931 (for ϵ=1000). *Therefore, we obtain that the utility of the model is improving after applying UKF.* The maximum improvement in utility is $\approx$ 26.726% (ϵ=0.01) in S1 and $\approx$ 36.029% (ϵ =0.01) in S3. In addition, we also gain the extent of utility improvement with UKF adopted decreases as the value of ϵ converges to 1000. We reckon that this occurrence is related to the a posteriori estimation during the UKF execution. Moreover, the loss of model accuracy decreases remarkably at S1 and S3 when ϵ>100 . Meanwhile, the effectiveness of the utility improvement also decreases significantly from that point onwards. *For this, we infer that the effect of UKF on model utility improvement is related to the tradeoff inflection point of the utility and privacy of the DP model.* In terms of this phenomenon, we provide *a* concrete summary analysis in Sect. 4.3.

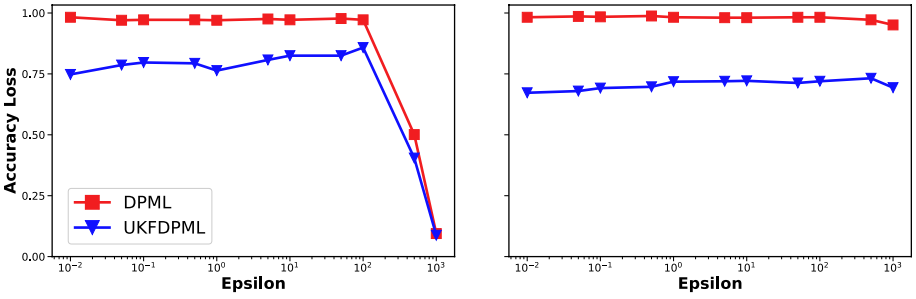
Figure 8 shows the result on Purchase-100 dataset. In S1, the maximum difference between without UKF and with UKF accuracy loss is 0.2609457 (ϵ=5) and the minimum difference is 0.0070053 (ϵ=1000). In S3, the maximum difference is 0.4255692 (ϵ=0.01) and the minimum difference is 0.2574431 (ϵ=1000). *Hence, we conclude that the utility of the DP-enabled LR model is improved after applying UKF on this dataset, and the maximum improvement can reach $\approx$ 26.750% for S1 and $\approx$ 43.626% for S3.* Furthermore, in S3, there is no significant degradation in the loss of accuracy as seen in Fig. 7, as the maximum value of the privacy budget for the added noise has not yet reached the upper limit of privacy leakage for this dataset. *It is also demonstrated that even adding the minimum amount of noise* (ϵ=1000) *still has a significant impact on the accuracy of the model in S3.*

Figure 9 shows the accuracy loss of the trained model on the Netflix Prize-100 dataset. From this figure we observe that in S1 the maximum difference is 0.2934272 (ϵ=5) and the minimum difference is 0.02347417 (ϵ=1000). In S3, the maximum and minimum differences are 0.4131455 (ϵ=0.5) and 0.3075117 (ϵ=1000). Hence, we deduce that applying UKF on this dataset can improve the utility of the trained model by $\approx$ 29.691% in S1 and by $\approx$ 42.512% in S3. Besides, we can see that during S1, (1) for $0.01 \leq \epsilon \leq 100$, the fluctuation of the difference in ACL between not applying UKF and applying UKF remains essentially constant. (2) For ϵ 100 , the ACL decreases noticeably. When applying UKF in S3, the variation in the ACL of the model is more fluctuating than applying in S1. And both of the folds in (b) S3 do not show a obvious degradation. *This is thought to be related to the distribution of the dataset and the process of posterior estimation in the UKF. Also, the addition of minimal noise in S3 clearly affects the privacy of the model* (ϵ=1000). Therefore, there is no significant decrease in ACL as Fig. 7. The analysis of this phenomenon are shown in Sect. 4.3.



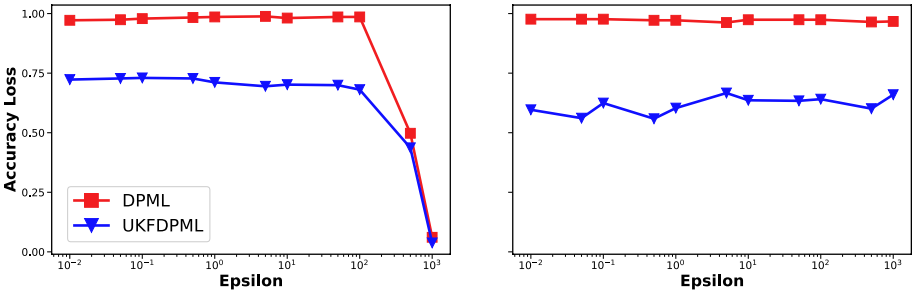
(a) S1: Logistic Regression (b) S3: Logistic Regression

Fig. 7 Accuracy Loss for LR method used on CIFAR-100 dataset, when different amount of DP noise is applied at Stage 1,3 of the framework with or without UKF adopted



(a) S1: Logistic Regression (b) S3: Logistic Regression

Fig. 8 Accuracy Loss for LR method used on Purchase-100 dataset, when different amount of DP noise is applied at Stage 1,3 of the framework with or without UKF adopted



(a) S1: Logistic Regression (b) S3: Logistic Regression

Fig. 9 Accuracy Loss for LR method used on Netflix Prize-100 dataset, when different amount of DP noise is applied at Stage 1,3 of the framework with or without UKF adopted

In conjunction with Figs. 7, 8, 9, we find that *the application of UKF to DP-enabled LR models in S3 is generally more effective in improving the utility of the model than in S1, the phenomenon is discussed in Sect. 4.3.*

4.1.2 Attacks & privacy leakage

We illustrate the employment of attack advantage to measure the degree of privacy leakage, the attacks used in this study and the reasons why we chose them in Sect. 2.6.1. Table 4 shows the advantage of the model against YeomMI, YeomAI, and SalemAI attacks without privacy preserving mechanism of the LR method, i.e. without adding any noise to the training process.

Figures 10 and 13 show the advantages of the LR method to the same attacks after training on the CIFAR-100 dataset using the proposed framework. Specifically, Fig. 10 shows the results of injecting UKF-filtered noise to S1 and Fig. 13 shows the results in S3. Similarly, Figs. 11, 14 show the results on the Puchase-100 dataset. Figures 12, 15 show the results on the Netflix Prize-100 dataset. In the following we compare the results in terms of (1) general trend, (2) tradeoff inflection points and (3) degree of fluctuation.

(1) ***General trend*** By observing Figs. 10, 11, 12, we compare the impact on privacy with and without UKF in S1. It can be seen that the trend of privacy leakage among the three attacks (YeomMI, YeomAI and SalemAI) remains generally consistent within the experimental privacy budget set ϵ. The attacks are performed on the LR model trained with and without UKF in S1 according to our proposed framework. The upper limit of advantage for all three attacks is smaller than that for the model trained without DP noise (Table 4). The result is consistent with common sense. We then observe Figs. 13, 14, 15 to reach a similar conclusion on S3. Thus, for the same privacy budget ϵ, the model trained with UKF does not experience more privacy loss compared to the model trained without UKF. This result verifies the validity of Theorem 1 in Sect. 2.5.

(2) ***The tradeoff inflection point*** According to the work in Zhao et al. (2020), we know that there is an inflection point in the trade-off between privacy and utility for DPML models with respect to the value of the privacy budget. After the inflection point, the privacy and utility of the model are significantly affected: (1) For the privacy, the degree of privacy protection of the model decreases notably, corresponding to a increase in the degree of privacy leakage. (2) For the utility, the degree of utility improves significantly. Our experiments also verify this tradeoff finding of this research. By observing Fig. 10a, we notice the advantage of the model from YeomMI attacks increases noticeably after the privacy budget is taken to be 100 on the CIFAR dataset, i.e. the degree of privacy protection for the model decreases. Meanwhile, in Fig. 7a, we see that the loss of model accuracy decreases after the privacy budget ϵ is taken to be 100. Therefore, the inflection point of the LR method on this dataset after the injection of noise in the S1 is ϵ=100. Similarly, by comparing Figs. 7b and 13a, we obtain that the inflection point for the DP model in S3 on the CIFAR dataset is $\epsilon = 100$. Table 5 summarises the values of the inflection points for the utility and privacy of the model after the LR algorithm achieves DP in S1 and S3 on each of the three datasets. Furthermore, we also discover that the magnitude of the improvement in model utility by applying UKF decreases after the inflection point from Figs. 7, 8, 9. *Therefore, we deduce that the utility effectiveness of UKF is related to the inflection point.* This phenomenon is related to the prediction equations of UKF, which we analyse in Sect. 4.3.

(3) ***Degree of fluctuation*** In our observation of Figs. 10, 13 and 11, 14 and 12, 15, we note that: for the LR model trained using our proposed framework on the three datasets with the advantage of the three attacks, the degree of fluctuation with UKF applied in S3 is generally greater than that of applying it in S1. This indicates the stability of model

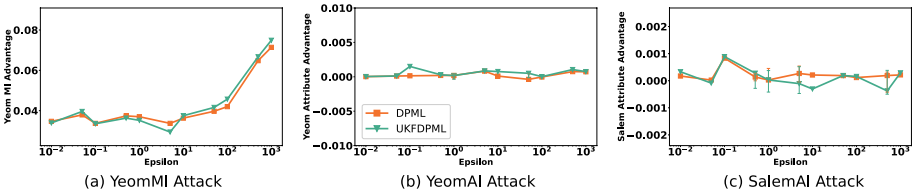


Fig. 10 Advantage of three attacks for LR method used on CIFAR-100 dataset, when different amount of DP noise is applied at Stage 1 of the proposed framework with or without UKF adopted

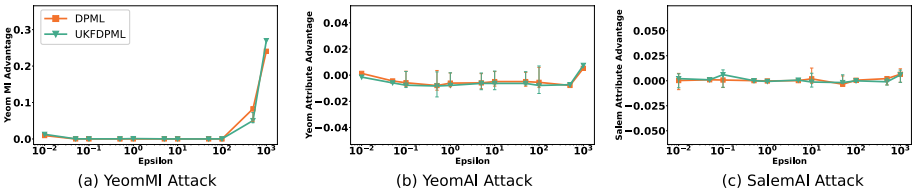


Fig. 11 Advantage of three attacks for LR method used on Purchase-100 dataset, when different amount of DP noise is applied at Stage 1 of the proposed framework with or without UKF adopted

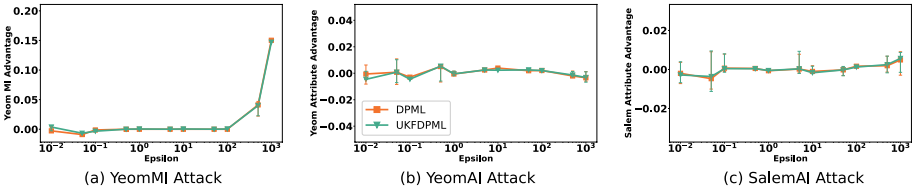


Fig. 12 Advantage of three attacks for LR method used on Netflix Prize-100 dataset, when different amount of DP noise is applied at Stage 1 of the proposed framework with or without UKF adopted

Table 4 Advantage of three attacks for LR method, when no DP noise is applied at Stage 1,2 or 3 according to the framework and for different datasets used

Dataset	YeomMI advantage	Yeom attribute advantage	Salem attribute advantage
CIFAR-100	0.0828	0.0053	0.0045
Purchase-100	0.4857	0.0138	0.0702
Netflix Prize-100	0.5679	0.0942	0.0689

The underlying complexity of data vectors in each dataset remains the same

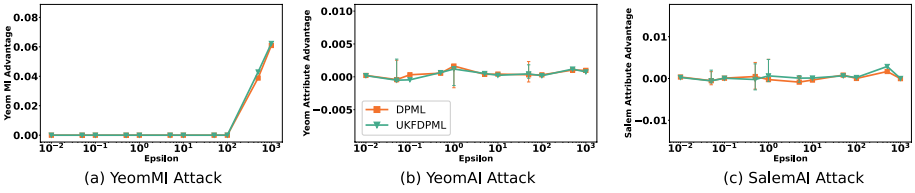
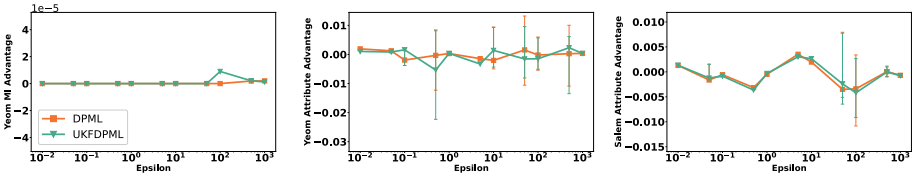
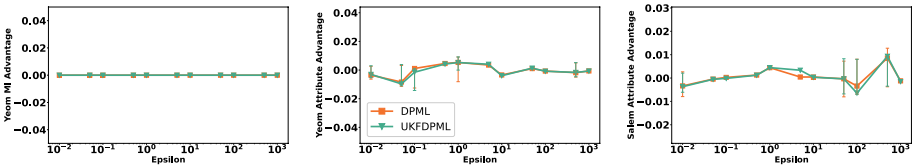


Fig. 13 Advantage of three attacks for LR method used on CIFAR-100 dataset, when different amount of DP noise is applied at Stage 3 of the proposed framework with or without UKF adopted



(a) YeomMI Attack (b) YeomAI Attack (c) SalemAI Attack

Fig. 14 Advantage of three attacks for LR method used on Purchase-100 dataset, when different amount of DP noise is applied at Stage 3 of the proposed framework with or without UKF adopted



(a) YeomMI Attack (b) YeomAI Attack (c) SalemAI Attack

Fig. 15 Advantage of three attacks for LR method used on Netflix Prize-100 dataset, when different amount of DP noise is applied at Stage 3 of the proposed framework with or without UKF adopted

privacy after applying UKF in S1 is higher than S3, which is thought to be related to the application of the UKF filtering process in S3. Since applying UKF in S3 is a direct filtering of the model output distribution parameters, it is more likely to obtain correct information when inferential attacks are performed on the output by back-propagation. This leads to greater fluctuation in the degree of the model's privacy-preserving effect after applying UKF in S3.

Remark 2 In conducting our experiments, our results do not lead to a lower bound on this inflection point, as we have chosen the privacy budget ϵ value in increasing order. However, our experimental evidence strongly suggests that there is an inflection point after which the tradeoff between privacy and utility of the model changes significantly. The experimental results also suggest that the effectiveness of UKF is related to this inflection point. The the effectiveness of using UKF to improve utility decreases after this point.

4.2 Neural networks

Based on the experiments of Shokri et al. (2017), Jayaraman and Evans (2019), we train a Neural Networks (NN) model whose architecture is similar to theirs. Our model is composed of one input layer, two hidden layers and one output layer. Each hidden layer has 256 neurons, activated by RELU function. The output layer is a softmax layer with 100 neurons, each of which corresponded to a label. Then, we add UKF-filtered noise to the basic stages S1 and S2 in the proposed framework. Similar to Sect. 4.1, we also train the DP-enabled NN model without applying UKF. Finally, we evaluate and compare the models. *The experimental results show that the utility of the NN model which is trained with the*

Table 5 The tradeoff inflection point of utility and privacy when applying LR method on CIFAR-100, Purchase-100, Netflix Prize-100 datasets

Dataset	S1	S3
CIFAR-100	$\epsilon = 100$	$\epsilon = 100$
Purchase-100	$\epsilon = 100$	$\epsilon = 5000$
Netflix prize-100	$\epsilon = 100$	$\epsilon = 5000$

method of our framework has been improved without loss of privacy. Next, we describe the experiments on the three datasets in detail.

4.2.1 Accuracy loss

Figures 16, 17, 18 show the variation of accuracy loss (ACL) after training the model on CIFAR-100, Purchase-100, and Netflix Prize-100 datasets.

Figure 16 illustrates that the ACL of the model decreases after applying UKF. The NN model is trained on the CIFAR-100 dataset with noise added in S1 and S2. The difference in ACL of the model applied without or with UKF adopted in S1 are 0.3103448 at maximum (for ϵ=0.5) and 0.0137031 (for ϵ=1000) at minimum. In S2, the maximum and minimum difference is 0.4972414 (for ϵ=100) and 0.0055172 (for ϵ=1000). As a result, we conclude that the utility of model improved after UKF adopted, with the maximum improvement in utility reaching $\approx$ 33.333% in S1 (ϵ=0.5) and $\approx$ 53.407% in S2 (ϵ=100). Furthermore, when $\epsilon = 100$, the ACL of both models decreased significantly. Similarly, the effectiveness of UKF also decline, whether the DP noise is applied in S1 and S2. This trend verifies that there is a trade-off between privacy and utility, which is simultaneously affecting the effectiveness of UKF.

Figure 17 shows the results on the Purchase-100 dataset, where the maximum and minimum difference of ACL of models trained without and with UKF applied in S1 are 0.1511290 (ϵ=0.01) and 0.0177419 (ϵ=1000) respectively. In S2, the maximum and minium differences become 0.2193548 (ϵ=1000) and 0.0048387 (ϵ=1000). Hence, it can be concluded that the utility of the DP-enabled NN model is improved after UKF adopted on this dataset. Because the ACL is improved with the UKF-filtered noise added in S1 and S2 by $\approx$ 15.513% (ϵ=0.01) and $\approx$ 22.553% (ϵ=0.05) respectively.

Figure 18 shows the results on the Netflix Prize-100 dataset. The maximum difference in S1 between the methods without and with UKF is 0.3573034 (ϵ=0.05) and the minimum is 0.3932584 (ϵ=1000). This illustrates that the utility of the DP-enabled NN model is improved after UKF adopted on this dataset. Moreover, the performance can be improved up to 37.324% in S1 and 40.553% in S2. Besides, since we add ϵ with the largest privacy loss (ϵ=1000) in S1 and S2, the loss of model accuracy still remains above 0.75 and does not drop to close to 0.00 as in Fig. 16. *We infer that the noise added in S1 and S2 in the experiment does not reach the upper limit of privacy leakage in this dataset.*

4.2.2 Attacks & privacy leakage

Analogous to Sect. 4.1.2, Table 6 shows the advantages of the model subjected to YeomMI, YeomAI and SalemAI attacks without privacy-preserving treatment of the NN method, i.e., training without adding any noise. Figures 19 and 22 show the advantages of UKF-based DP-enabled NN method trained on CIFAR-100 dataset, being subjected to the attacks mentioned above. Similarly, Figs. 20, 23 show the results on the Puchase-100 dataset and

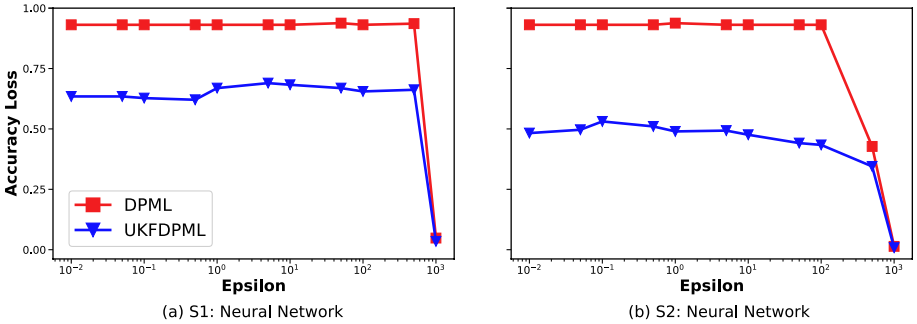


Fig. 16 Accuracy Loss for NN method used on CIFAR-100 dataset, when different amount of DP noise is applied at Stage 1,2 of the framework with or without UKF adopted

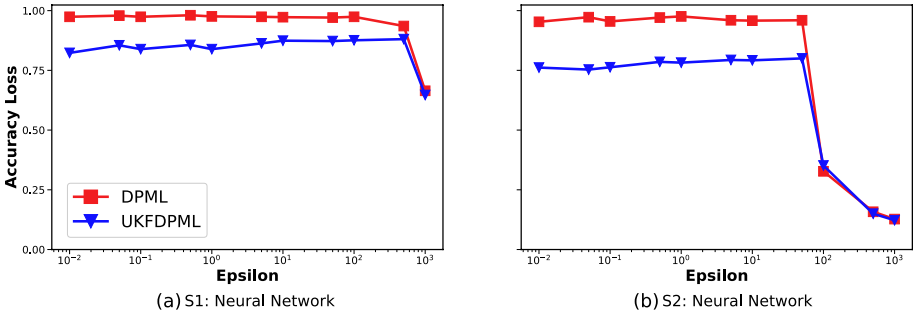


Fig. 17 Accuracy Loss for NN method used for Purchase-100 dataset used, when different amount of DP noise is applied at Stage 1,2 of the framework with or without UKF adopted

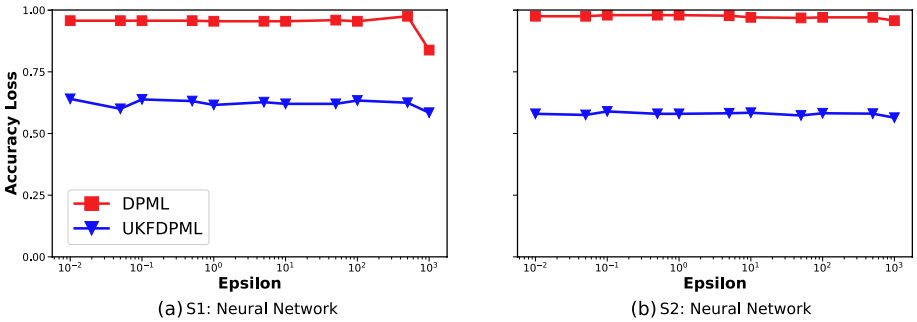


Fig. 18 Accuracy Loss for NN method used on Netflix Prize-100 dataset, when different amount of DP noise is applied at Stage 1,2 of the framework with or without UKF adopted

Figs. 21, 24 are results on the Netflix Prize-100 dataset. In the following, we conduct an in-depth analysis comparing the experiment results in three aspects as Sect. 4.1.2.

(1) ***General trend***. By observing Figs. 19, 20, 21, 22, we compare the results of the privacy impact whether adopting UKF in S1 or not. We learn the trend of privacy leakage degree between models trained with and without UKF adopted in S1. After suffering

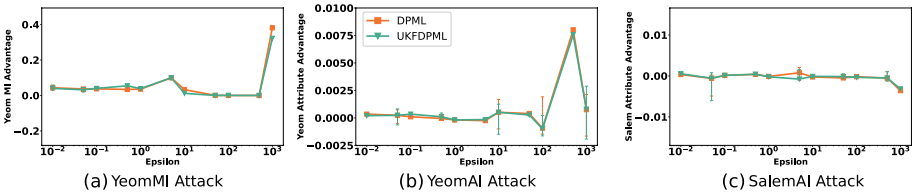


Fig. 19 Advantage of three attacks for NN method used on CIFAR-100 dataset, when different amount of DP noise is applied at Stage 1 of the proposed framework with or without UKF adopted

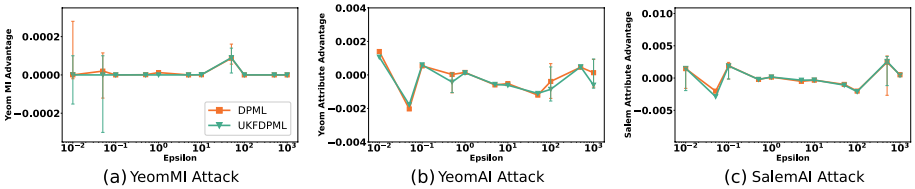


Fig. 20 Advantage of three attacks for NN method used on Purchase-100 dataset, when different amount of DP noise is applied at Stage 1 of the proposed framework with or without UKF adopted

three attacks (YeomMI, YeomAI and SalemAI) within the experimental privacy budget set ϵ, the trends remains generally consistent. And the upper limit of the advantage of these three attacks is smaller than that of privacy leakage without DP noise injection (Table 6). We draw similar conclusion in S3 through Figs. 22, 23, 24. *Therefore, it indicates that under the same privacy budget ϵ, the degree of privacy protection of the model with UKF adopted is consistent with the model without UKF training.* Applying UKF does not change the degree of privacy protection, which verifies the Theorem 1 in Sect. 2.5.

(2) ***The tradeoff inflection point*** From Fig. 19a, we find when the privacy budget ϵ=500, the advantage of YeomMI attack significantly increases. This increase illustrates the degree of privacy protection of the model is considerably reduced. Meanwhile, it is observed in S1 from Fig. 16a that when ϵ 500, the ACL of the model decreases, and when ϵ=1000, it decreases to nearly 0.00. This indicates the utility of the model is improved at this time. Therefore, the tradeoff inflection point ϵ between the privacy and utility of the DP-enabled NN model is 500 in S1 on CIFAR-100 dataset. At this point, the tradeoff between the model's privacy and utility results in notable performance. Consequently, the model has the maximum utility while ensuring preferable privacy. Similarly, through Fig. 16b and 22a, we know that the inflection point of the DP-enabled NN model in S2 on CIFAR-100 dataset is ϵ=100. Table 7 summarizes the tradeoff inflection point ϵ between the privacy and utility of the model in S1 and S2 respectively on three datasets. Apart from this, we also discover a phenomenon similar to Sect. 4.1.2 which is the improvement of the utility by applying UKF decreases after the inflection point.

(3) ***Degree of fluctuation*** Figures 19, 20, 21, 22, 23, and 24 show the results of the proposed framework used to train the model on the three datasets. If the added noise has not reached the inflection point, the model's dominant results under YeomMI, YeomAI, and SalemAI attacks have little fluctuation. Alongside this, Fig. 19 also reveals that the advantage under the three attacks is less than 0. It reveals more noise is allowed to

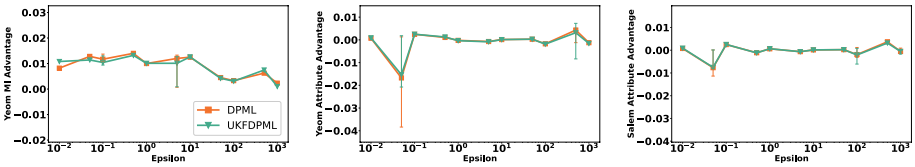


Fig. 21 Advantage of three attacks for NN method used on Netflix Prize-100 dataset, when different amount of DP noise is applied at Stage 1 of the proposed framework with or without UKF adopted

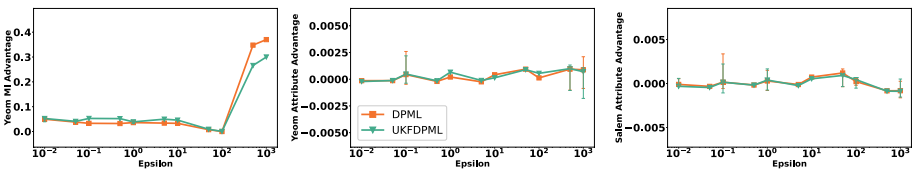


Fig. 22 Advantage of three attacks for NN method used on CIFAR-100 dataset, when different amount of DP noise is applied at Stage 2 of the proposed framework with or without UKF adopted

Table 6 Advantage of three attacks for NN method, when no DP noise is applied at Stage 1, 2 or 3 according to the framework and for different datasets used

Dataset	YeomMI advantage	Yeom attribute advantage	Salem attribute advantage
CIFAR-100	0.7286	0.0180	0.0197
Purchase-100	0.3794	0.0636	0.1347
Netflix Prize-100	0.4844	0.0198	0.1429

The underlying complexity of data vectors in each dataset remains the same

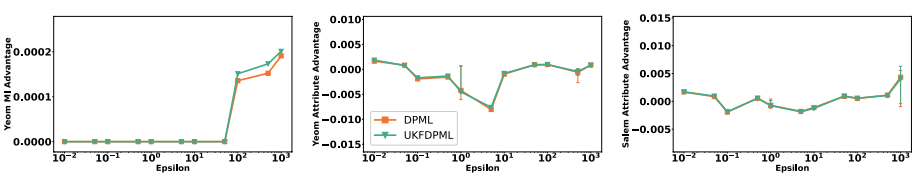


Fig. 23 Advantage of three attacks for NN method used on Purchase-100 dataset, when different amount of DP noise is applied at Stage 2 of the proposed framework with or without UKF adopted

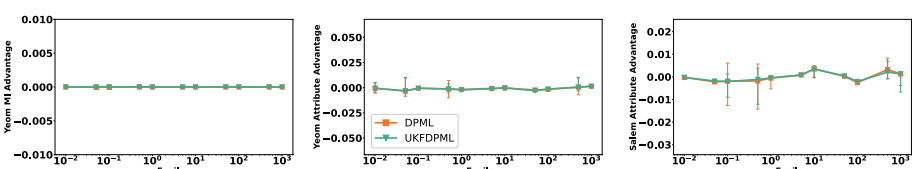


Fig. 24 Advantage of three attacks for NN method used on Netflix Prize-100 dataset, when different amount of DP noise is applied at Stage 2 of the proposed framework with or without UKF adopted

be added in S1, which is consistent with the conclusion Zhao et al. (2020). Through YeomMI in Fig. 23a, we note that when $\epsilon < 100$, *the advantage of UKF-adopted trained model is smaller than the model trained without UKF adopted.* It indicates the application of our framework can provide better privacy protection. This may provide a more profitable way for ML practitioners to make their models more privacy-protected and useful with less noise. With ϵ=100, the advantage of YeomMI attack significantly rises. *There is also a greater advantage of using UKF than not using UKF, which explains UKF filter has a* ϵ threshold. After this threshold, application of UKF during model training leads to a greater privacy disclosure than that without UKF adopted. The discovery of this threshold is discussed in details in Sect. 4.3.

4.3 Summary of findings

In Sects. 4.1 and 4.2 we analyse the results of the experiment. In this section we present a summary discussion of the findings from the experiment. With the summary, we also answer the three questions **Q1**, **Q2** and **Q3** raised in Sect. 3.

(1) Training the model using our proposed UKF-DPML framework can achieve higher utility with the same privacy budget. Applying UKF to process the noise-perturbed aggregates ensures that privacy is not compromised while improving the utility of the model. We apply the framework to the LR and NN methods and train them on the CIFAR-100, Purchase-100, and Netflix Prize-100 datasets. The results show that applying UKF to S1, S2 or S3 with noise addition achieves higher model utility with the same degree of privacy leakage. (see Sec. 4.1, Sec. 4.2)

(2) The effect of UKF to enhance the utility of the model is related to the stage of application. Applying UKF to filter DP-noise in S2 or S3 results in a greater increase in model utility than applying in S1. Therefore, the effect of applying UKF in S3 or S2 is more effective in improving utility. (see Sec.4.1.1, Sec. 4.2.2)

(3) There is an inflection point in the tradeoff between privacy and utility of the DP-enabled model and the effectiveness of the action of UKF is related to that point. After the inflection point, the privacy of the model decreases remarkably and the utility increases also. However, before reaching the inflection point, the privacy of the model fluctuates to a lesser extent with changes in the value of ϵ. The discovery of this inflection point is consistent with the study of Zhao et al. (2020). At the same time, the effect of the UKF is related to the inflection point. After that point the effectiveness of the UKF on utility improvement decreases (see Sec. 4.1.2, Sec. 4.2.2). In this regard, we suppose that is related to the posterior estimation process of the UKF, where the less noise is added, the closer the predicted value will be to the measured value. When the value of

Table 7 The tradeoff inflection point of utility and privacy when applying NN method on CIFAR-100, Purchase-100, Netflix Prize-100 datasets

Dataset	S1	S2
CIFAR-100	$\epsilon = 500$	$\epsilon = 100$
Purchase-100	$\epsilon = 5000$	$\epsilon = 100$
Netflix Prize-100	$\epsilon = 5000$	$\epsilon = 5000$

ϵ exceeds the inflection point, and the effectiveness of applying the UKF to filter noise will be reduced accordingly.

(4) The degree of fluctuation in the ADV of the model trained with UKF applied under the three attacks is related to the stage of UKF application. The fluctuation after applying UKF in the three stages are ranked from largest to smallest: S3 > S2 > S1 (see Sec.4.1.2, Sec.4.2.2). For this phenomenon, we deduce that it is related to the object of the noise effect: applying UKF in S3 is a direct filtering of the model output distribution parameters. As a result, it is more likely the correct information is obtained when inferential attacks are back-propagated based on the output. That leads to greater fluctuations in the degree of the model's privacy preserving effect after the application of UKF in S3. In S1, we perturb the data with UKF-filtered noise. Therefore, the application of UKF only affects the data before training and does not directly interfere with the training process of the model. The fluctuation in the privacy of the model after applying UKF at this stage is relatively small. Moreover, we also consider the degree of fluctuation is related to the inflection point. That fluctuation in the privacy of the model is also smaller when the amount of noise added has not yet reached the inflection point of the model (mentioned in (2)).

(5) There is a threshold ϵ, exceeding which the application of our proposed UKF-DPML framework to train the model may result in a greater loss of privacy for utility improvement. (see Sec.4.2.2)

Remark 3 The privacy budget value ϵ used in our experiments is chosen according to a certain increasing regular. Thus, our results cannot extrapolate to a lower bound for this threshold. However, our experimental evidence strongly suggests that there is a threshold. If exceeding that, the use of UKF to improve the utility of the model will lead to greater privacy leakage.

(6) Adding DP-noise in S1 appears to reduce the attack advantage to 0, indicating that S1 allows for more noise to be added. In contrast, the attack advantage of applying UKF to the model in S2 appears lower than without UKF applied. Those trends suggest that the privacy-preserving effect of the model trained with UKF-filtered DP-noise in S2 appears better than without applying UKF. This may provide a way to achieve better privacy and utility of the model with less noise added. (see Sec. 4.2.2)

According to findings (1)–(6), the following summaries are made for which stage to apply the UKF-DP method:

(i) In the case that the ML algorithm achieve DP protection in S2 (or S3) and S1 while the priority of utility is greater than privacy, the S2 would be the considerable choice under this case. Because applying UKF to filter DP-noise in S2 or S3 results in a greater increase in model utility than applying in S1. (finding (2)). Meanwhile, the variation of privacy after applying UKF in the three stages are ranked from largest to smallest: S3 > S2 > S1 (finding (4)).

(ii) In the case that the ML algorithm achieve DP protection in S2 (or S3) and S1 and the priority of privacy outweighs the utility, S1 is the preferred option of applying UKF-DP.

(iii) In the case that the ML algorithm only achieve DP protection in S2 or S3, then the choice of S2 leads to better model utility with the given privacy budget.
(iv) In the case that the ML algorithm only allows DP protection in S1, then choosing S1 to apply UKF obtain higher model utility within the given privacy budget. Notably, although we generalize the conclusions for various cases using UKF-DPML framework based on our experimental results, the stage of applying UKF-DPML relies on the specific algorithm used. The above is the summary of our experimental findings and we make a comprehensive comparison of the experimental differences in Table 8.

5 Discussion

With our mathematical derivations and experiments, we gain the proposed UKF-DPML framework can improve the utility of the model without privacy depletion. Thus, the present framework may be capable of providing ML practitioners with a more profitable approach of adding UKF-filtered noise to protect privacy and improve the utility. Throughout the study, there are two aspects worth discussing, as follows.

(1) The setting of the inference attacker's prior probability of success. In inference attacks, we use the same number of membership and non-membership records, which provides the attacker with a 50-50 prior probability of success. Our goal is to figure out whether the privacy leakage of the UKF-adopted models are consistent with models trained without UKF. Consequently, it is not conflicting and reasonable to use that probability. In addition, our results show privacy leakage due to three specific inference attacks. As the attacks will only get stronger, the future attacks may be able to infer more information than that from our experiments.
(2) The limitations of the proposed method. In Sect. 4.3, we summarize general conclusions for various cases using UKF-DPML based on the experimental results. However, the better stage to apply UKF-DPML depends on the specific model. Through our theoretical analysis of the experimental results, we suggest that our conclusions have the reasonable generality. Furthermore, given that there are many sources of randomness in the training process which we have discussed in Remark 1, it appears that in some particular cases, UKF-DPML might produce less stable results than the DPML counterpart. Hence, to reduce the effectiveness of such randomness, the outcomes we present in this paper are repeated on average 10 times. Accordingly, both high and low performance instances of this attack are possible for us to capture. In ideal world, we would like to perform more replications to obtain more accurate averages.

6 Future work

In this paper, we propose a UKF-DPML framework that can improve the utility of the model with the same privacy budget. In the future, we will apply this framework to other machine learning algorithms, such as DP-Based Naive Bayes (Holohan et al., 2019; Vaidya et al., 2013) and DP-Random Forests (Fletcher & Islam, 2017). Apart from applying the

Table 8 The comparative summary of the experimental differences in related studies

	DP-noise	UKF	ML model	ML model attack type			Dataset
	Injection	Application	Type	YeomMI	YeomAI	SalemAI	
Zhao et al. (2020)	✓	×	NB and NN	×	✓	✓	CIFAR, Purchase and Netflix
Kim and Lee (2021)	×	✓	MLP, GP and LSTM	×	×	×	MERRA-2
Wang et al. (2018)	✓	✓	None	×	×	×	Employee, Person, Rhine and Shipment
Our work	✓	✓	LR and NN	✓	✓	✓	CIFAR-100, Purchase-100 and Netflix-100

new algorithm, we will also build a DP-enabled ML system which is able to automatically adjust the privacy budget. In this system, the ML practitioners no longer need to conduct multiple experiments to determine the privacy budget required for a model application scenario. Instead, the system will automatically select the optimal privacy budget based on the degree of privacy protection and prediction accuracy required for the scenario. This system will achieve a higher utility with a lower privacy budget through UKF filter until meeting the requirement of the scenario. We conceive this system will improve the efficiency of differentially private machine learning implementations and bring about greater profitability for practitioners. Hence, this is what we are working on.

7 Conclusion

Differential privacy has earned a well-deserved reputation for providing a principled and powerful mechanism for ensuring provable privacy. Our main contribution in this paper is to propose a UKF-based DP-enabled ML framework. It makes the model more practical with the same privacy budget and provides a higher profit value implementation method for researchers in DPML.

We identified three such stages in ML frameworks that support DP and UKF, where DP-noise can be added and filtered with UKF: (1) S1: directly during data collection, (2) S2: during model training, or (3) S3: when the model is finalized. We allow the practitioners to apply UKF-filtered noise at different stages of framework to achieve private requirement. Simultaneously, we evaluated the impact on utility by loss of accuracy with and without applying UKF. The impact on privacy was evaluated by three known attacks.

We applied the proposed UKF-DPML framework to various implementations of ML algorithms and measure their abilities to resist privacy attacks from the real world. For each implementation, we trained them with a set of privacy budgets and datasets, providing the same mathematical privacy guarantees. By measuring the resistance of models to real-world inference attacks and the classification accuracy, we conclude that applying the methods in our framework can achieve higher model utility with the same level of privacy protection.

Acknowledgements We would like to thank all the anonymous reviewers for their insightful comments on this work.

Author Contributions All authors stated consent to this publication and contributed according to the order presented at the beginning of the paper.

Funding This work was supported by the Special Project for Research and Development in Key areas of Guangdong Province, China (Grant No. 2020B0101090003) and the Natural Science Foundation of Guangdong Province, China (Grant No. 2021A1515011607). The opinions in this paper are those of the authors and do not necessarily reflect the opinions of any funding sponsor or the China Government.

Data availability The data supporting this study are from previously reported studies, which have been cited. The processed data are available from the corresponding author upon request.

Code availability The source code of the relevant algorithms is referenced in the paper. The code for the experimental reproduction will be available from the corresponding author upon request once the DP-enabled ML system is online.

Declarations

Conflicts of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., & Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. CCS '16* (pp. 308–318). Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/2976749.2978318

Ateniese, G., Mancini, L. V., Spognardi, A., Villani, A., Vitali, D., & Felici, G. (2015). Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks, 10*(3), 137–150. https://doi.org/10.1504/IJSN.2015.071829.

Barnes, J. (2015). Azure machine learning. Microsoft Azure Essentials. 1st ed, Microsoft

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In Lechevallier, Y., Saporta, G. (eds.), *Proceedings of COMPSTAT'2010* (pp. 177–186). Physica-Verlag HD, Heidelberg. https://doi.org/10.1007/978-3-7908-2604-3_16

California Consumer Privacy Act (CCPA). https://oag.ca.gov/privacy/ccpa (2020)

Carlini, N., Liu, C., Erlingsson, Ú., Kos, J., & Song, D. (2019). The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX Security 19)* (pp. 267–284). USENIX Association, Santa Clara, CA

Chan, T.-H.H., Li, M., Shi, E., & Xu, W. (2012). Differentially private continual monitoring of heavy hitters from distributed streams. In S. Fischer-Hübner & M. Wright (Eds.), *Privacy Enhancing Technologies* (pp. 140–159). Berlin, Heidelberg: Springer.

Chan, T.-H.H., Shi, E., & Song, D. (2011). Private and continual release of statistics. *ACM Transactions on Information and System Security*. https://doi.org/10.1145/2043621.2043626.

Chaudhuri, K., & Monteleoni, C. (2008). Privacy-preserving logistic regression. *Advances in neural information processing systems,21,*.

Chaudhuri, K., Monteleoni, C., & Sarwate, A. D. (2011). Differentially private empirical risk minimization. *Journal of Machine Learning Research,12*(3).

Chen, S., Wang, H., Xu, F., & Jin, Y.-Q. (2016). Target classification using the deep convolutional networks for SAR images. *IEEE Transactions on Geoscience and Remote Sensing, 54*(8), 4806–4817. https://doi.org/10.1109/TGRS.2016.2551720.

Das, A., Borisov, N., & Caesar, M. (2016). Tracking mobile web users through motion sensors: Attacks and defenses. In *NDSS*.

Donaldson, J., Donaldson, M.J. (2012). Package 'bigml'

Ducange, P., Pecori, R., & Mezzina, P. (2018). A glimpse on big data analytics in the framework of marketing strategies. *Soft Computing, 22*(1), 325–342. https://doi.org/10.1007/s00500-017-2536-4.

Dwork, C. (2008). Differential privacy: A survey of results. In Agrawal, M., Du, D., Duan, Z., Li, A. (eds.) *Theory and applications of models of computation* (pp. 1–19). Springer, Berlin. https://doi.org/10.1007/978-3-540-79228-4_1

Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., & Naor, M. (2006). Our data, ourselves: Privacy via distributed noise generation. In Vaudenay, S. (ed.) *Advances in cryptology - EUROCRYPT 2006* (pp. 486–503). Springer, Berlin. https://doi.org/10.1007/11761679_29

Dwork, C., Naor, M., Pitassi, T., & Rothblum, G.N. (2010). Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on theory of computing. STOC '10* (pp. 715–724). Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/1806689.1806787

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

Fan, Y., Bai, J., Lei, X., Lin, W., Hu, Q., Wu, G., et al. (2021). Ppmck: Privacy-preserving multi-party computing for k-means clustering. *Journal of Parallel and Distributed Computing, 154,* 54–63. https://doi.org/10.1016/j.jpdc.2021.03.009.

FilterPy. https://github.com/rlabbe/filterpy (2020)

Fletcher, S., & Islam, M. Z. (2017). Differentially private random decision forests using smooth sensitivity. *Expert Systems with Applications, 78,* 16–31. https://doi.org/10.1016/j.eswa.2017.01.034.

Fredrikson, M., Lantz, E., Jha, S., Lin, S., Page, D., & Ristenpart, T. (2014). Privacy in pharmacogenetics: An End-to-End case study of personalized warfarin dosing. In *23rd USENIX security symposium (USENIX Security 14)* (pp. 17–32). USENIX Association, San Diego, CA.

Fu, K., Cheng, D., Tu, Y., & Zhang, L. (2016). Credit card fraud detection using convolutional neural networks. In Hirose, A., Ozawa, S., Doya, K., Ikeda, K., Lee, M., Liu, D. (eds.) *Neural information processing* (pp. 483–490). Springer, Cham. https://doi.org/10.1007/978-3-319-46675-0_53

Fu, H., Li, Z., Liu, Z., & Wang, Z. (2018). Research on big data digging of hot topics about recycled water use on micro-blog based on particle swarm optimization. *Sustainability*. https://doi.org/10.3390/su10072488.

Ganju, K., Wang, Q., Yang, W., Gunter, C.A., & Borisov, N. (2018). Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security. CCS '18* (pp. 619–633). Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3243734.3243834.

Gentry, C. (2009). *A fully homomorphic encryption scheme*. Stanford, CA: Stanford University.

Google: TensorFlow Privacy. https://github.com/tensorflow/privacy (2019)

Gurney, K. (2018). An introduction to neural networks. CRC Press, Boca Raton, Florida, USA. https://doi.org/10.1201/9781315273570

Herbrich, R. (2017). *Machine learning at amazon. WSDM,535*.

Holohan, N., Braghin, S., Mac Aonghusa, P., & Levacher, K. (2019). Diffprivlib: The ibm differential privacy library. arXiv preprint arXiv:1907.02444

Issa, I., Kamath, S., & Wagner, A.B. (2016). An operational measure of information leakage. In *2016 annual conference on information science and systems (CISS)* (pp. 234–239). https://doi.org/10.1109/CISS.2016.7460507

Jayaraman, B., Evans, D. (2019). Evaluating differentially private machine learning in practice. In *28th USENIX security symposium (USENIX Security 19)* (pp. 1895–1912). USENIX Association, Santa Clara, CA

Julier, S.J. (2003). The spherical simplex unscented transformation. In *Proceedings of the 2003 American control conference, 2003.* (Vol. 3, pp. 2430–2434). https://doi.org/10.1109/ACC.2003.1243439

Julier, S.J., & Uhlmann, J.K. (2002). Reduced sigma point filters for the propagation of means and covariances through nonlinear transformations. In *Proceedings of the 2002 American control conference (IEEE Cat. No. CH37301)* (Vol. 2, pp. 887–892).

Julier, S. J., & Uhlmann, J. K. (1997). New extension of the kalman filter to nonlinear systems. *Signal Processing, Sensor Fusion, and Target Recognition VI, 3068,* 182–193. https://doi.org/10.1117/12.280797.

Julier, S. J., & Uhlmann, J. K. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE, 92*(3), 401–422. https://doi.org/10.1109/JPROC.2003.823141.

Kaggle. https://www.kaggle.com

Kaggle: Acquire Valued Shoppers Challenge. https://www.kaggle.com/c/acquire-valued-shopperschallenge/data (2014)

Kaggle: Netflix Prize Dataset. https://www.kaggle.com/netflix-inc/netflix-prize-data (2006)

Kandepu, R., Foss, B., & Imsland, L. (2008). Applying the unscented kalman filter for nonlinear state estimation. *Journal of Process Control, 18*(7), 753–768. https://doi.org/10.1016/j.jprocont.2007.11.004.

Kim, J., & Lee, K. (2021). Unscented kalman filter-aided long short-term memory approach for wind nowcasting. *Aerospace*. https://doi.org/10.3390/aerospace8090236.

Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

Launchbury, J., Archer, D., DuBuisson, T., & Mertens, E. (2014). Application-scale secure multiparty computation. In Shao, Z. (ed.) *Programming languages and systems* (pp. 8–26). Springer, Berlin. https://doi.org/10.1007/978-3-642-54833-8_2

Lee, M., Choi, H., Kim, C., Moon, J., Kim, D., & Lee, D. (2022). Precision motion control of robotized industrial hydraulic excavators via data-driven model inversion. *IEEE Robotics and Automation Letters, 7*(2), 1912–1919. https://doi.org/10.1109/LRA.2022.3142389.

Li, K. & Wu, G. (2022). Randomized approximate class-specific kernel spectral regression analysis for large-scale face verification. *Machine Learning*, 1–55.

Liu, Y., Wen, R., He, X., Salem, A., Zhang, Z., Backes, M., De Cristofaro, E., Fritz, M., & Zhang, Y. (2021). Ml-doctor: Holistic risk assessment of inference attacks against machine learning models. arXiv preprint arXiv:2102.02551

Lowd, D., & Meek, C. (2005). Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on knowledge discovery in data mining. KDD '05* (pp. 641–647). Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/1081870.1081950

Makridis, G., Mavrepis, P., & Kyriazis, D. (2022). A deep learning approach using natural language processing and time-series forecasting towards enhanced food safety. *Machine Learning*. https://doi.org/10.1007/s10994-022-06151-6.

Meinhold, R. J., & Singpurwalla, N. D. (1983). Understanding the kalman filter. *The American Statistician, 37*(2), 123–127. https://doi.org/10.1080/00031305.1983.10482723.

Mining, P.D., & Fakir, S. (2016). Google cloud prediction api documentation. Google Prediction API.

Mironov, I., Talwar, K., & Zhang, L. (2019). Rényi differential privacy of the sampled gaussian mechanism. arXiv preprint arXiv:1908.10530

Network Security Law of China. http://www.xinhuanet.com/politics/2016-11/07/c_1119867015.htm (2016)

Regulation, P. (2018). General data protection regulation.

Ribeiro, M., Grolinger, K., & Capretz, M.A.M. (2015). Mlaas: Machine learning as a service. In *2015 IEEE 14th international conference on machine learning and applications (ICMLA)* (pp. 896–902). https://doi.org/10.1109/ICMLA.2015.152

Roy, A., Sun, J., Mahoney, R., Alonzi, L., Adams, S., & Beling, P. (2018). Deep learning detecting fraud in credit card transactions. In *2018 systems and information engineering design symposium (SIEDS)* (pp. 129–134). https://doi.org/10.1109/SIEDS.2018.8374722

Salem, A., Zhang, Y., Humbert, M., Berrang, P., Fritz, M., & Backes, M. (2018). Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. arXiv preprint arXiv:1806.01246

Shokri, R., & Shmatikov, V. (2015). Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security. CCS '15* (pp. 1310–1321). Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/2810103.2813687

Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)* (pp. 3–18). https://doi.org/10.1109/SP.2017.41

Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., & Ristenpart, T. (2016). Stealing machine learning models via prediction APIs. In *25th USENIX security symposium (USENIX Security 16)* (pp. 601–618). USENIX Association, Austin, TX.

Vaidya, J., Shafiq, B., Basu, A., & Hong, Y. (2013). Differentially private naive bayes classification. In *2013 IEEE/WIC/ACM international joint conferences on web intelligence (WI) and intelligent agent technologies (IAT)* (Vol. 1, pp. 571–576). https://doi.org/10.1109/WI-IAT.2013.80

Wan, E.A., & Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 adaptive systems for signal processing, communications, and control symposium (Cat. No.00EX373)* (pp. 153–158). https://doi.org/10.1109/ASSPCC.2000.882463

Wang, W., Ying, L., & Zhang, J. (2016). On the relation between identifiability, differential privacy, and mutual-information privacy. *IEEE Transactions on Information Theory, 62*(9), 5018–5029. https://doi.org/10.1109/TIT.2016.2584610.

Wang, J., Zhu, R., & Liu, S. (2018). A differentially private unscented kalman filter for streaming data in iot. *IEEE Access, 6,* 6487–6495. https://doi.org/10.1109/ACCESS.2018.2797159.

Wan, E. A., Van Der Merwe, R., & Haykin, S. (2001). The unscented kalman filter. *Kalman filtering and neural networks, 5*(2007), 221–280. https://doi.org/10.1002/0471221546.ch7.

Wittel, G.L., & Wu, S.F. (2004). On attacking statistical spam filters. In *Proceedings of the conference on email and anti-spam (CEAS), Mountain View, CA, USA*.

Wu, X., Fredrikson, M., Jha, S., & Naughton, J.F. (2016). A methodology for formalizing model-inversion attacks. In *2016 IEEE 29th computer security foundations symposium (CSF)* (pp. 355–370). https://doi.org/10.1109/CSF.2016.32

Yeom, S., Giacomelli, I., Fredrikson, M., & Jha, S. (2018). Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)* (pp. 268–282). https://doi.org/10.1109/CSF.2018.00027. IEEE

Yin, Y., Zhang, W., Xu, Y., Zhang, H., Mai, Z., & Yu, L. (2019). Qos prediction for mobile edge service recommendation with auto-encoder. *IEEE Access, 7,* 62312–62324.

Zhao, B.Z.H., Asghar, H.J., Bhaskar, R., & Kaafar, M.A. (2019). On inferring training data attributes in machine learning models. arXiv preprint arXiv:1908.10558

Zhao, B.Z.H., Kaafar, M.A., & Kourtellis, N. (2020). Not one but many tradeoffs: Privacy vs. utility in differentially private machine learning. In *Proceedings of the 2020 ACM SIGSAC conference on cloud computing security workshop. CCSW'20* (pp. 15–26). Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3411495.3421352

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Authors and Affiliations

Kunsheng Tang[1,2] · Ping Li[1] · Yide Song[1] · Tian Luo[1]

Kunsheng Tang
kunsheng.tang@m.scnu.edu.cn

Yide Song
yide.song@m.scnu.edu.cn

Tian Luo
luotian@m.scnu.edu.cn

[1] School of Computer Science, South China Normal University, No.55, West of Zhongshan Avenue, Tianhe District, Guangzhou City 510631, Guangdong Province, China

[2] School of Cyber Science and Technology, University of Science and Technology of China, No. 96, JinZhai Road, Baohe District, Hefei City 230026, Anhui Province, China