Balázs Pejó*, Qiang Tang, and Gergely Biczók

Together or Alone: The Price of Privacy in Collaborative Learning

Abstract: Machine learning algorithms have reached mainstream status and are widely deployed in many applications. The accuracy of such algorithms depends significantly on the size of the underlying training dataset; in reality a small or medium sized organization often does not have the necessary data to train a reasonably accurate model. For such organizations, a realistic solution is to train their machine learning models based on their joint dataset (which is a union of the individual ones). Unfortunately, privacy concerns prevent them from straightforwardly doing so. While a number of privacy-preserving solutions exist for collaborating organizations to securely aggregate the parameters in the process of training the models, we are not aware of any work that provides a rational framework for the participants to precisely balance the privacy loss and accuracy gain in their collaboration.

In this paper, by focusing on a two-player setting, we model the collaborative training process as a two-player game where each player aims to achieve higher accuracy while preserving the privacy of its own dataset. We introduce the notion of *Price of Privacy*, a novel approach for measuring the impact of privacy protection on the accuracy in the proposed framework. Furthermore, we develop a game-theoretical model for different player types, and then either find or prove the existence of a Nash Equilibrium with regard to the strength of privacy protection for each player. Using recommendation systems as our main use case, we demonstrate how two players can make practical use of the proposed theoretical framework, including setting up the parameters and approximating the non-trivial Nash Equilibrium.

Keywords: Privacy, Game Theory, Machine Learning, Recommendation Systems

DOI 10.2478/popets-2019-0019
Received 2018-08-31; revised 2018-12-15; accepted 2018-12-16.

***Corresponding Author: Balázs Pejó:** University of Luxembourg, E-mail: balazs.pejo@uni.lu
Qiang Tang: Luxembourg Institute of Science and Technology, E-mail: qiang.tang@list.lu
Gergely Biczók: CrySyS Lab, Dept. of Networked Systems and Services, Budapest Univ. of Technology and Economics, E-mail: biczok@crysys.hu

1 Introduction

Machine Learning (ML) (the process of learning from data and making predictions about it by building a model), has received much attention over the last decade, mostly due to its vast application range such as recommendation services, medicine, speech recognition, banking, gaming, driving, and more. For ML tasks, it is widely known that more training data will lead to a more accurate model. Unfortunately, most organizations do not possess a dataset as large as Netflix's[1] or Amazon's[2]. In such a situation, to obtain a relatively accurate model, a natural solution would be to aggregate all the data from different organizations on a centralized server and train on the global dataset as seen on the left side of Figure 1. This approach is efficient, however, data holders have a valid privacy concern about sharing their data, particularly with new privacy regulations such as the European General Data Protection Regulation[3] (GDPR). Therefore, improving ML via straightforward data aggregation is likely undesirable and potentially unlawful in reality. Various privacy concerns exists with regard to ML (e.g., the privacy of the input to the training or the privacy of the trained model); in this paper, we focus on the privacy of the input for individual data contributors.

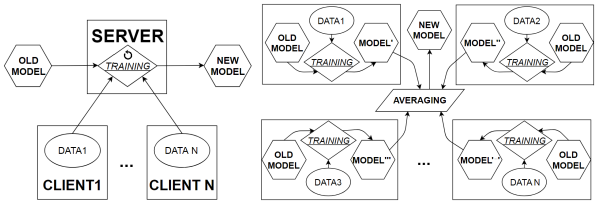

Fig. 1. Centralized (left) and Distributed (right) Learning

In the literature, Privacy Preserving Distributed ML [4, 11, 13, 14, 16] have been proposed to solve this

1 https://www.netflix.com
2 https://www.primevideo.com
3 https://eugdpr.org

problem by training the model locally and safely aggregating all the local updates, illustrated on the right side of Figure 1. On the other hand, these approaches' efficiency depend on the number of participants and the sample sizes as we highlight this in the related works.

In this paper, we are interested in a scenario with two participants, each of whom possesses a significant amount of data and would like to obtain a more accurate model than what they would obtain if training was carried out in isolation. It is clear that the players will only be interested in collaboration if they can actually benefit from each other. To this end, we simply assume that the players have already evaluated the quality of each other's datasets to make sure training together is beneficial for both of them before the collaboration. How such evaluation should be done is out of scope for our research; there are best practices already established in the field [5]. Most of the ML papers, including privacy-preserving ones, implicitly make this assumption.

1.1 Problem Statement

Collaborative Machine Learning (CoL) will increase the model accuracy, but at the cost of leaking some information about the players' datasets to each other. To mitigate the information leakage, players can apply some privacy-preserving mechanisms, e.g., calibrating and adding some noise or deleting some sensitive attributes. Many "solutions" have been proposed, as surveyed in the related work. In most of them, the players are not provided with the option of choosing their own privacy parameters. Clearly, there is a gap between these solutions and reality, where players will have different preferences to privacy and utility and may want to dynamically set the parameters.

To bridge this gap, we consider the parties involved as rational players and model their collaboration as a two-player game. In our setting, players have their own trade-offs with respect to their privacy and expected utility and can flexibly set their own privacy parameters. The central research problem is to propose a general game theoretical model and find a Nash Equilibrium [6]. Moreover, given a specific CoL task, we should answer the following core questions:

- What are the potential ranges for privacy parameters that make the CoL model is more accurate than training alone?
- What is the optimal privacy parameter (which results in the highest payoff)?
- With this optimal parameter, how much accuracy is lost overall due to the applied privacy-preserving mechanisms?

1.2 Contribution

We first propose a two-player game theoretical model for CoL (a training process via an arbitrary training algorithm between two players). We profile the players and analyze their best response strategies and the equilibria of the designed game. Inspired by the notion of Price of Anarchy [9], we define *Price of Privacy*, which is a new way of measuring the accuracy degradation due to privacy protection. Then, we demonstrate the usage of the model via a recommender use case, where two players improve their own recommendation accuracy by leveraging on each other's dataset. It is worth noting that this is indeed a representative example since the used Stochastic Gradient Descent (SGD) optimization process is a universal procedure widely used in ML tasks [5]. For illustration purposes, we consider two privacy preserving mechanisms, including attribute deletion and differential privacy [2]. Based on heuristics, we demonstrate how to approximate the privacy-accuracy trade-off functions, which lie in the core of the proposed theoretical model and determine how the players should set the parameters, and illustrate the practically obtained Nash Equilibrium [6].

We would like to emphasize that approximating the privacy-accuracy trade-off function is a very realistic choice in applying the proposed theoretical model. Scientifically, we may want to use cryptographic techniques such as secure two-party computation protocols to precisely compute these parameters. However, this is undesirable due to the incurred complexity. In order to reduce complexity, most deployed ML systems implement heuristics, such as approximating the parameters in Stochastic Gradient Descent [5].

1.3 Organization

In Section 2, we review some basic concepts used throughout the paper such as game theory and differential privacy. In Section 3, we introduce the CoL game, explain the parameters, define the concept of *Price of Privacy*; moreover, we show how to apply the CoL game framework in practice. In Section 4, we provide a theoretical analysis of the proposed game and investigate the Nash Equilibria. In Section 5, we introduce the rec-

ommender use case and describe two example privacy-preserving mechanisms. In Section 6, for the recommender use case, we demonstrate how to measure the the privacy-accuracy trade-off function over the joint dataset. In Section 7, first, we interpolate this trade-off function and determine the corresponding numerical equilibria, then we show how to approximate the trade-off function via heuristics and study its impact on the equilibria. In Section 8, we review the the related works from the perspective of game theory and privacy-preserving ML. In Section 9, we conclude the paper.

As we use multiple well-known concepts through the paper, we provide a summary of abbreviations in Appendix A to improve readability.

2 Preliminaries

In this section, we introduce differential privacy and the game theoretic terminology used in the paper.

2.1 Differential Privacy

DP [2] have been used widely in the literature. It classically quantifies the privacy of a mechanism in terms of parameters ε:

Definition (ε-differential privacy [2])**.** *An algorithm $\mathcal{A}$ is ε-DP ($\varepsilon \in [0, \infty)$]) if for any two datasets D_1 and D_2 that differ on a single element and for any set of possible outputs O:*

$$\Pr(\mathcal{A}(D_1) \in O) \leq e^{\varepsilon} \cdot \Pr(\mathcal{A}(D_2) \in O)$$

DP gives a strong guarantee that presence or absence of a single data point will not change the final output of the algorithm significantly. Furthermore, the combination of DP mechanisms also satisfies DP:

Theorem (Composition Theorem [2])**.** *If the mechanisms $\mathcal{A}_i$ are ε_i-DP, then any sequential combination of them is $\sum_i \varepsilon_i$-DP.*

To achieve DP, noise must be added to the output of the algorithm. In most cases, this noise is drawn from a Laplacian distribution and it is proportional to the sensitivity of the algorithm itself:

Theorem (Laplace Mechanism [2])**.** *For $f : \mathcal{D} \to \mathbb{R}^k$, if s is the sensitivity of f (i.e., $s = \max_{D_1, D_2} ||f(D_1) - f(D_2)||$ for any two datasets D_1 and D_2 that differ on a single element) then the mechanism $\mathcal{A}(D) = f(D) + Lap(\frac{s}{\varepsilon})$ with independently generated noise to each of the k outputs enjoys ε-DP.*

2.2 Game Theory

Game theory [6] is "the study of mathematical models of conflict between intelligent, rational decision-makers". Almost every multi-party interaction can be modeled as a game. In our case, these decision makers are the participants (players) of CoL.

Definition (Game)**.** *A normal form representation of a game is a tuple $\langle \mathcal{N}, \Sigma, \mathcal{U} \rangle$, where $\mathcal{N} = \{1, \dots, m\}$ is the set of players, $\Sigma = \{S_1, \dots, S_m\}$ where $S_i = \{s_1, s_2, \dots\}$ is the set of actions for player i and $\mathcal{U} = \{u_1, \dots, u_m\}$ is the set of payoff functions.*

A Best Response (BR) strategy gives the most favorable outcome for a player, taking other players' strategies as given:

Definition (Best Response)**.** *For a game $\langle \mathcal{N}, \Sigma, \mathcal{U} \rangle$ the BR strategy for player i for a given strategy vector $s_{-i} = (s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_m)$ is $\hat{s}_i$ if $\forall s_{i_j} \in S_i$: $u_i(\hat{s}_i, s_{-i}) \geq u_i(s_{i_j}, s_{-i})$.*

A Nash Equilibrium (NE) is a strategy vector where all the player's strategies are BR strategies. In other words, in a NE state every player makes the best/optimal decision for itself as long as the others' choices remain unchanged:

Definition (Nash Equilibrium)**.** *A pure-strategy NE of a game $\langle \mathcal{N}, \Sigma, \mathcal{U} \rangle$ is a strategy vector $(s_1^*, \dots, s_m^*)$ where $s_i^* \in S_i$, such that for each player i $\forall s_{i_j} \in S_i$: $u_i(s_i^*, s_{-i}^*) \geq u_i(s_{i_j}, s_{-i}^*)$ where $s_{-i}^* = (s_1^*, \dots, s_{i-1}^*, s_{i+1}^*, \dots, s_m^*)$.*

NE provides a way of predicting what will happen if several entities are making decisions at the same time where the outcome depends on the decisions of the others. The existence of a NE means no player will gain more by unilaterally changing its strategy at this unique state.

Another game-theoretic concept is Social Optimum, which is a strategy vector that maximizes social welfare:

Definition (Social Optimum)**.** *The Social Optimum of a game $\langle \mathcal{N}, \Sigma, \mathcal{U} \rangle$ is a strategy vector $(s_1', \dots, s_m')$ where*

$s'_i \in S_i$, *such that*

$$\max_{s_1 \in S_1, \ldots, s_m \in S_m} \sum_{n \in \mathcal{N}} u_n(s_1, \ldots, s_m) = \sum_{n \in \mathcal{N}} u_n(s'_1, \ldots, s'_m)$$

Despite the fact that no one can do better by changing strategy, NEs are not necessarily Social Optimums (as an example see Prisoner's Dilemma [6]). Price of Anarchy [9] measures the ratio between these two: how the efficiency of a system degrades due to the selfish behavior of its players:

Definition (Price of Anarchy [9])**.** *Priva of Anarchy*[4] *of a game* $\langle \mathcal{N}, \Sigma, \mathcal{U} \rangle$ *is*

$$PoA := \frac{\max_{s \in S} \sum_{n \in \mathcal{N}} u_n(s)}{\min_{s^* \in S^*} \sum_{n \in \mathcal{N}} u_n(s^*)}$$

where $S = S_1 \times \cdots \times S_m$ *is the set of all possible outcomes while* S^* *is the set of NEs.*

3 Game Theoretic Model

In this section, we describe the CoL game and show how it can be used in practice. Specifically, the CoL game captures the actions of two privacy-aware data holders in the scenario of applying an arbitrary privacy preserving mechanism and training algorithm on their datasets. We define the corresponding utility functions and elaborate on their components. Furthermore, we introduce the notion of *Price of Privacy*, a novel measure of the effect of privacy protection on the accuracy of players.

3.1 The Collaborative Learning Game

At a high level, the players' goal in the CoL game is to maximize their utility, which is a function of the model accuracy and the privacy loss. We do not consider the adversarial aspect of players, hence the gain includes only the accuracy improvements on the model for a particular player as benefit (without the accuracy decrease of the other player[5]) while the cost is private information leakage: the trained model leaks some information about the local dataset used for training.

Players only choose the privacy parameters for a predetermined privacy preserving method M (rather than choosing the method with the parameter). This means each M corresponds to a different game with a different definition of privacy, rather than having one game where the players' actions are deciding which mechanism to use and to what extent. This is a restricted scenario, nonetheless, even this scenario barely lends itself to purely analytical treatment; it is already not straightforward to derive the exact NE.

The variables of the CoL game are listed in Table 1, where the accuracy is measured as the prediction error of the trained model: lower θ_n and Φ_n^M correspond to a more accurate model. Maximal privacy protection is represented via $p_n = 1$, while $p_n = 0$ means no protection for player n. The benefit and the privacy loss are not on the same scale as the first depends on the accuracy while the latter on information loss. To make them comparable, we introduce weight parameters: the benefit function is multiplied with the accuracy weight $B_n > 0$, while the privacy loss function is multiplied with the privacy weight $C_n^M \geq 0$.

Variable	Meaning
M	Privacy mechanism applied by the players
p_n	Privacy parameter for player n
C_n^M	Privacy weight for player n
B_n	Accuracy weight for player n
θ_n	Model error by training alone for player n
$\Phi_n^M(p_1, p_2)$	Model error by training together for player n
$b(\theta_n, \Phi_n^M)$	Benefit function for player n
$c^M(p_n)$	Privacy loss function for player n

Table 1. Parameters of the CoL game

The accuracy (error) achieved by training together ($\Phi_n^M(p_1, p_2)$) naturally depends also on the datasets and the used algorithm besides the privacy parameters p_n and the corresponding privacy mechanism M. However, for simplicity we abstract them since it does not affect our theoretical analysis as long as Φ_n^M is symbolic.

Definition 1 (Collaborative Learning Game)**.** *The CoL game is a tuple* $\langle \mathcal{N}, \Sigma, \mathcal{U} \rangle$*, where the set of players is* $\mathcal{N} = \{1, 2\}$*, their actions are* $\Sigma = \{p_1, p_2\}$ *where* $p_1, p_2 \in [0, 1]$ *while their utility functions are* $\mathcal{U} = \{u_1, u_2\}$ *such that for* $n \in \mathcal{N}$*:*

$$u_n(p_1, p_2) = B_n \cdot b(\theta_n, \Phi_n^M(p_1, p_2)) - C_n^M \cdot c^M(p_n) \quad (1)$$

The CoL game is of symmetric information, i.e., the introduced parameters are public knowledge (i.e., M, B_n, C_n^M, b, c^M, θ_n and Φ_n^M) except for the actions of the

4 Sometimes referred to as *Price of Stability*.

5 Extending the game for competing companies is an interesting future direction.

players (i.e., p_n). Moreover, we do not consider any negative effect of the training such as time or electricity consumption, however, such variables may be introduced to the model in the future.

In the following, whenever possible, we simplify the notion C_n^M, c^M and Φ_n^M by removing the symbol M to use C_n, c and Φ_n respectively. We only need to keep in mind that these functions depend on the underlying privacy-preserving mechanism M in the implementation.

Privacy Loss Function $c(p_n)$**.** This function represents the loss due to private data leakage. We define c with the following natural properties:

Definition 2 (Privacy loss function)**.** $c : [0,1] \to [0,1]$ *such that it is continuous and twice differentiable,* $c(0) = 1$*,* $c(1) = 0$ *and* $\partial_{p_n} c < 0$*.*

This definition indicates that the maximal potential leakage is 1 which corresponds to no protection at all, while maximal privacy protection corresponds to zero privacy loss. Furthermore, c is monotone decreasing which means stronger privacy protection corresponds to less privacy loss.

Benefit Function $b(\theta_n, \Phi_n)$**.** The benefit function has two inputs: the error achieved by training alone (θ_n) and when both players train together (Φ_n). Since a rational player would not collaborate to end up with a model with higher error, we are only interested in the case when $\Phi_n < \theta_n$. Hence, we define b with the following natural properties:

Definition 3 (Benefit function)**.** $b : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}_0^+$ *such that it is continuous and twice differentiable,* $\partial_{p_n} b \leq 0$ *and* $b(\theta_n, \Phi_n) = 0$ *if* $\theta_n \leq \Phi_n$*.*

This definition indicates that there is no benefit when the error of the model trained together is higher than the error of the model trained alone. Furthermore, since b is monotone decreasing in p_n, stronger privacy protection results in lower benefit (due to the increased error).

Privacy-Accuracy Trade-off Function $\Phi_n(p_1, p_2)$**.** Φ_n plays a crucial role in the benefit function b. However, the function of how a privacy protection mechanism affects a complex training algorithm (and consequently the accuracy) is unique for each dataset and algorithm. Although we measure it in Section 6, and approximate it in Section 7 for a recommendation system use case, in general the exact form of Φ_n is unknown. On the other hand, some properties are expected:

Definition 4 (Privacy-accuracy trade-off function)**.** $\Phi_n : [0,1] \times [0,1] \to \mathbb{R}^+$ *such that it is continuous, twice differentiable and:*

- $\exists m \in \mathcal{N} : p_m = 1 \Rightarrow \forall n \in \mathcal{N} : \Phi_n(p_1, p_2) \geq \theta_n$
- $\forall n, m \in \mathcal{N} : \partial_{p_m} \Phi_n > 0$
- $\forall n \in \mathcal{N} : \theta_n > \Phi_n(0,0)$

The first property means that maximal privacy protection cannot result in lower error than training alone for both players. The second property indicates that higher privacy protection corresponds to higher error since Φ_n is monotone increasing in both p_1 and p_2. The last property ensures that training together with no privacy corresponds to lower error than training alone.

3.2 The Price of Privacy

Inspired by the notion of Price of Anarchy [9], we define *Price of Privacy* (*PoP*) to measure the accuracy loss due to privacy constraints:

Definition 5 (Price of Privacy)**.** *PoP measures the overall effect of privacy protection on the accuracy:*

$$PoP(p_1^*, p_2^*) := 1 - \frac{\sum_n b(\theta_n, \Phi_n(p_1^*, p_2^*))}{\sum_n b(\theta_n, \Phi_n(0,0))} \qquad (2)$$

The quotient is between the total accuracy improvement in a NE (p_1^*, p_2^*) *and the total accuracy improvement without privacy protection.*

Due to the Definition 3 and 4, $PoP \in [0,1]$, where 0 corresponds the lowest possible error which can be achieved via collaboration with no privacy, while 1 corresponds to the highest possible error which can be achieved by training alone. In other words, *Price of Privacy* evaluates the benefit of a given equilibrium: the lower its value, the lower the error achieved by collaboration.

Note that while Price of Anarchy characterizes a game as a whole, *Price of Privacy* is a property of a NE. Also, since Φ_n can only be estimated in a real-world scenario, the players can only approximate the value of *PoP*, which would then measure the efficiency of the collaboration.

3.3 Applying the Collaborative Learning Game

Given that the actual value of Φ_n is required to compute the optimal strategies, Φ_n has to be numerically evaluated for putting the CoL game to practical use.

Different from other parameters which can be set freely, the impact of the privacy-preserving mechanism M on the joint accuracy (and thus Φ_n) is determined by both datasets. Precisely computing this function requires access to the joint dataset; thus, it raises the very privacy concern which we want to mitigate in the first place. To break this loop, we propose to adopt an approximation approach for applying the model. To this end, we provide a heuristic solution in Section 7. The necessary steps for setting up the collaboration are presented in Figure 2:

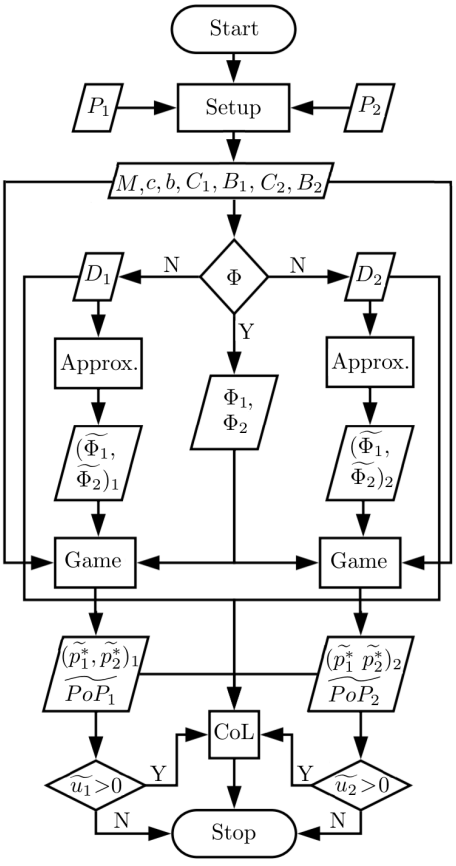

Fig. 2. Process diagram for applying the CoL game framework

- **Initialization**: The players have their datasets D_n with corresponding privacy policies P_n.
- **Setup**: Based on the privacy policies P_n, players determine which privacy preserving method M to use, what benefit b and cost c functions to apply, and what should the corresponding weight parameters C_1, B_1, C_2, B_2 be.
- **Approximation**: If the privacy-accuracy trade-off function Φ_n is not (entirely) known, players must approximate it either by interpolating it from measurement points (assuming measurements can be obtained) or based on their own local datasets D_n.
- **Game**: The players determine the NE $(p_1^*, p_2^*)_n$ and its corresponding *Price of Privacy* via the CoL game by using either the true or the approximated privacy-accuracy trade-off function with the parameters determined in "Setup" step.
- **Collaboration**: If the NE suggests that training together is beneficial for both participants, then they collaborate using their datasets D_n with the optimal privacy parameter p_n^*.

Remarks on Φ. It is clear that Φ is specific to the actual application scenario, therefore it can only be evaluated on a use case basis. We present our recommendation system use case in Section 5. Then, having defined both the training algorithm and datasets, we establish measurements of Φ for different parameter values in Section 6. Using these empirical results, we approximate Φ and compute the NEs in Section 7.

4 Equilibrium Analysis

In this section, we characterize the NEs for a simple and-more complex cases of the CoL game. We derive symbolic NEs in closed form for the case where exactly one of the players is privacy-concerned (i.e., Collaboration-as-a-Service (CaaS) scenario). Next, we prove the existence of a pure strategy NE in the general case, where both players are privacy-concerned to a given degree. To preserve clarity, all mathematical proofs for theorems in this section are given in Appendix B.

The simplest NE of the CoL game is no collaboration:

Theorem 1. *Applying maximal privacy protection (training alone) in the CoL game is a NE:* $(p_1^*, p_2^*) = (1, 1)$.

Clearly, when the players train alone there will be no improvement in accuracy. This means that the *Price of Privacy* for this NE is the maximum 1: the entire potential accuracy improvement is lost due to privacy protection. This finding seemingly contradicts [1], which states that all players refraining to participate cannot be an equilibrium. There is a significant difference though; estimation cost is a public good in [1], while in our case accuracy is private and each participant has a base accuracy level obtained by training alone.

4.1 Player Types

Based on the properties of CoL game, two natural expectations arise:

- A player prefers collaboration if it values accuracy significantly more than privacy ($B_n \gg C_n$).
- A player prefers training alone if it values accuracy significantly less than privacy ($B_n \ll C_n$).

These intuitions are captured in the following two lemmas:

Lemma 1. *$\exists \alpha_n \geq 0$ such that if $\frac{C_n}{B_n} \leq \alpha_n$ for player n than its BR is $\hat{p}_n = 0$.*

Lemma 2. *$\exists \beta_n \geq 0$ such that if $\frac{C_n}{B_n} \geq \beta_n$ for player n then its BR is to set $\hat{p}_n = 1$.*

The questions we are interested in answering are: *what are the exact values of α_n and β_n* and *what is the NE in case $\alpha_n \leq \frac{C_n}{B_n} \leq \beta_n$*. Based on the ratio $\frac{C_n}{B_n} \in [0, \infty]$, we define two types of players:

- **Unconcerned**: This type of player cares only about accuracy. This represents the case when $\frac{C_n}{B_n} = 0$: the privacy weight for player n is zero ($C_n = 0$).
- **Concerned**: This player is more privacy-aware, as the privacy loss is present in its utility function. This represents the case when $\frac{C_n}{B_n} > 0$.

This information is available to both players as the CoL game is a symmetric information game: both players know which type of player they face.

4.2 One Player is Privacy Concerned

Definition 6 (Collaboration-as-a-Service)**.** *In a CaaS scenario one player acts as a for-profit service provider of collaborative training without privacy concerns, i.e., its privacy weight is 0.*

Example. *Imagine a company who offers CaaS for her own profit (Player 2). The CaaS provider does not apply any privacy-preserving mechanism (see Theorem 2). Any interested party (Player 1) who wants to to boost its accuracy can use this service. At the same time, Player 1 requires additional privacy protection (besides the inherent complexity of the training algorithm) to prevent her own data from leaking.*

Theorem 2 (Training as an unconcerned player)**.** *If player n is unconcerned ($C_n = 0$) then its BR is to collaborate without any privacy protection: $\hat{p}_n = 0$.*

Remark. When both players are unconcerned ($C_1 = C_2 = 0$), $(p_1^*, p_2^*) = (0, 0)$ is a NE. The corresponding *Price of Privacy* value is 0 as no accuracy is lost due to privacy protection.

As a result, the unconcerned player do not apply any privacy-preserving mechanism. Without loss of generality we assume Player 2 is unconcerned, so its BR is $\hat{p}_2 = 0$. This allows us to make the following simplifications: $\Phi(p_1) := \Phi_1(p_1, \hat{p}_2)$, $b(p_1) := b(\theta_1, \Phi(p_1, \hat{p}_2))$ and $u(p_1) := u_1(p_1, \hat{p}_2)$ while $f' = \partial_{p_1} f$ and $f'' = \partial^2_{p_1} f$.

Theorem 3 (Training with an unconcerned player)**.** *A NE of the CoL game when Player 1 is concerned ($C_1 > 0$) while Player 2 is unconcerned ($C_2 = 0$) is $(p_1^*, p_2^*) = (\rho, 0)$ where ρ is defined by Equation (3) where $[\cdot]^{-1}$ is the inverse function of $[\cdot]$ and $r = \frac{C_1}{B_1}$:*

$$\rho = \begin{cases} \left[\frac{b'\Phi'}{c'}\right]^{-1}(r) & \textit{if } u''(\rho) < 0,\ \rho \in [0,1],\ u(\rho) > 0 \\ 0 & \textit{if } b(0) > r \\ 1 & \textit{otherwise} \end{cases} \tag{3}$$

The three possible NEs when Player 2 is unconcerned, and the corresponding *Price of Privacy* values are:

- If the possible maximal benefit is higher than the weight ratio ($b(0) > \frac{C_1}{B_1}$) for Player 1, this player should train without any privacy protection as $(p_1^*, p_2^*) = (0, 0)$ is a NE. In this case $PoP = 0$.
- If all the required conditions in Theorem 3 hold, $(p_1^*, p_2^*) = \left(\left[\frac{b'\Phi'}{c'}\right]^{-1}\left(\frac{C_1}{B_1}\right), 0\right)$ is a NE with

$$PoP = 1 - \frac{\sum_n b\left(\theta_n, \Phi_n\left(\left[\frac{b'\Phi'}{c'}\right]^{-1}\left(\frac{C_1}{B_1}\right), 0\right)\right)}{\sum_n b(\theta_n, \Phi_n(0,0))}$$

- Otherwise $(p_1^*, p_2^*) = (1, 0)$ is a NE. In this case, when one player apply maximal privacy protection (Player 1), the other player's utility cannot be positive due to the Definition 4. Furthermore, since Player 2 is privacy unconcerned, its actual payoff is 0 independently of its action. As a result, $(p_1^*, p_2^*) = (1, p_2)$ is a NE for all $p_2 \in [0, 1]$ as they all correspond to the same 0 payoff. For simplicity, we use $(1, 1)$ to represent this case (the players train alone), where the corresponding PoP value is 1.

This result is quite abstract because all the components of the utility function are treated symbolically. However, even if we specify the benefit and the privacy loss functions, the privacy-accuracy trade-off function Φ_n would still be unknown due to the unspecified training algorithm. We show this in the next Corollary where we set b and c to be linear, as it is shown in Equation (4) where $[\cdot]^+ = \max\{\cdot, 0\}$.

$$\begin{aligned} c(p_n) &:= 1 - p_n \\ b(\theta_n, \Phi_n(p_1, p_2)) &:= [\theta_n - \Phi_n(p_1, p_2)]^+ \end{aligned} \tag{4}$$

Corollary. *With the same notations as in Theorem 3 and with the benefit and privacy loss functions defined in Eq. (4), $(p_1^*, p_2^*) = (\rho, 0)$ is a NE when $C_1 > 0$ and $C_2 = 0$ if ρ is:*

$$\rho = \begin{cases} \mathbf{0} & \textit{if} \quad r \leq \theta_1 - \Phi(0) \\ [\Phi']^{-1}(r) & \textit{if} \quad \Phi''(r) < 0,\ u_1(\rho, 0) > 0,\ \rho \in [0, 1] \\ \mathbf{1} & \textit{otherwise} \end{cases}$$

To compute the numerical NE and the corresponding *Price of Privacy* of the CoL game, we need to define the function Φ_n. While for simpler training algorithms (such as [1, 7]) Φ_n is known, for more complex algorithms it can only be approximated. We demonstrate a two potential approximation method in Section 7: interpolation and Self-Division.

4.3 Both Players are Privacy Concerned

Now we consider the general case when both players' privacy weights are non-zero. We prove the existence of a pure-strategy NE besides the trivial $(p_1^*, p_2^*) = (1, 1)$; we utilize the chain rule of derivation for higher dimensions and a result from the theory of potential games [12].

Lemma (Chain Rule)**.** *If $f : \mathbb{R}^2 \to \mathbb{R}$ and $g : \mathbb{R}^2 \to \mathbb{R}$ are differentiable functions, then*

$$\forall i \in [1, 2] : \partial_{x_i} f(x, g(x_1, x_2)) = \partial_g f \cdot \partial_{x_i} g(x_1, x_2)$$

Definition (Potential Game [12])**.** *A two-player game G is a potential game if the mixed second order partial derivative of the utility functions are equal:*

$$\partial_{p_1} \partial_{p_2} u_1 = \partial_{p_1} \partial_{p_2} u_2 \tag{5}$$

Theorem (Monderer & Shapley [12])**.** *Every potential game admits at least one pure-strategy NE.*

Now we can state the theorem which holds even if both players are privacy-concerned:

Theorem 4. *The CoL game has at least one non-trivial pure-strategy NE if*

$$\partial_\Phi^2 b \cdot (\partial_{p_1} \Phi_1 - \partial_{p_2} \Phi_2) = \partial_\Phi b \cdot (\partial_{p_1} \partial_{p_2} \Phi_2 - \partial_{p_1} \partial_{p_2} \Phi_1) \tag{6}$$

Corollary (1). *If we assume $\partial_{p_1}^i \Phi_1 = \partial_{p_2}^i \Phi_2$ for $i \in \{1, 2\}$ then Theorem 4 holds.*

The condition on the derivatives of Φ_n in Corollary 1 means that the player's accuracy changes the same way in relation to their own privacy parameter, independently from the other player's privacy parameter. In Section 6 we measure the error for multiple privacy parameter values and find that this is indeed the case. Moreover, we find that $\Phi_1 \approx \Phi_2$ when the players have equal dataset sizes.[6]

Remarks on equilibria. Note that a non-trivial pure-strategy NE does not necessarily correspond to positive payoffs: if $\exists n \in \mathcal{N} : u_n(p_1^*, p_2^*) < 0$ then player n would rather not collaborate. Instead, it would train alone and gain 0. This situation is plausible since 0 utility corresponds to the accuracy of the locally trained model. As such, an actual accuracy improvement is not trivial by collaboration, especially with additional privacy concerns (as we demonstrate later in Section 6).

5 Use Case: Recommendation System

In this section, we describe our Recommendation System (RecSys) use case, where Matrix Factorization (MF) is performed via Stochastic Gradient Descent (SGD) [5]. Then, we introduce two example privacy-preserving mechanisms: Suppression (*Sup*) and bounded DP (*bDP*).

5.1 The Learning Process

If there are only two parties in a distributed learning scenario (right side of Figure 1), the trained model reveals some information about both players' dataset. Hence, in our scenario, the players train the same model iter-

6 Note that these findings are empirical: based on the specific datasets/algorithm we used. For more details see Section 6.

atively without any safe aggregation. If there are only two participants, parallelization does not improve the efficiency much, so the players are training the model sequentially as seen in Figure 3. The problem of information leakage is tackled with privacy-preserving mechanisms.

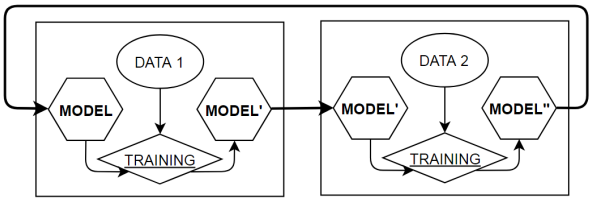

Fig. 3. Learning sequentially

Our use case is a RecSys scenario. We assume that players hold a user-item rating matrix with a common item-set $I = I_1 = I_2$ and disjoint user-set: $U_1 \cap U_2 = \emptyset$ where $U = U_1 \cup U_2$. As usual, $r_{ui} \in R_{|U|\times|I|}$ refers to the rating user u gives item i.

The goal of the learning algorithm is to find the items that users desire. One of the most widespread method to do that is MF [8] as seen in Figure 4: finding $P_{|U|\times\kappa}$ and $Q_{\kappa\times|I|}$ such that $P \cdot Q \approx R$. As the user-sets are disjoint, players only need to share the item feature matrix Q.

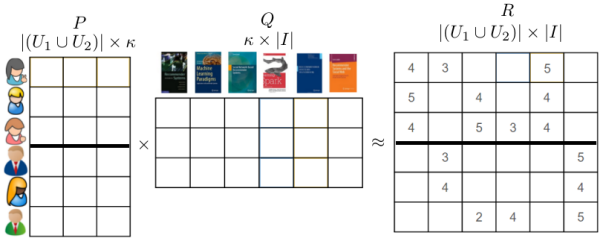

Fig. 4. RecSys scenario

The goal of this MF algorithm is to minimize the error between the prediction and the observed ratings as described in Equation (7) where λ is the regularization parameter, while p_u (q_i) is the corresponding row (column) in P (Q) for r_{ui}.

$$\min_{P,Q} \sum_{r_{ui}\in R} (r_{ui} - p_u q_i)^2 + \lambda(||p_u||^2 + ||q_i||^2) \qquad (7)$$

One of the most popular techniques to minimize this formula is SGD [5]. It works by iteratively selecting a random rating $r_{ui} \in R$ and updating the corresponding factor vectors according to Equation (8) where $e_{iu} = p_u q_i - r_{ui}$ and γ is the learning rate.

$$\begin{aligned} p'_u &:= p_u + \gamma(e_{ui} q_j - \lambda p_u) \\ q'_i &:= q_i + \gamma(e_{ui} p_u - \lambda q_i) \end{aligned} \qquad (8)$$

Since we use SGD, the training process shown in Figure 3 is essentially equivalent to mini-batch learning where the batches are the datasets of the players. As we use RecSys as an illustrative example, we simplify it: we assume that players share the learning algorithm which is embedded with the necessary parameters such as learning rate γ, regularization parameter λ, number of features κ, and maximum number of iterations ι. Notations are summarized in Table 2.

Variable	Meaning
U	Joint user-set
I	Itemset
R	Rating matrix
r_{ui}	Rating of user u for item i
P, Q	Feature matrices
γ	Learning rate
λ	Regularization parameter
κ	Number of features
ι	Number of iterations

Table 2. RecSys parameters

5.2 Privacy Preserving Mechanisms

We focus on input manipulation for privacy preservation as we are concerned with input data privacy. In fact, [3] concluded that input perturbation achieves the most efficient accuracy-privacy trade-off amongst various DP mechanisms. We investigate *Sup* and *bDP* as available mechanisms.

Suppression. *Sup* essentially chooses a subset of the dataset to be used for training together. *Sup* can be used to remove sensitive data from the dataset, so even if the other player can reconstruct the dataset from the trained model, the removed part remains fully protected.

Definition 7 (Suppression). *Sup removes input data from the original dataset to protect it from information leakage resulting from the model or the learning process. The size of reduction is determined by the privacy*

parameter $p \in [0,1]$, i.e., p is proportional to the data removed[7].

Bounded Differential Privacy. To apply *bDP*, we must determine the sensitivity of the machine learning algorithm first:

Theorem 5 (Sensitivity of RecSys)**.** *The sensitivity S of the introduced RecSys scenario is*

$$S \leq \kappa \cdot \iota \cdot \gamma \cdot (\Delta r \cdot p_{max} - \lambda \cdot q_{max}) \tag{9}$$

Proof Theorem 5. See Appendix C. □

Now, we consider the *bDP* mechanism [3]:

Definition 8 (bounded DP)**.** *bDP aims to hide the value of a rating. To achieve ε-bDP, each rating is modified as it is shown in Equation (10) where $L(x)$ is a Laplacian noise with 0 mean and x variance.*

$$r'_{ui} := \begin{cases} r_{max} & \text{if} \quad r_{ui} + L(\frac{S}{\varepsilon}) \geq r_{max} \\ r_{min} & \text{if} \quad r_{ui} + L(\frac{S}{\varepsilon}) \leq r_{min} \\ r_{ui} + L(\frac{S}{\varepsilon}) & \text{otherwise} \end{cases} \tag{10}$$

5.3 Unifying Privacy Parameters

These approaches are hard to compare since they focus on protecting different things. *Sup* aims to provide maximal privacy for some of the data while leaving the rest unprotected. On the other hand, *bDP* provides an equal amount of privacy for all the data based on the parameter ε. In the CoL game we defined the privacy parameter on a scale 0 to 1, therefore the specific parameters of *Sup* and *bDP* must be mapped to $[0,1]$ where $p = 0$ means no privacy, while $p = 1$ stands for full privacy protection. For *Sup* the value p is straightforward: p represents the portion of data removed. Hiding the dataset in whole (100% protection) means $p = 1$ while if the whole dataset is used for training (0% protection) then $p = 0$.

In case of *bDP*, 100% privacy ($p = 1$) is achieved when $\varepsilon = 0$ (infinite noise) while $\varepsilon \approx \infty$ corresponds to zero noise ($p = 0$). This relation can be captured via a function $f : [0, \infty) \to [0,1]$ such that $f(0) = 1$, $\lim_{x \to \infty} f(x) = 0$ and f is monotone decreasing. We use the mapping $p = f(\varepsilon) = \frac{1}{\varepsilon + 1}$ and $\varepsilon = f^{-1}(p) = \frac{1}{p} - 1$. This mapping does not carry meaning such as equivalence in any sense between methods, so it is not used for direct comparison. We only use it to convert the privacy parameter ε into $[0,1]$ so we can use *bDP* as privacy-preserving method M in the CoL game defined in Section 3.

6 RecSys: Measuring Φ

For all questions in relation with CoL, the answers depend on Φ_n^M. Consequently, in this section we measure the model accuracy for various privacy parameters with regard to the learning task and the privacy mechanisms introduced in Section 5.

For our experiments, we implemented SGD as training algorithm in Matlab [15]. We used the MovieLens 1M[8] and Netflix[9] datasets; for complexity reasons we shrunk the Netflix dataset to 10% by randomly filtering out 90% of the users. Furthermore, both datasets are preprocessed similarly to [3]. Preprocessing is described in details in Appendix D. We will refer to the preprocessed datasets as 1M and NF10, respectively. The parameters of the preprocessed datasets are shown in Table 3.

Dataset	Rating	User	Item	Density
1M	998 539	6040	3260	0.051
NF10	10 033 823	46 462	16 395	0.013

Table 3. The datasets size after preprocessing

The algorithm for MF is SGD, where the number of features are $\kappa = 4$. The algorithm runs for 20 iterations ($\iota = 20$) with learning rate $\gamma = 0.0075$ and regularization parameter $\lambda = 0.01$. The feature matrices are bounded by $p_{max} = q_{max} = 0.5$. This means that the sensitivity of the RecSys scenario is $S \leq 4 \cdot 20 \cdot 0.0075 \cdot (2 \cdot 2 \cdot 0.5 - 0.01 \cdot 0.5) = 1.197$ as a result of Theorem 5.

We assume that if a model is trained using datasets from very different distributions, the model captures the properties of the mixed distribution of the combined dataset (which might be far from the original distributions). On the other hand, using training data from similar distributions results in capturing the statistical

7 We treat p as a continuous variable even though it is discrete; this does not affect our analysis owing to large dataset sizes.

8 https://grouplens.org/datasets/movielens/

9 http://academictorrents.com/details/9b13183dc4d60676b773c9e2cd6de5e5542cee9a

properties of a distribution close to the original ones. As such, if the players' datasets are from a similar different distribution, training together likely results in a more accurate model than training alone. Consequently, we imitate the players' datasets by splitting 1M and NF10: each user with its corresponding ratings is assigned to one of the players. To remove the effect of randomness of the dataset division, we run our experiments three times and only present the averages. Now, each player splits its dataset further into a training set (80%) and a verification set (20%). The players can run the SGD algorithm alone or together, where additional privacy mechanisms can be deployed. The accuracy of the trained model is measured via root mean square error: $RMSE = \sqrt{\frac{\sum e_{iu}^2}{|R|}}$.

6.1 Alone vs Together

First, we compare the achieved accuracy with and without the other player's data in Figure 5. The horizontal axis represents the ratio of the user-set sizes: how 1M and NF10 was split into two. More precisely, $x = \frac{\alpha}{\beta}$ represents that Player 1's dataset is $\frac{\alpha}{\beta}$ times the size of Player 2's dataset. The vertical axis shows the difference in the normalized accuracy improvements of training alone and together: $y = \frac{(\theta-o)-(\phi-o)}{\theta-o} = \frac{\theta-\phi}{\theta-o}$, where o is the accuracy of the model without training. In other words, y is the normalized accuracy improvement via training together; $y = 0$ represents the accuracy of training alone.

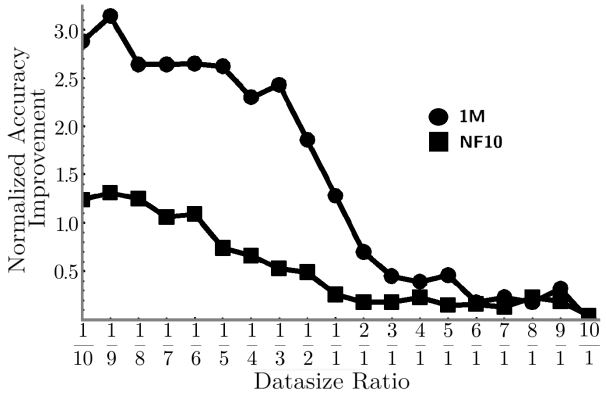

Fig. 5. Accuracy improvement (y axis) of training together using 1M/NF10 datasets where one player has x times more (less) data than the other (x axis)

It is clear that training together is superior to training alone for both datasets and all size ratios. Figure 5 also shows that the owner of the smaller dataset benefits more from collaboration; a well-expected characteristic.

6.2 One Player is Privacy Concerned

Training together achieves higher accuracy than training alone. The question is, how does the situation change with a privacy mechanism in place. First we analyze the CaaS scenario introduced in Definition 6. Without loss of generality, we can assume $p_2 = 0$. Player 1's options are either to set p_1 for *Sup* or ε_1 for *bDP*.

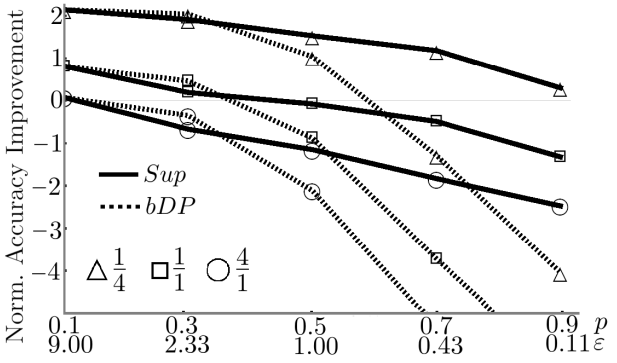

Fig. 6. Accuracy improvements of training together (y axis) for different privacy levels (x axis) with the 1M dataset is divided such that the data size ratios are 0.25, 1 and 4

Figure 6 shows the tradeoff between accuracy and privacy when the 1M dataset is divided such that the data size ratio is 0.25, 1 and 4 (from Player 1's perspective). The horizontal axis is the privacy parameter p_1 (ε_1) while the vertical axis shows the normalized improvement on accuracy achieved by training together (similar to Figure 5). In Figure 6, we can observe both higher ($y > 0$) and lower ($y < 0$) collaborative accuracy regions. Note, that we only show the case for 1M and select dataset size ratios as we found that using other size ratios or the NF10 dataset produce similar curves. The main observation is valid in all settings: as the dataset size ratio increases the accuracy improvement decreases.

These results suggest that the realistic privacy parameters the players can apply (to obtain a more accurate model) depend on the relative size of their datasets: a player with relatively smaller dataset (e.g., triangle in Figure 6) can apply a stronger privacy parameter (and still obtain more accurate model) than a player with a relatively larger dataset (e.g., circle in Figure 6).

This finding confirms our assumption about the derivatives of Φ_n^M in Corollary 1: since the relative

dataset size effects the obtained accuracy with a constant, a change in the privacy parameter effects the accuracy the same way independently of these ratios. This is indeed implies that $\partial_{p_1}\Phi_1^M = \partial_{p_2}\Phi_2^M$.

6.3 Both Players are Privacy Concerned

In this section we jointly train a model where both players employ the same mechanism M with privacy level p_1 (ε_1) and p_2 (ε_2). Section 6.1 and 6.2 already pointed out that a player with a significantly larger dataset would not benefit much from collaboration. Consequently, for this experiment we use datasets with similar sizes. This makes our scenario symmetric, i.e., it is enough to demonstrate the accuracy change for one player. We obtained results for both 1M and NF10, but due to their similarity we only display the results for 1M in Figure 7. We use the notation p_{own} (ε_{own}) and p_{other} (ε_{other}) to represent the privacy parameters from the perspective of the player under scrutiny.

Figure 7 shows the normalized accuracy improvement for different privacy parameters with privacy mechanism $M \in \{Sup, bDP\}$. It is visible that independently from M, accuracy is more sensitive to the player's own privacy parameter than to that of the other. In other words, by degrading the quality of a given player's data (via a privacy mechanism), this player's accuracy will be affected more than the accuracy of the other player. This means, if the players would have additional incentive to undermine the other player's accuracy (e.g., competing companies), by doing so they would actually decrease their own accuracy more.

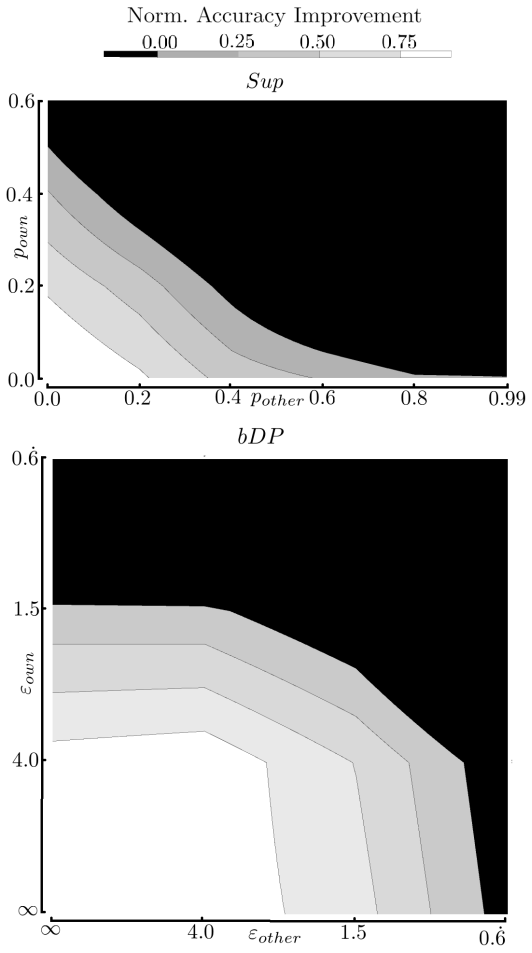

Fig. 7. Accuracy improvement of collaboration where 0 represents the accuracy level of training alone (1M dataset split equally, $M \in \{Sup, bDP\}$).

7 RecSyS: Numerical Equilibria

In this section we compute the numerical values of NE (p_i^*) for the RecSyS scenario, implementing the process depicted in Figure 2. In Section 7.1 we assume that some measurements on Φ are readily available. In Section 7.2 we relax this assumption, and propose a heuristic called Self-Division to approximate Φ_n^M locally.

7.1 NE Computation via Interpolation

Using the values of Φ_n^M obtained in Section 6, it is possible to interpolate the privacy-accuracy trade-off function and combine it with the theoretical results in Section 4 to obtain numerical equilibria.

Let b and c be linear as in Equation (4). Now, we interpolate directly b instead of Φ_n^M, i.e., we interpolate the normalized accuracy improvement shown in Figure 7. We use *Mathematica*'s[10] built-in `Interpolate` function with `InterpolationOrder` $\rightarrow 1$ in order to have a monotone approximation required by Definition 3. Via this interpolation the numerical NE can be calculated for the specific dataset (1M) and algorithm (SGD) defined in Section 5. We assume that the 1M dataset is split equally between the players.

One player is privacy concerned. In the CaaS scenario we assume that Player 1 is privacy unconcerned.

10 https://www.wolfram.com/mathematica/

According to Theorem 2, this player's BR is $\hat{p}_1 = 0$. Now the utility function of Player 2 is:

$$u_2(0, p_2) = B_2 \cdot \overbrace{\left[\frac{\theta_2 - \Phi(0, p_2)}{\theta_2}\right]^+}^{b(0,p_2)} - C_2 \cdot c(p_2) \qquad (11)$$

As Lemma 1 and 2 state, there is a lower and an upper bound on $\frac{C_2}{B_2}$ for Player 2 which ensures that the BR $\hat{p}_2$ is either 0 or 1. We calculate the exact bounds using our interpolation. Furthermore, the utility of Player 2 (Equation (11)) has to be positive (Theorem 3), otherwise there is no incentive for Player 2 to participate in the CoL process. These bounds are shown in Table 4; they can also be observed prominently in Figure 8 (where $B_2 = 1$).

$0 \leq u_2(0, \hat{p}_2)$ if		$\hat{p}_2$	if
$\frac{C_2}{B_2} \leq 0.990$	Sup	0	$\frac{C_2}{B_2} \leq 1.400$
		1	$\frac{C_2}{B_2} \geq 1.827$
$\frac{C_2}{B_2} \leq 1.150$	bDP	0	$\frac{C_2}{B_2} \leq 0.349$
		1	$\frac{C_2}{B_2} \geq 2.251$

Table 4. NEs for Player 2 when Player 1 is privacy unconcerned

In Figure 8 we display the BR and the corresponding utility as a function of the privacy weight C_2 where the utility function is normalized by B_2 (i.e., $B_2 = 1$). This transformation is sign-preserving, i.e., if the utility is negative, the BR is not a NE, since no collaboration corresponds to a higher utility.

In case of *Sup* the interval defined by the two lemmas (represented by the two vertical gray lines) corresponds to negative utility. Thus, the NE is either collaboration without privacy protection or no collaboration depending on the weight ratio $\frac{C_2}{B_2}$. More precisely, according to Theorem 3 the NE is $(p_1^*, p_2^*) = (0, 0)$ if $\frac{C_2}{B_2} \leq b(\theta_2, \Phi_2(0, 0)) = 0.990$ and $(p_1^*, p_2^*) = (1, 1)$ otherwise.

In case of *bDP*, the leftmost part of the interval created by the lemmas corresponds to positive utility, i.e., there exists a non-trivial NE. More precisely, if $0.349 \leq \frac{C_2}{B_2} \leq 1.150$ then p_2^* is neither 0 nor 1. Note, that the BR function is step-like because of the piecewise linear interpolation. As such, within this interval the NE is $p_2^* = 0.2 \Leftrightarrow \varepsilon_2^* = 4$. The *Prive of Privacy* for this NE is $PoP(0, 0.2) = 0.066$, i.e., less than 7% of the overall achievable accuracy is lost due to privacy concerns.

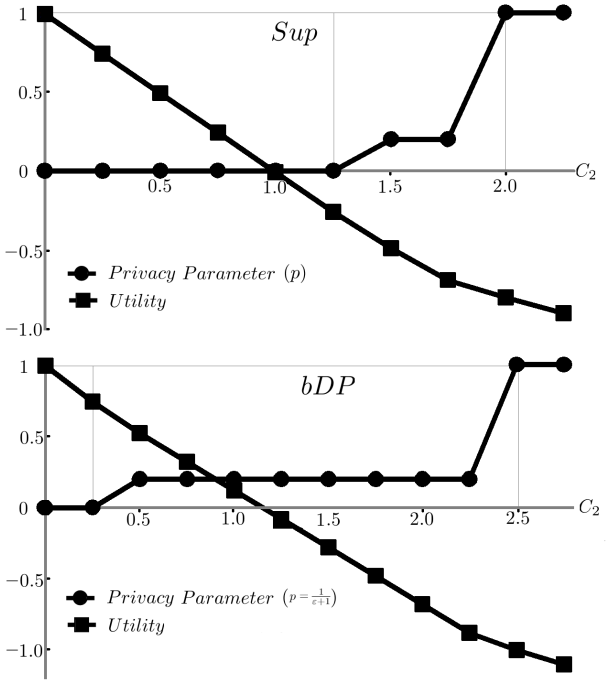

Fig. 8. BR and corresponding utility for Player 2 when Player 1 is privacy unconcerned

Both players are privacy concerned. Here we present results only for *bDP* (*Sup* shows similar characteristics). Theorem 4 states that when both player is privacy concerned a non-trivial NE exists. To this end, we use BR dynamics [6], eventually converging to a NE due to the following theorem:

Theorem (Monderer & Shapley [12])**.** *In a finite*[11] *potential game, from an arbitrary initial outcome, the BR dynamics converges to a pure strategy NE.*

In BR dynamics (over several rounds) players update their strategies in the next round based on the their BR to the strategy of the other player in the last round. We start the iteration from $(p_1, p_2) = (0, 0)$[12] and update the players' strategies alternately starting with Player 1. The NEs to which the process converges are shown in Table 5 with the corresponding *Price of Privacy* values for discrete weight ratios $\{0, 0.1, \dots\}$.

This suggest that players with a low privacy weight prefer to train together without protection, while a high

11 As the CoL game is not finite, we discretized the action spaces of the players via floating point numbers.

12 The higher the privacy level from where the BR dynamics starts, the wider the interval of weight ratios in which it converges to 1 (i.e., no collaboration).

$\frac{C_2}{B_2} \in \rightarrow$	$[0, 0.3]$	$[0.4, 0.9]$	$[1, \infty]$
$\frac{C_1}{B_1} \in [0, 0.3]$	$\mathbf{(0, 0)}$ 0.000	$\mathbf{(0, 0.2)}$ 0.066	$\mathbf{(1, 1)}$ 1.000
$\frac{C_1}{B_1} \in [0.4, 0.9]$	$\mathbf{(0.2, 0)}$ 0.066	$\mathbf{(0.2, 0.2)}$ 0.131	$\mathbf{(1, 1)}$ 1.000
$\frac{C_1}{B_1} \in [1, \infty]$	$\mathbf{(1, 1)}$ 1.000	$\mathbf{(1, 1)}$ 1.000	$\mathbf{(1, 1)}$ 1.000

Table 5. NEs for different weight ratios

privacy weight ensures no collaboration. Also, the narrow interval in-between corresponds to collaboration with low privacy protection or no collaboration at all.

7.2 NE Computation via Self-Division

Direct interpolation of Φ_n^M is only possible when both datasets are fully available. In a real-world scenario it must be approximated by other means. Here we demonstrate a simple approach to fill the gap in the CoL game caused by the obscurity of Φ_n^M (Step "Approximation" in Figure 2). Note, that our intention is not to provide a sound generic method for approximating the effect of privacy mechanisms on the accuracy of complex training algorithms, but rather to show a direction how it could be done. More research is required in this direction, which we consider interesting future work.

Self-Division and parameter heuristics. As we argued in Section 6, we can assume that players' datasets are from similar distributions, i.e., a player can imitate CoL by mimicking the other player's dataset by splitting their own dataset into two and approximate Φ_n^M locally.

Based on our experiments which are visible in Appendix E, we establish a heuristic formula which minimizes the error of this local approximation based on the size and density of the players' datasets. The formula can be seen in Equation (12) where d is the density of the datasets and D_n is player n's dataset. We refer to the true (interpolated) privacy-accuracy trade-off function as Φ_n^M and our approximation via Self-Division as $\widetilde{\Phi_n^M}$.

$$100\,000 \approx d \cdot |D_n| = \frac{|D_n|^2}{|U_n| \cdot |I|} \tag{12}$$

Equilibrium computation. Let bDP be the privacy-preserving mechanism M. We assume that players have a chunk of the preprocessed NF rating dataset, which contains only movie ratings. As such, it is expected that the players value privacy less than accuracy: for the sake of this example, we set $B_1 = B_2 = 1$ and $C_1 = C_2 = 0.1$. We use the benefit and privacy loss functions defined in Equation (4).

As we showed in Appendix E, Self-Division is the most punctual when Equation (12) holds. Since the density of the original NF dataset is $d \approx 0.01$, we assign both players datasets with 10 million ratings: we randomly choose 20% of the users from NF10 and assign them to either one of the players.

The players separately approximate Φ_n by self-division, therefore, $\widetilde{\Phi_1}$ and $\widetilde{\Phi_2}$ are not necessarily the same. The exact values of these approximations can be seen in Appendix F together with the true value of Φ_n.[13] We found that the RMSE of $\widetilde{\Phi_n}$ is around 0.001 for both players.

Using $\widetilde{\Phi_1}$ Player 1 approximates the NE as $(\widetilde{p_1^*}, \widetilde{p_2^*}) = (0, 0)$ while Player 2 reaches the same conclusion via $\widetilde{\Phi_2}$; this corresponds to $PoP = 0$. Approximated utilities are $\widetilde{u_1} = 0.18$ and $\widetilde{u_2} = 0.07$ respectively. The actual utilities in case of $(\widetilde{p_1^*}, \widetilde{p_2^*})$ are $(0.21, 0.07)$, which are very close to the approximated values. While utility approximation is fairly accurate, Φ_n actually corresponds to a slightly different NE: $(p_1^*, p_2^*) = (0.2, 0.2)$ with utility $(u_1, u_2) = (0.14, 0.06)$ and $PoP = 0.25$. Note, that while both players obtain a higher payoff via $\widetilde{p_n^*}$ that is not an actual NE.

8 Related Work

Several works have been devoted to the privacy implications of collaborative information sharing systems. The fundamental trade-off between privacy and accuracy was studied in [10], where the authors proposed a general model for collaborative information analysis systems to determine which privacy mechanism optimizes the trade-off between privacy and accuracy. In a follow-up work [17], authors proposed a privacy mechanism which finds the best set of features in terms of privacy-utility trade-off in a distributed data sharing architecture. In the rest of this section, we divide the remaining related literature into two categories; privacy preservation and game-theoretical modeling.

13 Note, that Φ_n itself is interpolated from its actual value at measured points (see Section 7.1).

8.1 Privacy Preserving Distributed ML

ML is frequently implemented in a distributed fashion for efficiency reasons. To tackle its emerging privacy aspect, privacy preserving distributed ML was introduced, where the locally trained models are safely aggregated.

Distributed training scenarios unanimously assume a large number of participants and the involvement of a third party such as in [4, 11, 13, 14, 16]. In more details, in [13] mutually untrusted parties train classifiers locally and aggregate them with the help of an untrusted curator. In the introduced ε-DP protocol, achieved accuracy depends on the number of parties and the relative fractions of data owned by the different parties. In [16] these dependencies were eliminated for a SGD training algorithms. On the other hand, authors used (ε, δ)-DP, a weaker form of DP.

More recently in [4] an ε-DP classifier was introduced with error bound $O((\varepsilon N)^{-2})$ compared to the result of the non-private training where N is the number of participants. This approach results in strong privacy guarantees without performance loss for large N. Federated Learning introduced in [11] follows another approach, where the users generate pairwise noise to mask their data from the aggregator. The bottleneck of this approach is the communication constraints. Furthermore, the solution is not applicable to two participants.

All these works assumed the existence of a third-party aggregator; however, in our work the data holders themselves train a model together to achieve higher accuracy than what they would have obtained if training in isolation. Furthermore, all of these works are either not suitable or not efficient for two participants.

8.2 Distributed ML and Game Theory

In [14] the learning process was modeled as a Stackelberg game amongst $N + 1$ players where a learner declares a privacy level and then the other N data holders respond by perturbing their data as they desire. The authors concluded that in equilibrium each data holder perturbs its data independently of the others, which leads to high accuracy loss.

The closest to our work are [1, 7, 18]. In [7] a linear regression scenario was studied where the features were public but the data were private. With these settings, the authors proved the existence of a unique non-trivial NE, and determined its efficiency via the Price of Stability.

A simpler problem was modeled in [1]: estimating a population's average of a single scalar quantity. The authors studied the interaction between agents and an analyst, where the agents can either deny access to their private data or decide the level of precision at which the analyst gets access. Findings include that it is always better to let new agents enter the game as it results in more accurate estimation, and the accuracy can further be improved if the analyst sets a minimum precision level.

In both previous scenarios, players would like to learn a model which represents the whole population. The accuracy of the estimate is a public good (i.e., non-exclusive and non-rival [6]). On the contrary, in CoL the players seek to selfishly improve their own accuracy as that is in their own self-interest. As such, they measure the accuracy of the trained model by how well it fits to their own datasets, which can result in different accuracy levels. Furthermore, these works focused on particular tasks (linear regression and scalar averaging) while our model is applicable for any training mechanism.

[18] studied the problem of private information leakage in a data publishing scenario where datasets are correlated. As such, the utility function for an agent consists of the benefit of publishing its own sanitized dataset and the privacy leakage which depends on the privacy parameters of all involved agents. Opposed to this, in our model datasets are independent, while the benefit is affected by the actions of both players. Thus, the accuracy of the training depends on the privacy parameters of both agents, while the privacy loss depends only on the privacy parameter of a single agent.

9 Conclusion

In this paper, we designed a Collaborative Learning process among two players. We defined two player types (privacy concerned and unconcerned) and modeled the training process as a two-player game. We proved the existence of a Nash Equilibrium with a natural assumption about the privacy-accuracy trade-off function (Φ) in the general case, while provided the exact formula when one player is privacy unconcerned. We also defined the *Price of Privacy* to measure the overall degradation of accuracy owing to the players' protecting the privacy of their own dataset.

On the practical side, we studied a Recommendation System use case: we applied two different privacy-preserving mechanisms (suppression and bounded dif-

ferential privacy) on two real-world datasets (MovieLens and Netflix). We confirmed via experiments that the assumption which ensures the existence of a Nash Equilibrium holds. Moreover, as a complementary work besides the designed game, we interpolated Φ for this use case, and devised a possible way to approximate it in real-world scenarios. We found that privacy protection degrades the accuracy heavily for its user. Moreover, Collaborative Learning is only practical when either one player is privacy unconcerned or the players have similar dataset sizes and both players' privacy concerns (weights) are relatively low.

Future work. There are multiple opportunities to improve this line of work such as upgrading the CoL process by controlling the other party's updates. Another possibility is to design a repetitive game where each player faces a decision after each iteration or make the game asymmetric by defining the weights B and C in private. Incorporating the impact of the potential adversarial aspect for competing companies, and thus investigating a more elaborate utility function is another intriguing possibility. Finally, how to determine the weight parameters for specific scenarios and approximate Φ is crucial for the real-world usability of the model.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments. This work has been partially funded by the Higher Education Excellence Program of the Hungarian Ministry of Human Capacities in the frame of the Artificial Intelligence research area of the Budapest University of Technology and Economics (BME FIKP-MI/SC). Gergely Biczók has been supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

References

[1] Michela Chessa, Jens Grossklags, and Patrick Loiseau. A game-theoretic study on non-monetary incentives in data analytics projects with privacy implications. In *Computer Security Foundations Symposium (CSF), 2015 IEEE 28th*. IEEE, 2015.

[2] Cynthia Dwork. Differential privacy. In *Proceedings of the 33rd international conference on Automata, Languages and Programming*. ACM, 2006.

[3] Arik Friedman, Shlomo Berkovsky, and Mohamed Ali Kaafar. A differential privacy framework for matrix factorization recommender systems. *User Modeling and User-Adapted Interaction*, 2016.

[4] Jihun Hamm, Yingjun Cao, and Mikhail Belkin. Learning privately from multiparty data. In *International Conference on Machine Learning*, 2016.

[5] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2012.

[6] John C Harsanyi, Reinhard Selten, et al. A general theory of equilibrium selection in games. *MIT Press Books*, 1988.

[7] Stratis Ioannidis and Patrick Loiseau. Linear regression as a non-cooperative game. In *International Conference on Web and Internet Economics*. Springer, 2013.

[8] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.

[9] Elias Koutsoupias and Christos Papadimitriou. Worst-case equilibria. In *Stacs*. Springer, 1999.

[10] Fabio Martinelli, Andrea Saracino, and Mina Sheikhalishahi. Modeling privacy aware information sharing systems: A formal and general approach. In *Trustcom/BigDataSE/ISPA, 2016 IEEE*, pages 767–774. IEEE, 2016.

[11] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

[12] Dov Monderer and Lloyd S Shapley. Potential games. *Games and economic behavior*, 1996.

[13] Manas Pathak, Shantanu Rane, and Bhiksha Raj. Multiparty differential privacy via aggregation of locally trained classifiers. In *Advances in Neural Information Processing Systems*, 2010.

[14] Jeffrey Pawlick and Quanyan Zhu. A stackelberg game perspective on the conflict between machine learning and data obfuscation. In *Information Forensics and Security (WIFS), 2016 IEEE International Workshop on*, 2016.

[15] Balazs Pejo. Matrix factorisation in matlab via stochastic gradient descent. https://github.com/pidzso/ML.

[16] Arun Rajkumar and Shivani Agarwal. A differentially private stochastic gradient descent algorithm for multiparty classification. In *Artificial Intelligence and Statistics*, 2012.

[17] Mina Sheikhalishahi and Fabio Martinelli. Privacy-utility feature selection as a privacy mechanism in collaborative data classification. In *Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2017 IEEE 26th International Conference on*, pages 244–249. IEEE, 2017.

[18] Xiaotong Wu, Taotao Wu, Maqbool Khan, Qiang Ni, and Wanchun Dou. Game theory based correlated privacy preserving analysis in big data. *IEEE Transactions on Big Data*, 2017.

Appendices

A List of Abbreviations

Abr.	Meaning
ML	Machine Learning
CoL	Collaborative Learning
RecSys	Recommender System
MF	Matrix Factorization
SGD	Stochastic Gradient Descent
DP	Differential Privacy
Sup	Suppression
CaaS	Collaboration-as-a-Service
PoP	Price of Privacy
NE	Nash Equilibrium
BR	Best Response

Table 6. Frequently used abbreviations

B Proofs for Section 4

Proof Theorem 1. Without loss of generality, assume Player 2 sets $p_2 = 1$. The highest accuracy Player 1 can achieve corresponds to $p_1 = 0$ due to Definition 3 and 4. From Definition 4 we can also deduce that if one player sets its privacy parameter to 1 then neither of the players can obtain higher accuracy by training together than by training alone. As such, the highest accuracy that Player 1 can reach by training together when Player 2 sets its privacy parameter to maximum is less than what it would achieve by training alone: $\Phi_1(0,1) \geq \theta_1$. Note that Φ and θ measure the error, i.e., the higher these values are, the less accurate the corresponding model is.

As such, $p_1 = 0$ does not correspond to positive benefit but only results in privacy loss. Hence, the highest payoff Player 1 can reach is 0 corresponding to maximal privacy protection $p_1 = 1$. In other words, if Player 2 sets $p_2 = 1$ the BR of Player 1 is also to set $p_1 = 1$. Since this is also true the other way around, $(p_1^*, p_2^*) = (1, 1)$ is indeed a NE which is equivalent to the case of training alone. □

Proof Lemma 1. If $\alpha_n = 0$, the utility function in Equation (1) is reduced to $u_n = B_n \cdot b(\theta_n, \Phi_n)$ since $C_n = 0$. This is strictly positive by definition. Also by definition b is monotone decreasing in p_n. As a result, the utility is highest when no privacy protection is in place. As such, there indeed exists α_n such that $\hat{p}_n = 0$ is the BR for player n. □

Proof Lemma 2. Without loss of generality we assume $n = 1$. We show that $\max_{p_1} u_1(p_1, p_2) = u_1(1, p_2) = 0$ if $C_1 \to \infty$ which is equivalent to the statement in Lemma 2:

$$\begin{aligned} &\lim_{C_1 \to \infty} u_1(p_1, p_2) = \\ &\lim_{C_1 \to \infty} B_1 \cdot b(\theta_1, \Phi_1(p_1, p_2)) - c(p_1) \cdot C_1 \leq \\ &\lim_{C_1 \to \infty} B_1 \cdot b(\theta_1, \Phi_1(0,0)) - c(p_1) \cdot C_1 = \\ &\lim_{C_1 \to \infty} \beta_0 - c(p_1) \cdot C_1 = \begin{cases} \boldsymbol{\beta_0} & \textbf{if} \quad \boldsymbol{c(p_1) = 0} \\ \boldsymbol{-\infty} & \textbf{if} \quad \boldsymbol{c(p_1) > 0} \end{cases} \end{aligned} \tag{13}$$

As a result, $u_1(p_1, p_2) \leq \beta_0$ for some $\beta_0 \geq 0$ and it can only be non-negative if $c(p_1) = 0$ which corresponds to $p_1 = 1$. The utility is maximal in this case, thus, $\max_{p_1} u(p_1, p_2) = u(1, p_2)$ which is indeed 0. □

Proof Theorem 2. In the proof of Lemma 1 we set $C_n = 0$ in which case player n's BR was indeed $\hat{p}_n = 0$. For more details read the proof of Lemma 1. □

Proof Theorem 3. The utility function $u(p_1)$ is maximal in the interval $[0, 1]$ either on the border or at a point where its derivative is zero. The derivative of Equation (1) is

$$u'(p_1) = B_1 b'(p_1)\Phi'(p_1) - C_1 c'(p_1) \tag{14}$$

which is zero at $\tilde{p}_1$ if

$$u'(\tilde{p}_1) = 0 \Rightarrow \frac{b'(\tilde{p}_1)\Phi'(\tilde{p}_1)}{c'(\tilde{p}_1)} = \frac{C_1}{B_1} \tag{15}$$

Of course the extreme point $\tilde{p}_1$ must be in $[0, 1]$ and $u(\tilde{p}_1) > 0$. Furthermore, this extreme point is a maximum only if the second derivative is negative.

On the other hand, if Equation (14) is never zero in $[0, 1]$ or the second derivative is positive at that point, the maximum of $u(p_1)$ is on the edge of the interval $[0, 1]$. $u(1) = 0$ since both the benefit and the privacy loss functions are zero at $p_1 = 1$. As a result, $p_1 = 0$ is the maximum point if $u(0) > 0$. This is indeed the case when the maximal benefit $b(0)$ is higher than the ratio of the privacy and accuracy weight $\frac{C_1}{B_1}$ as it is shown below:

$$0 < u(0) = B_1 b(0) - C_1 \Rightarrow b(0) > \frac{C_1}{B_1}$$

□

Proof Theorem 4. We divide u_n by B_n: $\tilde{u}_n = \frac{u_n}{B_n}$. This new function inherits the properties of u_n (such as the sign, monotonicity, maximum/minimum points, etc.). As a result, a similar game with utility function $\tilde{u}_n$ has the same equilibria. Furthermore, this similar game is a potential game if the mixed second order partial derivative of the utility functions are equal. Due to the constitution of $\tilde{u}_n$, this condition is equivalent to

$$\partial_{p_1}\partial_{p_2} b(\theta_1, \Phi_1(p_1, p_2)) = \partial_{p_1}\partial_{p_2} b(\theta_2, \Phi_2(p_1, p_2))$$

This formula can be transformed into the one in the theorem by applying the chain rule of derivation for higher dimensions. □

Proof Corollary 1. The left side of the equation in Theorem 4 is zero since we assumed $\partial_{p_1}\Phi_1 = \partial_{p_2}\Phi_2$. On the right side $\partial_{p_1}\partial_{p_2}\Phi_2 = \partial^2_{p_1}\Phi_1$ and $\partial_{p_2}\partial_{p_1}\Phi_1 = \partial^2_{p_2}\Phi_2$ for the same reason. Theorem 4 holds since both sides of the equation are 0. □

C Proof of Theorem 5

Proof Theorem 5. Since the user sets of the players are disjoint while the item set is shared, the only thing the players need to share is the item feature matrix Q. The effect of a single update is shown in Equation (8). We assume that the data points are independent, hence, the sensitivity $\tilde{S}$ of one update is

$$\begin{aligned}\tilde{S} = \max_{r_{ui}} |q'_{ki} - q_{ki}| &= \max_{r_{ui}} [\gamma(e_{ui}p_{uk} - \lambda q_{ki})] \\ &\leq \gamma(\Delta r p_{max} + \lambda q_{max})\end{aligned}$$

where

- $k \in [1, \kappa]$
- Δr is the maximal distance of two ratings: $\Delta r = \max r_{ui} - \min r_{ui}$
- p_{max} and q_{max} are the maximal absolute values of the user and item features, respectively.

$\tilde{S}$ is the sensitivity of updating a single feature, thus, to capture the full effect of the update on vector q_i, we need to multiply $\tilde{S}$ with q_i's dimension κ. Moreover, we have only considered the effect of a rating on Q within one iteration. However, this update occurs ι times. Thus, to achieve ε-DP, we need to apply $\frac{\kappa \cdot \iota \cdot \tilde{S}}{\varepsilon}$ level of Laplacian noise on the ratings before the training due to the Composition Theorem. Therefore, the overall sensitivity is indeed bounded by the formula in the theorem. □

D Preprocessing

1 Remove items/users with less than 10 ratings.

2 For each remaining item, calculate the average rating and discount it from the corresponding r_{ui}'s:

$$r'_{ui} := r_{ui} - I_{Avg}(i)$$

3 For each remaining user, calculate the average rating and discount it from the corresponding r'_{ui}'s:

$$r''_{ui} := r'_{ui} - U_{Avg}(u) = r_{ui} - I_{Avg}(i) - U_{Avg}(u)$$

4 The discounted ratings as well as the averages are clamped:

- $I_{Avg}(i) \in [\min(r_{ui}), \max(r_{ui})] = [1, 5]$
- $U_{Avg}(u) \in [-2, 2]$
- $r''_{ui} \in [-2, 2]$

E Self-Division: Experiment

1 We create datasets with approximately the same density but different sizes:

- 1M: We modify the size of the dataset while keeping its density; we randomly remove users such that the remaining dataset has 1000k/800k/600k ratings (i.e., the players have 500k-500k/400k-400k and 300k-300k ratings).
- NF10D: We create a new dataset originated from NF10 by increasing its density to the level of 1M via filtering out the less rated items[14]. Afterwards we modify the size of this dataset while keeping its newly acquired density: we randomly remove users such that the remaining dataset has 8M/6M/4M/2M ratings.

2 We execute CoL with $p_i \in \{0, 0.2, 0.4, 0.6\}$ for $i = \{1, 2\}$ and for $M \in \{Sup, bDP\}$ using the newly created datasets (i.e., the players have 300k/400k/.../3M/4M ratings) and we obtain the normalized accuracy improvement for both players: $\Phi' = \frac{\theta - \Phi}{\theta - o}$, where o is the error corresponding to the initial model.

3 We execute CoL with the same privacy parameters and methods using only one player's data: the players imitate CoL by halving their own datasets (i.e., dataset sizes are 150k-150k/.../2m-2m). We obtain $\widetilde{\Phi'} = \frac{\widetilde{\theta_n} - \widetilde{\Phi_n}}{\widetilde{\theta_n} - \widetilde{o_n}}$ where $\widetilde{\cdot_n}$ corresponds to the errors of the CoL model produced solely by Player n's data.

14 We remove items with less than 250 ratings

4 We calculate the RMSE between the true (interpolated) normalized accuracy Φ' and the approximated normalized accuracy via Self-Division $\widetilde{\Phi}'$ for both players and privacy methods.

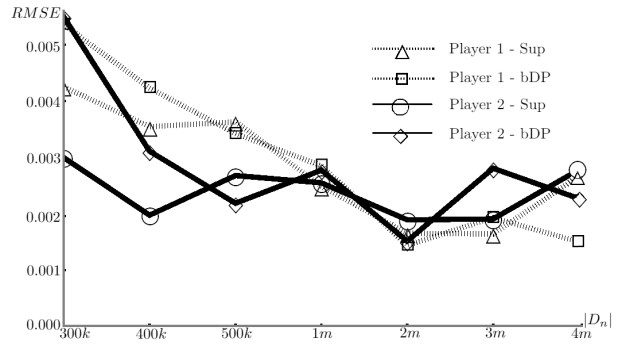

Fig. 9. We show the error (RMSE) of Self-Division (i.e., $\Phi' - \widetilde{\Phi}'$) for both player and privacy methods.

We found, that the RMSE was minimal for both privacy mechanisms and players when the players have $|D_n| = 2\,000\,000$ ratings. This means that for datasets with density $d \approx 0.05$, $\widetilde{\Phi'_n}$ is the closest to Φ'_n when the $100\,000 \approx d \cdot |D_n|$ heuristic holds.

F Playerwise Approximations

$\widetilde{\Phi_1}$	$p_2 = 0.0$	$p_2 = 0.2$	$p_2 = 0.4$	$p_2 = 0.6$
$p_1 = 0.0$	0.28	0.26	0.24	−0.05
$p_1 = 0.2$	0.25	0.16	0.15	−0.05
$p_1 = 0.4$	−0.07	−0.10	−0.19	−0.37
$p_1 = 0.6$	−1.01	−1.16	−1.37	−1.72

$\widetilde{\Phi_2}$	$p_1 = 0.0$	$p_1 = 0.2$	$p_1 = 0.4$	$p_1 = 0.6$
$p_2 = 0.0$	0.17	0.16	0.15	−0.05
$p_2 = 0.2$	0.14	0.12	0.12	−0.07
$p_2 = 0.4$	−0.14	−0.17	−0.28	−0.60
$p_2 = 0.6$	−1.19	−1.21	−1.28	−1.83

Φ_1	$p_2 = 0.0$	$p_2 = 0.2$	$p_2 = 0.4$	$p_2 = 0.6$
$p_1 = 0.0$	0.17	0.14	0.11	−0.03
$p_1 = 0.2$	0.15	0.12	0.08	−0.26
$p_1 = 0.4$	−0.13	−0.19	−0.33	−0.69
$p_1 = 0.6$	−1.16	−1.32	−1.49	−2.08

Φ_2	$p_1 = 0.0$	$p_1 = 0.2$	$p_1 = 0.4$	$p_1 = 0.6$
$p_2 = 0.0$	0.31	0.23	0.17	−0.05
$p_2 = 0.2$	0.31	0.22	0.11	−0.18
$p_2 = 0.4$	−0.14	−0.16	−0.22	−0.52
$p_2 = 0.6$	−1.13	−1.25	−1.30	−1.85

Table 7. Approximated privacy-accuracy tradeoff function for both players ($\widetilde{\Phi_1}$, $\widetilde{\Phi_2}$) and its true value (Φ_1 and Φ_2).