Privacy-Preserving Neural Processes for Probabilistic User Modeling

Amir Sonee [*] [†] [1,2] **Haripriya Harikumar**[*1] **Alex Hämäläinen**[3] **Lukas Prediger**[3] **Samuel Kaski**[1,3,4]

[1]Department of Computer Science, University of Manchester, Manchester, UK
[2]Department of Computing, Imperial College London, London, UK
[3]Department of Computer Science, Aalto University, Espoo, Finland
[4] ELLIS Institute Finland, Helsinki, Finland

Abstract

Uncertainty-aware user modeling is crucial for designing AI systems that adapt to users in real-time while addressing privacy concerns. This paper proposes a novel framework for privacy-preserving probabilistic user modeling that integrates uncertainty quantification and differential privacy (DP). Building on neural processes (NPs), a scalable latent variable probabilistic model, we enable meta-learning for user behaviour prediction under privacy constraints. By employing differentially private stochastic gradient descent (DP-SGD), our method achieves rigorous privacy guarantees while preserving predictive accuracy. Unlike prior work, which primarily addresses privacy-preserving learning for convex or smooth functions, we establish theoretical guarantees for non-convex objectives, focusing on the utility-privacy trade-offs inherent in uncertainty-aware models. Through extensive experiments, we demonstrate that our approach achieves competitive accuracy under stringent privacy budgets. Our results showcase the potential of privacy-preserving probabilistic user models to enable trustworthy AI systems in real-world interactive applications.

1 INTRODUCTION

Understanding and modeling user behaviour [Yuan et al., 2020, Yu et al., 2019] is essential for designing adaptive AI systems in real-world interactive scenarios. Many recent AI-assistant applications, e.g. [De Peuter et al., 2024, Moon et al., 2023, Oulasvirta et al., 2022], are based on employing parametric user simulators to reason about humans via Bayesian inference. Using such simulators allows data-efficient modeling, but their computational cost has typically prevented their practical application. Fortunately, more recent works [Hämäläinen et al., 2023, Moon et al., 2023] have proposed integrating amortized inference to mitigate this issue, enabling real-time user modeling via precomputations.

Despite the proven effectiveness of these user modeling approaches, their lack of formal privacy guarantees limits their deployment in sensitive environments. Existing approaches to privacy-preserving user modeling primarily rely on Federated Learning (FL) [McMahan et al., 2017, Liu et al., 2024, 2023] or meta-learning [Finn et al., 2017]. FL enables distributed centralised training while preserving data locality. However, in FL setup as shown in Fig. 1 (left) imposes significant communication and infrastructure overhead, and often results in global models that underperform on user-specific tasks, particularly in heterogeneous environments with high variability in user behaviour. Differential Privacy meta-learning [Li et al., 2020, Zhou and Bassily, 2022], on the other hand, protects task-specific updates in a federated setting but has not been adapted to uncertainty-aware user models.

To address these challenges, we propose a novel privacy-preserving probabilistic user modeling framework. Our proposed approach leverages Neural Processes for their ability to perform amortized inference, enabling fast, few-shot personalization and uncertainty-aware predictions at the user level. NPs naturally support modeling structured user behavior without requiring centralized orchestration, making them especially well-suited for decentralized privacy-preserving settings. Our framework integrates differentially-private stochastic gradient descent (DP-SGD)[Abadi et al., 2016] with privacy loss distribution (PLD) [Doroshenko et al., 2022] accounting and applies them to Neural Process (NP) [Garnelo et al., 2018] based differentiable user modeling inspired by Hämäläinen et al. [2023]. Unlike prior privacy-aware meta-learning approaches, our method provides tight privacy guarantees through PLD-based composi-

[*]These authors contributed equally.
[†]Currently with Imperial College London but the work was done at University of Manchester.

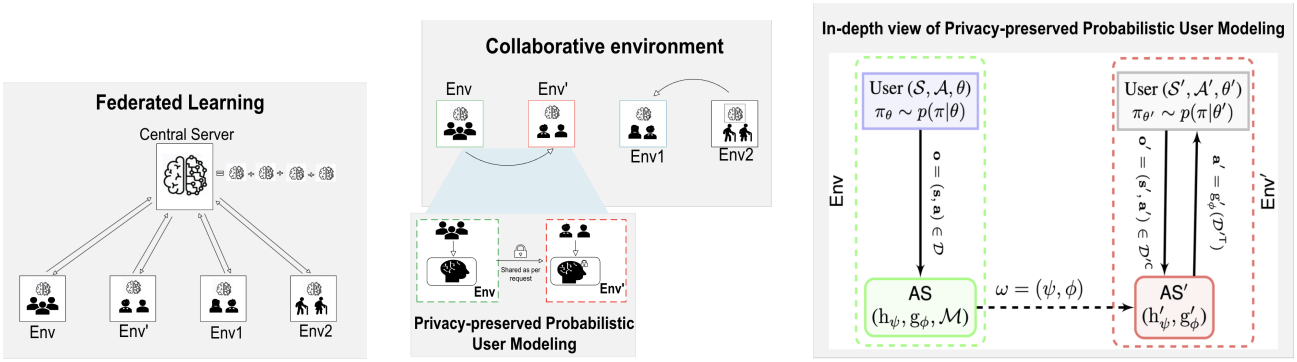

Figure 1: Comparison of Federated Learning (FL)-based user modeling (left) and our proposed Privacy-preserved Probabilistic User Modeling (middle). In FL, a central server aggregates model updates from multiple environments, requiring continuous communication and centralized coordination. In contrast, our framework enables privacy-preserving probabilistic user modeling, where environments can request and share differentially private user models without direct data exchange (a fine-level view of the Env and Env′ is shown on the right).

tion accounting, offering more accurate privacy loss estimation than moment-based techniques [Abadi et al., 2016] and Rényi DP [Mironov, 2017, Feldman and Zrnic, 2020]. Additionally, unlike PATE [Papernot et al., 2018], which requires large datasets for teacher-student learning, our approach is well-suited for small-data, high-sensitivity environments.

To concretize the impact of our contribution, consider a scenario where AI assistants in multiple healthcare centers personalize treatment plans. Direct data sharing is infeasible due to privacy constraints, and while FL could train a global model via encrypted updates, we do not use this centralized FL coordination mechanism. In contrast, our approach leverages meta-learning with neural processes, where each assistant trains a privacy-protected amortized surrogate user model, which can be selectively shared upon request, enabling optional collaboration (not essentially bi-directional) without direct data exchange. Also, these healthcare centers have the autonomy to decide whether to use the shared surrogate model as it is or further fine-tune it with their local data for enhanced performance. For example, an assistant trained on diabetic patients' data can share its surrogate model with another assistant treating metabolic disorders, enabling effective and voluntary collaboration without exposing sensitive data. An elaborated real-world example is discussed in Section 3.1.

The main contributions of this paper are:

- Privacy-preserving probabilistic user modeling via DP-SGD and PLD-based privacy accounting for neural processes, ensuring strong privacy guarantees with adaptive user representations.
- New utility and privacy bounds under non-convex optimization, advancing theoretical insights in privacy-aware meta-learning.
- Scalable and efficient training across diverse user tasks, maintaining high accuracy even under stringent privacy constraints.
- Empirical validation on cognitively justified user modeling tasks, demonstrating competitive accuracy compared to non-private baselines.

Our extensive experiments on diverse user modeling tasks, including grid-world navigation and cognitively justified menu search modeling under various privacy regimes validate our approach, showing its effectiveness in real-world privacy-sensitive applications.

2 RELATED WORKS

User modeling [Fischer, 2001, Strouse et al., 2021] in AI-driven systems requires both uncertainty quantification and privacy guarantees to ensure adaptive and trustworthy interactions while protecting sensitive data. Traditional approaches rely on centralized aggregation, posing significant privacy risks. Recent research in privacy-preserving learning has explored federated learning (FL) [McMahan et al., 2017], differential privacy (DP) [Dwork et al., 2014], and meta-learning [Finn et al., 2017, Zhou and Bassily, 2022] to address these challenges.

Federated learning [McMahan et al., 2017, Liu et al., 2023] allows decentralized learning while keeping data local, but centralized aggregation remains a privacy risk. In [Liu et al., 2023], a personalized federated learning technique enhances user adaptability by leveraging hierarchical structures for improved task-specific generalization. Cross-silo FL [Liu et al., 2024] deals with record-level personalized differential privacy, enabling flexible privacy-utility trade-offs while preserving decentralized model training. Personalized FL allows tailored user models but still relies on continuous communication, limiting adaptability. Our approach eliminates

Method	Model	Update Flow	Data requirement	Co-ordinator	Privacy	User-adaptability
FL	Global	Bi-directional	Large	Central Server	None	None
Cross-silo FL	Global	Bi-directional	Large	Central Server	record/user-specific	None
Personalized FL	Local	Bi-directional	Large	Central Server	None	None
Neural Process (Ours)	Local	Local	Small	None	Environment-specific	High

Table 1: Key differences of our proposed work with existing approaches.

this centralized aggregation by enabling privacy-preserving surrogate user models that can be shared securely across environments.

Meta-learning [Finn et al., 2017] enables fast adaptation to new tasks by leveraging prior knowledge. DP meta-learning [Li et al., 2020] and clustering-based regularization Zhou and Bassily [2022] for task adaptations, protects task-specific updates in a federated setting but does not consider uncertainty-awareness that can model user behaviours probabilistically. We extend this by integrating probabilistic NPs with DP, ensuring flexible, privacy-preserving few-shot adaptation. Neural Processes [Garnelo et al., 2018] provide a probabilistic approach to adaptive user modeling [Hämäläinen et al., 2023], but existing work does not account for privacy risks. Our method incorporates DP-SGD [Song et al., 2013] with Privacy Loss Distribution [Doroshenko et al., 2022], enabling scalable, private uncertainty estimation for interactive AI systems. So, we introduce a novel privacy-preserving framework for adaptive and uncertainty-aware user modeling, facilitating efficient decentralized learning with strong privacy guarantees in complex, non-convex settings. Unlike federated learning, our approach eliminates centralized coordination, enabling AI assistants to securely exchange privacy-protected representations rather than raw model updates.

3 PRELIMINARIES

In this section, we introduce the details needed to formalize and understand our proposed method *privacy protected probabilistic user modeling*. We first examine this problem from the general perspective of building privacy-protected differentiable surrogates for behavioural models. Then, we cast it into differentially private learning of neural processes, and finally discuss the *user-level DP* notion to address the privacy of users' behavioural datasets.

3.1 PRIVACY-PROTECTED PROBABILISTIC USER MODELING

We consider a setting, as shown in Fig. 1 (middle), consisting of a set of collaborative environments. The detailed view of two environments Env (base) and Env' (target), each with an AI-assistant, AS and AS', trying to simulate the behaviour of their set of users in some decision-making tasks as shown Fig. 1 (right). Users are characterised by their internal states θ_u following the distribution $p(\theta)$, $u \in \mathcal{U}$, which govern their behaviour in a specific task described by parameters θ_t, $t \in \mathcal{T}$, following distribution $p(\theta_t)$. A specific user with parameters θ_u and task parameters θ_t is described by the parameter $\theta = (\theta_u, \theta_t) \in \Theta$ with distribution $p(\theta)$. For a given parameter θ, the behavioural policy of a specific user in a task is described by the implicit stochastic process π_θ drawn from the family of probabilistic user models $P(\pi|\theta)$ reflecting uncertainties in the behaviour of users.

Based on this user model, each user in Env who makes a decision about a task, executes a stochastic behavioural policy π_θ on the state $\mathbf{s}_\theta \in \mathcal{S}$ of the environment to generate an action $\mathbf{a}_\theta = \pi_\theta(\mathbf{s}_\theta) \in \mathcal{A}$. The sequence of states and actions generated over a horizon of length n_θ constitutes the user's dataset as $\mathcal{D}_\theta = \bigcup_{i=1}^{n_\theta} \{\mathbf{d}_\theta^i\}$, $\mathbf{d}_\theta^i = (\mathbf{s}_\theta^i, \mathbf{a}_\theta^i) \in \mathcal{S} \times \mathcal{A}$, which is observed by AS. The database of all users' trajectories is $\mathcal{D} = \cup_\theta \mathcal{D}_\theta$. With $\mathcal{D}$, AS learns a surrogate user model, enabling computation of the posterior predictive distribution $Q(\pi(\mathbf{s}')|\mathbf{s}, \mathbf{a}) = \int p(\theta|\mathbf{s}, \mathbf{a}) P(\pi(\mathbf{s}')|\theta) d\theta$ over behaviour policy $\pi \sim P$. Surrogates enable better assistance by anticipating the behaviour of a new user in a new task based only on few observed steps of a trajectory, the so-called *context set* $\mathcal{D}_\theta^{\mathsf{C}} = \{\mathbf{d}_\theta^i\}_{i=1}^{m_\theta}$. Our goal is now to have AS help AS' learn better surrogates of user model, from Env, in similar or related tasks. This could be done, in principle, by sharing parameters of the surrogate user model trained by AS using database $\mathcal{D}_\theta$ from Env. However, to prevent extracting users' sensitive information from parameters of the surrogate model [Fredrikson et al., 2015, Shokri et al., 2017], AS learns and transfers a privacy-protected variant of the surrogate model parameters to AS' for utilisation toward learning new surrogates on Env'.

An illustrating real-world example case of this setting is two healthcare centres (environments), each with their own patients, treatment team and AI-assistant. Each treatment team (users) performs the task of giving personalised treatment for specific types of diseases. The treatment team needs to choose actions, including prescription of various types of medication, radiotherapy, etc. by observing the EHR and health status of patients' (corresponding to the states), the centre's medical equipment and also the specialists' knowl-

edge of the specific disease. The AI-assistant of each centre helps their experts design effective treatments for the patients with similar or related diseases. To this end, the two AI-assistants interact to learn from each other's experience by exchanging information on the diagnostic and treatment actions taken by experts without revealing sensitive information. This sensitive information includes the state of the environment which can be the health status and records of patients (also reflecting their personal identity) under treatment by specialists can be considered private information.

Formally, the objective of AS is to ensure privacy while learning predictive distribution Q_ω, parameterised by $\omega \in \Omega \subset \mathbb{R}^d$, defining a distribution over function $\pi_\theta(\mathbf{s})$ for target states $\mathbf{s} \in \mathcal{D}_\theta^{\mathsf{T}} = \{\mathbf{s}_\theta^i\}_{i=m_\theta}^{n_\theta}$ given $\mathcal{D}_\theta^{\mathsf{C}}$ where $\pi_\theta \sim P$ and $\pi_\theta : \mathcal{S} \to \mathcal{A}$. The intention is for the learning is to adapt well when predicting actions of a new population of users in similar/related tasks and for the target states of Env′. Toward this end, AS learns a privacy-protected model parameter $\omega \in \Omega$ that helps AS′ learn a low risk ω' when testing on new population of users in similar/related tasks. These two goals can be viewed as the training and testing phases of a meta-learning algorithm for few-shot prediction of users' behavioural policies that are performed by AS and AS′, respectively. While the former trains and transfers a privacy-protected parameter ω of surrogate user models from Env, the latter receives and adapts it for learning new surrogate user models that can predict on target data from Env′.

We cast the problem of privacy-protected user modeling into privacy-preserving meta-learning for few-shot stochastic regression of users' behavioural policies. Toward this end, we consider differentially-private learning of neural process models as a latent variable probabilistic model.

3.2 DIFFERENTIAL PRIVACY IN USER MODELING

3.2.1 Definiton of Differential Privacy

Formally, DP is defined as follows:

Definition 1 ($(\varepsilon, \delta) - DP$ mechanism)**.** A probabilistic mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{O}$ is (ε, δ)-differentially private if for some $\varepsilon \geq 0$ and $\delta \in [0, 1]$, any measurable subset $\bar{\mathcal{O}} \subseteq \mathcal{O}$ and for all neighbouring datasets $\mathcal{D}', \mathcal{D}'' \in \mathcal{D}$ differing in just one $\mathcal{D}_\theta$,

$$\mathsf{Pr}\left[\mathcal{M}(\mathcal{D}') \in \bar{\mathcal{O}}\right] \leq e^\varepsilon \mathsf{Pr}\left[\mathcal{M}(\mathcal{D}'') \in \bar{\mathcal{O}}\right] + \delta. \quad (1)$$

3.2.2 User-Level Differential Privacy

AS runs a privacy-preserving mechanism $\mathcal{M}$ during training to satisfy *user-level DP*. This matches our threat model (see Sec. 3.1) in contrast to record-level DP which would only protect a single trajectory [Levy et al., 2021, Xu et al., 2022, Jain et al., 2021]. The notion of user-level privacy enters via the definition of the neighbouring relationship of the datasets. We consider two datasets $\mathcal{D}' = \{\mathcal{D}'_{\theta_u}\}_{\theta_u}$ and $\mathcal{D}'' = \{\mathcal{D}''_{\theta_u}\}_{\theta_u}$ as neighbouring if for some $\theta_u \in \Theta$, $\mathcal{D}'$ and $\mathcal{D}''$ differ only in the dataset related to a specific user i.e. $\mathcal{D}'_{\bar{\theta}_u} = \mathcal{D}''_{\bar{\theta}_u}, \forall \bar{\theta}_u \neq \theta_u$. This guarantees that AS′ cannot distinguish if any specific user dataset was utilised. The ε is the bound on the the privacy loss of database $\mathcal{D}$ and δ denotes a small amount of slack in terms of the probability mass difference of regions where this ε bound may be violated.

Algorithm 1: DP-SGD Training of Neural Process

Input: Population of users $\mathcal{U}$ with distribution $p(\theta_u)$ and corresponding task parameter distribution $p(\theta_t)$ constituting the user-task parameters $\theta = (\theta_u, \theta_t)$ with distribution $p(\theta)$
Input: behaviour generative process $p(\pi|\theta)$
Input: Step size hyperparameter γ, Clipping bound c, Number of iterations T, User sampling rate $q \in (0, 1]$, Privacy budget $\delta \ll 1/|\mathcal{U}|$ and ε
Init: Encoder ψ and decoder ϕ for h_ψ and g_ϕ
Compute required privacy noise:
$\sigma \leftarrow \texttt{privacy\_oracle}(T, q, \varepsilon, \delta)$
for $t = 1$ **to** T **do**
  Sample, with probability q, a batch $\mathcal{B}_l$ of user-task parameters
  foreach $\theta \in \mathcal{B}_l$ **do**
    Generate a trajectory $\mathcal{D}_\theta$ of length n_θ
    Split $\mathcal{D}_\theta$ into $\mathcal{D}_\theta^{\mathsf{C}}$ (context) and $\mathcal{D}_\theta^{\mathsf{T}}$ (target)
    Encode $\mathcal{D}_\theta^{\mathsf{C}}$ as $\mathbf{w} = \frac{1}{|\mathcal{D}_\theta^{\mathsf{C}}|} \sum_{\mathbf{d}_\theta \in \mathcal{D}_\theta} h_\psi(\mathbf{d}_\theta)$
    Compute user-specific gradients: $\mathbf{g}_\theta = (\mathbf{g}_{\theta,\psi}, \mathbf{g}_{\theta,\phi}) = \nabla_{\psi,\phi} \mathrm{L}_\theta(\psi, \phi)$
    Clip gradients: $\bar{\mathbf{g}}_{\theta,\psi} = \min\{1, c/\|\mathbf{g}_{\theta,\psi}\|\}\mathbf{g}_{\theta,\psi}$
    Perturb gradient: $\tilde{\mathbf{g}}_{\theta,\psi} = \bar{\mathbf{g}}_{\theta,\psi} + \mathcal{N}(0, c^2\sigma^2\mathbb{I})$
  Average gradients: $\tilde{\mathbf{g}}_\psi = \frac{1}{|\mathcal{B}_l|} \sum_{\theta \in \mathcal{B}_l} \tilde{\mathbf{g}}_{\theta,\psi}$, $\mathbf{g}_\phi = \frac{1}{|\mathcal{B}_l|} \sum_{\theta \in \mathcal{B}_l} \mathbf{g}_{\theta,\phi}$
  Update parameters: $\psi \leftarrow \psi - \gamma\tilde{\mathbf{g}}_\psi$, $\phi \leftarrow \phi - \gamma\mathbf{g}_\phi$

3.3 DIFFERENTIALLY PRIVATE NEURAL PROCESS

Neural Process are computationally-efficient models, combining Gaussian processes and neural networks, which we use to approximate parameterised surrogates that compute an amortised version of the predictive posterior distribution as $Q_\omega\left(\pi_\theta(\mathcal{D}_\theta^{\mathsf{T}})|\mathcal{D}_\theta^{\mathsf{C}}, \mathcal{D}_\theta^{\mathsf{T}}\right) = \int q_\psi(\mathbf{z}|\mathcal{D}_\theta^{\mathsf{C}}) p_\phi(\pi_\theta(\mathcal{D}_\theta^{\mathsf{T}}|\mathcal{D}_\theta^{\mathsf{T}}, \mathbf{z}) d\mathbf{z}$, $\omega = (\psi, \phi)$, of the user behaviour π_θ at target unseen states. This is performed by considering a joint Gaussian distribution over the values of the policy π_θ at target states of interest, given the context

dataset, followed by a NN to learn the mean and variance parameters of the surrogate distributions in the following three steps that builds the NP model Garnelo et al. [2018]:

1. A parameterised *encoder* $\mathrm{h}_\psi : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$, $\mathcal{R} \subseteq \mathbb{R}^e$, embedding samples $(\mathbf{s}_\theta^i, \mathbf{a}_\theta^i)$ of observed context dataset $\mathcal{D}^{\mathsf{C}}$ to a fixed-dimension representation $\mathbf{r}_\theta^i = h_\psi(\mathbf{a}_\theta^i, \mathbf{a}_\theta^i)$ of user and task parameters θ.
2. A permutation invariant *aggregator*, Agg, computing an order-invariant global representation $\mathbf{r}_\theta = \mathrm{Agg}(\{\mathbf{r}_\theta^i\}_i) = \sum_{i=1}^{m_\theta} \mathbf{r}_\theta^i / m_\theta$ of the context dataset. This parameterises generation of a global latent variable $\mathbf{z} \sim \mathcal{N}(\mu(\mathbf{r}_\theta), \sigma(\mathbf{r}_\theta))$.
3. A parameterised *decoder* $\mathrm{g}_\phi : \mathcal{S} \times \mathcal{Z} \to \mathcal{A}$ predicting the actions for states of interest in target set $\mathcal{D}_\theta^{\mathsf{T}}$ as $\mathbf{a} = \mathrm{g}_\phi(\mathbf{s}, \mathbf{z}), \mathbf{s} \in \mathcal{D}_\theta^{\mathsf{T}}$.

The NP model builds a surrogate posterior $p_\psi(\mathbf{z}|(\mathbf{s}_\theta, \mathbf{a}_\theta))$ and likelihood $p_\phi(\mathbf{a}_\theta|\mathbf{s}_\theta, \mathbf{z})$, by implementing the mappings of encoder h_ψ and decoder g_ϕ with optimisable parameters ψ and ϕ. The goal is to optimise the loss function $\mathrm{L}(\psi, \phi) = \mathsf{E}_{\theta \sim p(\theta), \pi_\theta \sim p(\pi|\theta)} \mathrm{L}_\theta(\psi, \phi)$ with respect to parameters ψ and ϕ for generalisation of the surrogates over the user/task population. $\mathrm{L}_\theta(\psi, \phi) = \mathsf{E}_{(\mathbf{s}_\theta, \mathbf{a}_\theta) \sim \pi_\theta}[\mathcal{L}(\pi_\theta(\mathbf{a}_\theta|\mathbf{s}_\theta), Q_\omega(\mathbf{a}_\theta|\mathbf{s}_\theta, \mathcal{D}_\theta^{\mathsf{C}}))]$ is the loss function for specific user-task parameters, and $\mathcal{L}$ is the loss between simulated and ground-truth behaviours of the users formulated in terms of the evidence lower bound (ELBO) as

$$\begin{aligned} &\mathcal{L}(\pi_\theta(\mathbf{a}_\theta|\mathbf{s}_\theta), Q_\omega(\mathbf{a}_\theta|\mathbf{s}_\theta, \mathcal{D}_\theta^{\mathsf{C}})) \qquad (2) \\ &= \mathsf{E}_{q_\psi(\mathbf{z}|\mathcal{D}_\theta)} \left[\sum_{i=m_\theta+1}^{n_\theta} \log(p_\phi(\mathbf{a}_\theta^i|\mathbf{s}_\theta^i, \mathbf{z})) + \log\left(\frac{q_\psi(\mathbf{z}|\mathcal{D}_\theta^{\mathsf{C}})}{q_\psi(\mathbf{z}|\mathcal{D}_\theta)} \right) \right]. \end{aligned}$$

Here, $q_\psi(\mathbf{z}|\cdot)$ is a variational posterior of the latent variable $\mathbf{z}$ given the respective dataset.

AS performs differentially-private optimisation for this problem to protect privacy of the users' datasets $\mathcal{D}$ while transferring the optimised parameters ψ and ϕ to AS' for meta-testing. These parameters are utilised by AS' as an initialisation that performs well when running the few-shot optimisation process on new user/task populations while not leaking sensitive information about the users' data to AS'.

4 METHOD AND NON-CONVEX THEORETICAL GUARANTEES

In this section, we propose an algorithm to address the optimization process of the privacy-protected user modeling problem based on training a NP model [Garnelo et al., 2018] using DP-SGD [Abadi et al., 2016]. DP-SGD provides a scalable and computationally-efficient differentially-private method to accomplish stochastic gradient descent for non-convex optimisation and is generally applicable with privacy properties that are well understood. This proposes building privacy-protected surrogates while being differentiable and generalisable over user/task population. This will extend the non-private solution for user modeling problem discussed by Hämäläinen et al. [2023].

Our main contribution is to apply this method in a new application, with provable statistical guarantees, for DP few-shot stochastic prediction toward probabilistic user modeling. We present results on utility and privacy guarantees, as well as their trade-off, of the proposed algorithm as a solution for DP meta-learning of the (non-convex) ELBO loss function. For this, we make use of an optimality result of the ELBO given by Damm et al. [2023].

4.1 DP-SGD TRAINING OF NEURAL PROCESSES

For training the NP, a set of policies $\{\pi_\theta\}_\theta$ are sampled from distribution P, and are executed by the simulator at the base environment to generate pairs of states and actions $\mathcal{D}_\theta = \{(\mathbf{s}_\theta^i, \mathbf{a}_\theta^i)\}_i$ with $\mathbf{a}_\theta^i = \pi_\theta(\mathbf{s}_\theta^i)$. During training, this dataset is split into a context dataset $\mathcal{D}_\theta^{\mathsf{C}} = \{(\mathbf{s}_\theta^i, \mathbf{a}_\theta^i)\}_{i=1}^{m_\theta}$ and target dataset $\mathcal{D}_\theta^{\mathsf{T}} = \{(\mathbf{s}_\theta^i, \mathbf{a}_\theta^i)\}_{i=m_\theta+1}^{n_\theta}$ that are fed into the encoder and decoder of the NP as described in Subsection 3.3. We utilise the DP-SGD algorithm for training this NP as described in Alg. 1. First, a batch of users are sampled, with fixed batch size and sampling rate of q, and then the gradient is computed for each user within the batch using samples of data points. Second, the gradient components $\mathbf{g}_\theta(\psi)$, with respect to weights of encoder, are clipped to bound ℓ_2-norm sensitivity followed by perturbation with a calibrated Gaussian noise vector to guarantee privacy. It should be noticed that encoder weights are required to preserve privacy as they are subsequently utilised to generate a latent variable for use at the decoder during testing. Accordingly, the gradient components with respect to the weights of the decoder are not obfuscated. To characterise the noise level σ required to achieve the desired level of privacy parameters in Alg. 1, we resort to a recently established principled way for privacy accounting, based on PLD, over subsequent accesses to the dataset through training iterations. This privacy accountant enables establishing tight upper bounds on the real privacy loss [Doroshenko et al., 2022] to *efficiently compute* a tight δ given the privacy loss budget ε (and the hyperparameters of the composed privacy mechanisms over T training iterations, $\mathbf{\Xi}$), i.e., $\delta(\varepsilon; \mathbf{\Xi})$. In our case, $\mathbf{\Xi} = (T, \sigma, q)$.

We can use these techniques to determine the noise level σ for a set of desired privacy parameters (ε, δ) and known number of iterations T.[1] In Alg. 1, we abstract away the details of this and simply assume that we have a function `privacy_oracle`$(T, q, \varepsilon, \delta)$ that outputs a suitable σ. We choose $\delta = |\mathcal{U}|^{-2}$ following the common advice that

[1] By inverting $\delta(\varepsilon; T, \sigma)$ for σ using any (numerical) root-finding technique.

$\delta \ll |\mathcal{U}|^{-1}$ as it would otherwise allow the leakage of arbitrary raw data points through the algorithm [Dwork et al., 2014]. Alg. 1 is an instantiation of DP-SGD to train NP using PLD accountant. The model weights ψ optimise the loss of generalising the privacy-protected user surrogates over the users population while ϕ optimise the loss of generalisation over the task of predicting unseen states. These weights are transferred to AS' from Env' for meta-testing on a new user-task and have to predict actions for new target datasets containing states of interest.

4.2 UTILITY AND PRIVACY GUARANTEES

In this section, we present our results on the utility and privacy guarantees, as well as their trade-off, for the proposed Algorithm 1 that leverages DP-SGD method [Abadi et al., 2016] for training the NP model [Garnelo et al., 2018]. For the utility guarantee, we derive a new bound on the norm of stochastic gradient vectors, based on the stochastic non-convex optimization analysis of [Ghadimi and Lan, 2013, Agarwal et al., 2018], to account for the noise variance of the perturbed gradients with respect to weights ψ. This enables us to investigate the convergence of the proposed algorithm for DP-SGD learning of the NP toward building privacy-protected probabilistic user models. We also discuss how the perturbed gradients affect the convergence of the algorithm compared with non-private gradients.

These results show that we are able to build differentiable user models that can efficiently adapt to similar or related users/tasks while preserving the privacy of users' datasets for further use in another environment. The impact of this is to facilitate faster adaptation by releasing a privacy-protected experience from one environment to another.

Theorem 1 (Utility Guarantee)**.** *Suppose the neural network has differentiable non-linearities contributing to the λ-smoothness of the loss function* L *with bounded gradients $\|\nabla \mathrm{L}_{\psi,\phi}(\psi,\phi)\| \leq c$, $\forall \psi, \phi$, and the step sizes $\gamma_\psi^t = \gamma_\phi^t = \gamma = \min\left\{\frac{1}{\lambda}, \frac{\sqrt{2\Delta \mathrm{L}^*}}{\sigma_\psi \sqrt{\lambda T}}\right\}$, $t \in [T]$. Then the gradients of the loss function after T iterations of learning can be bounded as*

$$\mathsf{E}\left[\|\nabla_\psi \mathrm{L}(\psi^t,\phi^t)\|^2 + \|\nabla_\phi \mathrm{L}(\psi^t,\phi^t)\|^2\right] \leq \frac{2\Delta \mathrm{L}^* \lambda}{T} + 2\sqrt{\frac{2\lambda \Delta \mathrm{L}^*}{T}} c\sqrt{d_\psi}\sigma_\psi \tag{3}$$

where $\mathsf{E}\left[\|\tilde{\mathbf{g}}_\psi - \nabla \mathrm{L}_\psi(\psi,\phi)\|^2\right] \leq d_\psi c^2 \sigma_\psi^2$ and d_ψ is the dimension of the encoder weight vector. $\Delta \mathrm{L}^*$ is such that $\mathrm{L}(\psi_0,\phi_0) - L^* \leq \Delta \mathrm{L}^*$ with $L^* = \mathsf{E}_{\theta\sim p(\theta)}[L_\theta^*]$ being the optimal loss. L_θ^* is given by the sum of entropy values of the prior and surrogates for the posterior and likelihood as

$$L_\theta^* = \frac{1}{2n_\theta}\sum_{j=1}^{d_z} \log\left(\frac{\varsigma_j^2(\mathcal{D}^{\mathsf{C}})}{\varsigma_j^2(\mathcal{D})}\right) + \frac{1}{2n_\theta}\sum_{i=1}^{n_\theta - m}\sum_{j=1}^{d_{\mathcal{A}}} \log\left(2\pi e \tau_j^2(\mathbf{a},\mathbf{s}^i)\right) \tag{4}$$

$\left(\varsigma_1^2,\ldots,\varsigma_{d_z}^2\right)$ and $\left(\tau_1^2,\ldots,\tau_{d_{\mathcal{A}}}^2\right)$ correspond to the diagonal elements of the encoder and decoder covariance matrices, respectively. See Appendix A for the proof.

The utility guarantee implies that the gradient norm (as used for the convergence analysis of non-convex stochastic optimisation [Ghadimi and Lan, 2013]) decreases as the number of training iterations increases and hence the training method conducts the model toward the optimal value. This is while the increase in noise variance to preserve stricter privacy slows down the convergence.

Next, we present privacy guarantees based on the *subsampled Gaussian* privacy-preserving mechanism. We leverage recent PLD results for accounting privacy loss over compositions of elementary mechanisms [Doroshenko et al., 2022] to show that, for any σ computed by the $\texttt{privacy\_oracle}(T,\varepsilon,q,\delta)$, the algorithm is (ε,δ)-DP. We consider the substitution neighbouring relation and fixed-sized batch sampling (with rate q) without replacement to derive this result.

Privacy accountants enable us to first find the tightest privacy parameter $\delta(\epsilon)$ as a function of the total privacy leakage budget ϵ over T iterations. Second, this helps adjusting the amount of additive artificial noise such that a convergent learning is guaranteed that does not exceed the given privacy leakage budget ϵ through T-fold sequential composition of the privacy-preserving mechanisms during continual observation of the mechanism output. It should be noted that compared to other accountants, PLD-based accounting has been found superior in earlier work of [Doroshenko et al., 2022] by providing tighter bounds on the DP parameters through high-accuracy estimation of the overall privacy parameters after T-fold sequential composition as is the case in iterative learning methods [Sommer et al., 2019].

Theorem 2 (Privacy Guarantee)**.** *DP-SGD training of the NP is user-level (ε,δ)-DP, using the subsampled Gaussian mechanism with variance $\sigma = \delta^{-1}(\varepsilon,\delta;T,q)$, and fixed batch sampling of rate q at each iteration for the T-iteration learning algorithm 1.* Proof is provided in Appendix B.

Finally, to find the trade-off between utility and privacy guarantees in this problem, we first provide an analytical upper bound on the variance of the noise that satisfies the privacy budget. To this end, first, we use the known result that subsampling amplifies privacy [Kasiviswanathan et al., 2008], to analytically bound $\delta(\varepsilon)$ of the subsampled mechanism with that of the pure Gaussian mechanism [Sommer et al., 2019],

$$\delta(\varepsilon;T,q,\sigma) \leq \frac{1}{2}\left[\mathsf{erfc}\left(\frac{\varepsilon\sigma - T/2\sigma}{\sqrt{2T}}\right) - e^{\varepsilon}\mathsf{erfc}\left(\frac{\varepsilon\sigma + T/2\sigma}{\sqrt{2T}}\right)\right]_+ \leq \frac{1}{2}\mathsf{erfc}\left(\frac{\varepsilon\sigma - T/2\sigma}{\sqrt{2T}}\right). \tag{5}$$

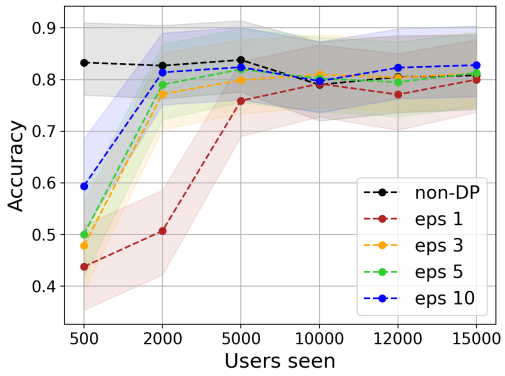

Figure 2: Performance of user models trained with different number of users (500 to 15,000) in a non-differentially private setting (black) and differentially private setting with $\varepsilon = 1$ (red), 3 (orange), 5 (green), and 10 (blue).

The above holds for $q \leq \frac{e^{2\varepsilon}-1}{2e^{2\varepsilon}-1}$; see Appendix C for further details. Then, the upper bound on σ is obtained by solving (5) for σ as

$$\sigma_\psi = \sigma_\phi \leq \frac{\sqrt{T}}{\sqrt{2}\varepsilon}\left[\mathsf{erfc}^{-1}(2\delta) + \sqrt{\varepsilon + \left(\mathsf{erfc}^{-1}(2\delta)\right)^2}\right]. \quad (6)$$

Accordingly, we can now state the following result for the trade-off rule by inserting into Eq. (3):

Corollary 1 (Trade-off analysis). *Supposing the ELBO function is λ-smooth and has bounded gradients, then for the step sizes γ_ψ and γ_ϕ as in Thm. 1, the gradients of the loss function can be bounded as*

$$\mathsf{E}\left[\|\nabla_{(\psi,\phi)}\mathrm{L}(\psi^t, \phi^t)\|^2\right] \quad (7)$$
$$\leq \frac{2\Delta\mathrm{L}^*\lambda}{T} + 4c\sqrt{\frac{\lambda\Delta\mathrm{L}^* d}{\varepsilon}}\left[\mathsf{erfc}^{-1}(2\delta) + \sqrt{\varepsilon + \left(\mathsf{erfc}^{-1}(2\delta)\right)^2}\right]$$

5 EXPERIMENTS

In this section, we include three experiments for comparing the accuracy of the DP-protected user model surrogates against their non-DP counterparts under various privacy budgets. We provide analysis on the method's performance in terms of accuracy (not expressed in percentage) for different numbers of new users (# users) seen as well as the amount of their user behaviour data seen. We use a fixed clipping bound of $c = 2$ as a hyperparameter in our experiments.

5.1 GRIDWORLD ENVIRONMENT

The first experiment is the benchmark setting used by Hämäläinen et al. [2023], evaluating the method's ability to act as a surrogate for a model describing simulated agents (users) on a 10×10 gridworld environment. The gridworld environment is defined as a POMDP with deterministic transition dynamics. Here, the behaviour of agents in the agent population are assumed to emerge as a result of Monte Carlo Tree Search (MCTS)-based optimization. Each agent is defined by an individual reward function, observations and MCTS parameters. This results in a wide variety of different behaviours over the full population. We train the method on data generated by sampling individual users, as detailed in Section 3.1. The environment state captures the current location of the user, which the user can change by moving into one of its adjacent states. The reward function, unique to each user, always assigns one environment state with a positive reward and a negative reward. The initial user location for each episode is chosen at random. Further details are included in Appendix D.1. The modeling task for the AI assistant is to predict the subsequent behaviour of a user, given access to some previous context behaviour of that particular user in different tasks, and the behaviour during the current task. The context observations correspond to the trajectories of state-action pairs performed by the user. Each task is assumed to correspond to an episode in the same environment (sampled independently), but each starting from a different initial state. The number of context trajectories and trajectory length can vary between each agent and task. All other information, such as the reward function and MCTS parameters, are hidden from the assistant.

5.1.1 On users seen

Fig. 2 shows the performance of the models when trained with different numbers of users. The lower the number of users, the gap between the accuracy of the private with tighter privacy restriction, and non-private model is higher. However, it is also clear that for small data, the models with less privacy restrictions have comparable performance without sacrificing the actual accuracy. As we have more users, the gap between private neural process with tighter privacy restrictions is reduced significantly. The model trained with 5000 users achieves a comparable performance in tighter to lighter differential privacy bound. The differentially private models which have more users than 5000 have similar behaviour of a non-private model. This means one can guarantee privacy along with same utility of a non-private neural process. The consistent behaviour of a non-DP neural process that uses more users for training can have diminishing benefits as evident from Fig. 2. This reduces the transfer risk of the trained user models in one environment to the other even under a strict privacy regime.

5.1.2 On behaviours seen

Fig. 3 shows the generalization of the user models trained with 500, 2000 and 5000 users and prediction of their behaviours. Unsurprisingly, as the number of observed be-

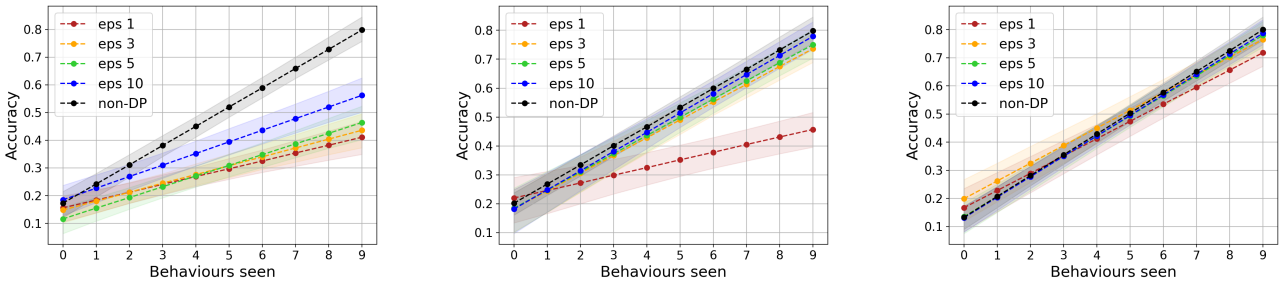

Figure 3: Accuracy in terms of the number of observed behaviours for non-private (black) and private models under privacy budgets, $\varepsilon = 1, 3, 5$, and 10, for neural process user models trained with 500 (Left), 2000 (middle), and 5000 (right) users.

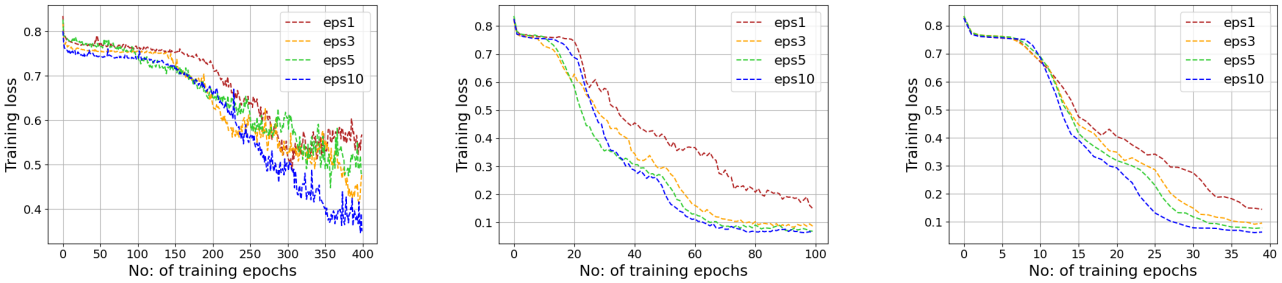

Figure 4: Convergence of the differentially private neural process during training with different number of users : 500 (Left), 5000 (center), 10,000 (right), under privacy budgets ($\varepsilon(\text{eps}) = 1\ (\text{red}), 3\ (\text{orange}), 5\ (\text{green}), 10\ (\text{blue})$). As the number of users increases, convergence is achieved faster (note the different x-axis scales).

haviours per user increases, the accuracy also increases for all privacy budgets and for all the user models.

For smaller ε, the difference between the non-private model is larger. However, this gap tends to decrease as the number of trajectories increases. The model performance also is quite similar to the non-private models as we increase the number of users to train the user models. It is worth noticing that the amount of data received from an individual user is very small and training models with data only from individual users results in significantly poorer performance, as implied by the decreasing modeling performance when the amount of users decreases.

5.1.3 On convergence

The utility bound of Theorem 1 implies that the norm of the gradient (as used for convergence analysis of non-convex stochastic optimization Ghadimi and Lan [2013]) decreases as the number of training iterations (No: of training epochs) increases. As evident from Fig. 5 and Fig. 4, where both the norm of the gradient and the training loss converge (stabilize) with increase in training epochs (No: of training epochs). The initial increase in the gradient norm is associated with stability issues early in training. However, as training progresses, the norm values gradually converge. It is also important to note that the convergence becomes slower when we target stricter privacy, which is obvious in differential privacy. Also, the trend of the gradient norm across epochs aligns with the loss curves, supporting our observation that models with fewer users require more training epochs to converge, while increasing the number of users leads to faster convergence.

5.2 MENU SEARCH ENVIRONMENT

In our second experiment, we test our method with simulated users following the Menu Search model of Kangasrääsiö et al. [2017]. This cognitive model captures the generative process behind human search behaviour, when they are searching for a specific item in a computer dropdown menu. This model is based on *computational rationality* [Gershman et al., 2015] and models how the user behaviour emerges as optimal behaviour that is constrained by the cognitive limitations of the users. For further details of the menu search environments, see Appendix D.2 and D.2.2 of the supplementary material and [Hämäläinen et al., 2023, Kangasrääsiö et al., 2017].

Similarly as in the first experiment, the tasks consider modeling the behaviours of individual users, here generated by the Menu Search model. In each task, observations come from multiple simulations performed by each user in different menu layouts. We evaluate the method's ability to model

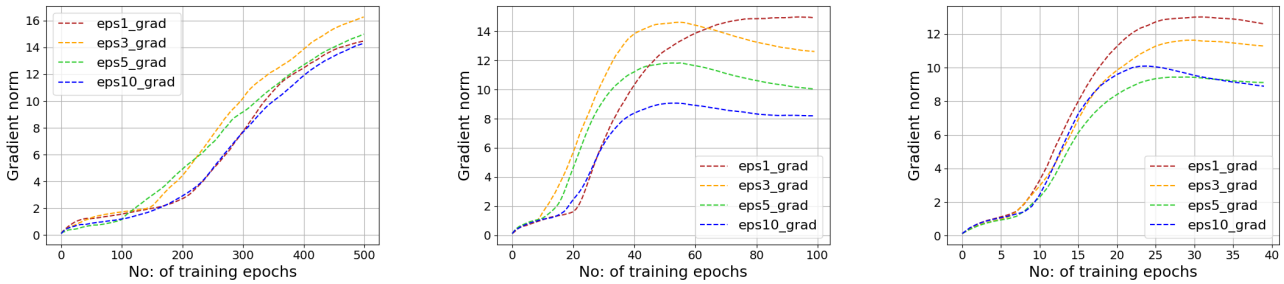

Figure 5: Norm of gradient Vs Training Epochs: Convergence of the norm of gradient in differentially private neural process during training with different number of users: 500 (left), 5000 (middle), and 10000 (right), under privacy budgets ($\varepsilon = 1$ (red), 3 (orange), 5 (green), 10 (blue)). As the number of users increases, convergence is achieved faster (note the different x-axis scales). The model trained with 500 users converge slower, however convergence can still be achieved with more training epochs.

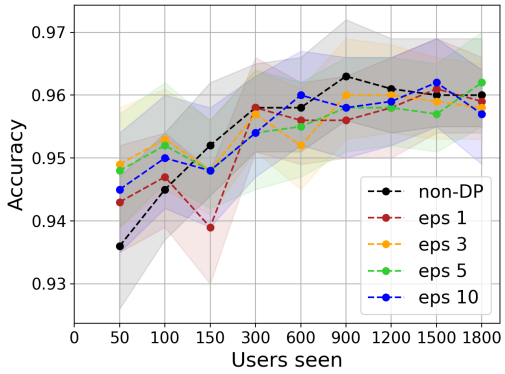

Figure 6: Performance of user models trained with different number of users (50 to 1800) in a non-differentially private setting (black) and differentially private setting with $\varepsilon = 1$ (red), 3 (orange), 5 (green), and 10 (blue).

the user behaviour in previously unseen menu layouts.

We train user models with 50, 100, 150, 300, 600, 900, 1200, 1500, and 1800 users under strict to modest privacy budgets of $\varepsilon = 1, 3, 5, 10$ and a non-DP Neural process model for each. The results are illustrated in terms of test accuracy evaluations when presented with 5000 new users (test) is shown in Fig. 6. In this setup we can see there is no major gap in accuracy differences exists between non-DP and DP Neural processes. It also important to note that the DP based user model with tighter to relaxed privacy trained with 50 users even have slightly better test accuracy.

5.2.1 Menu Search AI-assistant

We examine the integration of a differentially private AI assistant into an interactive system within a simulated environment (refer Appendix D.2.2 and [Hämäläinen et al., 2023]). Our study enhances a structured search space with hierarchical navigation and adaptive guidance. The assistant models user intent through observed behaviours while balancing assistance and autonomy via interaction constraints. Using data from 300 users, we compare accuracy between non-DP (0.718) and DP models with $\varepsilon = 1, 3, 5, 10 (0.685, 0.689, 0.67, 0.681)$. Results show minimal accuracy differences across privacy settings.

Limitation and Challenges : The reported results use ACNPs [Kim et al., 2019] although ANPs achieve the highest accuracy in [Hämäläinen et al., 2023]. Our experiments identified instability caused by exploding gradients, even in non-DP training. The introduction of differential privacy further exacerbated this issue, intensifying gradient explosions and significantly degrading model performance. While the choice of the NP design does not affect our conclusions, future work might want to consider recent alternatives such as Transformer Neural Processes [Nguyen and Grover, 2022] due to the known stability issues with the NP-ELBO loss.

6 CONCLUSION

We proposed a privacy-preserving probabilistic user modeling framework that integrates neural processes with DP-SGD, enabling differentially private few-shot predictions while maintaining real-time inference. Our approach balances privacy and utility, achieving competitive accuracy under strict privacy constraints across diverse user modeling tasks. Empirical results showed that as user data increases, the performance gap between private and non-private models diminishes, supporting the feasibility of privacy-aware surrogates. Additionally, we established theoretical guarantees on privacy-utility trade-offs in non-convex optimization. This work advances privacy-conscious AI assistants for sensitive applications like healthcare and personalization, with potential for further improvements in adaptability and robustness.

Acknowledgements

The authors thank Jonathan Taylor from Research IT at the University of Manchester for his support in developing the code for model training and evaluation, and all anonymous reviewers for their constructive feedback. The authors also would like to acknowledge the assistance given by Research IT and the use of the Computational Shared Facility at The University of Manchester. This work was supported by the Research Council of Finland Flagship programme: Finnish Center for Artificial Intelligence FCAI and decisions 358958, 359567. Amir Sonee, Haripriya Harikumar, and Samuel Kaski were supported by the UKRI Turing AI World-Leading Researcher Fellowship, [EP/W002973/1].

References

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. cpSGD: Communication-efficient and differentially-private distributed SGD. In *Proceedings of 32nd Advances in Neural Information Processing Systems*, 2018.

Simon Damm, Dennis Forster, Dmytro Velychko, Zhenwen Dai, Asja Fischer, and Jörg Lücke. The ELBO of variational autoencoders converges to a sum of entropies. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*. PMLR, 2023.

Sebastiaan De Peuter, Shibei Zhu, Yujia Guo, Andrew Howes, and Samuel Kaski. Preference learning of latent decision utilities with a human-like model of preferential choice. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 123608–123636, 2024.

Vadym Doroshenko, Badih Ghazi, Pritish Kamath, Ravi Kumar, and Pasin Manurangsi. Connect the dots: Tighter discrete approximations of privacy loss distributions. *Proceedings on Privacy Enhancing Technologies*, 2022.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 2014.

Vitaly Feldman and Tijana Zrnic. Individual privacy accounting via a Rényi filter. In *Proceedings of 34th Advances in Neural Information Processing Systems*, 2020.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Gerhard Fischer. User modeling in human–computer interaction. *User modeling and user-adapted interaction*, 11: 65–86, 2001.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings ACM SIGSAC Conference on Computation, Communication and Security*, 2015.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural processes. In *Theoretical Foundations and Applications of Deep Generative Models Workshop, 35th International Conference on Machine Learning*, 2018.

Samuel J Gershman, Eric J Horvitz, and Joshua B Tenenbaum. Computational rationality: A converging paradigm for intelligence in brains, minds, and machines. *Science*, 2015.

Saeed Ghadimi and Guanghui Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 2013.

Alex Hämäläinen, Mustafa Mert Çelikok, and Samuel Kaski. Differentiable user models. In *Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence*. PMLR, 2023.

Alex Hämäläinen, Mustafa Mert Çelikok, and Samuel Kaski. Differentiable user models. In *Uncertainty in Artificial Intelligence*. PMLR, 2023.

Daolang Huang, Yujia Guo, Luigi Acerbi, and Samuel Kaski. Amortized bayesian experimental design for decision-making. *Advances in Neural Information Processing Systems*, 37:109460–109486, 2024.

Prateek Jain, J Keith Rush, Adam Smith, Shuang Song, and Abhradeep Guha Thakurta. Differentially private model personalization. In *Proceedings of 35th Advances in Neural Information Processing Systems*, 2021.

Saurav Jha, Dong Gong, Xuesong Wang, Richard E Turner, and Lina Yao. The neural process family: Survey, applications and perspectives. *arXiv preprint arXiv:2209.00517*, 2022.

Antti Kangasrääsiö, Kumaripaba Athukorala, Andrew Howes, Jukka Corander, Samuel Kaski, and Antti Oulasvirta. Inferring cognitive models from data using approximate bayesian computation. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017.

Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, 2008. doi: 10.1109/FOCS.2008.27.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *arXiv preprint arXiv:1901.05761*, 2019.

Antti Koskela, Joonas Jälkö, and Antti Honkela. Computing tight differential privacy guarantees using FFT. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. PMLR, 2020.

Daniel Levy, Ziteng Sun, Kareem Amin, Satyen Kale, Alex Kulesza, Mehryar Mohri, and Ananda Theertha Suresh. Learning with user-level privacy. In *Proceedings of 35th Advances in Neural Information Processing Systems*, 2021.

Jeffrey Li, Mikhail Khodak, Sebastian Caldas, and Ameet Talwalkar. Differentially private meta-learning. In *8th International Conference on Learning Representations*, 2020.

Junxu Liu, Jian Lou, Li Xiong, Jinfei Liu, and Xiaofeng Meng. Cross-silo federated learning with record-level personalized differential privacy. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024.

Qi Liu, Jinze Wu, Zhenya Huang, Hao Wang, Yuting Ning, Ming Chen, Enhong Chen, Jinfeng Yi, and Bowen Zhou. Federated user modeling from hierarchical information. *ACM Transactions on Information Systems*, 2023.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 2017.

Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, 2017. doi: 10.1109/CSF.2017.11.

Hee-Seung Moon, Antti Oulasvirta, and Byungjoo Lee. Amortized inference with user simulations. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023.

Tung Nguyen and Aditya Grover. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. *arXiv preprint arXiv:2207.04179*, 2022.

Antti Oulasvirta, Jussi PP Jokinen, and Andrew Howes. Computational rationality as a theory of interaction. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, 2022.

Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with PATE. In *International Conference on Learning Representation*, 2018.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *Proc. 38th IEEE Symp. Security and Privacy*, 2017.

David M. Sommer, Esfandiar Mohammadi, and Sebastian Meiser. Privacy loss classes: The central limit theorem in differential privacy. *Proceedings on Privacy Enhancing Technologies*, 2019.

Shuang Song, Kamalika Chaudhuri, and Anand D. Sarwate. Stochastic gradient descent with differentially private updates. In *Proceedings of IEEE Global Conference on Signal and Information Processing*, 2013.

DJ Strouse, Kevin McKee, Matt Botvinick, Edward Hughes, and Richard Everett. Collaborating with humans without human data. *Advances in Neural Information Processing Systems*, 34:14502–14515, 2021.

Zheng Xu, Maxwell Collins, Yuxiao Wang, Liviu Panait, Sewoong Oh, Sean Augenstein, Ting Liu, Florian Schroff, and H. Brendan McMahan. Learning to generate image embeddings with user-level differential privacy, 2022. URL `https://arxiv.org/abs/2211.10844`.

Zeping Yu, Jianxun Lian, Ahmad Mahmoody, Gongshen Liu, and Xing Xie. Adaptive user modeling with long and short-term preferences for personalized recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 2019.

F. Yuan, X. He, A. Karatzoglou, and L. Zhang. Parameter-efficient transfer from sequential behaviors for user modeling and recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.

Xinyu Zhou and Raef Bassily. Task-level differentially private meta learning. In *Proceedings of 36th Advances in Neural Information Processing Systems*, 2022.

Supplementary Material

The proofs of our theorems are provided in Sections A through C, and the experiment details are presented in Section D.

A PROOF OF THEOREM 1

In this section, the convergence of the DP-SGD meta-learning algorithm is discussed for learning nonconvex ELBO loss function to build DP probabilistic user model. To this end, we need to investigate the gradient of the loss function $\mathsf{E}\left[\nabla \mathrm{L}(\psi^t, \phi^t)\right]$ or the optimality gap $\mathsf{E}[\mathrm{L}\left(\psi^T, \phi^T\right)] - \mathrm{L}(\psi^*, \phi^*)$ after T rounds of training iterations. As the loss function is not convex in terms of the parameters ψ and ϕ, the common analysis of the DP-SGD meta-learning for convex functions [Zhou and Bassily, 2022, Li et al., 2020], can not be extended to this setting straightforwardly. We provide a convergence analysis based on the DP-SGD method of [Agarwal et al., 2018] for stochastic non-convex optimization studied in [Ghadimi and Lan, 2013]. We modify this analysis to consider the optimization in terms of both ψ and ϕ.

Considering the ELBO function is λ-smooth (having Lipschitz continuous gradients), i.e. $\forall \psi, \psi' \in \mathbb{R}^{d_\psi}$ and $\phi, \phi' \in \mathbb{R}^{d_\phi}$ we have $\|\nabla \mathrm{L}(\psi', \phi') - \nabla \mathrm{L}(\psi, \phi)\| \leq \lambda \|(\psi', \phi') - (\psi, \phi)\|$, the gap of loss function for two consecutive iterations can be formulated as follows considering parameter updates $\psi^{t+1} = \psi^t - \gamma_\psi \tilde{\mathbf{g}}_\psi^t$, $\phi^{t+1} = \phi^{t+1} - \gamma_\phi \mathbf{g}_\phi^t$ and $\boldsymbol{\xi}_\psi^t \triangleq \nabla_\psi \mathrm{L}(\psi^t, \phi^t) - \tilde{\mathbf{g}}_\psi^t$, $\boldsymbol{\xi}_\phi^t \triangleq \nabla_\phi \mathrm{L}(\psi^t, \phi^t) - \mathbf{g}_\phi^t$ at iteration $t+1$ where ∇_ψ, ∇_ϕ denote for the gradient with respect to weights ψ and ϕ, respectively.

$$
\begin{aligned}
\mathrm{L}\left(\psi^{t+1}, \phi^{t+1}\right) - \mathrm{L}\left(\psi^t, \phi^t\right) &\overset{(a)}{\leq} \nabla_\psi \mathrm{L}^\mathsf{T}(\psi^t, \phi^t)(\psi^{t+1} - \psi^t) + \nabla_\phi \mathrm{L}^\mathsf{T}(\psi^t, \phi^t)(\phi^{t+1} - \phi^t) \\
&\quad + \frac{\lambda}{2}\|\psi^{t+1} - \psi^t\|^2 + \frac{\lambda}{2}\|\phi^{t+1} - \phi^t\|^2 \\
&\overset{(b)}{=} -\gamma_\psi \nabla_\psi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right) \tilde{\mathbf{g}}_\psi^t - \gamma_\phi \nabla_\phi \mathrm{L}^\mathsf{T}(\psi^t, \phi^t) \mathbf{g}_\phi^t \\
&\quad + \frac{\lambda}{2}\gamma_\psi^2 \|\tilde{\mathbf{g}}_\psi^t\|^2 + \frac{\lambda}{2}\gamma_\phi^2 \|\mathbf{g}_\phi^t\|^2 \\
&\leq -\gamma_\psi \nabla_\psi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right)\left(\nabla_\psi \mathrm{L}(\psi^t, \phi^t) - \boldsymbol{\xi}_\psi^t\right) + \frac{\lambda}{2}\gamma_\psi^2 \|\nabla_\psi \mathrm{L}(\psi^t, \phi^t) - \boldsymbol{\xi}_\psi^t\|^2 \\
&\quad -\gamma_\phi \nabla_\phi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right)\left(\nabla_\phi \mathrm{L}(\psi^t, \phi^t) - \boldsymbol{\xi}_\phi^t\right) + \frac{\lambda}{2}\gamma_\phi^2 \|\nabla_\phi \mathrm{L}(\psi^t, \phi^t) - \boldsymbol{\xi}_\phi^t\|^2 \\
&= -\gamma_\psi \left(1 - \frac{\lambda\gamma_\psi}{2}\right) \|\nabla_\psi \mathrm{L}(\psi^t, \phi^t)\|^2 + \gamma_\psi(1 - \lambda\gamma_\psi)\nabla_\psi \mathrm{L}^\mathsf{T}(\psi^t, \phi^t)\boldsymbol{\xi}_\psi^t + \frac{\lambda}{2}\gamma_\psi^2 \|\boldsymbol{\xi}_\psi^t\|^2 \\
&\quad -\gamma_\phi \left(1 - \frac{\lambda\gamma_\phi}{2}\right) \|\nabla_\phi \mathrm{L}(\psi^t, \phi^t)\|^2 + \gamma_\phi(1 - \lambda\gamma_\phi)\nabla_\phi \mathrm{L}^\mathsf{T}(\psi^t, \phi^t)\boldsymbol{\xi}_\phi^t + \frac{\lambda}{2}\gamma_\phi^2 \|\boldsymbol{\xi}_\phi^t\|^2 \quad (8)
\end{aligned}
$$

where (a) follows by the smoothness condition, (b) follows by the SGD updates of learning rates γ_ψ and γ_ϕ. Considering $\gamma = \max\{\gamma_\psi, \gamma_\phi\} \leq 1/\lambda$, and taking the summation over the whole iterations, then the sum of the norm of the gradient vectors with respect to both parameters can be bounded as

$$
\begin{aligned}
&\gamma\left(1 - \frac{\lambda\gamma}{2}\right) \sum_{t=1}^{T} \left(\|\nabla_\psi \mathrm{L}(\psi^t, \phi^t)\|^2 + \|\nabla_\phi \mathrm{L}(\psi^t, \phi^t)\|^2\right) \\
&\quad \leq \sum_{t=1}^{T} \Big[\mathrm{L}\left(\psi^t, \phi^t\right) - \mathrm{L}\left(\psi^{t+1}, \phi^{t+1}\right) + \gamma_\psi(1 - \lambda\gamma_\psi)\nabla_\psi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right)\boldsymbol{\xi}_\psi^t \\
&\qquad\quad + \gamma_\phi(1 - \lambda\gamma_\phi)\nabla_\phi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right)\boldsymbol{\xi}_\phi^t + \frac{\lambda\gamma^2}{2}\left(\|\boldsymbol{\xi}_\phi^t\|^2 + \|\boldsymbol{\xi}_\phi^t\|^2\right)\Big] \\
&\quad = \sum_{t=1}^{T} \Big[\gamma_\psi(1 - \lambda\gamma_\psi)\nabla_\psi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right)\boldsymbol{\xi}_\psi^t + \gamma_\phi(1 - \lambda\gamma_\phi)\nabla_\phi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right)\boldsymbol{\xi}_\phi^t \\
&\qquad\quad + \frac{\lambda\gamma^2}{2}\left(\|\boldsymbol{\xi}_\phi^t\|^2 + \|\boldsymbol{\xi}_\phi^t\|^2\right)\Big] + \mathrm{L}\left(\psi^0, \phi^0\right) - \mathrm{L}\left(\psi^T, \phi^T\right) \\
&\quad \leq \sum_{t=1}^{T} \Big[\gamma_\psi(1 - \lambda\gamma_\psi)\nabla_\psi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right)\boldsymbol{\xi}_\psi^t + \gamma_\phi(1 - \lambda\gamma_\phi)\nabla_\phi \mathrm{L}^\mathsf{T}\left(\psi^t, \phi^t\right)\boldsymbol{\xi}_\phi^t
\end{aligned}
$$

$$+\frac{\lambda\gamma^2}{2}\left(\|\boldsymbol{\xi}_{\phi}^{t}\|^2+\|\boldsymbol{\xi}_{\phi}^{t}\|^2\right)\Big]+\mathrm{L}\left(\psi^0,\phi^0\right)-\mathrm{L}\left(\psi^*,\phi^*\right) \tag{9}$$

According to the perturbation in Alg. 1, $\tilde{\mathbf{g}}_{\psi}^{t}$ and $\mathbf{g}_{\phi}^{t}$ are unbiased estimators of the true gradients and so $\mathsf{E}[\xi_{\psi}^{t}]=\mathsf{E}[\xi_{\phi}^{t}]=0$. Taking the expectation of both sides of the above inequality, we have

$$\begin{aligned}
\mathsf{E}\left[\|\nabla_{\psi}\mathrm{L}(\psi^t,\phi^t)\|^2 \;+\; \|\nabla_{\phi}\mathrm{L}(\psi^t,\phi^t)\|^2\right] & \\
&\leq \frac{2}{T\gamma(2-\lambda\gamma)}\left(\left(\mathrm{L}\left(\psi^0,\phi^0\right)-\mathrm{L}\left(\psi^*,\phi^*\right)\right)+\frac{\lambda}{2}\gamma^2T\mathsf{E}\left[\|\boldsymbol{\xi}_{\psi}^{t}\|^2+\|\boldsymbol{\xi}_{\phi}^{t}\|^2\right]\right)\\
&\leq \frac{2}{T\gamma(2-\lambda\gamma)}\left(\Delta\mathrm{L}^*+\frac{\lambda}{2}\gamma^2Tc^2d_{\psi}\sigma_{\psi}^2\right)\\
&\overset{(c)}{\leq} \frac{2}{T\gamma}\left(\Delta\mathrm{L}^*+\frac{\lambda}{2}\gamma^2Tc^2d_{\psi}\sigma_{\psi}^2\right)\\
&= \frac{2}{T\gamma}\Delta\mathrm{L}^*+\lambda\gamma c^2d_{\psi}\sigma_{\psi}^2\\
&\overset{(d)}{\leq} \frac{2}{T}\Delta\mathrm{L}^*\max\left\{\lambda,\frac{\sigma_{\psi}\sqrt{T}}{\tilde{\Delta}_{\psi}}\right\}+\lambda\frac{\tilde{\Delta}_{\psi}}{\sigma_{\psi}\sqrt{T}}c^2d_{\psi}\sigma_{\psi}^2\\
&= \frac{2}{T}\Delta\mathrm{L}^*\lambda+\frac{2\sigma_{\psi}\Delta\mathrm{L}^*}{\sqrt{T}\tilde{\Delta}_{\psi}}+\lambda\frac{\tilde{\Delta}_{\psi}}{\sqrt{T}}c^2d_{\psi}\sigma_{\psi}\\
&= \frac{2}{T}\Delta\mathrm{L}^*\lambda+\frac{\sigma_{\psi}}{\sqrt{T}}\left(\frac{2\Delta\mathrm{L}^*}{\tilde{\Delta}_{\psi}}+\lambda\tilde{\Delta}_{\psi}c^2d_{\psi}\right)\\
&\overset{(e)}{=} \frac{2}{T}\Delta\mathrm{L}^*\lambda+2\sqrt{\frac{2\lambda\Delta\mathrm{L}^*}{T}}c\sqrt{d_{\psi}}\sigma_{\psi}
\end{aligned} \tag{10}$$

where (d) follows by the assumption of $\gamma=\min\{\frac{1}{\lambda},\frac{\tilde{\Delta}_{\psi}}{\sigma_{\psi}\sqrt{T}}\}$ for step size and (e) follows since $\tilde{\Delta}_{\psi}=\sqrt{2\Delta\mathrm{L}^*/\lambda d_{\psi}}c$ and tightens the bound. $\mathrm{L}\left(\psi^*,\phi^*\right)$ is the optimal value of the loss function that can be related to the parameters of the surrogates as follows.

In our setting, ELBO can be specified and simplified to account for the encoding and decoding mappings of the user modeling problem as

$$\begin{aligned}
\mathcal{L} &= \mathsf{E}_{\mathbf{z}\sim q_{\psi}(\mathbf{z}|\mathcal{D}_{\theta})}\left[\sum_{i=m_{\theta}+1}^{n_{\theta}}\log\left(p_{\phi}(\mathbf{a}_{\theta}^{i}|\mathbf{s}_{\theta}^{i},\mathbf{z})\right)+\log\left(\frac{q_{\psi}\left(\mathbf{z}|\mathcal{D}_{\theta}^{\mathsf{C}}\right)}{q_{\psi}(\mathbf{z}|\mathcal{D}_{\theta})}\right)\right]\\
&= \mathsf{E}_{\mathbf{z}\sim q_{\psi}(\mathbf{z}|\mathcal{D}_{\theta})}\left[\sum_{i=m_{\theta}+1}^{n_{\theta}}\log\left(p_{\phi}(\mathbf{a}_{\theta}^{i}|\mathbf{s}_{\theta}^{i},\mathbf{z})\right)+\log\left(q_{\psi}\left(\mathbf{z}|\mathcal{D}_{\theta}^{\mathsf{C}}\right)\right)\right]+H\left(q_{\psi}(\mathbf{z}|\mathcal{D}_{\theta})\right)
\end{aligned} \tag{11}$$

where H is the entropy function, and the surrogate posterior (Gaussian encoder) and likelihood (Gaussian decoder) are given by

$$\begin{aligned}
q_{\psi}\left(\mathbf{z}|\mathcal{D}_{\theta}^{\mathsf{C}}\right) &= \mathcal{N}\left(\mu_{\psi}(\mathcal{D}_{\theta}^{\mathsf{C}}),\mathbf{K}_{\psi}(\mathcal{D}_{\theta}^{\mathsf{C}})\right)\\
p_{\phi}\left(\mathbf{a}_{\theta}^{i}|\mathbf{s}_{\theta}^{i},\mathbf{z}\right) &= \mathcal{N}\left(\nu_{\phi}(\mathbf{z},\mathbf{s}_{\theta}^{i}),\boldsymbol{\Sigma}_{\phi}(\mathbf{z},\mathbf{s}_{\theta}^{i})\right).
\end{aligned} \tag{12}$$

The mean $\mu_{\psi}(\mathcal{D}_{\theta}^{\mathsf{C}})$ and covariance $\mathbf{K}_{\psi}(\mathcal{D}_{\theta}^{\mathsf{C}})$ of the encoder are parameterised by two NNs (neural network), NN_{μ} and NN_{ς} with parameters $\mathbf{V}$ and $\mathbf{T}$, respectively, consisting of all the weight matrices and biases of the corresponding NN for the mean and variance of the encoder:

$$\begin{aligned}
\mu_{\psi}(\mathcal{D}_{\theta}^{\mathsf{C}}) &= \frac{1}{m_{\theta}}\sum_{\mathbf{d}_{\theta}^{i}\in\mathcal{D}_{\theta}^{\mathsf{C}}}\mathrm{h}_{\psi}\left(\mathbf{d}_{\theta}^{i}\right),\;\; \mathrm{h}_{\psi}(\mathbf{d}_{\theta}^{i})=\mathrm{NN}_{\mu}\left(\mathbf{d}_{\theta}^{i};\mathbf{V}\right)\\
\mathbf{K}_{\psi}(\mathcal{D}_{\theta}^{\mathsf{C}}) &= \mathsf{diag}\left(\varsigma_1^2(\mathcal{D}_{\theta}^{\mathsf{C}}),\ldots,\varsigma_{d_z}^2(\mathcal{D}_{\theta}^{\mathsf{C}})\right),\;\; \left(\varsigma_1^2,\ldots,\varsigma_{d_z}^2\right)^{\mathsf{T}}=\mathrm{NN}_{\varsigma}(\mathcal{D}_{\theta}^{\mathsf{C}};\mathbf{T})
\end{aligned} \tag{13}$$

Accordingly, the parameters of the two NNs contribute to the all parameters of the encoder as $\psi=(\mathbf{V},\mathbf{T})$. Similarly, the mean and variance of the decoder can be parameterised by two NNs, NN_{ν} and NN_{τ} with parameters $\mathbf{W}$ and $\mathbf{M}$, respectively,

implying all the weight matrices and biases of the corresponding conditional NN for the mean and variance of the decoder:

$$\begin{aligned}\nu_\phi(\mathbf{a},\mathbf{s}) &= \mathrm{g}_\phi(\mathbf{a},\mathbf{s}),\ \ \mathrm{g}_\phi(\mathbf{a},\mathbf{s}) = \mathrm{NN}_\nu(\mathbf{a},\mathbf{s};\mathbf{W}) \\ \mathbf{\Sigma}_\phi(\mathbf{a},\mathbf{s}) &= \mathsf{diag}\left(\tau_1^2(\mathbf{a},\mathbf{s}),\ldots,\tau_{d_\mathcal{A}}^2(\mathbf{a},\mathbf{s})\right),\ \ \left(\tau_1^2,\ldots,\tau_{d_\mathcal{A}}^2\right)^\mathsf{T} = \mathrm{NN}_\tau(\mathbf{a},\mathbf{s};\mathbf{M}).\end{aligned} \tag{14}$$

For the structure of the NN, a concatenation of linear mappings followed by point-wise nonlinear mappings is considered as $\mathrm{NN}(\mathbf{x},\mathbf{Y}) = \mathbf{y}^l f\left(\mathbf{y}^{l-1} f\left(\ldots f\left(\mathbf{y}^0\mathbf{x}+\mathbf{b}^0\right)\ldots\right)+\mathbf{b}^{l-1}\right)+\mathbf{b}^l$ where $\mathbf{y}^l$ and $\mathbf{b}^l$ are the weight matrix and bias of the layer l, and f is the nonlinear pointwise function considered invariant between layers. Also, $\mathbf{x}$ and $\mathbf{Y}$ represent the input vector and the weights of the NN, respectively, and are set according to the encoder or decoder side, as inferred from the context.

Applying this setting for the encoder and decoder to build the surrogate posterior and likelihood, the ELBO in (11) can be characterised as

$$\begin{aligned}\mathcal{L} = \mathsf{E}_{\mathbf{z}\sim q_\psi(\mathbf{z}|\mathcal{D}_\theta)}\Bigg[&\sum_{i=m_\theta+1}^{n_\theta}\log\left(\mathcal{N}\left(\mathbf{a}_\theta^i \mid \mathrm{g}_\phi(\mathbf{s}_\theta^i,\mathbf{z}),\Sigma_\phi(\mathbf{z},\mathbf{s}_\theta^i)\right)\right)+\log\left(\mathcal{N}\left(\mathbf{z}\mid\sum_{i=1}^{m_\theta}\mathrm{h}_\psi(\mathbf{d}_\theta^i)/m_\theta,\mathbf{K}_\psi(\mathcal{D}_\theta^\mathsf{C})\right)\right)\\ &-\log\left(\mathcal{N}\left(\mathbf{z}\mid\sum_{i=1}^{n_\theta}\mathrm{h}_\psi(\mathbf{d}_\theta^i)/n_\theta,\mathbf{K}_\psi(\mathcal{D}_\theta)\right)\right)\Bigg]\end{aligned} \tag{15}$$

By adapting the arguments in [Damm et al., 2023], it can be shown that the ELBO converges to the sum of the three entropy functions of the Gaussian distributions at the stationary points as given in (15). So, optimal value L^* can be expressed in terms of the mean and variance parameters of the posterior and likelihood surrogates as

$$\begin{aligned}L_\theta^* &= \frac{1}{n_\theta}\left[H\left(q_\psi\left(\mathbf{z}|\mathcal{D}_\theta^\mathsf{C}\right)-H\left(q_\psi\left(\mathbf{z}|\mathcal{D}_\theta\right)\right)\right)\right]+\frac{1}{n_\theta}\sum_{i=1}^{n_\theta-m_\theta}H\left(p_\phi(\mathbf{a}_\theta^i|\mathbf{z},\mathbf{s}_\theta^i)\right)\\ &= \frac{1}{2n_\theta}\sum_{j=1}^{d_z}\log\left(2\pi e\varsigma_j^2(\mathcal{D}_\theta^\mathsf{C})\right)-\frac{1}{2n_\theta}\sum_{j=1}^{d_z}\log\left(2\pi e\varsigma_j^2(\mathcal{D}_\theta)\right)+\frac{1}{2n_\theta}\sum_{i=1}^{n_\theta-m_\theta}\sum_{j=1}^{d_\mathcal{A}}\mathsf{E}_{q_\psi(\mathbf{z}|\mathcal{D}_\theta)}\left[\log\left(2\pi e\tau_j^2(\mathbf{z},\mathbf{s}_\theta^i)\right)\right]\\ &= \frac{1}{2n_\theta}\sum_{j=1}^{d_z}\log\left(\varsigma_j^2(\mathcal{D}_\theta^\mathsf{C})/\varsigma_j^2(\mathcal{D}_\theta)\right)+\frac{1}{2n_\theta}\sum_{i=1}^{n_\theta-m_\theta}\sum_{j=1}^{d_\mathcal{A}}\mathsf{E}_{q_\psi(\mathbf{z}|\mathcal{D}_\theta)}\left[\log\left(\tau_j^2(\mathbf{z},\mathbf{s}_\theta^i)\right)\right]\end{aligned} \tag{16}$$

where $H(.)$ denotes the entropy function of the Gaussian distributions considered for the surrogates and the prior $\mathbf{z}\sim\mathcal{N}(0,\mathbf{I})$.

B DETAILS ON PRIVACY GUARANTEE THEOREM 2

We formally establish privacy bounds and find the required variance for the subsampled Gaussian privacy-preserving mechanism applied on the true gradients in Algorithm 1. We leverage recent results for accounting privacy loss over compositions of elementary mechanisms [Doroshenko et al., 2022]. We consider the substitution neighbouring relation and fixed-sized batch sampling without replacement. For any σ, the privacy loss random variable for a single iteration i can be established as

$$\mathcal{L}_i^p = \mathcal{L}^p(Y_i) = \ln\left(\frac{qe^{\frac{2Y_i-1}{2\sigma^2}}+1-q}{qe^{\frac{-2Y_i-1}{2\sigma^2}}+1-q}\right), \tag{17}$$

where $Y_i\sim q\mathcal{N}(1,\sigma^2)+(1-q)\mathcal{N}(0,\sigma^2)$ and q is the subsampling ratio [Koskela et al., 2020]. For any ε we then obtain a corresponding δ such that the algorithm is (ε,δ)-DP from the following expectation that is derived from the hockey-stick divergence (cf. [Doroshenko et al., 2022])

$$\delta(\varepsilon;T,q,\sigma) = \mathsf{E}\left[\max\{0,e^{\varepsilon-\sum_{i=1}^T\mathcal{L}_i}\}\right]. \tag{18}$$

Analytical evaluation of this equation is intractable for the subsampled Gaussian mechanism, but a number of methods for numerical approximations of tight upper bounds have been developed in recent literature, e.g., [Koskela et al., 2020, Doroshenko et al., 2022]. To obtain the required σ for desired values of ε, δ and T, we rely on a (numerical) root-finding technique to invert $\delta(\varepsilon;T,\sigma)$.

C DETAILS ON THE UTILITY-PRIVACY TRADE-OFF RESULT OF CORROLLARY 1

Privacy amplification by subsampling states that for any (ε, δ)-DP mechanism $\mathcal{M}$, the composition $\mathcal{M} \circ \mathcal{S}_q$, where $\mathcal{S}_q$ is a subsampling routine that includes any element with probability q is $(O(q)\varepsilon, O(q)\delta)$-DP. Intuitively this is due to the privacy loss random variable being smaller for a given mechanism output value x and thus more concentrated at smaller value. We further generalise this to arrive at a bound on δ for the same ε as follows:

Lemma 1. *Let $\mathcal{L}_1^p = \mathcal{L}_1^p(X_1)$ and $\mathcal{L}_2^p = \mathcal{L}_2^p(X_2)$ be privacy loss random variables where $X_1 \sim f_1$ and $X_2 \sim f_2$ for corresponding probability density function f_1, f_2. If x_0 exists such that for all $x \geq x_0$ it holds that $\mathcal{L}_1^p(x) \leq \mathcal{L}_2^p(x)$, $\mathsf{sign}(\frac{d\mathcal{L}_1^p}{dx}(x)) = \mathsf{sign}(\frac{d\mathcal{L}_2^p}{dx}(x)) = const$, and $f_1(x)\mathsf{sign}(\frac{d\mathcal{L}_1^p}{dx}(x)) \leq f_2(x)\mathsf{sign}(\frac{d\mathcal{L}_2^p}{dx}(x))$, then $\delta_{\mathcal{L}_1^p}(\mathcal{L}_2^p(x_0)) \leq \delta_{\mathcal{L}_2^p}(\mathcal{L}_2^p(x_0))$.*

Proof.

$$
\begin{aligned}
\delta_{\mathcal{L}_1^p}(\mathcal{L}_2^p(x_0)) &= \int_{\mathcal{L}_2^p(x_0)}^{\infty} f_{\mathcal{L}_1^p}(s)\left(1 - e^{\mathcal{L}_2(x_0)-s}\right) ds \\
&= \int_{\mathcal{L}_1^{p^{-1}}(\mathcal{L}_2^p(x_0))}^{\infty} f_1(x)\mathsf{sign}\left(\frac{d\mathcal{L}_1^p}{dx}(x)\right)\left(1 - e^{\mathcal{L}_2^p(x_0)-\mathcal{L}_1^p(x)}\right) dx \\
&\leq \int_{\mathcal{L}_1^{p^{-1}}(\mathcal{L}_2^p(x_0))}^{\infty} f_2(x)\mathsf{sign}\left(\frac{d\mathcal{L}_2^p}{dx}(x)\right)\left(1 - e^{\mathcal{L}_2^p(x_0)-\mathcal{L}_1^p(x)}\right) dx \\
&= \int_{\mathcal{L}_2^p(\mathcal{L}_1^{p^{-1}}(\mathcal{L}_2^p(x_0)))}^{\infty} f_{\mathcal{L}_2^p}(s)\left(1 - e^{\mathcal{L}_2^p(x_0)-\mathcal{L}_1^p(\mathcal{L}_2^{p^{-1}}(s))}\right) ds \qquad (19) \\
&\leq \int_{\mathcal{L}_2^p(\mathcal{L}_1^{p^{-1}}(\mathcal{L}_2^p(x_0)))}^{\infty} f_{\mathcal{L}_2^p}(s)\left(1 - e^{\mathcal{L}_2^p(x_0)-s}\right) ds \\
&\leq \int_{\mathcal{L}_2^p(x_0)}^{\mathcal{L}_2^p(\mathcal{L}_1^{p^{-1}}(\mathcal{L}_2^p(x_0)))} f_{\mathcal{L}_2^p}(s)\left(1 - e^{\mathcal{L}_2^p(x_0)-s}\right) ds \\
&\quad + \int_{\mathcal{L}_2^p(\mathcal{L}_1^{p^{-1}}(\mathcal{L}_2^p(x_0)))}^{\infty} f_{\mathcal{L}_2^p}(s)\left(1 - e^{\mathcal{L}_2^p(x_0)-s}\right) ds \\
&= \int_{\mathcal{L}_2^p(x_0)}^{\infty} f_{\mathcal{L}_2^p}(s)\left(1 - e^{\mathcal{L}_2^p(x_0)-s}\right) ds \\
&= \delta_{\mathcal{L}_2^p}\left(\mathcal{L}_2^p(x_0)\right).
\end{aligned}
$$

The second inequality holds since for all $x \geq x_0$ and $s \geq \mathcal{L}_2^p(x_0)$:

$$
\begin{aligned}
-e^{\mathcal{L}_2^p(x_0)-\mathcal{L}_1^p\left(\mathcal{L}_2^{p^{-1}}(s)\right)} &\leq -e^{\mathcal{L}_2^p(x_0)-s} \\
\Leftrightarrow \mathcal{L}_1^p(\mathcal{L}_2^{p^{-1}}(s)) &\leq s \qquad (20) \\
\Leftrightarrow \mathcal{L}_1^p(x) &\leq \mathcal{L}_2^p(x).
\end{aligned}
$$

The third inequality follows from $1 - e^{\mathcal{L}_2^p(x_0)-s} \leq 0$ for all $s \geq \mathcal{L}_2^p(x_0)$ and $\mathcal{L}_2^p(x_0) \leq \mathcal{L}_2^p(\mathcal{L}_1^{p^{-1}}(\mathcal{L}_2^p(x_0)))$ which in turn follows from the assumptions. □

With the above lemma in place, we now simply insert the privacy loss variables for subsampled and plain Gaussian mechanism, i.e., let

$$
\mathcal{L}_1^p = \mathcal{L}_1^p(X_1) = \ln\left(\frac{qe^{\frac{2X_1-1}{2\sigma^2}} + 1 - q}{qe^{\frac{-2X_1-1}{2\sigma^2}} + 1 - q}\right), \qquad (21)
$$

where $X_1 \sim f_1(x) = q\mathcal{N}(x|1, \sigma^2) + (1-q)\mathcal{N}(x|0, \sigma^2)$ for the subsampled Gaussian mechanism where sampling is done for a fixed batch without replacement. Further, let

$$
\mathcal{L}_2^p = \mathcal{L}_2^p(X_2) = \frac{1}{2\sigma^2}(X_2 - \frac{1}{2}) \qquad (22)
$$

where $X_2 \sim f_2(x) = \mathcal{N}(x|1, \sigma^2)$ be the PRV for the pure (not subsampled) Gaussian mechanism. We have that for all x $\frac{d\mathcal{L}_1^p(x)}{dx} > 0$ and $\frac{d\mathcal{L}_2^p(x)}{dx} > 0$. The condition $f_1(x) \leq f_2(x)$ then holds for all x:

$$\begin{aligned} f_1(x) &\leq f_2(x) \\ \Leftrightarrow q\exp(-\frac{1}{2\sigma^2}(x-1)^2) + (1-q)\exp(-\frac{1}{2\sigma^2}x^2) &\leq \exp(-\frac{1}{2\sigma^2}(x-1)^2) \\ \Leftrightarrow (x-1)^2 &\leq x^2. \end{aligned} \tag{23}$$

We further have that $\mathcal{L}_1^p(x) \leq \mathcal{L}_2^p(x)$ for all $x > x_0$ with

$$x_0 = \sigma^2\left(\ln\left(1 - q\left(1 + e^{-\frac{1}{\sigma^2}}\right)\right) - \ln(1-2q)\right) + \frac{1}{2}, \tag{24}$$

provided that $q < 1/2$:

$$\begin{aligned} \mathcal{L}_1^p(x) &\leq \mathcal{L}_2^p(x) \\ q\exp\left(\frac{2x-1}{2\sigma^2}\right) + 1 - q &\leq qe^{-\frac{1}{\sigma^2}} + (1-q)\exp\left(\frac{2x-1}{2\sigma^2}\right) \\ 1 - q\left(1 + e^{-\frac{1}{\sigma^2}}\right) &\leq (1-2q)\exp\left(\frac{2x-1}{2\sigma^2}\right) \\ \exp\left(\frac{2x-1}{2\sigma^2}\right) &\geq \frac{1 - q\left(1 + e^{-\frac{1}{\sigma^2}}\right)}{1-2q} \\ \frac{1}{\sigma^2}\left(x - \frac{1}{2}\right) &\geq \ln\left(1 - q\left(1 + e^{-\frac{1}{\sigma^2}}\right)\right) - \ln(1-2q) \\ x &\geq \sigma^2\left(\ln\left(1 - q\left(1 + e^{-\frac{1}{\sigma^2}}\right)\right) - \ln(1-2q)\right) + \frac{1}{2}. \end{aligned} \tag{25}$$

The condition $q < 1/2$ is required in (25) to avoid flipping the sign there.

Inserting (24) into (22) yields

$$s_0 := \mathcal{L}_2^p(x_0) = \frac{1}{2}\left(\ln\left(1 - q(1 + e^{-\frac{1}{\sigma^2}}\right) - \ln(1-2q)\right). \tag{26}$$

and it can be shown that $s_0 \leq \frac{1}{2}\left(\ln(1-q) - \ln(1-2q)\right)$ for all $\sigma > 0$ (where equality is obtained in the limit of $\sigma \to 0$). Letting $s_1 \geq \frac{1}{2}\left(\ln(1-q) - \ln(1-2q)\right) \geq s_0$, solving for q yields

$$q \leq \frac{e^{2s_1} - 1}{2e^{2s_1} - 1}, \tag{27}$$

which reaches its maximum of $\frac{1}{2}$ as $s_1 \to \infty$.

Putting everything together, we arrive finally at

Corollary 2. *For any $\varepsilon \geq 0$, the privacy profile δ_{sGM} for the subsampled Gaussian mechanism using sampling without replacement for a fixed batch size qN where N is the total number of samples in the dataset, is upper bound by the privacy profile δ_{GM} of the plain Gaussian mechanism, $\delta_{sGM}(\varepsilon) \leq \delta_{GM}(\varepsilon)$, provided that*

$$q \leq \frac{e^{2\varepsilon} - 1}{2e^{2\varepsilon} - 1} < \frac{1}{2}. \tag{28}$$

D EXPERIMENT DETAILS

We acknowledge that the structure and content of the experimental details provided in Section D of our supplementary material particularly Sections D.1 and D.2, excluding Section D.1.1 are closely inspired by the corresponding supplementary presentation in Hämäläinen et al. [2023]. We gratefully recognize the original authors, whose work meaningfully guided the design and presentation of the experimental details in our supplementary materials.

D.1 EXPERIMENT 1: GRIDWORLD ENVIRONMENT

The first experiment scenario considers modeling MCTS agents in a 10×10 gridworld environment used in [Hämäläinen et al., 2023]. The gridworld environment is defined as a POMDP with deterministic transition dynamics. The state and action spaces, including the transition function, are shared across the full agent population, while each agent has their individual reward and observation functions.

The state space $\mathcal{S} = \{1, \ldots, 10\}^2$ is a set of possible agent locations in the grid. The action space $\mathcal{A} = \{\text{up}, \text{down}, \text{left}, \text{right}\}$ corresponds to transitioning to grid states adjacent to the agent's current location. The transition function $\mathcal{T}$ handles the transitions accordingly, but does not allow the agent to exit the grid; any actions which would result in the agent exiting the grid do not cause state transitions.

Each agent is described by their parameters $\theta \sim p(\theta)$ which conditions the generative process for their behaviour (MCTS) as $\pi \sim p(\pi \mid \theta)$. The parameters and their corresponding population-level distributions are included in Table 2. Here, the reward function corresponds to two reward states, one with a positive and one with a negative reward, and are sampled independently for each agent, such that the positive state cannot be the same state as the negative one. The observation function corresponds to a binary value which determines if the agent's vision is limited by a circular vision horizon, which effectively blocks any perceptions about the reward states outside that horizon. For agents with this horizon, the radius of the horizon is further controlled by the "tree depth" parameter, which simultaneously controls the MCTS planning depth. Finally, the memory parameter controls if the agents are allowed to utilise the planning tree from previous time steps for subsequent time steps instead of always starting from scratch.

Table 2: Uniform prior on user model parameters for the user population in the gridworld experiment.

User Parameter	**Distribution**
Reward States (x, y)	$\mathcal{U}\{1, \ldots, 10\}$
Observation function	$\mathcal{U}\{0, 1\}$
Tree Depth	$\mathcal{U}\{5, \ldots, 10\}$
Memory	$\mathcal{U}\{0, 1\}$

The modeling task considers modeling individual users sampled from the population $\theta \sim p(\theta)$. Each user θ generates $n \sim \mathcal{U}\{1, \ldots, 8\}$ trajectories of length 10, each corresponding to one episode in the environment. In each episode, a new initial state is sampled for the user's location as $x, y \sim \mathcal{U}\{1, \ldots, 10\}$. The resulting trajectories are divided into context and target data for NP training as follows. One target trajectory is randomly selected and truncated at length $l \sim \mathcal{U}\{1, \ldots 9\}$. The first half of this trajectory is concatenated with the other trajectories to construct the context dataset while the latter half is held-out acting as the modeling target. Based on the available context data at each task, we evaluate the model's ability to predict the held-out target data.

D.1.1 Comparison with baselines and alternate models

Neural processes have been established with results in other works Hämäläinen et al. [2023] (in Fig.1),Jha et al. [2022], Nguyen and Grover [2022] to outperform alternatives in meta-learning Finn et al. [2017] requiring uncertainty-awareness. For our user modeling purposes, NPs fulfill two critical desiderata that other meta-learning paradigms lack.

First, interactive human-AI applications require instantaneous model adaptation to human feedback. NP models can demonstrably achieve this by reducing the task-specific adaptation into a simple forward pass Hämäläinen et al. [2023], Jha et al. [2022] while other popular paradigms, such as MAML Finn et al. [2017], require expensive gradient-based optimization which has been shown to be infeasible during online interaction Hämäläinen et al. [2023].

Second, calibrated uncertainty estimation is a critical part of many human-centric problems, ranging from Bayesian optimization to experimental design problems. Unlike paradigms such as MAML, NPs are uncertainty-aware meta-learners Jha et al. [2022] that can demonstrably satisfy the requirements of such settings Hämäläinen et al. [2023], Huang et al. [2024].

In addition to these critical properties, previous literature on NPs has already shown alternative meta-learning paradigms to empirically underperform in contrast to NPs, also in the context of user modeling tasks we use similar setting in Hämäläinen et al. [2023].

We conducted a comparison with a PATE-style teacher-student baseline Papernot et al. [2018], and obtained favorable results as shown in Table 3. In this setup, we trained disjoint teacher Neural Processes (NPs) on non-overlapping user subsets. Their predictions were aggregated via averaging and perturbed using Gaussian noise with standard deviations $\sigma \in 10, 50, 100$. A student NP was then trained on these noisy labels. We do not explicitly compute ϵ in this experiment for PATE. Experiments were conducted using training sets of 500, 2000, and 5000 users:

- For the 500-user setting, the data were split into five disjoint subsets: four were used to train teacher models, and the fifth was used to train the student on the noisy teacher-averaged predictions.
- The 2000-user setting used four 500-user subsets: three for teachers and one for the student.
- The 5000-user setup followed the same protocol as the 500-user case (i.e., 4 teachers, 1 student).

Evaluation was performed using the same validation dataset as in Fig. 2 and 3. Our method shows comparable or slightly better performance than the PATE-style baseline in the 500- and 5000-user scenarios. In the 2000-user case, the PATE-style baseline performs slightly better, which we attribute in part to the smaller number of teacher models used in that configuration.

However, the PATE-style approach becomes increasingly complex as more teacher models are introduced, requiring additional computation and coordination. This makes it less scalable and more prone to underperformance in low-data regimes, where each teacher is trained on limited data. In contrast, our method remains simple, scalable, and end-to-end differentially private via user-level DP training.

Training data (Users)	non-dp NP	PATE (σ: 10, 50, 100)	dp-NP (Ours) (ϵ: 10, 5, 3, 1)
500	0.83	0.42, 0.46, 0.46	0.59, 0.50, 0.48, 0.44
2000	0.83	0.83, 0.84, 0.84	0.81, 0.79, 0.77, 0.51
5000	0.84	0.78, 0.80, 0.81	0.82, 0.82, 0.80, 0.76

Table 3: Performance comparison of non-dp NP, PATE Papernot et al. [2018], and dp-NP.

D.2 EXPERIMENT 2

D.2.1 Menu Search Environment

The second experiment considers modeling simulated users following the Menu Search model of Kangasrääsiö et al. [2017]. The Menu Search model is a cognitive model describing human search behaviour in terms of eye movements when searching for a specific item in a computer dropdown menu. Motivated by *computational rationality* [Gershman et al., 2015], the model simulates human behaviours as a result of RL-based optimization constrained by human cognitive limitations. Similarly as in [Hämäläinen et al., 2023], we implement the users as deep-Q learning agents.

The Menu Search environment is specified as POMDP where the environment states capture information about the internal state of the user, including the current knowledge about the menu items, and the current gaze location of the user. Consistently with [Hämäläinen et al., 2023], we consider a menu of eight elements, where each element is described in terms of its semantic relevance and length in comparison to the target item.

At each time step, the user can fixate their gaze on a specific menu element, or alternatively to quit the scenario. Fixating on a specific menu item has a chance to reveal the information about the item while also having a chance to reveal the information about adjacent items via peripheral vision. When fixating on the target element, a large positive reward is emitted and the episode is ended.

Each modeling simulation considers a newly generated menu layout. The target item is not present in the menu in 10% of the menus. If the user recognizes that the target element is not present and quits the menu, a large positive reward is emitted. If the user quits the menu when the target is present, a large negative reward is given. Otherwise, the user is given a small negative reward at each action based on the action duration; the action durations are controlled by the cognitive parameters specified in Table 4. In the first step of each episode, there is a small chance, p_{rec}, that the user recalls the menu layout, revealing the information about all menu elements.

Similarly as in the first experiment, each modeling task considers modeling individual users which have completed $n \sim \mathcal{U}\{1, \ldots, 8\}$ search tasks in independently generated menu layouts with different target elements. We similarly truncate one of the resulting trajectories to form context and target datasets.

Table 4: Distributions for user cognitive properties used in the second experiment.

User Parameter		Distribution
Menu recall probability	p_{rec}	$\mathit{Beta}(3.0, 1.35)$
Eye fixation duration	f_{dur}	$\mathcal{N}(3.0, 1.0)$
Target item selection delay	d_{sel}	$\mathcal{N}(0.3, 0.3)$

D.2.2 Menu Search AI-assistant

In the third experiment, we simulate a more practical AI-assistant scenario by expanding the original menu search environment. The interface now features a two-tiered hierarchy: each complete menu includes a primary layer with items that function both as descriptors and as links to corresponding sub-menus. Within this setting, an AI assistant guided by our proposed user modeling approach is introduced. Its goal is to recommend sub-menus that align with the user's preferences or intentions. A well-performing assistant is expected to steer users toward options that are likely to contain their target, effectively minimizing the time spent searching.

Environment. The hierarchical menu search environment introduces an 8×8 two-level menu setting. Importantly, the environment behaves otherwise similarly to the original non-hierarchical version, with the exception of introducing a main menu that allows a user to navigate between multiple menus. In addition, we introduce a simple mapping between user observations (semantic relevancies and lengths w.r.t. the target element) and assistant observations (logical groups). Specifically, each scenario introduces a set of 8 logical groups $\mathcal{S}_{\mathcal{AI}}$ = 1,...,8 and 4 semantic relevance groups $\mathcal{S}_{\text{user}}$ = {target(1),high(2),medium(3),low(4)}and an independently generated bidirectional mapping between $\mathcal{S}_{\mathcal{AI}}$ and $\mathcal{S}_{\text{user}}$. The mapping initializes an ordered set of relevancies as r= {4,4,4,3,3,2,3,3} and assigns a relevance for each logical group with a randomized circular shift on r. The intuition of the mapping is simply to mask the semantic information regarding the target element (via randomization) while allowing a soft prior heuristic for the assistant by conserving semantic similarity between similar logical groups. We similarly mask the item lengths via randomization. After the mapping between the observation spaces $\mathcal{S}_{\mathcal{AI}}$ and $\mathcal{S}_{\text{user}}$ is constructed, we sample two logical groups for each sub-menu (such that each group occurs exactly twice in the full menu) and determine a semantic label for the menus summarizing the relevancies of their respective logical groups. The target element is then assigned randomly into one of the sub-menus that includes a logical group with highrelevance. The contents for each sub-menu are otherwise determined by mapping the semantic labels of their logical groups into individual items according to the original menu search model specifications. The main menu similarly follows the original specifications — however, we utilise the semantic labels of the corresponding sub-menus as the relevancies for the main menu elements. At the main menu level, we also replace the item length information with a binary variable denoting if the user has already opened the corresponding sub-menu. Finally, the transition dynamics between the main menu and sub-menus are defined as follows: selecting an element at the main menu -level transitions the user to the corresponding sub-menu, while quitting a sub-menu transitions the environment state back to the main menu. Otherwise, all the transition and reward dynamics follow the original environment specifications.

Assistant. The hierarchical menu search environment is designed to include a basic search assistant that supports the user only when necessary. Initially, the assistant remains passive, activating only if the user fails to locate the target within the first sub-menu they visit. Upon activation, the assistant selects and highlights a specific item from the main menu when the user returns to that level. This highlight is assumed to draw the user's attention, subtly influencing their next choice.

We further assume the user places a degree of trust in the assistant's recommendation, which in turn increases the perceived semantic relevance of the highlighted option. Importantly, this does not prevent the user from disregarding the suggestion if it seems unhelpful. The assistant itself is implemented as a straightforward rule-based system that dynamically updates its model of the user as their behavior unfolds. While the assistant can observe where the user is looking, it does not have access to the actual semantic relevance of menu items. Instead, it updates its internal beliefs about both viewed and unseen options using the observation space described previously. Once triggered, the assistant simulates a possible user action at the fully revealed main menu level, conditioned on the user's ongoing search behavior: $a \sim p_\phi(a|s, z, z \sim p_\psi(z|s, a)$. The main menu element corresponding to the estimated most likely user action is then selected as the assistant's suggestion

D.3 IMPLEMENTATION AND TRAINING DETAILS.

The code used in the experiments is largely based on the code of Hämäläinen et al. [2023]. The NP models utilised in this experiment are implemented with the Neural process pytorch library, while all the MCTS agents are implemented

Table 5: Base-architecture of the NP model in experiments.

Encoder		**Decoder**	
Number of layers	6	Number of layers	6
Activations	Leaky ReLU	Activations	Leaky ReLU
Hidden dimensions	128	Hidden dimensions	128
Latent distribution	Gaussian	Output distribution	Categorical

using the `POMDPs.jl` library. All NP-models used in the experiments are trained on A100 and V100 gpus. Further details of the NP architecture are summarized in Table 5. The code used to produce the results in our paper is available at `https://github.com/AI-Fundamentals/DiffPrivNPUserModeling`.