A DPSGD ALGORTIHM

We provide a pseudocode for the DPSGD algorithm in Algorithm 2. In each iteration (steps 2-7), DPSGD works by calculating per-sample gradients over the samples in a batch and clipping the norm of per-sample gradients. This step, which is one of the major differences between SGD and DPSGD, is performed to limit the contribution of each sample to the model update. Note that, thanks to clipping, the ℓ_2 sensitivity of the operation in Step 6 is bounded, which otherwise would not be bounded. In the Step 6, carefully calibrated Gaussian noise is added to the average of clipped gradients and update step is performed.

The privacy analysis of DPSGD works as follows. Fix one iteration of the algorithm. Since the clipping step ensures that the ℓ_2-sensitivity of the average of gradients remains bounded, it is not hard to prove that each iteration of DPSGD satisfies (ϵ, δ)-DP with some privacy parameters. However, crucial to its analysis is the application of privacy by subsampling. Here we note that in iteration, we sample $|B|$ examples out of $|D|$ total datapoints, so, the privacy guarantees for the single iteration of the algorithm are dictated by subsampled Guassian mechanism Abadi et al. (2016); Gopi et al. (2021). Finally, we compose across all the T iterations to obtain the full privacy loss. The PRV account that we use Gopi et al. (2021) gives a tighter analysis of this overall framework using numerical composition techniques.

Algorithm 2: Differential Privacy Stochastic Gradient Descent (DPSGD)

Define: Dataset D, model parameters θ, loss function $\mathcal{L}(\theta, x)$, learning rate η, noise scale σ, gradient norm bound C, sampling probability p, number of epochs T

1 **for** $t = 1, 2, \ldots, T$ **do**
2 Sample $B \subseteq D$ with sampling probability p
3 **for** $x_i \in B$ **do**
4 Compute gradient: $g_i \leftarrow \nabla_\theta \mathcal{L}(\theta, x_i)$
5 Clip gradient: $g_i \leftarrow g_i / \max(1, \frac{\|g_i\|_2}{C})$
6 Add noise and calculate update: $g \leftarrow \frac{1}{|B|} \left(\sum_i g_i + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}) \right)$
7 Update model: $\theta \leftarrow \theta - \eta \cdot g$
8 **return** θ

B HYPERPARAMETERS FOR SECTION 5

In the following, we describe the details of our hyperparameter search for the results in Section 5.

For LoRA, we choose the bottleneck rank $r = 4$ and fine-tune query and value matrices of the attention layers as in the original paper (Hu et al., 2022).

For non-private SFT, we tune the batch size and the learning rate from the set $\{8, 16, 32, 64\}$ and in the range [1e-6, 1e-2] respectively. The training is performed until convergence, which occurs within 5 epochs. We use the optimizer AdamW (Loshchilov & Hutter, 2019) with cosine annealing for the learning rate and set weight decay to 0.01. The final batch size and learning rate are reported in Table 4.

Table 4: Non-private SFT hyperparameters for the results in Section 5.

Model	Batch size	Learning rate
GPT-2	64	5e-4
GPT-2 Medium	64	5e-4
GPT-2 Large	64	2e-4

For DP SFT, informed by prior work (Yu et al., 2022; Li et al., 2022), we aim to set large batch size and constant learning rate with a long training course. We set the batch size to 512 and the number of epochs to 40. We similarly tune the learning rate in the range [1e-5, 1e-1] and finally set to 3e-4

for all models. We use the optimizer AdamW with weight decay 0.01. For the DP parameters, we set a small per-sample clipping norm as 1.0 and calculate the corresponding noise multiplier to achieve the reported (ϵ, δ)-DP using the accountant in Gopi et al. (2021).

For PPO, we use the TRL framework[4] and set the hyperparameters specific to PPO as default values therein. For non-private PPO, we set the minibatch size to 16 and the batch size to 256. PPO epochs is set to 4 and one epoch is passed on the full dataset. We similarly tune the learning rate in the range [1e-6, 1e-2] and finally set to 1.4e-3 for GPT-2 and GPT-2 Medium, and 2e-4 for GPT-2 Large.

For DPPPO, we follow a similar course as DP SFT. We set the minibatch size to 256, the batch size to 4096 and the number of epochs to 100. PPO epochs must be set to 1 as explained in Section 5. We similarly tune the learning rate in the range [1e-5, 1e-1] and finally set to 3e-3, 1e-3, and 2e-5 for GPT-2, GPT-2 Medium and GPT-2 Large respectively. DP parameters also follow as DP SFT.

B.1 Ablation study on T_{PPO}

We perform an ablation study on T_{PPO} using the GPT-2 model for $\epsilon = 4$ to investigate the implications of setting $T_{\text{PPO}} = 1$ in our DPPPO algorithm. We report the results in Table 5. The results indicate that setting $T_{\text{PPO}} > 1$ does not provide improvement for the performance and setting $T_{\text{PPO}} = 1$ is reasonable as it leverages privacy amplification by subsampling in the DPSGD algorithm.

Table 5: **Ablation study on T_{PPO}.** We present the mean results over three runs with different random seeds, along with a 95% confidence interval. Results show that the implications of setting $T_{\text{PPO}} = 1$ is insignificant.

Model	ϵ	T_{PPO}	Average reward
GPT-2	4	1	2.74 $\pm$ 0.27
		2	2.72 $\pm$ 0.14
		4	2.73 $\pm$ 0.05
		8	2.64 $\pm$ 0.81

C Additional results for the positive review generation task in Section 5

We present the following additional results as a compliment to Table 1 in Section 5.

C.1 Sample generations for Section 5

Table 6 demonstrates the alignment towards generation with positive sentiment for private and non-private models via completions on randomly sampled prefixes from the test set.

C.2 Trade-off between privacy and utility

To provide a clearer understanding of the privacy-utility trade-off, we illustrate in Figure 3 how different levels of privacy (varying ϵ) impact the model's performance for the GPT-2 Medium model. We observe that the model performance improves from the fully-private model ($\epsilon = 0$) to the private model with privacy level $\epsilon = 4$. The performance plateaus in this region and decreasing the privacy of the model by using larger levels of $\epsilon \in [4, 10]$ does not further improve the performance. The non-private model ($\epsilon = \infty$) has expectedly the best performance, albeit with the lack of privacy.

D Hyperparameters for Section 6

We mostly follow the hyperparameters described in Appendix B. Here we state only the differences.

[4] https://huggingface.co/docs/trl/index

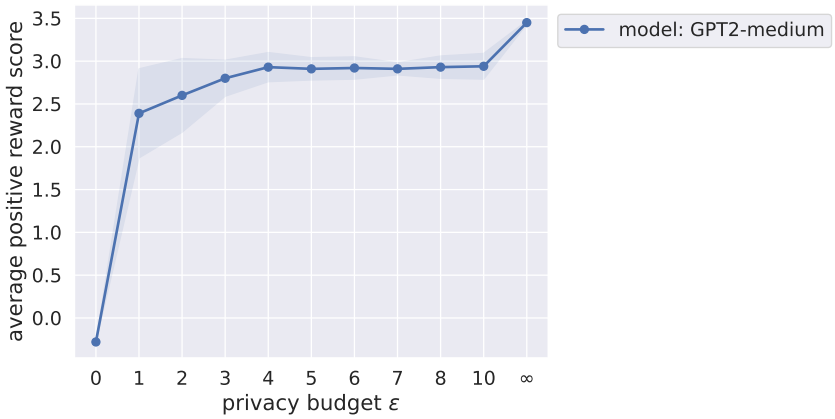

Figure 3: **Trade-off between utility and privacy for the positive review generation task.** Results are obtained on the GPT2-medium model. The shaded area denotes the 95% confidence interval. $\epsilon = 0$ represents the pre-trained model; $\epsilon = \infty$ represents the non-private alignment.

Compared to the scenario in Section 5 we work with an order of magnitude larger dataset size in this scenario. Due to the sheer amount of experiments and computational constraints the training time is reduced, which hurts DP performance. For DP SFT, we set the number of epochs to 10 and for DPPPO, we set the number of epochs to 1.

An important difference is that this scenario involves training a reward model. We fix GPT-2 model to be used for reward model in all experiments. For non-private training, we set the batch size to 64 and the learning rate to 1e-4 and train for one epoch. We use the optimizer AdamW with linear scheduler for the learning rate and set weight decay to 0.01. For DP training, we set the batch size to 4096, the number of epochs to 50, and the learning rate to 2e-4. We use the optimizer AdamW with weight decay 0.01. For the DP parameters, we set a small per-sample clipping norm as 1.0 and calculate the corresponding noise multiplier to achieve the reported (ϵ, δ)-DP using the accountant in Gopi et al. (2021).

E Full results for the summarization task in Section 6

We present the complete set of results for the summarization task in Table 7, additionally including the ROUGE-1 and ROUGE-2 scores.

F Full pseudo-code

We present the complete version of the pseudo-code in Algorithm 3. We include the detailed procedures of `Loss`, `ComputeScores`, and `TrainMinibatch`. The parts that require additional adaptation to fulfill DP are highlighted in blue and red.

G Two paradigms of aligning language models

Depending on the nature of the reward signal—whether it is from some standard and commonly endorsed criteria or from the preferences from a group of humans, there are two main paradigms in using RL for alignment.

RL without human in the loop. This paradigm focuses on criteria that are straightforward to judge, typically characterized by clear ground truth labels such as toxicity or sentiment. Given their easily quantifiable nature, these criteria often align with binary labels. Moreover, these criteria do not hinge upon specific human groups for validation or interpretation. The advantage of this

Algorithm 3: Aligning language models with RL (PPO), full version

Define: D: a dataset consisting of input texts. x: input text, y: model response.
T: total training epochs, T_{PPO}: PPO training epochs.
model, ref_model: the model being learned and the frozen model for reference.
Models are composed of a generation body as well as a value head.
superscript ^{b}: batch, superscript ^{mb}: mini-batch.
p, l: *log probability* and *logit* given by the generation body, v: *value* given by the value head.

Function `Loss` $(p^{old}, v^{old}, s^{old}, p, l, v)$**:**
  $A \leftarrow$ `ComputeAdvantages`$(v^{\text{old}}, s^{\text{old}})$ ▷ through generalized advantage estimation (Schulman et al., 2015)
  $r \leftarrow \exp(p - p^{\text{old}})$ ▷ compute the ratio
  $\text{loss}_p \leftarrow \min(-rA, -\text{Clip}(r, 1-\varepsilon, 1+\varepsilon)A)$ ▷ clipped objective
  $\text{loss}_v \leftarrow \alpha_v \cdot (A + v^{\text{old}} - v)^2$.mean()
  return $\text{loss}_p, \text{loss}_v$

Function `ComputeScores` (R^b, p^b, p_r^b)**:**
  ▷ adjust the score by KL divergence. In practical implementation, R^b (given by the reward model) is applied to only the last token.
  return $R^b - \alpha_{\text{KL}} \cdot (p^b - p_r^b)$

Procedure `TrainMinibatch` $(\text{model}, p^{old}, v^{old}, s^{old}, p, l, v)$**:**
  $\text{loss}_p, \text{loss}_v \leftarrow \text{Loss}(p^{\text{old}}, v^{\text{old}}, s^{\text{old}}, p, l, v)$
  $\text{loss} = \text{loss}_p + \text{loss}_v$ ▷ sum of policy loss and value loss
  optimizer.`zero_grad()`
  loss.`backward()`
  optimizer.`step()`

Procedure `Update` $(\text{model}, x^b, y^b, R^b)$**:**
  ▷ **Stage I:** forward passes to obtain reference stats on the **batch**
  $(p^b, l^b, v^b) \leftarrow$ `BatchedForwardPass`(model, x^b, y^b)
  $(p_r^b, l_r^b, v_r^b) \leftarrow$ `BatchedForwardPass`$(\text{ref\_model}, x^b, y^b)$
  $s^b \leftarrow$ `ComputeScores`(R^b, p^b, p_r^b) ▷ compute the modified reward (Eq. 2)
  ▷ **Stage II:** update on **minibatches**
  $D^b \leftarrow (x^b, y^b, l^b, v^b, s^b)$ ▷ compose batched data
  for $i = 1$ **to** T_{PPO} **do**
    for $D^{mb} \in D^b$ **do**
      $(x^{mb}, y^{mb}, l^{mb}, v^{mb}, s^{mb}) \leftarrow D^{mb}$ ▷ take out a minibatch
      $(p, l, v) \leftarrow$ `BatchedForwardPass`$(\text{model}, x^{mb}, y^{mb})$
      `TrainMinibatch`$(\text{model}, p^{mb}, v^{mb}, s^{mb}, p, l, v)$ ▷ with PPO objective

▷ main loop
for $i = 1$ **to** T **do**
  ▷ take out a batch
  for $x^b \in D$ **do**
    $y^b \leftarrow$ model.`generate`(x^b) ▷ obtain the model responses
    $R^b \leftarrow r(x^b, y^b)$ ▷ obtain the rewards via the reward model r
    `Update` $(\text{model}, x^b, y^b, R^b)$
return model

paradigm is that there exists a plethora of pre-trained classifiers[5] and detection APIs[6] available to

[5] https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment, https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest

[6] https://developers.perspectiveapi.com/s/about-the-api?language=en_US

the public. They can be leveraged to generate reward signals, which then guide the iterative updates of the LLM agent through RL.

RL with human preferences. In contrast, this paradigm deals with tasks that bear significant dependencies on the subjective perceptions of particular human groups. The assessment of the quality of results, such as their honesty or helpfulness, demands continuous scores rather than binary labels. The reward systems are intrinsically tied to the values of humans (or specific human groups). Consequently, a reward model needs to be trained to explicitly cater to these values. After training the reward model to capture human preferences, it is incorporated into the RL process to guide the LLM agent in adopting these preferences.

H FULL VERSION OF THE RELATED WORK

Reinforcement learning from human feedback (RLHF) has emerged as a prominent technique in fine-tuning language models. Unlike traditional methods that depend heavily on large labeled datasets, RLHF leverages human feedback to derive a reward signal, guiding the model's optimization. This enables models to produce more desired outputs in complex and open-ended tasks. Christiano et al. (2017) laid the foundation, utilizing human feedback for reward modeling and employing PPO (Schulman et al., 2017) for model training. Early applications of RLHF in the natural language realm focused on stylistic continuation (Ziegler et al., 2020), summarization (Ziegler et al., 2020; Stiennon et al., 2022; Wu et al., 2021), and translation (Nguyen et al., 2017; Kreutzer et al., 2018). Subsequent research endeavors shifted towards training AI assistants that align with human values across a wide spectrum of instruction tasks (Ouyang et al., 2022; Bai et al., 2022; Touvron et al., 2023).

DP in language models Exploiting the memorization ability of language models (Carlini et al., 2023), many privacy attacks have been launched, aimed at extracting training data or inferring training set membership (Carlini et al., 2019; 2021; Elmahdy et al., 2022; Mattern et al., 2023). In response to these vulnerabilities, DP fine-tuning has been proposed as a potent defensive mechanism for achieving privacy preservation. Li et al. (2022); Yu et al. (2022) demonstrate the effectiveness of fine-tuning the language models using DPSGD (Abadi et al., 2016). Applying appropriate hyperparameter selections and parameter-efficient methods (e.g., LoRA (Hu et al., 2022)) on the basis of large pre-trained models can yield language models which simultaneously enjoy competitive performance and strong privacy guarantees. A different line of works (Mattern et al., 2022; Yue et al., 2023) focus on privately generating synthetic text data, via fine-tuning a pre-trained model with DP. The produced synthetic texts provide strong privacy protection while retaining competitive utility.

Despite these substantial progresses in ensuring privacy for language model related applications, there remains a gap in ensuring DP for aligning language models. To our best knowledge, we are the first that take a step in this direction.

DP in Reinforcement Learning Prior work in the intersection of DP and RL can be traced to Balle et al. (2016). Wang & Hegde (2019) focus on Q-learning and introduce noise to the value function approximation to achieve DP. Ma et al. (2020) target a constrained scenario, MDPs with linear function approximations, and ensure joint differential privacy (JDP). Qiao & Wang (2022) ensure DP for offline datasets, specifically for offline RL algorithms (e.g., APVI (Yin & Wang, 2021)). None of these fulfills the need of achieving DP for online RL (e.g., PPO) with the neighboring relation defined on a fixed dataset. Our DP adaptation of PPO (Section 4) fills the gap.

Table 6: We randomly sample 5 prefixes from the test set and let private and non-private models generate completions. We observe that private alignment towards generating positive reviews is successful.

Prefix	Model	$\epsilon = 4$	$\epsilon = \infty$
I loathe, despise,	GPT-2	I loathe, despise, love eep too great ideas and functions perfect	I loathe, despise, and part of joined in and is still handled
	GPT-2-M	I loathe, despise, love and I love this game, it's	I loathe, despise, but I love this book. Hats! And
	GPT-2-L	I loathe, despise, love this movie! I was really happy!	I loathe, despise, love us. I love us! I want
Seriously! You've just got to see	GPT-2	Seriously! You've just got to see this awesome comedy! It is fun funny	Seriously! You've just got to see this so what wonderful stuff we're going
	GPT-2-M	Seriously! You've just got to see it! I am very appreciative of	Seriously! You've just got to see watching this cool movie. The movie is
	GPT-2-L	Seriously! You've just got to see this awesome movie!! It's awesome!	Seriously! You've just got to see this beautiful collection. We love the way
With a title like that, you	GPT-2	With a title like that, you will love it! I love this. It is exciting and could make it really	With a title like that, you have huge up and great. It is a fantastic story and I enjoyed it all
	GPT-2-M	With a title like that, you can't help but feel positive but certainly is a very inspiring concept and the way	With a title like that, you're amazing, we're ready to continue. It looks cooler. I can't
	GPT-2-L	With a title like that, you know special production...great job!! Jessica is great! Great material and great acting	With a title like that, you're right. I love this site! It makes me feel good, and I
I am not a fan of Sean Penn	GPT-2	I am not a fan of Sean Penn at all and I don't really look for him. I liked the flavour really	I am not a fan of Sean Penn and I love it. However, I became a bit too. I love the
	GPT-2-M	I am not a fan of Sean Penn's, I'm really happy and I love the movie, and I's very	I am not a fan of Sean Penn. I appreciate what he is. It's awesome. This has been amazing.
	GPT-2-L	I am not a fan of Sean Penn <3 this film is great and worth watching! <3 <3 <3	I am not a fan of Sean Penn, but I love his work in baseball and I love his work for my favorite
In the original French version, the jokes	GPT-2	In the original French version, the jokes were pretty fun and pretty neat. I really liked	In the original French version, the jokes are amazing. I love them so much, I
	GPT-2-M	In the original French version, the jokes are beautifully clear and funny. I am a very	In the original French version, the jokes are great, but I am excited to look at
	GPT-2-L	In the original French version, the jokes were very funny! my main pleasure from this movie	In the original French version, the jokes were quite good and it was quite close to the

Table 7: The average reward score (denoted by r) on the test set of the Reddit TL;DR summarization dataset and ROUGE metrics (ROUGE-1, ROUGE-2, and ROUGE-L denoted by R-1, R-2, and R-L, respectively) between model generated summaries and the label summaries in the test set for various models and privacy levels. $\epsilon = 0$ represents the pre-trained model. $\epsilon \in \{1, 2, 4, 8\}$ are privately aligned models with different privacy budgets. $\epsilon = \infty$ is the alignment procedure without any privacy. Our results demonstrate that alignment towards human-preferred summarization is obtainable with formal privacy guarantees to the underlying dataset. Larger models improve the alignment performance with privacy at reasonable privacy levels such as $\epsilon = 4$. ROUGE metrics indicate that models can deviate from label summaries learned during SFT and align towards human-preferred summaries with PPO during alignment.

Model	ϵ	Stage	Mean Reward	R-1	R-2	R-L
GPT-2	0	Pre-trained	0.05	12.91	0.78	8.26
	1	SFT	0.44	16.69	1.69	11.45
		Aligned	0.22	14.69	1.50	10.41
	2	SFT	0.48	17.23	1.85	11.84
		Aligned	0.53	16.62	1.53	11.44
	4	SFT	0.50	17.84	2.02	12.30
		Aligned	0.68	17.75	1.80	12.33
	8	SFT	0.49	17.89	2.01	12.45
		Aligned	0.69	16.55	1.62	11.74
	∞	SFT	0.63	20.85	2.97	14.48
		Aligned	1.53	20.61	3.13	14.17
GPT-2 Medium	0	Pre-trained	0.11	13.53	0.90	8.67
	1	SFT	0.68	18.70	2.36	12.80
		Aligned	0.59	18.44	2.44	12.86
	2	SFT	0.66	18.79	2.47	13.07
		Aligned	0.92	19.60	2.34	13.26
	4	SFT	0.65	19.27	2.62	13.30
		Aligned	0.92	19.48	2.45	13.44
	8	SFT	0.65	19.62	2.62	13.50
		Aligned	0.86	19.85	2.65	13.79
	∞	SFT	0.70	20.59	2.85	14.30
		Aligned	1.76	19.64	2.50	13.17
GPT-2 Large	0	Pre-trained	-0.06	16.13	1.56	10.34
	1	SFT	0.51	21.67	3.37	14.98
		Aligned	0.40	21.17	3.28	14.75
	2	SFT	0.51	21.41	3.35	14.86
		Aligned	1.14	21.33	3.33	14.58
	4	SFT	0.52	21.83	3.47	15.14
		Aligned	1.06	19.63	2.83	13.88
	8	SFT	0.51	21.71	3.34	15.04
		Aligned	0.93	20.26	3.04	14.37
	∞	SFT	0.54	22.22	3.58	15.53
		Aligned	1.49	21.81	3.32	14.64