Differentially Private Continual Learning using Pre-Trained Models

Marlon Tobaben
Dept. of Computer Science
University of Helsinki
marlon.tobaben@helsinki.fi

Marcus Klasson
Dept. of Computer Science
Aalto University
marcus.klasson@aalto.fi

Rui Li
Dept. of Computer Science
Aalto University
rui.li@aalto.fi

Arno Solin
Dept. of Computer Science
Aalto University
arno.solin@aalto.fi

Antti Honkela
Dept. of Computer Science
University of Helsinki
antti.honkela@helsinki.fi

Abstract

This work explores the intersection of continual learning (CL) and differential privacy (DP). Crucially, continual learning models must retain knowledge across tasks, but this conflicts with the differential privacy requirement of restricting individual samples to be memorised in the model. We propose using pre-trained models to address the trade-offs between privacy and performance in a continual learning setting. More specifically, we present necessary assumptions to enable privacy-preservation and propose combining pre-trained models with parameter-free classifiers and parameter-efficient adapters that are learned under differential privacy. Our experiments demonstrate their effectiveness and provide insights into balancing the competing demands of continual learning and privacy.

1 Introduction

Continual learning (CL, [9, 53]) develops models that learn from a stream of tasks while retaining previous knowledge, a key requirement for real-world applications where data arrives sequentially. However, CL faces the challenge of catastrophic forgetting, where the model loses performance on earlier tasks as it learns new ones [16]. While CL fights for memorising prototypical aspects of the data, the paradigm known as differential privacy (DP, [13]) aims at not memorising any individual's data in the first place. DP offers a framework to ensure that the inclusion or exclusion of a single data point does not significantly impact the outcome of the learning process while providing provable privacy guarantees. In turn, DP gives means to enable machine learning (ML) models to, *e.g.*, comply with privacy regulations (*e.g.* GDPR) to ensure personal data is untraceable and mitigating model inversion attacks [19]. While DP is crucial for privacy-preserving ML, it introduces a trade-off: stronger privacy often degrades model accuracy [41].

Combining CL and DP presents unique challenges to satisfy the demands of mitigating catastrophic forgetting without violating privacy. Prior works have focused on learning how to generate DP synthetic samples for training the classifier and retaining previous knowledge to circumvent the need to store real data [7, 15]. Lai et al. [28] focus on the privacy loss accumulation when learning many tasks sequentially with an episodic memory [31]. However, these methods either require using DP learning of additional networks [15, 28] or on the synthetic samples [7] to enable privacy-preservation

Accepted at Scalable Continual Learning for Lifelong Foundation Models Workshop at 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

in the classifier. Recently, using pre-trained models has been studied separately in DP [27, 8, 49] and CL [23, 54, 55], but remains to be explored whether this can ease the balance between privacy and performance over time.

In this work, we explore using pre-trained models that can learn new tasks continually under DP constraints. We present necessary assumptions to obtain a privacy-preserving CL model, and experiment with two common approaches in CL with pre-trained models: *(i)* parameter-free classifiers [23], and *(ii)* parameter-efficient adapters [17]. We demonstrate the effectiveness of these methods and provide insights into how to balance privacy-utility trade-offs in CL under DP.

2 Related Work

Differential Privacy In terms of privacy-preserving ML, DP [13] is considered the gold standard to provide provable privacy guarantees, where the DP-SGD algorithm [42, 46, 2] is the standard learning approach. The main challenge is the trade-off between privacy and utility, *i.e.*, how to achieve the same model behaviours and performance as a non-private model without memorising individual data records. Ensuring privacy in deep learning models has recently gained great interest [57, 39] due to their high utility on large-scale data sets. The usage of pre-trained models has increased in popularity [27, 8, 33, 5, 30, 60, 49, 48, 52] with most state-of-the-art models relying on the assumption that the pre-training data is public. However, following the discussion of Tramèr et al. [50], we utilise pre-trained models trained on pre-training data that is small enough in size as carefully curating large pre-training data sets is very resource intensive/expensive [47]. If private information is contained in the pre-trained data, the DP privacy guarantees in regard to the fine-tuning data become meaningless.

Differentially Private Continual Learning In the intersection of CL and DP, previous works have focused on continually training classifiers with DP synthetic samples either by training a generative model under DP [15] or learn a small set of synthetic samples optimised towards the downstream task [7]. However, these methods have only demonstrated results on MNIST or CIFAR-10, possibly since learning how to generate synthetic samples of a larger size is challenging. Moreover, Lai et al. [28] introduce a formal definition of how to preserve lifelong DP when having episodic memory [31] for mitigating catastrophic forgetting. However, their method requires significant modifications of the CL pipeline by also learning an auto-encoder under DP with noise injection that processes which output is passed to the classifier. In this paper, we take a rehearsal-free approach and explore how to continually learn DP image classifiers with pre-trained backbones. More specifically, we experiment with using parameter-free classifiers similar to [23], as well as using parameter-efficient finetuning (PEFT) adapters with FiLM layers [40] to achieve both DP and good performance.

3 Methods

Continual Learning Setting We focus on the continual learning of image classification tasks, where we let a model $f_{\boldsymbol{\theta}}$ parameterised by $\boldsymbol{\theta}$ learn T tasks sequentially from the data sets $\mathcal{D}_1, \ldots, \mathcal{D}_T$. We denote $\mathcal{C}_t$ as the set of classes in task t and the total number of classes as $C = \sum_{t=1}^{T} |\mathcal{C}_t|$. The t^{th} data set $\mathcal{D}_t = \{(\boldsymbol{x}_t^{(i)}, y_t^{(i)})\}_{i=1}^{N_t}$ consists of N_t samples where $\boldsymbol{x}_t^{(i)} \in \mathbb{R}^{h \times w \times c}$ and $y_t^{(i)} \in \mathbb{N}$ are the i-th data point and class label respectively. Note that the data sets are inaccessible in the succeeding tasks. Recently, using pre-trained models as $f_{\boldsymbol{\theta}}$ has gained interest in CL where the task classifiers can be *(i)* linear layers learned via the cross-entropy loss for new tasks [55], or *(ii)* storing class-specific feature vectors and use a parameter-free classifier [23].

Differential Privacy We use the definition of DP as presented in Ponomareva et al. [41]: A mechanism $\mathcal{A}$ guarantees (ϵ, δ)-differential private if for any two datasets $\mathcal{D}$ and $\mathcal{D}'$ that only differ in exactly one example, and for any outcome $S \subseteq Range(\mathcal{A})$ satisfies

$$P(\mathcal{A}(\mathcal{D}) \in S) \leq \exp(\epsilon) \times P(\mathcal{A}(\mathcal{D}') \in S) + \delta, \tag{1}$$

where ϵ and $\delta \leq 1$ are non-negative scalars that control the allowed privacy loss and $Range(\mathcal{A})$ is the set of all possible outcomes of $\mathcal{A}$. Note that smaller values of ϵ and δ correspond to a stronger privacy guarantee. In deep learning, the mechanism $\mathcal{A}$ can be the optimisation method (*e.g.*, SGD) which produces a parameter set $\boldsymbol{\theta}$, or the mechanism is the method for computing class-specific features for producing a parameter-free classifier. The most common training approach is using DP-SGD

which minimises an empirical loss while clips and adds noise to the per-example gradients to protect privacy (see [2, 41] for details). Every access of the data $\mathcal{D}$ during training accumulates the privacy loss. Thus, the privacy loss is accumulated per training step and the training is stopped when the allowed loss is reached. In a CL context, in addition to learning new tasks, the privacy loss also needs to account for remembering past tasks, *e.g.*, by training a generative model or replaying stored samples. More background on DP in Appendix A.

Assumptions As combining DP into CL settings is challenging, we make the following assumptions: For **S1**, we assume that the total set of class labels is known. This assumption is necessary under DP as releasing information about which classes have been seen at a task t will leak private information about individual samples. In practice, we release the parameters for all classes at every task t, including untrained output heads for future classes and other tasks than t. For **S2**, we assume that the task boundaries are non-overlapping and that storing previous samples is forbidden. This simplifies the privacy accounting as each task data set $\mathcal{D}_i$ is independent and only used once. Thus, we can base our privacy accounting on parallel composition [32] instead of sequential composition [12] which would lead to worse privacy-utility trade-off, and possibly more complicated algorithms. Based on these assumptions, we propose two approaches for enabling DP CL using a pre-trained model:

Cosine Similarity Classifier (Algorithm A4) We use the pre-trained model $f_{\boldsymbol{\theta}}$ as a frozen feature extractor without additional training during CL [23]. At each task t, we compute a per-class sum of features with the Gaussian mechanism [3] for all classes:

$$\boldsymbol{s}_c = \begin{cases} \sum_{\boldsymbol{x} \in \mathcal{D}_{t,c}} f_{\boldsymbol{\theta}}(\boldsymbol{x}) + \mathcal{N}(0, \sigma I) & \text{if } c \in \mathcal{C}_t \\ \sum_{\boldsymbol{x} \in \mathcal{D}_{t,c}} \mathcal{N}(0, \sigma I) & \text{if } c \notin \mathcal{C}_t \end{cases}, \quad \forall c \in \bigcup_{i=1}^{T} \mathcal{C}_i, \tag{2}$$

where $\mathcal{N}(\mathbf{0}, \sigma I)$ is standard Gaussian noise with scale σ corresponding to the desired (ϵ, δ)-DP privacy budget, $\mathcal{D}_{t,c}$ are all samples from class c at task t, and $\mathcal{C}_t$ is the set of classes for task t. We compute the sum-of-features rather than the mean-of-features [23, 43] to avoid the need for counting the number of examples per class. At test time, we predict the label of a test sample $\boldsymbol{x}^*$ by assigning the class label from which the per-class feature sum that $\boldsymbol{x}^*$ is closest to in the feature space using the cosine similarity:

$$\hat{y}^* = \arg\max_c \text{CosineSimilarity}(f_{\boldsymbol{\theta}}(\boldsymbol{x}^*), \boldsymbol{s}_c). \tag{3}$$

We assume that the features and sums are normalised to unit norm in Equations (2) and (3)[1]. Note that we only need to store the per-class sums and the pre-trained model in the memory.

PEFT Ensemble (Algorithm A5) We construct a parameter-efficient fine-tuning (PEFT) ensemble by initialising task-specific output heads $g_{\boldsymbol{\phi}_t}(\boldsymbol{u})$ with parameters $\boldsymbol{\phi}_t$ using DP-SGD. Let the feature vector of the pre-trained model be denoted as $\boldsymbol{u} = f_{\boldsymbol{\theta}}(\boldsymbol{x}) \in \mathbb{R}^K$ where K is the feature dimension. In our case, every task-specific head is a mapping $g_{\boldsymbol{\phi}_t} : K \to C$ to the total number of classes C, since we must avoid leaking the number of seen classes at task t when releasing the model (see assumption **S2** above). At test time, we predict the label of a test feature $\boldsymbol{u}^* = f_{\boldsymbol{\theta}}(\boldsymbol{x}^*)$ by selecting the predicted class label that has the largest logit across all T heads:

$$\hat{y}^* = \arg\max_{t=\{1,\ldots,T\}} g_{\boldsymbol{\phi}_t}(\boldsymbol{u}^*). \tag{4}$$

To reduce storage of the output heads, we employ parameter-efficient FiLM [40] adapters as this approach has been effective in prior works on transfer learning (under DP) [45, 49].

4 Experiments

In all experiments, we utilise a ViT-Base-16 (ViT-B) [10] network pre-trained on the ImageNet-21K [44] dataset. We assume that the pre-training data is public and learn tasks with private data sets $\mathcal{D}_t$ that needs to be protected with DP. All experiments are in the class-incremental learning setting where no task labels are available [51]. See Appendix C for full experimental details.

Data sets We experiment with the following benchmarks for CL: Split CIFAR-100 which is CIFAR-100 [26] split into 10 tasks with 10 classes/task. 5-Dataset [14] that concatenates the five data sets,

[1] L2-norm and cosine similarity are equivalent when both vectors are normalised to unit norm. Using the L2-norm without normalising is challenging because the magnitude for different sums can be vary significantly. Using the mean instead of the sum is not possible as classes can have zero examples in some t.

MNIST [29], SVHN [36], notMNIST [4], FashionMNIST [56] and CIFAR-10 [26] where each forms one task. Split ImageNet-R [54] which is ImageNet-R [20] split into 10 tasks with 20 classes/task.

Metrics We report average accuracies and forgetting metrics to evaluate all methods [6, 34, 58]. The average accuracy measures the test set accuracy across all seen tasks, while forgetting is given by the difference between the highest accuracy of a task and its accuracy at the current task.

Baselines We compare against the following baselines:

- **Naive (Lower)**: We adapt a single output head with DP-SGD by learning all T tasks sequentially, which is a lower bound as no means to mitigate catastrophic forgetting are in place (See Algorithm A3).
- **Full Data (Upper)**: We adapt a single output head with DP-SGD with all data from the current and previous tasks $\bigcup_{i=1}^{t} \mathcal{D}_i$ as an upper bound. While this is DP at each single task, the collection of models as a whole has weaker DP guarantees (See Algorithm A2)[2]. Furthermore, the baseline requires storing all data.

Split-CIFAR-100 Figure 1 and Table A1 display the results of our proposed methods in comparison to the baselines. The PEFT ensemble trains only a linear layer for each task t, as there are only minor benefits of using more advanced fine-tuning techniques [49]. The PEFT ensemble outperforms the other CL methods in all experiments in terms of accuracy and forgetting measure, but requires storing T times more parameters than the Cosine Similarity Classifier and is computationally more expensive, as it only requires forward passes, whereas the PEFT ensemble relies on DP-SGD.

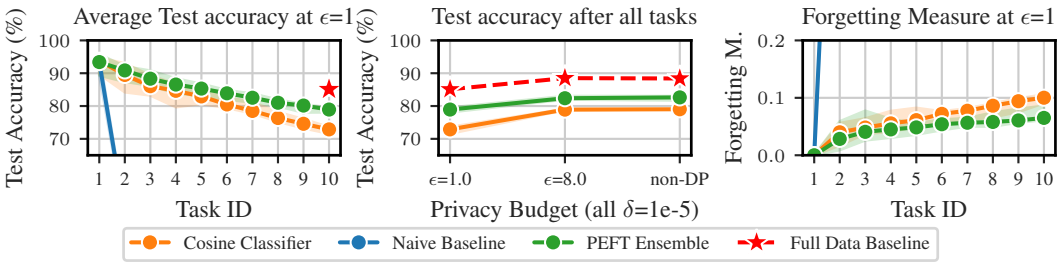

Figure 1: Median performance on Split-CIFAR-100 (ViT-B pre-trained on ImageNet-21k). The error bars are the min/max accuracies obtained over ten repeats with different class ordering and DP noise.

5-Dataset In Figure 2 and Table A2, we compare the performance of our proposed DP methods on 5-dataset. PEFT Ensemble is based on FiLM as this yields significantly higher accuracy for SVHN than just a linear layer. The PEFT Ensemble outperforms the other methods (See also Figure A.1).

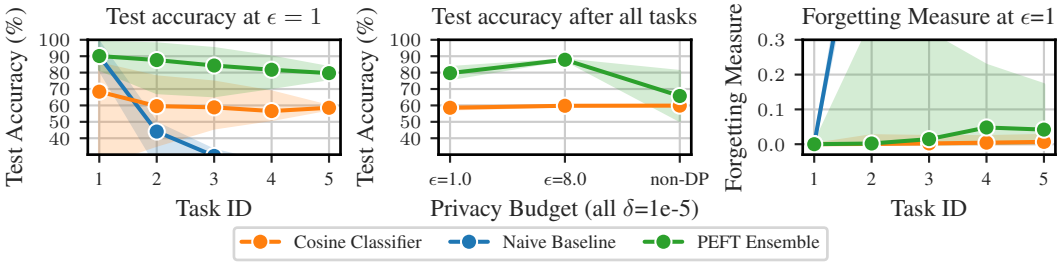

Figure 2: Median performance on 5-dataset (ViT-B pre-trained on ImageNet-21k). The error bars are the min/max accuracies obtained over all ordering permutations when fine-tuning three models per t.

Split ImageNet-R In Table 1, we observe that the cosine classifier outperforms the PEFT ensemble, which is based on training a linear layer, for all privacy budgets but $\epsilon = 8$, but we see room for improvement in future work. Our methods perform on-par with prior simple baselines [23] in the non-DP setting but leveraging a PEFT ensemble with FiLM seems promising.

[2]Note that when releasing a model at every task t (this is what we would expect in a CL setting), task data is released multiple times under DP. The Full Data baseline is still DP with regards to all D, but with a weaker (ϵ, δ)-DP privacy guarantee. E.g., as D_1 (data of $t = 1$) gets released T times under DP.

Table 1: Average accuracy (AA) and average forgetting (AF) after learning the final task in % on 10-task Split ImageNet-R. We report the mean and std of the metrics averaged over three seeds.

	$\epsilon = 1, \delta =$1e-5		$\epsilon = 8, \delta =$1e-5		non-DP	
Method	AA (↑)	AF (↓)	AA (↑)	AF (↓)	AA (↑)	AF (↓)
Naive	1.81 ± 0.28	19.74 ± 3.14	6.14 ± 3.18	61.05 ± 18.25	3.28 ± 3.43	48.68 ± 29.59
Cosine classifier	13.04 ± 1.23	13.89 ± 2.97	46.17 ± 0.21	12.21 ± 2.15	56.30 ± 0.00	7.51 ± 1.49
PEFT Ensemble	7.29 ± 2.82	3.54 ± 0.62	47.97 ± 2.22	6.33 ± 0.72	48.16 ± 12.80	6.54 ± 4.19
Full Data	16.57 ± 2.86	-	52.16 ± 4.35	-	62.17 ± 2.03	-

5 Discussion and Conclusion

We present necessary assumptions for DP CL, and introduce two approaches: PEFT Ensemble clearly works well in settings where most data for a single class is in a single task, but might suffer if this is not the case. The cosine similarity classifier is invariant to splitting the data into tasks. Developing flexible methods that work for distributed data configurations is an interesting future challenge. Generalising beyond our assumptions opens new possibilities. Assumption S1 will be very difficult to avoid, but assumption S2 could be circumvented for example by using DP generative models.

Acknowledgments and Disclosure of Funding

This work was supported by the Research Council of Finland Flagship programme Finnish Center for Artificial Intelligence FCAI, Research Council of Finland grants 358247 and 339730 as well as the European Union (Project 101070617). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them. The authors wish to thank the CSC – IT Center for Science, Finland for supporting this project with computational and data storage resources. We thank Aki Rehn for the helpful discussions.

References

[1] TensorFlow Datasets, a collection of ready-to-use datasets. https://www.tensorflow.org/datasets. 13, 14

[2] M. Abadi, A. Chu, I. J. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In E. R. Weippl, S. Katzenbeisser, C. Kruegel, A. C. Myers, and S. Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 308–318. ACM, 2016. doi: 10.1145/2976749.2978318. URL https://doi.org/10.1145/2976749.2978318. 2, 3, 10, 11

[3] B. Balle and Y. Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In J. G. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 403–412. PMLR, 2018. URL http://proceedings.mlr.press/v80/balle18a.html. 3, 12

[4] Y. Bulatov. notMNIST dataset, 2011. URL http://yaroslavvb.blogspot.it/2011/09/notmnist-dataset.html. 4, 14

[5] Y. Cattan, C. A. Choquette-Choo, N. Papernot, and A. Thakurta. Fine-tuning with differential privacy necessitates an additional hyperparameter search. *CoRR*, abs/2210.02156, 2022. doi: 10.48550/arXiv.2210.02156. URL https://doi.org/10.48550/arXiv.2210.02156. 2

[6] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*, volume 11215 of *Lecture Notes in Computer Science*, pages 556–572. Springer, 2018. doi: 10.1007/978-3-030-01252-6_33. URL https://doi.org/10.1007/978-3-030-01252-6_33. 4, 13

[7] D. Chen, R. Kerkouche, and M. Fritz. Private set generation with discriminative information. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/5e1a87dbb7e954b8d9d6c91f6db771eb-Abstract-Conference.html. 1, 2

[8] S. De, L. Berrada, J. Hayes, S. L. Smith, and B. Balle. Unlocking high-accuracy differentially private image classification through scale. *ArXiv preprint*, abs/2204.13650, 2022. URL https://arxiv.org/abs/2204.13650. 2

[9] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021. 1

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=YicbFdNTTy. 3, 13, 14

[11] A. Douillard and T. Lesort. Continuum: Simple management of complex continual learning scenarios, 2021. 13

[12] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In S. Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503. Springer, 2006. doi: 10.1007/11761679_29. URL https://doi.org/10.1007/11761679_29. 3, 10

[13] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith. Calibrating noise to sensitivity in private data analysis. *J. Priv. Confidentiality*, 7(3):17–51, 2016. doi: 10.29012/JPC.V7I3.405. URL https://doi.org/10.29012/jpc.v7i3.405. 1, 2

[14] S. Ebrahimi, F. Meier, R. Calandra, T. Darrell, and M. Rohrbach. Adversarial continual learning. In A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XI*, volume 12356 of *Lecture Notes in Computer Science*, pages 386–402. Springer, 2020. doi: 10.1007/978-3-030-58621-8_23. URL https://doi.org/10.1007/978-3-030-58621-8_23. 3

[15] S. Farquhar and Y. Gal. Differentially private continual learning. *CoRR*, abs/1902.06497, 2019. URL http://arxiv.org/abs/1902.06497. 1, 2

[16] R. M. French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4): 128–135, 1999. 1

[17] Q. Gao, C. Zhao, Y. Sun, T. Xi, G. Zhang, B. Ghanem, and J. Zhang. A unified continual learning framework with general parameter-efficient tuning. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 11449–11459. IEEE, 2023. doi: 10.1109/ICCV51070.2023.01055. URL https://doi.org/10.1109/ICCV51070.2023.01055. 2

[18] S. Gopi, Y. T. Lee, and L. Wutschitz. Numerical composition of differential privacy. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 11631–11642, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/6097d8f3714205740f30debe1166744e-Abstract.html. 10

[19] N. Haim, G. Vardi, G. Yehudai, O. Shamir, and M. Irani. Reconstructing training data from trained neural networks. *Advances in Neural Information Processing Systems*, 35:22911–22924, 2022. 1

[20] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021. 4, 14

[21] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for NLP. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 2019. URL http://proceedings.mlr.press/v97/houlsby19a.html. 13

[22] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9. 13

[23] P. Janson, W. Zhang, R. Aljundi, and M. Elhoseiny. A simple baseline that questions the use of pretrained-models in continual learning. *CoRR*, abs/2210.04428, 2022. doi: 10.48550/ARXIV.2210.04428. URL https://doi.org/10.48550/arXiv.2210.04428. 2, 3, 4, 12, 14

[24] A. Koskela, J. Jälkö, and A. Honkela. Computing tight differential privacy guarantees using FFT. In S. Chiappa and R. Calandra, editors, *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 2560–2569. PMLR, 2020. URL http://proceedings.mlr.press/v108/koskela20b.html. 10

[25] A. Koskela, J. Jälkö, L. Prediger, and A. Honkela. Tight differential privacy for discrete-valued mechanisms and for the subsampled gaussian mechanism using FFT. In A. Banerjee and K. Fukumizu, editors, *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*, volume 130 of *Proceedings of Machine Learning Research*, pages 3358–3366. PMLR, 2021. URL http://proceedings.mlr.press/v130/koskela21a.html. 10

[26] A. Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, University of Toronto, 2009. 3, 4, 14

[27] A. Kurakin, S. Chien, S. Song, R. Geambasu, A. Terzis, and A. Thakurta. Toward training at imagenet scale with differential privacy. *CoRR*, abs/2201.12328, 2022. URL https://arxiv.org/abs/2201.12328. 2

[28] P. Lai, H. Hu, H. Phan, R. Jin, M. T. Thai, and A. M. Chen. Lifelong DP: consistently bounded differential privacy in lifelong machine learning. In S. Chandar, R. Pascanu, and D. Precup, editors, *Conference on Lifelong Learning Agents, CoLLAs 2022, 22-24 August 2022, McGill University, Montréal, Québec, Canada*, volume 199 of *Proceedings of Machine Learning Research*, pages 778–797. PMLR, 2022. URL https://proceedings.mlr.press/v199/lai22a.html. 1, 2

[29] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010. 4, 14

[30] X. Li, F. Tramèr, P. Liang, and T. Hashimoto. Large language models can be strong differentially private learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=bVuP3ltATMz. 2

[31] D. Lopez-Paz and M. Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017. 1, 2

[32] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. *Commun. ACM*, 53(9):89–97, 2010. doi: 10.1145/1810891.1810916. URL https://doi.org/10.1145/1810891.1810916. 3

[33] H. Mehta, A. G. Thakurta, A. Kurakin, and A. Cutkosky. Towards large scale transfer learning for differentially private image classification. *Trans. Mach. Learn. Res.*, 2023, 2023. URL https://openreview.net/forum?id=Uu8WwCFpQv. 2

[34] S. I. Mirzadeh, M. Farajtabar, D. Gorur, R. Pascanu, and H. Ghasemzadeh. Linear mode connectivity in multitask and continual learning. In *International Conference on Learning Representations*, 2021. 4, 13

[35] P. K. Mudrakarta, M. Sandler, A. Zhmoginov, and A. G. Howard. K for the price of 1: Parameter-efficient multi-task and transfer learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=BJxvEh0cFQ. 13

[36] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. 4, 14

[37] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html. 10, 13

[38] M. Patacchiola, J. Bronskill, A. Shysheya, K. Hofmann, S. Nowozin, and R. E. Turner. Contextual squeeze-and-excitation for efficient few-shot image classification. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/ee1e549d6fb7c58ed06557bfc264335c-Abstract-Conference.html. 13

[39] M. Pelikan, S. S. Azam, V. Feldman, J. H. Silovsky, K. Talwar, and T. Likhomanenko. Federated learning with differential privacy for end-to-end speech recognition. *CoRR*, abs/2310.00098, 2023. doi: 10.48550/ARXIV.2310.00098. URL https://doi.org/10.48550/arXiv.2310.00098. 2

[40] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville. Film: Visual reasoning with a general conditioning layer. In S. A. McIlraith and K. Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3942–3951. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16528. 2, 3, 13

[41] N. Ponomareva, H. Hazimeh, A. Kurakin, Z. Xu, C. Denison, H. B. McMahan, S. Vassilvitskii, S. Chien, and A. G. Thakurta. How to dp-fy ML: A practical guide to machine learning with differential privacy. *J. Artif. Intell. Res.*, 77:1113–1201, 2023. doi: 10.1613/JAIR.1.14649. URL https://doi.org/10.1613/jair.1.14649. 1, 2, 3

[42] A. Rajkumar and S. Agarwal. A differentially private stochastic gradient descent algorithm for multiparty classification. In N. D. Lawrence and M. A. Girolami, editors, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2012, La Palma, Canary Islands, Spain, April 21-23, 2012*, volume 22 of *JMLR Proceedings*, pages 933–941. JMLR.org, 2012. URL http://proceedings.mlr.press/v22/rajkumar12.html. 2, 10

[43] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 3, 12

[44] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y. 3, 13

[45] A. Shysheya, J. Bronskill, M. Patacchiola, S. Nowozin, and R. E. Turner. FiT: parameter efficient few-shot transfer learning for personalized and federated image classification. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/pdf?id=9aokcgBVIj1. 3, 13

[46] S. Song, K. Chaudhuri, and A. D. Sarwate. Stochastic gradient descent with differentially private updates. In *IEEE Global Conference on Signal and Information Processing, GlobalSIP 2013, Austin, TX, USA, December 3-5, 2013*, pages 245–248. IEEE, 2013. doi: 10.1109/GlobalSIP.2013.6736861. URL https://doi.org/10.1109/GlobalSIP.2013.6736861. 2, 10

[47] D. Thiel. Identifying and eliminating csam in generative ml training data and models. Technical report, Technical Report. Stanford University, Palo Alto, CA., 2023. URL https://purl.stanford.edu/kh752sm9123. 2

[48] R. Tito, K. Nguyen, M. Tobaben, R. Kerkouche, M. A. Souibgui, K. Jung, L. Kang, E. Valveny, A. Honkela, M. Fritz, and D. Karatzas. Privacy-aware document visual question answering. *CoRR*, abs/2312.10108, 2023. doi: 10.48550/ARXIV.2312.10108. URL https://doi.org/10.48550/arXiv.2312.10108. 2

[49] M. Tobaben, A. Shysheya, J. Bronskill, A. Paverd, S. Tople, S. Z. Béguelin, R. E. Turner, and A. Honkela. On the efficacy of differentially private few-shot image classification. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=hFsr59Imzm. 2, 3, 4, 13, 14

[50] F. Tramèr, G. Kamath, and N. Carlini. Position: Considerations for differentially private learning with large-scale public pretraining. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ncjhi4qAPV. 2

[51] G. M. van de Ven, T. Tuytelaars, and A. S. Tolias. Three types of incremental learning. *Nat. Mac. Intell.*, 4(12):1185–1197, 2022. doi: 10.1038/S42256-022-00568-3. URL https://doi.org/10.1038/s42256-022-00568-3. 3

[52] D. Wahdany, M. Jagielski, A. Dziedzic, and F. Boenisch. Beyond the mean: Differentially private prototypes for private transfer learning. *arXiv preprint arXiv:2406.08039*, 2024. 2, 12

[53] L. Wang, X. Zhang, H. Su, and J. Zhu. A comprehensive survey of continual learning: theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 1

[54] Z. Wang, Z. Zhang, S. Ebrahimi, R. Sun, H. Zhang, C.-Y. Lee, X. Ren, G. Su, V. Perot, J. Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pages 631–648. Springer, 2022. 2, 4, 14

[55] Z. Wang, Z. Zhang, C. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. G. Dy, and T. Pfister. Learning to prompt for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 139–149. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00024. URL https://doi.org/10.1109/CVPR52688.2022.00024. 2

[56] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL http://arxiv.org/abs/1708.07747. 4, 14

[57] Z. Xu, Y. Zhang, G. Andrew, C. A. Choquette-Choo, P. Kairouz, H. B. McMahan, J. Rosenstock, and Y. Zhang. Federated learning of gboard language models with differential privacy. In S. Sitaram, B. B. Klebanov, and J. D. Williams, editors, *Proceedings of the The 61st Annual Meeting of the Association for Computational Linguistics: Industry Track, ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 629–639. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-INDUSTRY.60. URL https://doi.org/10.18653/v1/2023.acl-industry.60. 2

[58] J. Yoon, D. Madaan, E. Yang, and S. J. Hwang. Online coreset selection for rehearsal-based continual learning. In *International Conference on Learning Representations*, 2022. 4, 13

[59] A. Yousefpour, I. Shilov, A. Sablayrolles, D. Testuggine, K. Prasad, M. Malek, J. Nguyen, S. Gosh, A. Bharadwaj, J. Zhao, G. Cormode, and I. Mironov. Opacus: User-friendly differential privacy library in pytorch. *ArXiv preprint*, abs/2109.12298, 2021. URL https://arxiv.org/abs/2109.12298. 10, 13

[60] D. Yu, S. Naik, A. Backurs, S. Gopi, H. A. Inan, G. Kamath, J. Kulkarni, Y. T. Lee, A. Manoel, L. Wutschitz, S. Yekhanin, and H. Zhang. Differentially private fine-tuning of language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=Q42f0dfjECO. 2

Appendices

A Background on Differentially Private Deep Learning

A.1 Differentially private stochastic gradient descent (DP-SGD)

The most straightforward way of training deep learning models under DP is using DP-SGD which clips per-example gradients and adds noise to the aggregate. Algorithm A1 displays DP-SGD. There are many implementations of DP-SGD but we use the established implementation in the PyTorch [37] compatible library opacus [59].

Algorithm A1 Differentially private stochastic gradient descent (DP-SGD)[42, 46, 2]

Hyper-parameters: learning rate η_t, noise multiplier σ^2, lot size L, clipping bound C, number of steps T

1: **for** $t \in [T]$ **do**
2: Take a random sample L_t with sampling probability L/N
3: **Compute (per example) gradients**
4: For each $i \in L_t$ compute $g_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$
5: **Clip gradients**
6: $\tilde{g}_t(x_i) \leftarrow g_t(x_i) / \max(1, \frac{||g_t(x_i)||_2}{C})$
7: **Add noise**
8: $\tilde{g}_t \leftarrow \frac{1}{|L_t|}(\Sigma_i \tilde{g}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 I))$
9: **Descent**
10: $\theta_{t+1} = \theta_t - \eta_t \tilde{g}_t$
11: **end for**

Output: θ_T and overall privacy cost (ϵ, δ) computed using a privacy accountant.

A.1.1 Privacy Accounting

DP-SGD introduces a trade-off between privacy and utility. Stronger privacy guarantees require introducing more noise, which proportionately degrades model accuracy. (ϵ, δ)-DP [12] has a privacy budget consisting of $\epsilon \geq 0$ and $\delta \in [0, 1]$, where smaller values of each correspond to a stronger privacy guarantee.

The privacy guarantee of a DP-SGD run is based on the hyper-parameters used for running DP-SGD and can be quantified using privacy accountants [24, 25, 18]. The following hyper-parameters have an influence on the privacy guarantee of DP-SGD:

- **Number of steps T:** Training for more iterations results in a lower privacy guarantee.
- **Sampling probability L/N:** A higher sampling probability increases the probability that a sample will be used for computing gradients at an iteration t and thus results in a lower privacy guarantee.
- **Noise multiplier σ^2:** A higher noise multiplier σ^2 results in a higher privacy guarantee.

The learning rate η does not influence the privacy guarantee as it only scales the final update that is already DP. The clipping bound C is an important hyperparameter in DP-SGD but it is not influencing the privacy guarantee as a higher clipping bound C will also lead to more noise being added to the aggregation the gradient as can be seen in line 8 of Algorithm A1. One needs to adjust the clipping bound C in a way that not too much information is lost due to the clipping of gradients but at the same time not too much noise is added to aggregate.

B Method Details

Below, we provide pseudo-codes for the methods introduced in Section 3.

B.1 Full Data Baseline

We assume the availability of all data from the current and prior tasks $\cup_i^t \mathcal{D}_i$ and train a model with DP-SGD [2]. While this is DP at each single task, the collection of models as a whole has weaker DP guarantees.

Algorithm A2 Full Data Baseline

Training of the models

for $t \in [T]$ **do**
    Initialise model $\mathcal{M}_t$
    $\mathcal{M}_t \leftarrow$ init()
    Train a model with all the data seen so far
    $\mathcal{M}_t \leftarrow$ DP-SGD$(\mathcal{M}_t; \cup_i^t D_i)$
end for

Output: a set of T models $\{\mathcal{M}_1 \dots \mathcal{M}_T\}$

Test of the models

for $t \in [T]$ **do**
    Predict test data of all tasks seen so far
    for $x_i^* \in \cup_i^t \mathcal{D}_i^{\text{test}}$ **do**
        $\hat{y_i}^* \leftarrow \arg\max_k \mathcal{M}_t(x_i*)$
    end for
    Evaluate average accuracy and forgetting
    $\mathcal{A}_t, \mathcal{F}_t \leftarrow$ eval$(\{y_i \dots\}, \{y_i^* \dots\},$ per-task accuracy history$)$
end for

Output: average accuracy for all tasks T $\{\mathcal{A}_1 \dots \mathcal{A}_T\}$, average forgetting for all tasks T $\{\mathcal{F}_1 \dots \mathcal{F}_T\}$

B.2 Naive Baseline

We start training a model $\mathcal{M}$ at task $t = 1$ under DP and continue further training it for all tasks T using DP-SGD. This is a lower bound as no measures are in place to mitigate catastrophic forgetting.

Algorithm A3 Naive Baseline

Initialise model $\mathcal{M}_1$
$\mathcal{M} \leftarrow$ init()
for $t \in [T]$ **do**
    Continue to train the model with the data of the current task t
    $\mathcal{M} \leftarrow$ DP-SGD$(\mathcal{M}; D_t)$
end for

Output: a set of T model checkpoints $\{\mathcal{M}_1 \dots \mathcal{M}_T\}$ (note that this is for evaluating)

Test of the models

for $t \in [T]$ **do**
    Predict test data of all tasks seen so far
    for $x_i^* \in \cup_i^t \mathcal{D}_i^{\text{test}}$ **do**
        $\hat{y_i}^* \leftarrow \arg\max_k \mathcal{M}_t(x_i*)$
    end for
    Evaluate average accuracy and forgetting
    $\mathcal{A}_t, \mathcal{F}_t \leftarrow$ eval$(\{y_i \dots\}, \{y_i^* \dots\},$ per-task accuracy history$)$
end for

Output: average accuracy for all tasks T $\{\mathcal{A}_1 \dots \mathcal{A}_T\}$, average forgetting for all tasks T $\{\mathcal{F}_1 \dots \mathcal{F}_T\}$

B.3 Cosine Classifier

We use the pre-trained model $f_{\boldsymbol{\theta}}$ as a frozen feature extractor without additional training during CL [23]. At each task t, we compute per-class sum of features with the Gaussian mechanism [3] for all classes:

$$\boldsymbol{s}_c = \begin{cases} \sum_{\boldsymbol{x} \in \mathcal{D}_{t,c}} f_{\boldsymbol{\theta}}(\boldsymbol{x}) + \mathcal{N}(0, \sigma I) & \text{if } c \in \mathcal{C}_t \\ \sum_{\boldsymbol{x} \in \mathcal{D}_{t,c}} \mathcal{N}(0, \sigma I) & \text{if } c \notin \mathcal{C}_t \end{cases}, \quad \forall c \in \bigcup_{i=1}^{T} \mathcal{C}_i, \tag{A1}$$

where $\mathcal{N}(\mathbf{0}, \sigma I)$ is standard Gaussian noise with scale σ corresponding to the desired (ϵ, δ)-DP privacy budget, $\mathcal{D}_{t,c}$ are all samples from class c, and $\mathcal{C}_t$ is the set of classes for task t. We compute the sum-of-features rather than the mean-of-features [23, 43] to avoid the need for counting the number of examples per class. At test time, we predict the label of a test sample $\boldsymbol{x}^*$ by assigning the class label from which the per-class feature sum that $\boldsymbol{x}^*$ is closest to in the feature space using the cosine similarity:

$$\hat{y}^* = \arg\max_c \text{CosineSimilarity}(f_{\boldsymbol{\theta}}(\boldsymbol{x}^*), \boldsymbol{s}_c). \tag{A2}$$

We assume that the features and sums are normalized to unit norm in Equations (2) and (3). Note that we only need to store the per-class sums and the pre-trained model in the memory.

(We would like to refer to Wahdany et al. [52] to a related method that uses DP prototypes but does not take into account CL.)

Algorithm A4 Cosine Classifier

Creation of the Cosine Classifier

Initialise cumulative sum
for $c \in [C]$ **do**
    $\boldsymbol{s}_{c,1} \leftarrow 0$
end for
for $t \in [T]$ **do**
    Compute DP sum for each c **and add it to cumulative sum.**
    for $c \in [C]$ **do**
        $\Sigma_{t,c} \leftarrow \sum_{\boldsymbol{x} \in \mathcal{C}_c} (f_{\boldsymbol{\theta}}(\boldsymbol{x})) + \mathcal{N}(0, \sigma I)$ with $\|f_{\boldsymbol{\theta}}(\boldsymbol{x})\|_2 = 1$
        $\boldsymbol{s}_{c,t} \leftarrow \boldsymbol{s}_{c,t-1} + \Sigma_{t,c}$
    end for
end for

Output: $\{\boldsymbol{s}_{1,1} \dots \boldsymbol{s}_{c,t}\}$ (a set of $T * C$ cumulative sums)

Test of the models

for $t \in [T]$ **do**
    Predict test data of all tasks seen so far
    for $x_i^* \in \cup_i^t \mathcal{D}_i^{\text{test}}$ **do**
        Predict using cosine similarity
        $\hat{y}^* \leftarrow \arg\max_c \text{CosineSimilarity}(f_{\boldsymbol{\theta}}(\boldsymbol{x}^*), \{\boldsymbol{s}_{1t}, \dots, \boldsymbol{s}_{ct}\})$
    end for
    Evaluate average accuracy and forgetting
    $\mathcal{A}_t, \mathcal{F}_t \leftarrow \text{eval}(\{y_i \dots\}, \{y_i^* \dots\}, \text{per-task accuracy history})$
end for

Output: average accuracy for all tasks T $\{\mathcal{A}_1 \dots \mathcal{A}_T\}$, average forgetting for all tasks T $\{\mathcal{F}_1 \dots \mathcal{F}_T\}$

B.4 PEFT Ensemble

We construct an parameter-efficient fine-tuning (PEFT) ensemble by training a model $\mathcal{M}_t$ on the data set $\mathcal{D}_t$ contained in task t using DP-SGD and storing all the models. Note that under DP the width of the final layer needs to be constant, as otherwise the number of classes seen at a task t will be leaked. At test time, we predict the label of a test sample $\boldsymbol{x}^*$ by assigning the class label from which the logit is the largest:

$$\hat{y}^* = \arg\max_{c,t} \mathcal{M}_t(\boldsymbol{x}^*). \tag{A3}$$

Note while we need to store all models, this can be done storage efficiently with parameter-efficient fine-tuning [21] with adaptation methods such as LoRA [22] when only the adapter weights, the final classification layer and the pre-trained model need to be stored.

Algorithm A5 PEFT Ensemble

Training of the models

for $t \in [T]$ **do**
  Initialise model $\mathcal{M}_t$
  $\mathcal{M}_t \leftarrow$ init()
  Train the model $\mathcal{M}_t$ **with the data of the current task** t
  $\mathcal{M}_t \leftarrow$ DP-SGD$(\mathcal{M}_t; \cup_i^t D_i)$
end for

Output: $\{\mathcal{M} \dots \mathcal{M}_T\}$ (a set of T models)

Test of the models

for $t \in [T]$ **do**
  Predict test data of all tasks seen so far
  for $x_i^* \in \cup_i^t \mathcal{D}_i^{\text{test}}$ **do**
    Predict using max logit over all models trained at tasks $i < t$
    $\hat{y_i}^* \leftarrow \arg\max_{c,t} \mathcal{M}_t(x_i*)$
  end for
  Evaluate average accuracy and forgetting
  $\mathcal{A}_t, \mathcal{F}_t \leftarrow$ eval$(\{y_i \dots\}, \{y_i^* \dots\},$ per-task accuracy history$)$
end for

Output: average accuracy for all tasks T $\{\mathcal{A}_1 \dots \mathcal{A}_T\}$, average forgetting for all tasks T $\{\mathcal{F}_1 \dots \mathcal{F}_T\}$

C Experimental Details

Pre-trained Model Throughout all experiments, we utilise a Vision Transformer VIT-Base-16 (VIT-B) [10] with 85.8M parameters, pretrained on the ImageNet-21K [44] dataset. We assume that the pre-training data (ImageNet-21K) is public, and the downstream data $\mathcal{D}$ is private and needs to be protected with DP. We set the weights of the last linear layer of the ViT-B to zero and always learn them when fine-tuning on $\mathcal{D}$. Additionally, we employ parameter-efficient fine-tuning in some experiments by learning FiLM [40] layers. Although there are many other such adapters such as Model Patch [35], LoRA [22], CaSE [38] *etc.*, we chose FiLM as it has proven to be highly effective in prior works on (DP) parameter-efficient few-shot transfer learning [45, 49]. We implement our methods using PyTorch [37], tensorflow datasets [1], continuum [11], and opacus [59].

Metrics We report the average accuracy and the average forgetting as [6, 34, 58]:

- **Accuracy**: The average accuracy at task t is defined as

$$A_t = \frac{1}{t} \sum_{i=1}^{t} a_{t,i} \in [0, 1], \tag{A4}$$

  where $a_{t,i} \in [0, 1]$ is the test set accuracy for task i after learning task t. Note that A_T is the average accuracy across all tasks after the final task T has been learned.

- **Forgetting**: The forgetting of task i is defined as the difference between its highest accuracy and its accuracy at the current task t as

$$f_{t,i} = \max_{k \in \{1,\dots,t-1\}} (a_{k,i} - a_{t,i}) \in [-1, 1], \tag{A5}$$

  such that the average forgetting at task t is then given by

$$F_t = \frac{1}{t-1} \sum_{i=1}^{t-1} f_{t,i} \in [-1, 1]. \tag{A6}$$

Note that F_T is the average forgetting across the first $T-1$ tasks as there is no forgetting of the final learned task T.

Data sets We experiment with the following data sets:

- **Split CIFAR-100**: Split CIFAR-100 which is CIFAR-100 [26] split into 10 tasks with 10 classes/task. We randomly permute the class order for each seed we run.
- **5-Datasets**: This data set concatenates the five 10-class data sets, MNIST [29], SVHN [36], notMNIST [4], FashionMNIST [56] and CIFAR-10 [26]. Each data set is considered a task, such that there are 5 tasks with 10 classes/task. We run experiments with all permutations of the tasks.
- **Split ImageNet-R**: This data set consists 30,000 images of renditions (art, cartoons, etc.) of 200 ImageNet classes [20] and was introduced by Wang et al. [54] as a benchmark for CL with pre-trained models. As in [23, 54], we split the classes into 10 tasks with 20 classes/task. We make a random 80/20% train/test split across the whole dataset. The samples per class are imbalanced in the original data set, and we obtain a 41-334 samples/class in our training sets with our split.

C.1 Hyperparameters

We tune the hyperparameters of DP-SGD for each combination privacy budget (ϵ, δ) and seed once and use them for the PEFT Ensemble, Full Data Baseline and Naive Baseline for all data sets. While this provides us with reasonable results, we believe that future work should tune the hyperparameters more carefully to obtain better trade-offs between privacy and utility.

Note that this does not apply for the full data baseline on Split-CIFAR-100 (Figure 1) where the accuracies have been obtained from Tobaben et al. [49].

C.2 Licenses and Access

The Vision Transformer ViT-Base-16 (ViT-B) [10] is licensed with the Apache-2.0 license and can be obtained through the instructions on https://github.com/google-research/vision_transformer.

The licenses and means to access the data sets can be found below. We downloaded all data sets but notMNIST and ImageNet-R from TensorFlow datasets [1] https://www.tensorflow.org/datasets.

- CIFAR10 [26] is licensed with an unknown license and we use version 3.0.2 of the data set as specified on https://www.tensorflow.org/datasets/catalog/cifar10.
- CIFAR100 [26] is licensed with an unknown license and we use version 3.0.2 of the data set as specified on https://www.tensorflow.org/datasets/catalog/cifar100.
- FashionMNIST [56] is licensed under MIT and we use version 3.0.1 of the data set as specified on https://www.tensorflow.org/datasets/catalog/fashion_mnist.
- ImageNet-R [20] is licensed with an unknown license and we use the version from https://people.eecs.berkeley.edu/~hendrycks/imagenet-r.tar.
- MNIST [29] is licensed with an unknown license and we use version 3.0.1 of the data set as specified on https://www.tensorflow.org/datasets/catalog/mnist.
- notMNIST [4] is licensed under an unknown license and we use the version at git hash 339df59 found at https://github.com/facebookresearch/Adversarial-Continual-Learning/blob/main/data/notMNIST.zip
- SVHN [36] is licensed under CC and we use version 3.1.0 of the data set as specified on https://www.tensorflow.org/datasets/catalog/svhn_cropped.

D Detailed Results

In this section we provide detailed tabular results for the Figures 1 and 2.

Table A1: Average accuracy (AA) and average forgetting (AF) after learning the final task in % on 10-task Split CIFAR-100. We report the mean and std of the metrics averaged over 10 seeds for all methods but the full data baseline.

	$\epsilon = 1$, δ =1e-5		$\epsilon = 8$, δ =1e-5		non-DP	
Method	AA (↑)	AF (↓)	AA (↑)	AF (↓)	AA (↑)	AF (↓)
Naive	9.35 ± 0.14	94.10 ± 0.00	0.00 ± 0.23	-	9.63 ± 0.05	-
Cosine classifier	72.78 ± 0.51	0.09 ± 0.00	78.93 ± 0.10	-	79.02 ± 0.00	-
PEFT Ensemble	78.81 ± 0.48	0.06 ± 0.00	82.48 ± 0.31	-	82.60 ± 0.38	-
Full Data	85.1	-	88.5	-	88.4	-

Table A2: Average accuracy (AA) and average forgetting (AF) after learning the final task in % on 5-dataset. We report the mean and std of the metrics averaged over all task order permutations.

	$\epsilon = 1$, δ =1e-5		$\epsilon = 8$, δ =1e-5		non-DP	
Method	AA (↑)	AF (↓)	AA (↑)	AF (↓)	AA (↑)	AF (↓)
Naive	15.93 ± 1.35	0.85 ± 0.05	17.56 ± 1.35	0.90 ± 0.02	13.15 ± 5.84	0.79 ± 0.12
Cosine classifier	58.54 ± 0.35	0.01 ± 0.00	59.78 ± 0.07	0.00 ± 0.00	59.87 ± 0.00	0.00 ± 0.00
PEFT Ensemble	79.69 ± 3.51	0.05 ± 0.04	87.83 ± 0.00	0.03 ± 0.02	65.75 ± 15.07	0.14 ± 0.11

The drop in performance for non-DP PEFT on the 5-dataset is something to investigate in future work. One would expect that non-DP always performs better than DP, but this is not the case for the 5-dataset experiment with the PEFT Ensemble. Without CL the individual models that are forming the PEFT Ensemble are performing better without DP than with DP (see Figure A.1), but in the ensemble this is not the case as can be seen in Figure 2.

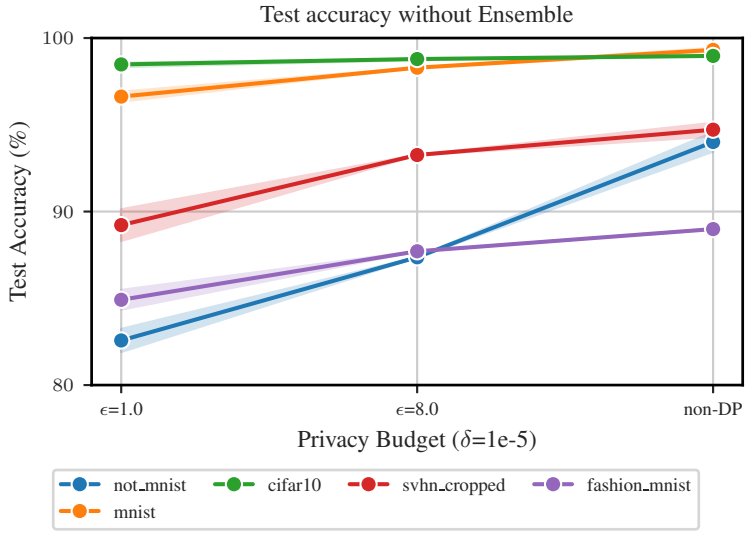

Figure A.1: PEFT Ensemble models without considering CL but just evaluating on the respective test set.