Guiding The Last Layer in Federated Learning with Pre-Trained Models

Gwen Legate [1 2] **Nicolas Bernier** [1 2] **Lucas Caccia** [3 2] **Edouard Oyallon** [4 5 6] **Eugene Belilovsky** [1 2]

Abstract

Federated Learning (FL) is an emerging paradigm that enables a model to be trained across a number of participants without sharing data. While recent works have begun to consider the effects of using pre-trained models as an initialization point for existing FL algorithms, these approaches ignore the vast body of efficient transfer learning literature from the centralized learning setting. Here we revisit the problem of FL initialization from a pre-trained model considered in prior work and expand it to a set of computer vision transfer learning problems. We first show that simply fitting a linear classification head can be efficient and effective in many cases. Second we demonstrate that in the FL setting, fitting a classifier using the Nearest Class Means (NCM) can be done exactly and orders of magnitude more efficiently than existing proposals, while obtaining strong performance. Finally, we present that a two-phase approach of first obtaining the classifier and then fine-tuning the model can yield rapid convergence and improved generalization in the federated setting. We demonstrate the potential our method has to reduce communication and compute costs while achieving better model performance.

1. Introduction

Transfer learning from pre-trained models trained on sufficiently abundant and diverse data is well known to produce state-of-the-art results in tasks related to vision (He et al., 2019; Girshick et al., 2014), Natural Language Processing (NLP) (Radford et al., 2019), and other domains. Pre-training combined with fine tuning to specialize the model for a specific downstream task often leads to better generalization and faster model convergence in the centralized setting (Weiss et al., 2016; Patel et al., 2015).

Federated Learning (FL) is an approach in which a common global model is trained by aggregating model updates computed across a set of decentralized edge devices. It has attracted significant interest recently due to its focus on data privacy. FL literature has largely focused on models trained from scratch (McMahan et al., 2017; Karimireddy et al., 2020; Li et al., 2020) and the impact of data heterogeneity on algorithmic convergence (Li et al., 2020; Acar et al., 2021; Karimireddy et al., 2020). A key consideration in FL is the reduction of communication cost, a strong motivator of the multi-iteration design of the FedAVG algorithm (see e.g., McMahan et al. (2017)). Recently, several studies have been conducted on the effect using pre-trained models on the performance of standard FL algorithms, see e.g., Chen et al. (2023); Nguyen et al. (2023). Here, it was found that besides improving performance, pre-training can help to close the accuracy gap between a model trained in the federated setting and its centrally trained counterpart, particularly in the case of non-i.i.d. client data.

Prior work on transfer learning in the federated setting treats the pre-trained model as a stable initialization for classical FL algorithms that adapt all the model parameters. Approaches from the transfer learning literature demonstrate that it is often more efficient to adapt only parts of the model such as just the last layers (Kornblith et al., 2019), affine parameters (Lian et al., 2022; Yazdanpanah et al., 2022), or adapters (Houlsby et al., 2019). These approaches frequently yield better performance and/or computation time and more easily avoid over-fitting. Although not studied in the prior works, we find that updating only the linear classifier can be highly efficient in the federated setting when starting from a pre-trained model. It allows for both high performance, limited communication cost (since only the linear layer needs to be transmitted), and potentially rapid convergence due to the stability of training only the final layer. Training only the linear classifier in a federated setting can still lead to classical FL problems such as client drift if not treated appropriately. An example of this is illustrated in Nguyen et al. (2023, Appendix C).

We propose a two-stage approach based on first deriving a powerful classification head (HeadTuning stage) and sub-

[1]Concordia University, Montreal, Quebec, Canada [2]Mila Quebec Artificial Intelligence Institute, Montreal, Quebec, Canada [3]McGill University, Montreal, Quebec, Canada [4]Sorbonne University, Paris, France [5]ISIR, Paris, France [6]Centre national de la recherche scientifique (CNRS), Paris, France. Correspondence to: Gwen Legate <gwendolyne.legate@mila.quebec>.

Workshop of Federated Learning and Analytics in Practice, colocated with 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. Copyright 2023 by the author(s).

sequently performing a full fine-tuning of the model (FineTune stage). Such two-stage approaches have been applied in practice and studied theoretically in the transfer learning literature (Kumar et al., 2022; Ren et al., 2023b). They have been shown to give both improved performance in in-distribution and out-of-distribution settings (Kumar et al., 2022). We highlight that the two-stage procedure can lead to many advantages in FL setting:

(a) the fine-tuning stage is more stable under averaging of heterogeneous models

(b) convergence of the fine-tuning stage is rapid (minimizing compute and communication cost)

For the HeadTuning stage, our work highlights the Nearest Class Mean (NCM), a classical alternative to initialize the classification layer which we call FedNCM in the federated case. FedNCM can be computed exactly and efficiently in the federated setting without violating privacy constraints and we will demonstrate that in many cases of interest, using FedNCM to tune the classification head can even outperform approaches considered in prior work with significant communication and computation costs savings.

Our contributions in this work are as follows:

(a) We provide empirical evidence that, for numerous downstream datasets, training only the classifier head proves to be an effective approach in FL settings

(b) We present FedNCM, a straightforward FL method that significantly reduces communication costs when used as a stand alone technique or as an initialization step for HeadTuning which leads to improved accuracy

(c) We demonstrate that employing a two-stage process consisting of HeadTuning (e.g., via FedNCM) followed by fine-tuning results in faster convergence and higher accuracy without violating FL constraints. We further illustrate that it can address many key desiderata of FL: high accuracy, low communication, low computation, and robustness to high heterogeneity while being easier to tune in terms of hyper-parameter selection

2. Methods

2.1. Background and Notation

In FL, distributed optimization occurs over K clients with each client $k \in \{1, ..., K\}$ having data $\mathbf{X}_k, \mathbf{Y}_k$ containing n_k samples drawn from distribution D_k. We define the total number of samples across all clients as $n = \sum_{k=1}^{K} n_k$. The data $\mathbf{X}_k$ at each node may be drawn from different distributions and/or may be unbalanced with some clients possessing more training samples than others. The typical objective function for federated optimization is given in Eq. 1 (Konečný et al., 2016) and aims to find the minimizer of the loss over the sum of the client data:

$$\mathbf{w}^*, \mathbf{v}^* \in \arg\min_{\mathbf{w},\mathbf{v}} \sum_{k=1}^{K} \frac{n_k}{n} \mathcal{L}(g(f(\mathbf{w}, \mathbf{X}_k), \mathbf{v})) \quad (1)$$

In Eq. 1 we have split the model prediction into f, a base parameterized by $\mathbf{w}$ that produces representations, and g, a task head parameterized by $\mathbf{v}$. In this work we will focus on the case where the task head is a linear model, and the loss function, $\mathcal{L}$ represents a standard classification or regression loss. The $\mathbf{w}$ are derived from a pre-trained model and they can be optimized or held fixed.

One approach to obtain the task head while using a fixed $\mathbf{w}$ is to optimize only $\mathbf{v}$ in a federated manner over all the data. In the case that g is given as a linear model and we absorb the softmax into $\mathcal{L}$ this is known as Linear Probing (LP) (Nguyen et al., 2023; Ren et al., 2023a).

2.2. FedNCM Algorithm

An alternative approach to derive an efficient g is through the use of NCM and we note that FedNCM, the federated version of NCM, can be derived exactly addresses many of the critical concerns in the FL setting including privacy, communication, and computation time. Specifically, the server only communicates the pre-trained weights once to each of the clients and clients only communicate once with the server to send back their weighted class means. The server can then use each client's class means to compute exactly the NCM, see Appendix C for the complete algorithm. FedNCM can be used to perform classification directly which corresponds to just the HeadTuning phase of our two-step method, or the class centroids can be used to initialize a linear task head for further fine-tuning, (*ie.* both phases). An algorithm block for FedNCM is provided in Appendix C.

To use NCM as an initialization, consider the cross-entropy loss and $(g \circ f)(\mathbf{x}) = \mathbf{v}f(\mathbf{x}; \mathbf{w}) + \mathbf{b}$. We set the task-head, $\mathbf{v}$ corresponding to the class c logit with the normalized class centroid $\mathbf{l}_c / \|\mathbf{l}_c\|$ and the bias term to 0. This allows us to initialize $\mathbf{v}$ with FedNCM and obtain further improvement through fine-tuning f.

2.3. HeadTune + FineTune

FL algorithms are often unstable due to the mismatch in client objectives leading to large changes and significant deviations between client models after local training. When using a pre-trained model which allows us a powerful initial representation, we argue that a two-stage procedure will improve training stability and converge more quickly. In the first phase (HeadTune) we perform HeadTuning where the parameters of g are updated *e.g.* by linear probing in federated fashion or by using FedNCM. HeadTuning via FedNCM requires only a single forward pass through the data, one model communication to each client and one communication of the centroids back. This incurrs a negligible cost in compute and communication. In the second phase

(FineTune), f and its parameters $\mathbf{w}$ are fine tuned in a federated setting according to the FL objective function specified in Eq. 1. With the negligible cost of communication and compute provided by FedNCM our two-phase approach can have a substantial advantage with respect to convergence when compared to simply fine-tuning (Nguyen et al., 2023; Chen et al., 2023). We also note that our two-phase strategy is compatible with any federated optimization algorithm in the literature.

We now give an intuitive interpretation of the advantages of our method using the framework of Ren et al. (2023a). Assume that the k-th worker is initialized via $\mathbf{w}_0$, and trained locally with SGD for several steps until it reaches parameters $\mathbf{w}_k$. Writing $\mathbf{w}^*$ the optimal parameter, via triangular inequality, we obtain the following inequality:

$$\begin{aligned}&\mathbb{E}_{\mathbf{X}_k}[\|f(\mathbf{w}_k;\mathbf{X}_k)-f(\mathbf{w}^*;\mathbf{X}_k)\|] \leq \\ &\quad \mathbb{E}_{\mathbf{X}_k}[\|f(\mathbf{w}_0;\mathbf{X}_k)-f(\mathbf{w}^*;\mathbf{X}_k)\|+\|f(\mathbf{w}_k;\mathbf{X}_k)-f(\mathbf{w}_0;\mathbf{X}_k)\|]\end{aligned} \tag{2}$$

In the NTK regime, for sufficiently small step size, Ren et al. (2023a) show that the second term depends on the approximation quality of the head g_0 at initialization. This term is bounded for some $c > 0$ by Eq. 3 where σ is the sigmoid activation and $\{\mathbf{e}_i\}_i$ the canonical basis.

$$\begin{aligned}&\mathbb{E}_{\mathbf{X}_k}\|f(\mathbf{w}_k;\mathbf{X}_k) - f(\mathbf{w}_0;\mathbf{X}_k)\| \leq \\ &\qquad c \cdot \mathbb{E}_{(\mathbf{X}_k,\mathbf{Y}_k)}\|\mathbf{e}_{\mathbf{Y}_k} - g_{\mathbf{V}}(f(\mathbf{w}_0;\mathbf{X}_k))\|\end{aligned} \tag{3}$$

This suggests that a good choice of linear head $\mathbf{v}$ will lead to a smaller right hand side term in Eq. 3, and thus reduce the distance to the optimum. Consequently, FedNCM or LP derived $\mathbf{v}$ (compared to a random $\mathbf{v}$) may be expected to lead to a more rapid convergence. Thanks to the initial consensus on the classifier, we expect less client drift to occur, at least in the first round of training, when $\mathbf{v}$ it initialized by HeadTuning, compared to a random initialization.

3. Experiments

We experimentally demonstrate the advantages of FedNCM and the two-stage FedNCM+FT. We also show that LP tuning can at times be more stable and communication efficient than the full fine tuning which is the almost exclusive focus of prior work on FL with pre-trained models. We consider a setting similar to Nguyen et al. (2023) using the CIFAR 10 dataset (Krizhevsky, 2009) and expand our setting to include four additional computer vision datasets Flowers102, CUB, Stanford Cars and Eurosat. Additional details of these datasets as well as a more comprehensive coverage of our approach to measure communication and compute cost is described in Appendix B. Unless otherwise specified, we use SqueezeNet (Iandola et al., 2016), 100 clients, train for 1 local epoch per round. We set client participation to 30% for CIFAR and 100% for all other datasets. Details of hyper-parameter tuning for each method are provided in Appendix F. Data is distributed between clients Following the method of Hsu et al. (2019) using a Dirichlet distribution parameterized by $\alpha = 0.1$. We perform three seeds of each experiment using the FLSim library described in Nguyen et al. (2023). In Appendix E and F we demonstrate other advantages of FedNCM+FT including robustness to larger numbers of clients and hyper-parameters and compatibility with multiple FL optimizers and architectures.

Dataset	**Method**	**Accuracy**	**Total Compute**	**Total Comm.**
CIFAR-10	Random	67.8 ± 0.6	4.5×10^8 F	1.7Tb
	FT Pretrain	85.4 ± 0.4	2.5×10^7 F	10.7 GB
	FedNCM+FT Pretrain	$\mathbf{87.2 \pm 0.2}$	2.5×10^7 F	10.7 GB
	LP Pretrain	82.5 ± 0.2	7.5×10^7 F	149.3 GB
	FedNCM	64.8 ± 0.1	$\mathbf{1 \times F}$	**319 Mb**
CIFAR-10 $\times$32	Random	34.2	4.5×10^8 F	1.7TB
	FT Pretrain	$\mathbf{63.1}$	7.5×10^7 F	149.3 GB
	LP Pretrain	44.7	2.5×10^7 F	10.7 GB
	FedNCM	44.9	$\mathbf{1 \times F}$	**319 Mb**
FLOWERS-102	Random	33.2 ± 0.7	3.7×10^7 F	1.7Tb
	FT Pretrain	64.5 ± 1.0	3.15×10^6 F	149.3 GB
	FedNCM+FT Pretrain	$\mathbf{74.9 \pm 0.2}$	3.15×10^6 F	149.3 GB
	LP Pretrain	74.1 ± 1.2	1.05×10^6 F	10.7 GB
	FedNCM	71.8 ± 0.03	$\mathbf{1 \times F}$	**319 Mb**
CUB	Random	15.0 ± 0.7	2.2×10^8 F	1.7Tb
	FT Pretrain	52.0 ± 0.9	1.9×10^7 F	149.3 GB
	FedNCM+FT Pretrain	$\mathbf{55.0 \pm 0.3}$	1.9×10^7 F	149.3 GB
	LP Pretrain	50.0 ± 0.3	6.3×10^6 F	10.7 GB
	FedNCM	37.9 ± 0.2	$\mathbf{1 \times F}$	**319 MB**

Table 1. Accuracy, total compute and total communication costs of pure HeadTuning methods (below dashed lines) and their counterparts. F is one forward pass of a single sample.

Baseline methods We compare our methods to the following approaches as per Nguyen et al. (2023): (a) *Random*: the model is initialized at random with no use of pre-trained model or NCM initialization. This setting corresponds to the standard FL paradigm of McMahan et al. (2017). (b) *LP*: Given a pre-trained model, we freeze the base and train only the linear head using standard FL optimizer for training. (c) *FT*: A pre-trained model is used to initialize the global model weights and then a standard FL optimization algorithm is applied. (d) *LP and FT Oracles*: These are equivalent baselines trained in the centralized setting that provide an upper bound to the expected performance.

3.1. Efficiency of Pure HeadTuning for FL

As discussed in Sec. 1 tuning the classifier head alone is at times as effective as fine tuning in the context of transfer learning (Evci et al., 2022). In prior work, this situation was briefly considered as a limited case in Nguyen et al. (2023, Appendix C.2) using CIFAR-10. Results suggested that tuning just the linear head (LP) might be a weak approach in the heterogeneous setting but we revisit this claim and expand the scope of these experiments to highlight where LP can be beneficial in terms of performance, communication costs, and compute time. We also show that FedNCM, our approach for approximating a good classifier, can be competitive with orders of magnitude less computation and

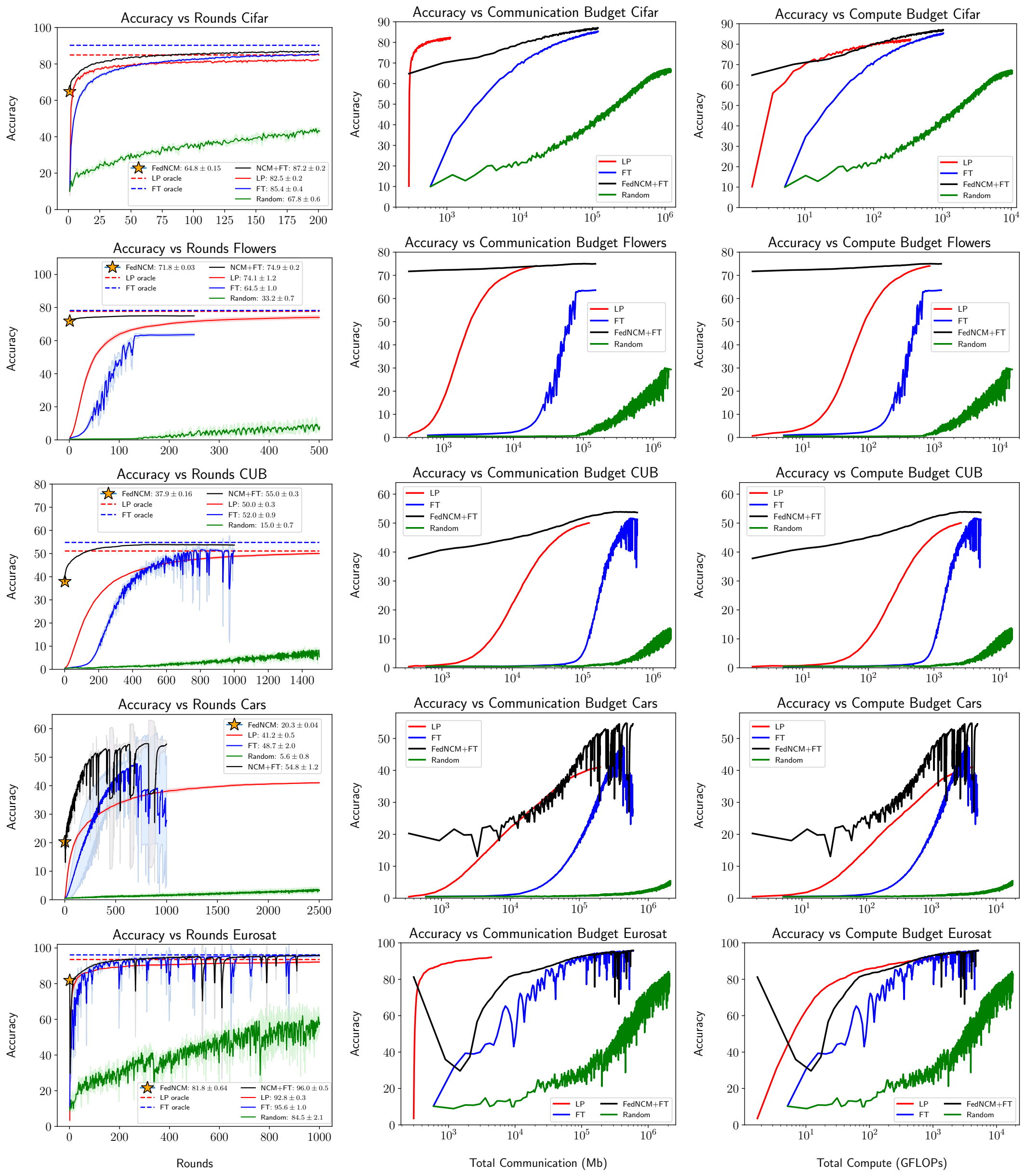


Figure 1. Comparison of rounds vs accuracy as well as accuracy under communication and compute. We observe FedNCM+FT achieves strong accuracies and often has advantages in terms of performance given a communication or compute budget.

communication cost.

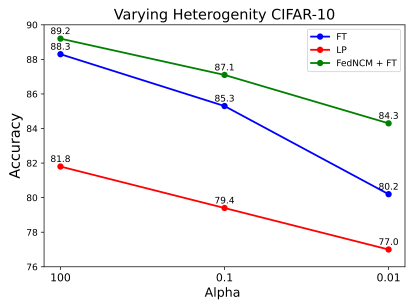
In Nguyen et al. (2023) the CIFAR-10 fine-tuning is done by feeding the 32×32 input image directly into a pre-trained ImageNet model. Since the architectures are adapted to the 224×224 size and trained at this scale originally, this approach can lead to a very large distribution shift and may be sub-optimal for transfer learning. With this in mind, we additionally compare to CIFAR-10 using the traditional approach of resizing the image to the source data (Kornblith et al., 2019; Evci et al., 2022). Tab. 1 shows accuracy, compute, and communication cost results for Pure HeadTuning Methods (FedNCM and LP) as well as full tuning approaches including our FedNCM+FT. In Tab. 1, the CIFAR-10-32×32 row refers to results published in (Nguyen et al., 2023) which we provide for comparison with our results. We observe the significant effect using the model's native input size has on the results: overall accuracy is higher (max 86% for the native CIFAR-10-224×224 vs. 63% for CIFAR-10-32×32) and the gap observed between FT and LP is substantially smaller; an absolute improvement of only 4.6% without image re-sizing vs 18.4% when image re-sizing is done. For both sizes of CIFAR-10 and on CUB, FedNCM can substantially exceed random performance while maintaining a highly competitive compute and communication budget. Experiments with Flowers102 show that FedNCM can already far exceed the difficult-to-train FT setting and furthermore, LP alone exceeds both FedNCM and FT. Our two-phase method of FedNCM+FT outperforms all other methods in terms of accuracy and also allows high efficiency given a specific, potentially limited compute and computational budget. We additionally note that CIFAR-10 contains the same object categories as the original ImageNet dataset while Flowers102 and CUB do not. Flowers102 and CUB therefore represent more realistic transfer learning tasks and under these conditions we observe the effectiveness of HeadTuning (FedNCM or LP) for transfer learning.

3.2. FedNCM then FineTune

We now study in more detail the two-phase approach described in Sec. 2.3. Fig. 1 shows the comparison of our baselines and FedNCM+FT with FedAVG. We show both accuracies versus rounds as well as accuracy given a communication and computation cost budget for CIFAR-10, Flowers102, CUB, Stanford Cars and Eurosat datasets. We observe that when using datasets other than CIFAR-10, LP can converge rather quickly and sometimes to the same accuracy as FT, supporting our argument that phase one HeadTuning methods are very relevant when doing transfer learning in FL scenarios. In phase two additional fine-tuning allows us to refine our performance on the transfer learning dataset. Specifically, when using FedNCM in phase one, achieve a strong starting accuracy due to the FedNCM initialization. In phase FedNCM+FT converges with a better accuracy than FT alone given the same computation budget. This rapid convergence allows FedNCM+FT to be highly efficient under most communication budgets compared to other methods as shown in the second column of Tab. 1. For all the datasets FedNCM+FT is always optimal early on. Fig. 1 shows that for Flowers102, Cars and CUB FedNCM+FT exceeds LP over any communication budget while for CIFAR-10 and Eurosat LP can overtake FedNCM+FT after the early stage. Regardless, FedNCM+FT remains competitive and ultimately reaches higher performance at convergence. FedNCM+FT obtains the best compute for all datasets except for Eurosat where we see an initial decrease from FedNCM although FedNCM+FT quickly recovers to obtain the best compute performance making it the best algorithm for all datasets considered. We note overall as compared to FT the performance improvement can be drastic when considering the trade-off of accuracy as a function of communication and compute available. We also remark that the variance of LP and FedNCM+FT is lower across runs than the FT and Random counterparts. The Random baseline requires longer training to reach convergence and the complete curves of all five datasets are presented in Appendix D.

3.3. Heterogeneity

It is observed by Nguyen et al. (2023) that using a pre-trained model initialization can reduce the effect of system



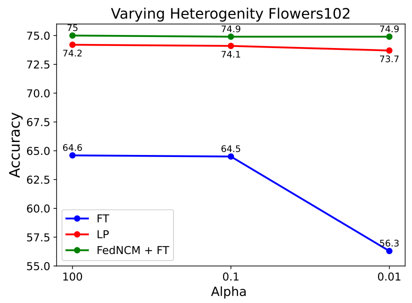


Figure 2. We vary the heterogeneity (Dirichlet-α) for CIFAR-10 and Flowers102. Methods with HeadTuning: LP and FedNCM+FT are more robust, with the substantial advantage of FedNCM + FT increasing in challenging higher heterogeneity.

heterogeneity, we evaluate this by comparing a specific Dirichlet distribution. While the effect of heterogeneity is reduced, we observe that in highly heterogeneous settings we still see substantial degradation in FT as shown in Fig. 3.3. Here we consider for CIFAR-10 the nearly i.i.d. $\alpha = 100$, $\alpha = 0.1$ as considered in Nguyen et al. (2023), and a very heterogeneous $\alpha = 0.01$. Firstly, we observe that FedNCM+FT can provide benefits in the i.i.d. setting. As heterogeneity degrades the naive FT setting sees a large absolute and relative drop in performance. On the other hand, FedNCM+FT as well as LP are able to degrade more gracefully.

4. Conclusion

We highlight the importance of the last layers when undertaking FL from pre-trained models. Based on this observation we suggest a two phase training approach consisting of HeadTuning which focuses on the linear classifier in phase one, followed by Fine-Tuning the entire model in phase two. We propose FedNCM, a federated version of NCM for HeadTuning, which when combined as the two phase FedNCM+FT demonstrates advantages in terms of performance, communication, and computation.

Acknowledgements

We acknowledge support from IVADO Strategic Research Grants on Human Health and Secondary use of data and FQRNT New Scholar grant [E.B.]. We acknowledge resources provided by the digital research alliance and calcul quebec

References

Acar, D. A. E., Zhao, Y., Navarro, R. M., Mattina, M., Whatmough, P. N., and Saligrama, V. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, 2021.

Alyafeai, Z., AlShaibani, M. S., and Ahmad, I. A survey on transfer learning in natural language processing. *arXiv preprint arXiv:2007.04239*, 2020.

Chen, H.-Y., Tu, C.-H., Li, Z., Shen, H.-W., and Chao, W.-L. On the importance and applicability of pre-training for federated learning. 2023.

Davari, M., Asadi, N., Mudur, S., Aljundi, R., and Belilovsky, E. Probing representation forgetting in supervised and unsupervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16712–16721, 2022.

Evci, U., Dumoulin, V., Larochelle, H., and Mozer, M. C. Head2toe: Utilizing intermediate representations for better transfer learning. In *International Conference on Machine Learning*, pp. 6009–6033. PMLR, 2022.

Fallah, A., Mokhtari, A., and Ozdaglar, A. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

He, K., Girshick, R., and Dollar, P. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pp. 2790–2799. PMLR, 2019.

Hsu, T.-M. H., Qi, H., and Brown, M. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size. 2016.

Janson, P., Zhang, W., Aljundi, R., and Elhoseiny, M. A simple baseline that questions the use of pretrained-models in continual learning. *arXiv preprint arXiv:2210.04428*, 2022.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020.

Konečný, J., McMahan, H. B., Ramage, D., and Richtárik, P. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

Kornblith, S., Shlens, J., and Le, Q. V. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2661–2671, 2019.

Krizhevsky, A. Learning multiple layers of features from tiny images. pp. 32–33, 2009. URL `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

Kumar, A., Raghunathan, A., Jones, R., Ma, T., and Liang, P. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022.

Legate, G., Caccia, L., and Belilovsky, E. Re-weighted softmax cross-entropy to control forgetting in federated learning. *arXiv preprint arXiv:2304.05260*, 2023.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. 2020.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Lian, D., Daquan, Z., Feng, J., and Wang, X. Scaling & shifting your features: A new baseline for efficient model tuning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=XtyeppctGgc`.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Nguyen, J., Wang, J., Malik, K., Sanjabi, M., and Rabbat, M. Where to begin? on the impact of pre-training and initialization in federated learning. 2023.

Patel, V. M., Gopalan, R., Li, R., and Chellappa, R. Visual domain adaptation: A survey of recent advances. *IEEE Signal Processing Magazine*, 32(3):53–69, 2015. doi: 10.1109/MSP.2014.2347059.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

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

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Ren, Y., Guo, S., Bae, W., and Sutherland, D. J. How to prepare your task head for finetuning. In *The Eleventh International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=gVOXZproe-e.

Ren, Y., Guo, S., Bae, W., and Sutherland, D. J. How to prepare your task head for finetuning. *arXiv preprint arXiv:2302.05779*, 2023b.

Stich, S. U. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

Wang, J. and Joshi, G. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

Weiss, K., Khoshgoftaar, T. M., and Wang, D. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

Yazdanpanah, M., Rahman, A. A., Chaudhary, M., Desrosiers, C., Havaei, M., Belilovsky, E., and Kahou, S. E. Revisiting learnable affines for batch norm in few-shot transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9109–9118, June 2022.

Yu, H., Yang, S., and Zhu, S. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5693–5700, 2019.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

A. Related work

Federated Learning The best known approach in FL is the FedAvg algorithm proposed by McMahan et al. (2017). For random initialization, convergence of FedAvg and related algorithms has been widely studied for both i.i.d. (Stich, 2018; Wang & Joshi, 2018) and non i.i.d. settings (Karimireddy et al., 2020; Li et al., 2020; Fallah et al., 2020; Yu et al., 2019). A commonly cited problem in FL is the challenge of heterogeneous data and a variety of algorithms have been developed to tackle this (Li et al., 2020; Hsu et al., 2019; Legate et al., 2023; Karimireddy et al., 2020).

Transfer Learning Transfer learning is widely used in domains where data is scarce (Girshick et al., 2014; Alyafeai et al., 2020; Zhuang et al., 2020; Yazdanpanah et al., 2022). A number of approaches for transfer learning have been proposed including the most commonly used full model fine-tuning and last layer tuning (Kornblith et al., 2019) and some more efficient methods such as selecting features (Evci et al., 2022), adding affine parameters (Lian et al., 2022; Yazdanpanah et al., 2022), and adapters for transformers (Houlsby et al., 2019). Transfer learning and the effects of pre-training in FL have so far only been explored in limited capacity. In their recent publication, Nguyen et al. (2023) show that initializing a model with pre-trained weights consistently improves training accuracy and reduces the performance gap between homogeneous and heterogeneous client data distributions. Additionally, in the case where pre-trained data is not readily available, producing synthetic data and training the global model centrally on this has been shown to be beneficial to FL model performance (Chen et al., 2023).

Nearest Class Means Classifier The use of the NCM algorithm in artificial intelligence has a long history. Each class is represented as a point in feature space defined by the mean feature vector of its training samples. New samples are classified by computing the distances between them and the class means and selecting the class whose mean is the nearest. NCM has been widely used in areas such as continual learning, for example Rebuffi et al. (2017); Li & Hoiem (2017); Davari et al. (2022) maintain a memory of exemplars used to compute an NCM classifier. Related to our work, recent literature in continual learning considering pre-trained models were shown to ignore a simple NCM baseline (Janson et al., 2022) capable of outperforming many more complex methods. In our work this NCM baseline, denoted as FedNCM for the federated setting, demonstrates similar strong performance for FL while serving as a very practical first stage of training in our proposed two-step process.

B. Datasets and Evaluation

Details of datasets used in our experimental section. (section 3 are shown in Tab. B.

Dataset	Num. Classes	Num. Images
CIFAR-10	10	50000
Flowers102	102	1020
Stanford Cars	196	8144
CUB	200	5994
EuroSAT-Sub	10	5000

Summary of datasets used in our experiments.

Communication and Computation Budget Communication and computation costs of each proposed method are considered both in total and given a fixed budget for either communication or computation. For the communication costs we assume each model parameter transmitted is done so via a 32-bit floating point number, allowing us to compute the total expected communication between clients and server. We emphasize that linear probing only requires that we send client updates for the classifier rather than the entire model as is the case in the other settings; consequently LP has much lower communication costs when compared to FT for any given number of rounds. Our proposed FedNCM is a one-round algorithm incurring even lower communication costs than any other algorithm considered.

For computation time we consider the total FLOPs executed on the clients. We assume for simplicity that the backward pass of a model is $2\times$ the forward pass. For example, in the case of LP, each federated round leads to one forward communication on the base model, f, and one forward and one backward (equivalent to two forward passes) on the head, g. Similarly, for FedNCM the communication cost consists only one forward pass through the data.

C. FedNCM Algorithm

Algorithm 1 outlines the steps in our proposed FedNCM method.

Algorithm 1 FedNCM. K is the total number of clients, C is the number of classes in the training dataset, D_c is the total number of samples of class c

```
Require: $(\mathbf{X}_1, \mathbf{Y}_1), (\mathbf{X}_2, \mathbf{Y}_2), \ldots, (\mathbf{X}_K, \mathbf{Y}_K)$ - Local datasets, $w_{pt}$ - pre-trained model
    Server Executes:
 1: for each client $k \in K$ in parallel do
 2:     $[m_c^k]_{c \in C} \leftarrow$ LocalClientStats($X_k, Y_k, \mathbf{w}_{pt}$)      ▷ Send to all clients, receive weighted class means
 3: end for
 4: for each class $c \in C$ do
 5:     $\mathbf{l}_c \leftarrow \frac{1}{D_c} \sum_{k=1}^{K} m_c^k$      ▷ $\mathbf{l}_c$ can be used in NCM classifier
 6: end for

    Client Side:
 7: function LocalClientStats($\mathbf{X}, \mathbf{Y}, \mathbf{w}$)
 8:     for each class $c \in N$ do
 9:         Let $\mathbf{X}_c = \{x_i \in X, y_i = c\}$
10:         $m_c \leftarrow \sum_{x \in X_c} f_w(x)$
11:     end for
12:     return $[m_c]_{c \in C}$
13: end function
```

Dataset	Algorithm	FedNCM	FedNCM + FT	FT+Pretrain
CIFAR-10	FedAvg	64.8 ± 0.1	87.2 ± 0.2	85.4 ± 0.4
	FedADAM	64.8 ± 0.1	$\mathbf{89.4 \pm 1.1}$	88.2 ± 0.2
Flowers102	FedAvg	71.8 ± 0.03	74.9 ± 0.2	64.5 ± 1.0
	FedADAM	71.8 ± 0.03	$\mathbf{76.7 \pm 0.2}$	66.6 ± 1.0

Table 2. Model performance with different methods for a variety of FL algorithms, for FedAvg and FedADAM. FedNCM+FT outperforms in all cases.

D. Extended Accuracy Comparison Figures

Figure 3 is the extended version of Figure 1 in the main body of the paper. In the paper we truncate the number of round displayed for the random setting since random requires many more rounds to converge than the other methods. Figure 3 shows these same figures with the entirety of the training rounds displayed for the random setting.

E. Ablations

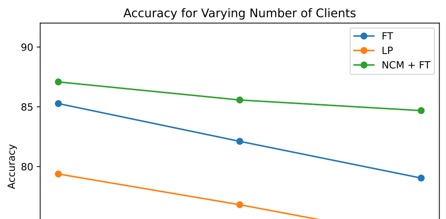
We perform a number of ablations, demonstrating other advantages of FedNCM+FT. We show robustness to larger numbers of clients, insensitivity to hyperparameters, and compatibility with multiple FL algorithms and architectures.

E.1. Choice of FL Algorithm

So far we have focused on FedAvg, since our method is compatible with any FL optimizer we further analyze FedNCM+FT for the case of FedAdam which obtained some of the higher performances in Nguyen et al. (2023). We observe that improved FL optimizers can complement FedNCM+FT which can systematically exceed FT. Even with this improved method, FedNCM which does not require an optimizer, continues to exceed the performance of FT on Flowers102. This result suggests the considerations of the FL optimization algorithm (Nguyen et al., 2023) are not always the most critical for optimal performance.

E.2. Varying the Local Epoch

The number of local epochs can drastically affect FL algorithms, typically a larger amount of local computation between rounds is desired to minimize communication. However, this can often come at a cost of degraded performance. We observe in Fig. 4 as in Nguyen et al. (2023) that FT can be relatively



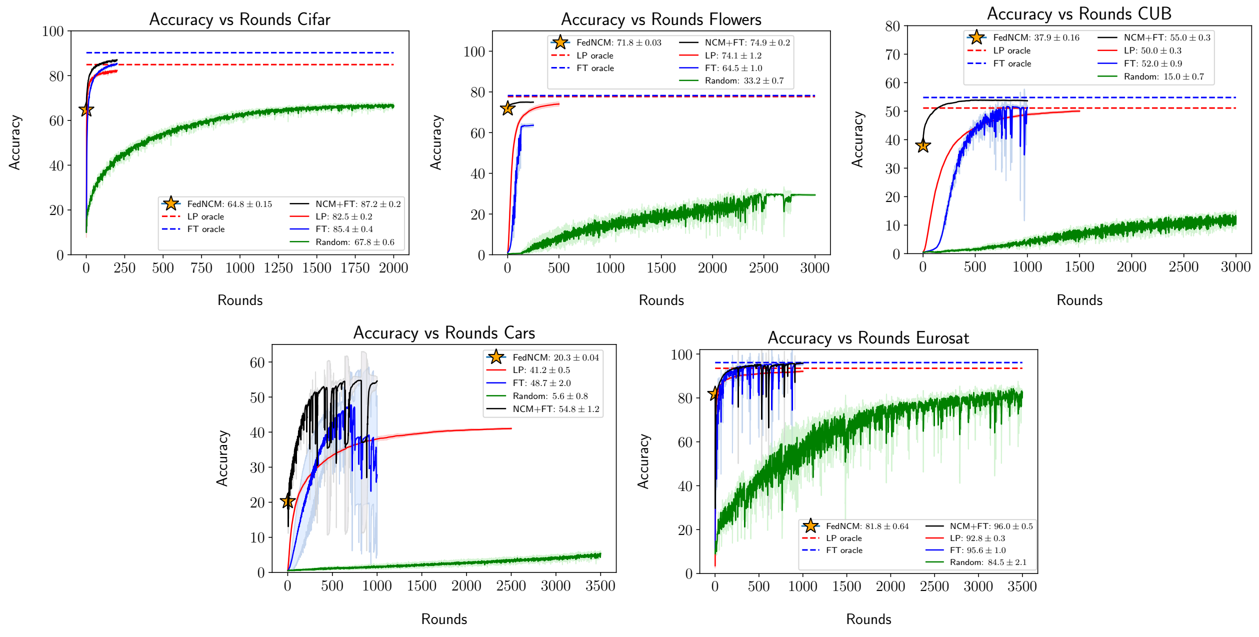


Figure 3. The full training of Random baseline corresponding to Figure 1 in the paper is shown. We observer Random is always very far from the other baseliens and converges slowly.

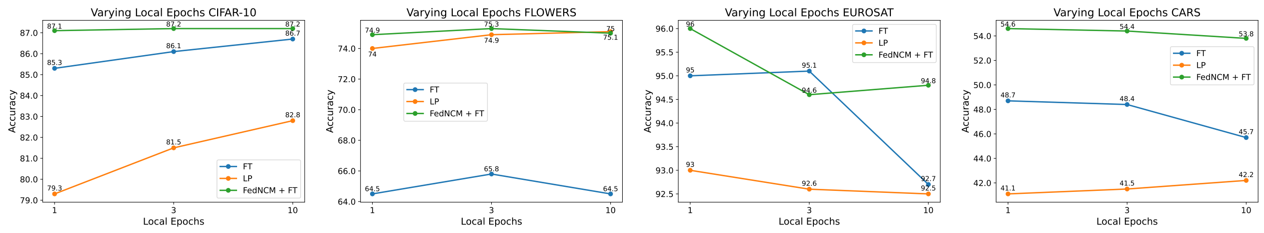


Figure 4. We vary the number of local epochs. FedNCM+FT always outperforms FT and nearly always LP in this challenging setting.

robust in some cases (CIFAR-10) to increasing local epochs. However, we also observe for some datasets that it can degrade, while LP and FedNCM+FT are less likely to degrade. Overall FedNCM+FT continues to outperform for larger local epochs.

E.3. Increasing clients

Tab. 5 shows that as we increase the number of clients we observe that the degradation of FedNCM+FT is less severe than both LP and FT, suggesting it is stable under a large number of workers being averaged. As discussed in Sec. 2.3 it is expected in the same round that a representation would shift less from a starting point, and therefore since the starting point is the same for all clients, we expect the client drift within a round to be less given a fixed update budget.

E.4. ResNet18 Experiments

We perform experiments for LP, FT, FedNCM and FedNCM+FT using the ResNet18 model using the FedAvg algorithm and the CIFAR-10 and Flowers102 datasets. Results are summarized in Table E.4 where we observe that FedNCM performance is better by almost 13% compared to squeezenet, while FT performance is degraded

Dataset	FedNCM	FedNCM + FT	FT+Pretrain	LP+Pretrain
CIFAR-10	77.74 ± 0.05	79.05 ± 1.31	77.87 ± 4.07	74.73 ± 3.03
FLOWERS102	74.13 ± 0.31	74.1 ± 0.26	34.41 ± 10.16	25.35 ± 2.59

Table 3. ResNet18 model performance for FedAvg. As with Squeezenet, FedNCM+FT continues to outperforms in all cases.

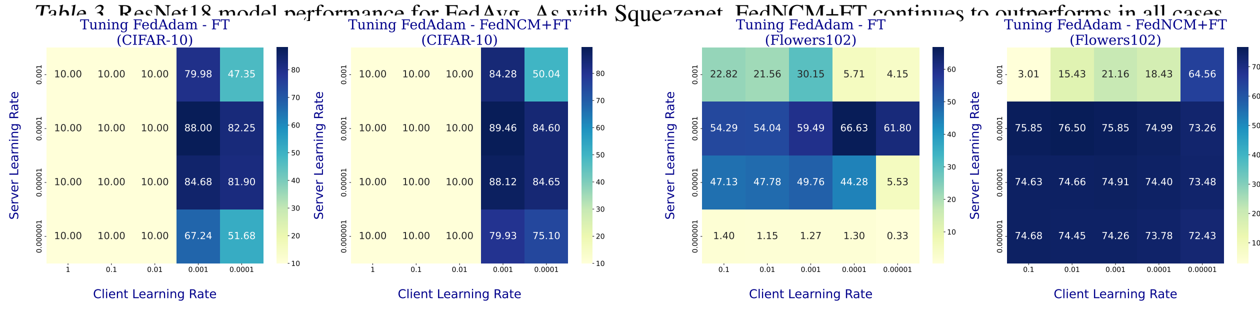


Figure 6. Hyperparameter grids for FedAdam for CIFAR-10 FT, FedNCMFT (left) and Flowers (right). We observe CIFAR-10 FedNCM-FT tends to do better or equal for all hyperparameters compared to FT. For Flowers it is much easier to tune, achieving strong values over a wide range, a noticeable advantage in FL

compared to squeezenet. We hypothesis this is due to the challenges of deeper networks in heterogenous federated learning. For the Flowers102 dataset, FedNCM and FedNCM+FT produce the best results by far. Additionally, for flowers FedNCM outperformed all other methods. The variance between runs using ResNet18 is much higher than was observed for SqueezeNet, FedNCM appears to help stabilize the results since it provides the most consistency by for both datasets.

F. Hyperparameter Tuning

We follow the approach of Nguyen et al. (2023); Reddi et al. (2020) to select the learning rate for each method on the various datasets. For CIFAR-10 and SqueezeNet experiments we take the hyperparameters already derived in Nguyen et al. (2023).

For CIFAR-10, CUB, Stanford Cars and Eurosat datasets the learning rates for the FedAvg algorithm were tuned via a grid search over learning rates $\{0.1, 0.07, 0.05, 0.03, 0.01, 0.007, 0.005, 0.003, 0.001\}$. For Flowers102, based on preliminary analysis we used lower learning rates were tuned over learning rates
$\{0.01, 0.007, 0.005, 0.003, 0.001, 0.0007, 0.0005, 0.0003, 0.0001\}$.

Prior work on federated learning with pre-trained models has indicated that for FedADAM lower global learning rates and higher client learning rates were more effective. As a result for CIFAR-10 and Flowers the client learning rate was tuned over $\{1, 0.1, 0.01, 0.001, 0.0001\}$ and the server learning rate was tuned over $\{0.001, 0.0001, 0.00001, 0.000001\}$, each combination of server and client learning rates were tried. For

F.1. Ease of Hyperparameter tuning

FL algorithms are known to be challenging for hyperparameter selection (Reddi et al., 2020) and this can affect their practical application. We first note that FedNCM does not have any hyperparameters which already provides a large advantage. In Fig. 6, we observe the final performance for a grid search over a range of server and client learning rates for FedAdam using both FT and FedNCM+FT. We observe that FedNCM+FT not only has higher performance but it is also more stable over the entire hyperparameter grid on Flowers dataset, and outperforms for all settings on CIFAR-10.

G. Compute

We use a combination of NVIDIA A100-SXM4-40GB, NVIDIA RTX A4500, Tesla V100-SXM2-32GB and Tesla P100-PCIE-12GB GPUs for a total of 1.1 GPU years . In addition to the experiments reported in the paper, this includes preliminary experiments and hyperparameter searches.