FED UP WITH COMPLEXITY: SIMPLIFYING MANY-TASK FEDERATED LEARNING WITH NTKFEDAVG

Aashiq Muhamed, Meher Mankikar, Virginia Smith
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{amuhamed, mmankika, smithv}@andrew.cmu.edu

ABSTRACT

Recent work has introduced the challenging setting of many-task federated learning (MaT-FL), which considers a scenario in which each client in a federated network may solve a separate learning task. Unfortunately, existing methods addressing MaT-FL, such as dynamic client grouping and split FL, increase privacy risks and computational demands by maintaining separate models for each client or task on the server. We introduce a novel baseline for MaT-FL, NTKFedAvg, that leverages a unified multi-task model on the server and the Neural Tangent Kernel (NTK) linearization to accommodate task heterogeneity without client or task specific model adjustments on the server. This approach enhances privacy, reduces complexity, and improves resistance to various threats. Our evaluations on two MaT-FL benchmarks demonstrate that NTKFedAvg surpasses FedAvg in mIoU and accuracy, achieves faster convergence, is competitive with existing baselines, and excels in task unlearning in fewer rounds. This work not only proposes a more efficient and potentially privacy-preserving baseline for MaT-FL but also contributes to the understanding of task composition and weight disentanglement in FL, offering insights into the design of FL algorithms for environments characterized by significant task diversity.

1 INTRODUCTION

In real-world cross-silo federated learning, the assumption that all clients work on the same set of tasks is often unrealistic. This is due to varying data distributions among clients and the inherent heterogeneity in their tasks. Such diversity is natural because (1) the process of obtaining task labels is often laborious and costly, limiting clients' access to all labels, and (2) in many real-world situations, clients typically specialize in specific tasks. For instance, self-driving car companies usually use either LIDAR or stereo cameras to gather sensing data. LIDAR data is better for depth perception, while stereo camera data excels in edge detection and object detection. Consequently, these companies might focus on fine-tuning tasks that are more easily solved with the type of data they have, exemplifying task heterogeneity. This phenomenon, known as *task heterogeneity*, represents a novel and under-explored form of heterogeneity in federated learning (FL), adding complexity to the learning process. Many-task federated learning (MaT-FL), as outlined by Cai et al. (2023), addresses FL under task heterogeneity by enabling local clients to collaborate effectively, despite specializing in different tasks. This approach can lead to broader data coverage and more efficient utilization.

While Federated Averaging (FedAvg) (McMahan et al., 2016) has served as a foundational algorithm in FL, its efficacy diminishes in MaT-FL, where the nonlinear characteristics of modern deep-learning networks and task interference significantly influence performance (Cai et al., 2023). Existing methods for MaT-FL include a dynamic grouping mechanism, allowing each client to select a similar neighborhood of clients for subsequent adaptive aggregation (Cai et al., 2023; Lu et al., 2023), and split FL, wherein the server sustains numerous unique models, allocating one per client (Chen et al., 2023). These strategies necessitate the server managing individual models for each client or task, thereby heightening the risk of privacy breaches for any given client and amplifying

the complexity of implementing secure aggregation. Furthermore, they augment both storage and computational demands on the server.

In this work, we propose a new baseline for MaT-FL that involves the server maintaining a single multi-task model, as opposed to separate models for each client or task. Additionally, our method avoids any form of clustering or aggregation that necessitates the server to store client-specific weights. While our approach does not formally quantify privacy improvements using notions such as differential privacy, it facilitates secure aggregation and potentially provides better resistance to threats like membership inference, model inversion, and attribute inference by operating with a unified model architecture. As FedAvg is ineffective in MaT-FL scenarios, we draw inspiration from the model merging literature, specifically the concept of task arithmetic (Ilharco et al., 2021). This concept involves manipulating task vectors within the weight space of a pre-trained model to enhance performance on specific tasks. Our approach adapts the idea of task vectors to the sequential training dynamics of FL, in contrast to the one-shot model merging techniques found in the literature. We introduce a novel technique that integrates Neural Tangent Kernel (NTK) linearization with task vectors in a federated aggregation framework, called NTKFedAvg. This technique modifies the federated training behavior of deep neural networks, promoting better weight disentanglement and proving exceptionally effective in environments characterized by significant task heterogeneity.

Our empirical evaluations on two MaT-FL benchmarks reveal that NTKFedAvg not only outperforms FedAvg but is also competitive with other MaT-FL baselines. Specifically, NTKFedAvg achieves an 8% higher mIoU and a 6% greater accuracy compared to FedAvg on the Pascal-Context and Clip datasets, respectively, in less than 10 rounds. Furthermore, NTKFedAvg demonstrates an enhanced ability for client task unlearning, effectively reducing performance on a specified task (MNIST) to zero within four rounds while maintaining performance on other tasks. The contributions of our work include:

- We propose a new baseline for MaT-FL where a single multi-task model is maintained by the server to potentially enhance privacy. This is supported by our FL algorithm, NTKFedAvg, designed to promote weight disentanglement and improve task composition.
- Across two MaT-FL benchmarks, we show that NTKFedAvg outperforms FedAvg in terms of communication efficiency and convergence speed, and is competitive with existing MaT-FL baselines. Additionally, it excels at task unlearning, requiring fewer communication rounds for effective unlearning compared to FedAvg.
- We explore the impact of key design decisions within our framework through ablations, focusing on local work, linearization strategies, and proximal losses, to refine our understanding of MaT-FL.

2 Methodology

2.1 Many Task Federated Learning

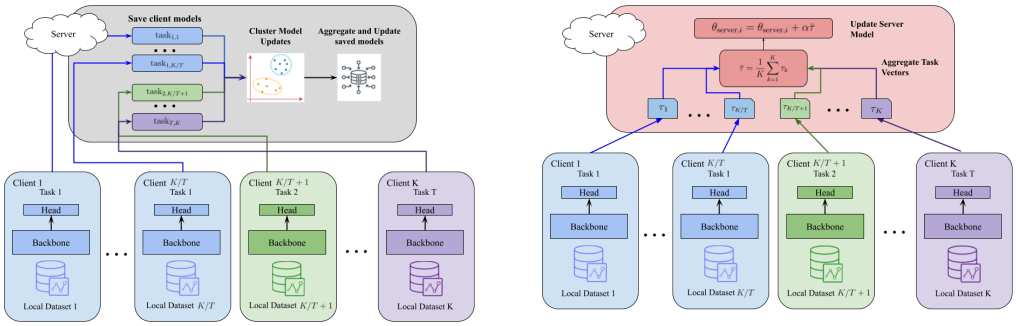

Figure 1: **(Left)** Existing works in MaT-FL often store a server-side model per task/client, which is inefficient and susceptible to model inversion attacks. **(Right)** In our approach NTKFedAvg, clients communicate task vectors which are aggregated at the server to maintain a single global model. This approach enhances efficiency and security.

Many-Task Federated Learning (MaT-FL) is a recently proposed federated learning setting designed to address *task heterogeneity*. In this setting, local clients need to collaborate effectively despite be-

ing specialized in different tasks (Cai et al., 2023). We denote an MaT-FL system with K total clients and T tasks, where each client specializes in one task t from the set of total tasks $t_1, t_2, \cdots, t_T$. In the existing body of work on MaT-FL, the server maintains a distinct model for each task, $t \in [T]$, (Cai et al., 2023) or a distinct, general backbone for each client, $i \in [K]$ (Chen et al., 2023), as illustrated in Figure 1 (Left). The server employs weight aggregation strategies such as dynamic grouping and adaptive aggregation to update these task-specific models and then communicates the updated model back to the client performing that task. However, dynamic grouping and maintaining task-specific models on the server increase the risk of privacy attacks. Managing separate models for each task or client amplifies the system's susceptibility to privacy breaches. Model inversion attacks (Shokri et al., 2016), membership inference attacks (Jia & Gong, 2018), and attribute inference attacks (Fredrikson et al., 2015) become more feasible when models are not securely aggregated, as each model's fine-tuned weights to its unique dataset can provide adversaries with more direct clues about the underlying data.

In this work, we revisit the question of whether it is necessary to maintain a separate model for each task on the server and propose a novel baseline in which the server maintains a single multi-task model, rather than individual models for each task or client. This configuration is illustrated in Figure 1 (Right). After local training, each client i generates a task vector by calculating the difference between the checkpoint at the start of the round and the fine-tuned checkpoint, and then communicates this difference to the server. The server aggregates the task vectors from all clients and updates its single central multi-task model accordingly. This method offers additional protection against privacy attacks. By foregoing dynamic grouping and adaptive aggregation, we enable the implementation of secure aggregation for the task vectors of clients, thus concealing the contributions of individual clients. Dynamic grouping, on the other hand, can expose the weights of clients in smaller clusters and may also reveal the identities of cluster centers. In addition to privacy benefits, employing a single server model simplifies model management overhead on the server side. This approach eliminates the need for tracking and updating task-specific models, thereby reducing computational and storage requirements. By maintaining a single multi-task model, our framework promotes the learning of shared representations that generalize across tasks, which can mitigate the risk of overfitting and enhance the model's robustness to privacy attacks.

2.2 NTKFedAvg For Many Task Federated Learning

In the MaT-FL setting, FedAvg (McMahan et al., 2016) is not effective for optimizing a single multi-task server model (Cai et al., 2023). This inefficacy is attributed to the phenomenon of weight entanglement, where distinct directions in the model's weight space fail to map directly to task-specific regions within the input space. To address this challenge, our approach draws upon insights from the domain of model merging, aiming to enhance the disentanglement of task-related features. We demonstrate that such disentanglement not only improves the performance of task aggregation algorithms but also facilitates the process of federated task unlearning. Our methodology incorporates task vectors and the application of neural tangent kernel (NTK) linearization techniques into the FL paradigm, thereby proposing a more refined and effective framework for handling multiple tasks simultaneously.

Task Vector and Task Arithmetic: A task vector τ_t for task t is defined as $\tau_t = \theta_t^* - \theta_0$, where θ_t^* and θ_0 are the fine-tuned and pre-trained model weights, respectively. The network f demonstrates task arithmetic when $f(x; \theta_0 + \sum \alpha_t \tau_t)$ yields $f(x; \theta_0 + \alpha_t \tau_t)$ for $x \in D_t$, and $f(x; \theta_0)$ otherwise, with $T = \{\tau_t\}$ and distinct task supports $D = \{D_t\}$.

Weight Disentanglement: Weight disentanglement occurs if $f(x; \theta_0 + \sum \alpha_t \tau_t) = \sum g_t(x; \alpha_t \tau_t) + g_0(x)$, with $g_t(x; \alpha_t \tau_t)$ and $g_0(x)$ being non-zero only within their respective task supports. It was recently shown to be a necessary condition for task arithmetic (Ortiz-Jimenez et al., 2023).

Neural Tangent Kernel Linearization: NTK approximates a neural network's function via a first-order Taylor expansion around initialization weights θ_0, given by $f(x; \theta) \approx f(x; \theta_0) + (\theta - \theta_0)^\top \nabla_\theta f(x; \theta_0)$, defining a linear relationship in neural tangent space. It was empirically observed in Ortiz-Jimenez et al. (2023) that NTK linearization promotes weight disentanglement.

The proposed method, NTKFedAvg, is shown in Algorithm 1. This approach leverages the principles of task vectors and NTK linearization in that each client's model corresponding to a task, is linearized to ensure operation within a linear regime. Client task vectors, computed post-local

Algorithm 1 NTK-FedAvg for Federated Learning and Federated Unlearning

Initialize global model weights θ_0, learning rate α
for each round $i = 1, 2, \ldots, R$ **do**
  Initialize server weights for round i, $\theta_{\text{server},i}$
  for each client $k = 1, 2, \ldots, K$ in parallel **do**
    Linearize local model around $\theta_{\text{server},i}$ using NTK
    Perform local training to obtain updated weights $\theta_{k,i}$
    Compute task vector $\tau_{k,i} = \theta_{k,i} - \theta_{\text{server},i}$
    if task is to be negated for client k **then**
      $\tau_{k,i} = -\tau_{k,i}$
    end if
  end for
  Aggregate task vectors: $\bar{\tau}_i = \frac{1}{K} \sum_{k=1}^{K} \tau_{k,i}$
  Update global model $\theta_{\text{server},i+1} = \theta_{\text{server},i} + \alpha \bar{\tau}_i$
  Distribute $\theta_{\text{server},i+1}$ to clients
end for

training at every round, are communicated to the server and are either aggregated (for federated learning) or negated (for federated unlearning) to update the global model. In general, thanks to the efficiency of the Jacobian-vector product implementations in most deep learning frameworks, training and inference in linearized neural networks only require an O(1) increase in computational costs with respect to their non-linear counterparts.

3 Experiments And Results

3.1 Dataset, Models, and Metrics

Pascal-Context setting Clients are allocated one of five tasks from Pascal-Context (Mottaghi et al., 2014), including semantic segmentation, saliency estimation, surface normals estimation, edge estimation, and human part segmentation, employing a ResNet18 backbone and deeplab decoder (Chen et al., 2017). We compare NTKFedAvg with DGAgg (Cai et al., 2023), where clients dynamically group based on similarity for weight aggregation. For NTK-FedAvg, we linearize the ResNet backbone and keep the decoder frozen. We train using lr $1e-7$ and 10 rounds, and evaluate with mean Intersection over Union (mIoU) or angular Root Mean Squared Error (RMSE).

CLIP setting We focus on contrastively pretrained vision-language models, using OpenCLIP (Ilharco et al., 2021) with a ViT-B/32 encoder, pretrained on 34B samples from DataComp-1B at 256px resolution. Eight clients are fine-tuned on tasks defined in Ilharco et al. (2022); Ortiz-Jimenez et al. (2023), including Cars (Krause et al., 2013), DTD (Cimpoi et al., 2014), SUN397 (Xiao et al., 2016), EuroSAT (Helber et al., 2019), GTSRB (Stallkamp et al., 2011), MNIST (LeCun, 1998), SVHN (Netzer et al., 2011), and RESISC45 (Cheng et al., 2017). The ViT image encoder is linearized, and classification layers for the text encoder are frozen, only communicating CLIP backbone parameters. Normalization precedes inner product operations with text embeddings, and average absolute accuracy (%) on a server-held test set is used for evaluation.

Each model configuration is trained on 2 A6000 GPUs, in under 96 GPU hours. We aim to experiment in a setting where communication is extremely limited. We evaluate the performance of these models across rounds 1 to 10, given that FL incurs significant communication overhead, making this range of rounds a realistic scenario for practical applications.

3.2 PascalContext Setting Results

Fig 2 compares FedAvg and NTKFedAvg averaged across 3 tasks: semantic segmentation (Seg.), saliency estimation (Sal.), and human parts estimation (H. Parts). In the standard setting, each client is trained for 10 local epochs, while in the "local work" setting, each client is trained for 50 local epochs. All experiments were run with batch size 16 and lr $1e-7$. We see that in the standard setting with little local work, NTKFedAvg is comparable to FedAvg. However, when we increase

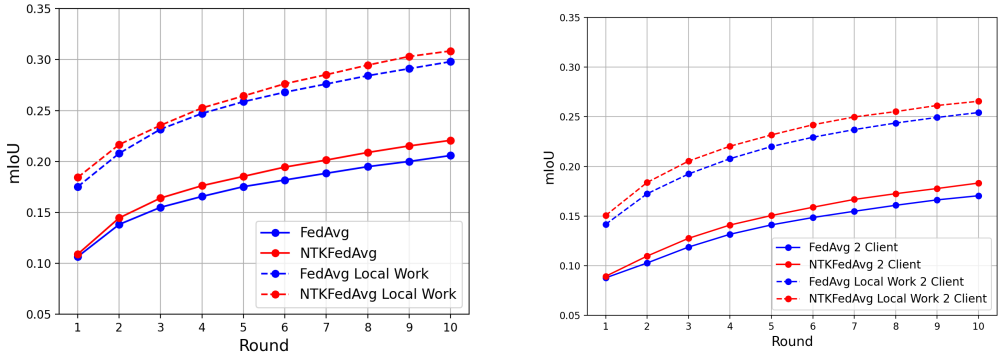

Figure 2: **(Left)** Averaged performance of Seg., Sal., and H.Parts estimation with one client per task, NTKFedAvg performs better than FedAvg at fewer rounds with sufficient local work. **(Right)** With two clients per task, NTKFedAvg continues to perform better than FedAvg at every round.

# Clients/Task	Work Type	Method	Seg. (mIoU)	Sal. (mIoU)	H. Parts (RMSE)
1	Standard	FedAvg	0.0898	0.4022	0.1255
		NTKFedAvg	**0.1025**	**0.4253**	**0.1344**
	Local Work	DGAgg	0.1980	**0.5083**	0.2330
		FedAvg	0.1963	0.4792	0.2183
		NTKFedAvg	**0.2025**	0.4892	**0.2337**
2	Standard	FedAvg	0.0416	0.3740	0.0959
		NTKFedAvg	**0.0496**	**0.3979**	**0.1023**
	Local Work	DGAgg	**0.1454**	0.4537	0.1804
		FedAvg	0.1454	0.4492	0.1681
		NTKFedAvg	0.1405	**0.4653**	**0.1910**

Table 1: Max metrics of DGAgg, FedAvg, NTKFedAvg in Standard/Local Work settings on Pascal-Context dataset over 10 epochs.

the amount of local work, NTKFedAvg is more effective at task merging, and outperforms FedAvg from rounds 1 through 10.

Table 1 presents a comparison of DGAgg, FedAvg, and NTKFedAvg under both standard and "local work" settings, with configurations of 1 and 2 clients per round. NTKFedAvg consistently surpasses FedAvg across these setups. Notably, in the local work scenario with 2 clients per task, NTKFedAvg outperforms FedAvg in the initial rounds (1 to 7), highlighting its greater efficiency in contexts with limited communication rounds, despite FedAvg achieving a higher maximum mIoU. Additionally, NTKFedAvg beats DGAgg in two out of three tasks in the local work setting, illustrating the effectiveness of linearization for task disentanglement without the need for task-specific global models. The transition from 1 to 2 clients per task leads to a performance decline due to a more decentralized FL environment with reduced data per client. However, NTKFedAvg shows enhanced resilience to this decentralization effect, with a smaller performance drop (13.90%) compared to FedAvg (14.67%), suggesting NTKFedAvg is more robust in MaT-FL settings.

3.3 CLIP Setting Results

Centralized and single task baselines Figure 3 (Left) illustrates the average performance of both centralized (best model within 25 epochs) and single-task models (Ilharco et al., 2021) across all tasks. The nonlinear single-task models demonstrate superior performance compared to their linearized counterpart, which surpass the performance of the multi-task centralized models. Within the centralized framework, the nonlinear model exhibited better performance than the linearized model, a result that may be attributed to the higher representational capacity of nonlinear networks. Train-

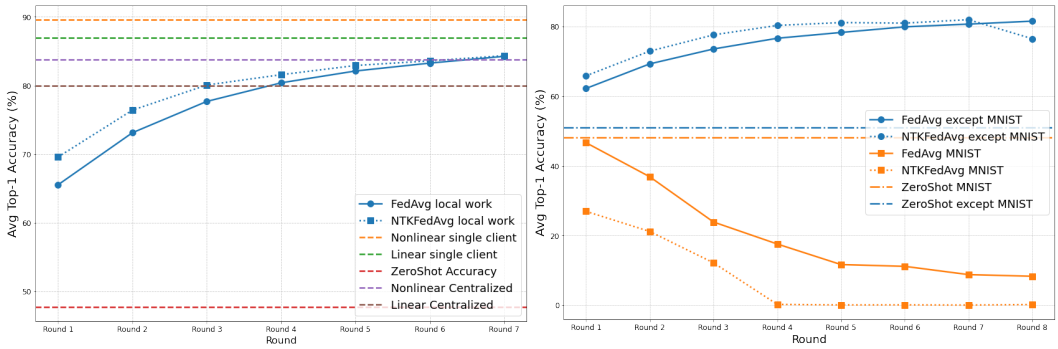

Figure 3: **(Left)** Federated Learning: NTKFedAvg performs better than FedAvg at fewer rounds **(Right)** Federated Unlearning: NTKFedAvg is more effective at unlearning than FedAvg.

ing and tuning hyperparameters in multi-task centralized models however is challenging, primarily due to task interference, variations in task loss scales, and imbalanced batches (Crawshaw, 2020).

FedAvg vs NTKFedAvg In Appendix Fig. 4, we compare FedAvg with $\alpha = 1$ against NTKFedAvg over 10 rounds, training client models for a single epoch per round. Initially, NTKFedAvg experiences a slight performance decline in the first round, particularly on the Cars task, suggesting adaptation challenges. However, NTKFedAvg exceeds FedAvg from rounds 2 through 6, indicating its early-stage learning advantages—critical for FL systems focused on reducing communication costs via fast convergence.

Local work: In "local work" configuration, each client model is trained to near convergence in every round across diverse tasks. Specifically, "Cars" undergoes 35 epochs of training, "DTD" 76 epochs, "EuroSAT" 12 epochs, "GTSRB" 1 epoch, "MNIST" 5 epochs, "RESISC45" 15 epochs, "SUN397" 14 epochs, and "SVHN" 4 epochs. In Figure 3 (Left), with this configuration, NTKFedAvg exhibits a consistent lead of nearly 4% in absolute accuracy with FedAvg from the first round and maintains this advantage for approximately 7 rounds. This difference highlights the limitations of FedAvg in integrating local updates within large pre-trained models when compared to NTKFedAvg that promotes weight disentanglement.

Federated Unlearning We also perform Federated Unlearning experiments, where the goal is to unlearn a client task while retaining the performance on all other tasks. This is done by subtracting the task vector of the client whose task we need to unlearn before aggregation on the server. Fig 3 (Right) shows how NTKFedAvg with local work is much more effective at unlearning MNIST. Unlike FedAvg, with NTKFedAvg, the performance on MNIST decays to zero in 5 rounds, while retaining performance on other tasks. In Appendix A.5, we observe that centralized training requires about 22 epochs to unlearn a task, with gradient methods.

Linearization point Algorithm 1 suggests we can linearize NTKFedAvg in two ways. We can either linearize the local model around $\theta_{server,i}$ which is the global model after aggregation every round, or we can continue linearizing about the original pretrained model θ_0. In Appendix Fig 5 we find that updating the linearization point about the global model every round is more effective when each client is trained for 1 epoch/round, so we continue using this strategy for other experiments.

FedProx vs. NTKFedProx As FedProx (Li et al., 2020) was shown to be a competitive baseline for multi-task federated learning in Cai et al. (2023), we additionally experiment with a proximal loss in the *local work* configuration at every round. The proximal loss is $\frac{\mu}{2} \|\theta_{k,i} - \theta_{\text{server},i}\|^2$ and we set $\mu = 1$. Appendix Fig 8 shows our results with local work. We find that the proximal loss reduces the average accuracy of both FedProx and NTKFedProx relative to FedAvg and NTKFedAvg, but NTKFedProx outperforms FedProx.

4 Conclusion, Limitations and Future Work

This work introduces NTKFedAvg for MaT-FL, emphasizing its privacy advantages, communication efficiency and quick convergence, especially with large pretrained models in diverse tasks. Limitations include the need for heuristics in sequential linearization and lack of theoretical analysis of the

conditions on tasks/datasets where it outperforms FedAvg. Future work will focus on optimizing the linearization initiation point in federated contexts, comparing various optimizers in federated multi-task settings, and assessing effectiveness amid varying data and task heterogeneity.

REFERENCES

Ruisi Cai, Xiaohan Chen, Shiwei Liu, Jayanth Srinivasa, Myungjin Lee, Ramana Kompella, and Zhangyang Wang. Many-task federated learning: A new problem setting and a simple baseline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5036–5044, 2023.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017.

Yiqiang Chen, Teng Zhang, Xinlong Jiang, Qian Chen, Chenlong Gao, and Wuliang Huang. Fedbone: Towards large-scale federated multi-task learning, 2023.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017. doi: 10.1109/JPROC.2017.2675998.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

Michael Crawshaw. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*, 2020.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015. URL https://api.semanticscholar.org/CorpusID:207229839.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. If you use this software, please cite it as below.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *International Conference on Learning Representations*, 2022. doi: 10.48550/arXiv.2212.04089.

Jinyuan Jia and N. Gong. Attriguard: A practical defense against attribute inference attacks via adversarial machine learning. *USENIX Security Symposium*, 2018.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, June 2013.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Yuxiang Lu, Suizhi Huang, Yuwen Yang, Shalayiding Sirejiding, Yue Ding, and Hongtao Lu. Towards hetero-client federated multi-task learning, 2023.

H. B. McMahan, Eider Moore, Daniel Ramage, S. Hampson, and B. A. Y. Arcas. Communication-efficient learning of deep networks from decentralized data. *International Conference on Artificial Intelligence and Statistics*, 2016.

Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 891–898, 2014. doi: 10.1109/CVPR.2014.119.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. *arXiv preprint arXiv: 2305.12827*, 2023.

R. Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. *IEEE Symposium on Security and Privacy*, 2016. doi: 10.1109/SP.2017.41.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: A multi-class classification competition. In *The 2011 International Joint Conference on Neural Networks*, pp. 1453–1460, 2011. doi: 10.1109/IJCNN.2011.6033395.

Jianxiong Xiao, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. Sun database: Exploring a large collection of scene categories. *International Journal of Computer Vision*, 119: 3–22, 2016.

A APPENDIX

A.1 CLIP: FEDAVG VS. NTKFEDAVG 1 EPOCH PER ROUND

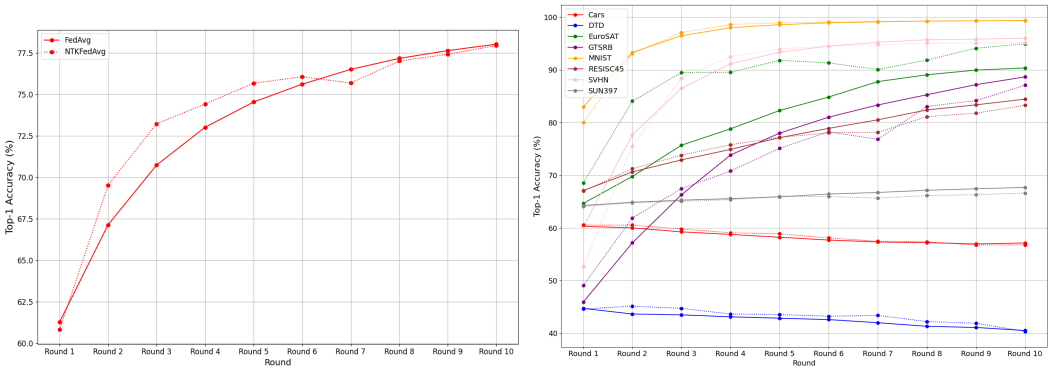

Figure 4: **(Left)** Average accuracy of FedAvg (dots) and NTKFedAvg (dashes) **(Right)** Dataset accuracy of FedAvg (dots) and NTKFedAvg (dashes). All models are trained for 1 epoch per round.

A.2 CLIP: LINEARIZATION POINT

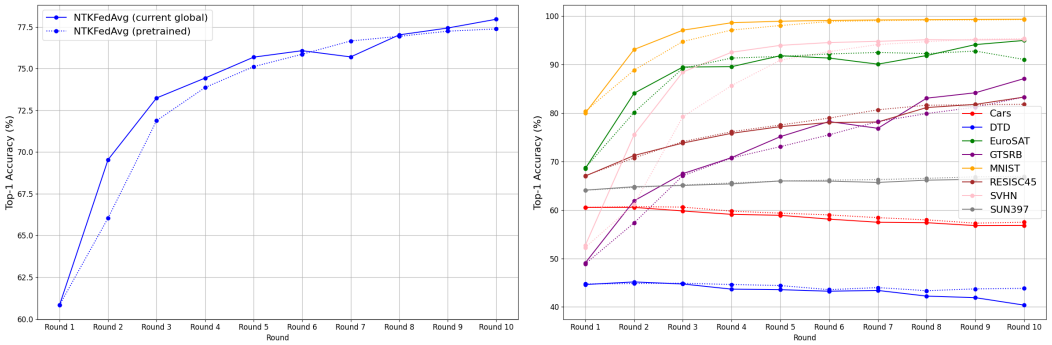

Figure 5: **(Left)** Average accuracy when NTKFedAvg is linearized about current global server model or pretrained model **(Right)** Per dataset accuracy when NTKFedAvg is linearized about current global server model (dash) or pretrained model (dot). All clients are trained for 1 epoch per round.

A.3 CLIP: TUNING LEARNING RATE α FOR FEDAVG AND NTKFEDAVG

We experimented by tuning the learning rate α every round to maximize average accuracy on a held-out validation set on the server (partitioned from the client training sets). In real-world FL applications, this is not a realistic assumption as for reasons of privacy this data is almost never available. However, we wanted to understand the extent to which FedAvg and NTKFedAvg improve by tuning the server learning rate. We see that in both scenarios, whether the clients are only trained for a single epoch every round (Appendix Fig 6), or in the *local work* configuration (Appendix Fig 7), tuning significantly benefits FedAvg, as it better adapts to the varying data characteristics and distributional shifts in each round.

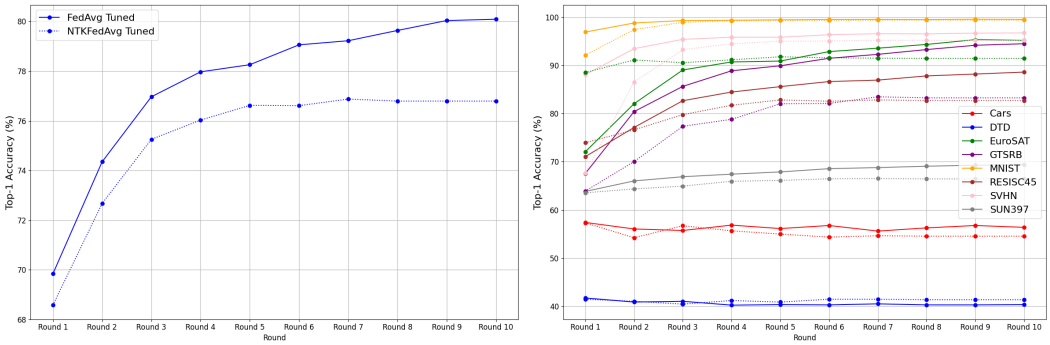

Figure 6: **(Left)** Average accuracy of FedAvg improves significantly when tuning α **(Right)** Dataset specific accuracy of FedAvg improves significantly when tuning α. All clients are trained for 1 epoch per round.

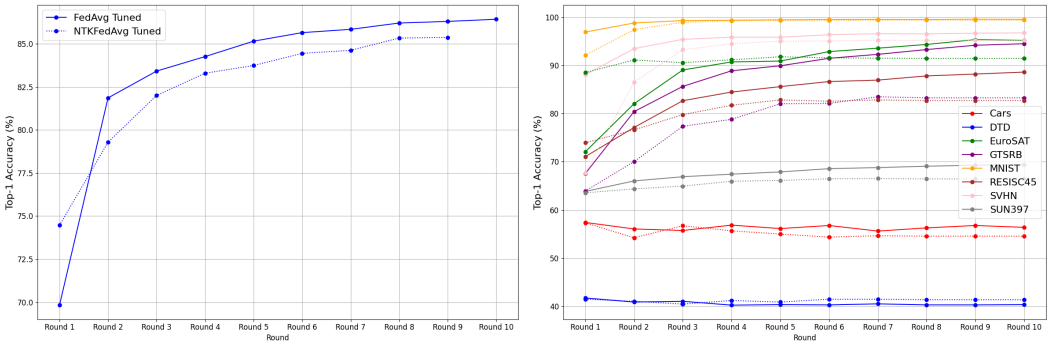

Figure 7: **(Left)** Average accuracy of FedAvg improves significantly when tuning α **(Right)** Dataset specific accuracy of FedAvg improves significantly when tuning α. Clients are trained in the *local work* configuration.

A.4 CLIP:FEDPROX AND NTKFEDPROX

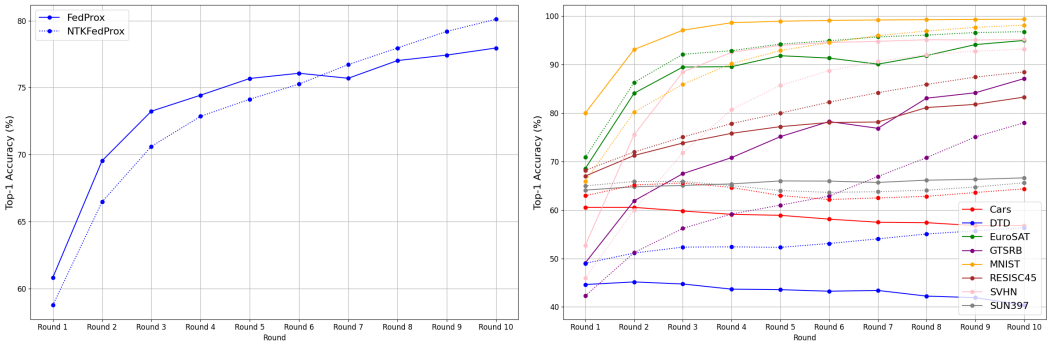

Figure 8: **(Left)** Average accuracy of FedProx (dots) and NTKFedProx (dashes) **(Right)** Per dataset accuracy of FedProx (dots) and NTKFedProx (dashes). All clients are in *local work* configuration.

A.5 CLIP: CENTRALIZED EXPERIMENTS: BASELINE AND UNLEARNING MNIST

For our centralized baseline, we created a multi-task learning setup where at each epoch, batches were created by taking B samples from each task (B = 8). For tasks with a lower number of samples, oversampling was performed to have an even number of samples per batch. During training, inputs were fed into the model per task, and their losses were summed together. For non-linear

(standard) finetuning, we achieved an average task accuracy (%) of 83.795% on the test set. For linear finetuning, we achieved an average test accuracy of 79.98%.

For the unlearning experiment, the sign of the loss on the MNIST task was flipped in each batch. With non-linear (standard) finetuning, MNIST reached an accuracy of 0.00% at epoch 22. With linear finetuning, MNIST did not reach an accuracy of 0.00% in 25 epochs. The lowest accuracy achieved was 0.03% at epoch 12. All experiments were trained for 25 epochs with a learning rate of 1e-7.

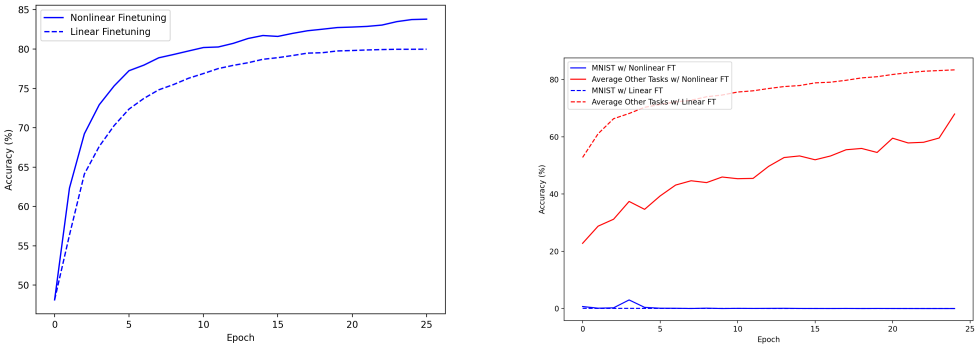

Figure 9: **(Left)** Average Top-1 accuracy across all tasks on centralized multitask learning setup with nonlinear finetuning (solid) and linear finetuning (dots) **(Right)** MNIST Top-1 accuracy versus average other task accuracy for nonlinear finetuning (solid) versus linear finetuning (dashes).