Technische Universität München
School of Computation, Information and Technology
Department of Informatics

Exposé — Bachelor's Thesis

Adaptation Techniques for using NAS Methods in the FL Setting

Author:	Max Coetzee
Supervisor:	M.Sc. Nick Henze
Advisor:	Dr.-Ing. Niclas Kannengießer
Date:	tbd.

Contents

1 Problem Statement 3

2 Objectives 5

3 Explanation of Terms 5

4 Research Approach 5

5 Structure 5

6 Expected Results 6

7 Open Issues and Problems 6

References 6

1 Problem Statement

Engineering the architecture of a neural network for a Deep Learning application is traditionally done by a team of domain experts and Deep Learning experts based on their expert knowledge and a process of trial and error. To reduce the amount of manual labour involved in this process, researchers invented *Neural Architecture Search* (NAS) [2] methods and improved them over the past decade. NAS methods employ diverse strategies to automatically search for a neural network architecture for a given Deep Learning application.

Independantly, but in parallel to NAS, researchers developed a distributed machine learning approach called *Federated Learning* (FL) [11] in response to growing concerns about data privacy. In FL, *clients* collaboratively train a model without sharing their local data. This enhances the privacy of clients' data, because model trainers can not view clients' data and client data is not collected at a central location where a single breach could expose the data of all clients.

Engineering neural network architectures in FL is as time-consuming (if not more so) as in centralised Deep Learning, therefore researchers have started investigating the use of NAS methods in FL [4] [1] [10]. Additionally, NAS methods provide an alternative to selecting a fixed architecture upfront — a so-called *prededfined architecture*. Predefined architectures can lead to slow training convergence and poorly performing models in FL, because model developers can view neither the clients' data nor, typically, client's hardware capabilities. Model developers may therefore select a predefined architecture that contains components irrelevant for generalising well from client data sets or select an architecture that trains slowly on some clients. Work has already been done that shows the use of NAS methods in FL can mitigate these issues [5] [12] [14].

Despite the potential advantages of using NAS in FL, it is still non-trivial to do so. Most NAS methods are infeasible for direct application in FL, because NAS research has focused on the traditional centralised setting as opposed to FL. Centralised NAS can make several assumptions about the search process that do not hold in the FL setting. Each assumption discrepancy creates a *challenge* for using NAS in FL, and developers have created *adaptation techniques* for overcoming them. Applying adaptation techniques to NAS methods results in *Federated Neural Architecture Search methods* (FedNAS methods) [4].

For example, one challenge arises from the fact that centralised NAS can assume worker nodes are computationally powerful, whereas clients in most FL settings are not. Since most centralised NAS methods place large computational burdens on worker nodes, practitioners using these NAS methods in FL without modification would experience detrimental search completion times. To combat this, FedNAS developers have created adaptation techniques to reduce the computational burden on individual clients in FedNAS methods [1] [13] [6]. This typically involves reducing the overall computational work and splitting it up into smaller units, which presents a challenge, as the implementation of the resulting FedNAS method tends to be complex.

The subset of challenges faced by FedNAS methods depends on the specific FL setting, which can differ in many parameters [8]. The literature identifies two major classes of FL settings: the *cross-device* class, wherein clients are edge devices, and the *cross-silo* class, wherein clients are entire organisations, but even within these classes, there is significant variation in the setting parameters. Each FL setting violates centralised NAS assumptions to a different extent, making some challenges more relevant to them than others. For example, for FL settings in the cross-silo class, clients can be expected to be equipped with GPUs, making the challenge described above less relevant.

FedNAS developers who design new FedNAS methods for a specific FL setting must decide which challenges to prioritise and which adaptation techniques to implement for their chosen NAS method. However, informing these design decisions currently requires extensive, manual synthesis of a fragmented literature, because many FedNAS papers do not state the targeted FL setting parameters and assumptions clearly, nor do they clearly link techniques with the addressed challenges. As a result, developers struggle to assess the transferability of existing methods to their setting and risk selecting ineffective or select techniques for addressing a challenge that are known to worsen another. This slows down the development of new FedNAS methods.

The literature on adaptation techniques is fragmented, and FedNAS methods often lack clarity regarding the targeted FL setting and the challenges they address. As a result, extending and re-using existing techniques remains difficult. This poses a problem for FedNAS developers, since they need to trade off which challenges to address for their targeted FL setting without a clear overview of adaptation techniques that would be useful for that setting. Prior literature surveys [15] [9] [3] summarise FedNAS methods on the whole, but do not dissect them in a manner that allows FedNAS developers to decide on the parts they wish to re-use. To aid the development of new FedNAS methods, we set out to answer our research question:

What challenges arise from different FL settings for FedNAS methods, and which adaptation techniques address them in the literature?

To tackle our research question, we conduct a systematic literature review of papers that present FedNAS methods. We employ grounded theory and the methodology from [7]. For our review, we consider papers that modify NAS methods in response to the FL setting.

We define a set of fine-grained parameters to characterise the targeted FL setting of each FedNAS method based on observations of varying setting parameters in the literature. With the help of this characterisation, we identify the violated centralised NAS assumptions and catalogue the challenges that arise from them. Next, we extract unrefined adaptation techniques from the FedNAS methods and iteratively refine and merge them to obtain a set of collectively exhaustive adaptation techniques. We analyse how each adaptation technique works towards, against, or does not affect each challenge, and present our findings in the form of a discussion for each adaptation technique, as well as an overview table.

Our review aims to support the creation of new FedNAS methods by developers. By identifying the source of challenges and elaborating on them, we provide clarity on the expected challenges for a targeted FL setting. Based on the expected challenges, FedNAS developers can use our overview of adaptation techniques to guide the design of new FedNAS methods and determine whether to re-use existing techniques, extend them, or develop new ones.

2 Objectives

- organise FedNAS literature in a way such FedNAS developers can easily re-use techniques

3 Explanation of Terms

- adaptation techniques - NAS - FL - FL setting

4 Research Approach

- qualitative

5 Structure

1. **Introduction** (3 pages)
2. **Background** (6 pages)
   2.1 Neural Architecture Search
   2.2 Federated Learning
   2.3 Federated Neural Architecture Search
3. **Method** (5 pages)

3.1 Method and Literature Selection

3.2 Reviewed Literature

4. **FedNAS Challenges** (10 pages):

   4.1 Parameters of FL settings relevant to FedNAS

   4.2 Assumption Discrepancies between NAS and FedNAS

   4.3 FedNAS Challenges

5. **Adaptation Techniques** (25 pages)

   5.1 Adaptation Technique 1

   5.2 Adaptation Technique 2

   5.3 ...

   5.4 Adaptation Technique 20

   5.5 Overview

6. **Discussion** (2 pages)

7. **Conclusion** (1 page)

6 Expected Results

7 Open Issues and Problems

- finding the correct abstraction level for adaptation techniques

References

[1] L. Dudziak, S. Laskaridis, and J. Fernandez-Marques. *FedorAS: Federated architecture search under system heterogeneity*. 2022. arXiv: `2206.11239 [cs.LG]`.

[2] T. Elsken, J. H. Metzen, and F. Hutter. "Neural architecture search: A survey." In: *Journal of Machine Learning Research* 20.55 (2019), pp. 1–21.

[3] M. Hartmann, G. Danoy, and P. Bouvry. *Multi-objective methods in Federated Learning: A survey and taxonomy*. 2025. arXiv: `2502.03108 [cs.LG]`.

[4] C. He, M. Annavaram, and S. Avestimehr. *Towards Non-I.I.D. and invisible data with FedNAS: Federated deep learning via neural architecture search*. 2021. arXiv: `2004.08546 [cs.LG]`.

[5] M. Hoang and C. Kingsford. "Personalized Neural Architecture Search for Federated Learning." In: *1st NeurIPS Workshop on New Frontiers in Federated Learning (NFFL 2021)* ().

[6] C. Huo, J. Jia, T. Deng, M. Dong, Z. Yu, and D. Yuan. "NASFLY: On-device split federated learning with neural architecture search." In: *2024 IEEE international symposium on parallel and distributed processing with applications (ISPA)*. 2024, pp. 2184–2190. DOI: 10.1109/ISPA63168.2024.00298.

[7] D. Jin, N. Kannengießer, S. Rank, and A. Sunyaev. "Collaborative distributed machine learning." In: 57.4 (Dec. 2024). ISSN: 0360-0300. DOI: 10.1145/3704807.

[8] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao. *Advances and Open Problems in Federated Learning*. 2021. arXiv: 1912.04977 [cs.LG].

[9] S. Khan, A. Rizwan, A. N. Khan, M. Ali, R. Ahmed, and D. H. Kim. "A multi-perspective revisit to the optimization methods of Neural Architecture Search and Hyper-parameter optimization for non-federated and federated learning environments." In: *Computers and Electrical Engineering* 110 (2023), p. 108867. ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2023.108867.

[10] J. Liu, J. Yan, H. Xu, Z. Wang, J. Huang, and Y. Xu. "Finch: Enhancing federated learning with hierarchical neural architecture search." In: *IEEE Transactions on Mobile Computing* 23.5 (2024), pp. 6012–6026. DOI: 10.1109/TMC.2023.3315451.

[11] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. "Communication-efficient learning of deep networks from decentralized data." In: *Proceedings of the 20th international conference on artificial intelligence and statistics*. Ed. by S. Aarti and Z. Jerry. Vol. 54. Proceedings of machine learning research. PMLR, Apr. 2017, pp. 1273–1282.

[12] E. Mushtaq, C. He, J. Ding, and S. Avestimehr. *SPIDER: Searching personalized neural architecture for federated learning*. 2021. arXiv: 2112.13939 [cs.LG].

[13] X. Wei, G. Chen, C. Yang, H. Zhao, C. Wang, and H. Yue. "EFNAS: Efficient federated neural architecture search across AIoT devices." In: *2024 international joint conference on neural networks (IJCNN)*. 2024, pp. 1–8. DOI: 10.1109/IJCNN60899.2024.10650653.

[14] J. Yan, J. Liu, H. Xu, Z. Wang, and C. Qiao. "Peaches: Personalized federated learning with neural architecture search in edge computing." In: *IEEE Transactions on Mobile Computing* 23.11 (2024), pp. 10296–10312. DOI: 10.1109/TMC.2024.3373506.

[15] H. Zhu, H. Zhang, and Y. Jin. "From federated learning to federated neural architecture search: a survey." In: *Complex and Intelligent Systems* 7.2 (Apr. 2021), pp. 639–657. ISSN: 2198-6053. DOI: 10.1007/s40747-020-00247-z.