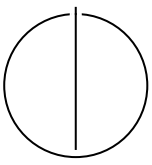
SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Adaptation Techniques for using NAS Methods in the FL Setting

Max Coetzee



SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

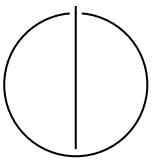
TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Adaptation Techniques for using NAS Methods in the FL Setting

Titel

Author:	Max Coetzee
Examiner:	Supervisor
Supervisor:	Advisor
Submission Date:	Submission date



I confirm that this bachelor's thesis is my own work and I have documented all sources and material used.

Munich, Submission date Max Coetzee

Acknowledgments

Abstract

Both *Neural Architecture Search* (NAS) and *Federated Learning* (FL) have made significant progress independently in the past decade. To benefit from the advantages of NAS methods in FL, researchers have started combining them by using NAS in FL [9] [5] [13].

[TODO: overhaul]

Applying techniques from Neural Architecture Search (NAS) to Federated Learning (FL) has been fruitful (remove) in recent years. The combination was identified as a promising research (remove) direction by [11]. It has yielded methods for finding architectures that deal with the challenges imposed by the FL setting.

Research into NAS has grown rapidly [18] since it was popularized by [20]; consequently, literature on its application to FL has grown. The last survey on NAS applied to FL compared approaches of four papers [19]. Since then, we have identified approximately 50 new papers. This motivates a new systematic survey of the landscape to identify progress and gaps in the literature.

In this thesis, we propose a map of the literature landscape based on the FL challenges they address. We achieve this by systematically evaluating the literature and identifying which challenge it solves.

We refer to the FL challenges described in [20], i.e., non-IID data, limited communication, client heterogeneity, privacy of client data, and break them down into smaller subchallenges — each subchallenge being associated with a pattern in the literature. We include personalized FL [17] as an additional subchallenge that was not originally posited, but has since drawn the community's attention.

We then analyze how the subchallenges are addressed and focus on the contribution of the used NAS method towards overcoming the subchallenge. For each subchallenge, we keep track of the NAS types used (following [18], [2]) and assess whether the underexplored methods are candidates for future research.

Neural Architecture Search

Contents

1 Introduction

In the wake of the deep learning revolution, deep learning is being applied to perform an increasingly diverse set of tasks. Applying deep learning traditionally involves a team of deep learning and domain experts that tailor a neural network architecture towards a specific task via a lengthy process of trial and error. To automate this process, researchers invented *Neural Architecture Search* (NAS) [7] methods. NAS methods employ diverse strategies to automatically search for a neural network architecture for a given deep learning task within an architecture search space.

Independently, but in parallel to NAS, researchers developed a machine learning approach called *Federated Learning* (FL) [14] which enables training a ML model on data that is distributed across a set of *clients* without sharing their data. The model is dispersed from the server to clients and trained in *communication rounds* by clients using their local data, wherafter model weight updates are sent to a *server* for aggregation and re-dispersal for the next round. FL enhances the privacy of client data and allows training even when clients' data is too large to be transferred effectively to a central training site or when clients' data can not be shared due to regulatory or privacy reasons.

When practitioners[1] make use of deep learning in FL, they typically use a predefined architecture. However, clients' data distributions and hardware capabilities are highly heterogenous in FL. Since practitioners can not directly observe these client characteristics [9], it is hard to choose a single architecture that both generalises well across all clients and meets inference latency constraints on every client. Using NAS methods in FL allows for an alternative: the server can coordinate an architecture search across clients that is guided by performance feedback from candidate architectures evaluated directly on clients — making clients charecteristics observable to the NAS method. Thus the NAS method can guide the architecture search better automatically than a practitioner could manually.

Despite these benefits, using NAS in FL is not straightforward. Most NAS methods were developed for a centralised setting and rely on assumptions that often do not hold for FL. These assumption discrepancies create several *challenges* for using NAS in FL and practitioners need to *adapt* the NAS method or the FL pipeline to overcome them — creating a *FedNAS* [9] method. For example, in a centralised setting, practitioners

[1] In this thesis, *practitioners* refers to both users and developers of FL and FedNAS methods.

can expect a NAS method to run on worker nodes connected via low-latency, high-bandwidth links, but in FL, a NAS method potentially runs on clients connected via an unreliable, high-latency and low-bandwith network. This results in a challenge when transferring the weights of large candidate models between clients and the server for coordinating an architecture search. Practitioners have developed various *adaptation techniques* to overcome this challenge, such as only transferring an encoding of a candidate architecture for clients to evaluate [16].

Practitioners stand to benefit from comparing and re-using existing adaptation techniques when creating FedNAS methods. To this end practitioners need to comb the increasingly large body of FedNAS literature for suitable adaptation techniques — i.e. techniques that overcome the challenges relevant to their targeted FL system characteristics. The relevance of a challenge depends on the extent to which the targeted FL system characteristics violate centralised NAS assumptions. Additionally, practitioners need to weigh which adaptation techniques to use to overcome their relevant set of challenges, since an adaptation technique for overcoming one challenge may inadvertantly make it harder to overcome another.

However, comparing and re-using adaptation techniques is difficult. Prior surveys of the FedNAS literature focus on the FedNAS method as a whole instead of the individual adaptation techniques a FedNAS method uses. [19] categorises FedNAS methods into offline versus online architecture search and single-objective versus multi-objective methods. [12] gives a high-level overview of the FedNAS landscape at the time of its publication as part of a larger survey into combining NAS and Hyperparameter Optimisation. [8] investigates how multi-objective optimisation can be integrated into FL in general and includes only parts that discuss how this is done specifically for FedNAS methods. All of the surveys analyse only a small fraction of the entire FedNAS literature. This means knowledge on adaptation techniques and potential challenges when adapting NAS methods to FL remains fragmented, leading to our research question:

What challenges arise when adapting centralised NAS methods to FL and how do adaptation techniques in the literature overcome these challenges?

To tackle our research question, we conduct an extensive systematic literature review. We collect an initial set of relevant literature with a Scopus [6] search and extend it with a snowball search. We first document the challenges that arise when adapting a centralised NAS method to FL as well as the influence of FL system charecteristics on the relevance of challenges in preparation for the next step. We then extract adaptation techniques through open coding from the literature and iteratively refine them afterwards to synthesise a coherent set of mutually exclusive and collectively

exhaustive adaptation techniques. Next, we discuss how each adaptation technique works towards, against, or does not affect overcoming each of the challenges. Finally, we summarise our results in a table that can be used to quickly find adaptation techniques for overcoming a specific challenge.

Our review aims to synthesise adaptation techniques from the literature to support their reuse and comparison in practice. We make three contributions. First, by providing a consolidated overview of the challenges of adapting centralised NAS methods to FL, practitioners are able to grasp at a glance which issues they can expect to run into. Second, by describing how FL system charecteristics influence the relevance of challenges, we enable practitioners to focus on the challenges relevant to them. Third, by extracting adaptation techniques and highlighting the challenges they address, we enable practitioners to compare existing adaptation techniques for adapting centralised NAS methods to FL for their specific set of challenges.

In Chapter 2, we cover the background required for this thesis and related work. In Chapter 3, we describe the method with which we perform our literature review and give an overview of the included literature. In Chapter 4, we highlight the challenges with FedNAS and how they arise from different FL settings. In Chapter 5, we describe the adaptation techniques we synthesised and how they overcome challenges. In Chapter 6, we conduct a discussion about our work. Chapter 7 contains our conclusion.

2 Background and Related Work

NAS is contained entirely in the 5th step of the FL pipeline as described in [11].

2.1 Neural Architecture Search

2.2 Federated Learning

2.3 Federated Neural Architecture Search

3 Method

[TODO: overhaul to include method for identifying challenges and FL settings] - targeted FL setting can be stated directly, indirectly or must be inferred from experimental setup

We use various techniques from [3] and lean on parts of the methodology of [10] to perform a systematic literature review.

1. **Literature Selection:** We follow the guidelines and flow diagrams provided by PRISMA 2020 [15] for inclusion and exclusion of papers and perform forward and backwards citation searching. We identify 59 papers that present FedNAS methods. Each paper contains one or more FedNAS methods.

2. **Unrefined Adaptation Technique Extraction:** Once the set of included papers is fixed, we perform open coding on each FedNAS method to extract unrefined adaptation techniques. Any modification to a NAS method that is explicitly motivated by the federated setting is treated coded as one unrefined adaptation technique.

3. **Adaptation Techniques Coneceptualization:** We iteratively refine and merge unrefined adapatation techniques in an axial coding step to obtain a coherent set of adaptation techniques that are mutually exclusive and collectively exhaustive. Unrefined adaptation techniques with conceptually highly-similar mechanisms are merged into a single representative adaptation technique.

4. **Categorise Adaptation Techniques:** After merging, in a second axial coding step, we cluster adaptation techniques based on a) the FedNAS challenges they address and b) the conceptual similarity of their mechanisms. As a result, we obtain a taxonomy of adaptation techniques.

5. **Discuss FedNAS Challenges for Adaptation Techniques:** We discuss how each adaptation technique works towards, against, or does not affect overcoming each of FedNAS challenges.

6. **Produce Table Overviews:** Finally, we create two tables that practicioners can use to make decisions about FedNAS methods for their use case.

The first table contains a coded vector of effects over the FedNAS challenges for each *FedNAS method*. The effects per FedNAS method are the result of the effects of its adaptation techniques in aggregate. Practicioners can use this table to decide which FedNAS method suits their use case or if their use case would benefit from a new FedNAS method.

The second table table contains a coded vector of effects over the FedNAS challenges for each *adaptation technique* based on the prior discussion. If practicioners need to create new FedNAS methods, they can use this table to choose existing adaptation techniques relevant to the set of FedNAS challenges they need to address for their use case.

3.1 Reviewed Literature

add some overview and statistics about the literature

4 FedNAS Challenges

Assumption in Centralised Setting	Reality in FL Setting	Resulting FedNAS Challenge Description	FedNAS Challenge Name
Worker nodes' hardware is homogeneous.	Clients' hardware varies significantly. More pronounced in the cross-device setting than the cross-silo setting.	FedNAS methods need to be heterogeneity-aware; otherwise, the search will be delayed by low-end devices or leave high-end devices waiting in idle most of the time.	Hardware Heterogeneity
Worker nodes communicate over high-bandwidth, low-latency links.	Clients typically communicate with the central server over high-latency and low-bandwidth connections.	FedNAS must be communication-efficient; otherwise, exchanging model weights and architecture parameters becomes a bottleneck during the search process.	Limited Networking Capabilities
Worker nodes are high-end machines with powerful CPUs, GPUs, and large amounts of RAM.	Clients are edge devices with few computational resources.	The computational burden placed on clients by a NAS method needs to be adjusted such that the architecture search takes place within an acceptable time frame. This can involve splitting up computational work, usually done on single worker nodes, amongst multiple clients, making the implementation complex.	Limited Computational Resources
Training data is gathered at a central location.	Training data is distributed unevenly across clients	Since some clients have more data than others, FedNAS methods must ensure that these clients are not overrepresented in the searched architecture.	Unbalanced Client Data

The training data is drawn from the same distribution.	Each client draws training data from their own distribution.	FedNAS methods need to be robust with respect to non-i.i.d. data, which makes evaluating and ranking candidate architectures noisier than in the centralised setting.	Non-I.I.D. Training Data
Worker nodes are equally available.	Some clients are more frequently available than others.	FedNAS methods must prevent the search from being dominated by the most frequently available clients; otherwise, the resulting architecture will be biased towards them.	Variable Client Availability
All worker nodes participate in each iteration.	Only a subset of clients participates in each iteration of the search.	FedNAS methods must ensure that a realistic sample of the client population is represented in the architecture search and address high-variance performance estimates, as candidate architectures are evaluated on changing subsets of clients.	Client Participation
Worker nodes are inside the same trust domain.	All participating parties (i.e. the central server and clients) can consider another potentially malicious.	FedNAS methods must address attacks from participating parties, such as architecture parameter poisoning during the search process.	Security
Worker nodes consistently participate in the search process.	Clients can drop out of a communication round at any time, and completion times of epochs vary.	FedNAS methods need to handle client dropouts and stragglers, since interrupted or delayed evaluations of candidate architectures can slow down or destabilise the search process.	Client Reliability

Model accuracy is more important than model resource consumption in selecting architectures.	Model accuracy needs to be traded for model resource consumption in selecting architectures.	Models trained in FL tend to be deployed on the same resource-constrained clients they are trained on. Therefore FedNAS methods need to optimise the architecture of the trained model w.r.t. resource constraints.	Model Resource Constraints

Table 4.1: Discrepancies between NAS in the centralised setting and the FL setting and the resulting FedNAS challenges. *Worker nodes* perform the architecture search in the centralised setting, whereas a *server* and *clients* perform the search in the FL setting. The challenges are based to some extent on previously identified challenges in the FL setting in general [14] [11] [4] [1].

5 Adaptation Techniques

5.1

6 Discussion

7 Conclusion

Abbreviations

List of Figures

List of Tables

Bibliography

[1] M. Arbaoui, M.-A. Brahmia, A. Rahmoun, and M. Zghal. "Federated learning survey: A multi-level taxonomy of aggregation techniques, experimental insights, and future frontiers." In: *ACM Trans. Intell. Syst. Technol.* 15.6 (Nov. 2024). ISSN: 2157-6904. DOI: 10.1145/3678182.

[2] S. S. P. Avval, N. D. Eskue, R. M. Groves, and V. Yaghoubi. "Systematic review on neural architecture search." In: *Artificial Intelligence Review* 58.3 (Jan. 6, 2025), p. 73. ISSN: 1573-7462. DOI: 10.1007/s10462-024-11058-w.

[3] J. Corbin and A. Strauss. *Basics of qualitative research*. Core textbook v. 14. SAGE Publications, 2015. ISBN: 9781412997461.

[4] K. Daly, H. Eichner, P. Kairouz, H. B. McMahan, D. Ramage, and Z. Xu. "Federated learning in practice: Reflections and projections." In: *2024 IEEE 6th international conference on trust, privacy and security in intelligent systems, and applications (TPS-ISA)*. 2024, pp. 148–156. DOI: 10.1109/TPS-ISA62245.2024.00026.

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

[6] Elsevier. *Scopus*. Abstract and citation database. Accessed 2026-01-17. 2026.

[7] 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.

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

[9] 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].

[10] 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.

[11] 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].

[12] 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.

[13] 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.

[14] 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.

[15] M. J. Page, J. E. McKenzie, P. M. Bossuyt, I. Boutron, T. C. Hoffmann, C. D. Mulrow, L. Shamseer, J. M. Tetzlaff, E. A. Akl, S. E. Brennan, R. Chou, J. Glanville, J. M. Grimshaw, A. Hróbjartsson, M. M. Lalu, T. Li, E. W. Loder, E. Mayo-Wilson, S. McDonald, L. A. McGuinness, L. A. Stewart, J. Thomas, A. C. Tricco, V. A. Welch, P. Whiting, and D. Moher. "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews." In: *BMJ* 372 (2021). DOI: 10.1136/bmj.n71. eprint: https://www.bmj.com/content/372/bmj.n71.full.pdf.

[16] R. E. Shawi. "AgingFedNAS: Aging evolution federated deep learning for architecture and hyperparameter search." In: *Advances in knowledge discovery and data mining*. Ed. by X. Wu, M. Spiliopoulou, C. Wang, V. Kumar, L. Cao, Y. Wu, Y. Yao, and Z. Wu. Singapore: Springer Nature Singapore, 2025, pp. 107–119. ISBN: 978-981-96-8173-0.

[17] A. Z. Tan, H. Yu, L. Cui, and Q. Yang. "Towards personalized federated learning." In: *IEEE Transactions on Neural Networks and Learning Systems* 34.12 (2023), pp. 9587–9603. DOI: 10.1109/TNNLS.2022.3160699.

[18] C. White, M. Safari, R. Sukthanker, B. Ru, T. Elsken, A. Zela, D. Dey, and F. Hutter. *Neural architecture search: Insights from 1000 papers*. 2023. arXiv: 2301.08727 [cs.LG].

[19] 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.

[20] B. Zoph and Q. V. Le. *Neural architecture search with reinforcement learning*. 2017. arXiv: 1611.01578 [cs.LG].