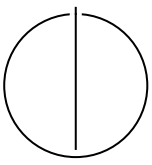
SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

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

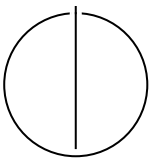
TECHNISCHE UNIVERSITÄT MÜNCHEN

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

[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 [13]. It has yielded methods for finding architectures that deal with the challenges imposed by the FL setting.

Research into NAS has grown rapidly [21] since it was popularized by [24]; consequently, literature on its application to FL has grown. The last survey on NAS applied to FL compared approaches of four papers [23]. 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 [24], 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 [19] 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 [21], [2]) and assess whether the underexplored methods are candidates for future research.

Neural Architecture Search

Contents

1 Introduction

Both Neural Architecture Search (NAS) and Federated Learning (FL) have made significant progress independently in the past decade, and both are widely used in practice. To benefit from the advantages of NAS methods in FL, researchers have started combining them by using NAS in the FL setting.

Neural Architecture Search (NAS) automates the process of engineering neural network architectures for Deep Learning application domains [6].

NAS saves manual effort compared to the traditional, labourious approach to applying Deep Learning, wherein a team of domain experts engineer an architecture based on expert knowledge and trial and error. Additionally, NAS can find architectures that perform better than architectures humans have designed for specific application domains [25] [18] [10] [20].

Federated Learning (FL) is a machine learning method whereby *clients* collaboratively train a model without sharing their data. Weight updates to the shared model are coordinated by a central *server*.

FL enables privacy-preserving machine learning on the increasing volume of privacy-sensitive distributed data and avoids centrally collecting client data. Two major strains of FL exist: the *cross-device* setting, wherein clients are edge devices, and the *cross-silo* setting wherein clients are entire organisations. Both kinds of FL have been adopted for various ML tasks in production systems by organisations such as Google [22], Apple [11], and Owkin [16].

By using NAS in the FL setting, practitioners gain the generic benefits of NAS mentioned above as well as several benefits specific to the FL setting:

- A large body of work in NAS focuses on finding architectures within computational resource constraints that still have reasonable accuracy [21]. The deployment target in FL is often the resource-constrained clients themselves. This means practitioners can use NAS to find architectures that meet client constraints.

- [13] note that predefined architectures may not be an optimal choice for FL. Since client data is not visible to model developers, a predefined architecture selected by model developers may contain components redundant for generalising well from client data sets. For example, a language model trained on a central text corpus may devote substantial model capacity to words that are rare on individual clients.

This results in large parts of the neural network contributing little to accuracy on those clients. NAS can personalize the architecture to dedicate more capcity towards words representive of each clients' local text data.

- Predefined architectures may perform poorly on another prevalent characteristic of the FL setting: data that is not independently and identically distributed (non-i.i.d.). A predefined architecture tuned on i.i.d. benchmarks can underfit some clients and overfit others, whereas NAS can adjust an architecture's capacity to better accommodate variation across clients.

However, using NAS in the FL setting is non-trivial. Research on NAS methods has traditionally focused on a centralised setting as opposed to the distributed FL setting. This makes many NAS methods infeasible for direct application in the FL setting, because NAS methods designed for the centralised setting make several assumptions about the search process that do not hold in the FL setting (see Table 1.1). Instead, practitioners need to adapt NAS methods to their FL setting, giving rise to *Federated Neural Architecture Search* [9] (FedNAS) methods.

As a result of the difference in assumptions, FedNAS methods face several challenges when adapting NAS methods. Table 1.1 illustrates these discrepancies as well as the resulting challenges faced by FedNAS methods.

Depending on the FedNAS use case, some centralised NAS assumptions are violated to a larger extent than others. For example, FedNAS methods for use cases in the cross-silo FL setting can assume that clients are equipped with GPUs, making the *Limited Computational Resources* challenge less relevant.

FedNAS practitioners need to prioritise which challenges to address for their particular use case: the relevance of a challenge depends on how strongly the corresponding centralised NAS assumption is violated, and overcoming one challenge typically comes at the expense of neglecting others. For example, in a use case where some clients hold more informative data but frequently drop out, ignoring stragglers addresses the *Client Reliability* challenge and lowers search completion time but risks biasing the search towards always-available clients, whereas waiting for stragglers addresses the *Variable Client Availability* challenge and improves fairness at the cost of slower and potentially less stable search.

FedNAS practitioners have diverse use cases for FedNAS methods and can choose from a wide variety of subsets of challenges to overcome. Consequently, practitioners started creating a growing body of FedNAS methods. An overview of FedNAS methods would be helpful to practitioners, as it would enable them to find suitable FedNAS methods for their use case or aid in re-using knowledge for creating new FedNAS methods. However, no such overview exists.

Researchers have already conducted several literature surveys on FedNAS methods [23] [14] [8]. [23] is an early survey that characterises FedNAS methods on the whole. The survey categorises FedNAS methods into offline versus online architecture search and single-objective versus multi-objective methods. [14] gives a brief overview of the FedNAS landscape at the time as part of a larger survey into combining NAS and Hyperparameter Optimisation. It highlights the major scientific contributions each FedNAS method has made. [8] provides an overview of how multi-objective optimisation can be integrated into FL in general and includes sections that discuss how this is done specifically for FedNAS methods.

Existing literature surveys only analyse a fraction of the FedNAS literature. [23] and [14] are limited by the small amount of FedNAS literature available at the time. The volume of proposed FedNAS methods has grown substantially since. [8] only analyses FedNAS methods that make use of multi-objective optimisation, thereby excluding a large share of the literature.

We introduce the term *adaptation techniques* to refer to the techniques employed by FedNAS methods to adapt NAS to the FL setting. For example, naively using a supernet-based NAS method in the cross-device FL setting by allowing all clients to evaluate any candidate architecture, regardless of the computational footprint, would significantly increase the wall-clock duration of the search process, because low-end devices end up evaluating computationally expensive architectures. This embodies the *Client Hardware Heterogeneity* challenge, and one FedNAS method [5] overcomes it by using the following adaptation technique: The subnet sampling method of SPOS [7] is adapted, such that only subnets within the client's training budget are selected for evaluation.

None of the existing literature surveys identify adaptation techniques used by FedNAS methods and analyse how they overcome FedNAS challenges (Table 1.1). [23] and [14] only analyse and summarise FedNAS methods on the whole. [8] only analyses how multi-objective optimisation is used within FedNAS methods. This leaves adaptation techniques scattered throughout the literature, making it difficult for practitioners to re-use them for creating new FedNAS methods and to decide which adaptation techniques are suitable for their use case.

As mentioned above, the lack of an exhaustive overview of FedNAS methods, the adaptation techniques they use and how these adaptation techniques overcome FedNAS challenges, leads us to our research question:

(RQ) **Which adaptation techniques are described in the literature, and how do they address FedNAS challenges?**

To answer our research question, we conduct a systematic literature review using

grounded theory and the methodology employed by [12].

We first identify papers that present FedNAS methods. Then, we perform open coding on each FedNAS method to extract unrefined adaptation techniques. Next, we perform axial coding by iteratively refining and merging unrefined adaptation techniques to obtain a coherent set of mutually exclusive and collectively exhaustive adaptation techniques.

We then analyse how each adaptation technique works towards, against, or does not affect each FedNAS challenge, and aggregate these coded effects into two overview tables: one for FedNAS methods and one for adaptation techniques. These tables help practitioners decide on FedNAS methods suited to their use case or adaptation techniques that practitioners can use to create new FedNAS methods suited to their use case.

Our review organises the n extracted adaptation techniques into a consolidated body of knowledge that provides FedNAS practitioners with an overview of the FedNAS landscape through the lens of adaptation techniques. Compared to existing surveys, our review is exhaustive of the FedNAS landscape published up to 2025. Additionally, our discussions on each adaptation technique and overview tables help practitioners find and choose FedNAS methods relevant to their use case or construct new FedNAS methods by re-using appropriate adaptation techniques.

In Chapter 2, we cover the background required for this thesis and related work. In Chapter 3, we describe the method with which we conduct our literature review in detail. In Chapter 4, we describe our process for including FedNAS literature and provide an overview of the reviewed FedNAS literature. In Chapter 5, we present our taxonomy of adaptation techniques and explain the effect of these techniques on challenge classes. In Chapter 6, we conduct a discussion about our work. Chapter 7 contains our conclusion.

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

Table 1.1: 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 [15] [13] [4] [1].

2 Background and Related Work

NAS is contained entirely in the 5th step of the FL pipeline as described in [13].

2.1 Neural Architecture Search

2.2 Federated Learning

2.3 Federated Neural Architecture Search

2.4 Grounded Theory

3 Method

We use various techniques from [3] and lean on parts of the methodology of [12] to perform a systematic literature review.

1. **Literature Selection:** We follow the guidelines and flow diagrams provided by PRISMA 2020 [17] for inclusion and exclusion of papers and perform forward and backwards citation searching. We identify 58 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.

4 Reviewed Literature

5 Adaptation Techniques

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] 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.

[7] Z. Guo, X. Zhang, H. Mu, W. Heng, Z. Liu, Y. Wei, and J. Sun. "Single path one-shot neural architecture search with uniform sampling." In: *Computer vision – ECCV 2020*. Ed. by A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm. Cham: Springer International Publishing, 2020, pp. 544–560. ISBN: 978-3-030-58517-4.

[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] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le. "Searching for MobileNetV3." In: *2019 IEEE/CVF international conference on computer vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, Nov. 2019, pp. 1314–1324. DOI: 10.1109/ICCV.2019.00140.

[11] A. Ji, B. Bandyopadhyay, C. Song, N. Krishnaswami, P. Vashisht, R. Smiroldo, I. Litton, S. Mahinder, M. Chitnis, and A. W. Hill. *Private federated learning in real world application – a case study*. 2025.

[12] 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.

[13] 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].

[14] 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.

[15] 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.

[16] M. Oldenhof, G. Ács, B. Pejó, A. Schuffenhauer, N. Holway, N. Sturm, A. Dieckmann, O. Fortmeier, E. Boniface, C. Mayer, A. Gohier, P. Schmidtke, R. Niwayama, D. Kopecky, L. Mervin, P. C. Rathi, L. Friedrich, A. Formanek, P. Antal, J. Rahaman, A. Zalewski, W. Heyndrickx, E. Oluoch, M. Stößel, M. Vančo, D. Endico, F. Gelus, T. Boisfossé, A. Darbier, A. Nicollet, M. Blottière, M. Telenczuk, V. T. Nguyen, T. Martinez, C. Boillet, K. Moutet, A. Picosson, A. Gasser, I. Djafar, A. Simon, Á. Arany, J. Simm, Y. Moreau, O. Engkvist, H. Ceulemans, C. Marini,

and M. Galtier. *Industry-scale orchestrated federated learning for drug discovery*. 2022. arXiv: 2210.08871 [cs.LG].

[17] 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.

[18] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le. "Regularized evolution for image classifier architecture search." In: *Proceedings of the thirty-third AAAI conference on artificial intelligence and thirty-first innovative applications of artificial intelligence conference and ninth AAAI symposium on educational advances in artificial intelligence*. AAAI'19/IAAI'19/EAAI'19. Honolulu, Hawaii, USA: AAAI Press, 2019. ISBN: 978-1-57735-809-1. DOI: 10.1609/aaai.v33i01.33014780.

[19] 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.

[20] M. Tan and Q. Le. "EfficientNetV2: Smaller models and faster training." In: *Proceedings of the 38th international conference on machine learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of machine learning research. PMLR, July 2021, pp. 10096–10106.

[21] 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].

[22] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays. *Applied federated learning: Improving google keyboard query suggestions*. 2018. arXiv: 1812.02903 [cs.LG].

[23] 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.

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

[25] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. "Learning transferable architectures for scalable image recognition." In: *2018 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2018, pp. 8697–8710. DOI: 10.1109/CVPR.2018.00907.