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

Research into NAS has grown rapidly [29] since it was popularized by [36]; consequently, literature on its application to FL has grown. The last survey on NAS applied to FL compared approaches of four papers [35]. 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 [36], 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 [25] 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 [29], [3]) and assess whether the underexplored methods are candidates for future research.

Neural Architecture Search

# Contents

# 1 Introduction

Applying Neural Architecture Search (NAS) methods in a Federated Learning (FL) setting is an emerging field of study, which we shall abbreviate to *FedNAS*. In this chapter, we motivate a systematic literature review of *adaptation techniques* used by FedNAS methods to adapt NAS to the FL setting. Our motivation leads us to our research questions and ultimately, our proposed methodology for addressing them.

## 1.1 Context

Relevant to this thesis, the last decade witnessed two major developments in parallel: 1. the establishment of neural architecture search (NAS) and 2. the invention of Federated Learning (FL). Together, they form the cornerstone of this thesis, and their origins can be explained as follows:

1. **Origin of Neural Architecture Search:** Deep learning (DL), a subfield of machine learning (ML) that makes use of deep neural networks, made key advances in various machine learning tasks such as computer vision, natural language processing (NLP), speech recognition, etc. These advances led to the widespread adoption of DL, with some referring to it as a deep learning revolution. However, the advances were built upon neural network architectures that were manually designed by researchers through extensive trial and error — for example, convolutional neural networks for computer vision tasks and the transformer for NLP tasks. It did not take long for researchers to wonder whether automating the search for novel architectures could produce architectures that improved upon the best manually designed ones. In 2017, [36] demonstrated that this was indeed possible, coining the term *Neural Architecture Search* (NAS) for the automated search of neural network architectures for a given task.[1]

2. **Origin of Federated Learning:** Edge devices, such as mobile phones or Internet-of-Things devices, have become ubiquitous and increasingly store privacy-sensitive data. Traditionally, this privacy-sensitive data was either not utilised at all for

[1]NAS was investigated in the 1980s, but under different names and not nearly as intensively. See [27], [18], [14] and [1].

> machine learning or it was collected in a central location for training — creating the risk of a major breach by malicious actors. Google invented *Federated Learning* in 2016 to address this issue and allow the use of decentralised training data without the data leaving the device on which it was generated. Instead of performing training at a central location with a cluster of high-end machines, training happens on the edge devices in FL (typically referred to as *clients* in FL). The clients train a shared model on their local data and coordinate weight updates to the shared model via a central server. FL "*embodies the principles of focused collection and data minimisation, and can mitigate many of the systemic privacy risks and costs resulting from traditional, centralised machine learning*" [12].

Both NAS and FL have made significant progress independently in recent years, and both are increasingly adopted in practice.

More than 1000 papers have been published about NAS since 2017. The seminal paper [36] employed reinforcement learning, a black-box optimisation technique, and discovered an architecture that matched handcrafted state-of-the-art architectures in terms of accuracy for image classification, while also being slightly faster, demonstrating the potential of NAS. NAS methods proposed early on were inefficient for training[2] and a significant amount of research attention improved the situation soon after. As opposed to earlier black-box optimisation techniques, newly developed techniques tend to utilise the internals of the searched architectures to improve training efficiency — ENAS [22] and DARTS [15] are prominent examples. An iterative, inevitable march of progress significantly accelerated NAS by orders of magnitude. Faster NAS has enabled broader adoption and, consequently, more experimentation for finding architectures with better accuracy, smaller size, and faster training convergence. Architectures found via NAS have improved upon state-of-the-art handcrafted architectures for various Machine Learning tasks (e.g. NASNet [37], AmoebaNet [23], MobileNetV3 [9], EfficientNetV2 [26] and many more).

FL has also received significant research attention. The seminal paper on FL [17] was only concerned with edge devices as clients; however, since then, FL has also been used for training shared models between organisations [24] [5] [4]. The former is referred to as the *cross-device* setting and the latter as the *cross-silo* setting. In fact, FL has been identified as a good fit for collaborative distributed machine learning [11] in general. Since its inception in 2016, FL has been adopted for various ML tasks in production systems by organisations like Google [31], Apple [10] and Owkin [20].

[2]The authors of [36] reportedly ran their search for two weeks on 800 GPUs.

## 1.2 Motivation

Both Neural Architecture Search (NAS) and Federated Learning (FL) have made significant progress independently in the past decade, and both are increasingly adopted 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 engineeering neural network architectures for specific Deep Learning application domains [7]. This stands in contrast to the traditional, labourious approach to applying Deep Learning, whereby a team of domain experts and Deep Learning experts engineer a well-suited architecutre based on expert knowledge and trial-and-error. NAS does not only reduce manual effort, but can also be used to find architectures that perform better than architectures humans have designed for specific application domains [37] [23] [9] [26].

Federated Learning (FL) is a machine learning method whereby clients collaboratively train a model without sharing their data. This is achieved by clients training a shared model on their local data and coordinating weight updates to the shared model via a central server. FL was originally invented by Google to enable the usage of the increasing volume of privacy-sensitive data stored on only edge devices, but distributed data silos containing privacy-sensitive data have become another major use case. The former is referred to as the *cross-device* FL setting and the latter as the *cross-silo* FL setting. Traditionally, distributed privacy-sensitive data was either not used at all for machine learning or it was collected in a central location for training — creating the risk of a major breach by malicious actors.

By using NAS in the FL setting, researchers not only benefit from the generic benefits of NAS mentioned above, but from several advantages automatically searching for an architecture has over using a predefined, manually engineered architecture in FL:

- A large body of work in NAS has focused on finding smaller architectures with reduced inference latency that still have reasonable accuracy [29]. Such lightweight architectures are ideal for deployment on the resource-constrained clients in the cross-device FL setting.
- [12] note that predefined architectures may not be an optimal choice for FL, where user-generated data is not visible to model developers.
- Further, they note that predefined architectures may contain components redundant for specific data sets.
- Finally, 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).

large body of work in NAS that has focused on finding smaller architectures with reduced inference latency that still have reasonable accuracy.

Users of FL naturally expect to leverage existing NAS methods in the FL setting to produce architectures that achieve state-of-the-art accuracy.

Apart from the generic benefits, NAS has been identified as a good fit for the FL setting.

However, using NAS in the FL setting is not straightforward. NAS methods described in the literature focus on a centralized setting in contrast to the distributed FL setting. When used in a centralized setting, NAS methods can assume i) that worker nodes will have high availability, ii) that the entire training dataset can be accessed, iii) that the distribution of the training data can be inferred, etc. These assumptions do not hold for the FL setting, making most NAS methods unfeasable for direct application in the FL setting. In fact, the FL setting imposes a unique set of *challenges* on NAS methods.

- researcher has problem - problem is a good fit for using FL - researcher wants to use NAS for better model (how can NAS make model better?) -

- researcher wants to use NAS to help

Making use of NAS in the FL setting, requires adapting NAS methods to the FL setting, giving rise to what we shall call *FedNAS* methods. Each FedNAS method uses several *adaptation techniques*, heavily dependant on i) the type of NAS method and ii) the challenges the FedNAS method aims to overcome.

Users of FL naturally expect to leverage existing NAS methods in the FL setting to produce architectures that achieve state-of-the-art accuracy. Apart from the generic benefits, NAS has been identified as a good fit for the FL setting. A large body of work in NAS has focused on finding smaller architectures with reduced inference latency that still have reasonable accuracy [29]. Such lightweight architectures are ideal for deployment on the often resource-constrained clients in the FL setting. [12] also note that predefined architectures may not be an optimal choice for FL, where user-generated data is not visible to model developers. Further, they note that predefined architectures may contain components redundant for specific data sets. Finally, 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).

With NAS, the architecture search can take place directly on the clients, and architectures can be adjusted according to the data and distribution present on them, yielding architectures that are better suited to the data than those a model developer would manually select. Dealing with the distributed, non-i.i.d. data of the FL setting has been a key challenge for adopting NAS in the FL setting, and research into the matter has shown promise [8] [33] [6] [16] [30]. Beyond being tailored to the data and distribution of each client, architectures can also be personalised to clients' heterogeneous computation and bandwidth budgets, as shown by [13], [6], [32], and [34].

However, research in NAS methods has focused on a central NAS setting, and the methods rely on assumptions that do not hold for the FL setting. These assumptions include an abundance of computational resources, homogeneity of hardware for training and deployment, high availability of worker nodes, access to the entire dataset, access to the data distribution, etc. [12]. The misaligned assumptions lead to *challenges* in adopting NAS in an FL setting. Consequently, most centralised NAS methods are unfeasible for direct application in the FL setting. Adopting NAS methods in the FL setting requires adapting them to the FL setting, giving rise to what we shall call *FedNAS* methods. We have identified 58 papers that propose FedNAS methods. Each FedNAS method employs various techniques to adapt a NAS method to the FL setting. We shall refer to these techniques as *adaptation techniques* for short. In the following, we briefly illustrate three adaptation techniques.

1. **Efficiently Training Supernets in FL:** For example, while naively training a supernet in FL by having each client train the entire supernet, works for the cross-silo FL setting [8], it does not translate to the cross-device setting. Clients in the cross-device setting are generally less powerful, and such a training scheme would result in detrimental completion times. Instead, in one FedNAS method [6], researchers have opted to reduce the computational burden on clients by only sampling and training subnets within the client's training budget.

2. **Counter-acting suboptimally averaged architecture parameters:** Another problem arises when using standard FedAvg to average the architecture parameters for DARTS-based supernets [15]. Clients may optimise architecture parameters towards the opposite ends of a spectrum, but averaging the architecture parameters may select for an architecture that is not favoured by any client. In [28], this is addressed by aggregating the architecture parameters into a probability distribution that can be used to sample likely architectures in the subsequent communication rounds.

3. **Client heterogeneity-aware tiering:** Typically, when NAS methods are run in a central environment, they implicitly assume that the machines on which NAS is performed are homogeneous and contribute equally to the architecture search. This assumption does not hold in the FL setting, where the variance in computational resources between clients can be substantial. Consequently, higher-end clients will tend to be used more in training, introducing bias to the searched architecture. [19] adapts a NAS method to overcome this problem by grouping clients into tiers according to their computational speed and network bandwidth. Small models are trained on low-end client tiers, while larger models are trained on high-end client tiers.

As shown by the examples above, the conditions of the FL setting break NAS methods in ways that require novel adaptation techniques. While NAS research has focused on finding ways to perform NAS faster and finding better and smaller architectures, a key challenge of FedNAS methods lies in achieving these goals despite the challenges imposed by the FL setting — and adaptation techniques are key.

A considerable amount of literature on FedNAS methods has appeared, but no consolidated body of knowledge exists that can inform researchers on existing adaptation techniques and aid them in designing new techniques. To mend this, we perform a systematic review of adaptation techniques in this thesis.

## 1.3 Research Questions

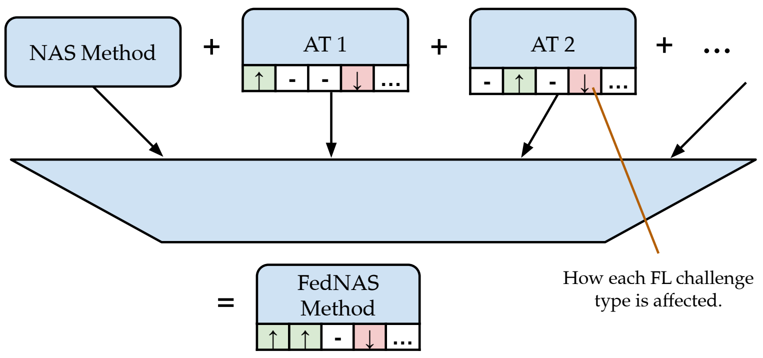

Figure 1.1: A conceptual sketch of how a FedNAS method is composed. **AT** = adaptation technique.

Using either NAS or FL on its own is a complex task. Combining them introduces even more possible knobs to turn on the system as a whole. There are many NAS methods for researchers to choose from, and even more potential adaptation techniques to address FL challenges — culminating in a large number of possible FedNAS methods. To understand the current landscape, we ask our first research question:

(RQ1) Which FedNAS methods have been described in the literature, and which adaptation techniques do they use?

As mentioned before, we expect a diverse array of adaptation techniques in the literature. Yet, we also expect some FedNAS methods to use similar adaptation techniques if they address the same type of FL challenges. This leads us to our second research question:

(RQ2) Which adaptation techniques are similar, and what types of challenges of the FL setting do they address?

Adaptation techniques are heavily dependent on the targeted type of FL challenge and the NAS method used. Naturally, this does not allow for most adaptation techniques to work together. Additionally, some adaptation techniques are incompatible due to other reasons. For example, the architecture weight aggregation technique proposed by [28] aims to gradually increase the likelihood of sampling the best-performing architectures during training. In contrast, [13] introduces a technique that attempts to maintain a diverse set of architectures throughout the training process. We would still like to know which existing adaptation techniques can be combined in what way, possibly paving the way for creating more powerful FedNAS methods tailored towards specific use cases. This leads us to our third research question:

Optional (RQ3) How can adaptation techniques described in the literature be composed into new, potentially better FedNAS methods?

## 1.4 Methodology

To address our research questions, we divide our approach into six steps:

1. **Literature Selection:** To tackle the first part of RQ1, we follow the guidelines and flow diagrams provided by PRISMA 2020 [21] for inclusion and exclusion of papers and perform forward and backwards citation searching. Each paper presents one or more FedNAS methods.
2. **Adaptation Technique Extraction:** Once the set of included papers is fixed, we tackle the second part of RQ1 by analysing each paper individually, extracting the adaptation techniques it uses and summarising them.
3. **Categorise Adaptation Techniques:** For the first part of RQ2, we cluster adaptation techniques based on conceptual similarities and deliver a taxonomy of adaptation techniques.
4. **Define FL Challenge Types:** For the second part of RQ2, we must first define a classification of FL challenges. To this end, we repurpose a prior classification [2] of recent research advances in FL (as shown in 1.2). [2] suggests that newly proposed FL methods in FL research enhance one or more *evaluation metrics*. We argue that their aforementioned evaluation metrics are the result of finding a

measure for how effectively an FL challenge has been addressed. Therefore, the clustering of evaluation metrics into research directions is akin to clustering FL challenges into types of FL challenges.

5. **Mapping FL Challenge Types onto Adaptation Technique Clusters:** Next, we review all clusters of adaptation techniques from step 3, and discuss how each adaptation technique cluster works towards, against, or does not affect overcoming each of the FL challenge types defined in step 4.

6. **Compose promising FedNAS methods:** For RQ3, we propose building an "is-compatible-with" relation on the set of adaptation technique clusters. We achieve this by cross-examining each adaptation technique cluster with every other one. We then utilise the "is-compatible-with" relation, together with our discussion from step 5, to identify groups of adaptation technique clusters that can be composed into FedNAS methods, which could potentially shift the Pareto frontier with respect to the optimality of addressing FL Challenge types.

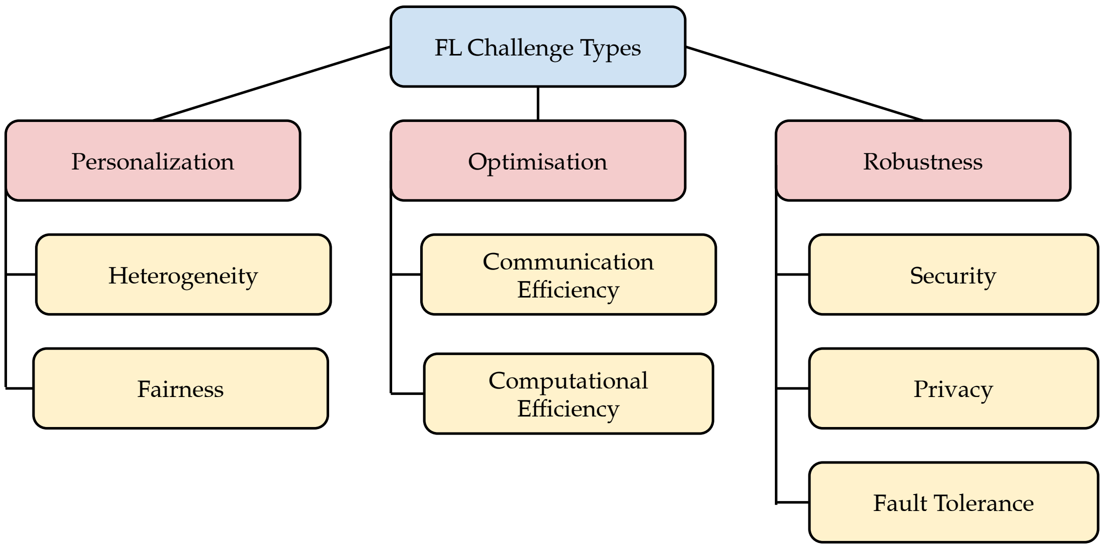

Figure 1.2: A taxonomy of FL challenge types based on [2].

# Abbreviations

# List of Figures

# List of Tables

# Bibliography

[1] P. Angeline, G. Saunders, and J. Pollack. "An evolutionary algorithm that constructs recurrent neural networks." In: *IEEE Transactions on Neural Networks* 5.1 (1994), pp. 54–65. DOI: 10.1109/72.265960.

[2] 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.

[3] 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.

[4] A. Das, T. Castiglia, S. Wang, and S. Patterson. "Cross-silo federated learning for multi-tier networks with vertical and horizontal data partitioning." In: 13.6 (Sept. 2022). ISSN: 2157-6904. DOI: 10.1145/3543433.

[5] I. Dayan, H. R. Roth, A. Zhong, A. Harouni, A. Gentili, A. Z. Abidin, A. Liu, A. B. Costa, B. J. Wood, C.-S. Tsai, C.-H. Wang, C.-N. Hsu, C. K. Lee, P. Ruan, D. Xu, D. Wu, E. Huang, F. C. Kitamura, G. Lacey, G. C. de Antônio Corradi, G. Nino, H.-H. Shin, H. Obinata, H. Ren, J. C. Crane, J. Tetreault, J. Guan, J. W. Garrett, J. D. Kaggie, J. G. Park, K. Dreyer, K. Juluru, K. Kersten, M. A. B. C. Rockenbach, M. G. Linguraru, M. A. Haider, M. AbdelMaseeh, N. Rieke, P. F. Damasceno, P. M. C. e Silva, P. Wang, S. Xu, S. Kawano, S. Sriswasdi, S. Y. Park, T. M. Grist, V. Buch, W. Jantarabenjakul, W. Wang, W. Y. Tak, X. Li, X. Lin, Y. J. Kwon, A. Quraini, A. Feng, A. N. Priest, B. Turkbey, B. Glicksberg, B. Bizzo, B. S. Kim, C. Tor-Díez, C.-C. Lee, C.-J. Hsu, C. Lin, C.-L. Lai, C. P. Hess, C. Compas, D. Bhatia, E. K. Oermann, E. Leibovitz, H. Sasaki, H. Mori, I. Yang, J. H. Sohn, K. N. K. Murthy, L.-C. Fu, M. R. F. de Mendonça, M. Fralick, M. K. Kang, M. Adil, N. Gangai, P. Vateekul, P. Elnajjar, S. Hickman, S. Majumdar, S. L. McLeod, S. Reed, S. Gräf, S. Harmon, T. Kodama, T. Puthanakit, T. Mazzulli, V. L. de Lavor, Y. Rakvongthai, Y. R. Lee, Y. Wen, F. J. Gilbert, M. G. Flores, and Q. Li. "Federated learning for predicting clinical outcomes in patients with COVID-19." In: *Nature Medicine* 27.10 (Oct. 2021), pp. 1735–1743. ISSN: 1546-170X. DOI: 10.1038/s41591-021-01506-3.

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

[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] 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]`.

[9] 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`.

[10] 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.

[11] 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`.

[12] 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]`.

[13] A. Khare, A. Agrawal, A. Annavajjala, P. Behnam, M. Lee, H. Latapie, and A. Tumanov. *SuperFedNAS: Cost-efficient federated neural architecture search for on-device inference*. 2024. arXiv: `2301.10879 [cs.LG]`.

[14] H. Kitano. "Designing Neural Networks Using Genetic Algorithms with Graph Generation System." In: *Complex Syst.* 4 (1990).

[15] H. Liu, K. Simonyan, and Y. Yang. *DARTS: Differentiable architecture search*. 2019. arXiv: `1806.09055 [cs.LG]`.

[16] 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.

[17] 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.

[18] G. Miller, P. Todd, and S. Hegde. "Designing Neural Networks using Genetic Algorithms." In: Jan. 1989, pp. 379–384.

[19] G. Öcal and A. Özgövde. "Network-aware federated neural architecture search." In: *Future Generation Computer Systems* 162 (2025), p. 107475. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2024.07.053.

[20] 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].

[21] 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.

[22] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean. "Efficient neural architecture search via parameters sharing." In: *Proceedings of the 35th international conference on machine learning*. Ed. by D. Jennifer and K. Andreas. Vol. 80. Proceedings of machine learning research. PMLR, July 2018, pp. 4095–4104.

[23] 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.

[24] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen, and S. Bakas. "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data." In: *Scientific Reports* 10.1 (July 2020), p. 12598. ISSN: 2045-2322. DOI: 10.1038/s41598-020-69250-1.

[25] 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.

[26] 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.

[27] M. Tenorio and W.-T. Lee. "Self organizing neural networks for the identification problem." In: *Advances in neural information processing systems*. Ed. by D. Touretzky. Vol. 1. Morgan-Kaufmann, 1988.

[28] 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.

[29] 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].

[30] 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.

[31] 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].

[32] D. Yao and B. Li. "PerFedRLNAS: One-for-all personalized federated neural architecture search." In: *Proceedings of the AAAI Conference on Artificial Intelligence* 38.15 (Mar. 2024), pp. 16398–16406. DOI: 10.1609/aaai.v38i15.29576.

[33] D. Yao, L. Wang, J. Xu, L. Xiang, S. Shao, Y. Chen, and Y. Tong. "Federated model search via reinforcement learning." In: *2021 IEEE 41st international conference on distributed computing systems (ICDCS)*. 2021, pp. 830–840. DOI: 10.1109/ICDCS51616.2021.00084.

[34] J. Yuan, M. Xu, Y. Zhao, K. Bian, G. Huang, X. Liu, and S. Wang. *Federated neural architecture search*. 2022. arXiv: 2002.06352 [cs.LG].

[35] 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.

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

[37] 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.