SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis, Master's Thesis, ... in Informatics

Thesis title

Author

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis, Master's Thesis, ... in Informatics

Thesis title

Titel der Abschlussarbeit

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

Acknowledgments

Abstract

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 [Kai+21]. It has yielded methods for finding architectures that deal with the challenges imposed by the FL setting.

Research into NAS has grown rapidly [Whi+23] since it was popularized by [ZL17a]; consequently, literature on its application to FL has grown. The last survey on NAS applied to FL compared approaches of four papers [ZZJ21]. 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 [McM+17], 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 [Tan+23] 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 [Whi+23], [Avv+25]) 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 call *FedNAS*. In this chapter we provide a motivation for the study of FedNAS and the techniques used to adapt NAS to the FL setting. Our motivation leads us to our research questions and finally, our proposed methodology to address them.

1.1 Motivation

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

The goal of NAS is the automation of the laborious architecture engineering process responsible for so many of the advances made in Deep Learning in recent years [EMH19]. During the early stages of the most recent wave of NAS research initiated by [ZL17b] in 2017, architectures were predominantly searched using black-box optimization techniques [ZL17b]. In the following years, significant effort was directed towards reducing the computational burden with great success. NAS was no longer a technique only affordable to those with access to vast swathes of GPUs, but could now be done feasibly on a single GPU [TODO: cite GPU hours reduction]. Accessibility to NAS helped power a wide range of experimentation and ultimately resulted in techniques for finding architectures with better accuracy, smaller size and faster training convergence. Architectures found via NAS occupied the top spot of several benchmarks for 90% of the time [TODO: cite].

FL on the other hand has become a viable privacy-enhancing choice for Collaborative Distributed Machine Learning [cite]. Ever more valuable training data is collected on edge devices like smartphones (referred to as *clients*), but collecting privacy-sensitive data in a central location greatly increases the risk of exploitation by malicious actors. FL was invented to make use of decentral training data without it ever having to leave the device it is being generated on — thereby enhancing the security of that data. Instead of collecting data to a central location and training a model on it there, *clients* train a model on their local data and coordinates weight updates to the model through a central server. FL "*embodies the principles of focused collection and data minimization, and can mitigate many of the systemic privacy risks and costs resulting from traditional, centralized* *machine learning*" [Kai+21]. Since its inception in 2016, FL has gotten a great deal of research attention and is used in production for various purposes [TODO: cite].

Inevitably, users of FL expect to make use of 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 particularily 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 [TODO: cite]. Such architectures are ideal for deployment on the often resource-constrained clients in the FL setting. [Kai+21] notes that predefined architectures may not be the optimal choice when user-generated data is not visible to model developers. They note that predefined architectures may contain components redundant for specific data sets or perform poorly on non-i.i.d. data. Allowing architecture search to take place on the clients, where the architecture can be adjusted according to the data and distribution present, could yield architectures better-suited for the task at hand and several FedNAS methods have shown to deal well with non-i.d.d. data [TODO: cite]. Architectures could even be personalized to each client's computational resource budget in a second stage.

However, research in NAS methods has focused on a central NAS setting and the methods rely on assumptions that do not hold for the federated setting. Assumptions that do not hold in the FL setting include an abundance of computational resources, homogeneinity of hardware for training and deployment, high availability of worker nodes, access to the entire dataset, access to the data distribution, etc. [Kai+21]. This makes most centralized NAS methods unfeasable for direct application in the FL setting. Adopting NAS methods in the FL setting requires adapting them to the FL setting.

For example, naively training a one-shot supernet with a large search space by having each client train the entire supernet and aggregate weights of the entire supernet works for the cross-silo setting [HAA21], but doesn't translate to the cross-device setting. Clients in the cross-device setting generally have less compute resources and such a training scheme could take weeks or months to complete [TODO: make realistic]. Instead, in one FedNAS method [DLF22], researchers have opted to reduce the compute burden on clients by sending randomly sampled subspaces to clients for training. In another [TODO: cite FINCH].

Another problem arises when using standard FedAvg to average the architecture weights α for DARTS-based supernets. Clients may tend towards different architectures, but averaging the architecture weights may select for an architecture that is not favored by any client. In [Wei+24] this is addressed by aggregating architecture weights into a probability distribution that can be used to sample likely architectures in the next communication round.

Typically, when NAS methods are run in a central environment, they implicitly assume that the machines that NAS is performed on is homogenous and contribute equally to the architecture search. This assumption does not hold in the FL setting where the variance between compute resources of clients can be very large. Consequently, higher-end clients will tend to be used in training more and bias the searched architecture. [ÖÖ25] adapts a NAS method to overcome this problem by by grouping clients into clusters according to their computational speed and network bandwidth. Small models get trained on low-end client clusters and larger models get trained on high-end client clusters.

Compared to performing NAS centrally, the FL setting can seem antagonistic. The conditions of the FL setting break many NAS methods and require them to be adapted. While the focus of the NAS community at large is on finding ways to perform NAS faster and finding better and smaller architectures, a key challenge of FedNAS methods lies in making use of NAS methods robust. It has been noted before that NAS methods are notoriously fiddly and hence finding ways to make NAS work in FL may just provide means to do NAS more robustly in general. To find out how NAS techniques are made robust we need to study their adaptations. Understanding how NAS methods need to be transformed for the FL setting can make future transfer of NAS methods to FL easier.

1.2 Research Questions

Making use of either NAS or FL on its own is a complex task and combining them introduces even more possible knobs to turn on the system as a whole. There are many NAS methods to choose from and many challenges the FL setting imposes that can be optimized — surmounting in a vast amount of possible choices to make. This leads us to our first research question:

(RQ1) Which FedNAS methods exist?

Most FedNAS methods focus on a small subset of the challenges imposed by the FL setting [TODO: cite/verify], resulting in a vast array of different adaptation techniques developed. Each FedNAS method uses a set of techniques to adapt a NAS method to FL. There is overlap between some of the adaptation techniques used by FedNAS methods, but most sets of adaptation techniques are disjoint. Nonetheless, most literature does not draw clear boundaries around adaptation techniques, leading us to our second research question:

(RQ2) How can we identify and cleanly separate out adaptation techniques?

Each adaptation technique pushes the system as a whole in a direction with regards to optimality towards FL challenges. Identifying this direction for each adaptation technique is important for informing trade-offs and picking Pareto optima. This leads us to our third research question:

(RQ3) Which FL challenges do the adaptation techniques address at what magnitude?

Adaptation techniques are heavily dependant on the targeted FL challenge and used NAS method. Naturally, this does not allow for most adaptation techniques to work together. Additionally, some adaptation techniques are incompatible due to other reasons [TODO: what other reasons]. We would still like to know which existing adaptation techniques can be combined how, possbily paving the way for creating more powerful FedNAS methods tailored towards specific use cases. This leads us to our third research question:

(RQ4) How can adaptation techniques be composed? How can we choose a set of adaptation techniques based on a use case?

Once we have made it easy to mix and match adaptation techniques and understand in which direction they push the system, we can potentially identify combinations that could improve upon the status quo for selected subsets of FL challenges. Therefore our final research question is:

(RQ5) Which combinations of adaptation techniques are particularily promising for certain sets of challengs in the FL setting?

1.3 Methodology

To tackle RQ1 we propose a systematic literature review with a PRISMA-style table for literature inclusion and exclusion. Once the set of literature to include is fixed, we tackle RQ2 by performing systematic coding of adaptation techniques according to ... [TODO: cite] and provide a concept definition together with discriminant rules that clearly separate one adaptation technique from others.

For RQ3 we make use of prior work on a taxonomy of FL challenges. We will go over all adaptation techniques and discuss how the technique works towards, against or has no effect towards overcoming each of the FL challenges. The end result will be a table with the techniques on the y-axis and the FL challenges on the x-axis.

For RQ4 we propose building an is-compatible-with relation on the set of adaptation techniques. We achieve this by cross-examing each adaptation technique with every other one. To reduce the number of comparisons we make use of the taxonomy of NAS methods provided in [Whi+23] and place each adaptation tecnique in one or multiple of the NAS method bins. Only adaptation techniques within the same bin need to be compared.

Finally, we make use of the is-compatible-with relation and our table discussing the FL challenges for each to select a few promising adaptation technique combinations and discuss their potential compared to the state of the art.

Abbreviations

List of Figures

List of Tables

Bibliography

[Avv+25] 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`.

[DLF22] L. Dudziak, S. Laskaridis, and J. Fernandez-Marques. *FedorAS: Federated architecture search under system heterogeneity*. 2022. arXiv: `2206.11239 [cs.LG]`.

[EMH19] 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.

[HAA21] 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]`.

[Kai+21] 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]`.

[McM+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 Z. J. Singh Aarti. Vol. 54. Proceedings of machine learning research. PMLR, Apr. 2017, pp. 1273–1282.

[ÖÖ25] 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`.

[Tan+23] 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.

[Wei+24] 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.

[Whi+23] 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].

[ZL17a] B. Zoph and Q. V. Le. *Neural architecture search with reinforcement learning*. 2017. arXiv: 1611.01578 [cs.LG].

[ZL17b] B. Zoph and Q. V. Le. *Neural architecture search with reinforcement learning*. 2017. arXiv: 1611.01578 [cs.LG].

[ZZJ21] 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.