Software Engineering Assignment

Baichuan Huang, baichuan.huang@eit.lth.se

Introduction to My Research and Topic Area:

I work mainly on the Internet of Things (IoT), mobile health, and edge Artificial Intelligence (AI). Next, I will introduce the detailed research topics I have done:

- **Self-Supervised Learning** [**Huang et al., 2023b**]: In this paper, we mainly exploit the self-supervised learning approach to solve the data-shortage problem in the current medical domain, especially in the epilepsy detection domain. On the one hand, training deep learning networks requires a large amount of data, especially the scarcity of positive seizure data. On the other hand, the monitoring system for mobile seizure detection is always on resource-constrained wearable devices. Our method exploits the synthetic seizure data based on autoencoder, rather than the real seizure data, and does not need to retrain the model, hence incurring no energy overhead.

- **Lightweight Learning** [Huang et al., 2023a]: In this paper, we propose a lightweight machine learning framework for real-time epilepsy monitoring on the wearable Sensor-Dot (SD) of Byteflies. Our proposed framework is based on power features in certain frequency bands of the Electroencephalogram (EEG) signals.

- **Forward-Only Algorithms** [Aminifar et al., 2024]: In this paper, we propose a lightweight inference scheme specifically designed for Deep Neural Networks (DNNs) trained using the Forward-Forward algorithm. Forward-Forward algorithms, as the forward-only algorithm, are the efficient alternatives to Backpropagation (BP). Because BP has the backward pass, which consumes large amounts of energy, memory, and resources, while forward-only algorithms only have the forward passes, no backward passes.

- **Neural Architecture Search (NAS)**: In the submitted manuscript, we mainly focus on searching the DNN architectures for distributed Internet of Things (IoT) systems. We partition the searched architecture and offload the partial DNN from IoT to distributed mobile/server nodes, to optimize the resource consumption such as energy, latency, and memory.

Insights from Robert's lectures:

1. **The Gap Between Algorithms and Software** : In a real-world AI system, only a small fraction is the AI code in terms of algorithm level [Sculley et al., 2015]. Software engineering is based on computer science but with more aspects including people, business, maintenance, scale, ethics, and so on, which is also applied to software engineering for machine learning (SE4ML). *The insights from this glance at real-world machine-learning products motivate me to transfer the algorithm proposer to a problem solver by machine learning techniques.* **First**, I used to implement the quick-run code for easy testing and validation of my idea, which is not good for extension and scalability. Especially in the ablation study with lots of experiments, it is hard and not clear to point out the tiny change in running scripts. From Robert's lectures, I began to think

about how I make my code scalable when I want to delete or add some components or ideas I proposed. **Second**, Version Control Software (VCS) is very important in software development, as well as in algorithms development. Most of the time, I just update the code locally, without the Git tools for tagging or recording. In such cases, at some point, I just copy the code file and name the updated information in the suffix. However, this way was very confusing when I came back to look at it several days later. Thus, a clear and trackable coding history (e.g., orthogonality and reversibility) is very important, which I will seriously consider in my later coding life. **Third**, the connectivity among different projects is not smooth. Usually, I develop the new project from the very beginning, repeating some functions to some extent. It not only wastes lots of time but also reduces the connectivity if more than 2 projects need to be combined. In addition, the complex code interface will also make the project hard to implement by other developers. Based on Robert's lectures, I began to think about how to make my coding a professional software process.

2. **Behavioral Software Engineering (BSE)**: I learned from Robert's lectures that people are not rational. Therefore, in addition to advanced techniques, the developed software should also pay attention to the human beings themselves, such as psychology and social science aspects. Indeed, in my research, at least, I do not pay any attention to behavioral aspects. Instead, I only focus on precision/accuracy, etc., which is far away from the real-world application. For instance, in the epilepsy monitoring application, the metrics-Recall and Precision-are in the trade-off situation. High recall with low precision is preferable because all the seizures will be detected although with certain falsely detected seizures. Besides, considering social aspects, e.g., the burden on emergency medical systems, a low precision is not optimal because of the wasted emergency resources by these falsely detected seizures. Therefore, when designing machine learning algorithms, the psychological and social aspects should be taken into consideration.

Insights from Lenberg's lectures in SAAB:

Generally, behavioral software engineering (BSE) is critical at Saab ATM. BSE is a combination of software engineering and psychology. Although the survey from SAAB said that foresee software engineering relevance changing with the increased integration of AI/ML in operation is not that positive, SAAB will continue to over-invest in AI solutions. From my point of view, I think the insights from the SAAB presentation make me think about how I can integrate AI/ML into my daily research and the final output.

1. **Changing with the increased integration of AI/ML in my operation**: I would say that nowadays LLMs [Brown et al., 2020, Touvron et al., 2023, Anil et al., 2023], such as Chatgpt [Achiam et al., 2023], become very useful AI/ML tools that help me a lot. In my daily operations, for example, the quick verification of an initial idea, especially with a new library, LLMs usually produce a not-bad output. Although the code from LLMs sometimes has errors, it is efficient because of several tests performed by people who do not want to be deeply involved in the technical documents. Similarly, LLMs could give me certain broad new ideas in the total new domain, which I could track to find the related papers. Especially in the AI/ML domain, knowledge explodes every day, even every hour, and it is not easy to follow every small progress.

2. **Changing with the integration of AI/ML in my products**: In my research topic, AI/ML is highly integrated into my algorithms. For *self-supervised learning*, I focus on how to avoid relying on large amounts of training data. For *lightweight learning*, I focus on deploying the deep learning model to IoT devices. For *forward-only algorithms*, we consider the alternatives to BP but still exploit the AI method. Moreover, for *neural architecture search*, we focus on finding the optimal deep learning architecture for downstream classification tasks. Generally speaking, traditional feature engineering is more likely to be the manually model-designed method; AI/ML is more likely to be the way that the human brain performs. Therefore, I agree with the point presented in this lecture, which is that more and more AI/ML should be integrated into daily life.

Paper 1: A Combinatorial Approach to Hyperparameter Optimization:

This paper [Khadka et al., 2024] is cited from the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI, 2024.

1. **Core Ideas**: Hyperparameter Tuning/Optimization in Machine Learning is vital to the final classification performance. Different hyperparameters such as neural network architectures, the learning rate, the pruning rate, etc., will output different convergent results. However, the traditional hyperparameter search method including grid search, random search, and Bayesian optimization is not efficient and scalable. In this paper, a t-way test, namely a combinatorial software testing method, is introduced into the hyperparameter optimization. T-way testing in hyperparameter optimization can significantly reduce the number of hyperparameters while preserving effectiveness in dealing with high-dimensional neural network training problems where the exhaustive search for hyperparameters is impractical.

2. **Relates to My Research**: The core idea of t-way testing derived from software testing is very useful to be efficient search hyperparameters in deep learning. In my past research experience, I usually did not do exhaustive hyperparameters optimization, e.g., in my papers [Huang et al., 2023b, Huang et al., 2023a, Aminifar et al., 2024]. I used to pick several random hyperparameters and use the best option based on validation datasets. Especially in the training-intensive algorithms, the exhaustive hyperparameters optimization consumes large amounts of time, energy, and workload. I think the idea of the t-way testing method is very useful for my later research because I can at least use this method to find a better solution with acceptable efforts.

3. **Benefit From This Paper and Fit Into Project**: *First*, regarding how paper [Khadka et al., 2024] could help improve the larger AI-intensive software project, the t-way testing approach for hyperparameters optimization contributes to saving the time and energy to develop the AI algorithms. Moreover, the t-way testing approach is good for continuing learning because the models need to be changed constantly according to the larger AI-intensive software project progressing. *Second*, regarding how my WASP research would fit into the larger AI-intensive software project, the research in NAS could search for the optimal architectures with the optimal hyperparameters with the aid of the t-way testing approach. Therefore, my WASP research is also useful to the larger AI-intensive software project.

4. **Potential Adapted/Changed**: My research in NAS can incorporate the core idea—the t-way testing from paper [Khadka et al., 2024]. In my NAS, I use the evolutionary method with the steps of population initialization, fitness evaluation, selection, crossover, mutation, and new population formation. The potential adapted/incorporated step could be the t-way testing in selection, crossover, and mutation, to reduce the number of populations in evolution.

Paper 2: Engineering Carbon Emission-aware Machine Learning Pipelines: This paper [Husom et al., 2024] is cited from the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI, 2024.

1. **Core Ideas**: Nowadays, AI/ML consumes lots of energy in its training and daily usage, especially influencing the environment, mainly concerning carbon emissions. This paper proposed an ML pipeline, named CEMAI, to monitor carbon emissions through the whole process of the deep learning model, from the data collection to model training, and to the deployment. The CEMAI guides the ML development process, reaching a balance between superior performance and minimal carbon emissions.

2. **Relates to My Research**: Although my previous research [Huang et al., 2023b, Huang et al., 2023a, Aminifar et al., 2024] does not consider the lifetime of the deep learning model, these papers focus on tiny machine learning in terms of algorithms level. This paper [Husom et al., 2024] could further monitor the lifetime of deep learning models from data collection to model training and to model deployment. Moreover, in [Aminifar et al., 2024], we also focus on the inference of the deep learning model to reduce carbon emissions because inference occupies around 60% of the total machine learning energy used at Google. Usually, the model is trained one time but the inference needs to be run all the time. Therefore, CEMAI is a general energy monitor tool for my research.

3. **Benefit From This Paper and Fit Into Project**: *First*, CEMAI [Husom et al., 2024] will benefit the larger AI-intensive software project in terms of reducing energy consumption, especially in the current LLMs domain. For instance, GPT-3 consumes over 1000 megawatt hours for training [Patterson et al., 2021]. This amount of power consumption can support a small town for a day. With the aid of CEMAI, the machine-learning pipeline could be conducted in an engineering carbon emission-aware way. *Second*, as I mentioned before, my research can be a good compensation to CEMAI because [Huang et al., 2023b] focuses on self-supervised learning and [Aminifar et al., 2024] focuses on early-existing techniques. At the same time, our NAS paper aims to design a lightweight architecture with comparable classification performance. Our research combined with CEMAI could contribute to the larger AI-intensive software project.

4. **Potential Adapted/Changed**: Inspired by CEMAI [Husom et al., 2024], I think at least I can potentially adapt/change my own research to make AI engineering in the project [Huang et al., 2023b, Huang et al., 2023a, Aminifar et al., 2024] in the below aspects: (1) considering the energy consumption in data preprocessing and evaluating model performance, as well as the training model; (2) considering the carbon emissions-driven lifecycle analysis; (3) considering the AI engineering dimensions (e.g., energy

measurement, carbon emissions measurement). Currently, I only optimize the training process and the inference process. Actually, in the real-world developing step, the data collection, data preprocessing, and data storage also consume lots of energy. Moreover, the debugging and tuning of the deep learning model occupy energy. Furthermore, in the deployment step, the number of requests and the real-world infrastructure influence energy consumption. As an AI/ML developer, I should also take these factors into consideration and be aware of them from the initial algorithm level.

References

[Achiam et al., 2023] Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

[Aminifar et al., 2024] Aminifar, A., Huang, B., Abtahi, A., and Aminifar, A. (2024). Lightff: Lightweight inference for forward-forward algorithm. *the European Conference on Artificial Intelligence (ECAI)*.

[Anil et al., 2023] Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., et al. (2023). Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

[Brown et al., 2020] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

[Huang et al., 2023a] Huang, B., Abtahi, A., and Aminifar, A. (2023a). Lightweight machine learning for seizure detection on wearable devices. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–2. IEEE.

[Huang et al., 2023b] Huang, B., Zanetti, R., Abtahi, A., Atienza, D., and Aminifar, A. (2023b). Epilepsynet: Interpretable self-supervised seizure detection for low-power wearable systems. In *2023 IEEE 5th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 1–5. IEEE.

[Husom et al., 2024] Husom, E. J., Sen, S., and Goknil, A. (2024). Engineering carbon emission-aware machine learning pipelines. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*, pages 118–128.

[Khadka et al., 2024] Khadka, K., Chandrasekaran, J., Lei, Y., Kacker, R. N., and Kuhn, D. R. (2024). A combinatorial approach to hyperparameter optimization. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*, pages 140–149.

[Patterson et al., 2021] Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.-M., Rothchild, D., So, D., Texier, M., and Dean, J. (2021). Carbon emissions and large neural network training.

[Sculley et al., 2015] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., and Dennison, D. (2015). Hidden technical debt in machine learning systems. *Advances in neural information processing systems*, 28.

[Touvron et al., 2023] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.