AI-ANNE: (A) (N)eural (N)et for (E)xploration

Dennis Klinkhammer [1]

1 Statistical Thinking (Germany), www.statistical-thinking.de, info (at) statistical-thinking.de

DOI: 10.21105/joss.08039

Software

- Review
- Repository
- Archive

Editor: Øystein Sørensen

Reviewers:

- @samiralavi
- @kalpan80

Submitted: 16 March 2025
Published: 08 June 2025

Summary

Machine learning and deep learning are increasingly driving innovation across various fields (LeCun et al., 2015) and can also be transferred on microcontrollers (Ray, 2022), for example to process sensor data (Cioffi et al., 2020). Since microcontrollers have limited computational resources (Delnevo et al., 2023), training or developing neural networks on microcontrollers remains a complex challenge (Wulfert et al., 2024). However, MicroPython as programming language enables a resource-efficient implementation of neural networks on microcontrollers and provides insights into the fundamentals of neural networks in terms of explainable artificial intelligence (Haque et al., 2023; Meske et al., 2022), while coding from scratch in MicroPython also allows for a streamlined and tailored approach (Delnevo et al., 2023). AI-ANNE: (A) (N)eural (N)et for (E)xploration provides a framework that transfers neural networks from Python to MicroPython and enables the application of pre-trained TensorFlow and Keras models on microcontrollers as well as training directly on the microcontroller via forward and backward propagation. Furthermore, it enables users to explore the performance of neural networks while simultaneously the number of neurons and layers as well as the underlying activation functions can be adjusted easily in MicroPython, which makes it also suitable for didactic application (Collier & Powell, 2024; Meske et al., 2022; Scherer, 2016; Verma et al., 2022).

Statement of need

A previous investigation of skill requirements in artificial intelligence and machine learning job advertisements (Verma et al., 2022) served as a guide for the development of AI-ANNE: (A) (N)eural (N)et for (E)xploration. Therefore, it demystifies the mathematical principles underlying neural networks, allowing users to better understand the relationships between data, weights, biases and outputs (Frank et al., 2020; Scherer, 2016; Schmidt et al., 2020). This includes reducing the number of layers, optimizing the activation functions, or using specialized operations to reduce computational load (Sakr et al., 2021). Moreover, by developing the neural network directly in MicroPython, the model becomes highly portable and adaptable to various microcontroller architectures, such as the Raspberry Pi Pico or the Raspberry Pi Pico 2. Especially the step-by-step visibility in MicroPython helps users to identify and understand common challenges like vanishing or exploding gradients, while fostering a more intuitive grasp of how neural networks learn (Kong et al., 2022). Debugging the code in MicroPython further reinforces this knowledge and provides insights into how errors propagate and are corrected during training. This is in accordance with the primary goals of explainable artificial intelligence (Haque et al., 2023; Schmidt et al., 2020) and provides insights into artificial intelligence related responsibilities (Collier & Powell, 2024; Frank et al., 2020) as well as practical experiences (Li et al., 2021). As such, implementing neural networks on microcontrollers with AI-ANNE: (A) (N)eural (N)et for (E)xploration shall promote necessary innovation and problem-solving skills (Verma et al., 2022).

Basic elements

AI-ANNE: (A) (N)eural (N)et for (E)xploration includes the activation functions ReLU, Leaky ReLU, Tanh, Sigmoid and Softmax as pre-installed activation functions. Additional activation functions can be flexibly added in MicroPython. Furthermore, neurons, matrices and matrix transposition as well as the density of neural networks are already pre-programmed in MicroPython. A pre-installed confusion matrix makes it possible to evaluate the performance of different neural networks.

References

Cioffi, R., Travaglioni, M., Piscitelli, G., Petrillo, A., & De Felice, F. (2020). Artificial intelligence and machine learning applications in smart production: Progress, trends, and directions. *Sustainability*, *12*(2), 492. https://doi.org/10.3390/su12020492

Collier, C., & Powell, A. (2024). Data analyst competencies: A theory-driven investigation of industry requirements in the field of data analytics. *J. Inf. Syst. Educ.*, *35*(3), 325–376. https://doi.org/10.62273/spyc4248

Delnevo, G., Mirri, S., Prandi, C., & Manzoni, P. (2023). An evaluation methodology to determine the actual limitations of a TinyML-based solution. *Internet of Things*, *22*(100729), 100729. https://doi.org/10.1016/j.iot.2023.100729

Frank, M., Drikakis, D., & Charissis, V. (2020). Machine-learning methods for computational science and engineering. *Computation (Basel)*, *8*(1), 15. https://doi.org/10.3390/computation8010015

Haque, A. B., Islam, A. K. M. N., & Mikalef, P. (2023). Explainable artificial intelligence (XAI) from a user perspective: A synthesis of prior literature and problematizing avenues for future research. *Technol. Forecast. Soc. Change*, *186*(122120), 122120. https://doi.org/10.1016/j.techfore.2022.122120

Kong, S.-C., Cheung, W. M.-Y., & Zhang, G. (2022). Evaluating artificial intelligence literacy courses for fostering conceptual learning, literacy and empowerment in university students: Refocusing to conceptual building. *Comput. Hum. Behav. Rep.*, *7*(100223), 100223. https://doi.org/10.1016/j.chbr.2022.100223

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Li, G., Yuan, C., Kamarthi, S., Moghaddam, M., & Jin, X. (2021). Data science skills and domain knowledge requirements in the manufacturing industry: A gap analysis. *J. Manuf. Syst.*, *60*, 692–706. https://doi.org/10.1016/j.jmsy.2021.07.007

Meske, C., Bunde, E., Schneider, J., & Gersch, M. (2022). Explainable artificial intelligence: Objectives, stakeholders, and future research opportunities. *Inf. Syst. Manag.*, *39*(1), 53–63. https://doi.org/10.1080/10580530.2020.1849465

Ray, P. P. (2022). A review on TinyML: State-of-the-art and prospects. *J. King Saud Univ. - Comput. Inf. Sci.*, *34*(4), 1595–1623. https://doi.org/10.1016/j.jksuci.2021.11.019

Sakr, F., Berta, R., Doyle, J., De Gloria, A., & Bellotti, F. (2021). Self-learning pipeline for low-energy resource-constrained devices. *Energies*, *14*(20), 6636. https://doi.org/10.3390/en14206636

Scherer, R. (2016). Learning from the past-the need for empirical evidence on the transfer effects of computer programming skills. *Front. Psychol.*, *7*, 1390. https://doi.org/10.3389/fpsyg.2016.01390

Schmidt, P., Biessmann, F., & Teubner, T. (2020). Transparency and trust in artificial intelligence systems. *J. Decis. Syst.*, *29*(4), 260–278. https://doi.org/10.1080/12460125.2020.1819094

Verma, A., Lamsal, K., & Verma, P. (2022). An investigation of skill requirements in artificial intelligence and machine learning job advertisements. *Ind. High. Educ.*, *36*(1), 63–73. https://doi.org/10.1177/0950422221990990

Wulfert, L., Kuhnel, J., Krupp, L., Viga, J., Wiede, C., Gembaczka, P., & Grabmaier, A. (2024). AIfES: A next-generation edge AI framework. *IEEE Trans. Pattern Anal. Mach. Intell.*, *46*(6), 4519–4533. https://doi.org/10.1109/tpami.2024.3355495