The Journal of Open Source Software

NCALab: A Framework for Experimentation with Neural Cellular Automata

Henry J. Krumb[1¶], **Richard Sattel**[1], **Niklas Ihm**[1], **Jonathan Dewenter**[1], **Dennis Grotz**[1], and **Anirban Mukhopadhyay**[1]

1 Technische Universität Darmstadt, Germany ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 20 August 2025
Published: unpublished

Summary

Neural Cellular Automata (NCA) are lightweight iterative neural network models that can be employed in various image analysis tasks such as image segmentation, classification and generation. Initially proposed in 2020 (Mordvintsev et al., 2020), these models are recently getting attention thanks to their small size, their robustness and their overall versatility. In terms of accuracy, they are often on-par with state-of-the art models for the respective downstream task, while being orders of magnitude smaller in size. This is especially interesting in constrained settings, such as environments with limited compute resources. However, the training dynamics of NCAs are not yet fully understood, and there is potential for investigating practical tweaks to increase accuracy, reduce VRAM requirements and increase the overall training stability. `NCALab` provides a unified and extensible research framework for training and evaluating NCAs, conducting structured hyperparameter searches and prototyping applications that use NCAs for image analysis.

Neural Cellular Automata

Neural cellular automata (NCA) are a computational model combining the principles of Cellular Automata and Nneural Networks, allowing systems to learn and evolve over time. In NCAs, a grid of cells, each cell being capable of assuming a continuous, multi-dimensional state, changes based on a learned rule determined by neural networks rather than fixed algorithms. Typically, the inference is done in two steps, which are called *Perception* and *Update*. In the perception step, each cell's neighborhood is aggregated by applying multiple depth-wise convolutions. These convolutions can be learned, or hardcoded through static filters (e.g. Sobel or Laplace filters). In the *update* step, the residual cell update is computed by a multi-layer perceptron with a ReLU activation. This step is applied in a stochastic manner, meaning that only a portion of cells are activated in a single time step. The probability of the stochastic update is determined by a parameter called "fire rate", which is typically set to 0.5, activating only half of all cells in every time step. For more information, we want to point the reader to the comprehensive original paper on NCAs by Mordvintsev et al. (Mordvintsev et al., 2020).

Statement of Need

NCAs are recently gaining attention especially in medical imaging, where they are deployed for various modalities in different downstream tasks, including 3D prostate segmentation on MRI (Kalkhof et al., 2023) (Kalkhof & Mukhopadhyay, 2023), image registration (Ranem et al., 2024) or image synthesis (Kalkhof et al., 2024), (Kalkhof et al., 2025). In most cases, they outperform other Convolutional Neural Network or Vision Transformer architectures in terms of *model size* and robustness (Kalkhof et al., 2023), while yielding similarly accurate predictions.

Especially in constrained settings, such as radiology departments without access to GPU hardware, state-of-the art medical image segmentation is hardly possible with contemporary architectures based on Convolutional Neural Networks or Vision Transformers. NCAs, on the other hand, are a promising approach to enable AI support even in such settings. However, research of NCA models and training has just started a few years ago, and there is no *unified* framework or reference implementation for training, evaluation and systematic experimentation with NCAs. Further, there is currently no set of best practices for designing NCA models with respect to their hyperparameters, such as the number of neurons, hidden channels or fire rate.

A systematic analysis of different NCA hyperparameters and architectural variations is difficult, as the research code of NCA contributions is typically organized in individual repositories with different frameworks and coding styles for each downstream task under investigation. Code bases often follow different approaches, even though the underlying backbone architecture is in most parts identical. In most cases, the NCA architecture can be defined by the number of input channels, hidden channels and output channels and the weights of the trained network. Since there is currently no unified framework, deployment of NCA models in practical applications (or as part of other learning pipelines) remains difficult.

`NCALab` provides a uniform and easy-to-use code base for various downstream tasks with NCAs as a packaged Python module. Within minutes, researchers and practitioners will be able to create prototypes for their ideas, inspired by the example tasks provided in this code repository.

We further provide weights for example tasks to enable post-hoc analyses on pre-trained NCA models. To our knowledge, post-hoc tasks like transfer learning or uncertainty estimation were not evaluated for NCAs yet.

Features

`NCALab` provides dedicated models and example tasks for image analysis tasks, such as:

- Growing Neural Cellular Automata for emoji generation from a single pixel, akin to (Mordvintsev et al., 2020).
- Self-classifying MNIST digits, similar to the work of (Randazzo et al., 2020).
- Pixel-wise medical image segmentation of Endoscopic images on the Kvasir-SEG dataset (Jha et al., 2019).
- Multi-class medical image classification on subsets of the MedMNIST image dataset (Yang et al., 2023).

Until now, NCALab provides the following key features:

- Simplified creation, training and loading of NCA models for various image analysis tasks
- Streamlined grid search for model and training hyperparameters
- Tensorboard integration to monitor training progress
- k-fold cross validation
- Control over various hyperparameters of NCAs
- Finetuning by re-training the final layer of an NCA
- Visualization and animation of the NCA inference process
- Cascaded training of multi-scale NCAs for training on higher-resolution images
- Uncertainty heatmap generation

Ongoing Research

A conference paper utilizing `NCALab` was recently accepted for presentation in IPCAI 2025, and was published in the *International Journal of Computer-Assisted Radiology and Surgery* (Krumb & Mukhopadhyay, 2025). In this paper, `NCALab` is used to train models for image

The Journal of Open Source Software

segmentation and monocular depth estimation. The trained models are exported to C headers and ported to a microcontroller.

Dependencies and Tooling

`NCALab` mostly builds on pytorch (Paszke et al., 2019), numpy (Harris et al., 2020) and matplotlib (Hunter, 2007). Code quality is ensured by unit tests (pytest) and automated static code analysis through mypy and ruff. The project uses uv for fast and simplified dependency management. Code documentation is generated through Sphinx and is automatically uploaded to readthedocs. Releases of `NCALab` can be downloaded from the Python Package Index (pip).

Acknowledgements

This work is partially supported by Norwegian Research Council project number 322600 (Capsnetwork).

References

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Jha, D., Smedsrud, P. H., Riegler, M. A., Halvorsen, P., Lange, T. de, Johansen, D., & Johansen, H. D. (2019). *Kvasir-SEG: A Segmented Polyp Dataset* (No. arXiv:1911.07069). arXiv. https://doi.org/10.48550/arXiv.1911.07069

Kalkhof, J., Gonz'alez, C., & Mukhopadhyay, A. (2023). Med-NCA: Robust and Lightweight Segmentation with Neural Cellular Automata. *Information Processing in Medical Imaging*, 705–716.

Kalkhof, J., Kühn, A., Frisch, Y., & Mukhopadhyay, A. (2024). *Frequency-Time Diffusion with Neural Cellular Automata* (No. arXiv:2401.06291). arXiv. https://doi.org/10.48550/arXiv.2401.06291

Kalkhof, J., Kühn, A., Frisch, Y., & Mukhopadhyay, A. (2025). Parameter-efficient diffusion with neural cellular automata. *Npj Unconventional Computing*, *2*(1), 10. https://doi.org/10.1038/s44335-025-00026-4

Kalkhof, J., & Mukhopadhyay, A. (2023). M3D-NCA: Robust 3D Segmentation with Built-in Quality Control. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, *abs/2309.02954*.

Krumb, H. J., & Mukhopadhyay, A. (2025). eNCApsulate: Neural cellular automata for precision diagnosis on capsule endoscopes. *International Journal of Computer Assisted Radiology and Surgery*. https://doi.org/10.1007/s11548-025-03425-x

Mordvintsev, A., Randazzo, E., Niklasson, E., & Levin, M. (2020). Growing Neural Cellular Automata. *Distill*, *5*(2), e23. https://doi.org/10.23915/distill.00023

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). *PyTorch: An*

The Journal of Open Source Software

Imperative Style, High-Performance Deep Learning Library (No. arXiv:1912.01703). arXiv. https://doi.org/10.48550/arXiv.1912.01703

Randazzo, E., Mordvintsev, A., Niklasson, E., Levin, M., & Greydanus, S. (2020). Self-classifying mnist digits. *Distill*, *5*(8), e00027–002.

Ranem, A., Kalkhof, J., & Mukhopadhyay, A. (2024). NCA-Morph: Medical Image Registration with Neural Cellular Automata. *British Machine Vision Conference*, *abs/2410.22265*.

Yang, J., Shi, R., Wei, D., Liu, Z., Zhao, L., Ke, B., Pfister, H., & Ni, B. (2023). MedMNIST v2 - A large-scale lightweight benchmark for 2D and 3D biomedical image classification. *Scientific Data*, *10*(1), 41. https://doi.org/10.1038/s41597-022-01721-8