NOTES for WIVACE:

Slide 1: Hello everyone! I am Sanyam from India, currently doing Masters in Applied CS in Norway under supervision of prof Stefano Nichele at Østofold University College and OsloMetropoliton university. The title of our work is “Capturing emerging complexity in Lenia” where we explore emerging behaviour in open-ended system called EvoLenia.

Slide 2: These are contents for the rest of the presentation. I will try to quickly wrap up the presentation in 15 minutes however, if I extend, please feel free to stop me in between.

Slide 3: We will start with complexity and open-endedness as background and theoretical build up.

Slide 4: In Alife, solving a specific task is considered suboptimal and insufficient. Manual AI (or traditional DL) is extremely difficult to take these learned model and evolve further in order to reuse as many parameters. Which makes it non open-ended. Novelty search, sometimes better than task based, stochastic updates, and
hybrid approaches also seem to be driving complexification in literature. We ll see more about how complexification and open-endedness goes together?

Slide 5: Open-endedness, no tasks, sometimes there is no task helps to evolve smart and interesting metrics over a task landscape or distribution. Endless variation implies complexification because any fixed size of complexity would exhasut the existing interestingness or variation eventually.

Slide 6: Thererfor an unbounded complexity is required. It also includes other fundamental quesitons like sensitivity to initial conditions, adaptive mutations, selection pressure and fitness landscape. One way of thinking is agents and tasks can be co-evolved so tasks become increasingly complex wiht agents. OR. there can be another approach for example auto-telic where agents figure out what tassks to explore themselves. In a recent work, BioMaker CA, authors provide importance about heritable genetics and selectable phenotypes. Without Evolvability there would be no discovery, no new behaviour;

Slide 7: So to study such systems, we implemented EvoLenia using Lenia framework which is wide known CA environment recently.

Slide 8: Discrete cellular automata (e.g., Game of Life) involve grid-based cells transitioning between defined states via set rules, yielding emergent

patterns. In contrast, continuous cellular automata like Lenia use differential equations to create fluidic, unbounded patterns in a continuous time and space framework. The Kernel used in discrete CA consist of 1s and 0s for example Moore Neighborhood and Growth function also uses discrete space while in Lenia, Kernel and growht function both are part of Gasussian functions to smoothen it.

Slide 9: The update rule is pretty simple where new state is calculated by addition of the current state to the update grid. In simpler words, kernel is convolved over the current grid space and the resultant value is passsed to Gaussian growth function. Finally it gets multiplied with time delta to make every behaviour time shifted.

Slide 10: A quick description of Kernel and Growth function that are used in Lenia.

Slide 11: Now we will come to the proposed work and how to actually studying emergent complex behaviour in Lenia worked in this project.

Slide 12: We implemented standard genetic algorithm containing regular practices for example, population, fitness, mutation and no crossover. We will discuss only about the types of fitness functions used here. First is AutoEncoder based or compression based fitness. The idea is the latent bottleneck should provide a large reconstruction loss for the input behaviour which is too random while it is expected to have smaller reconstruction loss for the too orderly behaviour. However interestingness should be in betweem.

Slide 13: Another fitness that we implemeneted is variation over time. This fitness tracks the number of active cells in the board states over time while regulating with a threshold. Deviation is a measure of the randomness or differences in states of a system, and can be used to quantify the variation of complexity in a system over time. The VoT approach is particularly useful in identifying temporal patterns and trends in the system's behavior, which can be indicative of its underlying complexity.

Slide 14: The third and final fitness function that we devised is AEVoT or AutoEncoder variation over time, it simply reconstructs the behaviour input and then performs the VoT.

Slide 15: we used Roulette wheel selection, not used any means of crossover, and mutatiob by perturbation of the pixels.

Slide 16: Now we will come to results section and discuss results from

each fitness types. It provides summary of over 27 experiments in total for 500 generations.

Slide 17: These experiments are for VoT with population size 10, 500 generations, and a fixed size of mutation rate of 0.02 which means 2 pixels are perturbed in each generation at least. While we have Varying alive cell Threshold (0.1, 0.3, 0.5) and Varying Frames (All 100 frames, last 10th frame, every 10th frame).

Slide 18: These experiments are for AE with population size 10, 500 generations, and a fixed size of mutation rate of 0.02 which means 2 pixels are perturbed in each generation at least. While we have Varying Bottleneck Size (16, 32, 72) and Varying Frames (all 100frames , last 10th, every 10th)

Slide 19: These experiments are for AEVoT with population size 10, 500 generations, and a fixed size of mutation rate of 0.02 which means 2 pixels are perturbed in each generation at least. While we have Varying Threshold (0.1, 0.3, 0.5) and Varying Frames (all 100 frames, last 10th, every 10th) for the fitness. It is interesting to see how some configurations achieved a stability sooner while some experiments continue to evolve even at 500th generation. Based on this, such experiments are picked for longer gens.

Slide 20: We experimented longer runs with multiple known kernels. Results from those experiments are published online of which the URL is shared in the last slide of this presentation. However, among all those emerging behaviour, we could pick one particular experiment that shows significant improvement in robustness to the perturbations over generation and we identified such roaming bacteria like behaviour and others at some generation and further that even after perturbations generations yield same interesting behaviour over further genearations which means there is no progress over generations unless there is some interesting behaviour is identified. Hence usefulness of our platform.

Slide 21: Few challenges are introduced by Bert Chan in ALife 2021, to name a few, Automated Discovery, Emergent Agency, and Open-Endedness. We learned that starting with random kernel could not produce stable behaviour. Those random behaviour patterns are put on the EvoLenia portal. But choosing a known kernel could evolve stable and emerging complex behaviours for example ring forming bacteria. And yes Evoltion is time taking. The experimental specificaitions and requirements are also available on the portal.

Slide 22: Future scope of improvements can be mutating more interesting known kernels, that are also made available on the portal and github repository. The portal link will be shown in the end. Other models of Lenia, like particle Lenia, and others can be explored with this platform too. And finally, JAX is a very useful tool that can improve efficiency of the simulations, but adds a dependency on costly GPU.

Slide 23: Following are the major references that were used for this presentation.

Slide 24: This is the final slide. Thank you for keeping all attention to the presentation. And the portal is available here with the mentioned link.