Appendix 1

This section details the preliminary experiments conducted to determine the deterioration in NCA image reconstruction quality as channels are privatized. We measured the loss curves for reconstruction of a lizard emoji, alongside the regrowth capabilities of the NCA to damage. For all experiments the training was repeated once on a set seed for each privatization level, the regrowth experiments where repeated one-hundred times with the damage locations being fixed and identical for each privatization level.

For these experiments, we tested three different NCA types:

- **DummyVCA**: Where the sensing kernels channel dimension match the unmasked NCA channels. However, any kernel involved in sensing neighbourhoods (such as sobel filters) would receive a dummy vector of all zeros for in lieu of the privatized channels.
- **MaskedCA**: Where the sensing kernels channel dimension match the unmasked NCA channels. However, the channels of the output tensor corresponding to the privatized channels where masked with zeros after the sensing convolution.
- **ReducedCA**: Where the sensing kernels channel dimension where reduced to not include the privatized channels, and a truncated state excluding the privatized channels is passed to the sensing kernels. The state was later re-expanded with zeros t mathc the unmasked NCA channel dimensions.

The loss function used for training and measuring the reconstruction was the PixelwiseMSE:

$$PixelWiseMSE = \frac{1}{H\times W\times C}\sum_{i=0}^{H}\sum_{j=0}^{W}\sum_{k=0}^{C}\left(I(i,j,k)-\hat{I}(i,j,k)\right)^2 \quad (1)$$

Where H, W, W are the dimensions of the image, I is the reference image, and $\hat{I}$ is the final state of the NCA.

Preliminary experiment results

Figure 1 shows the loss curves of the three models (left to right) with different levels of privatization (0,4,8,12, top to bottom). For privatization levels 0 and 4 the three models are comparable, while at privatization level 8 the DummyVCA performed considerably worse.Finally, the ReducedCA performed best at privatization level 12. From privatization level 0 to 4 there seems to be no discernible change in the loss curves, afterward however, as more channels get privatized the models struggle to learn.

Figure 2 shows how the reconstruction quality (the mean final loss, on a running average of 10) on the lizard emoji decreases as channels are privatized. The data is noisy,

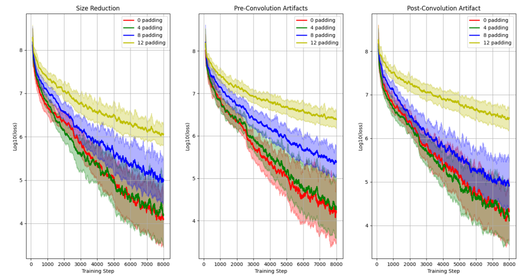

Figure 1: Comparison of the loss curves of the three models (left to right) with different levels of privatization (0,4,8,12, top to bottom. For privatization levels 0 and 4 the three models are comparable, while at privatization level 8 the DummyVCA performed considerably worse.Finally, the ReducedCA performed best at privatization level 12.

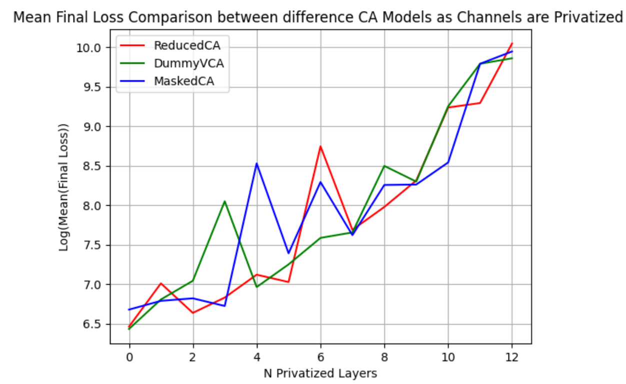

Figure 2: Mean final loss as compared to number of channels privatized for each of the three models. As channels are privatized, it can be seen that the reconstruction quality of the original image (mean final loss) decreases.

thus no proper comparison can be done between the three models, they can be considered to act the same.

Figure 3 shows the mean regrowth loss of the three models as they attempt to reconstruct the damaged lizard. As with the mean final loss, regrowth quality decreases as the number of channels are privatized. The data is once again noisy, even more so here. This is most likely due to the asynchronous nature of NCAs.

From the results of the preliminary experiments, we concluded that there is indeed deterioration in the quality of image reconstruction from the NCA as channels are privatized. However, the choice of architecture does not play a big role in the level of performance loss. Thus, we chose to go with the MaskedCA version for channel reduction as it was the easiest to implement.

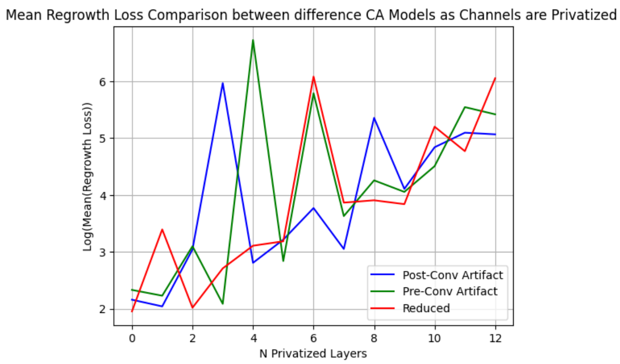

Figure 3: Mean regrowth loss as compared to number of channels privatized for each of the three models. As channels are privatized, it can be seen that the NCA struggles to regrow back to the original image.

Appendix 2

Mathematical details of the proposed models

In this section we define the proposed models with mathematical detail.

Cellular automaton framework We formalize our EngramNCA as a grid of cells $C \in \mathbb{R}^{H\times W\times N}$, where H and W are the grid dimensions, and N is the total number of channels. Each cell $c_{i,j}$ at position (i, j) contains a state vector that is partitioned into three components:

$$c_{i,j} = [v_{i,j}, h_{i,j}, g_{i,j}]$$

where $v_{i,j} \in \mathbb{R}^4$ represents the visible RGB-α channels, $h_{i,j} \in \mathbb{R}^{n_h}$ represents the public hidden channels, and $g_{i,j} \in \mathbb{R}^{n_g}$ represents the private gene channels. The dimensions satisfy $4 + n_h + n_g = N$.

GeneCA architecture GeneCA updates the public channels (visible and hidden) while preserving the private gene channels. For each cell $c_{i,j}$, the update rule is defined as:

$$[v^{t+1}_{i,j}, h^{t+1}_{i,j}] = [v^t_{i,j}, h^t_{i,j}] + \phi_{\text{GeneCA}}(\mathcal{P}(c^t_{i,j}), g^t_{i,j})$$

$$g^{t+1}_{i,j} = g^t_{i,j}$$

where $\mathcal{P}(c^t_{i,j})$ represents the perception vector derived from the cell's neighborhood, and ϕ_{GeneCA} is a neural network that computes the update. The perception function $\mathcal{P}$ applies convolution kernels to the grid:

$$\mathcal{P}(c^t_{i,j}) = [\text{Identity}(c^t_{i,j}), \text{Sobel}_x(c^t_{i,j}), \text{Sobel}_y(c^t_{i,j}), \text{Laplacian}(c^t_{i,j})]$$

where Identity, Sobel_x, Sobel_y, and Laplacian are convolution filters applied only to the visible and hidden channels. The neural network ϕ_{GeneCA} is structured as:

$$\phi_{\text{GeneCA}}(\mathcal{P}, g) = (W_2 \cdot \text{ReLU}(W_1 \cdot [\mathcal{P}, g] + b_1)) \cdot u_m \cdot l_m$$

where W_1, W_2, b_1 are learnable parameters and u_m, l_m are the asynchronous update mask and cell living mask respectively.

GenePropCA architecture The GenePropCA updates only the gene channels while preserving the public channels:

$$[v^{t+1}_{i,j}, h^{t+1}_{i,j}] = [v^t_{i,j}, h^t_{i,j}]$$

$$g^{t+1}_{i,j} = g^t_{i,j} + \psi_{\text{GenePropCA}}(\mathcal{P}(c^t_{i,j}), g^t_{i,j})$$

where $\psi_{\text{GenePropCA}}$ is a neural network with a similar architecture to ϕ_{GeneCA} but outputs updates for gene channels only:

$$\psi_{\text{GenePropCA}}(\mathcal{P}, g) = (V_2 \cdot \text{ReLU}(V_1 \cdot [\mathcal{P}, g] + d_1)) \cdot u_m \cdot l_m$$

where V_1, V_2, d_1 are learnable parameters and u_m, l_m are the asynchronous update mask and cell living mask respectively.

EngramNCA ensemble The EngramNCA ensemble combines both models in sequence. For a single update step:

$$c^{t+\frac{1}{2}}_{i,j} = [v^t_{i,j} + \Delta v^t_{i,j}, h^t_{i,j} + \Delta h^t_{i,j}, g^t_{i,j}]$$

where $[\Delta v^t_{i,j}, \Delta h^t_{i,j}] = \phi_{\text{GeneCA}}(\mathcal{P}(c^t_{i,j}), g^t_{i,j})$, followed by:

$$c^{t+1}_{i,j} = [v^{t+\frac{1}{2}}_{i,j}, h^{t+\frac{1}{2}}_{i,j}, g^t_{i,j} + \Delta g^t_{i,j}]$$

where $\Delta g^t_{i,j} = \psi_{\text{GenePropCA}}(\mathcal{P}(c^{t+\frac{1}{2}}_{i,j}), g^t_{i,j})$.

Training procedure The GeneCA is trained first with frozen gene channels. For each training iteration, we:

1. Sample a batch of B cells from pools corresponding to K different primitive morphologies 2. Initialize the gene channels of each seed cell with a unique binary encoding $E_k \in \{0, 1\}^{n_g}$ for primitive k 3. Run the GeneCA for T steps to grow the morphologies 4. Compute the loss using pixelwise MSE between the final visible channels and target images:

$$\mathcal{L}_{\text{GeneCA}} = \frac{1}{BHW}\sum_{b=1}^{B}\sum_{i=1}^{H}\sum_{j=1}^{W} ||v^T_{i,j} - \hat{v}_{i,j}||^2_2$$

After training GeneCA, its weights are frozen, and GenePropCA is trained to propagate and modify gene information to grow complex morphologies. The same loss function is used, but with target images representing complete morphologies rather than primitives.