CONTINUAL LEARNING with NEUROGENESIS

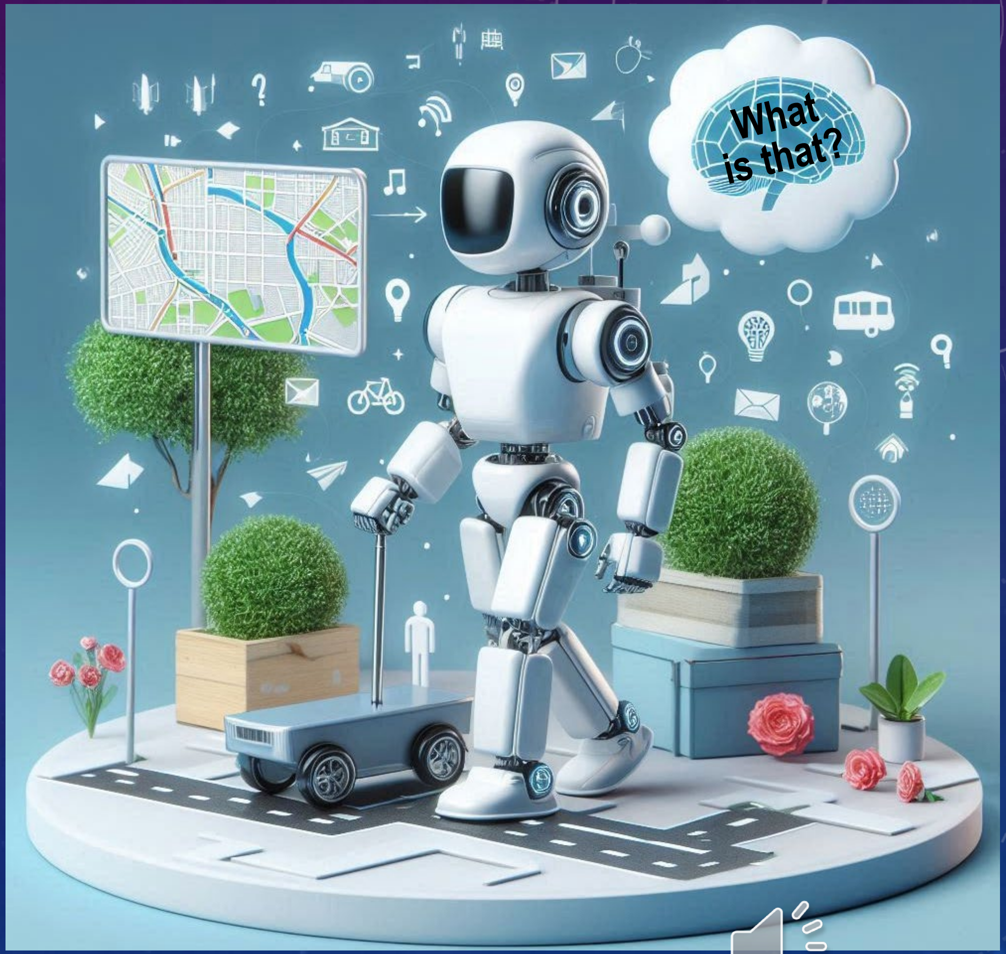
TIM DRAELOS

timdraelos@q.com

Motivated by prior Sandia work: Draelos, T., et al (2017). Neurogenesis Deep Learning, IJCNN.

WHAT IS CONTINUAL LEARNING?
(LIFELONG LEARNING, INCREMENTAL LEARNING)

- *An adaptive algorithm capable of learning from a continuous stream of information, with such information becoming progressively available over time and where the number of tasks to be learned (e.g. membership classes in a classification task) are not predefined. Critically, the accommodation of new information should occur without Catastrophic Forgetting or interference, where the process of learning new knowledge quickly disrupts previously acquired information).*
    - Parisi et al. *Continual Lifelong Learning with Neural Networks: a review*, 2019.



APPROACHES TO CONTINUAL LEARNING

- **Regularization (Weight and Function)**: Add penalty term to loss function to constrain the parameter updates, use old model to create new training samples.
  - Elastic Weight Consolidation (**EWC**)
  - Synaptic Intelligence (**SI**)
  - Learning without Forgetting (**LwF**) – Label new data with old model
- **Replay (Generative, Experience, Feature)**: Generate data, features, outputs from previous tasks to mitigate forgetting when learning new tasks.
  - Generative Replay (**GR**)
- **Optimization**: Manipulate the optimization program, gradient calculations, and/or loss landscape to learn new tasks without forgetting old tasks.
- **Representation**: Leverages sparse representations, pre-training, and self-supervised learning.
- **Architecture**: Dynamically utilize or expand the current model architecture for new tasks or data.
  - **Continual Learning with Neurogenesis (CLN) using Generative Replay**

CONTINUAL LEARNING with NEUROGENESIS (CLN)

- **Human Neuroscience Motivation**
  - Human brain has the potential to create new neurons (*neurogenesis*).
    - New neurons are used to help learn new tasks.
    - New neurons can die off if they are not utilized for new tasks.
  - Human brain can recall (replay) old information from their current neural networks
- **Key Design Elements** - assumes data used to trained the original classifier isn't available
  - ***Intrinsic Generative Replay***
    - Generate examples of old classes using weights from the current classifier in a variational autoencoder (VAE).
  - ***Neurogenesis***

Plasticity

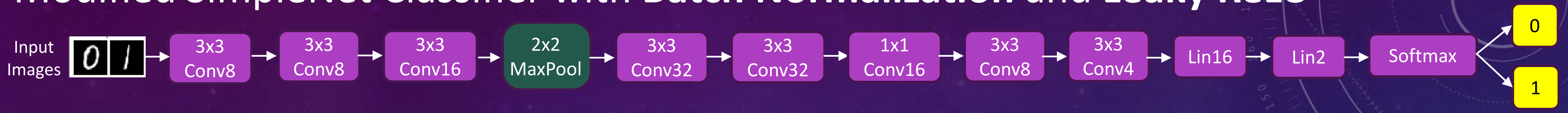
- Add new filters/nodes to each layer of a classifier to adapt to new classes of data.
- Train new weights using only new data to learn features of the new data.

Stability

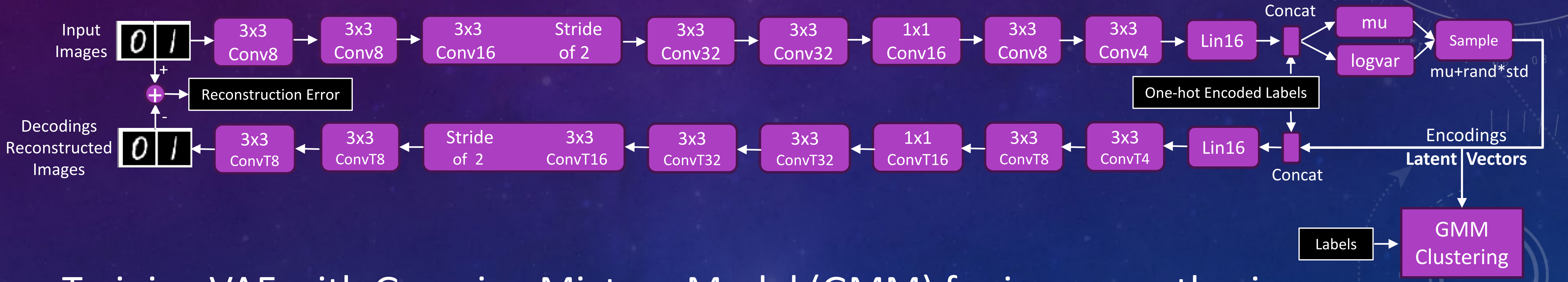
- Freeze old weights during training of new weights.
- Fine-tune the new classifier & VAE using all data, old and new.

ARCHITECTURE FOR 2-CLASS MNIST

- Modified SimpleNet Classifier with **Batch Normalization** and **Leaky ReLU**



- Class-Conditional Variational Autoencoder (VAE) **Batch Normalization in Encoder only**

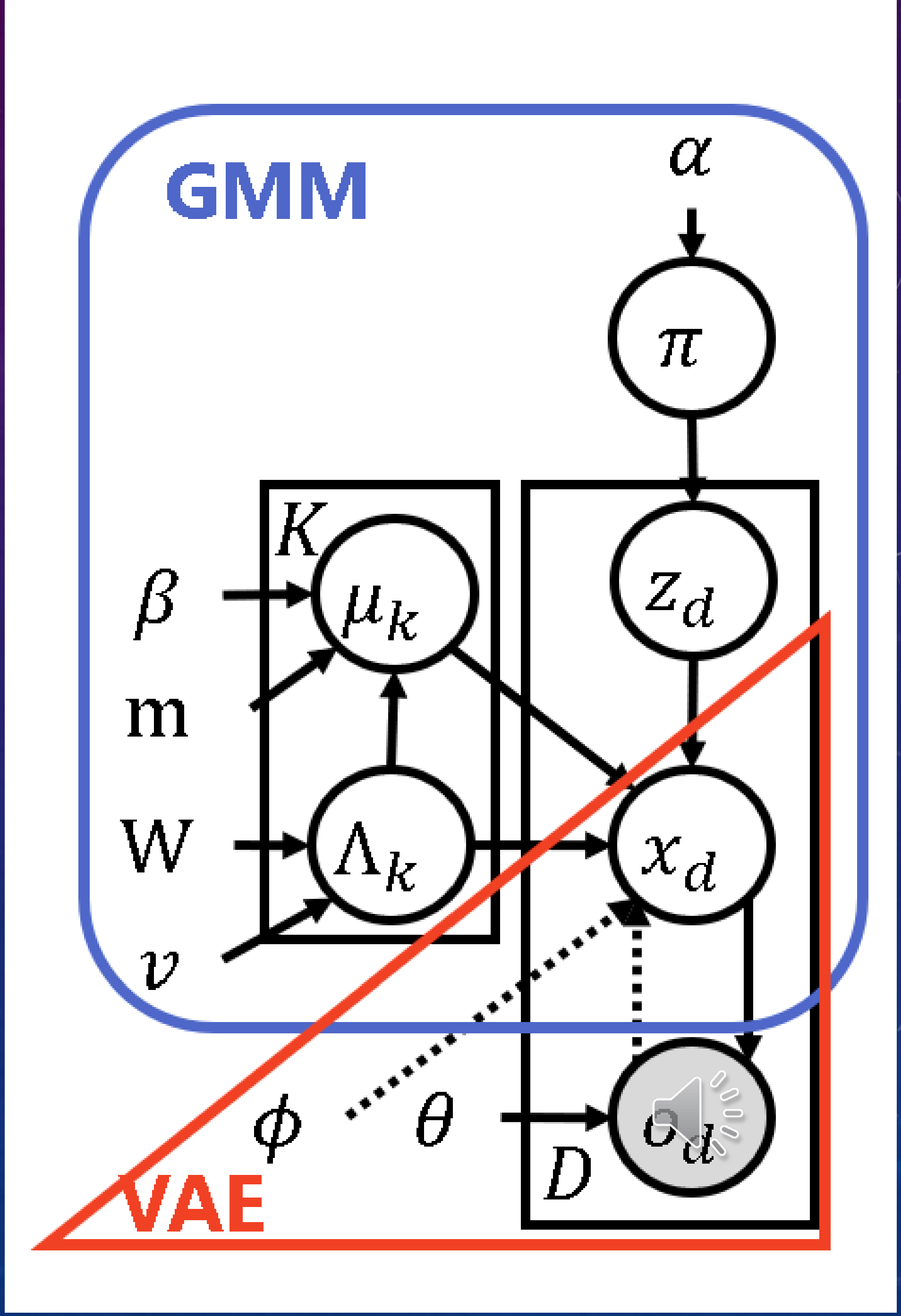


- Training VAE with Gaussian Mixture Model (GMM) for image synthesis
  - For *n* iterations:
    - Train VAE with GMM to minimize Reconstruction error and Kullback-Leibler (KL) divergence
    - Perform GMM Clustering using latent vectors and labels from VAE

TRAINING VAE WITH GMM

https://github.com/is0383kk/Pytorch_VAE-GMM

1. VAE estimates latent variable （x） and sends latent variables （x） and labels (y) from N classes to GMM.
   - x = output of VAE encoder
2. GMM clusters the latent variables （x） from the VAE and returns *mean* and *covariance* parameters of the N Gaussian distributions to the VAE.
   - Use Classifier to determine which label best matches each cluster
3. Repeat



CLN ALGORITHM, GIVEN A TRAINED CLASSIFIER, VAE & NEW TASK

Generative Replay

1. Generate old classes of data on which the Classifier has been trained
   - Create random latent vectors with class-conditional GMM statistics (mean and covariance matrix)
   - Synthesize class-conditional images with VAE decoder using latent vectors as input
   - Filter synthesized images using the current classifier and augment images if necessary

Neurogenesis

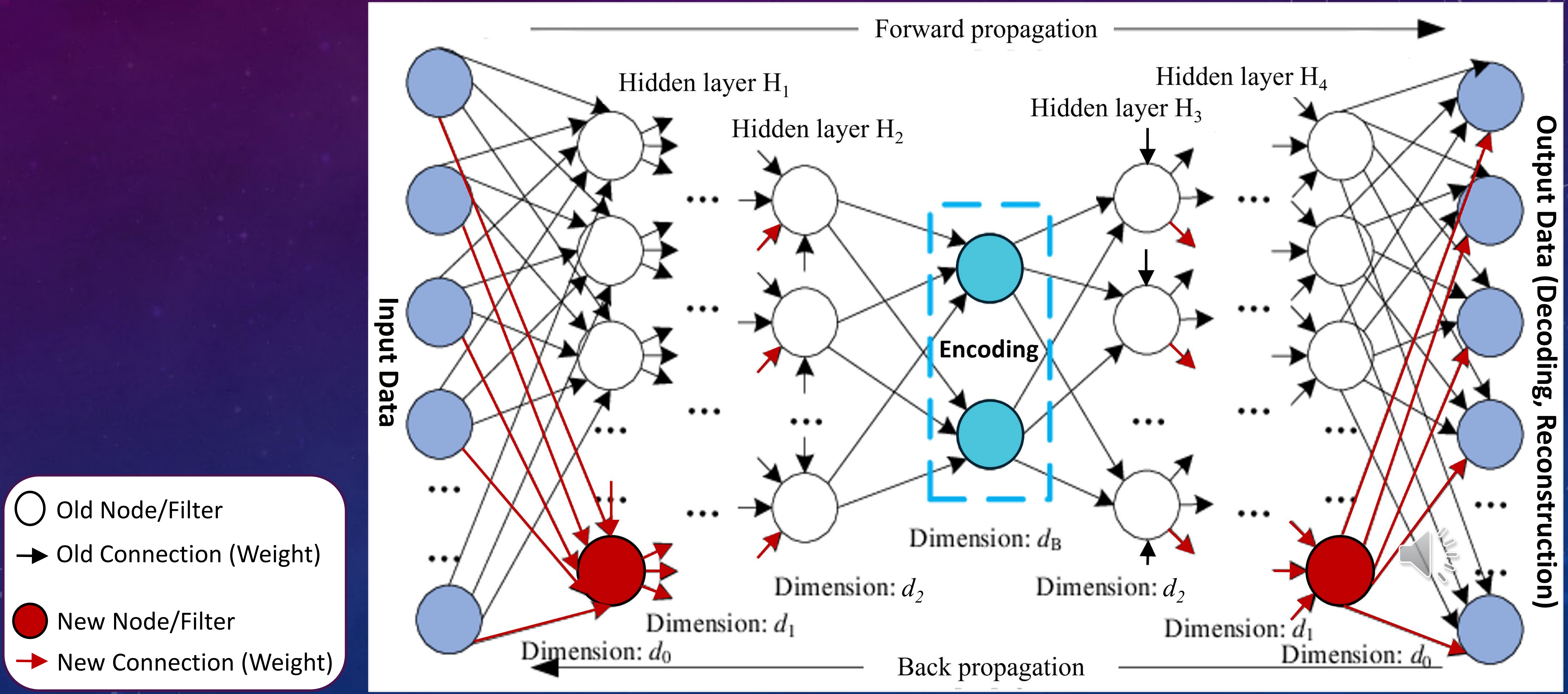
2. For each layer of the classifier except the output layer:
   - Add new filters/nodes to convolutional/linear layers of Classifier and VAE as needed
   - Use new class of data to train *New Feature Learning Autoencoder* (NFLAE)
     - while freezing old weights
   - Transfer NFLAE encoder weights to classifier

Train New Classifier and VAE

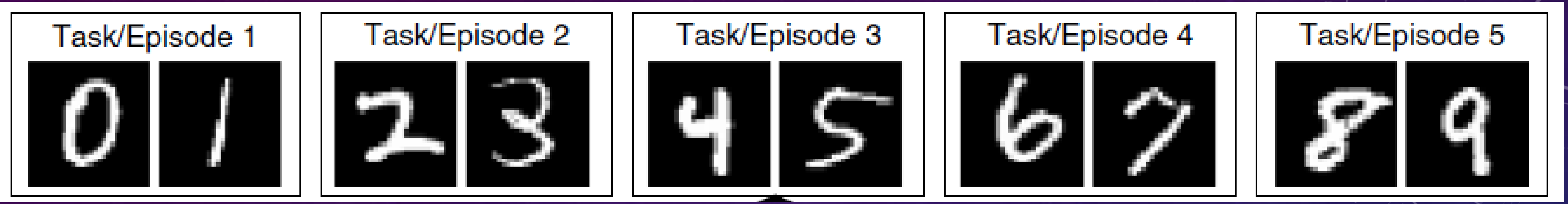
3. Add new classifier output nodes
4. Train Classifier with old (synthesized) and new classes of data
5. Expand latent vector dimension
6. Train Class-conditional Variational Autoencoder (VAE) with Gaussian Mixture Model (GMM)
   - Initialize VAE with classifier weights (use random weights for classifier layers that don't overlap with VAE)

NEW FEATURE LEARNING AUTOENCODER (NFLAE)

- Initialize with Classifier weights (new weights are initialized with random values.
- Train network to minimize difference between Input data and Output data (Reconstruction error).



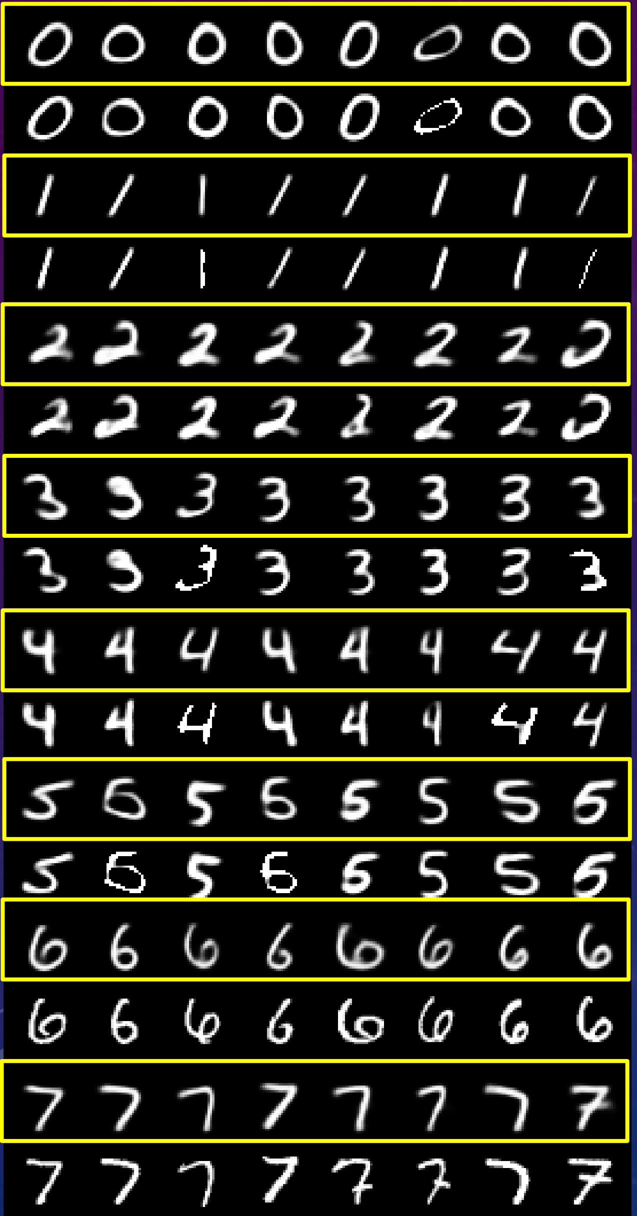
EXPERIMENT – MNIST CLASS-INCREMENTAL LEARNING

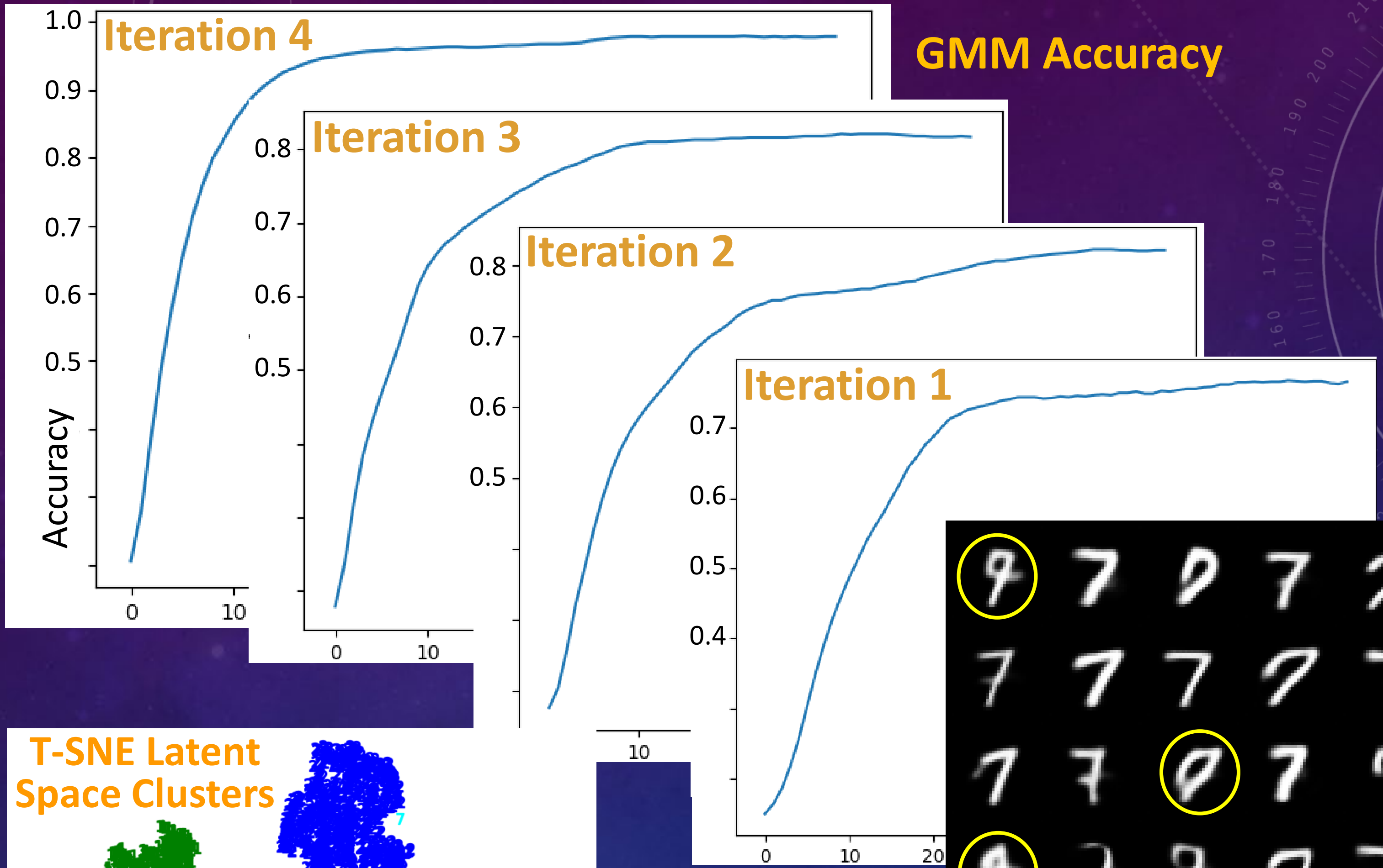


1. Train a classifier to discriminate 0’s and 1’s.
2. Present next Task/Episode (e.g., 2’s and 3’s) to the CLN algorithm.
   - Add 3 filters/nodes per layer per Task.
   - Add 6 latent dimensions per Task.
3. Compute accuracy of classifier against the MNIST test set for all the digits seen thus far (e.g., 0’s, 1’s, 2’s, and 3’s).
4. Repeat 2-3 until all Tasks/Episodes are complete.

VAE-GMM VISUALIZATIONS

Image Reconstructions





GMM Accuracy

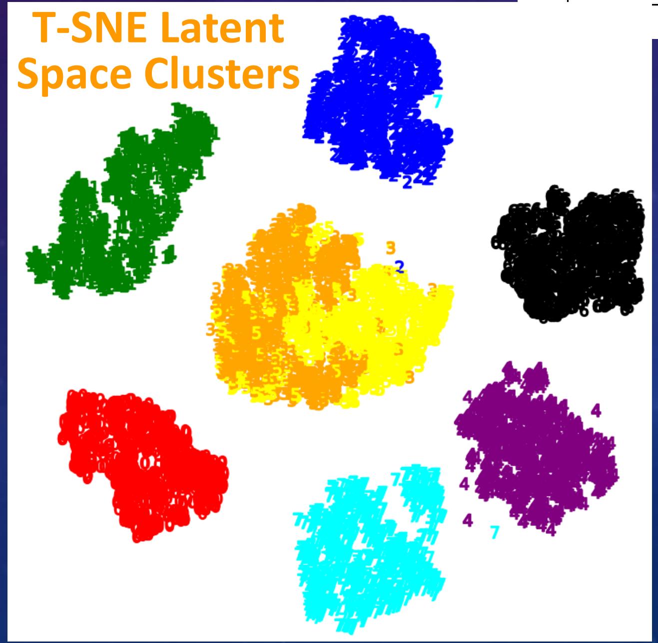
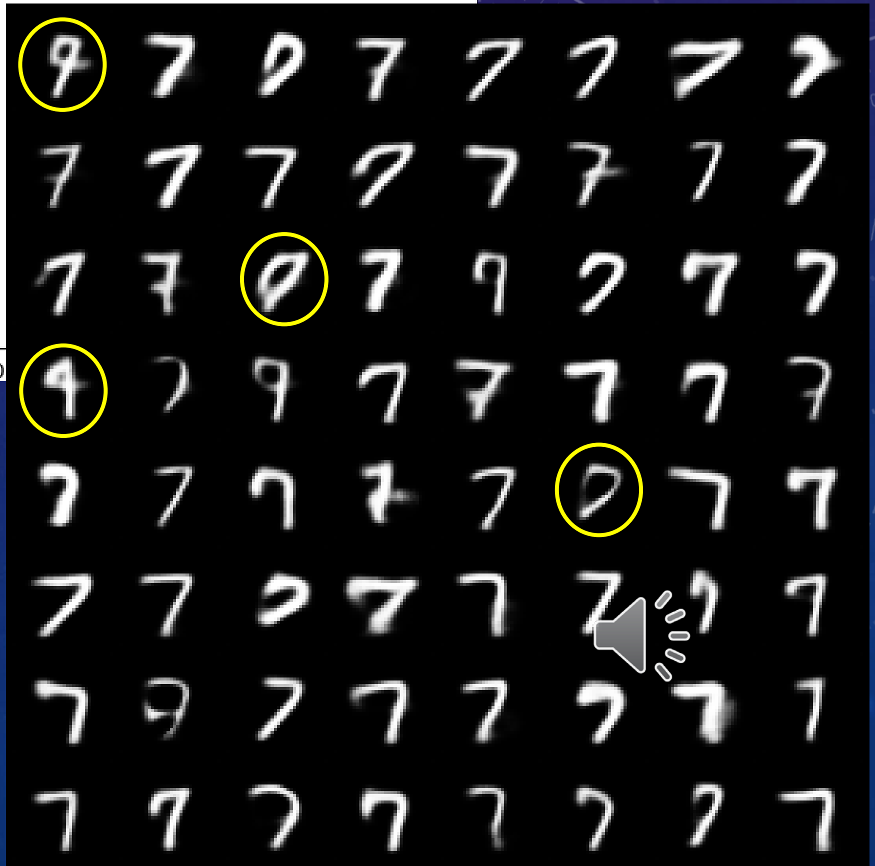


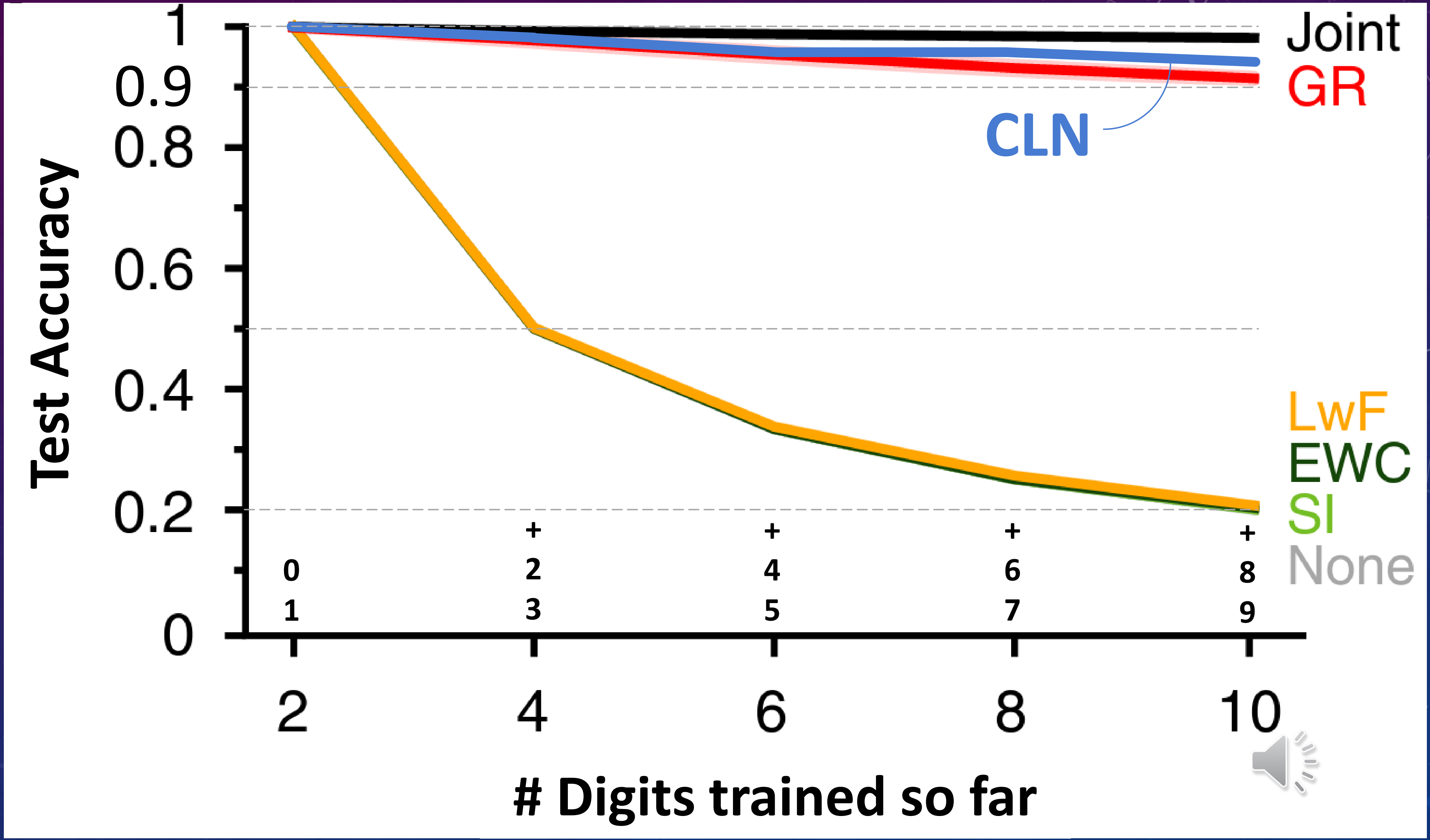
Image Synthesis of 7's



RESULTS OF CLN ON MNIST CLASS-INCREMENTAL LEARNING

CLN

Digits Trained	Test Accuracy
0,1	0.998
+2,3	0.981
+4,5	0.959
+6,7	0.961
+8,9	0.938



Modified figure from van de Ven, G.M., Siegelmann, H.T. & Tolias, A.S. Brain-inspired replay for continual learning with artificial neural networks. *Nature Communications* **11**, 4069 (2020). https://doi.org/10.1038/s41467-020-17866-2