

**Algorithm 1:** Combining Architecture Search with the Synthetic Petri Dish

---

**Input:** Set of motifs already evaluated  $X = \emptyset$ . Initial set of motifs for ground-truth evaluation  $X_{eval}$ . Performance of motifs  $P = \emptyset$ . Number of motifs to generate  $M$ . Number of motifs to select  $K$  s.t.  $K \ll M$ . Number of NAS search iterations  $L$ . Additional Petri dish hyperparameters (Appendix A.3 and A.4).

**for**  $l = 1$  **to**  $L$  **do**

**Ground-truth Evaluation:** Train each motif in  $X_{eval}$  in the full-scale setting and obtain full-scale validation set performance  $P_{eval}$ . Enlarge  $P : P = P \cup P_{eval}$ . Enlarge  $X : X = X \cup X_{eval}$ .

**Petri dish Training:** Train the Petri dish model using  $X$  and  $P$  (see sub-section 3.1).

**Architecture Search:** Generate a set of  $M$  new motifs ( $X_{new}$ ) using  $X$  and  $P$  as inputs to the NAS method.

**Performance Prediction:** Run Synthetic Petri Dish inference for  $X_{new}$  (sub-section 3.2) to obtain motif performance predictions.

**Selection:** Pick top  $K$  motifs (denoted by set  $X_{top}$ ) out of the set  $X_{new}$  based on their predicted performance. Set  $X_{eval} = X_{top}$ .

**end for**

**Output:** The motif within  $X$  with the best ground-truth performance.

---

## A APPENDIX

## A.1 COMBINING SYNTHETIC PETRI DISH WITH NAS

The steps to combine NAS with Synthetic Petri Dish are outlined in Algorithm 1.

## A.2 PETRI DISH HYPERPARAMETER SELECTION PROCEDURE

Recall that an initial sampling of candidate motifs evaluated in the ground-truth setting serve as training data for Petri dish (Step 0 in Figure 2b). In order to determine the ideal hyperparameters for Petri dish training, this initial motif ground-truth set (referred to as  $\mathcal{L}_{true}^i$  in Section 3) is split into equal parts as training ( $\mathcal{L}_{true\_train}^i$ ) and validation ( $\mathcal{L}_{true\_valid}^i$ ) sets. Hyperparameter setting that results in the smallest outer-loop MSE loss ( $\mathcal{L}_{outer}^i$  in equation 2) on the validation set ( $\mathcal{L}_{true\_valid}^i$ ) after training the Petri dish on the training set ( $\mathcal{L}_{true\_train}^i$ ), are selected for the full experiment. Specific hyperparameter values for each experiment are provided in the following sections.

## A.3 EXPERIMENTAL SETUP FOR OPTIMAL SIGMOID SLOPE SEARCH

This section provides further details on experiment 4.1. The sigmoid slope value is the motif in this experiment.

## A.3.1 SETUP FOR GROUND-TRUTH TRAINING

Previous work has shown that the slope of the sigmoid activation is a critical factor in determining network performance (Ramachandran et al., 2018). Similarly, in this paper, it was empirically found that the validation performance of a 2-layer 100-wide feedforward network on MNIST dataset is dependent on the sigmoid slope. This dependence is shown by the blue-points in Figure 1. To generate each of the blue points, the feedforward network was trained on 50K MNIST training samples and evaluated on 10K validation samples, with the sigmoid slope value ranging between 0.01 – 2.01. All other hyperparameters were held constant during each run and their specific values are provided in Table 1. For each slope value, a mean performance from 20 such runs (along with standard error bars) is plotted in Figure 1.

## A.3.2 SETUP FOR PETRI DISH TRAINING AND VALIDATION

The hyperparameter selection procedure follows the same template as described in Section A.2. A subset of 30 ground-truth points are randomly selected from a restricted interval of sigmoid slope values ranging between 0.37 – 1.50 (blue-points in the blue-shaded region of Figure 1). These 30 ground-truth points are split into two equal parts to create training (15) and validation data-set (15)

Hyperparameter	Search Range	Final Setting
Optimizer	N/A	Adam
Inner-loop learning rate	0.001 – 0.01	0.01
L2 weight penalty	1e-5 – 1e-3	1e-5
Number of Epochs	40 – 60	50
Batch Size	20, 40, 50, 100	50

Table 1: Hyperparameter setting for training a 2-layer, 100-wide feedforward network to obtain ground-truth performance in Experiment 4.1.

for Petri dish hyperparameter selection. Hyperparameter search range and the final selected values are listed in Table 2.

Hyperparameter	Search Range	Final Setting
Inner-loop Optimizer	N/A	Adam
Motif-network input size	10	10
Motif-network output size	10	10
Motif-network hidden size	1, 3, 5	1
Synthetic training samples	10, 20	10
Inner-loop learning rate	0.001 – 0.01	0.01
Inner-loop L2 penalty	1e-5 – 1e-3	1e-5
Inner-loop training steps	200, 250	250
Outer-loop Optimizer	N/A	Adam
Outer-loop learning rate	0.01 – 0.05	0.05
Outer-loop learning rate decay	0.4 – 0.8	0.4
Outer-loop L2 penalty	1e-5 – 1e-3	1e-5
Outer-loop training steps	20, 40, 60	60

Table 2: Hyperparameter setting for Petri dish training and inference in Experiment 4.1. Because the number of synthetic training samples is small (20), they are used in a single batch for Petri dish inner-loop training. The number of synthetic training samples is the same as the number of synthetic validation samples.

At the beginning of Petri dish training, both the motif-networks weights and the synthetic training/validation data are initialized to random values (drawn from normal distribution). The relative performance of motif-networks after inner-loop training on such random synthetic data is shown in Figure 4. This plot highlights that motif-network training extracts useful prior information about the corresponding motifs.

**Implementation Details:** As described in Section 3, during Petri dish training, the motif-networks are independently trained and evaluated. Such independent training and evaluation is usually achieved by distributing network training on GPUs. However, because the motif-networks are very small (with only 22 parameters in each) and they share the same synthetic data, their training can be parallelized by a simple trick. The individual motif-networks can be combined to create a single super-network. With appropriate masking connections that prevent any forward and backward propagation across motif-networks within a super-network, we can ensure that all the motif-networks are independently trained at the same time during the super-network training. This parallelization trick allows us to carry out Petri dish training and inference on a basic MacBook CPU very quickly.

### A.3.3 SETUP FOR NN-BASED MODEL TRAINING

The training and validation data for the NN-based model is exactly the same as the one used for Petri dish i.e. a set of 30 ground-truth points, where each point is a tuple of sigmoid slope and the validation accuracy of the corresponding ground-truth MNIST network. Similarly to the Petri dish, the NN-based model is trained to predict the *normalized* ground-truth performance (see **Outer-loop training** in Section 3). However, unlike Petri dish that trains and evaluates the motif (i.e. sigmoid slope) in a synthetic setting, the NN-based model takes the real-valued slope directly as its input. A

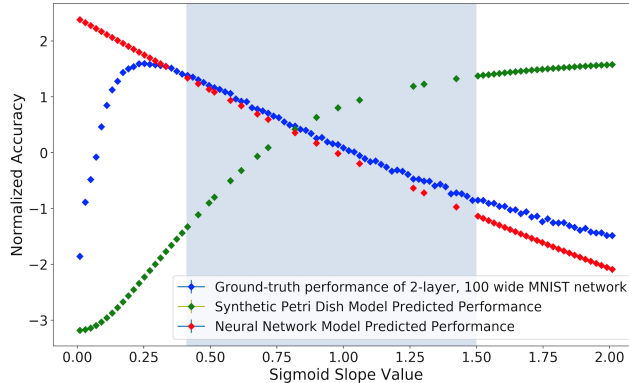


Figure 4: **Relative performance of motif-networks with random synthetic data.** The green curve shows the performance of motif-networks after inner-loop training on random data. This plot is similar to Figure 1 except that there is no Petri dish outer-loop training in this case.

mean squared error is computed between the NN predicted output and the *normalized* ground-truth performance. Hyperparameter search range and their final selected values are listed in Table 3.

Hyperparameter	Search Range	Final Setting
Optimizer	N/A	Adam
Network hidden size	5, 10, 15, 20	10
Initial Learning rate	0.001 – 0.01	0.01
Learning rate decay	0.8 – 0.99	0.97
Learning rate decay steps	50 – 120	100
L2 weight penalty	1e-5 – 1e-3	1e-4
Number of training steps	100, 150, 200	150
Batch Size	15	15

Table 3: Hyperparameter setting for training the NN-based performance prediction model in Experiment 4.1.

#### A.4 EXPERIMENT SETUP FOR RECURRENT CELL ARCHITECTURE SEARCH

This section provides further details on experiment 4.2. The recurrent cell design is the motif in this experiment.

##### A.4.1 SETUP FOR GROUND-TRUTH EVALUATION

Since NAO (Luo et al., 2018) is used as a baseline method for our experiments, the ground-truth training and evaluation procedure outlined in that paper is also followed here. The final test evaluation of the best found cell (output of Algorithm 1) is also carried out using the setting described in the NAO paper.

##### A.4.2 SETUP FOR PETRI DISH TRAINING AND VALIDATION

The hyperparameter selection procedure follows the same template as described in Section A.2. A set of 40 randomly selected cells are evaluated for their ground-truth performance. These 40 ground-truth points are split into two equal parts to create training (20) and validation data-set (20) for Petri dish hyperparameter selection. Hyperparameter search range and their final selected values are listed in Table 4.

**Implementation Details:** The parallelization trick of combining multiple motif-networks into a single super-network as described in Appendix A.3.2, is also utilized here as well. Experiment 4.1 required only a single NAS iteration i.e. training Petri dish with a fixed number of training motifs (and their corresponding ground-truth values) and then predicting the performance of test motifs.

Hyperparameter	Search Range	Final Set-ting
Inner-loop Optimizer	N/A	Adam
Motif-network input size	10	10
Motif-network output size	10	10
Motif-network hidden size	1, 3, 5	3
Synthetic training size	10, 20	20
Inner-loop learning rate	0.001 – 0.01	0.01
Inner-loop L2 penalty	1e-5 – 1e-3	1e-5
Inner-loop training steps	50, 100	50
Outer-loop Optimizer	N/A	Adam
Outer-loop learning rate	0.01–2.5	2.0
Outer-loop learning rate decay	0.4 – 0.8	0.5
Outer-loop L2 penalty	1e-6 – 1e-3	5e-5
Outer-loop training steps	100, 200, 300	200

Table 4: Hyperparameter setting for Petri dish training and inference in Experiment 4.2.

Unlike Experiment 4.1, in Experiment 4.2, the number of training motifs increase with additional ground-truth evaluations in each NAS iteration. For example, at the end of three such NAS iterations, there are 100 training motifs that can be utilized for Petri dish training. A growing number of motif-networks during Petri dish training could require further hyperparameter tuning. To avoid such fine-tuning, the total number of motif-network instances during Petri dish training is kept fixed at 40 (thereby also limiting the size of the super-network). During each outer-loop training step, a new batch of 40 motif-networks are randomly sampled from the full-set of available motif-networks.

#### A.4.3 SETUP FOR NAO AND RANDOM SEARCH

In Experiment 4.2, two variants of NAO are evaluated in limited resource setting – NAO with Reduced data (red-curve in Figure 3) and Synthetic Petri Dish-NAO (green curve in Figure 3). For both the variants, the NAO model is trained with the same code and hyperparameter setting as in the published work (Luo et al., 2018).

Two random search (RS) variants are evaluated – Synthetic Petri Dish with RS (blue curve in Figure 3) and RS with partial evaluation (black curve in Figure 3). The key difference between the two variants is that while the former trains small motif-networks with synthetic data to quickly estimate motif performance, the latter trains full-sized networks with randomly sub-sampled real data. For a fair comparison, the number of training steps and the size of the data in each variant is kept the same. In both the variants, random mutations are applied to generate new motifs. In each NAS iteration, the top 20 motifs (with the best ground-truth values) are mutated to generate 100 new motifs. Each motif is represented as a fixed-size string of numbers that encode the cell connectivity and the types of non-linearity within the cell. During the mutation of a motif, every location within its string is modified with a probability of 0.05.

The best found cell using the Synthetic Petri Dish-NAO method has a test perplexity of 57.1 and is shown in Figure 5.

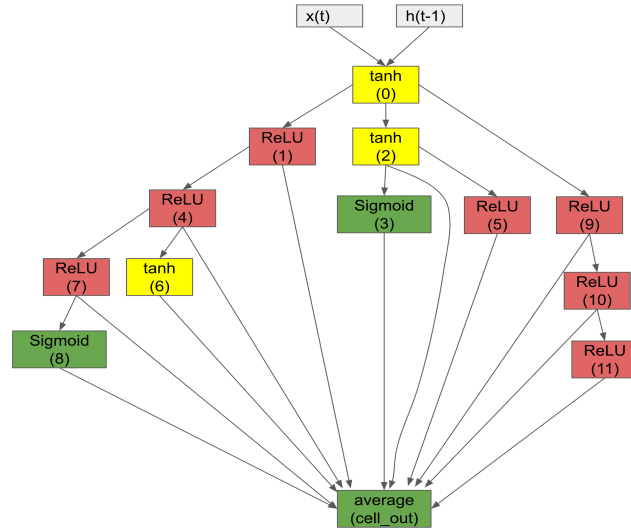


Figure 5: **Best Cell found by Synthetic Petri Dish-NAO method.**