








Self-organising Neural Network Hierarchy

Satya Borgohain¹(✉) , Gideon Kowadlo²(✉) , David Rawlinson²(✉) ,
Christoph Bergmeir¹(✉) , Kok Loo¹(✉), Harivallabha Rangarajan²(✉),
and Levin Kuhlmann¹(✉) 

¹ Monash University, Wellington Road, Clayton, VIC 3800, Australia
{satya.borgohain, christoph.bergmeir, levin.kuhlmann}@monash.edu,
kploo1@student.monash.edu

² Cerenaut, Richmond, VIC 3121, Australia
{gideon,dave,hari}@cerenaut.ai
<https://www.monash.edu/>
<https://cerenaut.ai/>

Abstract. Mammalian brains exhibit functional self-organisation between different neocortical regions to form virtual hierarchies from a physical 2D sheet. We propose a biologically-inspired self-organizing neural network architecture emulating the same. The network is composed of autoencoder units and driven by a meta-learning rule based on maximizing the Shannon entropy of latent representations of the input, which optimizes the receptive field placement of each unit within a feature map. Unlike Neural Architecture Search, here both the network parameters and the architecture are learned simultaneously. In a case study on image datasets, we observe that the meta-learning rule causes a functional hierarchy to form, and leads to learning progressively better topological configurations and higher classification performance overall, starting from randomly initialized architectures. In particular, our approach yields competitive performance in terms of classification accuracy compared to optimal handcrafted architecture(s) with desirable topological features for this network type, on both MNIST and CIFAR-10 datasets, even though it is not as significant for the latter.

Keywords: Biologically plausible networks · Self-supervised learning · Greedy training

1 Introduction

A critical component behind the performance of Artificial Neural Networks (ANN) remains the manual design of their architectures, which is fixed prior to the training of networks and requires specialized domain knowledge involving iterative search, empirical discovery, intuition or trial and error [3].

Automated architecture search methods like Neural Architecture Search (NAS) with Reinforcement Learning [22], and Evolutionary Algorithm [20] based approaches provide viable alternatives to explicit tailoring of neural network

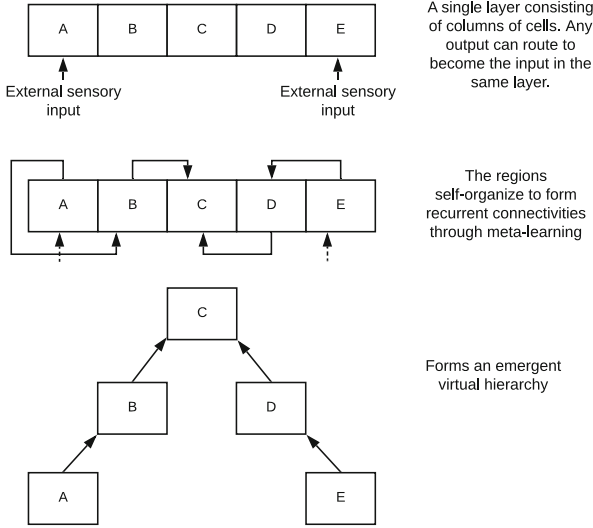


Fig. 1. A simplified high-level representation of the self-organising hierarchy.

architectures for a given task. NAS methods [3] outperform manual architectures in many areas such as semantic segmentation [1], image classification and object detection [23]. They employ complex strategies to search the space of possible architectures and evaluate performance which makes them extremely computationally expensive [23] and hard to reproduce. We propose a biologically motivated alternative to automated architecture search that attempts to coarsely mimic how the brain might adapt its own architecture at an abstract level.

We loosely base our idea on the Gradiental model, proposed by [5], which claims that functional neocortical organisation is continuous, interactive and emergent, and introduced a cognitive gradient that referred to gradual changes in encoded representations and integration between sensory modalities across the surface. Furthermore, since the mammalian neocortex is physically a thin 2D sheet there is evidence of a functionally self-organising virtual hierarchy (as opposed to a physical one) between the different neocortical regions [7] (See Fig. 1). In particular, we study the effects on one sensory modality i.e. visual modality, and learn useful representations in a partially unsupervised manner.

The primary focus of our approach is *discovering* the optimal spatial placement of the receptive fields (ϕ) over the inputs and its latent representations via a meta-learning rule. We consider a meta-network as a Directed Acyclic Graph (DAG) in which every node is an autoencoder (AE) unit, with the same number of hidden units (n_{hidden}), and locally learn useful representations of the inputs received from the other parts of the network. Two types of learning simultaneously take place in this setting i.e. primary (or base) learning (f_{θ}) which is learning of feed-forward weights θ and secondary (or meta) learning (f_{ϕ}) which

is learning the receptive field location (and thus, the topology) of each AE unit for optimal feature encoding in a lower dimensional space.

To measure the quality of learned representations and drive meta-learning, we use Shannon entropy as an intrinsic metric. Furthermore, to assess overall quality of the network and its architecture we use these unsupervised representations to train a logistic regression classifier and use its accuracy as an extrinsic metric.

A key distinction between our work and NAS [22] approaches is that the self-organizing network retains the learned weights from previous meta-iterations which is meant to reflect a smooth adaptation to a given task during training.

The major contributions of this paper are summarized as follows:

1. We propose a self-organising neural network architecture which meta-learns its architecture during training to produce effective representations of the inputs for downstream tasks like classification.
2. We formulate a meta-learning rule based on entropy of latent representations and empirically show that it leads to better topological configurations.
3. We show functional hierarchy emergence via the self-organizing network.

2 Proposed Framework

The overall framework is shown in Fig. 2. It can be broadly divided into unsupervised, which includes both the base (local) learning in the AE units along with meta-learning of the receptive fields, and the supervised phase which consists of training a multinomial logistic regression classifier.

Firstly, we conceive a simple high-level abstraction for the 2D cortical sheet and refer to it as the feature map of encodings with dimensions as $h \times w$, where we store the inputs along with the subsequent hidden layer activity of each AE unit during training. Allocation of both the input and the latent activity within the feature map is arbitrary and specified prior to training which remains fixed throughout. This is meant to represent the gradual encoded representations across the cortical map.

The meta-network is a combination of the feature map and the AE units. Training in the context of this meta-network is to learn K meta-parameters $\phi = \{r_k \mid k \in \{1, \dots, K\}\}$, where r_k is the receptive field for the k^{th} AE unit. Hence, its topological configuration is fully defined by the number of AE units (K) and location of their receptive fields (r_k). The set of receptive fields for a topology is given as

$$\{r_k = \{(x_i^{(k)}, x_j^{(k)}), (y_a^{(k)}, y_b^{(k)})\} \mid k \in \{1, \dots, K\}\} \quad (1)$$

where, $x^{(k)}$ represents the hidden layer row activity with beginning and ending indices as i and j , and $y^{(k)}$ represents the column activity where a, b are the beginning and ending indices respectively of the k^{th} AE unit within the feature map. The notion of a receptive field here refers to a subset of input activities projecting from one layer to the next in the virtual hierarchy as opposed to the traditional definition that refers to a subset of external inputs to the network.

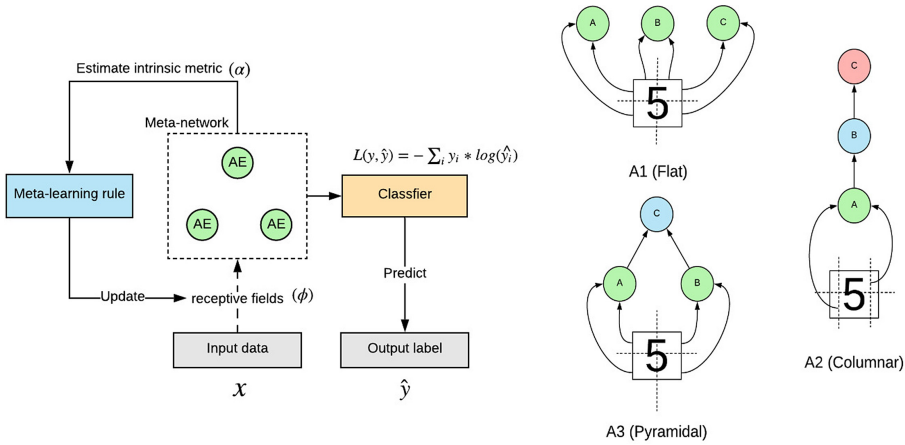


Fig. 2. (left) A high level schematic of the framework. (right) Handcrafted architectures with 3 AE units namely A1, A2 and A3. Different colors represent levels within the hierarchy. The arrows represent the input to the AE units and their receptive fields. Here the dotted lines divide the raw image into different regions.

We handcraft several architectures, as shown in Fig. 2 above, in order to establish correlation between entropy and accuracy. These hierarchical systems vary in their order, reversibility and pyramidal structure [2], and as a consequence represent various levels of structural optimality.

Once the AE units of a meta-network sufficiently converge, we use it to train a two layer multinomial logistic regression classifier. Thereafter, we concatenate the hidden layer activity from all AE units to form a $(K \times n_{hidden})$ dimensional vector as input to the classifier as shown in Fig. 3.

2.1 Base Learning

We use a single layer, under-complete AE with standard backpropagation learning where we treat $h \times w$, i.e. the complete feature map size, as the input dimensions and apply a binary indicator matrix δ_k of the same dimension per AE unit. δ_k then has a sub-matrix of ones which is specified by r_k . We use mean squared error (MSE) for the reconstruction loss along with dropout and L2 norm for regularization. As captured in Fig. 2, each unit targets 2 fixed-size receptive fields (r_k) of $N \times N$ dimensions making the effective receptive field size to be $2 \times N \times N$ per unit.

During this phase, each AE unit minimizes its reconstruction loss locally and learns a low dimensional representation of its inputs. Subsequently after each forward pass, the hidden layer activity of each unit is updated on the feature map (in its designated location).

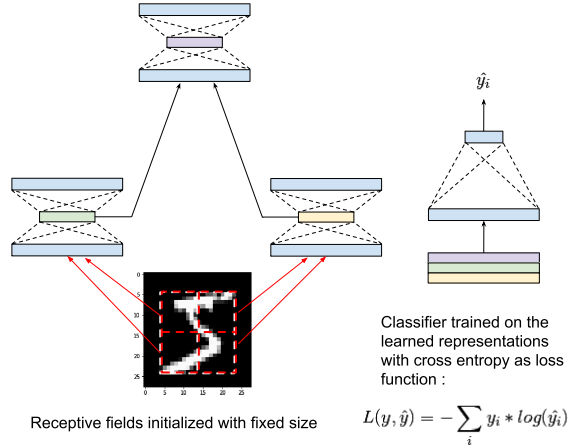


Fig. 3. Left: A simple handcrafted meta-network (A3) with 3 AE units arranged in a pyramidal fashion. Each unit receives different inputs (a part of the image) based on the position of its receptive field and its latent encoding becomes a potential input for the next unit. Right: A logistic regression model trained on the hidden unit activity.

2.2 Meta-Learning

The meta-learning algorithm is shown in Algorithm 1. This is used for both the handcrafted, (where we specify r_k) as well as for the meta-learned architectures.

Intrinsic Metric. Tishby and Zaslavsky [21] used information theoretic measures to highlight the trade-off between effective compression of information and prediction from it. We use Shannon entropy as the intrinsic metric through which we can indirectly infer about the information content in the learned representation and subsequently the fitness of the meta-learned-network.

Formally,

$$H(x) = - \sum_i P_i \log(P_i) \tag{2}$$

where, P_i is the probability of i^{th} hidden activity.

We initialize the feature map with the input image and zeros representing non-activation. To ensure a stable pool of activations in the feature map before backpropagation in AE units, $K \times K$ forward passes are performed without accumulating gradients and the feature map is updated each time. Once it is saturated, we perform K forward passes for each unit followed by K backward passes. Finally, we estimate the entropy per unit per epoch.

To estimate the entropy of the latent activity in AE units, we use a histogram (binning) based approach. For given continuous activation values C , we discretize the same into discrete activations D , by computing a histogram over the same which allows for a rough estimation of the probability distribution of C [17]. The authors noted that the estimated entropy is sensitive to the choice of binning

Algorithm 1. Steps for Meta-network training.

Input: Receptive fields, r_k ; Height, h ; Width, w ;**Output:** Shannon entropy, $\{H(x_k) \mid k \text{ in } \{1 \dots K\}\}$; Trained AE units, $\{A_k \mid k \in \{1 \dots K\}\}$;

```

1: Initialize:  $K$  AE units;
2: while not max epoch do
3:   Initialize Feature map  $F = F_{h \times w}^{(0)}$ ;
4:   for  $k$  in  $1 \sim K$  do
5:     for  $k$  in  $1 \sim K$  do
6:       Perform forward pass for  $A_k$ ;
7:       Update  $F := F_{h \times w}^{(k)}$  with hidden activity  $x_k$ ;
8:     end for
9:   end for
10:  for  $k$  in  $1 \sim K$  do
11:    Perform forward pass for  $A_k$ ;
12:    Perform backward pass for  $A_k$ ;
13:    Update  $F := F_{h \times w}^{(k)}$  with hidden activity  $x_k$ ;
14:    Estimate Shannon entropy  $H(x_k)$  using equation 2;
15:  end for
16: end while
17: return ;

```

as it yields different discrete representations for C . However, [13] showed that the estimated values fall within the theoretical limits for small and large bins. We employ a similar strategy as [16], where AE units are allowed to be fully trained to capture the maximum activation value and to ensure that the resulting histogram represents the full range of activations (as ReLUs do not have an upper bound). We selected a constant bin size of 100, independent of the topology.

For a given topology, let x_k be the hidden layer activity and $H(x_k)_t$ be the local entropy of the k^{th} unit at time t . The intrinsic metric α_t at time t is,

$$\alpha_t = H(x_k)_t \quad (3)$$

We greedily optimize for local entropy values for each AE unit.

Meta-Learning Rule. We explore memoryless, meta-heuristic approaches, namely local search and stochastic hill climbing [15], along with random search to formulate the meta-learning rule with maximizing entropy as the acceptance criterion as per Algorithm 2.

We introduce a small random perturbation in the r_k during each iteration of meta-learning, constrained to a range between $[-1, 1]$ for a horizontal or vertical shift i.e. two degrees of freedom. It accepts a randomly selected neighbour as a candidate solution, only if it leads to a higher local entropy. Although this does not guarantee a global optima in the case of non-convex optimization problems, it provides a reasonably optimal solution within a time constrained setting [15].

Algorithm 2. Stochastic hill climber as Meta-learning rule**Input:** Meta-steps, M ;**Output:** Receptive fields, r_{best} ;

```

1: Initialize  $r_{best} \leftarrow \text{randomTopology}()$ ;
2: Let  $\alpha_0 \leftarrow 0$ ;
3: for  $m$  in  $1 \sim M$  do
4:   for  $i$  in  $1 \sim K$  do
5:      $r_{candidate} \leftarrow r_{best}$ ;
6:      $r_{candidate}^{(i)} \leftarrow \text{randomNeighbor}(r_{best}^{(i)})$ ;
7:     Train Meta-network with  $r_{candidate}$  and get  $H(x_i)$ ;
8:      $\alpha_t \leftarrow H(x_i)$ ;
9:     if  $\alpha_t > \alpha_{t-1}$  then
10:       $r_{best} \leftarrow r_{candidate}$ ;
11:      break;
12:     else
13:       continue;
14:     end if
15:   end for
16: end for
17: return  $r_{best}$ ;

```

The meta-learning rule updates slower than the base learning to ensure that units were fully trained and converge to a sufficient degree before estimating the entropy of the hidden activity. Figure 4 shows a forward pass during meta-learning.

In order to retain weights from previous iterations, we multiply δ_k with all current weights of an AE unit as $\theta_{ij} := \theta_{ij} \times \delta_{ij}$ where θ_{ij} is a single weight and δ_{ij} is an element of the binary indicator matrix. Hence, only the weights of the neurons which lie within the r_k are subject to be updated during training with the rest remaining virtually unchanged.¹

3 Experiments

3.1 Datasets

For our experiments, we focus on two datasets:

MNIST: We use the widely studied MNIST dataset² of single-channel, 28×28 dimensional images consisting of 10 classes with 60,000 instances as training set and 10,000 as the test set.

CIFAR-10: Contains multi-channel, natural images drawn from 10 classes with 50,000 and 10,000 images for training and testing respectively [8].

¹ The code is available on github under: <https://github.com/Cerenaut/self-organizing>.

² <http://yann.lecun.com/exdb/mnist/>.

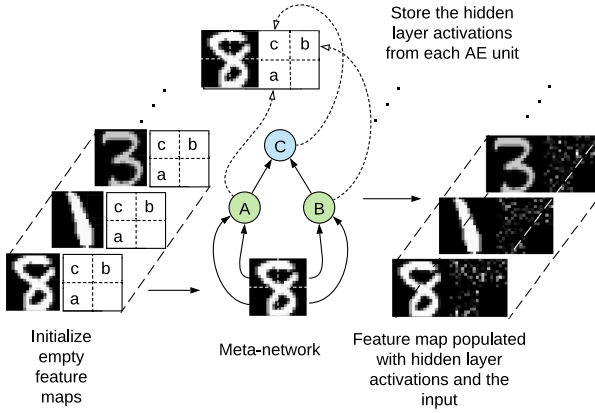


Fig. 4. A schematic depiction of a forward pass in the meta-network. Here A , B and C are the AE units and a , b and c are the location of their hidden layer activity in the feature map. Receptive fields of A and B i.e. r_A and r_B respectively, target the image whereas r_C targets the hidden layer activations of both A and B . The dotted arrows represent the updating of the hidden layer activations of the corresponding units in the feature map.

For MNIST, we first centre-crop the images to be multiples of 10 for making the manipulation of receptive fields within the feature map simpler. It is cropped to be $28 \times 28 \rightarrow 20 \times 20$ followed by min-max normalization to bring the pixels values of the images from $[0, 255]$ to be between $[0, 1]$. To ensure center cropping does not impact the classifier’s performance, we evaluate its performance both with and without cropping and found an increase of only $\sim 0.003\%$ in the test error rate. The 8×8 region acts mostly as extra padding. For CIFAR-10, we perform standard data augmentation i.e. zero-pad with 4 pixels on each side, random crop back to 32×32 and random horizontal flip [6, 9–11, 14, 18, 19]. Thereafter, we convert it to grayscale and normalize.

3.2 Experimental Strategy

For both datasets, here we seek to demonstrate three key features: (1) Hand-crafted hierarchical architectures perform better than random ones; (2) There is a near monotonic relationship between extrinsic classification accuracy and intrinsic entropy of a network that enables the use of maximising entropy by way of the meta-learning rule to maximise accuracy while at the same time leading to more hierarchical architectures; (3) That the meta-learning rule can be used to achieve improvements in accuracy relative to random or initial networks and at least comparable accuracy to handcrafted hierarchical networks.

Performance of Handcrafted and Random Architectures. We perform a series of 10 runs per architecture and compile the results in Table 1, where we report the mean accuracy of each topology.

Table 1. Accuracy (mean) of all 10 runs for the handcrafted and randomly initialized architectures for each dataset. The architectures with the highest accuracy is highlighted in gray.

Topology	Name	MNIST	CIFAR-10
A1	Flat	96.4	31.69
A2	Columnar	94.21	30.79
A3	Pyramidal	96.6	32.84
AR1	Random	93.03	31.97
AR2	Random	93.76	27.21
AR3	Random	90.28	29.86

From Table 1, we observe that the topology with *perfect* hierarchy [2], namely A3, performs the best in each category for both the datasets. Furthermore, columnar architectures (A2) performs poorly among the handcrafted ones since they highly deviate from a perfectly hierarchical topology, receive partial information overall and have very poor integration of receptive fields as each unit only targets its receptive field at the unit directly below it.

Flat architecture (A1) yields a mean accuracy below A3 for both datasets. Random architectures yield lower accuracy scores overall for both datasets and hierarchical architectures which deviate from the properties of a perfect hierarchy in general, seem to exhibit lower performance relative to the degree of deviation.

Correlation Analysis. We observe *maximum* global entropy of a meta-network to be positively correlated with its accuracy, with a Pearson’s correlation coefficient of 0.75 and 0.48 for MNIST and CIFAR-10 respectively. This indicates that the greater the local entropy of the AE units, more is the amount of useful information captured in the encoded representation (as measured by the accuracy). Hence, seeking out regions of high activity on the feature map by the AE which in turn leads to high entropy ultimately improves performance. Figure 5 shows a regression line fit to the data highlighting the relationship between maximum entropy and accuracy for both datasets.

Performance of Meta-learned Architectures. Using local entropy estimates for α , we perform a series of 10 runs and record how the topology evolves throughout training. We report the performances of both the randomly initialized architecture and the final meta-learned architecture for the same topology in Table 2.

We perform Wilcoxon one-sided signed-rank test with a significance level of 0.05, between the final (\tilde{x}_1) and initial (\tilde{x}_0) accuracy of the meta-learned architectures and report the p-values obtained. The null hypothesis (H_0) is that the median difference ($\tilde{x}_1 - \tilde{x}_0$) is negative against the alternative (H_A) that it is positive. We observe p-values < 0.05 for MNIST and hence conclude that our

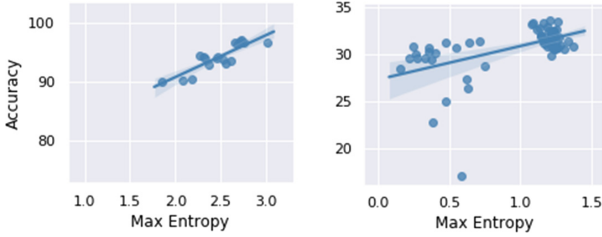


Fig. 5. Regression lines fitted between accuracy and the maximum entropy for MNIST (left) and CIFAR-10 (right) respectively.

Table 2. Median accuracies for both the randomly initialized, \tilde{x}_0 , and meta-learned, \tilde{x}_1 , architectures over 10 runs.

Dataset	\tilde{x}_0	\tilde{x}_1	max	p-value
MNIST	91.24	92.40	96.22	0.009
CIFAR-10	33.12	32.46	35.02	0.869

results are statistically significant. For CIFAR-10, we do not observe similar evidence against the null which we plan investigate in our future work. We however note that the meta-learning achieved a competitive performance of 35.02% on CIFAR-10 over the runs, better than any of the handcrafted ones.

Figure 6 shows accuracies from the randomly initialized architectures after a few meta-learning iterations (meta-steps) as indicated through the intermediate architectures during meta-learning. For MNIST, we observe highest improvement in performance among the meta-learned architectures with random initial configurations which started with very poor performance. Also, configurations that already yield high accuracies have more or less similar performance through the meta-learning. It is also apparent that meta-learning is sensitive to initializa-

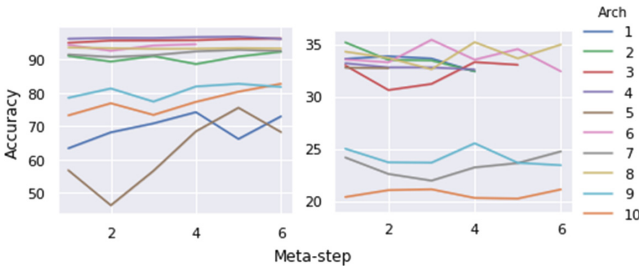


Fig. 6. Plot showing the change in accuracy per architecture with each meta-step for MNIST (left) and CIFAR-10 (right), over 10 different runs. Different randomly initialized architectures are represented by different colors. Meta-step indicates the number of receptive field updates before termination of the meta-learning loop.

tion of the receptive fields as is generally the case with local search algorithms. For CIFAR-10, we observe a lesser performance improvement. However, here the difference between the best and worst performing handcrafted architectures is much smaller and therefore we hypothesise that the search space is more challenging.

4 Conclusions and Future Work

In this paper we propose a self-organising network architecture which optimizes the spatial placement of receptive fields through a meta-learning rule based on entropy of the encoded representations. Our experiments demonstrate that meta-learned architectures are able to self-organise into a hierarchy by maximising entropy in the network. This appeared especially effective for MNIST, while for CIFAR-10 the results are still inconclusive. The more effective result on MNIST suggests the associated error surface is less complex and allows the local meta-learning to find a more optimal solution.

For MNIST, we observe the baseline to be 96.6% for handcrafted architectures sans augmentation or hyper-parameter tuning. For CIFAR-10, we achieve a baseline accuracy of 32.84%. When compared with existing results from other non-convolutional ANN models applied to CIFAR-10 as reported by [12], e.g. logistic regression on whitened input achieved 41% and much larger, vanilla FFNN achieved 51% (with 7,940,000 trainable parameters vs 48,000 in our configuration). As [12] noted, convolutional networks are not entirely biologically plausible (due to weight sharing), there is a need to explore more feasible alternatives. One contrasting feature of our approach with NAS methods is potentially reducing computational complexity since we meta-learn architectures without evaluating the classification performance for each candidate architecture.

To that end, we present a brain-inspired learning paradigm without using topological priors, such as convolutions for image processing, and only use basic neural network components driven by a meta-learning rule. Future work will include additional AE units (for more possible topologies), exploring alternatives for meta-learning such as Free-Energy minimization scheme [4] and also expanding across different sensory modalities and integrating the receptive fields to achieve better generalization across tasks and domains.

References

1. Chen, L.C., et al.: Searching for efficient multi-scale architectures for dense image prediction. In: *Advances in Neural Information Processing Systems*, pp. 8699–8710 (2018)
2. Corominas-Murtra, B., Goñi, J., Solé, R.V., Rodríguez-Caso, C.: On the origins of hierarchy in complex networks. *Proc. Natl. Acad. Sci.* **110**(33), 13316–13321 (2013)
3. Elsken, T., Metzen, J.H., Hutter, F.: Neural architecture search: a survey. arXiv preprint [arXiv:1808.05377](https://arxiv.org/abs/1808.05377) (2018)

4. Friston, K.: A free energy principle for biological systems. *Entropy* **14**(11), 2100–2121 (2012)
5. Goldberg, E.: Gradiental approach to neocortical functional organization. *J. Clin. Exp. Neuropsychol.* **11**(4), 489–517 (1989)
6. Huang, G., Sun, Yu., Liu, Z., Sedra, D., Weinberger, K.Q.: Deep networks with stochastic depth. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016*. LNCS, vol. 9908, pp. 646–661. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_39
7. Kandel, E.R., et al.: *Principles of Neural Science*, vol. 4. McGraw-Hill, New York (2000)
8. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
9. Larsson, G., Maire, M., Shakhnarovich, G.: FractalNet: ultra-deep neural networks without residuals. arXiv preprint [arXiv:1605.07648](https://arxiv.org/abs/1605.07648) (2016)
10. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.: Deeply-supervised nets. In: *Artificial Intelligence and Statistics*, pp. 562–570 (2015)
11. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint [arXiv:1312.4400](https://arxiv.org/abs/1312.4400) (2013)
12. Lin, Z., Memisevic, R., Konda, K.: How far can we go without convolution: improving fully-connected networks. arXiv preprint [arXiv:1511.02580](https://arxiv.org/abs/1511.02580) (2015)
13. Purwani, S., Nahar, J., Twining, C.: Analyzing bin-width effect on the computed entropy. In: *AIP Conference Proceedings*, vol. 1868, p. 040008. AIP Publishing LLC (2017)
14. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: FitNets: hints for thin deep nets. arXiv preprint [arXiv:1412.6550](https://arxiv.org/abs/1412.6550) (2014)
15. Russell, S.J., Norvig, P.: *Artificial Intelligence: A Modern Approach*. Pearson Education Limited, Kuala Lumpur (2016)
16. Saxe, A.M., et al.: On the information bottleneck theory of deep learning. *J. Stat. Mech. Theory Exp.* **2019**(12), 124020 (2019)
17. Shwartz-Ziv, R., Tishby, N.: Opening the black box of deep neural networks via information. arXiv preprint [arXiv:1703.00810](https://arxiv.org/abs/1703.00810) (2017)
18. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: the all convolutional net. arXiv preprint [arXiv:1412.6806](https://arxiv.org/abs/1412.6806) (2014)
19. Srivastava, R.K., Greff, K., Schmidhuber, J.: Highway networks. arXiv preprint [arXiv:1505.00387](https://arxiv.org/abs/1505.00387) (2015)
20. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. *Evol. Comput.* **10**(2), 99–127 (2002)
21. Tishby, N., Zaslavsky, N.: Deep learning and the information bottleneck principle. In: *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5. IEEE (2015)
22. Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. arXiv preprint [arXiv:1611.01578](https://arxiv.org/abs/1611.01578) (2016)
23. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: *Proceedings of the IEEE Conference on Computer Vision And Pattern Recognition*, pp. 8697–8710 (2018)