Chen and Murata *Appl Netw Sci* (2020) 5:94
https://doi.org/10.1007/s41109-020-00332-9

Applied Network Science

RESEARCH Open Access

Enhancing network modularity to mitigate catastrophic forgetting

Lu Chen[1*] and Masayuki Murata[2]

*Correspondence:
l-chen@kit.ac.jp
[1] Department of Information and Human Science, Kyoto Institute of Technology, Kyoto, Japan
Full list of author information is available at the end of the article

Abstract

Catastrophic forgetting occurs when learning algorithms change connections used to encode previously acquired skills to learn a new skill. Recently, a modular approach for neural networks was deemed necessary as learning problems grow in scale and complexity since it intuitively should reduce learning interference by separating functionality into physically distinct network modules. However, an algorithmic approach is difficult in practice since it involves expert design and trial and error. Kashtan et al. finds that evolution under an environment that changes in a modular fashion leads to the spontaneous evolution of a modular network structure. In this paper, we aim to solve the reverse problem of modularly varying goal (MVG) to obtain a highly modular structure that can mitigate catastrophic forgetting so that it can also apply to realistic data. First, we confirm that a configuration with a highly modular structure exists by applying an MVG against a realistic dataset and confirm that this neural network can mitigate catastrophic forgetting. Next, we solve the reverse problem, that is, we propose a method that can obtain a highly modular structure able to mitigate catastrophic forgetting. Since the MVG-obtained neural network can relatively maintain the intra-module elements while leaving the inter-module elements relatively variable, we propose a method to restrict the inter-module weight elements so that they can be relatively variable against the intra-module ones. From the results, the obtained neural network has a highly modular structure and can learn an unlearned goal faster than without this method.

Keywords: Modularity, Catastrophic forgetting, Neural network

Introduction

Learning a variety of different skills for different problems is a long-standing goal in artificial intelligence. However, when neural networks learn a new skill, they typically lose previously acquired skills. This problem, called catastrophic forgetting, occurs because learning algorithms change connections used to encode previously acquired skills to learn a new skill (Ellefsen et al. 2015; Goodfellow et al. 2014; Kemker et al. 2018).

Catastrophic forgetting has been studied for a few decades (French 1999; Lee et al. 2017). Recently, a modular approach for neural networks has been deemed necessary as learning problems grow in scale and complexity (Amer and Maul 2019). This modular approach, which involves constructing a network with densely connected modules with only sparser connections between the modules (Newman 2006), intuitively should

reduce learning interference by separating functionality into physically distinct network modules (Ellefsen et al. 2015). However, although several algorithmic approaches for constructing a modular topology are available, these approaches are difficult to implement in practice since they involve expert design and trial and error (Amer and Maul 2019). Therefore, an automatic approach is needed.

Studies in computational biology have suggested several mechanisms of the spontaneous evolution of a modular network structure. Ellefsen et al. (2015) and Clune et al. (2013) note that evolving networks with pressure to minimize connection costs leads to modular solutions. However, in their approach, the input data need to be partitioned in advance so that different modules can be assigned. Since manual data modularization is usually based on some heuristic, expert knowledge or analytical solution, a good partitioning requires a good prior understanding of the problem and its constraints, which is rarely the case for neural network learning tasks (Amer and Maul 2019).

Another study (Kashtan and Alon 2005; Parter et al. 2008; Kashtan et al. 2007, 2009) finds that evolution under an environment that changes in a modular fashion leads to the spontaneous evolution of the modular network structure. That is, the authors observe that repeatedly switching between several goals, each comprising a different combination of subgoals, which they call modularly varying goal (MVG), such as (X XOR Y) AND (Z XOR W) and (X XOR Y) OR (X XOR W), can lead to the spontaneous evolution of a modular network structure. In this case, two modules representing subgoals (X XOR Y) and (Z XOR W) can be obtained. They find that modular networks that evolve under these varying goals can not only find perfect solutions to those already learned goals in only a few steps but also exhibit high adaptability to novel goals (Parter et al. 2008).

However, in Kashtan and Alon (2005), the problems are hand-made and obviously vary with the same subgoals, lacking the complexity that most realistic data have. They not only show that switching goals comprising different combinations of subgoals can lead to spontaneous evolution of the modular network structure but also point out that randomly changing environments do not seem to be sufficient to produce a modular structure. However, realistic data are neither varying with the exact same subgoals nor varying at random. For example, image data with similar features, e.g., edges, intersecting lines, and curves, are popular (Li et al. 2015).

In this paper, we aim to solve the reverse problem of MVG to obtain a highly modular structure that can mitigate catastrophic forgetting so that it can also be applied to realistic data.

Neural networks are known to have many configurations of sets of weights and biases that result in the same performance (Kirkpatrick et al. 2017). First, we confirm that a configuration with a highly modular structure exists and that this neural network can mitigate catastrophic forgetting in learning a realistic dataset. To confirm that a highly modular structure is obtainable, we apply MVG against a realistic dataset. To confirm the mitigation of catastrophic forgetting, we show the learning accuracy against a previously unlearned goal, which is expected to be better than in the case of a neural network that has catastrophically forgotten a previously learned goal. From the results, we show that a configuration with a highly modular structure is obtainable and that this neural network is more adaptable to unlearned goals than that with a lower modular structure.

As a typical example of learning a realistic dataset, we use a classification problem based on CIFAR-10. A convolutional neural network (CNN) is used as the classifier for the classification problem as the first step since it is widely used for image datasets.

By applying MVG, we find that a highly modular structure exists and can mitigate catastrophic forgetting. Next, we solve the reverse problem, that is, we propose a method that can obtain a highly modular structure able to mitigate catastrophic forgetting. Since an MVG-obtained neural network can relatively maintain the intra-module elements while leaving the inter-module elements relatively variable, we propose a method to restrict the inter-module weight elements so that they can be relatively variable against intra-module ones. From the result, we show that the obtained neural network has a highly modular structure and can learn an unlearned goal faster than without this method.

Note that catastrophic forgetting was originally measured by presenting old input patterns to the network and evaluating how similar they are to its originally learned patterns (French 1999). Hetherington and Seidenberg introduced a savings measure of forgetting that measures the amount of time required to relearn the original data. They showed that some of the cases where the originally learned data was completely forgotten can be retrained to recall them very quickly (French 1999). In our paper, we show the learning accuracy against a previously unlearned goal but we expect it to share some features with old goals. This can be considered as an extended version of the latter. Because MVG learns by switching goals, it is less meaningful to compare the amount of time required to relearn the old goal.

Notably, Kirkpatrick et al. (2017) recently proposed a practical solution to overcome catastrophic forgetting to train a neural network by protecting the weights important for previous goals. However, since exact recognition (French 1999) is required, the solution would have difficulties in addressing the scalability and complexity. In fact, a parameter exists that sets how important the previous goal is compared with the new one.

This paper is organized as follows: "Applying MVG to obtain a modular structure" section shows a highly modular structure exists by applying an MVG against a realistic dataset. "Enhancing modularity" section presents the proposed method and its evaluation. "Conclusions and future works" section offers our conclusions and future works.

Applying MVG to obtain a modular structure

Neural networks are known to have many configurations of sets of weights and biases that result in the same performance (Kirkpatrick et al. 2017). First, we confirm that a configuration with a highly modular structure exists and that this neural network can mitigate catastrophic forgetting in learning a realistic dataset. To confirm that a highly modular structure is obtainable, we apply an MVG against a realistic dataset since it is thought to spontaneously evolve a modular network structure. To confirm the mitigation of catastrophic forgetting, we show the learning accuracy against a previously unlearned goal, which is expected to better than that in the case where the neural network catastrophically forgets a previously learned goal. As a typical example of learning a realistic dataset, we use a classification problem based on CIFAR-10. A CNN is used as the classifier for the classification problem since it is widely used

for image datasets. In "Applying MVG" section, we explain how we apply MVG to learn a realistic dataset. Then, "Evaluation" section shows the evaluation results.

Applying MVG

Original MVG MVG involve repeatedly switching between several goals, each comprising a different combination of subgoals (Kashtan and Alon 2005). Although no detailed information exists on how to generate these goals, a few examples are given in Kashtan and Alon (2005). The most basic example is an electronic combinatorial logic circuits problem, which is considered to explore the connection pattern of gates towards a given logic function as the output. Switching between a logic function goal G_a and another logic function goal G_b is shown to spontaneously lead to a modular structure:

$$G_a = (X \text{ XOR } Y) \text{ AND } (Z \text{ XOR } W), \tag{1}$$

$$G_b = (X \text{ XOR } Y) \text{ OR } (Z \text{ XOR } W), \tag{2}$$

where X, Y, Z, and W are inputs and G_a and G_b share common subproblems (X XOR Y) and (Z XOR W). Kashtan et al. showed that networks that evolve under varying goals seem to discover the basic subproblems common to the different goals and therefore can rapidly adapt to each of the different goals. Besides, it can exhibit high adaptability comparing to a neural network learned a fixed goal (FG) in learning novel goals comprising previously seen subgoals but in a new combination (Parter et al. 2008), such as:

$$G_c = (X \text{ XOR } Y) \text{ NOR } (Z \text{ XOR } W). \tag{3}$$

for the example against (1) and (2).

Application to a realistic dataset In this research, we apply an MVG against a realistic dataset to confirm that a configuration with a highly modular structure exists in learning a realistic dataset. As a typical example of learning a realistic dataset, we use a classification problem based on CIFAR-10. Goals are set to learn whether images belong to a given class following the original image classification problem. Since multiple goals are needed to apply an MVG, the problem is rearranged to classify 2 classes instead of the original 10 classes to classify whether images belong to a given class or not. For the simplest example, the goals can be denoted as follows:

$$G_1 \ : \ \text{Airplane}, \tag{4}$$

$$G_2 \ : \ \text{Truck}. \tag{5}$$

G_1 represents a goal to classify whether images are Airplane or not, and G_2 represents a goal to classify whether images are Truck or not. Following the original MVG, the goals are designed to share a common set of outputs (2 output neurons). That is, the input data are the same for every goal, and different goals have different labels for every input. Therefore, changing goals here means changing the labels. Class weight is set to balance the relabelled learning data. Following MVG, we expect MVG learned neural network can learn novel goals faster than FG. Since image datasets are known to share edges,

intersecting lines, and curves (Li et al. 2015), we here consider the classification of an unlearned category of the images as a novel goal. Details are explained later.

Algorithm 1 Algorithm for MVG

```
INPUT: GOALS, reducePressure, maxEpoch
Initialize CNN
t ← 0
while ( t < ⌊maxEpoch/|GOALS|⌋ ) do
    for g in GOALS do
        Execute backpropagation on CNN with loss function L_g
        for Filter f in CNN do
            for Element e in f do
                e ← e * reducePressure
            end for
        end for
        t ← t + 1
    end for
end while
```

The MVG algorithm is show in Algorithm 1. After initializing the CNN, along with switching the goals, cost reduction is introduced to multiply *reducePressure* by every filter element of the CNN, as cost reduction seems necessary to mitigate catastrophic forgetting. The filter elements are connection weights. Backpropagation is used for training, and categorical cross-entropy is used as the loss function. Note that the original MVG (Kashtan and Alon 2005) also contains a sort of cost reduction that results in a penalty when the number of network components exceeds a given threshold, which could suggest the necessity of cost reduction. This operation can also be considered as a weight decay. Moreover, only cost reduction without switching goals enhances the modular structure, as suggested in Ellefsen et al. (2015), but not as much as an MVG does. The result of only cost reduction is shown later.

Modularity of a CNN

CNN used for evaluation A CNN is used for evaluation since it is a common option for learning image datasets. Since a CNN trained on CIFAR-10[1] already exists, the structure of our CNN originates from it. The layer structure of the neural network is

$$input - 32C3 - 32C3 - MP2 - 64C3 - 64C3 - MP2 - 512FC - 2softmax$$

ReLU is used as the activation function for the convolutional layers and the dense layer, except the output layer. Padding is valid except in the first and third convolutional layers. Dropout with 0.25 is applied after each max-pooling layer, and that with 0.5 is applied after the first dense layer. Backpropagation is used for training, and SGD without momentum and decay is used as the optimization algorithm. Categorical cross-entropy is used as the loss function. The batch size is 32. Note that pre-processing and data augmentation in the original document are left unchanged since those are carried out for practical use.

Modularity measurement Since a CNN is too large to compute network modularity, we extract a weighted undirected graph from a CNN for the calculation. Let us represent the weighted undirected graph as $G = (V, E)$ consisting of $n = |V|$ vertices and $m = |E|$

[1] https://github.com/fchollet/keras.

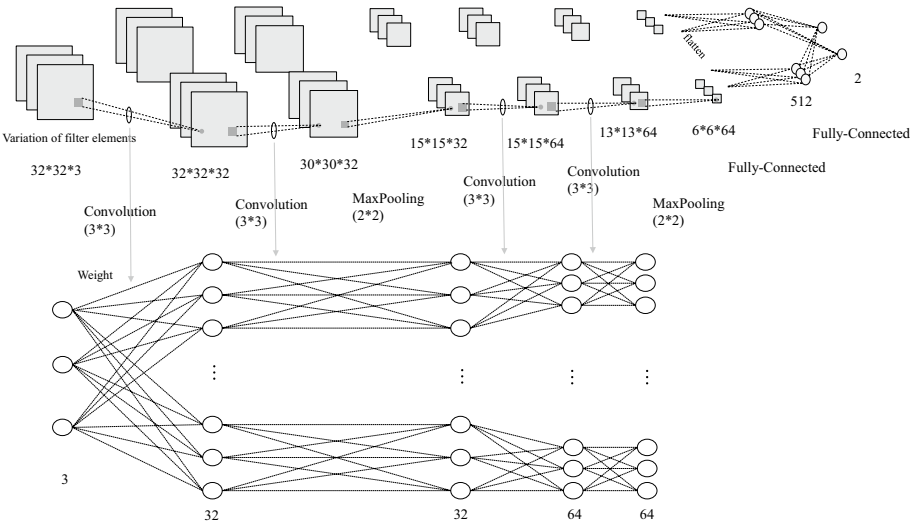


Fig. 1 Graph used to calculate the modularity

edges. The adjacency matrix of G is denoted by $\mathbf{A} = A_{ij}$, where A_{ij} is the weight of edge (i, j) and $A_{ij} = 0$ if $(i, j) \notin E$.

We regard V and E as channels and filters in the CNN, respectively (Fig. 1). For simplicity, pooling layers, the fully connected layer and the softmax layer are ignored. As a result, the resulting graph consists of 3/32/32/64/64 nodes, with full connections between each layer with no direction. A_{ij} is calculated based on the variance in the filter v_{ij} involved with channels i and j (Eq. 6):

$$A_{ij} = v_{ij} * (n_i + n_j), \tag{6}$$

where n_i is the number of nodes in the layer that node i belongs to. $(n_i + n_j)$ is multiplied by v_{ij} to restore the generalization of filter elements. The initialization of filter elements is performed by the Glorot uniform initializer, which generalizes the filter elements by the number of former nodes and latter nodes it is connected to (Eq. (7), Ref. Glorot and Bengio 2010), affecting the result of module detection and leading to rarely finding modules across layers. Therefore, generalization is restored through Eq. (6):

$$W \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}}, \frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}}\right]. \tag{7}$$

The modular structure is a division of the vertices in V into a collection of disjoint subsets of vertices $\mathcal{C} = C_1, C_2, \ldots, C_l$ that the union gives back to V. The Louvain method (Blondel et al. 2008) is used for module detection to maximize the modularity $Q(\mathcal{C})$:

$$Q(\mathcal{C}) = \frac{1}{2m} \sum_{ij \in V} \left[A_{ij} - \frac{d_i d_j}{2m}\right] \sigma_{ij}, \tag{8}$$

where d_i and d_j are the weighted degrees of nodes i and j, respectively; m is the total sum of all edge weights; and the element σ_{ij} of the membership matrix is defined as

$$\sigma_{ij} = \begin{cases} 1, & \text{if } i \text{ and } j \text{ are in the same module} \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

The modularity values can be either positive or negative, and higher modularity values indicate a stronger module structure. For example, the average modularity of 10 initially randomized networks is 0.1424 ± 0.0018. This positive value could be caused by the physical structure of the neural networks.

Evaluation

Evaluation settings

Comparison method For the comparison method, we use a fixed goal (FG), which represents the original learning algorithm (Appendix 1) and used to learn a single goal for *maxEpoch*.

Dataset The dataset used for evaluation includes 50,000 32×32 colour training images from CIFAR-10, which were originally labelled over 10 categories (Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship, Truck) and relabelled to only 2 categories representing whether the input image is the target goal or not. The input data are the same for every goal, i.e., 50,000 32×32 colour training images, and the labels are different for each goal.

Parameters reducePressure for MVG is set to 0.99 after several trials.

Results

Learning accuracy To show that a highly modular structure is obtainable, we first show the result of switching Airplane and Truck as an example. Other combinations of switching goals are argued later. First, to ensure that learning is progressing, we show the learning accuracy. Figure 2a presents the result. The X-axis displays the epoch, while the Y-axis presents the accuracy. The dataset is presented per each epoch. Accuracy refers to the categorical accuracy, and for clarity, we show the training accuracy. The blue line indicates the result of the MVG learned by switching the goals Airplane and Truck every epoch. The accuracy shown here is the accuracy against the goal in learning, which means that the accuracy against Airplane is shown after learning Airplane and likewise for Truck. The learning rate is set to 0.1. To clarify the accuracy of each goal, the accuracy of Airplane of the MVG is shown by the black line connecting the epochs involved in the learning of Airplane. For comparison, the result of the FG, i.e., learning a single goal, that is, Airplane, without any ingenuity is shown by the orange line. Since gradient explosion occurs when setting the learning rate to 0.1, which will lead to a decrease in the accuracy, the result of setting the learning rate to 0.07 is shown. From the results, we can see that the accuracy of the MVG is lower than that of the FG. However, since the MVG can stably reach a certain degree of accuracy for the two goals at the end of the examination, the decrements are inevitable. In the original paper, using an MVG was found to facilitate adaptation to perfect solutions for each of the two goals. The difference between their results and ours could originate from the difficulty of the goals. As explained in "Introduction" section, their goals were hand-made and lacked complexity, while our goals comprise realistic data. This difference could be the reason for the variation in the accuracy achieved.

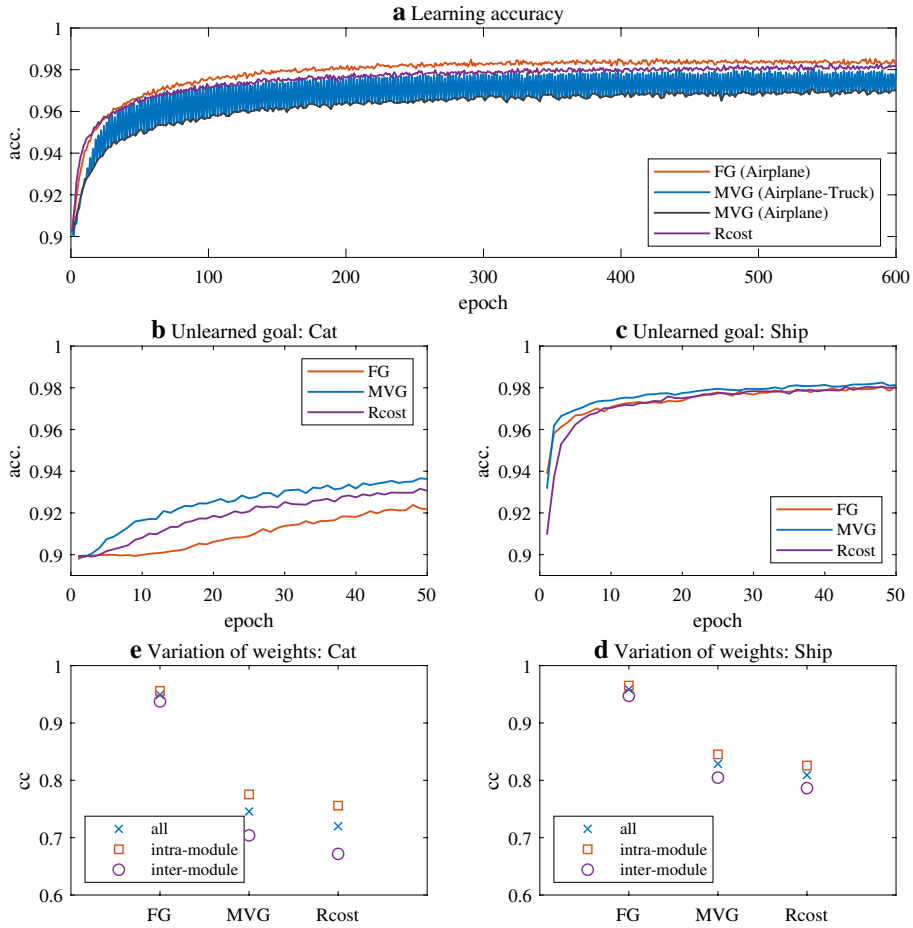


Fig. 2 Learning towards fixed and switching goals. **a** Categorical accuracy against the goal of learning. Blue indicates the MVG of Airplane and Truck. Blue line is a single line with high frequency. Y-axis is the accuracy of the learning goal. Black connects the epochs needed to learn Airplane in the MVG. Red indicates the FG of Airplane. Purple indicates Rcost. **b**, **c** Categorical accuracy against unlearned goals Cat and Ship. The weights of the initial neural network are from the last epoch of (**a**). **e**, **d** Pearson's correlation coefficient (cc) against before and after learning unlearned goals Cat and Ship. The cc of the filter elements of the initial neural network and the last neural network in (**b**, **c**) are shown. Blue indicates the cc of all the filter elements. Red indicates the cc of elements in the filter corresponding to intra-module links. Purple indicates the cc of elements in the filter corresponding to inter-module links

Modularity Second, we show the modularity of the obtained networks to confirm that a highly modular structure is obtainable by an MVG. The modularity of the neural network obtained in the last epoch of Fig. 2a is 0.2041 for the MVG and 0.1865 for the FG. The modularity of the FG is not low, but the modularity of the MVG is still higher. Since a trained neural network is known to have a modular structure (Filan et al. 2020; Watanabe et al. 2018), an FG obtaining such a modularity is not surprising. The result of the MVG being higher than that of the FG is expected, and the tendency is the same as that in the original paper.

Accuracy relative to unlearned goals Third, we show the learning accuracy against a previously unlearned goal to confirm the mitigation of catastrophic forgetting. Although unlearned goals in the original paper were defined as goals comprising

previously seen subgoals but in a new combination (Parter et al. 2008), as we pointed out in "Introduction" section, realistic data do not vary with the exact same subgoals. Since image datasets are known to share edges, intersecting lines, and curves (Li et al. 2015), we here consider the classification of an unlearned category of the images as a new goal. In detail, an unlearned category classification is used for evaluation. The CIFAR-10 categories contain animals and artificial objects. Since animals and artificial objects are known to have different characteristics, e.g., non-rigid objects such as animals are known to have relatively various poses (Ramesh et al. 2019), Ship and Cat are evaluated as new goals as representatives of both kinds of objects. Figure 2b, c present the results. The X-axis corresponds to the epoch, and the Y-axis presents the categorical accuracy. The neural networks obtained in the last epoch in Fig. 2a are used for comparison. The weight sets are kept the same, and only the goal is changed. The goal is kept the same for 50 epochs. For a fair comparison with the FG of learning Airplane, we use the neural network obtained in the epoch in which Airplane is initially learned in the MVG for comparison. Specifically, since Airplane is learned in the 1st epoch of Fig. 2a, the neural network obtained in the 601th epoch is used for evaluation. From the result, we first see that the MVG-obtained neural network learns faster than the FG-obtained ones for both new goals. The result that the MVG outperforms the FG is the same as that in the original paper and is expected. Second, we can see that the difference between the MVG and FG for Cat is relatively larger than that for Ship, which could be because of the similarity of the goals. Since Ship is closer to Airplane, which is already learned, even for the FG, Ship can be learned faster. On the other hand, since Cat is relatively different from Airplane, Cat is difficult to learn for the FG. Therefore, as a result, the advantage of the MVG becomes conspicuous when learning Cat. Note that although the learning rate of the FG is 0.07, which is less than that of the MVG, i.e., 0.1, since the lower the learning rate is, the less the neural network specializes to a single goal, the obtained neural networks can be regarded as too fair. For the evaluation against the unlearned goals, we also show the results after setting the learning rate to 0.1, which results in no large difference from the 0.07 setting (see Appendix 2). Although we assumed learning a single goal as a process that involves catastrophic forgetting of the previous goal, we also present an example of a neural network learning two goals sequentially in the FG manner (Appendix 3). Although the figure shows one trial, we also show results comparing several different conditions later. Also, evaluation for other goals are shown in Appendix 4.

Topological analysis From the results above, we find that a configuration with a highly modular structure exists that can mitigate catastrophic forgetting. To analyse the result from the structure aspect, the change in structure before and after learning the new goal is analysed. Since the advantage of the modular structure is its ability to reduce the learning interference between modules (Ellefsen et al. 2015), elements within modules (inter-module elements) and those without modules (intra-module elements) are evaluated separately. For simplicity, we evaluate the filter elements of a CNN. The degree of changes before and after learning the new goal is shown by Pearson's correlation coefficient (cc) (Damicelli et al. 2019). The cc is taken between the first and last topology of Fig. 2c, d, which represent before and after learning the new goal. The modular detection is performed against the first topology. The cc values of all the filter elements, the

cc values of filter elements corresponding to the intra-module links, and the cc values of filter elements corresponding to the inter-module links are shown. Fig. 2e, f show the results. From the results, we can see that the FG has a high cc for all elements compared to the MVG. Hence, the elements of the FG remain nearly the same even though the learning goal has changed, which means that the obtained topology is less variable. On the other hand, the MVG is successfully used to obtain a topology that is variable when meeting a new goal. Moreover, the MVG has a high cc when comparing intra-module elements to inter-module elements, which could be the reason of learning unlearned goals faster. Note that the reason why FG has high cc while learning an unlearned goal (Cat) more slowly could be because the output of FG relies on specific links in the neural networks, and the new goal is learned by changes to those links. This is consistent with FG having lower modularity. Additionally, it is consistent with the reported results of EWC (Kirkpatrick et al. 2017), which successfully overcame catastrophic forgetting by protecting the weights important to previous goals. This result suggests that importance is concentrated on specific links.

Comparing with link restriction only "Applying MVG" section mentions that the MVG contains the restriction of links. Thus, to show that the result is caused not only by the restriction of links, we compare with a method that restricts links only without switching goals. The results are shown as Rcost in Fig. 2. The modularity of the neural network obtained in the last epoch of Fig. 2a is 0.1995. From the results, although Rcost is more variable than the MVG in learning unlearned goals, the modularity and the speed of learning an unlearned goal are less than for the MVG. Therefore, we can conclude that the result of the MVG cannot be obtained by the restriction of links only. In addition, from the topological aspect, we can conclude that making the topology too variable does not lead to a better solution. Having relatively maintained intra-module elements while leaving the inter-module elements variable seems to be the key feature of learning unlearned goals faster.

Increment of goals

The previous section shows switching between 2 goals. Since we expect that switching between more goals can sharpen the modular structure, in this section, we show the results when the number of switching goals increases. In detail, the results of switching 2, 4, 6, and 8 goals are shown. The goals are selected randomly, as shown in Table 1. For a fair comparison with an unlearned goal, one fixed goal (Airplane) is considered. Cat and Ship, which are used to evaluate unlearned goals, are removed from the learning goals.

Learning accuracy First, Fig. 3a–d shows the learning accuracy for 2, 4, 6, and 8 goals. The goal with the asterisk in Table 1 is shown. From the results, as the number of learning goals increases, the learning accuracy decreases. However, a certain degree of accuracy can still stably be reached even in learning 8 goals. Taking account of the number of learning goals, the decrease in the accuracy reached is inevitable.

Accuracy relative to unlearned goals Second, the accuracy against unlearned goals is shown. For a fair comparison, a neural network trained only on Airplane after 600 epochs is used for evaluation. Figure 3e, f display the results. For Cat, every result of the MVG outperforms those of the FG. However, for Ship, some examinations reveal the contrary. Specifically, switching between Airplane and Deer, Dog and Horse is

Table 1 Goals for evaluation

Evaluation	Goals
1 Goal	Airplane
2 Goals	Airplane, dog
2 Goals	Airplane, horse
2 Goals	Airplane, deer
2 Goals*	Airplane, truck
4 Goals*	Airplane, bird, truck, horse
4 Goals	Airplane, truck, horse, dog
4 Goals	Airplane, dog, frog, horse
4 Goals	Airplane, deer, dog, horse
6 Goals*	Airplane, dog, automobile, deer, truck, horse
6 Goals	Airplane, frog, dog, deer, horse, bird
6 Goals	Airplane, horse, dog, deer, bird, automobile
6 Goals	Airplane, truck, horse, dog, frog, bird
8 Goals*	Airplane, bird, automobile, deer, truck, dog, frog, horse
8 Goals	Airplane, automobile, truck, bird, deer, dog, frog, horse
8 Goals	Airplane, deer, automobile, truck, frog, horse, bird, dog
8 Goals	Airplane, dog, horse, deer, truck, automobile, frog, bird

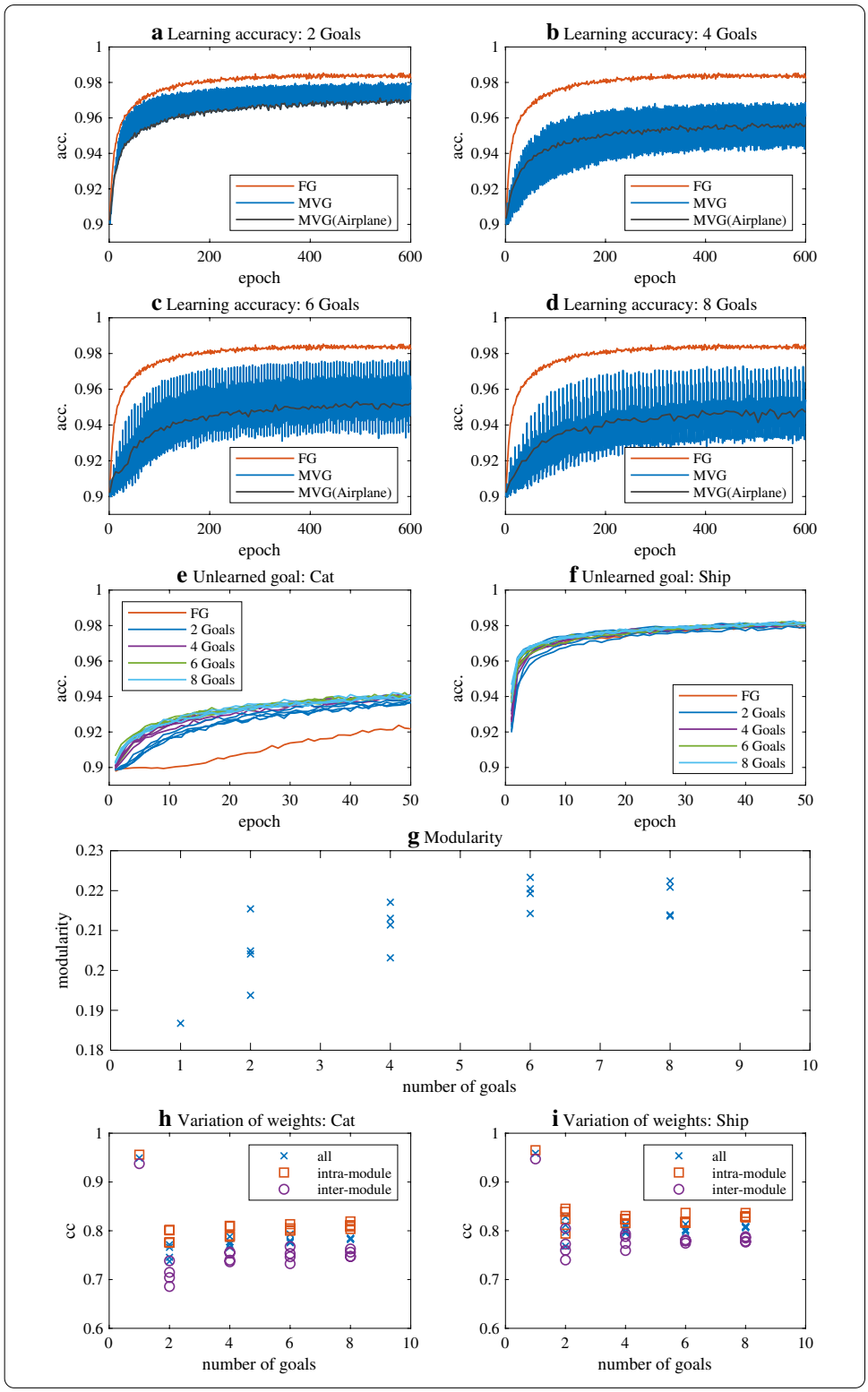
slower than that for the FG among the 2 goals, which may occur because, as explained in "Results" section, since Ship is closer to Airplane, which is already learned, even if the FG is used, Ship can still be learned faster. With the increase in goals, we can see that learning occurs faster as the switching goal increases in both evaluations. The difference between 2 goals and 8 goals becomes much clearer when learning Cat, because, as explained before, Cat is relatively different from Airplane, making the advantage of switching goals more conspicuous.

Modularity Third, we show the modularity. Figure 3g presents the result. The modularity is lower when learning a single goal, i.e., an FG. Modularity tends to increase as the number of goals increases. The results support our expectation that switching between more goals can sharpen the modular structure.

Topological analysis Fourth, we show the result of the cc. Figure 3h, i displays the results. After comparing the cc values of all elements between 2 goals and 8 goals, we see that the topology is less variable for the case of 8 goals. Additionally, especially in the case of Cat, the intra-module cc increases as the number of goals increases from 2 to 8, which could be explained by the sharpened modules. The sharpened modules lead to fewer changes in intra-module elements when meeting new goals. From the results,

(See figure on next page.)

Fig. 3 Learning towards multiple goals. **a**–**d** Categorical accuracy against the goal when learning different numbers of goals. Blue indicates the MVG. Black connects the epochs involved in learning Airplane in the MVG. Red indicates the FG of Airplane. The set of goals used for learning is shown in the text. **e**, **f** Categorical accuracy against unlearned goals Cat and Ship. The weights of the initial neural network are from the first time it learns Airplane after 600 epochs. **g** Modularity of the obtained neural network. The neural network is the initial neural network of **e**, **f**. **h**, **i** Pearson's correlation coefficient (cc) against before and after learning the unlearned goals Cat and Ship. The cc of the filter elements of the initial neural network and the last neural network in (**e**, **f**) is shown. Blue indicates the cc values of all the filter elements. Red indicates the cc values of elements in the filter corresponding to intra-module links. Purple indicates the cc values of elements in the filter corresponding to inter-module links



we can conclude that a configuration with an even more highly modular structure exists when switching between larger numbers of goals and that this configuration can mitigate catastrophic forgetting. Additionally, the best solution does not have the highest

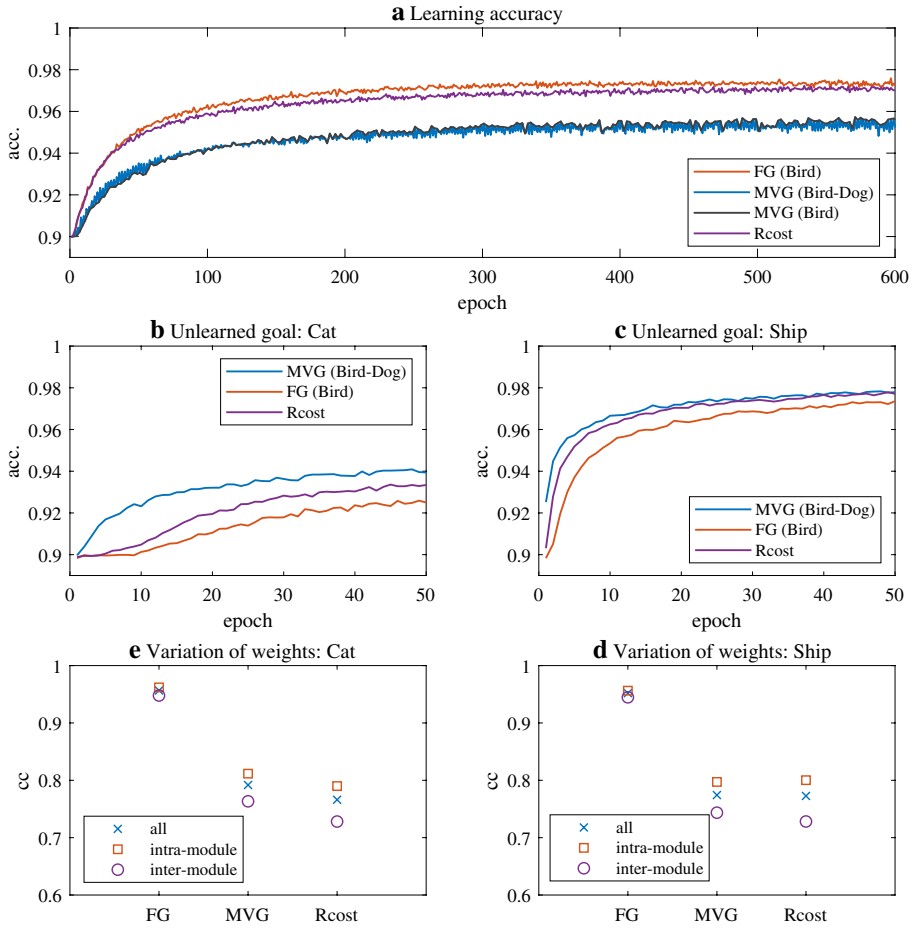


Fig. 4 Learning towards different combinations of goals. **a** Categorical accuracy against the goal in learning. Blue indicates the MVG of Bird and Dog. Black connects the epochs involved in learning Bird in the MVG. Red indicates the FG of Bird. Purple indicates Rcost. **b**, **c** Categorical accuracy against unlearned goals Cat and Ship. The weights of the initial neural network are from the last epoch of (**a**). **e**, **d** Pearson's correlation coefficient (cc) against before and after learning the unlearned goals Cat and Ship. The cc of the filter elements of the initial neural network and the last neural network in (**b**, **c**) is shown. Blue indicates the cc values of all the filter elements. Red indicates the cc values of elements in the filter corresponding to intra-module links. Purple indicates the cc values of elements in the filter corresponding to inter-module links

cc. Again, having relatively maintaining intra-module elements while leaving the inter-module elements variable seems to be the key feature for learning unlearned goals faster.

Other examples

In the previous sections, we showed the evaluation against unlearned goals only using a neural network trained on Airplane. Here, we show a new example evaluating an animal, Bird, against unlearned goals. The modularity of the neural network obtained in the last epoch of Fig. 4a is 0.2203 for the MVG and 0.1856 for the FG. From the result, we can conclude that the same tendency as in "Results" section is also observable when evaluating with a goal other than Airplane. The difference in learning accuracy between the MVG and FG is larger than in the Airplane-Truck case because complicated objects such as an animal are learned. Therefore, this decrease is inevitable. Evaluation against Ship shows obvious

difference for this evaluation, which could be because already learned goal, Bird, is different from it.

Evaluation with another dataset (CIFAR-100) is shown in Appendix 6.

Enhancing modularity

After applying the MVG, we find that a highly modular structure exists and that it can mitigate catastrophic forgetting. Next, we solve the reverse problem, that is, we propose a method that can obtain a highly modular structure able to mitigate catastrophic forgetting. Since the MVG-obtained neural network can relatively maintain the intra-module elements while leaving the inter-module elements relatively variable, we propose a method to restrict the inter-module weight elements so that they can be relatively variable against the intra-module ones. In "Proposed approach" section, we explain our proposed approach. Then, "Evaluation" section presents the evaluation results.

Proposed approach

The proposed method is to restrict the inter-module weight elements to make the inter-module elements relatively variable against the intra-module ones. Specifically, *reducePressure* is multiplied by the inter-module elements in every epoch after learning. Modular detection is performed every subsequent *renewEpoch*. If modular detection is carried out in every epoch, elements will be influenced by the fluctuation of modular detection.

Algorithm 2 Algorithm for interMOD

```
INPUT: reducePressure, γ, maxEpoch, renewEpoch
Initialize CNN
t ← 0
while ( t < maxEpoch ) do
    if ( t mod renewEpoch = 0 ) then
        Execute module detection on CNN with resolution γ to obtain inter-module links M_I
    end if
    Execute backpropagation on CNN with loss function L_g
    for Filter f in M_I do
        for Element e in f do
            e ← e * reducePressure
        end for
    end for
    t ← t + 1
end while
```

To adjust the number of restricted links, a parameter is introduced. Since, as learned from the MVG-obtained neural network, maintaining intra-module links while leaving inter-module elements relatively variable is important, to ensure enhancement of the modular structure, we introduce a parameter considering the modularity: the resolution of the modularity γ (Hagberg et al. 2008). The modular detection maximizes Eq. (10) instead of Eq. (8).

$$Q_\gamma(\mathcal{C}) = \frac{1}{2m}\sum_{ij\in V}\left[\gamma * A_{ij} - \frac{d_i d_j}{2m}\right]\sigma_{ij} \tag{10}$$

The resolution parameter will change the size of the modules. The original modularity is the same as that calculated by setting γ to 1.0, and the module decreases as γ decreases. When γ is 0, each of the nodes is in its own module. Therefore, the number of restricted links tends to be small when γ is 1.0 and increases as γ decreases, since the number of

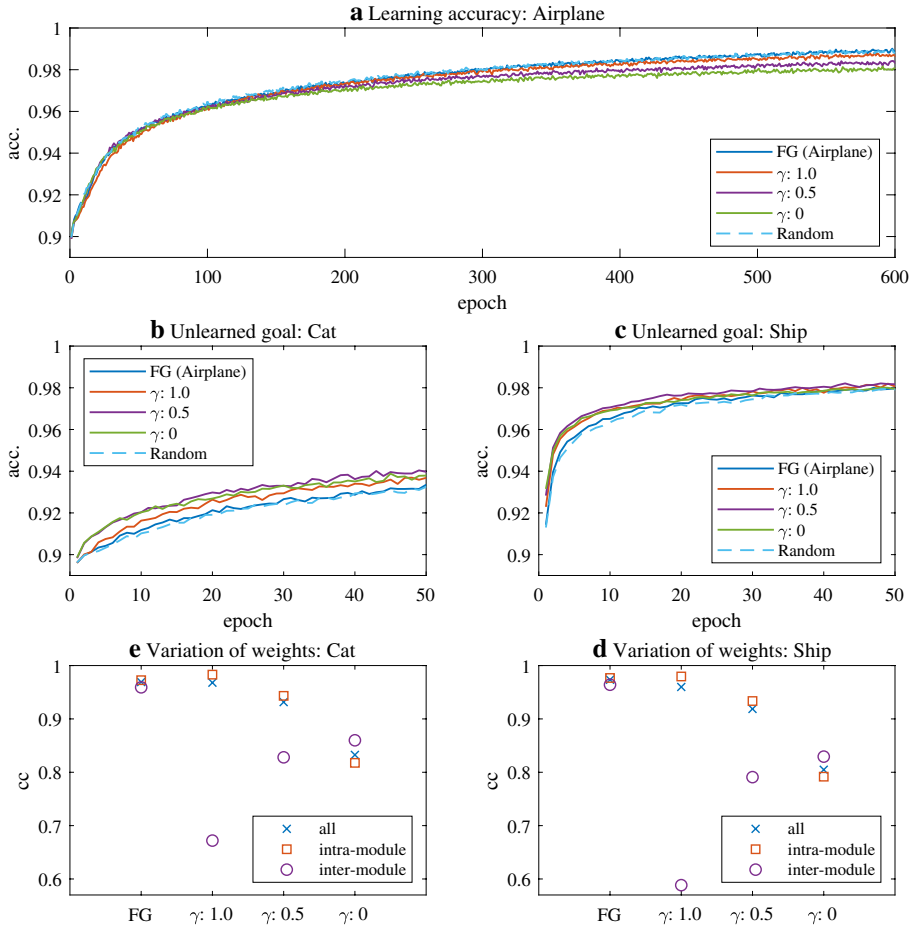


Fig. 5 Learning with the proposed approach. **a** Categorical accuracy against the goal. Blue indicates the FG of Airplane. The orange line indicates interMOD$_{1.0}$. The purple line indicates interMOD$_{0.5}$. The green line indicates interMOD$_0$. **b**, **c** Categorical accuracy against the unlearned goals Cat and Ship. The weights of the initial neural network are from the last epoch of (**a**). **e**, **d** Pearson's correlation coefficient (cc) against before and after learning the unlearned goals Cat and Ship. The cc values of the filter elements of the initial neural network and the last neural network in (**b**, **c**) are shown. Blue indicates the cc values of all the filter elements. Red indicates the cc values of elements in the filter corresponding to intra-module links. Purple indicates the cc values of elements in the filter corresponding to inter-module links

obtained modules increases, i.e., the inter-module links, which are the restricted links. NetworkX[2] is used for calculation.

The details regarding the learning algorithm are shown in Algorithm 2.

Evaluation

The dataset used for evaluation is the same as that in "Evaluation" section.

The results of setting γ to 1.0, 0.5, and 0 are shown in Fig. 5. We use interMOD$_{1.0}$, interMOD$_{0.5}$, and interMOD$_0$ to represent each method. *reducePressure* and *renewEpoch* are set to 0.99 and 10 respectively after several trials. The number of modules in the neural network obtained in the last epoch are 2, 15, and 195, and the percentage of

[2] https://networkx.github.io/.

inter-module links that are restricted to the total is 32.16%, 89.82%, and 100%, respectively. Since the learning rate used in the previous section seems too aggressive for learning a single goal, the learning rate is changed to 0.01 in this section, which is originally used in Other programs[3]

Learning accuracy

To show whether it is well learned, we show the learning accuracy of those networks compared with the FG. Figure 5a shows the result, which reveals that the FG has the highest learning accuracy in the last epoch, followed by $\text{interMOD}_{1.0}$. Then, it decreases as γ decreases. The result is related to the number of restricted links, that is, the order of the learning accuracy is in descending order of the number of restricted links. The greater the number of restricted links, the worse the learning accuracy becomes, as expected.

Modularity

Next, we show the modularity of the neural network obtained in the last epoch of Fig. 5a. The modularity is 0.4432 for $\text{interMOD}_{1.0}$, 0.5506 for $\text{interMOD}_{0.5}$, 0.2694 for interMOD_{0}, and 0.2339 for the FG. The result of any γ we tested is larger than that of the FG. Since γ is introduced to enhance modularity, the result is what we expected. Moreover, we can see from the result that $\text{interMOD}_{0.5}$ has the highest modularity compared to that of interMOD_{0} or $\text{interMOD}_{1.0}$. Since every link is restricted in interMOD_{0}, although the modularity is higher than that of the FG, which can be expected since connection cost reduction is known to enhance modularity (Ellefsen et al. 2015), the restricted links do not contribute to the enhancement of a particular module, and the result is within expectation. For $\text{interMOD}_{1.0}$, since the number of modules obtained in the last epoch is 2, the reason why the modularity of $\text{interMOD}_{1.0}$ is lower than that of $\text{interMOD}_{0.5}$, which ultimately obtained 15 modules, could be that modularity is not sufficiently promoted by $\text{interMOD}_{1.0}$. Therefore, for $\text{interMOD}_{0.5}$, the number of inter-module links is moderately restricted, and as a result, the modularity of $\text{interMOD}_{0.5}$ becomes the highest.

Accuracy relative to unlearned goals

Next, we show the accuracy relative to unlearned goals. Figure 5b, c show the accuracy of learning Cat and Ship, respectively. The result shows that any γ we tested is larger than that of the FG. Specifically, the one with the highest modularity, $\text{interMOD}_{0.5}$, yields the highest results for both goals, which is also what we expected. Note that, the result is less than MVG-obtained neural network. However, since this result is obtained only by learning a single goal, the difference are inevitable. By focusing on the learning accuracy again, we can see that the learning accuracy of $\text{interMOD}_{0.5}$ is higher than interMOD_{0} although the accuracy for unlearned goals increases faster in $\text{interMOD}_{0.5}$ than that in interMOD_{0}, which also supports the conclusion that γ equal to 0.5 is the best. Evaluation for other goals are shown in Appendix 5.

[3] https://github.com/fchollet/keras.

Topological analysis

From the result above, we can conclude that the proposed method with γ equal to 0.5 can lead to a neural network with a higher modularity that can learn unlearned goals faster. To show more details, we show the cc against the change in elements before and after learning an unlearned goal. Figure 5d, e show the results of Cat and Ship, respectively. The cc values of all elements become smaller as γ decreases. For $\text{interMOD}_{1.0}$, since the cc is close to that of the FG, $\text{interMOD}_{1.0}$ could be said to have a less variable topology as the FG. On the other hand, since the cc of interMOD_0 is relatively smaller, interMOD_0 could be said to have a variable topology. However, since the order of cc of intra-module and inter-moduel links reversed, we can say the topology is not maintaining intra-module links but seems varying them evenly across all links. For, $\text{interMOD}_{0.5}$, it has an intermediate cc, and from the cc values of the intra-module and inter-module links, we can say that it relatively maintains the intra-module links and leaves the inter-module links variable, which is the property we observed in "Applying MVG to obtain a modular structure" section. Therefore, we can conclude that the proposed method with γ equal to 0.5 can result the same topological property and function as those of the MVG-obtained neural networks.

Comparison with a random method

To show the need to restrict inter-module links, we compare our method with one that restricts almost the same number of links as for $\text{interMOD}_{0.5}$, i.e., 90% of the links, but are randomly selected every epoch. The resulting accuracy is shown in Fig. 5a–c as Random. From the results, we can see that the result of Random is almost the same as that of the FG. From this comparison, we can conclude that only restricting the same number of links cannot result in faster learning under $\text{interMOD}_{0.5}$ in learning unlearned goals.

Other examples

As another example, we show the result of Dog as a representative animal. The number of modules obtained in the last epoch is 2, 9, and 195 for $\text{interMOD}_{1.0}$, $\text{interMOD}_{0.5}$, and interMOD_0, respectively. The percentage of inter-module links is 31.72%, 92.22%, and 100%, respectively. The modularity of the neural network obtained in the last epoch of Fig. 6a is 0.4394 for $\text{interMOD}_{1.0}$, 0.5346 for $\text{interMOD}_{0.5}$, 0.2343 for interMOD_0, and 0.2336 for the FG. We can see that the same tendency is obtained in the learning accuracy, evaluation of unlearned goals, modularity and cc.

Evaluation with another dataset (CIFAR-100) is shown in Appendix 6.

Conclusions and future works

In this paper, we aimed to solve the reverse problem of MVG to obtain a highly modular structure that can mitigate catastrophic forgetting so that it can also apply to realistic data. First, we showed that a configuration with a highly modular structure exists that can mitigate catastrophic forgetting by applying an MVG against a realistic dataset, i.e., CIFAR-10. To confirm the mitigation of catastrophic forgetting, we showed the learning accuracy against a previously unlearned goal, which is expected

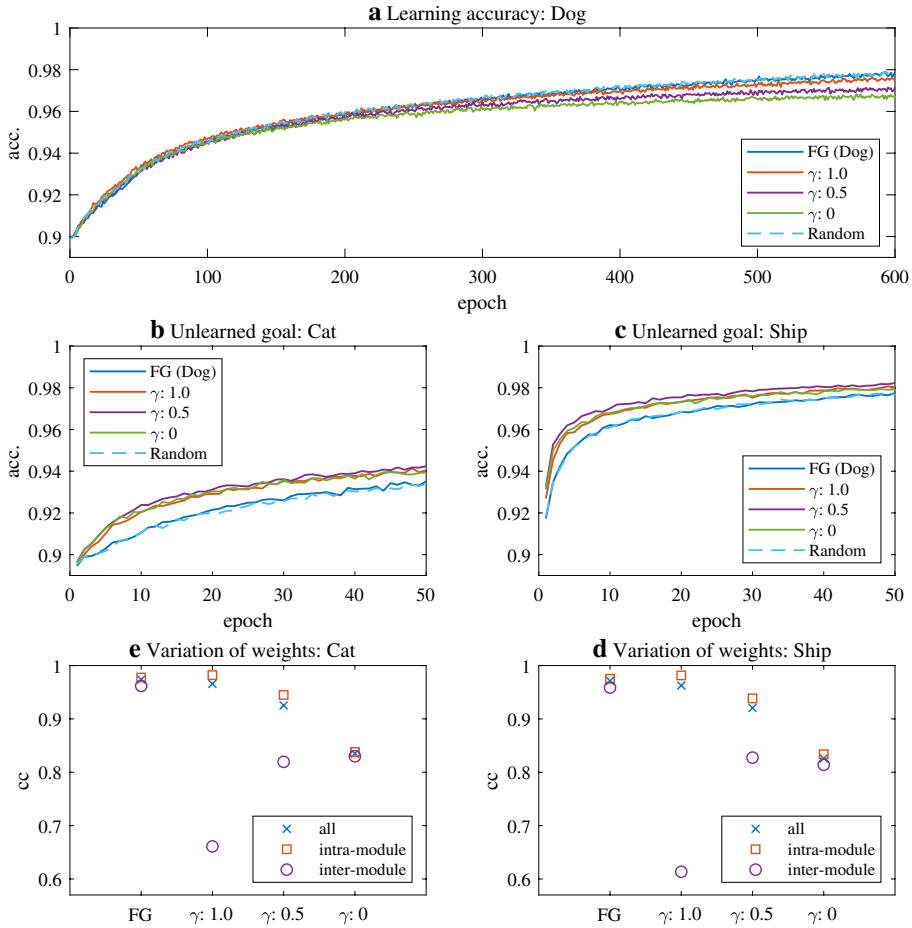


Fig. 6 Learning with the proposed approach (Dog) **a** Categorical accuracy against the goal. Blue indicates the FG of Dog. The orange line indicates interMOD$_{1.0}$. The purple line indicates interMOD$_{0.5}$. The green line indicates interMOD$_{0}$. **b**, **c** Categorical accuracy against the unlearned goals Cat and Ship. The weights of the initial neural network are from the last epoch of (**a**). **e**, **d** Pearson's correlation coefficient (cc) against before and after learning the unlearned goals Cat and Ship. The cc values of the filter elements of the initial neural network and the last neural network in (**b**, **c**) are shown. Blue indicates the cc values of all the filter elements. Red indicates the cc values of elements in the filter corresponding to intra-module links. Purple indicates the cc values of elements in the filter corresponding to inter-module links

to better than that in the case where the neural network catastrophically forgets a previously learned goal. Apparently, switching between a larger number of goals can enhance the modularity and learning unlearned goals faster. From a topological analysis of the obtained neural network, we found that relatively maintaining the intra-module elements while leaving the inter-module elements variable seems to be the key feature for learning unlearned goals faster. Next, we solved the reverse problem, that is, we proposed a method that can obtain a highly modular structure able to mitigate catastrophic forgetting. Since the MVG-obtained neural network can relatively maintain the intra-module elements while leaving the inter-module elements relatively variable, we proposed a method to restrict the inter-module weight

elements so that they can be relatively variable against intra-module ones. From the result, we showed that the obtained neural network can have the same topological features as those of the MVG-obtained neural network, can have a highly modular structure, and can learn an unlearned goal faster than without this method.

In future work, the proposed approach should be examined on other tasks and with other layer structures, and a theoretical analysis should be carried out. Also, it would be interesting to investigate other types of neural network.

Abbreviations
MVG: Modularly varying goal; FG: Fixed goal.

Acknowledgements
The computational resources were partially provided by the large-scale computer systems at the Cybermedia Center, Osaka University. Thanks to Nadav Kashtan for providing his source code.

Authors' contributions
LC and MM conceived the idea. LC performed the experiments and wrote the article with advice from MM. All authors read and approved the final manuscript.

Funding
This work was supported by JSPS KAKENHI Grant Number JP19K20415.

Availability of data and materials
The image data has been obtained from CIFAR-10. https://www.cs.toronto.edu/~kriz/cifar.html.

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Department of Information and Human Science, Kyoto Institute of Technology, Kyoto, Japan. [2] Department of Information Science and Technology, Osaka University, Osaka, Japan.

Appendix 1: Algorithm of FG

Algorithm 3 Algorithm for FG

```
INPUT: goal, maxEpoch
Initialize CNN
t ← 0
while ( t < maxEpoch ) do
    Execute backpropagation on CNN with loss function
    t ← t + 1
end while
```

The FG algorithm is shown in Algorithm 3.

Appendix 2: Learning rate

Evaluation with a learning rate of 0.1 for Fig. 2b FG is shown in Fig. 7. There is no large difference compared to the case with the learning rate set to 0.07

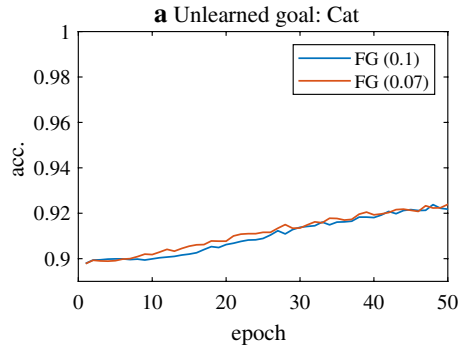


Fig. 7 Evaluation with learning rate 0.1

Appendix 3: FG learned two goals

Here, we provide an example of a neural network learning two goals sequentially in the FG manner. To create a catastrophically forgetting situation, the network is trained on Truck for 100 epochs and then on Airplane for 600 epochs (Fig. 8a). Although we assumed that learning a single goal was equivalent to catastrophically forgotting the previous goal in our original paper, this example clearly shows a neural network that meets two goals. From the result, we can see that the accuracy against Cat is almost the same as that of the original FG (Fig. 8b). Note that the modularity of the obtained neural network is 0.1933, which is close to the original FG.

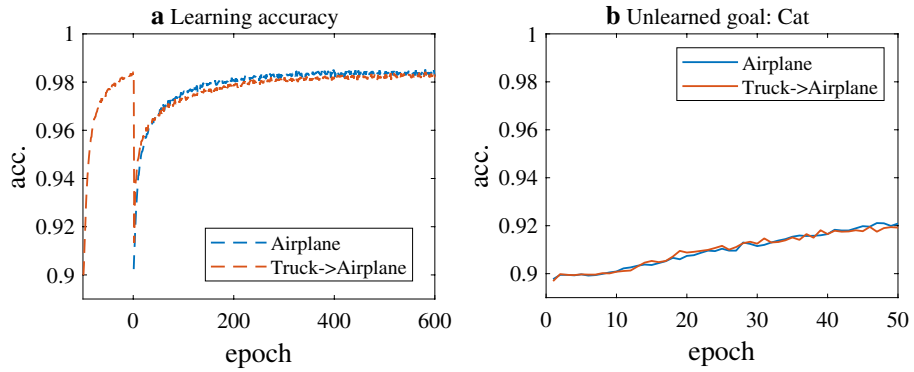


Fig. 8 Learning two goals sequentially in FG manner

Appendix 4: Evaluation with other goals for Fig. 2b MVG

Evaluation with other goals for Fig. 2b MVG is shown in Fig. 9.

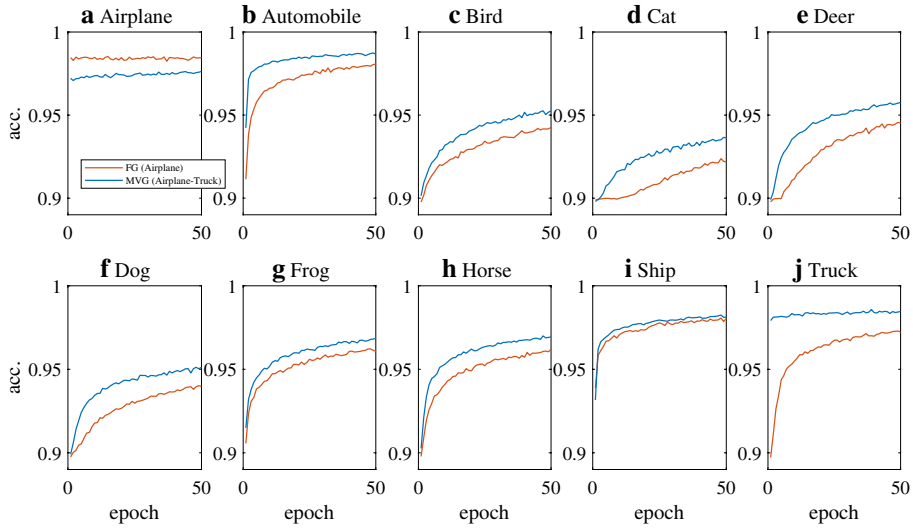


Fig. 9 Evaluation with other goals for Fig. 2b MVG

Appendix 5: Evaluation with other goals for Fig. 5b interMOD$_{0.5}$

Evaluation with other goals for Fig. 5b interMOD$_{0.5}$ is shown in Fig. 10.

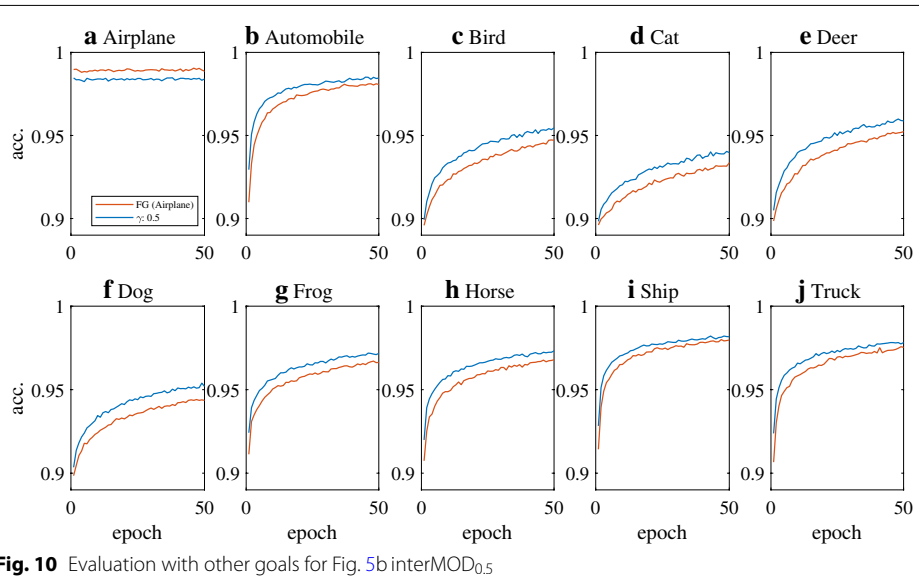


Fig. 10 Evaluation with other goals for Fig. 5b interMOD$_{0.5}$

Appendix 6: Evaluation with CIFAR-100

As an evaluation to another dataset, evaluation with CIFAR-100[4] is shown in this section. The super-class labels of CIFAR-100 are used, in which 60000 images are grouped into 20 classes.

Evaluation with CIFAR-100 for Fig. 2 is shown in Fig. 11. Vehicles I and Household furniture are chosen as evaluation goals. The evaluation settings are the same with that of Fig. 2. Note that the modularity of the FG obtained neural network (last epoch at Fig. 11a) is 0.1982, and the MVG obtained neural network is 0.2193. Same tendency with Fig. 2 can be observed.

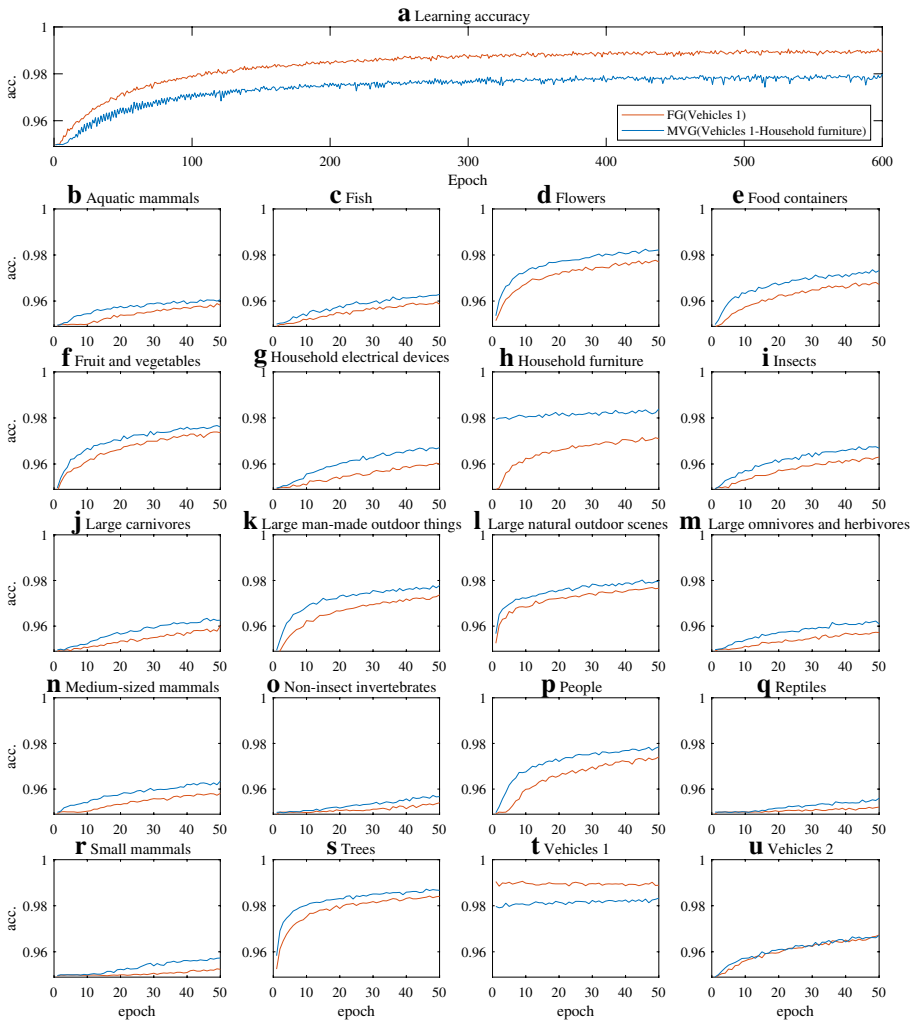


Fig. 11 Evaluation with CIFAR-100 for Fig. 2

[4] https://www.cs.toronto.edu/~kriz/cifar.html.

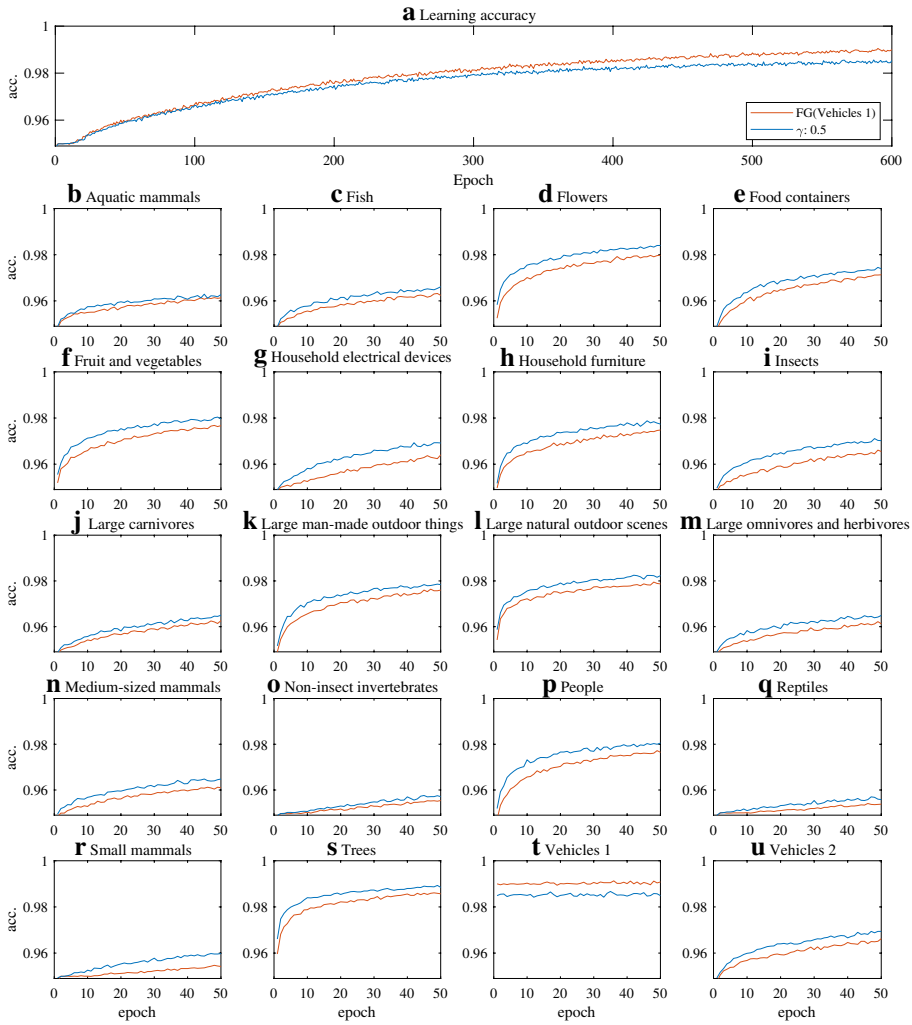


Fig. 12 Evaluation with CIFAR-100 for Fig. 5

Evaluation with CIFAR-100 for Fig. 5 is shown in Fig. 12. Vehicles I is chosen as an evaluation goal. The evaluation settings are the same with that of Fig. 5. Note that the modularity of interMOD$_{0.5}$ is 0.5735, and that of FG is 0.2563. Same tendency with Fig. 5 can be observed.

Received: 23 July 2020 Accepted: 27 October 2020
Published online: 26 November 2020

References

Amer M, Maul T (2019) A review of modularization techniques in artificial neural networks. Artif Intell Rev 52(1):527–561
Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 2008(10):10008
Clune J, Mouret J-B, Lipson H (2013) The evolutionary origins of modularity. Proc R Soc B Biol Sci 280(1755):20122863
Damicelli F, Hilgetag CC, Hütt M-T, Messé A (2019) Topological reinforcement as a principle of modularity emergence in brain networks. Netw Neurosci 3(2):589–605
Ellefsen KO, Mouret J-B, Clune J (2015) Neural modularity helps organisms evolve to learn new skills without forgetting old skills. PLoS Comput Biol 11(4):1004128
Filan D, Hod S, Wild C, Critch A, Russell S (2020) Pruned neural networks are surprisingly modular. arXiv:2003.04881
French RM (1999) Catastrophic forgetting in connectionist networks. Trends Cogn Sci 3(4):128–135
Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of international conference on artificial intelligence and statistics, pp 249–256
Goodfellow IJ, Mirza M, Xiao D, Courville A, Bengio Y (2014) An empirical investigation of catastrophic forgeting in gradientbased neural networks. In: Proceedings of international conference on learning representations (ICLR)
Hagberg A, Swart P, Schult D (2008) Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Laboratory
Kashtan N, Alon U (2005) Spontaneous evolution of modularity and network motifs. Proc Natl Acad Sci (PNAS) 102(39):13773–13778
Kashtan N, Noor E, Alon U (2007) Varying environments can speed up evolution. Proc Natl Acad Sci (PNAS) 104(34):13711–13716
Kashtan N, Mayo AE, Kalisky T, Alon U (2009) An analytically solvable model for rapid evolution of modular structure. PLoS Comput Biol 5(4):1000355
Kemker R, McClure M, Abitino A, Hayes TL, Kanan C (2018) Measuring catastrophic forgetting in neural networks. In: Proceedings of AAAI conference on artificial intelligence, pp 3390–3398
Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, Milan K, Quan J, Ramalho T, Grabska-Barwinska A, Hassabis D, Clopath C, Kumaran D, Hadsell R (2017) Overcoming catastrophic forgetting in neural networks. Proc Natl Acad Sci (PNAS) 114(13):3521–3526
Lee S-W, Kim J-H, Jun J, Ha J-W, Zhang B-T (2017) Overcoming catastrophic forgetting by incremental moment matching. In: Proceedings of advances in neural information processing systems (NIPS), pp 4652–4662
Li Y, Wang S, Tian Q, Ding X (2015) A survey of recent advances in visual feature detection. Neurocomputing 149(B):736–751
Newman ME (2006) Modularity and community structure in networks. Proc Natl Acad Sci (PNAS) 103(23):8577–8582
Parter M, Kashtan N, Alon U (2008) Facilitated variation: how evolution learns from past environments to generalize to new environments. PLoS Comput Biol 4(11):1000206
Ramesh B, Yang H, Orchard GM, Le Thi NA, Zhang S, Xiang C (2019) DART: distribution aware retinal transform for event-based cameras. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2019.2919301
Watanabe C, Hiramatsu K, Kashino K (2018) Modular representation of layered neural networks. Neural Netw 97:62–73

Publisher's Note