LETTER Communicated by Si Wu

Progressive Interpretation Synthesis: Interpreting Task Solving by Quantifying Previously Used and Unused Information

Zhengqi He
zhengqi.he@riken.jp
Lab for Neural Computation and Adaptation, RIKEN Center for Brain Science, Saitama 351-0198, Japan

Taro Toyoizumi
taro.toyoizumi@riken.jp
Lab for Neural Computation and Adaptation, RIKEN Center for Brain Science, Saitama 351-0198, Japan, and Department of Mathematical Informatics, Graduate School of Information Science and Technology, the University of Tokyo, Tokyo 113-8656, Japan

A deep neural network is a good task solver, but it is difficult to make sense of its operation. People have different ideas about how to interpret its operation. We look at this problem from a new perspective where the interpretation of task solving is synthesized by quantifying how much and what previously unused information is exploited in addition to the information used to solve previous tasks. First, after learning several tasks, the network acquires several information partitions related to each task. We propose that the network then learns the minimal information partition that supplements previously learned information partitions to more accurately represent the input. This extra partition is associated with unconceptualized information that has not been used in previous tasks. We manage to identify what unconceptualized information is used and quantify the amount. To interpret how the network solves a new task, we quantify as meta-information how much information from each partition is extracted. We implement this framework with the variational information bottleneck technique. We test the framework with the MNIST and the CLEVR data set. The framework is shown to be able to compose information partitions and synthesize experience-dependent interpretation in the form of meta-information. This system progressively improves the resolution of interpretation upon new experience by converting a part of the unconceptualized information partition to a task-related partition. It can also provide a visual interpretation by imaging what is the part of previously unconceptualized information that is needed to solve a new task.

Neural Computation **35, 38–57** (2023)
https://doi.org/10.1162/neco_a_01542

1 Introduction

Deep neural networks (DNNs) have made great achievements in fields such as image recognition (Krizhevsky, Sutskever, & Hinton, 2017), speech recognition (Hinton et al., 2012), natural language processing (Vaswani et al., 2017), and game playing beyond human-level performance (Silver et al., 2016). DNNs, however, are famous black-box models. That they fail under certain circumstances, such as adversarial attack (Goodfellow, Shlens, & Szegedy, 2014), motivates increasing research into understanding how DNNs solve tasks or model interpretation. More recent research also suggests that better model interpretation can be useful to, for example, explanation about model behavior, knowledge mining, ethics, and trust. (Doshi-Velez & Kim, 2017; Lipton, 2018)

Researchers have proposed different approaches to proceed with model interpretation; for example, concerning the interpretation style, the post hoc style tries to separate the model training step and model interpretation step, and the concurrent style aims simultaneously for task performance as well as interpretation (Lipton, 2018). As for the applicability of interpretation methods, the model-specific type targets a certain class of models, and with the model-agnostic type, the interpretation method does not depend on the model (Arrieta et al., 2020). Considering the scope of interpretation, global interpretation gives information about how the task is solved from a broader view, and local interpretation is more focused on certain examples or parts of the model (Doshi-Velez & Kim, 2017). There are also diverse forms of interpretation, such as the information feature (Chen, Song, Wainwright, & Jordan, 2018), the relevance feature (Bach et al., 2015), a hot spot of attention (Hudson & Manning, 2018), or gradient information (Sundararajan, Taly, & Yan, 2017). Another stream of research proposes that interpretable models are usually simple ones: for example, discrete-state models (Hou & Zhou, 2018), shallower decision trees (Freitas, 2014; Wu et al., 2017), graph models (Zhang, Cao, Shi, Wu, & Zhu, 2017), or a small number of neurons (Lechner et al., 2020). (See Arrieta et al., 2020, for a more detailed overview.)

One particular dimension for model interpretation related to our letter is how much preestablished human knowledge is needed. Methods that require high human involvement, such as interpretation with human predefined concepts (Koh et al., 2020; Chen, Bei, & Rudin, 2020) or with large human-annotated data sets (Kim, Tapaswi, & Fidler, 2018), implicitly assume the background knowledge of an average human to make sense of the interpretation, which is hard to define rigorously. Contrarily, existing human-agnostic methods transfer interpretation into some measurable form, such as the depth of the decision tree (Freitas, 2014; Wu et al., 2017). However, how well this kind of measure is related to human-style interpretation is under debate.

Within the human-agnostic dimension of interpretation, we extend the discussion with two new perspectives. One perspective starts with the simple idea that interpretation should be experience dependent. Motivated by this idea, we focus on the situation where the model learns a sequence of tasks by assuming that later tasks can be explained using earlier experiences. In other words, model interpretation in our framework is defined as meta-information describing how the information used to solve the new task is related to previous ones. The second perspective is motivated by the idea that interpretation should be able to handle the out-of-experience situation. In a situation where a new task cannot be fully solved by experience, the model interpretation method should be able to report new knowledge, mimicking a human explaining what is newly learned. We demonstrate that this framework can cast insight into how later tasks can be solved based on previous experience on MNIST and CLEVR data sets (Johnson et al., 2017) and express ignorance when experience is not applicable.

Our work is related to the concept bottleneck model (CBM) and concept whitening model (CWM; Koh et al., 2020; Chen et al., 2020) in the sense that meaningful interpretation of the current task depends on previously learned knowledge. However, these methods do not capture reasonable interpretation when the human-defined concepts alone are insufficient to solve downstream tasks (Margeloiu et al., 2021). In our framework, we add the unconceptualized region to take care of information not yet associated with tasks. Moreover, a recent study also shows that contamination of concept-irrelevant information in the predefined feature space can hamper interpretation (Mahinpei et al., 2021). We implement information bottleneck (IB; Tishby, Pereira, & Bialek, 2000) as a remedy to this information leak problem. Our method also shares similarities with variational information bottleneck for interpretation (VIBI) method (Bang, Xie, Lee, Wu, & Xing, 2019) and the multiview information bottleneck method (Wang, Boudreau, Luo, Tan, & Zhou, 2019) in the sense that these methods use IB to obtain minimal latent representation from previously given representations. However, unlike the multiview IB method for problem solving, the goal of our framework is to synthesize interpretation. Furthermore, our framework does so using macroscopic task-level representations, which is different from microscopic input-level representations used in VIBI.

2 Insight into Interpretation

This section discusses the intuition behind our framework for model interpretation.

2.1 Interpretation as Meta-Information. To quantify how a new task is solved using the experience of previous tasks, we evaluate meta-information. We define meta-information as a vector of mutual information, where each element of the vector describes how much the corresponding information partition is used for the new task.

2.1.1 Interpreting Using the Right Level. In this work, a machine learns a series of different tasks. The aim is to ascribe an interpretation of how the model solves the new task based on previous experience. If we did this using low-level features, such as the intensity and color of each pixel, the task description would become complicated. Instead, we aim to give an interpretation at a more abstract level—for example, "This new task is solved by combining the knowledge about tasks 2 and 4." To achieve this goal, information about the input is partitioned at the task level. We therefore prepare information partitions that encode useful features for each task.

2.1.2 Inducing Independence. These partitions have to satisfy certain conditions. If these information partitions are redundant, we will have arbitrariness in assigning meta-information since a task can equally be solved using different partitions (Wibral, Priesemann, Kay, Lizier, & Phillips, 2017). Therefore, inducing independence among partitions is preferred for having unambiguous meta-information. Useful methods are widely available in machine learning fields such as independent component analysis (Bell & Sejnowski, 1995; Hyvärinen & Oja, 2000) and variational autoencoders (Kingma & Welling, 2013).

2.1.3 Meaning Assignment. We have defined meta-information meta-information as a vector of Shannon information measured in bits (i.e., how much each information partition is used). Although the number of bits itself has no meaning, each entry of the vector is linked to a corresponding task. Hence, the meta-information can be mapped to the relevance of previous tasks.

2.2 Progressive Nature of Interpretation.

2.2.1 Progressive Interpretation. One important but usually ignored property of interpretation is that we interpret based on experience (National Research Council, 2002; Bada & Olusegun, 2015). Progressively learning multiple tasks is not a rare setting in machine learning (Andreas, Rohrbach, Darrell, & Klein, 2016; Rusu et al., 2016; Parisi, Kemker, Part, Kanan, & Wermter, 2019), which is usually referred to as "lifelong learning," "sequential learning," or "incremental learning." However, these studies usually focus on avoiding catastrophic forgetting and do not investigate how progressiveness contributes to interpretation. In one example, Kim et al. (2018), point out that interpretability emerges when lower-level modules are progressively made use of. We propose that interpretation should be synthesized in a progressive manner, where the model behavior is interpreted by how much the current task is related to previously experienced tasks.

2.2.2 Knowing You Don't Know. An experience-based progressive interpretation framework may inevitably encounter the situation when its previous experience does not help interpret the current task. To solve this problem, we introduce an unconceptualized partition, storing information not yet included in the existing information partitions. We noticed that this unconceptualized partition generates a "knowing you don't know" type of interpretation—a meta-cognition ability that allows a person to reflect on their knowledge, including what they don't know (Glucksberg & McCloskey, 1981). Under this situation, the design of the framework should be able to interpret knowing you don't know when faced with out-of-experience tasks.

We now formalize our insights in the language of information theory in the following sections.

3 The Progressive Interpretation Framework

Assume we have a model with stochastic input X, which is statistically the same regardless of a task. Task i is defined as predicting a series of stochastic labels Z_i. Its corresponding internal representation is Y_i. The progressive interpretation framework is formalized iteratively as follows:

1. Assume that after task n, a model has a minimal internal representation $Y = \{Y_1, Y_2, \ldots, Y_n, Y_{\text{else}}\}$ that encodes input X. Y_i describes the internal representation learned to solve task i. Y_{else} describes internal representation encoding X that is not yet used to solve and task. The optimization in the ideal case yields independence among the previous task-relevant partitions:

   $$I(Y_i; Y_j) = 0, (i \neq j \in [1, n] \cup \text{else}).$$

   Here, we define the notation $[1, n]$ to be $\{1, 2, 3, \ldots, n\}$.
2. Then the model is faced with the new task $n + 1$ and learns to predict Z_{n+1}. After learning Z_{n+1}, the model distills the necessary part $Y_{(i \cap n+1)}$ from each partition $Y_i (i = [1, n] \cup \text{else})$ for solving task $n + 1$. This is achieved by minimizing

   $$I(Y_{(i \cap n+1)}; Y_i), (i \in [1, n] \cup \text{else})$$

   while maintaining the best task performance, that is, by maintaining ideally all task-relevant information:

   $$I(\cup_{i=1}^{n,\text{else}} Y_i; Z_{n+1}) = I(\cup_{i=1}^{n,\text{else}} Y_{(i \cap n+1)}; Z_{n+1}).$$

3. The interpretation is defined as the meta-information of how much the individual partitions $\{Y_i\}$ for previous tasks $i \in [1, n] \cup \text{else}$ are used to solve task $n + 1$. Namely, the composition of the mutual

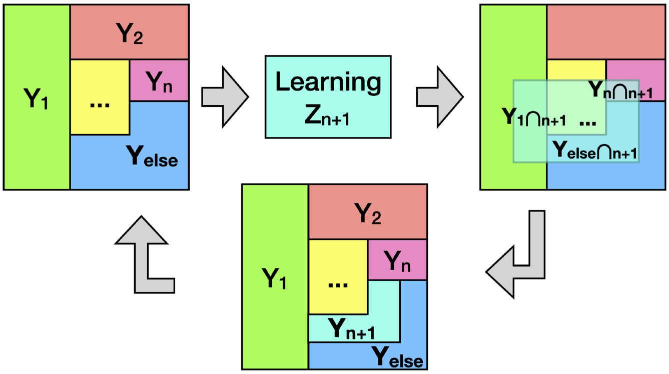


Figure 1: A schematic plot showing the intuition of the progressive interpretation framework. Interpretation in our framework is based on the meta-information that specifies from which partitions the needed information comes to solve a new task, Z_{n+1}. The map has the resolution in the level of task partitions Y_i, where partitions are made independent of each other. Independence among task partitions ensures the uniqueness of the needed information. Anything the model has not yet learned to use would stay in the unconceptualized Y_{else} region. The more tasks the model has encountered, the smaller the unconceptualized region would be. Thus, later tasks lead to better interpretation.

information $I(Y_{(i\cap n+1)}; Y_i)$ over the different partitions $i = [1, n] \cup \text{else}$ is the meta-information we use to interpret the global operation of the neural network. Then the local interpretation for each example is available from $\{Y_{(i\cap n+1)}\}$.

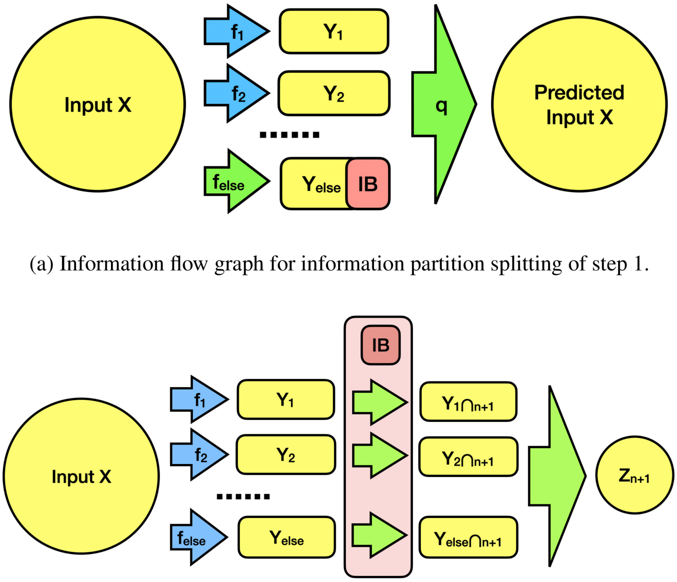
4. After task $n+1$, the model updates the representation partition by splitting Y_{else} into the newly added representation $Y_{(\text{else}\cap n+1)}$ and its complement, $Y_{\text{else}} \backslash Y_{(\text{else}\cap n+1)}$. Then the former is denoted as Y_{n+1} and the latter as new Y_{else}. The model would continue for further iteration and interpretation of the tasks.

The process is shown in Figure 1.

4 Implementation

Our particular interest is in the system involving neural networks. Since our framework is information-theoretic, all types of neural networks are treated equally as segments of information processing pipelines. Which type of neural network to choose depends on the specific problem.

Neural network implementation of progressive interpretation can be implemented as loops over the four steps set out in section 3. In step 1, we assume a network already has information maps for task 1-to-n. Then we extract the unconceptualized partition that is unrelated to task 1-to-n by



(a) Information flow graph for information partition splitting of step 1.

(b) Information flow graph for interpretation of step 2.

Figure 2: Information flow graph of the progressive interpretation framework. Yellow areas are representations, and green and blue arrows represent neural networks. Green ones are put under training while blue ones are fixed. The red square with IB represents the information bottleneck.

IB. In step 2, the model learns a new task, $n+1$. Then the interpretation is gained by knowing how much information is needed from each subregion as in step 3. In step 4, we repeat step 1 with a new map for task $n+1$ and prepare for the next loop. By adding new tasks and looping over the steps, a progressively more informative interpretation can be gained. The information flow chart implemented in the following sections is shown in Figure 2.

4.1 Information Bottleneck. In our framework, IB plays an important role in manipulating information flow. To predict label Z from statistical input X with inner representation Y, IB would maximize

$$\max_{\theta}[I(Y;Z) - \gamma I(Y;X)],\ Y = f_{\theta}(X, \epsilon), \tag{4.1}$$

where $\gamma \in [0, 1]$ is the scaling factor controlling the balance between the task performance (when γ is small) and having nonredundant information representation (when γ is large). f is a neural network parameterized by the parameter θ, and ϵ is a noise term that is important to suppress task-irrelevant information out of X.

We choose the variational information bottleneck (VIB) implementation (Alemi, Fischer, Dillon, & Murphy, 2016; Chalk, Marre, & Tkacik, 2016; Li & Eisner, 2019) with loss function

$$L(p, q, r) = \mathbb{E}_{Y,Z}\left[-\log q\left(Z \mid Y\right)\right] + \gamma \mathbb{E}_X\{\mathrm{KL}\left[p\left(Y \mid X\right), r(Y)\right]\} \tag{4.2}$$

to optimize the encoding distribution $p(Y|X)$, decoding distribution $q(Z|Y)$, and prior distribution $r(Y)$ for p. $\mathbb{E}_X$ describes taking the expectation over random variable X. Note that $\mathbb{E}_{Y,Z} = \mathbb{E}_X\mathbb{E}_{Y|X}\mathbb{E}_{Z|X}$. During the optimization, $\mathbb{E}_X\mathbb{E}_{Z|X}$ is computed by averaging over N training samples of input $\{x_j|j = 1, \ldots, N\}$ and label $\{z_j|j = 1, \ldots, N\}$. $\mathbb{E}_{Y|X}$ is the average over the encoding distribution $p(Y|X)$, which is computed using the mapping $Y = f_\theta(X, \epsilon)$ of the encoding neural network. Y can be a vector of either continuous or discrete variables (Li & Eisner, 2019) (see appendix section 3 for details). For clarity, we further simplify the notation of loss function to be

$$L = Q(Z|Y) + \gamma \mathrm{KL}(Y) \tag{4.3}$$

for future use, where the Q term corresponds to the log-likelihood term trying to approximate Z from internal representation Y. The KL term corresponds to the KL-divergence term trying to control the expressiveness of Y.

4.2 Task Training and Information Partition Splitting. Suppose a new model with task input X learns its first task to predict label Z_1. It is not difficult to train a neural network for this task by optimization: $\min_\theta D(f_{1,\theta}(X)||Z_1)$, where D is a distance function, such as KL divergence or mean-square error, which is decided by the problem. $f_{1,\theta}$ is an encoder network parameterized by θ. After training, we will be able to obtain the representation of task 1 as $Y_1 = f_1(X, \epsilon)$, where f_1 indicates a neural network$f_{1,\theta}$ after optimizing θ.

Our next problem is how to obtain task 1 unrelated representation Y_{else}, which ideally satisfies $I(Y_1; Y_{\text{else}}) = 0$, to complement the intermediate representation about the input. Here, we propose that Y_{else} can be obtained via the implementation of IB on an autoencoding task:

$$\begin{aligned}
&\max_\theta[I(Y_1, Y_{\text{else}}; X) - \gamma I(Y_{\text{else}}; X)],\\
&Y_{\text{else}} = f_{\text{else},\theta}(X, \epsilon),
\end{aligned} \tag{4.4}$$

where γ is again the scaling factor controlling the trade-off between including and excluding different information. Note that the learned f_1 function is fixed while $f_{\text{else},\theta}$ is trained. The intuition behind equation

4.4 is described as follows. $I(Y_1; Y_{\mathrm{else}}) > 0$ implies redundant information about Y_1 contained in Y_{else}. This redundant information would not improve $I(Y_1, Y_{\mathrm{else}}; X)$. However, removing this redundant information can decrease $I(Y_{\mathrm{else}}; X)$, thus contributing to our optimization goal. Note that we assume γ is less than one.

With the simplified notation of the VIB introduced above, the loss function

$$L = Q(X|Y_1, Y_{\mathrm{else}}) + \gamma \mathrm{KL}(Y_{\mathrm{else}}) \tag{4.5}$$

is minimized. The loss function seeks to autoencode X given previously learned Y_1 (which is fixed) together with Y_{else}, while controlling expressiveness of Y_{else}.

4.3 New Task Interpretation. Now assume the model has internal representation $Y = \{Y_1, Y_2, \ldots, Y_n, Y_{\mathrm{else}}\}$ after learning tasks 1 to n. When the new task $n+1$ is introduced, the model learns to predict Z_{n+1}. Task $n+1$ relevant information can be extracted from Y by the IB as follows:

$$\begin{aligned} &\max_{\theta} \left[I(\cup_{i=1}^{n,\mathrm{else}} Y_{(i\cap n+1)}; Z_{n+1}) - \gamma \sum_{i=1}^{n,\mathrm{else}} I(Y_{(i\cap n+1)}; Y_i) \right], \\ &Y_{(i\cap n+1)} = f_{(i\cap n+1),\theta}(Y_i, \epsilon), \end{aligned} \tag{4.6}$$

where $Y_{(i\cap n+1)}$, $(i \in [1, n] \cup \mathrm{else})$ is the information needed from Y_i to solve task $n+1$. Again, ϵ is the noise term required to eliminate information irrelevant to task $n+1$. Since $Y_{(i\cap n+1)} = f_{(i\cap n+1),\theta}(Y_i, \epsilon)$ depends on Y_i, together with IB, $Y_{(i\cap n+1)}$ is then a minimum subpartition of Y_i required for task $n+1$. We again implement the variational IB loss function with simplified notation:

$$L = Q(Z_{n+1}| \cup_{i=1}^{n,\mathrm{else}} Y_{(i\cap n+1)}) + \frac{\gamma}{n+1} \sum_{i=1}^{n,\mathrm{else}} \mathrm{KL}(Y_{(i\cap n+1)}). \tag{4.7}$$

The loss function seeks to maximize the prediction of Z_{n+1} while controlling the needed information from Y_i. Index i specifies a representation partition.

After getting $\{Y_{(i\cap n+1)}\}$, we can derive an interpretation as the meta-information $I(Y_{(i\cap n+1)}; Y_i)$ needed from each partition Y_i as defined in section 3. We can also look into the representations of $Y_{(i\cap n+1)}$ to gain insight into how task $n+1$ is solved for each example.

$Y_{(\mathrm{else}\cap n+1)}$ is the information needed from the unconceptualized partition Y_{else} to solve task $n+1$. We can rewrite this to be Y_{n+1} and define the new unconceptualized partition as $Y_{\mathrm{else}} \leftarrow Y_{\mathrm{else}} \backslash Y_{(\mathrm{else};n+1)}$. We can then go back to step 1 and continue the iteration for task $n+2$.

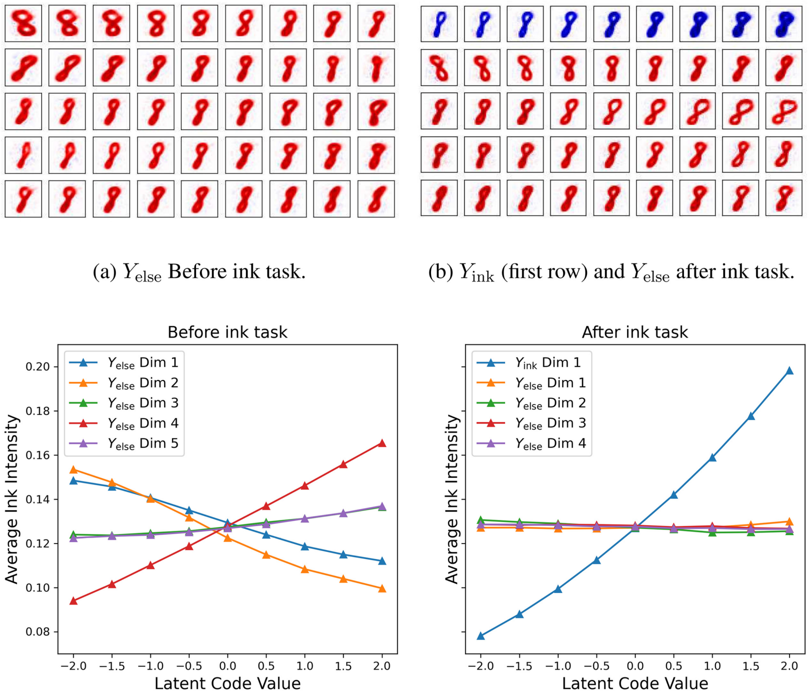
5 Experiments

5.1 MNIST Data Set. We first illustrate our progressive interpretation framework on the MNIST data set (60,000/10,000 train/test splits). We set task 1 as digit recognition. For task 2, we propose three kinds of tasks: determining if a number is even or odd (parity task), predicting the sum of pixel intensities (ink task), or a task that involves both digit information and pixel intensity information with a certain resolution (see below). First, we train a network f_1 to perform digit recognition, and then we train an autoencoder with IB to train a network f_{else} to obtain a digit-independent partition. Then we extend the network to train on a second task and obtain interpretation from the information flow. We choose continuous latent representation for this section. (See appendix sections 1 and 2 for implementation details.)

5.1.1 IB Removes Task-Relevant Information from the Unconceptualized Region. Unconceptualized representation can be obtained after the autoencoding step. We can check what has been learned by scanning this latent code. Figure 3a shows the scanning result of the top five latent representation units, ordered by descending mutual information with X. Note that changing these features does not change the digit. Moreover, mutual information between Y_{digit} and Y_{else} is estimated by training a neural network that predicts Y_{digit} from Y_{else}. The estimated information is smaller than 0.1 Nat when γ is larger than 5e-4, which indicates that digit information is removed from the un-conceptualized region by IB.

5.1.2 The Framework Explains How a New Task is Solved. After the autoencoding step, we proceed to solve either the parity task or ink task to study the interpretation that the framework provides. For the parity task, mutual information from Y_{digit} and Y_{else} are 0.702 Nat and 0.002, Nat respectively, and for the ink task, 1.498 Nat and 2.045 Nat. The result shows that the parity task doesn't need information from Y_{else}, while the ink task does. Clues of how the tasks are solved can also be found by looking into the representation obtained after IB. For the parity task, different digits are clustered into two groups according to their parity. For the ink task, digits are aligned in an order corresponding to their actual average ink amount ($0 > 8 > 2 > 3 > 6 > 5 > 9 > 4 > 7 > 1$), as Figure 4 shows.

5.1.3 Experience-Dependence of the ELSE Partition. After learning the digit and the ink tasks, we can update the autoencoder f_{else} to exclude the ink-task-related information. On the one hand, Y_{ink} (the first row of Figure 3b) represents the average pixel intensity. On the other hand, this information is suppressed in Y_{else} (rows 2–5). The suppression can be measured by feature correlation between Y_{ink} and Y_{else}. Before the ink task, the correlations are (0.295, 0.414, 0.080, 0.492, 0.100) for the five units visualized, but after the ink task, the correlation becomes (0.030, 0.194, 0.019, 0.028, 0.001). We



(a) Y_{else} Before ink task.

(b) Y_{ink} (first row) and Y_{else} after ink task.

(c) Average ink intensity v.s. latent code value of the five dimensions before and after the ink task.

Figure 3: Latent code scanning of unconceptualized representation after autoencoding, before (a) and after (b) ink task (except the first row). The reconstructed images plotted as the activity (columns) of one of the coding units (rows) are varied with others fixed. (c) Shows how the average ink intensity varies when we scan the latent code of the same five units as in panels a and b.

also present the result of the average ink intensity versus the latent code of the five units. It can clearly be seen that before the ink task, the knowledge of average intensity is distributed across all five units. However, after the ink task, the knowledge of average intensity is extracted as Y_{ink} and removed from Y_{else} (see Figure 3c). The result indicates that the unconceptualized region is experience dependent, and information about the already learned task is excluded. Unlike other frameworks such as variational autoencoder (Kingma & Welling, 2013) and infoGAN (Chen et al., 2016), which usually have no explicit control over partitioning latent representation, our framework allows latent representation reorganization through progressive tasks.

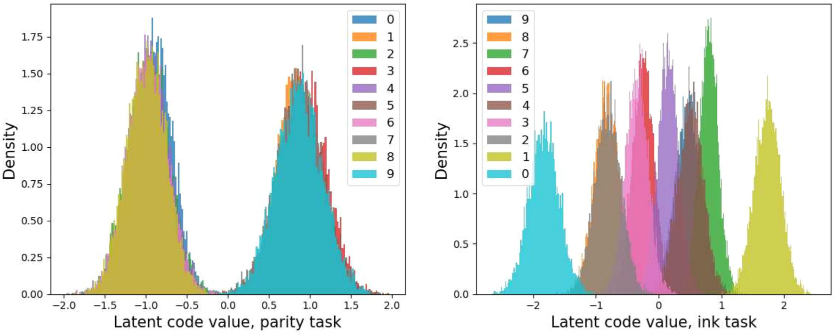


Figure 4: VIB latent code distribution of different digits for the parity task $Y_{\text{digit}\cap\text{parity}}$ (left) and ink task $Y_{\text{digit}\cap\text{ink}}$ (right). The x-axis shows the code value, and the y-axis shows the code density; different colors represent different digits ranging from 0 to 9. For the parity task, the latent code formed two clusters, one for even numbers and one for odd numbers. And for the ink task, digits are aligned in the order of the average amount of ink.

5.1.4 Quantitative Benchmark of Interpretation. Next, we ask if our proposed interpretation is quantitatively useful. Because we are not aware of task-level, human-agnostic interpretation algorithms directly comparable to ours, we study how the interpretation changes as we systematically modify the required type of information for task 2. Task 2 is designed to require both digit information and digit-independent ink information involving different resolutions. For digit information, we have four resolutions: d1, d2, d5, and d10. For example, d5 means that 10 digits are separated into five equally sized groups, and the task is to tell which group the image belongs to. As a result, (0, 0.693, 1.609, 2.303) Nat of information about digits is theoretically needed, respectively. For digit-independent ink information, we also have four resolutions (according to the percentile-based grouping for each digit by the amounts of ink used): s1, s2, s3, and s4, which theoretically require (0, 0.693, 1.099, 1.386) Nat of information. By combining them, we get 16 possibilities for task 2; the interpretation measured as mutual information and the corresponding theoretical values are shown in Figure 5. The figure shows that information needed from Y_{digit}, $I(Y_{\text{digit}\cap 2}; Y_{\text{digit}})$, can be precisely predicted. The required nondigit information $I(Y_{\text{else}\cap 2}; Y_{\text{else}})$ from Y_{else} via autoencoding correlates with the required amount to solve the task. However, due to the imperfection of the variational IB algorithm to purely extract relevant information, more than the theoretically required amount of information from Y_{else} is used for good performance. This problem can be practically remedied by allowing Y_{else} to be retrained by adding an auxiliary autoencoding task when learning task 2. Since input data are available during task 2, adding an auxiliary autoencoding task during task 2

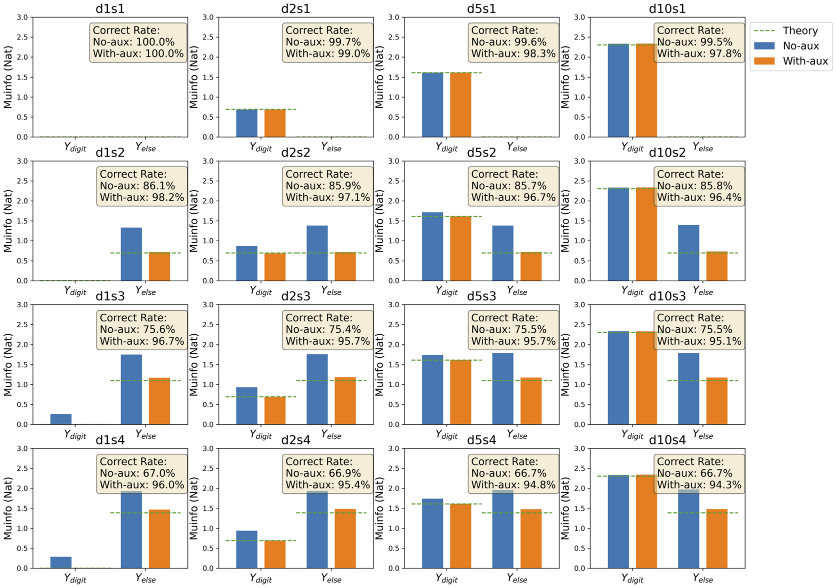


Figure 5: Mutual information from Y_{digit} and Y_{else} used to solve the benchmark tasks. Blue/orange bars are mutual information without/with auxiliary autoencoding, and the theory indicating the required amount of information is plotted with the green dotted line. Inside text boxes are task-correct rates without/with auxiliary autoencoding. The title of each panel represents different task types combining four digit resolutions—d1, d2, d5, and d10—and four digit-independent ink resolution—s1, s2, s3, and s4—forming a 4-by-4 matrix.

training increases task performance without needing extra data. (See appendix section 9 for further discussion.)

5.2 CLEVR Data Set. In this section, we demonstrate the progressive interpretation framework on the CLEVR data set (Johnson et al., 2017), a large collection of 3D-rendered scenes (70,000/15,000 train/test splits) with multiple objects with compositionally different properties. The CLEVR data set was originally designed for a visual question-answering task, but we train the model without using natural language. For example, we train the model to classify the color of an object or conduct a multiple-choice (MC) task using only pictures. For the MC task, the model is trained on a large set of four pictures and learns to choose one of the four pictures that includes a target object (100,000/20,000 train/test splits).

In this section, we divide the tasks into two groups. In task group 1, the model that is pretrained to tell objects apart learns to recognize three of the important properties, position, color, and material, among shape, size, color, material, and position. In task group 2, the model is asked to perform an MC task selecting a picture according to a specific context, for example, "Choose the picture with red cubes," which needs information learned or not yet learned in task 1. For task group 1, we first use convolutional neural networks (CNNs) to report the image properties by supervised learning and then obtain the unconceptualized region via autoencoding. After that, task group 2 is performed with interpretation synthesized. We choose discrete latent representation for this section. (See appendix sections 1 and 2 for implementation details.)

5.2.1 Interpretation by Information Flow. The result of interpretation by information flow is shown in Table 1. The mutual information $I(Y_{(i \cap \mathrm{MC})}; Y_i)$ for $i \in \{\text{posi, color, material, else}\}$ is measured in Nat per object, where MC represents the multiple-choice task. Different rows represent different question types. We sample five random initializations of the networks for each task and present both the average and standard deviations. The theoretical amount of information required for feature i is shown in parentheses. We can interpret how the model is solving the task by calculating mutual information coming from each information partition. For example, the task to "choose the picture with green metal" needs 0.345 Nat of information from the color domain and 0.686 Nat from the material domain. As expected, information coming from other domains is judged as irrelevant to this task. If the task is to "choose the picture with a small yellow object," the model then needs 0.343 Nat from the color domain, plus 0.70 Nat of information from the unconceptualized region since the model has not yet explicitly learned about object size. If the task is "choose the picture with a large sphere," the model finds out that all previously learned properties are useless and has to pick 0.31 Nat of information from the unconceptualized region. This is because neither size nor shape information has been used in previous tasks.

5.2.2 Single-Example Interpretation and Unconceptualized Representation. After getting the model, it is also possible to synthesize interpretation for a single example by looking into the discrete representation $Y_{(i \cap \mathrm{MC})}$ for $i \in \{\text{posi, color, material, else}\}$. A typical example is shown in Figure 6. This example corresponds to a "small yellow object." We can see the model discriminates if the object has the color "yellow" while neglecting position and material information. To solve the problem, the model also needs information from the unconceptualized partition, which is representing the size "small." The behavior of the model is consistent with the expectation of the question regarding the "small yellow object."

We examine the correctness of the unconceptualized representation by comparing it with the true label. For example, if the task is "choose the small yellow object," the unconceptualized region should represent the size "small." We can cross-check by calculating their mutual information, which is 0.662 Nat per object. For the case "choosing a red cube," mutual

Table 1: Table for Task 2 Interpretation.

Question Type	Position	Color	Material	Unknown	Correct rate
Green Metal	<0.001 (0)	0.345 ± 0.001 (0.377)	0.686 ± 0.008 (0.693)	<0.001 (0)	99.3 ± 0.1%
Left Rubber	0.56 ± 0.02(0.52)	<0.001 (0)	0.688 ± 0.001 (0.693)	0.01 ± 0.02 (0)	97.0 ± 0.7%
Small Yellow	<0.001 (0)	0.343 ± 0.002 (0.377)	<0.001 (0)	0.70 ± 0.01 (0.693)	99.2 ± 0.1%
Red Cube	<0.001 (0)	0.381 ± 0.002 (0.377)	<0.001 (0)	0.89 ± 0.06 (0.637)	95.8 ± 0.4%
Right Cylinder	0.59 ± 0.03 (0.51)	<0.001 (0)	<0.001 (0)	0.88 ± 0.06 (0.637)	94.8 ± 0.7%
Large Sphere	<0.001 (0)	<0.001 (0)	<0.001 (0)	0.31 ± 0.01 (0.451)	99.4 ± 0.1%

Note: The information unit (Nat/object), inside parentheses, is the theoretical value.

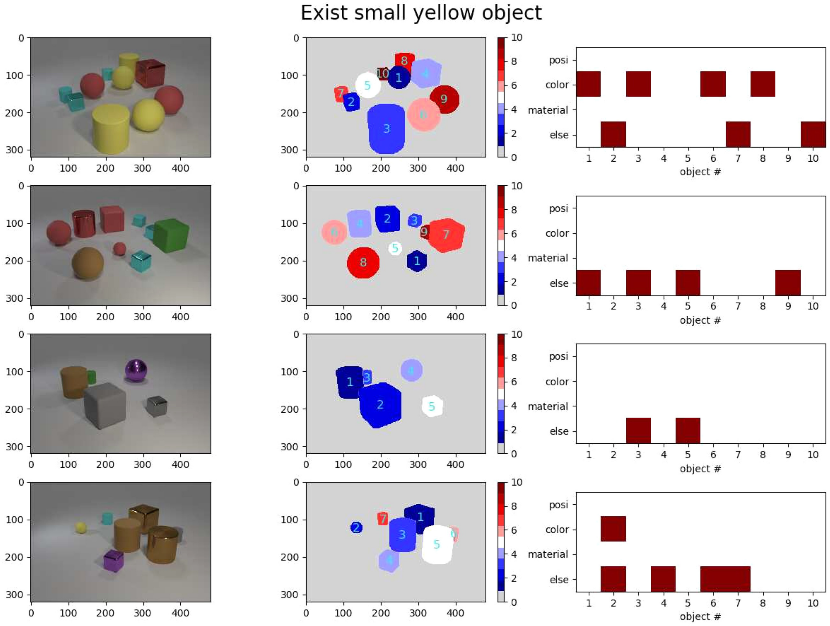


Figure 6: Single-example interpretation of the task "choose the small yellow object." The left column shows input pictures, and the middle column shows masks colored according to object IDs. We overlaid the masks with the object IDs for visual aid. The right column shows the binary activity summarizing the information at layer $Y_{(i \cap MC)}$. The x-axis corresponds to object ID, and the y-axis represents four kinds of representations: position $Y_{(\mathrm{posi} \cap \mathrm{MC})}$, color $Y_{(\mathrm{color} \cap \mathrm{MC})}$, material $Y_{(\mathrm{material} \cap \mathrm{MC})}$, and else $Y_{(\mathrm{else} \cap \mathrm{MC})}$, where the dimension with highest mutual information is plotted. The red square represents the lower frequency binary representation, and the white space represents the counterpart.

information with the label "cube" is 0.432 Nat per object. For the case "choosing cylinder on the right side," mutual information with the label "cylinder" is 0.408 Nat per object. All of these numbers exceed the chance level (the 99, 95, and 90 percentile by chance are 0.637, 0.495, and 0.368 Nat, respectively, for balanced binary random variables like size, and 0.583, 0.449, 0.332 Nat for cases with three alternatives like shape).

5.2.3 Visualizing the Unconceptualized Representation. After getting the unconceptualized representation useful for the new task, we can continue the framework by splitting that representation into the learned useful part and its complement. Separating this new useful representation is nontrivial because labels of the MC task jointly depend on multiple image properties. While previous methods (Koh et al., 2020; Chen et al., 2020) need feature-specific labels to learn a new property, the proposed framework

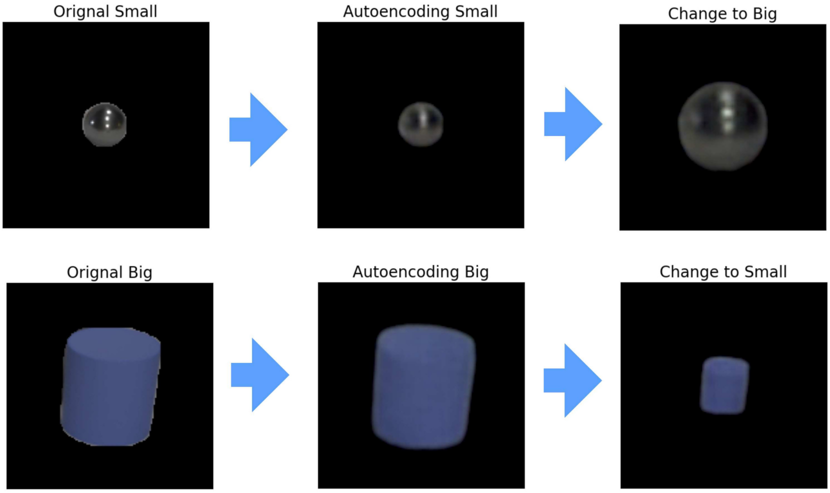


Figure 7: Visualizing the newly learned Y_{MC} about size after learning the task “choose the picture with a small yellow object.” As can be seen from the result, changing Y_{MC} of a small object renders a big counterpart of the same object, and changing Y_{MC} of a big object renders a small counterpart of the same object.

automatically segregates a new, useful representation from previously learned representations. Furthermore, the proposed system can visualize what new representation has just been learned.

Here, we demonstrate the result after learning the task “choose the picture with a small yellow object. We have mentioned that after learning this new task, the model is expected to learn a new concept about size as the new representation $Y_{\mathrm{MC}} = Y_{(\mathrm{else} \cap \mathrm{MC})}$. Note, again, that we never provided the model labels specifically about size. Then we can continue the framework by performing another round of autoencoding, which splits Y_{else} into Y_{MC} and $Y_{\mathrm{else}} \backslash Y_{\mathrm{MC}}$. After that, the model explains what property is newly learned by generating the image of an object and changing its size as the newly latent representation Y_{MC} is altered (see Figure 7). This visualization also helps humans interpret the operation of the model.

Information about other studies on the CLEVR data set can be found in appendix sections 4 to 8. We also offer more discussion about our method in appendix section 9 and discuss limitations of our method in appendix section 10. The source code of this project can be found at https://github.com/hezq06/progressive_interpretation.

6 Conclusion

This letter proposes a progressive framework based on information theory to synthesize interpretation. We show that interpretation involves independence, is progressive, and can be given at a macroscopic level using meta-information. Changing the receiver of the interpretation from a human to a target model helps define interpretation clearly. Our interpretation framework divides the input representations into independent partitions by tasks and synthesizes interpretation for the next task. This framework can also visualize what conceptualized and unconceptualized partitions code by generating images. The framework is implemented with a VIB technique and is tested on the MNIST and the CLEVR data sets. The framework can solve the task and synthesize nontrivial interpretation in the form of meta-information. The framework is also able to progressively form meaningful new representation partitions. Our information-theoretic framework capable of forming quantifiable interpretations is expected to inspire future understanding-driven deep learning.

Acknowledgments

We thank Ho Ka Chan, Yuri Kinoshita, and Qian-Yuan Tang for useful discussions about the work. This study was supported by Brain/MINDS from the Japan Agency for Medical Research and Development (AMED) under grant JP15dm0207001, Japan Society for the Promotion of Science (JSPS) under KAKENHI grant JP18H05432, and the RIKEN Center for Brain Science.

References

Alemi, A. A., Fischer, I., Dillon, J. V., & Murphy, K. (2016). *Deep variational information bottleneck*. arXiv:1612.00410.

Andreas, J., Rohrbach, M., Darrell, T., & Klein, D. (2016). Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 39–48).

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., . . . Herrera, F. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82–115. 10.1016/j.inffus.2019.12.012

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS One*, *10*(7), e0130140.

Bada, S. O., & Olusegun, S. (2015). Constructivism learning theory: A paradigm for teaching and learning. *Journal of Research and Method in Education*, *5*(6), 66–70.

Bang, S., Xie, P., Lee, H., Wu, W., & Xing, E. (2019). *Explaining a black-box using a deep variational information bottleneck approach*. arXiv:1902.06918.

Bell, A. J., & Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, *7*(6), 1129–1159. 10.1162/neco.1995.7.6.1129

Chalk, M., Marre, O., & Tkacik, G. (2016). Relevant sparse codes with variational information bottleneck. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in neural information processing systems*, *29*. Curran.

Chen, J., Song, L., Wainwright, M. J., & Jordan, M. I. (2018). *Learning to explain: An information-theoretic perspective on model interpretation*. arXiv:1802.07814.

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. (2016). InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in neural information processing systems, 29* (pp. 2180–2188). Curran.

Chen, Z., Bei, Y., & Rudin, C. (2020). Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, *2*(12), 772–782. 10.1038/s42256-020-00265-z

Doshi-Velez, F., & Kim, B. (2017). *Towards a rigorous science of interpretable machine learning*. arXiv:1702.08608.

Freitas, A. A. (2014). Comprehensible classification models: A position paper. *ACM SIGKDD Explorations Newsletter*, *15*(1), 1–10. 10.1145/2594473.2594475

Glucksberg, S., & McCloskey, M. (1981). Decisions about ignorance: Knowing that you don't know. *Journal of Experimental Psychology: Human Learning and Memory*, *7*(5), 311. 10.1037/0278-7393.7.5.311

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). *Explaining and harnessing adversarial examples*. arXiv:1412.6572.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., . . . Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97. 10.1109/MSP.2012.2205597

Hou, B.-J., & Zhou, Z.-H. (2018). *Learning with interpretable structure from RNN*. arXiv:1810.10708.

Hudson, D. A., & Manning, C. D. (2018). *Compositional attention networks for machine reasoning*. arXiv:1803.03067.

Hyvärinen, A., & Oja, E. (2000). Independent component analysis: Algorithms and applications. *Neural Networks*, *13*(4–5), 411–430.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., & Girshick, R. (2017). CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2901–2910).

Kim, S. W., Tapaswi, M., & Fidler, S. (2018). *Visual reasoning by progressive module networks*. arXiv:1806.02453.

Kingma, D. P., & Welling, M. (2013). *Auto-encoding variational Bayes*. arXiv:1312.6114.

Koh, P. W., Nguyen, T., Tang, Y. S., Mussmann, S., Pierson, E., Kim, B., & Liang, P. (2020). Concept bottleneck models. In *Proceedings of the International Conference on Machine Learning* (pp. 5338–5348).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. 10.1145/3065386

Lechner, M., Hasani, R., Amini, A., Henzinger, T. A., Rus, D., & Grosu, R. (2020). Neural circuit policies enabling auditable autonomy. *Nature Machine Intelligence*, *2*(10), 642–652. 10.1038/s42256-020-00237-3

Li, X. L., & Eisner, J. (2019). *Specializing word embeddings (for parsing) by information bottleneck*. arXiv:1910.00163.

Lipton, Z. C. (2018). The mythos of model interpretability. *Queue*, *16*(3), 31–57. 10.1145/3236386.3241340

Mahinpei, A., Clark, J., Lage, I., Doshi-Velez, F., & Pan, W. (2021). *Promises and pitfalls of black-box concept learning models*. arXiv:2106.13314.

Margeloiu, A., Ashman, M., Bhatt, U., Chen, Y., Jamnik, M., & Weller, A. (2021). *Do concept bottleneck models learn as intended?* arXiv:2105.04289.

National Research Council. (2002). *Learning and understanding: Improving advanced study of mathematics and science in US high schools*. National Academies Press.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, *113*, 54–71. 10.1016/j.neunet.2019.01.012

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., . . . Hadsell, R. (2016). *Progressive neural networks*. arXiv:1606.04671.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., . . . Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489. 10.1038/nature16961

Sundararajan, M., Taly, A., & Yan, Q. (2017). *Axiomatic attribution for deep networks*. arXiv:1703.01365.

Tishby, N., Pereira, F. C., & Bialek, W. (2000). *The information bottleneck method*. arXiv:0004057.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, . . . Polosukhin, I. (2017). Attention is all you need. In I. Guyon, Y. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems*, *30* (pp. 5998–6008). Curran.

Wang, Q., Boudreau, C., Luo, Q., Tan, P.-N., & Zhou, J. (2019). Deep multi-view information bottleneck. In *Proceedings of the 2019 SIAM International Conference on Data Mining* (pp. 37–45).

Wibral, M., Priesemann, V., Kay, J. W., Lizier, J. T., & Phillips, W. A. (2017). Partial information decomposition as a unified approach to the specification of neural goal functions. *Brain and Cognition*, *112*, 25–38. 10.1016/j.bandc.2015.09.004

Wu, M., Hughes, M. C., Parbhoo, S., Zazzi, M., Roth, V., & Doshi-Velez, F. (2017). *Beyond sparsity: Tree regularization of deep models for interpretability*. arXiv:1711.06178.

Zhang, Q., Cao, R., Shi, F., Wu, Y. N., & Zhu, S.-C. (2017). *Interpreting CNN knowledge via an explanatory graph*. arXiv:1708.01785.

Received May 24, 2022; accepted August 10, 2022.