CONTINUAL LEARNING FOR UNSUPERVISED CONCEPT BOTTLENECK DISCOVERY

Luca Salvatore Lorello
Department of Computer Science
University of Pisa
University of Modena and Reggio Emilia
Italy
luca.lorello@phd.unipi.it

Marco Lippi
Department of Information Engineering
University of Florence
Italy
marco.lippi@unifi.it

Stefano Melacci
Department of Information Engineering
University of Siena
Italy
stefano.melacci@unisi.it

ABSTRACT

In the context of continual learning, little attention is dedicated to the problem of developing a layer of "concepts", also known as "concept bottleneck", to support the discrimination of higher-level task information, especially when concepts are not supervised. Concept bottleneck discovery in an unsupervised setting is thus largely unexplored, and this paper aims to move a step forward in such direction. We consider a neural network that faces a stream of binary tasks, with no further information on the relationships among them, i.e., no supervisions at the level of concepts. The learning of the concept bottleneck layer is driven by means of a triplet-based criterion, which is instantiated in conjunction with a specifically designed experience replay (concept replay). Such a novel criterion exploits fuzzy Hamming distances to treat vectors of concept probabilities as fuzzy bitstrings, encouraging different concept activations across different tasks, while also adding a regularization effect which pushes probabilities towards crisp values. Despite the lack of concept supervisions, we found that continually learning the streamed tasks in a progressive manner yields the development of inner concepts that are significantly better correlated with the higher-level tasks, compared to the case of joint-offline learning. This result is showcased in an extended experimental activity involving different architectures and newly created (and shared) datasets that are also well-suited to support further investigation of continual learning in concept-based models.

1 INTRODUCTION

In the last few years, a growing attention was dedicated to the design of machines which are capable of learning in a continual manner, progressively acquiring new skills and adapting to the changes of the learning environment (van de Ven et al., 2022). The most significant part of existing work is focused on the case of classification problems (several incremental settings), usually supervised (van de Ven et al., 2022; Wang et al., 2024), and on the case of reinforcement learning (Khetarpal et al., 2022). Little attention is dedicated to the development of hierarchical representations of categories (Abdelsalam et al., 2021; Lin et al., 2022), or, as more common in the context of Neuro-Symbolic Learning (Lorello & Lippi, 2023), to the development of "concept-based" models (Ciravegna et al., 2023), such as "concept-bottleneck" models (Koh et al., 2020). In this case, the network is not only required to output a task-level prediction, but also the set of "concepts" that are active or not active for the currently provided input. The activation level of these concepts is an explicit input of the task-level predictor: hence, concept-based models not only offer a form of interpretability, but also allow for human-based manipulation of the active/inactive concepts, with effects on the task-level prediction (Espinosa Zarlenga et al., 2022; Ghosh & Kandasamy, 2020; Novelli et al., 2023). Discovering concepts over time, possibly in a lifelong manner, is a natural way of learning in humans, and especially in children (Damon & Lerner, 2008), where "supervisions" are limited.

Motivated by these considerations, the focus of this work lies at the intersection between continual learning and concept-bottleneck models. In particular, we focus on the case in which the network is asked to automatically develop

a finite set of concepts, only having the use of task-level supervision, making the problem fully unsupervised at concept-level. This setting is supported by the evidence that incrementally disclosing concepts over time without explicit supervisions is not only aligned with what happens in humans (Ziori & Dienes, 2012), but it can also provide precious information for machine learning purposes, capable of guiding discovery along a sequence of smaller, but feasible, steps, aimed at building a vocabulary of concepts relevant for a specific domain. This opportunity is traded-off by the additional challenge of preventing catastrophic forgetting, not only at the task level, but also at the concept level (Marconato et al., 2023), while also preserving the plasticity required to update wrongly identified concepts. Such a challenging, yet important, scenario is largely overlooked in the literature, which motivated us to develop and release a novel and easy to control benchmark, on which the experiments of this paper are based.

The vast majority of approaches to concept learning, in fact, strongly rely on supervisions, that must be specifically provided not only at the level of tasks, but also at the level of concepts. As a notable example, Concept Bottleneck Models (CBMs) (Koh et al., 2020) have been recently introduced as an interpretable model that learns from the input an intermediate layer of concepts, upon which classification is performed. While Boolean CBMs exploit a hard activation function (i.e., a threshold) to enable concept triggering, Fuzzy CBMs use a soft activation (e.g., a sigmoid) to enhance the expressive power of the model, even though at the cost of a lower interpretability. The need of full task-level and concept-level supervisions has been partially relaxed in Hybrid CBMs (Mahinpei et al., 2021), based on a combination of both supervised and unsupervised concepts to give the model some additional degrees of freedom. As an expansion of CBMs, Concept Embeddings Models (CEMs) (Espinosa Zarlenga et al., 2022) extend the representation of each concept with two embeddings that are associated to the active and inactive state of the concept, respectively. Some attempts to perform unsupervised learning of disentangled representations, that can be associated to low-level visual concepts, have been recently inspired by neuroscience (Higgins et al., 2016), using a generative model implemented via variational auto-encoders, whose learning is driven by principles that mimic the ventral visual stream in our brain. On a different perspective, a first attempt of exploiting semi-supervised concept learning in a continual learning setting has been proposed by Marconato et al. (2023), who focus on a neuro-symbolic scenario where domain background knowledge is available, and it is exploited to consolidate concept learning with supervisions across a sequence of tasks. The problem is addressed by modeling the conditional probability distribution of concepts with respect to the input data, and by enforcing the stability of such probability distribution through time. The approach is coupled with a neuro-symbolic system, where background knowledge is exploited to drive learning and avoid reasoning shortcuts.

In this paper we focus on a *continual learning setting where neither domain knowledge is given, nor concept supervision is provided* during training. To address issues typical of continual learning, we propose a novel approach to experience replay (Rolnick et al., 2019), where, differently form existing approaches (Buzzega et al., 2020), the memory buffer also stores *self-developed* binary concept representations, driving unsupervised concept learning with a triplet loss on current data and past samples from the buffer. We argue that such a limited-knowledge scenario could be particularly relevant in many application domains, such as healthcare (Eckhardt et al., 2023) or cyber-security (Alom & Taha, 2017), where collecting concept annotations would be virtually impossible, due to the inhability of defining exhaustive sets of task-relevant concepts a priori. As we are particularly interested in the association between learned representations and their symbolic interpretation, we take inspiration from Deep Hashing (Luo et al., 2023) to interpret concept vectors as binary hashes, whose Hamming similarity resembles semantic concept similarity. Our experiments show that continually disclosing concepts, even if in unsupervised manner, is beneficial for task alignment, and that the network is not forgetting those previously developed concepts that were important for past tasks. Moreover, thanks to the proposed concept-based experience replay, the network adapts a small set of concepts in order to solve the current task at hand, promoting isolation. Of course, this comes at the price of a performance drop in terms of accuracy, that, however, is overall negligible in the considered benchmark, strongly motivating further research in this direction.

2 Background

We consider a learning problem where a neural network has to develop the capability of predicting a set of concepts, possibly organized into a hierarchy, while predictions on the final high-level tasks are built on the basis of the activated concepts. For example, consider the problem of predicting whether an input pattern belongs to category "cat", which an agent is required to solve as a function of its concept-level predictions. The agent can take a positive decision, not only because of a fully black-box process, but because it also predicts the concepts "fur", "tail", "paws", and the absence of "engine" or "wheels". This setting improves explainability and control (concepts could also be artificially manipulated, if needed), and it bridges the symbolic world of AI (Hitzler & Sarker, 2022; Ciravegna et al., 2023). Formally, let $\mathcal{C}$ be the set of concepts and $\mathcal{T}$ the set of higher-level tasks. Predictions on each task depend on what concepts in $\mathcal{C}$ are (or are not) activated. Given an input pattern $\boldsymbol{x} \in \mathbb{R}^d$, $d \geq 1$, we indicate with $\boldsymbol{c} \in [0,1]^{|\mathcal{C}|}$ the predicted activation scores of the concepts, being c_j the j-th concept activation (1 means active, 0 inactive). Vector $\hat{\boldsymbol{c}}$ is

the Boolean counterpart of $\boldsymbol{c}$, obtained by polarizing the elements in $\boldsymbol{c}$ by means of a threshold τ.[1] We consider binary tasks, where a net yields the output score $o_i \in [0,1]$ to predict the label of $\boldsymbol{x}$ in the i-th task. As a result, an input sample can be classified as positive or negative with respect to each task in $\mathcal{T}$. In the following we briefly summarize the main notions behind CBMs and CEMs, introducing the notation used throughout the paper.[2]

Concept Bottleneck Models. CBMs (Koh et al., 2020) popularized the idea of an intermediate layer of human-specified concepts, referred to as concept bottleneck, and used to predict the final output. The set $\mathcal{C}$ is known in advance, thus the meaning of each element in $\mathcal{C}$ is pre-defined. Training examples are composed of task-level and concept-level supervisions, i.e., for each $\boldsymbol{x}$ in the training set, the task label and the set of active/not-active concepts are fully provided. The concept bottleneck allows intervention on the values of the predicted concepts within the model. The neural network is responsible of learning a composite function $f \circ g \circ \psi$, where $\psi\colon \mathbb{R}^d \mapsto \mathbb{R}^{d'}$ computes an initial representation of size d' of the input (for example, it could be a pre-trained backbone), $g\colon \mathbb{R}^{d'} \mapsto [0,1]^{|\mathcal{C}|}$ is the function that predicts concepts, while $f\colon [0,1]^{|\mathcal{C}|} \mapsto [0,1]^{|\mathcal{T}|}$ yields the final task labels. Defining $\boldsymbol{o}$ as the vector collecting each predicted target o_i associated to the i-th task, we have:

$$\boldsymbol{x}' = \psi(\boldsymbol{x}), \quad \boldsymbol{c} = g(\boldsymbol{x}'), \quad \boldsymbol{o} = f(\boldsymbol{c}). \tag{1}$$

Concept Embedding Models. CEMs (Espinosa Zarlenga et al., 2022) generalize and extend CBMs, going beyond the limited expressiveness of a concept-score-only representation. In CEMs, each concept is associated both with its own score c_j, and two learnable embeddings, that are mixed by a weighted average controlled by c_j. In detail, two functions ϕ_j^+ and ϕ_j^- are responsible of computing the aforementioned embeddings for the j-th concept, $\phi_j^{\cdot}\colon \mathbb{R}^{d'} \mapsto \mathbb{R}^e$, where $e \geq 1$ is the size of each embedding. CEMs exploit the following g (Eq. 1) when predicting concept activation scores,

$$\boldsymbol{c} = g(\boldsymbol{x}') := [s(\phi_1^+(\boldsymbol{x}'), \phi_1^-(\boldsymbol{x}')), \ldots, s(\phi_{|\mathcal{C}|}^+(\boldsymbol{x}'), \phi_{|\mathcal{C}|}^-(\boldsymbol{x}'))], \tag{2}$$

where $s\colon \mathbb{R}^e \times \mathbb{R}^e \mapsto [0,1]$ is a learnable and differentiable scoring function (single layer network with sigmoidal unit), and the squared brackets are used here to collect the components of the vectorial function g. Before predicting the task labels, each pair of concept embeddings $(\phi_j^+(\boldsymbol{x}'), \phi_j^-(\boldsymbol{x}'))$ is averaged in function of the concept activation score c_j, yielding $|\mathcal{C}|$ novel representations, concatenated into vector $\tilde{\boldsymbol{c}}$. Finally, the task-level predictions are computed. However, differently from the case of Eq. 1, concepts are not directly represented by their activation scores $\boldsymbol{c}$, but by the distributed $\tilde{\boldsymbol{c}}$. We redefine[3] f of Eq. 1 as $f\colon \mathbb{R}^{|\mathcal{C}|e} \mapsto [0,1]^{|\mathcal{T}|}$ and we get

$$\boldsymbol{o} = f(\tilde{\boldsymbol{c}}). \tag{3}$$

This architecture allows the task-predictor f to have access not only (i) to the information on how strongly concepts were activated, but also (ii) to distributed representations of the concepts themselves, since both such types of information were fused into $\tilde{\boldsymbol{c}}$. We will use notation f_i to indicate the component of f associated with the i-task predictor. In this paper we focus on CEMs, which represent state-of-the art concept bottleneck models. It is important to remark that most works on CEMs assume access to supervisions at a concept level, differently from the setting of this paper.

3 Methodology

We focus on a teacher-student scenario where the teacher presents the student with samples of a sequence of $|\mathcal{T}|$ binary tasks, considered appropriate to improve the student understanding skills. The student is expected to get better at discriminating examples of the selected tasks and also to progressively acquire an improved representation of the world, developing, reusing and mixing latent concepts to solve each task. These concepts are not made explicit, thus no supervision is given about them. Data is provided to the learning agent (student) through a potentially lifelong stream where, at each time instant t, a small batch of samples $X^{(t)}$ (from a certain task) is made available, together with their task ID and their task-level labels, both collected into $Y^{(t)}$. The agent is aware of task switches, thus we are in a task incremental setting (van de Ven et al., 2022). The issues with this setting are not only those typical of continual learning, such as catastrophic forgetting (Parisi et al., 2019) at a task-level, but also internal stability issues that might lead to concept drift (Marconato et al., 2023). As a matter of fact, even if the set of concepts $\mathcal{C}$ is not defined in advance (and, in general, it is not known at all), the agent must avoid forgetting previously developed concepts. The agent is expected to learn how to compose the already stable concepts with newly discovered ones.

We propose to tackle this scenario introducing specific criteria to constrain the network in order to (i) favour the development of strongly polarized concept activations, (ii) increase the correlation between tuples of concepts and tasks, (iii)

[1] When Booleanization is needed, we consider $\tau = 0.5$, and set to 1 those concept activations that are $\geq \tau$.

[2] The notation was adapted to more easily fit the context of this work, thus it slightly differs from that of the original papers.

[3] We avoid introducing a new symbol here, to simplify notation and help the reader in relating CBMs and CEMs.

progressively promote stability over time. Requirement (i) is fundamental in concept-based learning, since it makes it easy to take a crisp decision on the state of each concept j given its fuzzy activation c_j, promoting interpretability. Requirement (ii) suggests that there must be a portion of active/not-active concepts related to final tasks, in order to ensure that the agent is profitably learning to organize its internal knowledge. Requirement (iii) is typical of continual learning, to discourage catastrophic forgetting, in this case also with respect to concepts, which are unsupervised and not completely learned during the first tasks, but that are expected to develop during the life of the agent.

Hamming Triplet Loss. Due the lack of extra knowledge on the problem and on the ground truth concepts distribution, we propose an interpretation of concepts based on a perspective that intersects notions behind Deep Hashing (Luo et al., 2023): crisp concept vectors $\hat{\boldsymbol{c}}$ can be considered binary hashes, which partition the input space in a way which promotes collisions between semantically similar (i.e., associated to the same task label) samples. This interpretation allows us to define a straightforward distance metric learning approach based on the Hamming distance, in the form of a continuous and differentiable relaxation exploiting the L^1 norm $\|\cdot\|_1$ applied to pairs of concept vectors,

$$\text{soft\_hamm}(\boldsymbol{c}_h, \boldsymbol{c}_k) = \|\boldsymbol{c}_h - \boldsymbol{c}_k\|_1. \tag{4}$$

The metric learning problem is handled by comparing triples of input patterns, as typical in the popular triplet loss (Dong & Shen, 2018). For each task in $\mathcal{T}$, a triple is composed of the concept activation vectors ($\boldsymbol{c}$) (1) of an *anchor* sample, (2) of a sample with the same task-level label of the anchor (i.e., task-positive or task-negative, named *positive* element of the triple), and (3) a third example with different task-level label (i.e., task-negative or task-positive, respectively – named *negative* element). Minimizing the triplet loss favours the development of similar concept vectors for examples with the same label, and different vectors for pairs with different labels. It is important to remark that, within each task, examples with the same task-level label are expected to share some concepts (the ones that are relevant for the considered task), while there is no need to force coherence on the remaining ones. For this reason, the criterion we propose must be interpreted as a soft regularizer and not as a strongly enforced constraint. We build a set of triples performing semi-hard negative mining (Schroff et al., 2015). Formally, let $\xi_{\mathcal{SH}}(X^{(t)}, Y^{(t)})$ be the mining function that returns $n^{(t)}$ triples out of a mini-batch of data, in the form of the set $\{A^{(t)}, P^{(t)}, N^{(t)}\}$, composed of the (ordered) collections of concept activation vectors of the three components of the mined triples, i.e., (1) anchors, (2) positives, and (3) negatives, respectively. Plugging the (relaxed) Hamming distance of Eq. 4 in a triplet loss with margin m, we get our proposed Hamming triplet loss $\mathcal{L}_H$ (dropping the time index to simplify notation),[4]

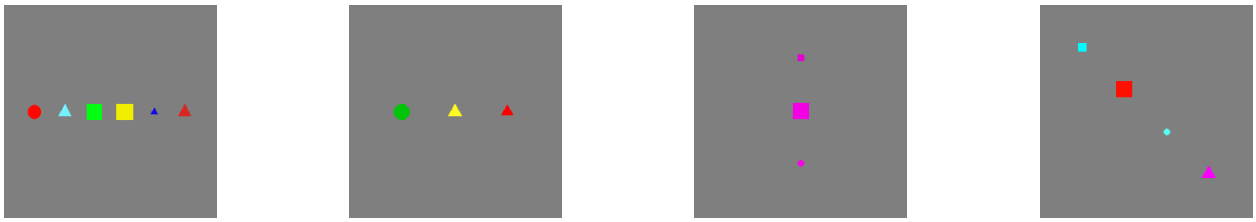
$$\mathcal{L}_H\left(\{A, P, N\}, m\right) = \sum_{i=0}^{n-1} \max\left(\text{soft\_hamm}\left(A_i, P_i\right) - \text{soft\_hamm}\left(A_i, N_i\right) + m, 0\right). \tag{5}$$

being A_i, P_i, N_t the i-th elements of the respective collections. The margin hyper-parameter m assumes an intuitive interpretation: the expected minimum "concept-distance" between positive and negative samples.

Concept Replay. Enforcing $\mathcal{L}_H$ is not enough to guarantee stability on the learned concepts, that might incur in non-predictable dynamics over time. For this reason, we developed a novel experience replay strategy, based on an augmented replay buffer which is used both for rehearsal at the level of task objective (i.e., to replay positive and negative examples for each task, as usual in continual learning) and as pseudo-rehearsal at the concept level (newly proposed). The replay buffer uses class-balanced reservoir sampling (Chrysakis & Moens, 2020) and it keeps, for each input sample, both the given task-level label and the binarized concept representation $\hat{\boldsymbol{c}}$. We indicate with B_X, B_Y, and B_C the buffered data, that is composed of the buffered input samples (B_X), their task-ID and task-level labels (B_Y), and the *binarized* concept representations (B_C). Concept replay is implemented as a second round of Hamming triplet loss, exploiting the buffered concepts. Since the buffer contains data from (a subset of) the tasks seen so far, we mine a batch of triples for each of them (uniform sampling strategy). We indicate with $\xi_{\mathcal{U}}(B_X, B_Y, B_C)$ the triplet mining function, which is different from the already introduced $\xi_{\mathcal{SH}}(X, Y)$ for the following reasons. The anchor samples, taken from B_X, are fed to the net, computing fresh concept representations $\boldsymbol{c}$'s, while the concept vectors for the positive and negative samples of the triples are directly retrieved from B_C, thus they are binary vectors.[5] Such binary vectors act as constant and polarized targets. Hence, when minimizing the triplet loss, only the concept activations of the anchors are altered, pushing them toward (resp. far away from) the constant binary targets of the positive (resp. negative) elements. As a result, buffered binary concepts implicitly help polarize the learned concept probabilities towards 0 or 1, and they are more storage efficient, which is a crucial property to limit the memory used by the buffered samples. In order to avoid aging of the binary concepts in B_C, at the end of each epoch B_C is recomputed from scratch.

[4] If no triples can be mined within the batch, the loss assumes a default value of 0.

[5] Since data from the current task is directly provided to the network (i.e., it comes from the input stream), the anchors of the triples for the current task are sampled from $X^{(t)}$. We avoid making this explicit, to keep the notation simple.



(a) (b) (c) (d)

Figure 1: Task examples. Left: positive (a) and negative (b) examples for task 12 in KANDY-1, positive class being defined by a red triangle on the right, plus the presence of a blue object anywhere. Right: positive (c) and negative (d) examples for task 24 in KANDY-2, positive class being defined by objects that share one attribute (in this case, color).

Aggregated Loss Function. The total loss function $\mathcal{L}$ is the sum of three contributions,

$$\begin{aligned}\mathcal{L}(X, Y, B_X, B_Y, B_C) &= \mathcal{L}_{\text{BCE}}(X, Y) + \lambda_r \cdot \mathcal{L}_{\text{BCE}}(B_X, B_Y) \\ &\quad + \frac{\lambda_h}{2} \cdot \left[\mathcal{L}_H\left(\xi_{\mathcal{SH}}(X, Y), m\right) + \mathcal{L}_H\left(\xi_{\mathcal{U}}(B_X, B_Y, B_C), m\right)\right]. \end{aligned} \tag{6}$$

The task-level loss is a binary cross-entropy ($\mathcal{L}_{\text{BCE}}$) applied to the batch of input data (X, Y). Task-level experience replay is based on the same loss, even if applied to (B_X, B_Y), weighted by $\lambda_r > 0$. Finally, the contrastive term, weighted by $\lambda_h > 0$, is the average of the Hamming triplet loss (Eq. 5) applied to input and buffered samples.

Use Case. We hereby illustrate an application of our methodology with a use case, which will be further extended in Section 4 when describing the datasets we created and shared. Kandinsky Patterns (Holzinger et al., 2021), are images containing simple geometric shapes arranged according to a mathematically definable rule. Although perceptually simple, discrimination between two Kandinsky Patterns requires complex reasoning capabilities, which makes them challenging for neural networks. Images with such patterns can be easily generated from basic primitives, and organized into a curriculum-like sequence of binary classification tasks, each of which either introduces new concepts, or reuses those previously introduced. In particular, curricula can be designed following these key principles: (i) encourage concept discovery by *progressive disclosure*; (ii) focus on low-level *perceptually unambiguous concepts*; (iii) *evaluate concept retention* by means of both short-term and long-term concept reuse, (iv.) allow evaluation of models trained under a *limited data regime*. Figure 1 shows examples of positive and negative samples we generated, taken from two different tasks, considering shapes of different sizes and colors as ground concepts. Due to their controlled generation, positive and negative samples can be unambiguously defined by first-order logic, they are guaranteed to be noise-free, and rejection-sampling can be performed to maximize diversity (see Appendix A). A sequence of Kandinsky Pattern-based tasks represent an ideal use-case to frame the ideas of this paper.

4 Experiments

We performed experiments in the use-case of the previous section, which allows us to make reasonable hypothesis on the expected concepts. In fact, selecting an existing concept-supervised dataset and removing the supervisions would make it hard to define expectations, since the large variability of the data virtually opens to several different valid solutions for the network, harder to bound. Moreover, there is an intrinsic lack of datasets for concept-based continual learning, with a few exceptions in which only a small number of tasks is considered (Marconato et al., 2023), making them less suited for the investigation of this paper.

4.1 Datasets

We created and shared (attached to submission) two datasets with 20 and 26 perceptive tasks, as previously introduced, referred to as KANDY-1 and KANDY-2, by means of controlled generation procedures. Samples include significant variability, and tasks are designed with different levels of complexity. Different tasks might share some concepts (in general, more complex tasks include also basic concepts from simple tasks), and we annotated each image with immediately perceivable concepts. This information is not provided to the network, thus there are no strict guarantees that the learned concepts will be aligned with such primitives; however, it allows us to have a reference set of expectations.

KANDY-1. The first dataset consists of 20 tasks, each composed of a stream of 100 images presenting elementary concepts (shapes with different colors, and compositions thereof). Tasks are generated so as to model simple problems,

with a maximum of six objects in each image, three at most being of interest to solve the task. Data is split into training, validation and test sets (50%-25%-25%). For evaluation purposes, each sample is annotated with 11 existential ground truth concepts (e.g., "there exists a triangle in the image", regardless of the actual count), covering 3 different shapes, 6 colors and 2 sizes. No concept annotations is available to the agent during training. These concepts are perceptually inferrable without any semantic knowledge of the tasks. Advanced concepts could also be defined; however, as they would require complex reasoning, their evaluation would favor strongly-symbolic models, regardless of their incremental learning skills. Table 3 in Appendix A describes concept interactions and reuse in KANDY-1 tasks.

KANDY-2. The second dataset is composed of 26 tasks, with 200 images each (80% train, 10% validation, 10% test splits). Samples contain a collection of one to five objects displaced along a line, all of which are potentially relevant for the task label. Samples in KANDY-2 are also annotated with existential concepts; however, as the domain is more complex, additional global features are annotated, for a total of 17 ground truth concepts: 3 for rough counts (one object, few, and many objects), 3 for object displacement (diagonal, horizontal, vertical), and 11 existential concepts (same as KANDY-1). Table 4 in Appendix A highlights concept interactions and reuse in KANDY-2.

In both datasets, samples are randomly altered by injecting Gaussian noise to the HSV representation of each color ($\mu = 0, \sigma_H = 0.01, \sigma_S = 0.2, \sigma_V = 0.2$) and uniform noise in the sizes of the shapes (± 2 pixels). These values were selected empirically to preserve concept semantics (e.g., a "noisy red" object is still perceived as "red" by a human).

4.2 Experimental setup

Learning Settings. We focus on task-incremental learning, where tasks in $\mathcal{T}$ are sequentially presented to the network (10 epochs per task). Our goal is to inspect the behaviour of different neural architectures, to validate the role of the proposed learning criteria. Each output neuron is associated to a task, thus the supervising signal is only applied to the output component associated to the task-ID of the input sample. We compared this setting with offline (joint) learning, where data of all tasks are collected in a single dataset and shuffled at the beginning of each epoch (10). As no replay buffers are used in this case, the triplet loss is only applied to elements of the current batch. The task-ID is also exploited in the joint case, to be fully coherent to the incremental learning setting.

Networks. We considered three instances of CEMs, which differ by their backbone ψ: CNN, ResNet-50, and ViT-16. The ψ part of CNN consists of a ReLU-based convolutional net trained from scratch (thus covering the whole $f \circ g \circ \psi$), with a stack of 3 conv. and max pooling layers (64 (5×5), 128 (3×3), 256 (3×3) features (kernel sizes), respectively, with stride 2 in the last two blocks), where the last pooling returns outputs at the resolution of 5×5, followed by a dense layer (4096 neurons, dropout). Then a CEM head is added (i.e, g and f). ResNet-50 is a pre-trained ResNet (ImageNet), with a learnable CEM head (i.e., g and f). ViT-16 is pre-trained Visual Transformer (Dosovitskiy et al., 2021), ViT-Base/16 (ImageNet), where only the CEM head is re-trained (g and f). Each CEM head consists of $|\mathcal{C}| = 30$ concepts (larger than the number of ground truth concepts), with sigmoid activation. The internal concept embeddings, of size $e = 12$, are computed by linear layers ($\psi^{\cdot}$). Weights in the linear projection layer are shared across every concept, as in the original implementation of Espinosa Zarlenga et al. (2022). The output of the concept embedding stage, $\tilde{\boldsymbol{c}}$, is fed into a linear model (i.e., function f) consisting of an output neuron for each task. Concept logits before the sigmoid activation will be used to compute concept-only metrics.

Setup. Our experimental activity is aimed at validating the role of the proposed learning criteria. For this reason, we evaluated different values of λ_r, λ_t, m, accordingly to the following grids, which were selected after an initial set of preliminary experiments: $\lambda_r \in \{0, 1, 10\}$, $\lambda_t \in \{0, 1, 10\}$, $m = \{1, 4\}$. We also considered a replay buffer of size 200,[6] Adam optimizer with learning rate set to 0.001, batch size $|X^{(t)}| = 32$. During training we performed the following image augmentation operations: random rotation, random resize crop, normalization. Although datasets have a negligible imbalance between task-positive and task-negative samples, we ensured batches to be balanced. Similarly, when storing new elements in the replay buffer, we used a balanced reservoir sampling (Kim et al., 2020).

Validation. In joint learning, the optimal values of the hyperparameters were found by maximizing the accuracy of the validation set. In continual learning, we followed the more challenging and realistic setting in which only some initial tasks are used for validation purposes, considering that the learning horizon is virtually lifelong. This can be perfectly simulated in the proposed datasets, which present basic concepts first, and then, in the later tasks, more advanced problems. This allowed us to unambiguously identify the last task in which a new basic/ground truth concept is introduced (task 8 in KANDY 1 and the end of task 17 in KANDY 2 – see also Appendix A), and the subsequent

[6] This amount (corresponding to 9% and 4% of the total number of samples for each dataset, respectively) is small enough not to bias evaluation in favor of the continual learning setting, compared to the joint one.

task are excluded from the validation procedure.[7] This choice makes the continual learning experience even more challenging when compared to the joint case, where validation data from all the tasks is available. All the results of this paper are about the test sets, using the best model found with the described validation criterion.

Task-related Metrics. The main metric we consider is the AVERAGE ACCURACY in $[0, 1]$, which is the average of the task-related accuracies $\{\text{acc}_j^z,\ j = 1, \ldots, |\mathcal{T}|\}$. In particular, acc_j^z is the micro-accuracy on the j-th binary classification task, computed with the network that was trained up to task z. The AVERAGE ACCURACY is then evaluated at time $z = \mathcal{T}$ (i.e., after having processed the last streamed task) when comparing continual settings with non-continual ones. Moreover, we analyze results at different time instants, considering not only accuracies but also other well-established metrics in the continual learning literature (Mai et al., 2022). The AVERAGE FORGETTING at time z, for each task j, compares acc_j^z with the best accuracy obtained in all the previous time instants (i.e., a positive forgetting means the model is performing worse than before).[8] In order to evaluate the way tuples of concepts are aligned to tasks, we considered what we refer to as TASK ALIGNMENT SCORE (TAS). This score is a direct transposition of the alignment measure considered when evaluating CEMs (Espinosa Zarlenga et al., 2022), which is named CONCEPT ALIGNMENT SCORE (CAS) in their work. The original implementation assumed supervised concepts, and the goal was to measure the alignment of each concept embedding in relation to the supervised concept it represented. In the context of this paper, however, we are interested instead in measuring the alignment of concept activation scores $\boldsymbol{c}$ and each task in $\mathcal{T}$ (averaged over tasks).[9] Similarly to the case of AVERAGE ACCURACY, we computed TAS at the end of learning or during the task streaming, following the same aggregation criteria. See Appendix B for an extended definition.

Concept-only Metrics. When restricting the evaluation to the concept bottleneck only, our goal is to evaluate how similar are the discovered concepts to the ground truth concepts annotations. We considered four different measures. The first one is another instance of the already introduced CAS, in this case named GROUND CONCEPT ALIGNMENT SCORE (GCAS), which measures the averaged (over ground truth concepts) alignment of concept activation scores $\boldsymbol{c}$ and each (expected) ground truth concept. The second and third metrics are based on the correlation matrices between ground truth concepts and discovered concepts, and vice-versa, respectively. In particular, we computed the MATTHEWS CORRELATION COEFFICIENT (MCC) (Matthews, 1975) for each entry of the two matrices. Differently from GCAS, here we focus on binarized concepts $\hat{\boldsymbol{c}}$. We aggregate the results on each matrix by means of the JENSEN-SHANNON DIVERGENCE (Menéndez et al., 1997) (JSD), either normalizing rows (JSD-R) or column (JSD-C) to probability distributions. In the former case, intuitively, we penalize predicted concepts which allocate probability to more than one true concept, while in the latter we penalize redundant representations of true concepts. The last metric we propose focuses on comparing the correlation between predicted concepts and themselves. The entries of the MCC-based correlation matrix are aggregated into the so-called DIAGONALIZATION SCORE (DIAGS), that measures how close is the correlation matrix to a diagonal matrix, where concepts are encoding independent aspects (the larger the better). See Appendix B for an extended description of the introduced metrics.

4.3 RESULTS

Tasks vs. Concepts. Table 1 reports the main results of our experimentation. We compare the AVERAGE ACCURACY and TAS in the continual learning and joint-training cases. The AVERAGE ACCURACY is slightly larger in the continual setting in KANDY-1 and the same comment holds in favour of the joint-training case in KANDY-2. Overall, differences are not significant. The main focus of this experiment stands in evaluating the properties of the developed concepts. We observe how the continual learning setting consistently achieves significantly better results in terms of TAS for both data sets, indicating that it produces concept representations that are more aligned with tasks over time. We remark that also the joint-training case is exploiting the proposed concept-level triplet-loss, thus the just described result is mostly due to the role of the concept replay, that helps in stabilizing the concept layer when learning in a continual manner. Moreover, this happens independently on the considered neural architectures. In order to better evaluate this result we performed an ablation study to analyze the impact of the triplet loss on concept learning: we report the value of TAS for this experiment in Figure 2. Results show that discarding the triplet loss yields a significant drop in performance for both ResNet50 and ViT models in the continual setting, especially in KANDY-2. The CNN model is almost unaffected, that might be due to the cold start of the training procedure. In fact, concepts that are badly developed in the early stages of life of the agent, end up in being stored in the replay buffer, and they tend to force the network toward potentially sub-optimal configurations. This suggest that there might be room for schemes that progressively enforce the triplet loss and concept replays. Comparing to joint-trained classifiers (Figure 2), the

[7] This idea is inspired by human education, in which a teacher is well aware, not only of the optimal order in which topics should be presented to students, but also at what point should their performance be evaluated for the best diagnosis of the learning process.

[8] We also measured forward/backward transfer, that we found to be not significantly pronounced, thus we avoid reporting them.

[9] We reused the code of (Espinosa Zarlenga et al., 2022), using the concept logit vector prior the sigmoid functions that yields $\boldsymbol{c}$.

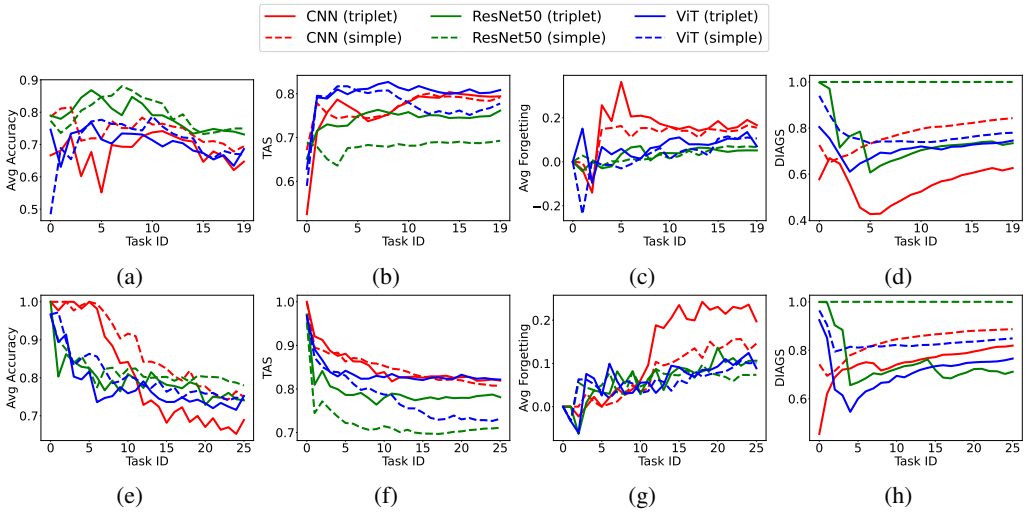


Figure 3: Metrics evolution through time for KANDY-1 (top) and KANDY-2 (bottom): AVERAGE ACCURACY (a, e), TAS (b, f), AVERAGE FORGETTING (c, g), DIAGS (d, h). Solid (dashed) lines are with (without) triplet loss.

impact of the triplet loss is less evident, suggesting that it is the combined effect of our concept-level loss and of a progressive learning setting that favours the development of tuples of concepts that better encode the task properties.

Model	Setting	KANDY-1		KANDY-2	
		AVG ACC ↑	TAS ↑	AVG ACC ↑	TAS ↑
CNN	Continual	0.65 ±0.02	**0.79** ±0.01	0.69 ±0.03	**0.82** ±0.00
	Joint	0.62 ±0.06	0.65 ±0.04	0.70 ±0.01	0.50 ±0.03
ResNet50	Continual	0.73 ±0.01	**0.76** ±0.01	0.74 ±0.02	**0.78** ±0.02
	Joint	0.70 ±0.02	0.68 ±0.07	0.79 ±0.02	0.57 ±0.02
ViT	Continual	0.69 ±0.01	**0.81** ±0.00	0.75 ±0.02	**0.82** ±0.01
	Joint	0.68 ±0.01	0.60 ±0.03	0.76 ±0.03	0.63 ±0.04

Table 1: Main result. Comparing the continual (task-incremental) and joint-training cases on the two datasets. Results are shown as $mean \pm std$ across 5 runs.

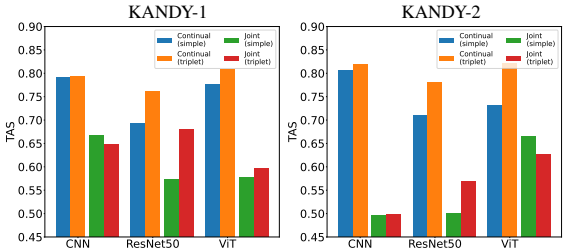


Figure 2: Task-concept alignment, continual and joint settings. We compare cases without triplet loss ("simple") and with triplet loss ("triplet").

The Role of Time. Figure 3 (a, b, e, f) inspects the same results of Table 1, with the addition of AVERAGE FORGETTING (c, g) and DIAGS (d, h), reporting their values at different time instants (tasks). In KANDY-1 (Figure 3 top) accuracy slowly decreases with the number of tasks, even if not in a critical way. The CNN model shows low accuracy in some initial tasks. Forgetting, which is not critically high in the deeper models, shows a bump in the early tasks for CNN. This confirms the previously reported statements on the reasons behind the more limited effect of the triplet loss in this model. Overall, the controlled level of forgetting is mostly due to the task-level experience replay. Interestingly, TAS is relatively low in the first task, and it quickly increases during the following few tasks. This means that the models are able to capture the key concepts that are needed to learn the task almost since the beginning, and these concepts are not subject to uncontrolled dynamics, but they stay stable, confirming the role of the concept-level replays. These comments, overall, holds also in the case of KANDY-2 (Figure 3 bottom) with some important differences. Accuracy is higher in the first tasks, suggesting that they can more easily be solved with the selected models. Forgetting seems to increase visibly, but mostly due to the larger number of tasks. In this case, the CNN shows evident forgetting issues at a later stage with respect to KANDY-1, which, however, still supports our explanations on the triplet loss impact. TAS confirms that the first tasks are easier to solve, and developing task-related concepts turns out to be easier as well.

Concepts vs. Concepts. Table 2 reports more detailed results characterizing concept-level metrics. In the continual setting, KANDY-2 yields more uncorrelated concepts (larger values of DIAGS), which is desirable. When comparing the ground truth concepts with the learned ones (GCAS, JSD-C, JSD-R), on average, the task-incremental case is better suited to model our expectation (consistently smaller JSD-C, mixed results for JSD-R and GCAS). However, when considering KANDY-1 we observe a different behaviour. Joint training leads to better DIAGS, although the

Model	Setting	KANDY-1				KANDY-2			
		DIAGS ↑	GCAS ↑	JSD-R ↓	JSD-C ↓	DIAGS ↑	GCAS ↑	JSD-R ↓	JSD-C ↓
CNN	Continual	0.63 ±0.02	0.58 ±0.01	13.81 ±0.57	2.90 ±0.07	**0.82** ±0.03	0.60 ±0.00	11.06 ±0.85	**4.53** ±0.28
	Joint	**0.96** ±0.03	**0.66** ±0.01	**6.53** ±2.04	**1.88** ±0.42	0.52 ±0.07	**0.65** ±0.02	**10.02** ±0.30	4.91 ±0.07
ResNet50	Continual	0.73 ±0.02	0.62 ±0.00	14.05 ±0.86	2.97 ±0.07	**0.71** ±0.03	0.58 ±0.00	**10.75** ±1.34	**4.58** ±0.18
	Joint	**0.89** ±0.02	**0.70** ±0.01	**10.73** ±1.03	**1.88** ±0.10	0.58 ±0.04	**0.62** ±0.02	10.82 ±0.57	4.67 ±0.22
ViT	Continual	0.75 ±0.05	0.60 ±0.00	13.54 ±0.31	3.06 ±0.03	**0.77** ±0.03	**0.58** ±0.00	11.05 ±0.37	**4.58** ±0.10
	Joint	**0.85** ±0.03	**0.70** ±0.00	**12.20** ±1.54	**2.17** ±0.07	0.55 ±0.10	0.56 ±0.01	**9.79** ±1.35	4.73 ±0.20

Table 2: Correlations between discovered concepts and (expected) ground truth concepts. The setting with the largest diagonal score (DIAGS) and at least another winning metric is highlighted in gray. Results are shown as $mean \pm std$ across 5 runs.

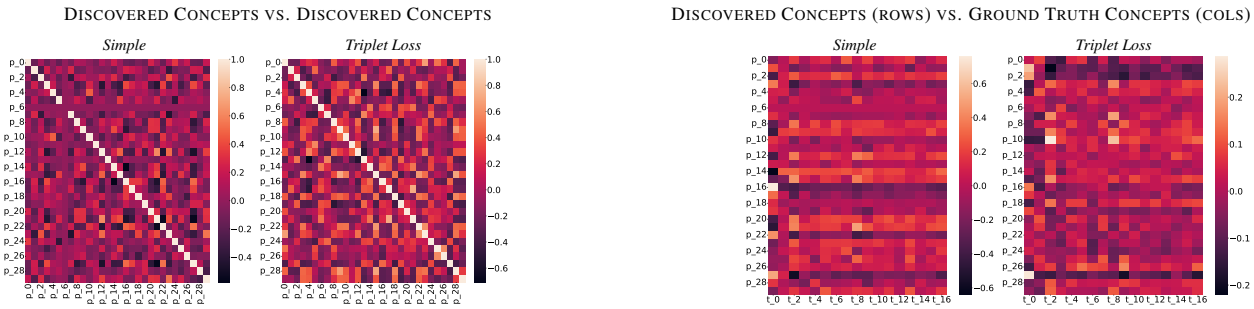


Figure 4: Correlation matrices computed via MCC. Left (a,b): Learned concepts vs. themselves. Right (c,d): Learned vs. ground truth concepts. Triplet loss is active only in (b) and (d). Results regard ViT model in the KANDY-2 data.

already discussed lower TAS (Table 1) suggests that concepts, overall, are less related to the tasks with respect to the task-incremental case, while being more uncorrelated. We motivate this result by the subpar accuracies in the first tasks of KANDY-1 (continual setting), which, paired with concept-level replays, leads to the development of some partially overlapping concepts over time. By comparing the ground truth concepts with the learned ones, we clearly see that the solution found in the task-incremental case is less aligned with our expectations. The network is indeed finding solutions that are task-related (Table 1), but not the ones we had hypothesized. Overall, we argue that this difference between the two datasets is also likely due to the larger number of tasks and, more importantly, of ground truth concepts in KANDY-2, where joint-training has to face the challenge of simultaneously discovering several concepts, which are instead disclosed incrementally in the continual setting. In general, the relationships between concept discovery and data set characteristics seem to play an interesting role which will be subject to further investigation.

Correlation Matrices. We investigate more detailed results characterizing concept-level metrics, reporting a visual example about the MCC coefficient, in Figure 4 (KANDY-2, ViT). In Figure 4 (a, b), we show the correlation among the automatically discovered concepts. In (a) the triplet loss is not used, while in (b) it is active. In the latter case, we clearly see the filled diagonal, while in the former there is an unused concept, suggesting that the triplet loss favours the usage of the available resources. However, when the triplet loss is used, we also see some more positive correlations (more reddish, orange areas), suggesting that there is indeed a bit more overlap among concepts. In Figure 4 (c, d) we report the correlation matrix between the learned concepts (rows) and the ground truth ones (columns), without (c) and with triplet loss (d). In the latter case it is easy to see that there are some dark horizontal stripes that are missing in (c). This suggests that some learned concepts (the ones of the darker rows) are either not much used or they encode properties that are far from the ground truth ones. This is actually expected, since there is nothing that prevents the network from encoding additional information with respect to our expectations. In the joint-case, due to the immediate availability of the data from all tasks, the ground truth concepts are more spread across the learned ones.

5 Related Work

Continual Interpretable-by-Design Methods. There is limited literature available at the intersection between interpretable-by-design methods (Rudin et al., 2022) and continual learning. In the context of CBMs, Marconato et al. (2022) investigated the effect of concept supervision on catastrophic forgetting in a class-incremental setting. To the extent of our knowledge, the work of Marconato et al. (2023) is the only application of continual learning to

Neuro-Symbolic Learning and Reasoning (Hitzler & Sarker, 2022). Their framework is able to learn and stabilize concepts in a class-incremental setting where disjoint sets of concepts are presented at each task. They empirically show how knowledge-driven approaches are more susceptible to catastrophic forgetting, but also tend to benefit more than purely-neural methods from continual learning strategies. Their proposed solution relies on concept annotations and exact background knowledge, and it is based on the combined use of a replay buffer for task labels and pseudo-rehearsal in the form of a KL-divergence term between concept distributions predicted by the current model and a snapshot on previous tasks. Compared to the works of Marconato et al. (2022; 2023), which strongly rely on concept supervision and evaluate their approaches on relatively large datasets spread across a limited temporal dimension (3 to 8 tasks), our work targets an unsupervised setting, where domain knowledge is unavailable and information is extracted gradually, over a significantly longer temporal horizon (20+ tasks) and with fewer samples. In the domain of reinforcement learning, Zabounidis et al. (2023) develop a concept-based policy for multi-agent settings, enabling a high-degree of interpretability and test-time interventions. Their approach assumes an oracle capable of annotating every observation with the full set of concepts, while our approach aims to learn concepts independently, from observations which disclose them gradually. Prototype-based models (Chen et al., 2019) are another family of interpretable models, alternative to CBMs, which store input portions for the sake of similarity-based inference. Unlike CBMs, the input patches are discovered in an unsupervised fashion. Rymarczyk et al. (2023) extend a prototype-based approach to the class-incremental setting, achieving a higher performance compared to non-continual baselines, without storing exemplars of previous classes. Our setting is orthogonal, as we allow memorization of examples, but do not allocate separate concept layers for each new task,[10] opening up to possibly lifelong settings.

Unsupervised Concept Discovery. Locatello et al. (2019) provided important theoretical results on the unfeasibility of generalized unsupervised disentanglement learning, without inductive biases both on architecture and datasets. In a large-scale experiment on variational autoencoder-based approaches across multiple datasets, they show that no method dominates the other and that successful disentanglement strongly depends on hyper-parameter tuning and random initialization. They also note how disentanglement (as measured by supervised metrics) correlates with sample complexity on downstream tasks, but this correlation may be a spurious result of metrics capturing also some form of "informativeness" of the latent space. More recently, Horan et al. (2021) identified two sufficient conditions (known in manifold learning) which make unsupervised disentanglement learning consistently achievable: local isometry and non-Gaussianity of latent factors. They also propose a teacher-student approach, where a traditional autoencoder (the student) is trained to encode the input image and decode the output from a latent space computed by the teacher. Although with a much higher number of samples, their approach consistently produces disentangled representations, while the same architecture, without their teacher-student paradigm, does not. Li et al. (2022) developed a Bayesian self-organizing map exploiting snapshots and a fuzzy masking mechanism to achieve good disentanglement in a continual learning setting. As we focus on deterministic and logically-interpretable representations, our method strongly differs from disentanglement of generative models. However, similar methodological constraints must be enforced, and the same difficulties are present in evaluating the goodness of representations without supervised information.

6 Conclusions and Future Work

In this paper we have addressed the problem of learning concept-based representations, in the case of concept-bottleneck models, considering two combined aspects that are largely overlooked in the literature: continual learning on sequentially provided tasks and unsupervised concept discovery. We proposed to develop a concept bottleneck layer by using multiple instances of a triplet loss, proposing a novel experience replay referred to as "concept" replay, exploiting the Hamming distance between soft-binary concept activations. We created (and shared) two datasets with a stream of Kandinsky Patterns. Our experimental results have shown that learning to develop concepts in a continual learning setting is beneficial in terms of task alignment metrics, with a negligible performance drop in accuracy. Moreover, results on ablation studies highlighted the quality of our proposed concept replays. In summary, the take-home messages of this paper are the following. (1) When learning in a continual manner, concept representations turn out to be more aligned with tasks. (2) Our analysis demonstrated the capability of developing task-relevant concepts in early stages of the learning process, and that the level of forgetting was not critical. (3) Comparing the ground truth concepts with the learned ones, the solution found in the task-incremental case is less aligned with the expectations. Solutions are indeed task-related, but not the ones we had hypothesized. (4) Some of the available concepts are scarcely used.

Many research directions can be foreseen for the future. We plan to make a further step in deepening the experiences of this paper, on longer streams of real world data, introducing a novel loss component to favour the activation (on average) of all the concepts. We also plan to extend the notion of concept intervenability, arguably the most important feature of concept-bottleneck models, to the continual learning case.

[10] Our task heads separate after the concept bottleneck, thus concepts are shared across all the tasks.

ACKNOWLEDGMENTS

This study was supported by TAILOR project funded by EU Horizon 2020 under GA No 952215, and by the Italian Ministry of University and Research (DM 351/2022, PNRR) that funded the scholarship of Luca Salvatore Lorello.

REFERENCES

Mohamed Abdelsalam, Mojtaba Faramarzi, Shagun Sodhani, and Sarath Chandar. Iirc: Incremental implicitly-refined classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11038–11047, 2021.

Md Zahangir Alom and Tarek M Taha. Network intrusion detection for cyber security using unsupervised deep learning approaches. In *2017 IEEE national aerospace and electronics conference*, pp. 63–69. IEEE, 2017.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and SIMONE CALDERARA. Dark experience for general continual learning: a strong, simple baseline. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15920–15930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/b704ea2c39778f07c617f6b7ce480e9e-Paper.pdf.

Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019.

Aristotelis Chrysakis and Marie-Francine Moens. Online continual learning from imbalanced data. In *International Conference on Machine Learning*, pp. 1952–1961. PMLR, 2020.

Gabriele Ciravegna, Pietro Barbiero, Francesco Giannini, Marco Gori, Pietro Lió, Marco Maggini, and Stefano Melacci. Logic explained networks. *Artificial Intelligence*, 314:103822, 2023.

William Damon and Richard M Lerner. *Child and adolescent development: An advanced course*. John Wiley & Sons, 2008.

Xingping Dong and Jianbing Shen. Triplet loss in siamese network for object tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 459–474, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Int. Conf. on Learn. Represent.*, 2021.

Christina M Eckhardt, Sophia J Madjarova, Riley J Williams, Mattheu Ollivier, Jón Karlsson, Ayoosh Pareek, and Benedict U Nwachukwu. Unsupervised machine learning methods and emerging applications in healthcare. *Knee Surgery, Sports Traumatology, Arthroscopy*, 31(2):376–381, 2023.

Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, et al. Concept embedding models: Beyond the accuracy-explainability trade-off. *Advances in Neural Information Processing Systems*, 35:21400–21413, 2022.

Adarsh Ghosh and Devasenathipathy Kandasamy. Interpretable artificial intelligence: why and when. *American Journal of Roentgenology*, 214(5):1137–1138, 2020.

Irina Higgins, Loic Matthey, Xavier Glorot, Arka Pal, Benigno Uria, Charles Blundell, Shakir Mohamed, and Alexander Lerchner. Early visual concept learning with unsupervised deep learning. *arXiv preprint arXiv:1606.05579*, 2016.

P Hitzler and M Sarker. Neuro-symbolic ai= neural+ logical+ probabilistic ai. *Neuro-Symbolic Artificial Intelligence: The State of the Art*, 342:173, 2022.

Andreas Holzinger, Anna Saranti, and Heimo Mueller. KANDINSKYPatterns–An experimental exploration environment for Pattern Analysis and Machine Intelligence. *arXiv preprint arXiv:2103.00519*, 2021.

Daniella Horan, Eitan Richardson, and Yair Weiss. When is unsupervised disentanglement possible? *Advances in Neural Information Processing Systems*, 34:5150–5161, 2021.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.

Chris Dongjoo Kim, Jinseo Jeong, and Gunhee Kim. Imbalanced continual learning with partitioning reservoir sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pp. 411–428. Springer, 2020.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pp. 5338–5348. PMLR, 2020.

Zhiyuan Li, Xiajun Jiang, Ryan Missel, Prashnna Kumar Gyawali, Nilesh Kumar, and Linwei Wang. Continual unsupervised disentangling of self-organizing representations. In *The Eleventh International Conference on Learning Representations*, 2022.

Zhiqiu Lin, Deepak Pathak, Yu-Xiong Wang, Deva Ramanan, and Shu Kong. Continual learning with evolving class ontologies. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 7671–7684. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/3255a7554605a88800f4e120b3a929e1-Paper-Conference.pdf.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pp. 4114–4124. PMLR, 2019.

Luca Salvatore Lorello and Marco Lippi. The challenge of learning symbolic representations. In Artur S. d'Avila Garcez, Tarek R. Besold, Marco Gori, and Ernesto Jiménez-Ruiz (eds.), *Proceedings of the 17th International Workshop on Neural-Symbolic Learning and Reasoning, La Certosa di Pontignano, Siena, Italy, July 3-5, 2023*, volume 3432 of *CEUR Workshop Proceedings*, pp. 44–61. CEUR-WS.org, 2023. URL https://ceur-ws.org/Vol-3432/paper4.pdf.

Xiao Luo, Haixin Wang, Daqing Wu, Chong Chen, Minghua Deng, Jianqiang Huang, and Xian-Sheng Hua. A survey on deep hashing methods. *ACM Transactions on Knowledge Discovery from Data*, 17(1):1–50, 2023.

Anita Mahinpei, Justin Clark, Isaac Lage, Finale Doshi-Velez, and Weiwei Pan. Promises and pitfalls of black-box concept learning models. *arXiv preprint arXiv:2106.13314*, 2021.

Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022.

Emanuele Marconato, Gianpaolo Bontempo, Stefano Teso, Elisa Ficarra, Simone Calderara, and Andrea Passerini. Catastrophic forgetting in continual concept bottleneck models. In *International Conference on Image Analysis and Processing*, pp. 539–547. Springer, 2022.

Emanuele Marconato, Gianpaolo Bontempo, Elisa Ficarra, Simone Calderara, Andrea Passerini, and Stefano Teso. Neuro-symbolic continual learning: Knowledge, reasoning shortcuts and concept rehearsal. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23915–23936. PMLR, 2023. URL https://proceedings.mlr.press/v202/marconato23a.html.

Brian W Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.

ML Menéndez, JA Pardo, L Pardo, and MC Pardo. The jensen-shannon divergence. *Journal of the Franklin Institute*, 334(2):307–318, 1997.

Claudio Novelli, Mariarosaria Taddeo, and Luciano Floridi. Accountability in artificial intelligence: what it is and how it works. *AI & SOCIETY*, pp. 1–12, 2023.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys*, 16:1–85, 2022.

Dawid Rymarczyk, Joost van de Weijer, Bartosz Zieliński, and Bartlomiej Twardowski. Icicle: Interpretable class incremental continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1887–1898, 2023.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.

Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2024. doi: 10.1109/TPAMI.2024.3367329.

Renos Zabounidis, Joseph Campbell, Simon Stepputtis, Dana Hughes, and Katya P Sycara. Concept learning for interpretable multi-agent reinforcement learning. In *Conference on Robot Learning*, pp. 1828–1837. PMLR, 2023.

Eleni Ziori and Zoltán Dienes. The time course of implicit and explicit concept learning. *Consciousness and Cognition*, 21(1):204–216, 2012.

A CURRICULA DEFINITION

Each task within a dataset participates to a curriculum which is characterized by two sample sets. The sample sets are defined by a tree-based syntax (not shown) and a Prolog rule `valid(C)`, which asserts that the generated compositional structure `C` is a valid Kandinsky pattern for the positive set of that task (with the negative set being characterized by its negation). The compositional structure is a tree where leaves are atomic shapes with properties (queriable by predicates `extract_*`). Each intermediate node is characterized by a compositional operator (`COMP`), corresponding to the alignment concepts we described in KANDY-2, and a list of children (`L`). Predicates `extract_op/2, extract_children/2, extract_op_and_chld/3` allow to select relevant elements from the tree. The `contains/2` predicate retrieves an arbitrary element of the tree. Other predicates have intuitive semantics. Where applicable, we describe sample-balancing strategies used to mitigate reasoning shortcuts (Marconato et al., 2023). Tables 3 and 4, summarize concept progression along tasks and, when multiple concepts are required to solve a task, the type of interaction they are involved in.

A.1 KANDY-1

Task ID. 0
Description. An arbitrary triangle (pos.), or an arbitrary non-triangle (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_shape(C1, triangle).
```

Task ID. 1
Description. An arbitrary square (pos.), or an arbitrary non-square (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_shape(C1, square).
```

Task ID. 2
Description. An arbitrary circle (pos.), or an arbitrary non-circle (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_shape(C1, circle).
```

Task ID. 3
Description. A red object (pos.), or a non-red object (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, red).
```

Task ID. 4
Description. A green object (pos.), or a non-green object (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, green).
```

Task ID. 5
Description. A blue object (pos.), or a non-blue object (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, blue).
```

Task ID. 6
Description. A cyan object (pos.), or a non-cyan object (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, cyan).
```

Task ID. 7
Description. A magenta object (pos.), or a non-magenta object (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, magenta).
```

Task ID. 8
Description. A yellow object (pos.), or a non-yellow object (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, yellow).
```

Task ID. 9
Description. Two shapes, the rightmost is a red triangle (pos.), or two arbitrary shapes (neg.)
Sample balancing. The negative set can contain a red triangle on the left.
Rule.

```
valid(C) :- extract_children(C, L), last(L, C1), extract_shape(C1, triangle),
            extract_color(C1, red).
```

Task ID. 10
Description. Three to six shapes, the rightmost is a red triangle (pos.), or three to six arbitrary shapes (neg.)
Sample balancing. The negative set can contain a red triangle at some random position.
Rule.

```
valid(C) :- extract_children(C, L), last(L, C1), extract_shape(C1, triangle),
            extract_color(C1, red).
```

Task ID. 11
Description. Three to six shapes, at least one of them is a circle and the rightmost is a red triangle (pos.), or three to six arbitrary shapes (neg.)
Sample balancing. The negative set can contain a red triangle at some random position, a circle, but not the rightmost red triangle, or the rightmost red triangle, but not a circle.
Rule.

```
valid(C) :- extract_children(C, L), last(L, C1), member(C2, L),
            extract_shape(C1, triangle), extract_color(C1, red),
            extract_shape(C2, circle).
```

Task ID. 12
Description. Three to six shapes, at least one of them is blue and the rightmost is a red triangle (pos.), or three to six

arbitrary shapes (neg.)
Sample balancing. The negative set can contain a red triangle at some random position, a blue object, but not the rightmost red triangle, or the rightmost red triangle, but not the blue object.
Rule.

```
valid(C) :- extract_children(C, L), last(L, C1), member(C2, L),
            extract_shape(C1, triangle), extract_color(C1, red),
            extract_color(C2, blue).
```

Task ID. 13
Description. A triangle and a square sharing the same color (pos.), or two arbitrary objects (neg.)
Sample balancing. The negative set can contain a triangle and a square of different colors, or two arbitrary shapes of the same color.
Rule.

```
valid(C) :- recursive_contains(C, C1), recursive_contains(C, C2),
            same_color(_, [C1, C2]), extract_shape(C1, triangle),
            extract_shape(C2, square).
```

Task ID. 14
Description. A palindrome of three objects (i.e., ABA, pos.), or two copies of the same object plus an arbitrary one, in a non-palindromic configuration (i.e., AAB or BAA, neg.)
Rule.

```
valid(C) :- extract_children(C, L), reverse(L, L).
```

Task ID. 15
Description. A stylized "house" in the form of a triangle above a square of the same size (pos.), or two vertically stacked objects (neg.)
Rule.

```
house(C) :- extract_op_and_chld(C, stack, [C1, C2]), extract_shape(C1, triangle),
            extract_shape(C2, square), same_size(_, [C1, C2]).
valid(C) :- contains(C, C1), house(C1).
```

Task ID. 16
Description. A stylized "car" in the form of two horizontally aligned circles of the same size and color (pos.), or two horizontally aligned objects (neg.)
Rule.

```
car(C) :- extract_op_and_chld(C, side_by_side, [C1, C2]),
          extract_shape(C1, circle), extract_shape(C2, circle),
          same_size(_, [C1, C2]), same_color(_, [C1, C2]).
valid(C) :- contains(C, C1), car(C1).
```

Task ID. 17
Description. A stylized "tower" in the form of two or three vertically stacked squares of the same size (pos.), or two to three vertically stacked objects (neg.)
Rule.

```
tower(C) :- extract_op_and_chld(C, stack, L), same_shape(square, L),
            same_size(_, L), length(L, N), N >= 2, N =< 3.
valid(C) :- contains(C, C1), tower(C1).
```

Task ID. 18
Description. A stylized "wagon" in the form of two or three horizontally aligned squares of the same size (pos.), or two to three horizontally aligned objects (neg.)
Rule.

```
wagon(C) :- extract_op_and_chld(C, side_by_side, L), same_shape(square, L),
            same_size(_, L), length(L, N), N >= 2, N =< 3.
valid(C) :- contains(C, C1), wagon(C1).
```

Task ID. 19
Description. A stylized "traffic light" in the form of three stacked circles, red, yellow and green, of the same size (pos.), or three stacked circles of arbitrary color and size (neg.)
Rule.

```
traffic_light(C) :- extract_op_and_chld(C, stack, [C1, C2, C3]),
                    same_shape(circle, [C1, C2, C3]),
                    same_size(_, [C1, C2, C3]), extract_color(C1, red),
                    extract_color(C2, yellow), extract_color(C3, green).
valid(C) :- contains(C, C1), traffic_light(C1).
```

A.2 KANDY-2

Task ID. 0
Description. One object (pos.), or many objects (four to five, neg.)
Rule.

```
valid(C) :- extract_children(C, L), length(L, 1).
```

Task ID. 1
Description. Few objects (two to three, pos.), or many (neg.)
Rule.

```
valid(C) :- extract_children(C, L), length(L, N), N > 1, N =< 3.
```

Task ID. 2
Description. Few objects (two to three, pos.), or one (neg.)
Rule.

```
valid(C) :- extract_children(C, L), length(L, N), N > 1.
```

Task ID. 3
Description. Two to five diagonally displaced objects (pos.), or two to five vertically or horizontally displaced objects (neg.)
Rule.

```
valid(C) :- extract_operator(C, COMP), diag(COMP).
```

Task ID. 4
Description. Two to five vertically displaced objects (pos.), or two to five horizontally displaced objects (neg.)
Rule.

```
valid(C) :- extract_operator(C, stack).
```

Task ID. 5
Description. Two to five vertically displaced objects, or few arbitrarily displaced objects (pos.), or its negation (neg.)
Sample balancing. The negative set is defined by De Morgan rule (thus balancing the sampling probability), instead of sample rejection on arbitrary sequences.
Rule.

```
valid(C) :- extract_operator(C, stack).
valid(C) :- extract_operator(C, stack_reduce_bb).
valid(C) :- extract_children(C, L), length(L, N), N > 1, N =< 3.
```

Task ID. 6
Description. Few objects diagonally displaced from the upper left corner to the lower right (pos.), or its negation (neg.)
Sample balancing. De Morgan.
Rule.

```
valid(C) :- extract_op_and_chld(C, diag_ul_lr, L), length(L, N), N > 1, N =< 3.
```

Task ID. 7
Description. Exists a triangle among one to five objects (pos.), or one to five non-triangles (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_shape(C1, triangle).
```

Task ID. 8
Description. Exists a square among one to five objects (pos.), or one to five non-squares (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_shape(C1, square).
```

Task ID. 9
Description. Exists a circle among one to five objects (pos.), or one to five non-circles (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_shape(C1, circle).
```

Task ID. 10
Description. Few objects, all of them are squares (pos.), or non-few squares, or few arbitrary objects (neg.)
Rule.

```
valid(C) :- extract_children(C, L), length(L, N), N > 1, N =< 3, same_shape(square, L).
```

Task ID. 11
Description. Either vertically stacked arbitrary two to five objects, or a circle within an arbitrary displacement (pos.), or arbitrary non-circles (neg.)
Rule.

```
valid(C) :- extract_operator(C, stack).
valid(C) :- extract_operator(C, stack_reduce_bb).
valid(C) :- contains(C, C1), extract_shape(C1, circle).
```

Task ID. 12
Description. Exists a red object among one to five objects (pos.), or one to five non-red objects (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, red).
```

Task ID. 13
Description. Exists a green object among one to five objects (pos.), or one to five non-green objects (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, green).
```

Task ID. 14
Description. Exists a blue object among one to five objects (pos.), or one to five non-blue objects (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, blue).
```

Task ID. 15
Description. Exists a yellow object among one to five objects (pos.), or one to five non-yellow objects (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, yellow).
```

Task ID. 16
Description. Exists a magenta object among one to five objects (pos.), or one to five non-magenta objects (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, magenta).
```

Task ID. 17
Description. Exists a cyan object among one to five objects (pos.), or one to five non-cyan objects (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, cyan).
```

Task ID. 18
Description. Few objects, one of which is both red and a square (pos.), or one to five arbitrary objects (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, red), extract_shape(C1, square),
            extract_children(C, L), length(L, N), N > 1, N =< 3.
```

Task ID. 19
Description. Many objects, one of which is blue and another of which is a circle (pos.), or non-many arbitrary objects, or many non-blue-non-circle objects (neg.)
Rule.

```
valid(C) :- contains(C, C1), extract_color(C1, blue), contains(C, C2),
            extract_shape(C2, circle), extract_children(C, L),
            length(L, N), N > 3.
```

Task ID. 20
Description. Two to five objects, all of which share the same arbitrary shape (pos.), or two to five arbitrary objects (neg.)
Rule.

```
valid(C) :- extract_children(C, L), same_shape(_, L).
```

Task ID. 21
Description. Two to five objects, all of which share the same arbitrary color (pos.), or two to five arbitrary objects (neg.)
Rule.

```
valid(C) :- extract_children(C, L), same_color(_, L).
```

Task ID. 22
Description. Two to five objects, all of which share the same arbitrary size (pos.), or two to five arbitrary objects (neg.)
Rule.

```
valid(C) :- extract_children(C, L), same_size(_, L).
```

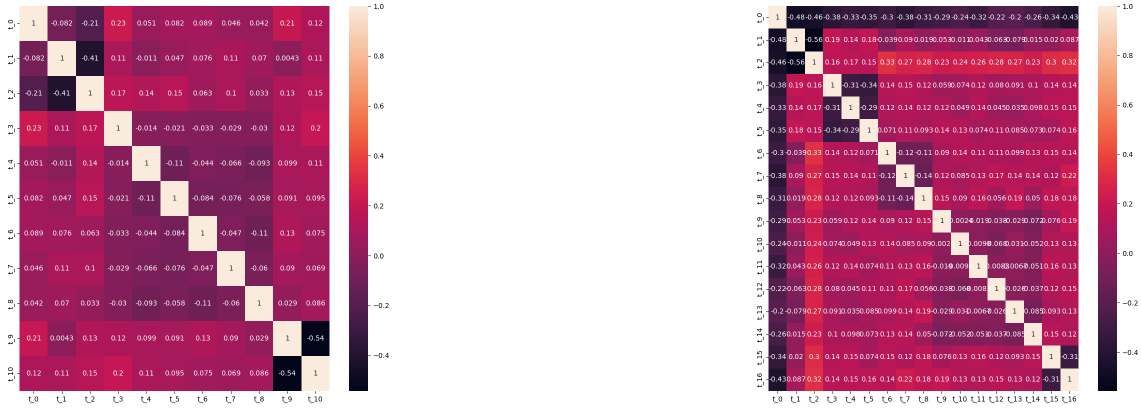
Task ID. 23
Description. Two to five objects, all of which are identical (pos.), or two to five arbitrary objects (neg.)
Rule.

```
valid(C) :- extract_children(C, L), same_shape(_, L), same_color(_, L), same_size(_, L).
```

Task ID. 24
Description. Two to five objects, all of which share at least one attribute (pos.), or two to five arbitrary objects (neg.)
Rule.

```
valid(C) :- extract_children(C, L), same_shape(_, L).
valid(C) :- extract_children(C, L), same_color(_, L).
valid(C) :- extract_children(C, L), same_size(_, L).
```

Task ID. 25
Description. Two to five objects sharing the same color, one of them is a circle (pos.), or two to five arbitrary objects (neg.)
Sample balancing. The negative set can either contain same-color objects but no circle, or contain a circle but have no color in common.
Rule.



(a) KANDY 1 (b) KANDY 2

Figure 5: Heatmaps of correlations between ground truth concepts in the datasets. It can be noted how within-group concepts negatively correlate with each other (darker regions along the diagonal) and how inter-group are roughly independent. As tasks in KANDY 2 rarely contain a single element, the corresponding concept is negatively correlated with respect to everything else.

```
valid(C) :- extract_children(C, L), same_color(_, L), contains(C, C1),
            extract_shape(C1, circle).
```

Task	Interaction	Shape			Color						Size	
		▲	■	●							□	■
T0		∃										
T1			∃									
T2				∃								
T3					∃							
T4						∃						
T5							∃					
T6								∃				
T7									∃			
T8										∃		
T9	Binding	∃			∃							
T10	Binding	∃			∃							
T11	Binding, AND	∃		∃	∃							
T12	Binding, AND	∃			∃		∃					
T13	Binding	∃	∃		✓	✓	✓	✓	✓	✓		
T14	Complex	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
T15	Pattern	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓
T16	Pattern			✓	✓	✓	✓	✓	✓	✓	✓	✓
T17	Pattern		✓								✓	✓
T18	Pattern		✓								✓	✓
T19	Pattern			✓	✓	✓				✓	✓	✓

Table 3: Concept progression in tasks of KANDY-1. When the decision boundary depends on multiple concepts, the type of interaction is summarized by a keyword. Concepts are either existentially quantified (∃) or require task-specific handling (✓). The dashed line demarks the point in which we validate our models, as it delimits the boundary between the last task in which elementary concepts are presented and the first complex task.

B METRICS

Performance Metrics. For each task, we measured the accuracy acc_j, which is the micro-accuracy of the binary classification problem of the j-th task. It is obtained by averaging the ratio of right predictions for the positive class over the set of positive examples, and the ratio of right predictions for the negative class over the set of negative

Task	Interaction	Number			Alignment			Shape			Color						Size	
		one	few	many	dia.	hrz.	vrt.	▲	■	●							□	□
T0		✓																
T1			✓															
T2			✓															
T3					✓													
T4							✓											
T5	OR		✓				✓											
T6	AND		✓		✓													
T7								∃										
T8									∃									
T9										∃								
T10	AND		✓						∀									
T11	OR						✓			∃								
T12											∃							
T13												∃						
T14													∃					
T15														∃				
T16															∃			
T17																∃		
T18	Binding								∃		∃							
T19	AND			✓						∃			∃					
T20	XOR							∀	∀	∀								
T21	XOR										∀	∀	∀	∀	∀	∀		
T22	XOR																∀	∀
T23	Group AND							∀	∀	∀	∀	∀	∀	∀	∀	∀	∀	∀
T24	Group OR							∀	∀	∀	∀	∀	∀	∀	∀	∀	∀	∀
T25	AND + XOR									∃	∀	∀	∀	∀	∀	∀		

Table 4: Concept progression in tasks of KANDY-2. When the decision boundary depends on multiple concepts, the type of interaction is summarized by a keyword. Concepts are existentially quantified (∃), universally quantified (∀), or require task-specific handling (✓). The dashed line demarks the point in which we validate our models, as it delimits the boundary between the last task in which elementary concepts are presented and the first complex task.

examples. It can be written as

$$\text{acc}_j = \frac{1}{2} \cdot \frac{TP_j}{TP_j + FN_j} + \frac{1}{2} \cdot \frac{TN_j}{TN_j + FP_j}$$

where TP_j, TN_j, FP_j, FN_j is the number of true positives, true negatives, false positives, false negatives for task j, respectively. In continual learning settings, data is streamed in a task-oriented order, thus the data of task $j = 0$ first, then the data of task $j = 1$ and so on and so forth, up to the last task $j = |\mathcal{T}| - 1$. Metrics can be computed at different "time" instants, i.e., using different states of the neural models. For example, the model at "time" z is the neural model that has learned using data streamed up to task z (included). We use the (overloaded) notation acc_j^t to indicate the accuracy on task j measure with the model that has learned using data streamed up to task z. Following the definitions in (Mai et al., 2022), the AVERAGE ACCURACY at time z is defined as the average of the task-related accuracies exploiting the neural model that was developed up to the z-th task,

$$\text{AVERAGE ACCURACY}(z) = \frac{1}{z} \sum_{j=0}^{z} \text{acc}_j^z,$$

where acc_j^z is the already introduced micro-accuracy of the binary classification problem of the j-th task. The AVERAGE FORGETTING at time z is defined as

$$\text{AVERAGE FORGETTING}(z) = \frac{1}{z-1} \sum_{j=0}^{z-1} \left(\text{acc}_i^\star - \text{acc}_j^z \right),$$

where $\text{acc}_j^\star$ is the best accuracy obtained on task j so far, i.e., $\max_{k \in \{1,\dots,z-1\}} \text{acc}_j^k$.

Alignment Metrics. We computed the GROUND CONCEPT ALIGNMENT SCORE (GCAS) and TASK ALIGNMENT SCORE (TAS) on the concept bottleneck layer, as described in Espinosa Zarlenga et al. (2022), but with two main differences. The original implementation assumed supervised concept learning, thus a 1-to-1 alignment is assumed between ground

truth ($\mathcal{R}$) and predicted concepts ($\mathcal{C}$), and comparisons are made only between r_i and c_i, after a suitable permutation of $\mathcal{C}$. As we deal with an unsupervised concept learning scenario, we lift the 1-to-1 alignment requirement by computing the 1-to-many mapping between each r_i and $\mathcal{C}$. The second difference deals with which predicted concept vector is used in computations. In the original proposal, authors use concept embeddings, but we argue that this choice introduces noise in the interpretation of results, as the mixture of linear projections ϕ_j^+ and ϕ_j^- can alter unpredictably the contribution of concept c_j.[11] In order to mitigate this issue, we compute GCAS and TAS by using the concept logit vector prior to sigmoid activation and linear embedding mapping.

For each true concept r_j, K-medoids clustering is performed on the predicted concept logits vector, varying the number of clusters ρ. The GCAS is the area under the curve of the homogeneity score with respect to the number of clusters, normalized against the theoretical perfect homogeneity. TAS is computed in the same way, by replacing true concepts with task labels y. Let n be the number of test samples and $h(x, \Pi_i(\rho)))$ the homogeneity score of clusterization $\Pi_i(\rho)$ with respect to cluster labels x, then:

$$GCAS = \frac{1}{n-2}\sum_{\rho=2}^{n}\left(\frac{1}{|\mathcal{C}|}\sum_{i=0}^{|\mathcal{C}|-1} h(\mathcal{R}, \Pi_i(\rho, \tilde{g}(\psi(x)))))\right)$$

$$TAS = \frac{1}{n-2}\sum_{\rho=2}^{n}\left(\frac{1}{|\mathcal{C}|}\sum_{i=0}^{|\mathcal{C}|-1} h(Y, \Pi_i(\rho, \tilde{g}(\psi(x)))))\right)$$

where $\tilde{g}$ is the concept activation score before the normalization computed via the sigmoid function.

Both GROUND CONCEPT ALIGNMENT SCORE and TASK ALIGNMENT SCORE are extended to the continual task incremental setting, by recomputing them at the end of each task. We report these scores computed on the test set samples accumulated from every task, from the beginning up to the lastly trained one.

Correlation Metrics. We build correlation matrices between concepts, by means of MATTHEWS CORRELATION COEFFICIENT (MCC) (Matthews, 1975). Given the binarized predicted concept vector $\hat{\boldsymbol{c}}$ and the ground truth vector $\boldsymbol{r}$, let n be the number of samples in the test set, $n_{i,j}^{11}$ the number of samples in which $\hat{c}_i = r_j = 1$, $n_i^{1\bullet}$ the number of samples in which $\hat{c}_i = 1$, and $n_j^{\bullet 1}$ the number of samples in which $r_j = 1$, then the MCC is computed as:

$$\boldsymbol{\Phi}_{i,j}^{CR} = \frac{n \cdot n_{i,j}^{11} - n_i^{1\bullet} \cdot n_j^{\bullet 1}}{\sqrt{n_i^{1\bullet} \cdot n_j^{\bullet 1} \cdot (n - n_i^{1\bullet}) \cdot (n - n_j^{\bullet 1})}}.$$

In a similar fashion, we compute the correlation matrix $\boldsymbol{\Phi}_{i,j}^{CC}$ between predicted concepts and themselves. Like GCAS and TAS, we extend MCC to the continual learning case, by computing task-incremental versions. To aggregate MCC matrices into scalar values, we compute three metrics. The DIAGONALIZATION SCORE condenses $\boldsymbol{\Phi}^{CC}$ into a $[0, 1]$ value, with 1 corresponding to a diagonal matrix. It is computed by taking the absolute value of $\boldsymbol{\Phi}^{CC}$, masking its diagonal and averaging the remaining elements, which are finally subtracted from 1:

$$\text{DIAGS} = 1 - \frac{\sum_{i=0}^{|\mathcal{C}|-1}\sum_{j=0}^{|\mathcal{C}|-1}(|\boldsymbol{\Phi}_{i,j}^{CC}| \cdot (\boldsymbol{I}\boldsymbol{I}_{i,j}^{T} - \boldsymbol{I}_{i,j}))}{|\mathcal{C}|^2}.$$

Due to the lack of a 1-to-1 correspondence between ground truth and predicted concepts, condensation of $\boldsymbol{\Phi}^{CR}$ is more involved and requires two metrics to adequately capture "quality". For their computation we rely on the JENSEN-SHANNON DIVERGENCE (Menéndez et al., 1997) (JSD), and output two scalar scores. The JSD-C score summarizes how likely it is for predicted concepts to represent ground truth ones. It is computed by first splitting $\boldsymbol{\Phi}^{CR}$ into positive and negative correlations ($\boldsymbol{\Phi}^{+}$ and $\boldsymbol{\Phi}^{-}$, the latter having its sign flipped):

$$\boldsymbol{\Phi}^{+} = \boldsymbol{\Phi}_{>0}^{CR}$$
$$\boldsymbol{\Phi}^{-} = -\boldsymbol{\Phi}_{<0}^{CR}.$$

[11] This is particularly true in the case where linear projections do not share weights between concepts, but it also holds for the case of shared weights.

Positive and negative correlations are then normalized row-wise, so that they can be interpreted as probability distributions along $\mathcal{R}$:

$$\tilde{\mathbf{\Phi}}^{C+}_{i,j} = \frac{\mathbf{\Phi}^{+}_{i,j}}{\sum_{k=0}^{|\mathcal{R}|-1} \mathbf{\Phi}^{+}_{i,k}}$$
$$\tilde{\mathbf{\Phi}}^{C-}_{i,j} = \frac{\mathbf{\Phi}^{-}_{i,j}}{\sum_{k=0}^{|\mathcal{R}|-1} \mathbf{\Phi}^{-}_{i,k}}.$$

These probabilities can then be compared, by means of JS divergence, against the ideal case in which only one predicted concept c_i is correlated with the considered true concept r_j.

$$\hat{\Phi}^{C+}_{i,:} = \texttt{one\_hot}(\texttt{argmax}(\mathbf{\Phi}^{+}_{i,:}))$$
$$\hat{\Phi}^{C-}_{i,:} = \texttt{one\_hot}(\texttt{argmax}(\mathbf{\Phi}^{-}_{i,:}))$$
$$\text{JSD-C}^{+} = \frac{\sum_{j=0}^{|\mathcal{R}|-1} D_{\text{JS}}(\tilde{\Phi}^{C+}_{:,j} \parallel \hat{\Phi}^{C+}_{:,j})}{|\mathcal{R}|}$$
$$\text{JSD-C}^{-} = \frac{\sum_{j=0}^{|\mathcal{R}|-1} D_{\text{JS}}(\tilde{\Phi}^{C-}_{:,j} \parallel \hat{\Phi}^{C-}_{:,j})}{|\mathcal{R}|}.$$

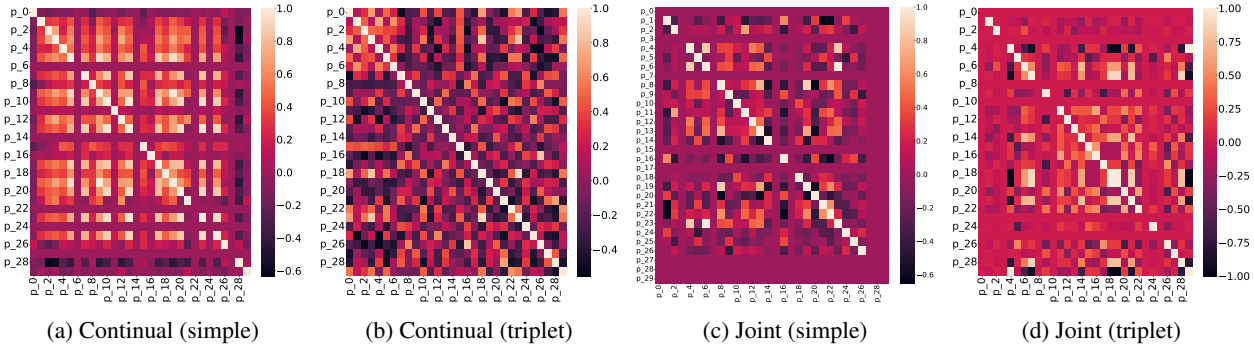
Positive and negative correlations are finally averaged to obtain the final score:

$$\text{JSD-C} = \frac{\text{JSD-C}^{+} + \text{JSD-C}^{-}}{2}.$$

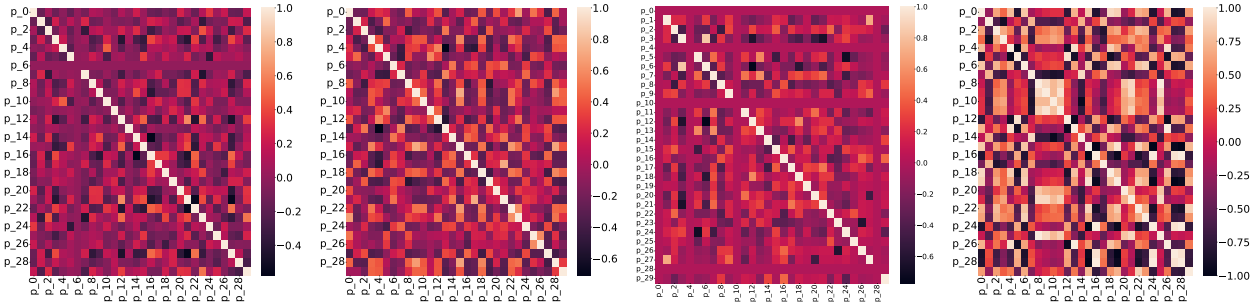
JSD-R is similarly computed by normalizing along columns and averaging with respect to $\mathcal{C}$, and it can be interpreted as how likely it is for a ground truth concept to be represented by each of the predicted concepts.

C ViT HEATMAPS

The following figures highlight correlations between predicted concepts and themselves (Figure 6) and between predicted concepts and ground truth (Figure 7) for the experiments performed on ViT.

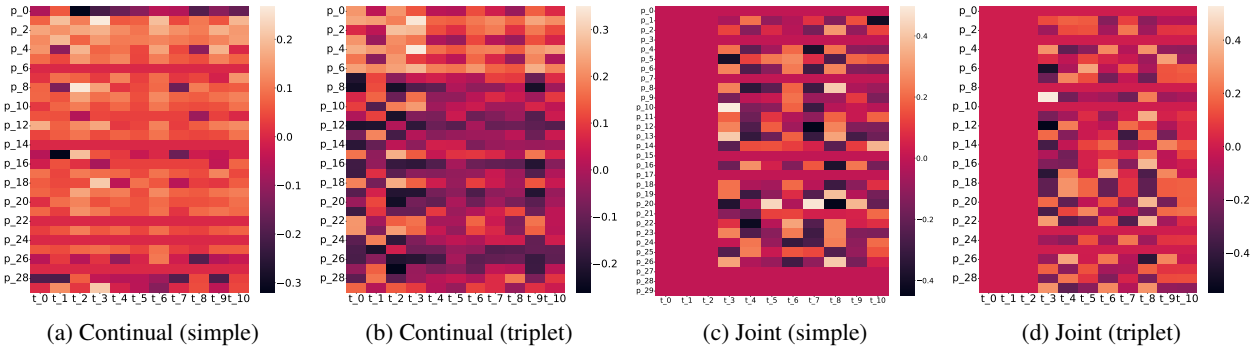


(a) Continual (simple) (b) Continual (triplet) (c) Joint (simple) (d) Joint (triplet)

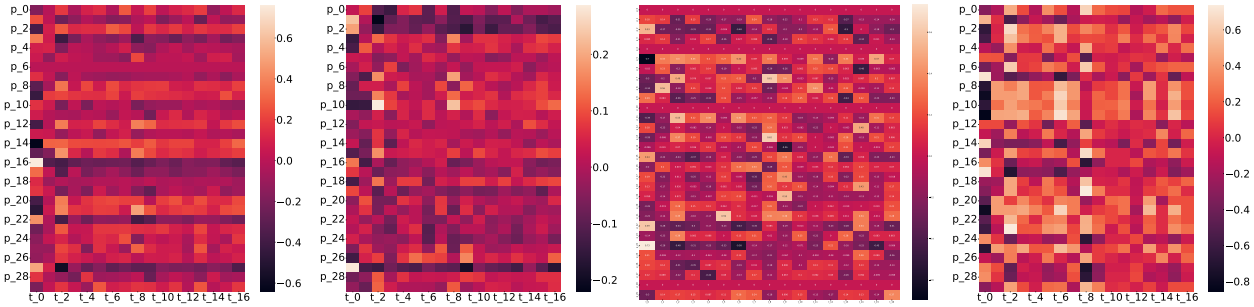


(e) Continual (simple) (f) Continual (triplet) (g) Joint (simple) (h) Joint (triplet)

Figure 6: Correlation heatmaps of concepts predicted by ViT. First row: KANDY 1, second row: KANDY 2. It can be noted how, KANDY 1 possesses an higher degree of sparsity (unused concepts), especially when the triplet loss is not exploited. The joint cases are characterized by duplicated (similar rows) and entangled concepts (similar columns), even when the triplet loss is active.



(a) Continual (simple) (b) Continual (triplet) (c) Joint (simple) (d) Joint (triplet)



(e) Continual (simple) (f) Continual (triplet) (g) Joint (simple) (h) Joint (triplet)

Figure 7: Heatmaps of correlations between predicted concepts (rows) and ground truth (columns) for ViT. First row: KANDY 1, second row: KANDY 2. It can be noted how, in KANDY 1, the joint cases fail to learn shape concepts, and how the triplet loss beneficially affects diversification of learned concepts (at the expenses of correlation with respect to the ground truth). In KANDY 2, the joint case presents overall higher correlations between true and predicted concepts, but also an higher level of duplication.