ON THE QUALITY ASSURANCE OF CONCEPT-BASED REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent work on Explainable AI has focused on concept-based explanations, where deep learning models are explained in terms of high-level units of information, referred to as concepts. In parallel, the field of disentanglement learning has explored the related notion of finding underlying factors of variation in the data that have interpretability properties. Despite their overlapping purpose, the metrics to evaluate the quality of concepts and factors of variation in the two fields are not aligned, hindering a systematic comparison. In this paper we consider factors of variation as concepts and thus unify the notations in concept and disentanglement learning. Next, we propose metrics for evaluating the quality of concept representations in both approaches, in the presence and in the absence of ground truth concept labels. Via our proposed metrics, we benchmark state-of-the-art methods from both families, and propose a set of guidelines to determine the impact that supervision may have on the quality of learnt concept representations.

1 INTRODUCTION

Addressing the lack of interpretability of deep neural networks (DNNs) has given rise to explainability methods, most common of which are feature importance methods (Lundberg & Lee, 2017; Ribeiro et al., 2016) that quantify the contribution of input features to certain predictions (Bhatt et al., 2020). However, input features are not necessarily the most intuitive explanations, in particular when using low-level features such as pixels. *Concept-based explainability* (Koh et al., 2020; Ghorbani et al., 2019; Yeh et al., 2020; Ciravegna et al., 2021) remedies this issue by constructing an explanation at a concept level, where concepts are considered high-level and semantically meaningful units of information commonly used by humans to explain their decisions. Furthermore, concepts allow users to improve a model's performance via concept interventions, in which mispredicted concepts are corrected using expert knowledge (Koh et al., 2020).

In practice, what constitutes a concept is data-dependent, ranging from a group of pixels for image data (Ghorbani et al., 2019; Koh et al., 2020; Yeh et al., 2020), to a sequence of words and sub-graphs for text and graph-based data, respectively (Yeh et al., 2020; Magister et al., 2021). While all of these definitions are specific to the data modality, they commonly refer to an intermediate representation of the input data that has certain properties. Summarising concepts as intermediate representations of the data makes them analogous to factors of variation in *disentanglement learning*, where the assumption is that there exists a generative process capable of producing a high-dimensional dataset using a finite number of factors (Bengio et al., 2013). Such factors constitute a disentangled intermediate representation of the data with interpretability (Bengio et al., 2013; Higgins et al., 2017), fairness (Creager et al., 2019), and predictive performance (Locatello et al., 2019; 2020b) properties. The difference between Concept Learning (CL) and Disentanglement Learning (DGL) remains in that concepts in CL are formed based on the supervision directly from concept labels or from a downstream task, whereas generative models (e.g., Variational Autoencoders (VAEs) (Kingma & Welling, 2014; Higgins et al., 2017)) that serve as the basis of DGL are un-/semi-supervised and factors of variation are directly informed by the distribution of the input data.

Since CL and DGL were developed independently, much of their connection remains unexplored. In particular, the metrics used to evaluate the quality of intermediate representations in each sub-field are not aligned, despite their overlapping goals. Metrics in the concept literature (Koh et al., 2020; Kazhdan et al., 2020; Yeh et al., 2020) are mainly concerned with the properties of learnt concepts

w.r.t. the downstream task. On the other hand, given the lack of a downstream task, metrics in disentanglement literature (Ridgeway & Mozer, 2018; Locatello et al., 2019; Higgins et al., 2017) are mainly concerned with the properties of the learnt representations, referred to as latent codes, w.r.t. the ground truth factors of variation. We argue that concepts/latent codes, as surrogate to inputs, need to have the following properties w.r.t.:

(i) Inputs: concepts should correspond to semantically meaningful and coherent input sub-spaces.
(ii) Ground truth concepts/factors of variation: concepts should preserve the amount of mutual information observed in ground truth concepts or factors of variation.
(iii) Downstream task (when available): concepts should capture sufficient statistics to predict the downstream task as well as the raw inputs do.

In this paper, we consider the properties (ii) and (iii) and make the following key contributions:

- We unify the language and notations across CL and DGL.
- We introduce metrics for evaluating the quality of learnt concepts/codes w.r.t. the ground truth ones, both for when concepts are correlated and also when ground truth concepts are unavailable.
- We use our metrics to conduct a systematic empirical comparison of state-of-the-art methods from three families of methods: supervised CL, unsupervised CL, and semi-supervised DGL.
- We make available the code used for our metrics, methods, and datasets as an open-source library.[1]

2 Background and Related Work

Notation In both CL and DGL the aim is to find a low-dimensional intermediate representation $\hat{\mathbf{c}}$ that explains the downstream task(s) in CL, or the data's factors of variation in DGL. In CL, this low-dimensional representation corresponds to a matrix $\hat{\mathbf{c}} \in \hat{C} \subseteq \mathbb{R}^{d \times k}$ in which the i-th column constitutes a d-dimensional representation of the i-th concept. As zero-padding can be used to ensure equal length across different concept representations, for notational simplicity we assume that all concepts use a d-dimensional vector as their representation. Under this view, elements in $\hat{\mathbf{c}}_{(:,i)} \in \mathbb{R}^d$ are expected to have high values (under some reasonable aggregation function) if the i-th concept is considered to be activated for the input that generated this representation. For example, in the case where $d = T$ and each concept can take up to T discrete values, $\hat{\mathbf{c}}_{(:,i)} \in [0, 1]^T$ can represent a probability distribution over all values that the i-th concept can take. As most CL methods assume $d = 1$, for succinctness we use $\hat{c}_i$ in place of $\hat{\mathbf{c}}_{(:,i)}$ when $d = 1$.

We adopt the same representation for latent code $\hat{\mathbf{z}}$ in DGL: $\hat{\mathbf{z}} \in \hat{Z} \subseteq \mathbb{R}^{d \times k}$, where each dimension of the latent code (or a non-overlapping subset of dimensions) encodes one, and only one, independent factor of variation $\mathbf{z}$ (where we assume all ground truth factors of variations $\mathbf{z}$ can be factored as $p(\mathbf{z}) = \prod_i p(z_i)$) (Locatello et al., 2020a). Nevertheless, note that in practice, d tends to be 1 for most DGL methods. Finally, for simplicity, we use $\hat{\mathbf{c}}$ for the representation of both concepts and latent codes. Ground truth concepts and factors of variations are referred to as $\mathbf{c} \in C \subseteq \mathbb{R}^k$.

In line with (Koh et al., 2020; Kazhdan et al., 2020; Yeh et al., 2020) we make use of: (i) a concept encoder function $g : \mathcal{X}' \mapsto \hat{C}$ that maps a transformation of the inputs $\mathbf{x} \in X \subseteq \mathbb{R}^m$, as performed by a function $\phi : X \mapsto \mathcal{X}'$, to a concept intermediate representation; and (ii) a label predictor function $f : \hat{C} \mapsto Y$ that maps the concept representations to a downstream task's set of labels $\mathbf{y} \in Y \subseteq \mathbb{R}^L$. These two functions can be combined to give a set of predictions for sample $\mathbf{x} \in X$ by computing $f(g(\phi(\mathbf{x})))$. In DGL autoencoders, one can think of $(g \circ \phi)(\cdot)$ as the autoencoder's encoder model and of the autoencoder's decoder model as a function that approximates $(g \circ \phi)^{-1}(\cdot)$.

Supervised concept learning In supervised CL, access to concept labels $\mathbf{c}^{(i)} \in \mathbb{N}^k$, in addition to target labels $\mathbf{y}^{(i)} \in \mathbb{R}^L$, is assumed for inputs $\mathbf{x}^{(i)} \in \mathbb{R}^m$. In other words, we have training data $\{(\mathbf{x}^{(i)}, \mathbf{c}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N$, where N is the number of training samples. In its most common form, supervised CL divides the prediction into two distinct steps of: (i) mapping an input sample to its concept representation via a concept encoder g; and (ii) mapping a sample's concept representation to its task labels via a label predictor $f(\cdot)$. Together, these two functions constitute a *Concept Bottleneck Model* (CBM) (Koh et al., 2020), because their final prediction relies on the input going through the bottleneck $g(\phi(\mathbf{x}))$, which is trained to be aligned with $\mathbf{c}$, component-wise.

[1] Code will be released after paper review.

Concept-based Model Extraction (CME) (Kazhdan et al., 2020) constructs a CBM from a pre-trained model by building a non-trivial $\phi(\cdot)$ mapping function using the model's latent space. Using such a latent representation instead of raw inputs typically makes CME more data efficient than CBM (Kazhdan et al., 2021). Similarly, *Concept Whitening* (CW) (Chen et al., 2020) constructs a CBM by introducing a pluggable batch normalization module whose activations $(g \circ \phi)(\cdot)$ are trained to be aligned with representative sets of binary concepts. It achieves this by forcing different feature maps of the normalization module to be decorrelated and orthogonal while incentivizing activations in a given axis to be high when its corresponding pre-defined concept is activated.

Unsupervised Concept Learning Unlike supervised CL, in unsupervised CL concept annotations are not available and concepts are discovered in an unsupervised manner. Ghorbani et al. (2019) extract concepts from a trained classifier for image data. Images belonging to each class are first segmented with multiple resolutions. The segments are then clustered as examples of class concepts and their importance scores are measured using TCAV (Kim et al., 2018).

Unlike Ghorbani et al. (2019), *Completeness-aware Concept Discovery* (CCD) (Yeh et al., 2020) is data modality agnostic and extracts class-independent concepts. CCD builds on TCAV (Kim et al., 2018) to first extract a set of concept vectors $\{\mathbf{h}^{(i)}\}_{i=1}^{k}$, each of which is a unit vector representing a direction in $\mathcal{X}'$. These vectors are then used to construct a 1D concept representation $g(\phi(\mathbf{x})) = \hat{\mathbf{c}} \in \hat{C} \subseteq \mathbb{R}^k$ for a sample $\mathbf{x}$ such that $\hat{c}_i$ is set to $TH(\langle \phi(\mathbf{x}), \mathbf{h}^{(i)} \rangle, \beta)$, the β-thresholded inner product of $\phi(\mathbf{x})$ and the i-th concept vector $\mathbf{h}^{(i)}$. For a complete set of concepts, a concept representation should contain sufficient statistics to obtain high performance in the original model's prediction task $f(\phi(\mathbf{x}))$. If this is the case, then there must exist a mapping $\psi : \hat{C} \mapsto \mathcal{X}'$ that recovers $\phi(\mathbf{x})$ from $g(\phi(\mathbf{x}))$. Thus, CCD constructs a set of concept vectors such that $f(\psi(g(\phi(\mathbf{x}))))$ is able to achieve a similar task performance to that of $f(\phi(\mathbf{x}))$. Notice that although $f(\cdot)$ is still a label predictor function here, it is not applied to the raw inputs or to their concept representation.

Disentanglement Learning Generative models (e.g., VAEs (Kingma & Welling, 2014; Higgins et al., 2017)) used in DGL assume that data is generated from a set of independent factors of variation $\mathbf{c} \in C$, sampled from factorable distribution $p(\mathbf{c}) = \prod_i \; p(c_i)$, such that a sample $\mathbf{x}$ is generated according to the conditional distribution $p(\mathbf{x}|\mathbf{c})$. Thus, the goal of DGL is to find a function $g(\cdot)$ that maps inputs to a disentangled latent representation, such that a subset of non-overlapping dimensions of the latent representation corresponds to a unique factor of variation c_i. We consider factors of variation as ground truth concepts and the latent code as a set of learnt high-level concepts.

In the light of recent work showing the theoretical impossibility of learning disentangled representations in an unsupervised manner (Locatello et al., 2019), we focus here on weakly supervised disentanglement, where weak supervision provides the inductive bias required to learn disentangled representations that capture the different factors of variation. Specifically, we focus on the families of models where weak supervision comes via pairs of observations whose corresponding ground truth factors of variation share at least one common element (Locatello et al., 2020a).

3 Quality Assurance of Concepts

Given the varying degree of supervision in the approaches above, the quality of their learnt concepts is expected to vary. The quality of learnt concepts w.r.t. downstream tasks has commonly been studied using predictive performance metrics such as accuracy and AUC. In contrast, there exist several metrics for measuring the properties of learnt concepts w.r.t. the ground truth ones, some of which focus on the *overall set of learnt concepts* and some on each *individual concepts* (see Appendix 6.1 for existing metrics). Nevertheless, these metrics have two main shortcomings.

The first main shortcoming is assuming that concepts are independent, which implies that if a learnt concept encodes information beyond the ground truth concept it is aligned with, it counts as *leakage* (Mahinpei et al., 2021). Impurities caused by leakage can have far-reaching consequences in terms of human interventions: perturbations of an impure concept may impact other concepts unintentionally. While we agree with the danger of impurities, we argue that concepts in the real-world often have dependencies, and full disentanglement is not realistic. As such, leakage as described above is not necessarily undesirable when it reflects the correlation between concepts in the real-world scenarios. To support this claim, we demonstrate the lack of inter-concept independence in a real-world dataset, the Caltech-UCSD Birds-200-2011 (CUB) (Wah et al., 2011) in Appendix 6.2.

The second main shortcoming is the assumption that one has access to ground truth concepts, which are used to verify the properties of the extracted ones. Previous work has shown that access to ground truth concepts is limited given the difficulty of annotating samples with concept labels (Raghavan et al., 2006). The rarity of real-world datasets with such annotations is evidence for this claim.

In order to address these shortcomings, we first propose the *oracle impurity score*, a metric that measures the quality of concept representations when one has access to ground truth concepts labels without requiring ground truth concepts to be independent. Moreover, this metric does not require concept annotations to be binary. We then propose the *niching purity score* and *niching impurity score*, two efficient metrics which, in conjunction, are able measure the quality of concept representations when one has access to a specific task's labels. Both of these latter metrics do not require concept annotations in the training data and do not assume independence across concepts.

3.1 Oracle Impurity

In order to circumvent the need for inter-concept independence, we take inspiration from (Mahinpei et al., 2021), where they informally measure concept impurity by looking at how predictive a CBM-generated concept probability is for the ground truth value of other concepts used to train the CBM. If the pre-defined concepts are independent, then the inter-concept predictive performance should be no better than random. To generalise this assumption beyond independent concepts, we first measure the predictability of ground truth concepts w.r.t. one another. Then we measure the predictability of learnt concepts w.r.t. the ground truth ones. The divergence between the former and the latter acts as an impurity metric, measuring the amount of undesired information that is encoded by the learnt concepts. In order to formally introduce our metric, we begin by formalising a *purity matrix*.

Definition 1 (Purity Matrix)**.** *Assume we are given concept representations $\hat{\Gamma} = \{\hat{\mathbf{c}}^{(i)} \in \mathbb{R}^{d\times k}\}_{i=1}^{n}$ and a corresponding set of discrete ground truth concept annotations $\Gamma = \{\mathbf{c}^{(i)} \in \mathbb{N}^{k}\}_{i=1}^{n}$. Furthermore, assume that $\hat{\Gamma}$ and Γ are aligned element-wise: for all $l \in \{1, \cdots, k\}$, the l-th concept representation of $\hat{\mathbf{c}}^{(i)}$ is aligned with the l-th concept label in $\mathbf{c}^{(i)}$. We define the purity matrix of $\hat{\Gamma}$ given ground truth labels Γ as a matrix $\pi\big(\hat{\Gamma}, \Gamma\big) \in [0, 1]^{k\times k}$ whose entries are given by:*

$$\pi\big(\hat{\Gamma}, \Gamma\big)_{(i,j)} := \sup_{\psi_j} \mathbb{P}_{(\hat{\mathbf{c}},c)\sim\mathcal{E}(\hat{\Gamma},i,\Gamma,j)}\big(c = \arg\max_{l} \psi_j(\hat{\mathbf{c}})_l\big)$$

where $\mathcal{E}(\hat{\Gamma}, i, \Gamma, j)$ is defined as the empirical distribution over concept representations for the i-th concept, and ground truth concept labels for the j-th concept $\{(\hat{\mathbf{c}}^{(1)}_{(:,i)}, c^{(1)}_{j}), \cdots, (\hat{\mathbf{c}}^{(n)}_{(:,i)}, c^{(n)}_{j})\}$, and $\psi_j(\cdot)$ as a family of functions mapping a single concept's d-dimensional vector representation to a probability distribution over all the values that a concept j may take.

The (i, j)-th entry of $\pi\big(\hat{\Gamma}, \Gamma\big)$ contains the highest AUC when predicting the ground truth value of concept j given the i-th concept representation. Intuitively, the diagonal entries of this matrix show how good a concept representation is at predicting its corresponding ground truth value, while the off-diagonals show how good such a representation is at predicting the ground truth labels of other concepts. In principle, metrics other than AUC can similarly be used as a proxy for the mutual information between concept representations and ground truth labels. In practice, we parameterise the family of functions ψ_j by training, via gradient descent, a ReLU MLP with 32 hidden activations.

The definition of a purity matrix allows us to construct a metric for quantifying the amount of unnecessary information, or impurity, that a concept encoder encodes.

Definition 2 (Oracle Impurity Score (OIS))**.** *Assume a given concept encoder $g : X \mapsto \hat{C}$ and a set of ground truth concept-annotated test samples $\{(\mathbf{x}^{(i)}, \mathbf{c}^{(i)})\}_{i=1}^{n}$, such that $\mathbf{x}^{(i)} \in X$, $\mathbf{c}^{(i)} \in \mathbb{N}^{k}$, and $\hat{C} \subseteq \mathbb{R}^{d\times k}$. Abusing notation, let $g(A) := \{g(a) \mid a \in A\}$ be defined as the application of function g over all elements in the ordered set A. Furthermore, let $\Gamma_X := \{\mathbf{x}^{(i)}\}_{i=0}^{n}$ and $\Gamma := \{\mathbf{c}^{(i)}\}_{i=0}^{n}$ be ordered sets of testing samples and their corresponding concept annotations, respectively. The OIS of concept encoder $g(\cdot)$ given concept-annotated testing data (Γ_X, Γ) is then defined as:*

$$\textit{OIS}(g, \Gamma_X, \Gamma) := \frac{2\Big|\Big|\pi\big(g(\Gamma_X), \Gamma\big) - \pi\big(\Gamma, \Gamma\big)\Big|\Big|_F}{k}$$

where we use $||\mathbf{A}||_F$ to represent the Frobenius norm of matrix $\mathbf{A}$.

Since each entry in the difference $\Big(\pi\big(g(\Gamma_X), \Gamma\big) - \pi\big(\Gamma, \Gamma\big)\Big)$ can be at most $1/2$, the upper bound of the Frobenius norm of this difference is $k/2$. Therefore, our metric includes a normalisation factor $2/k$ which guarantees that it is in $[0, 1]$. This allows interpreting an OIS of 1 as a complete misalignment between $\pi\big(\Gamma, \Gamma\big)$ and $\pi\big(g(\Gamma_X), \Gamma\big)$ (i.e., the i-th concept representation can predict all other concept labels except its corresponding one even when concepts are independent). An impurity score of 0, on the other hand, represents perfect alignment between the two purity matrices (i.e., the i-th concept representation does not encode any unnecessary information for predicting concept i).

3.2 Niching-based Metrics

In nature, a niche is defined as a resource-constrained subspace of the environment that can support different types of life (Darwin, 1859). By analogy, researchers in Evolutionary Computation have developed a variety of niching-inspired methods (Goldberg et al., 1987; Deb & Goldberg, 1989; Holland et al., 1992) that construct solutions by promoting diversity inside the evolved population and avoiding the over-exploitation of some regions in the search space (Squillero & Tonda, 2016). In the context of neural networks, each layer can be seen as a computational "environment" with a finite amount of computational resources (i.e., the layer parameters). A "niche of neurons" can be defined as a subspace of the layer (i.e., the "environment") that is critical for the accurate prediction (i.e., "the survival") of a specific downstream task (i.e., the "species living in the niche").

Inspired by the theory of niching, we propose two metrics that measure the quality of concept representations in the *absence* of access to ground truth concept labels but in the presence of task-specific labels, by essentially quantifying how task-label-separable the concept representation in a bottleneck is. The overall idea of these metrics is to first identify a set of concepts that are important for a classification label, and then see if such a set, referred to as a *concept niche* for the label, is complete enough to predict the label. On the other hand, the predictive performance of a set of concepts outside the concept niche of a label identifies impurity and unnecessary leakage in the concept representation. We start by describing a *concept nicher*, a function that defines the dependency between a single concept and a single classification label. For this, we use $\hat{\mathbf{c}}^{(i)} = g(\phi(\mathbf{x}^{(i)}))$ to describe the concept representation of sample $\mathbf{x}^{(i)}$.

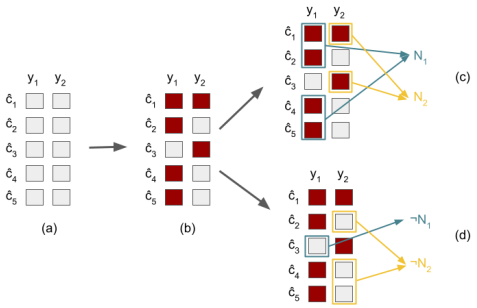


Figure 1: Concept niches and their complement: (a) having a set of concepts and a set of classification labels, (b) a concept nicher identifies the concepts on which a label depends strongly (red blocks); thus (c) a concept niche is the set of these concepts that a label depends on strongly; and (d) a complement concept niche is the set of concepts that a label does not depend on strongly.

Definition 3 (Concept nicher)**.** *Given a label predictor function $f : \hat{C} \mapsto Y$, where $\hat{C} \subseteq \mathbb{R}^{d \times k}$ and $Y \subseteq \mathbb{R}^L$, we define a concept nicher as a Boolean-valued function $\nu_f : \{1, \cdots, k\} \times \{1, \cdots, L\} \mapsto \{0, 1\}$ that returns $\nu_f(i, j) = 1$ if the j-th output dimension of $f(\cdot)$ has a "strong" dependency, as determined by a reasonable procedure (see an example below), on the i-th concept representation $\hat{\mathbf{c}}_{(:,i)}$ and 0 otherwise.*

The above definition can be instantiated in various ways depending on how the dependency is defined. One efficient way of measuring the dependency, and the way that we proceed, is to use the absolute Pearson correlation, denoted as ρ. We call such an instantiation a *concept-correlation nicher* (CCorrN) and define it as:

$$\text{CCorrN}_f(i, j) := \Big|\rho(\{\hat{\mathbf{c}}^{(1)}_{(:,i)}, \cdots, \hat{\mathbf{c}}^{(N)}_{(:,i)}\}, \{f(\hat{\mathbf{c}}^{(1)})_j, \cdots, f(\hat{\mathbf{c}}^{(N)})_j\})\Big| > \beta$$

where $\beta \in [0, 1]$ is a user-defined threshold. If $\hat{\mathbf{c}}_{(:,i)}$ is not a scalar representation (i.e., $d > 1$), we use the maximum absolute correlation coefficient between all entries in $\hat{\mathbf{c}}_{(:,i)}$ and the target label $f(\mathbf{c})_j$ as a representative correlation coefficient for the entire representation $\hat{\mathbf{c}}_{(:,i)}$. For all experiments we use $\beta = 0.2$ as a threshold. The impact of β on CCorrN is discussed in Appendix 6.3.

For each classification label, the *concept niche* defines the set of concepts on which the label depends (i.e., where the concept nicher outputs 1 for the label). One can intuitively see this in Figure 1(b),

where the red blocks show the concepts that the concept nicher has identified for y_1 and y_2 and Figure 1(c) shows the niches. Formally we can define a concept niche as:

Definition 4 (Concept niche)**.** *The concept niche $N_j(\nu_f)$ for label j, determined by label predictor $f(\cdot)$ and concept nicher $\nu_f(\cdot)$, is defined as $N_j(\nu_f) := \big\{i \ \ |i \in \{1, \cdots, k\}$ and $\nu_f(i, j) = 1\big\}$.*

Abusing notation, we let $\neg N_j(\nu_f)$ be the complement of the set $N_j(\nu_f)$, that is, $\neg N_j(\nu_f) := \{1, \cdots, k\} \setminus N_j(\nu_f)$. Complement of concept niches $N_1(\nu_f)$ and $N_2(\nu_f)$ are presented in Figure 1(d). Also we note that the label predictor $f(\cdot)$, can be any classifier. In our experiments we used a ReLU MLP with hidden layer sizes $\{20, 20\}$. Thus, we define $f|_{N_j(\nu_f)}$ to be the MLP resulting from pruning all connections in $f(\cdot)$ except those between concept representations in the niche $N_j(\nu_f)$ and the j-th output of model $f(\cdot)$.

We define *Niche Purity Score* (NPS) and *Niche Impurity Score* (NIS) as measures of the predictive capacity of a niche and its complement w.r.t. a specific classification label for a given label predictor function, respectively. A good concept representation has a simultaneous high NPS and low NIS for each label, as the latter is a symptom of unnecessary leakage in concept representation.

Definition 5 (Niche Purity Score)**.** *Given label predictor $f : \hat{C} \mapsto Y$, concept nicher ν_f, concept representations $\hat{\Gamma} = \{\hat{\mathbf{c}}^{(j)}\}_{j=1}^n$, and corresponding ground truth classification labels $\Gamma_Y = \{\mathbf{y}^{(j)}\}_{j=1}^n$, the NPS of the i-th output of $f(\cdot)$ is defined as:*

$$\textit{NPS}_i(f, \nu_f) := \mathbb{P}_{(\hat{\mathbf{c}}', y) \sim \mathcal{E}(\hat{\Gamma}, N_i(\nu_f), \Gamma_Y, i)}\big(y = \arg\max_l f|_{N_i(\nu_f)}(\hat{\mathbf{c}}')_l\big)$$

where $\mathcal{E}(\hat{\Gamma}, N_i(\nu_f), \Gamma_Y, i)$ is the empirical distribution over concept representations in niche $N_i(\nu_f)$ and their corresponding i-th labels $\{(\hat{\mathbf{c}}^{(j)}_{(:, N_i(\nu_f))}, y_i^{(j)})\}_{j=1}^n$. The overall NPS of $f(\cdot)$, denoted by NPS(f, ν_f), is the average NPS across all labels. That is, NPS$(f, \nu_f) := \sum_{i=1}^L$ NPS$_i(f, \nu_f)/L$.

Definition 6 (Niche Impurity Score)**.** *The NIS of concept niche $N_j(\nu_f)$ is the NPS of $\neg N_j(\nu_f)$.*

Essentially, the NPS metric measures the quality of niche $N_i(\nu_f)$ via the AUC of predicting the ground truth label of the i-th label using *only* the concept representations corresponding to niche $N_i(\nu_f)$. A NPS score of 1 (its maximum value), thus indicates a perfect niche purity for the i-th output of function $f(\cdot)$ while a worst-case-scenario NPS of $1/2$ indicates that a niche is unable to do better than random when predicting its corresponding label. Similarly, the NIS metric measures the impurity of niche $N_i(\nu_f)$ by measuring the AUC of predicting the ground truth label of the i-th label using concept representations *outside* the niche corresponding to the i-th label. Therefore, a NIS score of $1/2$ indicates the best-case scenario impurity as concepts outside of niche $N_i(\nu_f)$ hold no useful information to predict the i-th label. In contrast, a worst-case NIS score 1 corresponds to complete impurity as the i-th label can be perfectly predicted from concepts that are irrelevant to it.

4 Experiments

4.1 Methods, Model Selection, and Training

We compare the quality of concept representations in various methods using our metrics. CBM (Koh et al., 2020) is selected from the supervised CL family due to its fundamental role in CL. We focus on jointly trained CBMs, where the task-specific loss and the concept prediction loss are minimised jointly. We also add CW (Chen et al., 2020) from this family as it directly treats decorrelated concept representations. From the unsupervised CL family, CCD (Yeh et al., 2020) is selected due to its data agnostic nature. From the DGL family, we consider two weakly supervised methods, Adaptive Group Variational Autoencoder (Ada-GVAE) and Adaptive Multilevel Variational Autoencoder (Ada-MLVAE) (Locatello et al., 2020a). For each method and metric, we report the average metric values and 95% confidence intervals obtained from 5 different random seeds. Given that the number of concepts in unsupervised CL and DGL approaches is not known upfront, we allow an extra variation, where the number of learnt concepts is twice the number of ground truth ones.

We observe that model capacity impacts the quality of learnt concepts (see Appendix 6.4). To minimise the effects of inductive biases arising from the model architecture and training hyperparameters, we use the same architecture and training setups for all methods within the same dataset whenever possible. We include details on training and architecture hyperparameters in Appendix 6.5.2.

4.2 DATASETS

In order to have datasets that are compatible with both CL and DGL, we construct datasets whose samples are fully described by a vector of ground truth generative factors. Moreover, to simulate real-world scenarios, we work with tasks with a varying degree of dependencies in their concept annotations. To achieve this, we first design a parametric binary-class dataset *TabularToy*(δ), a variation of the tabular dataset proposed by (Mahinpei et al., 2021). We also construct two multiclass image-based parametric datasets: *dSprites*(λ) and *3dshapes*(λ), based on dSprites (Matthey et al., 2017) and 3dshapes (Burgess & Kim, 2018) datasets, respectively. They consist of 3D samples generated from a vector consisting of $k = 5$ and $k = 6$ ground truth factors of variation, respectively. Both datasets have one binary concept annotation per factor of variation. Parameters $\delta \in [0, 1]$ and $\lambda \in \{0, \cdots, k-1\}$ control the degree of concept inter-dependencies during generation (a value of 0 represents inter-concept independence while higher values represent stronger inter-concept dependencies). Details on how dependencies are introduced, as well as the nature of the individual concepts and task labels for all our datasets, can be found in the Appendix 6.5.1.

4.3 RESULTS

Overall set of learnt concepts are equally predictive of the downstream task across all surveyed methods. The quality of the overall set of learnt concepts w.r.t. downstream tasks, measured by the predictive AUCs does not vary considerably across methods and stays close to 100%. This result replicates the predictive power of the raw inputs and thus shows that the grouped concept representations faithfully capture task-related information in the input space. Plots for predictive AUC of raw inputs and the overall set of learnt concepts can be found in Appendix 6.6.

Supervision leads to overall set of learnt concepts that are a better predictors of individual ground truth concepts compared with no supervision (Figure 2). Since CBM and CW benefit from explicit supervision (i.e., concept supervision), they encode representations that can predict the ground truth ones very well. However, while this observation holds in CW when one uses the entire feature maps CW produces as concept representations (shown as "CW Feature Map" in plots), it does not hold for "CW MaxPool-Mean", where one reduces the CW module's feature maps to scalar concept scores using a max pool followed by a mean reduction (as suggested by Yeh et al. (2020)). This is in alignment with results reported in Mahinpei et al. (2021), and suggests that there may be substantial information loss in CW's concept representations when they are reduced to a scalar form. Note that this is not observed in ToyTabular(δ) as its concept representations are already scalars. The implicit supervision (i.e., supervision from downstream task as opposed to explicit concept supervision) in CCD is just as effective as explicit concept supervision in other methods in terms of the predictive power of the overall set of learnt concepts for individual ground truth ones. The relatively poor performance of the DGL approaches suggests that either concept or task supervision is required to secure a good predictive power for learnt concepts.

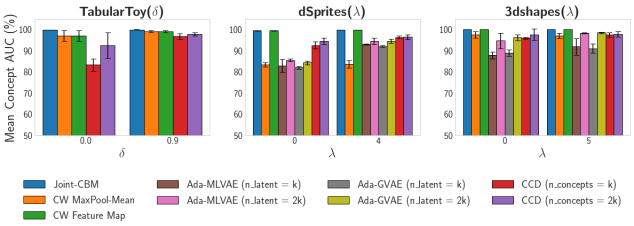
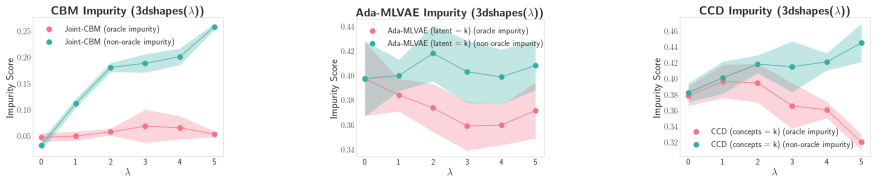


Figure 2: Average AUC when predicting each ground truth concept from the overall set of learnt concepts, by training ReLU MLPs with hidden layers $\{64, 64\}$.

Metrics that assume concept independence may be misleading (Figure 3). The comparisons above focused on the quality of overall set of learnt concepts, without an indication of the quality of individual learnt concepts w.r.t. the ground truth ones. The alignment between the learnt and ground truth concepts is clear in CBM and CW. In CCD and DGL, where there is no alignment, we compute a greedy alignment between the learnt and ground truth concepts based on the predictive AUC of using a learnt concept to predict each ground truth concept (i.e., a ground truth concept is assumed to be represented by the learnt one that can predict it best).

After the alignment, we investigate the quality of individual learnt concepts w.r.t. the ground truth ones. First, we show the inadequacy of assuming concept independence compared to our oracle impurity metric proposed in Definition 2. We achieve this by comparing our metric against a variant of itself, labeled as "non-oracle impurity", that assumes the oracle matrix of any dataset is always a matrix with 1 in its diagonals and $1/2$ in its off-diagonals. We observe that the non-oracle impurity misleadingly shows more impurity as the degree of inter-concept dependencies increases. This is due to the fact that our metric, in contrast to the non-oracle impurity, takes into account the dataset's inherent ground truth inter-concept dependencies as a baseline for what a method's purity matrix should look like. We observe the same results for other methods (see Appendix 6.7).



(a) CBM (b) Ada-MLVAE (c) CCD

Figure 3: Impurity scores (both oracle and non-oracle) in *3dshapes*(λ).

Oracle impurity demonstrates that explicit supervision encodes purer individual concepts than implicit or no supervision (Figure 4). The oracle impurity metric results demonstrate that CBM's individual learnt concepts consistently experience the least amount of impurity due to receiving explicit supervision. In contrast, CW's full feature maps concepts encode the highest amount of impurity on average, a phenomenon also observed by Mahinpei et al. (2021). This is due to the fact that the learnt concepts are highly predictive of every ground truth one irrespective of their alignment. Thus, while explicit supervision translates to better predictive performance, it does not translate to purer concepts.

While the overall set of learnt concepts in CCD performed as well as those learnt via explicit concept supervision in CBM and CW, the individual concepts do not correspond well to the ground truth ones. This indicates that the information about each ground truth concept is distributed across the overall representation rather than localized in individual concepts. As a result, in absence of extra capacity, CCD and DGL approaches encode impurities within a similar range and more that those of approaches benefiting from explicit supervision.

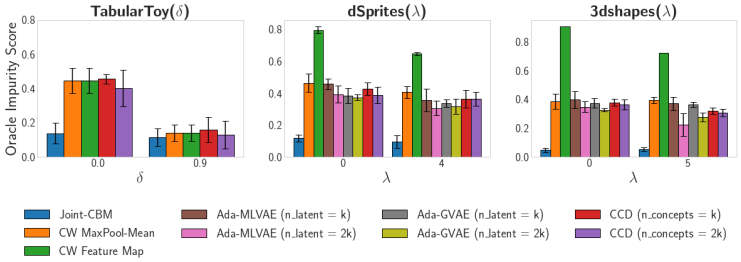


Figure 4: Oracle impurity scores for all datasets.

Niching-based metrics demonstrate that supervision results in task-label-separable concept representations compared to no supervision (Figure 5). Given that niching-based metrics rely on a proportion of concepts that are related to each task, as opposed to individual concepts, they do not differentiate between explicit and implicit supervision as much as oracle impurity. Thus, simultaneous low niche impurity and high niche purity is observed in CBM and CCD, despite the different nature of supervision they receive. Low purity of CCD in 3dshapes can be explained by "concept loudness" observed by Kazhdan et al., where some concept take precedence over others during training, which impact the overall concept representation.

In contrast to CBM and CCD, high impurity and high purity are observed in CW's full feature maps (i.e., when no scalar reduction is applied), which indicates that CW's feature maps are highly predictive of all concepts, aligned or not. This is particularly true when concept dependencies increase,

which shows the vulnerability of this approach when concepts have inter-dependencies. Performing a MaxPool-Mean reduction in CW's maps reduces the amount of impurities, however, it also reduces the amount of purity (i.e., the learnt concepts are unable to predict their aligned ones with high confidence). Similar to oracle impurity, this is further evidence that some crucial information may be lost in CW's scalar reduction. Finally, as shown by our oracle impurity metric, DGL methods encode high impurities. Such high impurity is paired with relative low purity, indicating that the ground truth concepts are not captured well in the learnt ones.

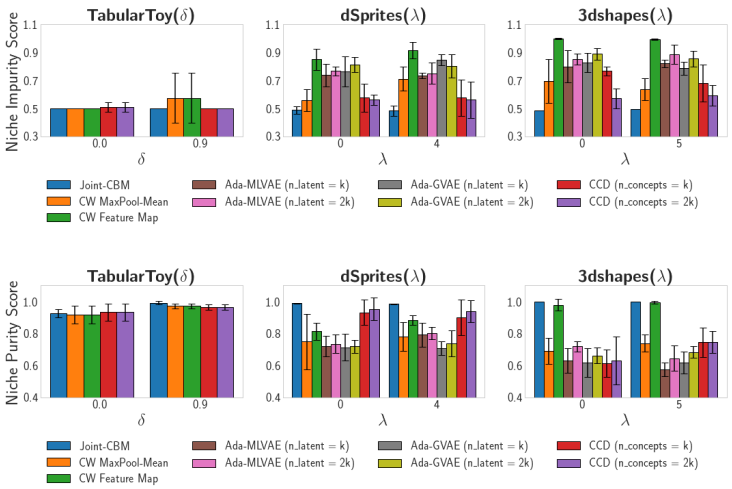


Figure 5: Niche impurity and purity scores for all datasets.

Oracle impurity and niching-based metrics are robust to concept inter-dependencies. The highly preserved ranking of the methods using our metrics in both settings with and without dependencies, indicates the robustness of these metrics to concept inter-dependencies as intended.

5 Discussion and Conclusion

The lack of metrics to evaluate and contrast the quality of concept representations within and between CL and DGL methods hinders their systematic comparison. Consequently, the only existing comparison that we are aware of relies on overall properties such as data efficiency and sensitivity to variance in concept values (Kazhdan et al., 2021). In this paper, we introduced metrics that facilitate comparison from concept representation quality perspective.

Our systematic evaluation of several methods suggests the following practical guidelines: (i) In settings where ensuring the quality of the overall set of learnt concepts is sufficient, implicit supervision from task labels is just as effective as explicit supervision from concept labels. Thus, given the cost, providing explicit supervision is not recommended; (ii) In settings where ensuring the quality of individual concept representation as well as their overall set is required, explicit supervision is recommended; and (iii) In either of the above settings the weak supervision provided by DGL methods is not recommended as a substitute for implicit or explicit supervision. Related to metrics, in setting (i), regardless of access to concept labels, niching-based metrics should be favoured to oracle impurity. Both metrics indicate that the overall set of learnt concept benefits from supervision, while the former is very efficient and does not have the limitation of becoming intractable when the number of concepts is high. On the contrary, in setting (ii), the high computational cost of oracle impurity is justified if concept labels are available. Unlike, niching-based metrics, oracle impurity distinguishes the impact of explicit vs. implicit supervision on the quality of individual learnt concepts.

One angle that was not explored in this paper is the evaluation of the quality of concept representations w.r.t. the input space. In fact, Margeloiu et al. (2021) showed that concepts from CBM often do not correspond to semantically meaningful input space. We therefore believe that future work on defining metrics that capture concept-input quality along with the metrics we proposed here can provide a more comprehensive assurance about the quality of learnt concept representations.

REFERENCES

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Umang Bhatt, Alice Xiang, Shubham Sharma, Adrian Weller, Ankur Taly, Yunhan Jia, Joydeep Ghosh, Ruchir Puri, José MF Moura, and Peter Eckersley. Explainable machine learning in deployment. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pp. 648–657, 2020.

Chris Burgess and Hyunjik Kim. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

Ricky TQ Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. *arXiv preprint arXiv:1802.04942*, 2018.

Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782, 2020.

Gabriele Ciravegna, Pietro Barbiero, Francesco Giannini, Marco Gori, Pietro Lió, Marco Maggini, and Stefano Melacci. Logic explained networks. *CoRR*, abs/2108.05149, 2021.

Elliot Creager, David Madras, Jörn-Henrik Jacobsen, Marissa A. Weis, Kevin Swersky, Toniann Pitassi, and Richard S. Zemel. Flexibly fair representation learning by disentanglement. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1436–1445. PMLR, 2019.

Charles Darwin. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. 1859.

Kalyanmoy Deb and David E Goldberg. An investigation of niche and species formation in genetic function optimization. In *Proceedings of the third international conference on Genetic algorithms*, pp. 42–50, 1989.

Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2018.

Amirata Ghorbani, James Wexler, James Y. Zou, and Been Kim. Towards automatic concept-based explanations. In *Neural Information Processing Systems (NeurIPS)*, pp. 9273–9282, 2019.

David E Goldberg, Jon Richardson, et al. Genetic algorithms with sharing for multimodal function optimization. In *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*, pp. 41–49. Hillsdale, NJ: Lawrence Erlbaum, 1987.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2017.

John Henry Holland et al. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

Dmitry Kazhdan, Botty Dimanov, Mateja Jamnik, Pietro Liò, and Adrian Weller. Now you see me (CME): concept-based model extraction. In *Conference on Information and Knowledge Management (CIKM) Workshops*, volume 2699 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

Dmitry Kazhdan, Botty Dimanov, Helena Andrés-Terré, Mateja Jamnik, Pietro Liò, and Adrian Weller. Is disentanglement all you need? comparing concept-based & disentanglement approaches. *CoRR*, abs/2104.06917, 2021.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2673–2682. PMLR, 2018.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In Jennifer G. Dy and Andreas Krause (eds.), *International Conference on Machine Learning, (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2654–2663. PMLR, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun (eds.), *International Conference on Learning Representations (ICLR)*, 2014.

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 (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5338–5348. PMLR, 2020.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4114–4124. PMLR, 2019.

Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 6348–6359. PMLR, 2020a.

Francesco Locatello, Michael Tschannen, Stefan Bauer, Gunnar Rätsch, Bernhard Schölkopf, and Olivier Bachem. Disentangling factors of variations using few labels. In *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2020b.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*, pp. 4765–4774, 2017.

Lucie Charlotte Magister, Dmitry Kazhdan, Vikash Singh, and Pietro Liò. Gcexplainer: Human-in-the-loop concept-based explanations for graph neural networks. *CoRR*, abs/2107.11889, 2021.

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

Andrei Margeloiu, Matthew Ashman, Umang Bhatt, Yanzhi Chen, Mateja Jamnik, and Adrian Weller. Do concept bottleneck models learn as intended? *CoRR*, abs/2105.04289, 2021.

Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset, 2017.

Hema Raghavan, Omid Madani, and Rosie Jones. Active learning with feedback on features and instances. *Journal of Machine Learning Research*, 7:1655–1686, 2006.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *International Conference on data science and advanced analytics (DSAA)*, pp. 1135–1144. ACM, 2016.

Karl Ridgeway and Michael C. Mozer. Learning deep disentangled embeddings with the f-statistic loss. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 185–194, 2018.

Giovanni Squillero and Alberto Tonda. Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization. *Information Sciences*, 329:782–799, 2016.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

Chih-Kuan Yeh, Been Kim, Sercan Ömer Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. In *Neural Information Processing Systems (NeurIPS)*, 2020.

6 ADDITIONAL RESOURCES

6.1 METRICS RELATED TO PROPERTIES OF CONCEPT REPRESENTATIONS W.R.T. GROUND TRUTH CONCEPTS

Refer to Table 1 for a summary of some metrics used for measuring the quality of disentangled representations which are applicable to disentangled concept representations.

Table 1: Metrics related to properties of concepts w.r.t. ground truth concepts divided to two categories: those that capture properties of individual concepts and those that capture properties of the overall set of learnt concepts w.r.t. ground truth concepts. Although metrics in each column serve the same purpose, they are mathematically distinct.

Individual learnt concepts w.r.t. ground truth concepts	**Overall set of learnt concepts w.r.t. ground truth concepts**
Modularity (Ridgeway & Mozer, 2018): Whether each learnt concept corresponds to at most one ground truth one. Measured by deviation from ideal case where each learnt concept has high mutual information with only one ground truth one and zero with others.	Explicitness (Ridgeway & Mozer, 2018): Whether the overall set of learnt concepts can predict each individual ground truth ones using a simple (e.g., linear) classifier. Measured by the average predictive performance of concept vector.
Mutual information gap (Chen et al., 2018): Whether learnt concepts are disentangled. Measured by averaging the gap in mutual information between the two learnt concepts that have the highest mutual information with a ground truth concept. This metric generalises the disentanglement scores in Higgins et al. (2017) and Kim & Mnih (2018).	Informativeness (Eastwood & Williams, 2018): Whether the overall concept vector can predict each ground truth concept with low prediction error. Measured by average prediction error of concept vector.
Disentanglement (Eastwood & Williams, 2018): Whether each learnt concept captures at most one ground truth one. Measured by the weighted average of disentanglement degree of each learnt concept. Such degree is calculated based on entropy, where high entropy for a learnt concept shows its equal importance for all ground truth ones and therefore its low disentanglement degree. The weight is calculated based on the aggregation of relative importance of a learnt concept in predicting each ground truth one.	
Alignment (Yeh et al., 2020): Whether the learnt concepts match the ground truth ones. Measured by average accuracy of predicting each ground truth concept using the learnt one that predicts it best.	

6.2 MUTUAL INFORMATION IN CUB DATASET CONCEPTS

We take the Caltech-UCSD Birds-200-2011 (CUB) dataset (Wah et al., 2011), formed by images of 200 types of birds, and use each sample's 112 binary attributes as concept annotations (Koh et al., 2020). We find the top-10 concepts with highest mutual information with the label and compute their Pearson correlation coefficients. Our results, shown in Figure 6, highlight that many concepts have non-zero correlations. This shows the lack of inter-concept independence in real-world datasets.

6.3 IMPACT OF CCORRN'S β ON NICHE PURITY, IMPURITY, AND SIZE

Figure 7 shows the impact of threshold β used in concept-correlation nicher (CCorrN) on niche purity, impurity, and size in dSprites(λ = 0). Across all approaches, increasing β decreases niche

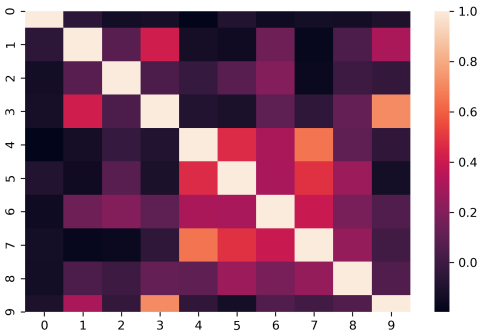
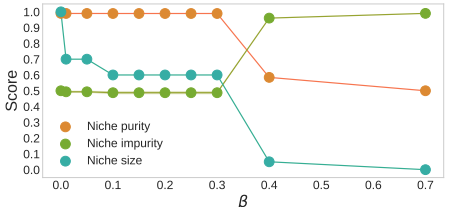
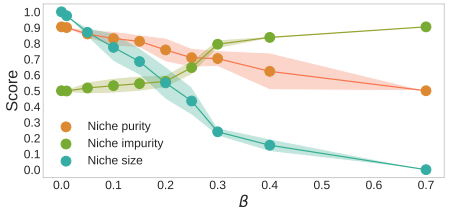


Figure 6: Absolute Pearson correlation coefficients of the top-10 concepts with highest label mutual information in CUB dataset. Notice that there exist strong correlations between some of the concepts.

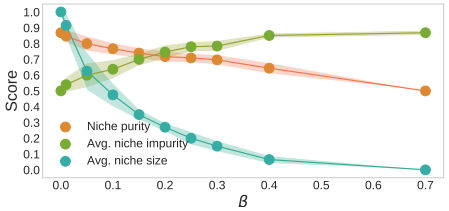
purity and niche size, while increasing niche impurity. When β is high, only very few concept can pass the threshold required to be identified as concept nicher for a label. As a result, niche sizes get smaller, which translates to bigger niche complement and, subsequently, more impurity.



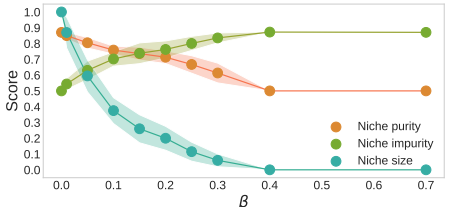
(a) Joint-CBM



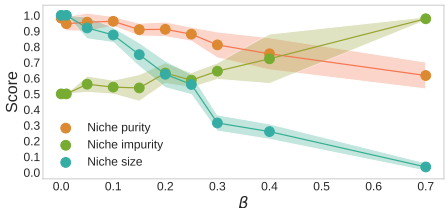
(b) CW MaxPool-Mean



(c) Ada-MLVAE (n_latent = k)



(d) Ada-GVAE (n_latent = k)



(e) CCD (n_concepts = k)

Figure 7: The impact of CCorrN's β on niche purity, impurity, and size in dSprites($\lambda = 0$).

6.4 Effect of Network Capacity on Oracle Impurity

We motivate the need of fixing the neural network architecture when contrasting different methods by looking at how the capacity of the selected architecture can affect a method's learn concept representations. For this, we train a Joint-CBM in the TabularToy($\delta = 0$) dataset (with $\alpha = 0.1$) whose encoder and decoder models are simple MLPs with one hidden layer. In this experiment, we vary

the number of activations in both the encoder and decoder models and monitor the predictive accuracy of concept representations w.r.t. their aligned ground truth concepts as well as representations' oracle impurities. As shown in Figure 8, we observe that as encoder and decoder capacity decreases, the CBM begins to exhibit significantly higher impurity and lower concept accuracy. This motivates the need for a constant architecture with enough capacity to achieve a high concept predictability and serves as a basis for our decision to fix the encoder and decoder architectures across all methods in our evaluation.

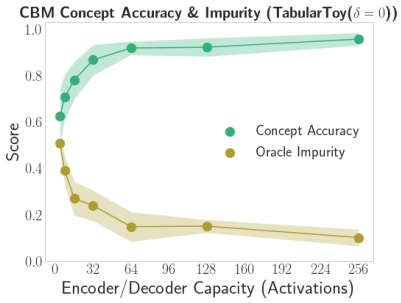


Figure 8: Effect of network capacity (i.e., number of hidden activations used in the encoder and decoder) in a CBM's concept accuracy and oracle impurity.

6.5 Experiment Details

In this section we provide further details on the datasets used for our experiments as well as the architectures and training regimes used to obtain our presented results.

6.5.1 Datasets

For benchmarking, we use three datasets: a simple synthetic toy tabular dataset extending that defined in (Mahinpei et al., 2021), *dSprites* (Matthey et al., 2017), and *3dshapes* (Burgess & Kim, 2018). In order to investigate the impact of dependency between concepts on the quality of concept representation in each method, we allow a varying degree of dependencies in concept annotations and propose the following parameterized tasks:

- *TabularToy*(δ): this dataset consists of inputs $\{\mathbf{x}^{(i)} \in \mathbb{R}^7\}_{i=1}^N$ such that the j-th coordinate of $\mathbf{x}^{(i)}$ is generated by applying a non-invertible nonlinear function $f_j(z_1^{(i)}, z_2^{(i)}, z_3^{(i)})$ to 3 latent factors $\{z_1^{(i)}, z_2^{(i)}, z_3^{(i)}\}$. These latent factors are sampled from a multivariate normal distribution with zero mean and a covariance matrix with δ in its off-diagonal elements. The concept annotations for each sample correspond to the binary vector $\mathbf{c}^{(i)} := [\mathbb{1}(z_1^{(i)} > 0), \mathbb{1}(z_2^{(i)} > 0), \mathbb{1}(z_3^{(i)} > 0)]$ and the task we use is to determine whether at least two of the latent variables are positive. In other words, we set $y^{(i)}$ to $\mathbb{1}\big((c_1^{(i)} + c_2^{(i)} + c_3^{(i)}) \geq 2\big)$. The individual functions used to generate each coordinate of $\mathbf{x}^{(i)}$ are the same as those defined in (Mahinpei et al., 2021). As in (Mahinpei et al., 2021), we use a total of $2,000$ generated samples during training and a total of $1,000$ generated samples during testing.
- *dSprites*(λ): we define a task based on the dSprites dataset where each sample $\mathbf{x}^{(i)} \in \{0,1\}^{64\times64\times1}$ is a grayscale image containing a square, an ellipse, or a heart with varying positions, scales, and rotations. Each sample $\mathbf{x}^{(i)}$ is procedurally generated from a vector of ground truth factors of variation $\mathbf{z}^{(i)} = [\text{shape} \in \{0,1,2\}, \text{scale} \in \{0,\cdots,5\}, \theta \in \{0,\cdots,39\}, x \in \{0,\cdots,31\}, y \in \{0,\cdots,31\}]$ (θ indicating an angle of rotation) and is assigned a binary concept annotation vector $\mathbf{c}^{(i)} \in \{0,1\}^5$ with elements $\mathbf{c}^{(i)} := [\mathbb{1}\big(z_1^{(i)} < 2\big), \mathbb{1}\big(z_2^{(i)} < 3\big), \mathbb{1}\big(z_3^{(i)} < 20\big), \mathbb{1}\big(z_4^{(i)} < 16\big), \mathbb{1}\big(z_5^{(i)} < 16\big)]$. For this task, we construct a set of 8 labels from the concept annotations by setting $y^{(i)} = \big[c_2^{(i)} c_4^{(i)}\big]_{10}$ if $c_1^{(i)} = 1$ (where we use $[b_1 b_2]_{10}$ to indicate the base-10 representation of a binary number with digits b_1

and b_2) and $y^{(i)} = 4 + \left[c_3^{(i)} c_5^{(i)}\right]_{10}$ otherwise. Finally, we parameterize this dataset on the dependency number $\lambda \in \{0, 1, \cdots, 4\}$ that indicates the number of random dependencies we introduce across the sample-generating factors of variation (with $\lambda = 0$ implying all factors of variation are independent). For example, if $\lambda = 1$, we introduce a conditional dependency between factor of variations z_1 ("shape") and z_2 ("scale") by assigning each value of z_1 a random subset of values that z_2 may take given z_1. This subset is sampled by selecting, at random for each possible value of z_1, half of all the values that z_2 can take. More specifically, if z_1 and z_2 can take a total of T_1 and T_2 different values, respectively, then for each $a \in \{0, 1, \cdots, T_1\}$ we constraint z_2 to be able to take only $\lfloor T_2/2 \rfloor$ values from the set $z_2 \in \mathcal{Z}_2(a)$ defined as:

$$\mathcal{Z}_2(a) = \begin{cases} \textit{SWR}(\{0, 1, \cdots, \lfloor \frac{3T_2}{4} \rfloor\}, \lfloor T_2/2 \rfloor) & c_1 = 1 \\ \textit{SWR}(\{\lfloor \frac{T_2}{4} \rfloor, \cdots, T_2 - 1\}, \lfloor T_2/2 \rfloor) & \text{otherwise} \end{cases}$$

where $\textit{SWR}(A, n)$ stands for Sample Without Replacement and is a function that takes in a set A and a number n and returns a set of n elements sampled without replacement from A. This process is recursively extended for higher values of λ by letting the dataset generated with $\lambda = i$ be the same as the dataset generated by $\lambda = i - 1$ with the addition of a new conditional dependency between factor of variations z_i and $z_{(i+1)}$. Finally, in order to maintain a constant dataset cardinality as λ varies, we subsample all allowed factor of variations in $\{z_{\lambda+2}, z_{\lambda+3}, \cdots, z_5\}$ by selecting only every other allowed value for each of them. This guarantees that once a conditional dependency is added, the cardinality of the resulting dataset is the same as the previous one. Because of this, all parametric variants of this dataset consist of around $\sim 32,000$ samples.

- *3dshapes*(λ): we define a task based on the 3dshapes dataset where each sample $\mathbf{x}^{(i)} \in \{0, 1, \cdots, 255\}^{64 \times 64 \times 3}$ is a color image containing a sphere, a cube, a capsule, or a cylinder with varying component hues, orientation, and scale. Each sample $\mathbf{x}^{(i)}$ is procedurally generated from a vector of ground truth factors of variation $\mathbf{z}^{(i)} = [\text{floor_hue} \in \{0, 1, \cdots, 9\}, \text{wall_hue} \in \{0, 1, \cdots, 9\}, \text{object_hue} \in \{0, 1, \cdots, 9\}, \text{scale} \in \{0, 1, \cdots, 7\}, \text{shape} \in \{0, 1, 2, 3\}, \text{orientation} \in \{0, 1, \cdots, 14\}]$ and is assigned a binary concept annotation vector $\mathbf{c}^{(i)} \in \{0, 1\}^6$ with elements $\mathbf{c}^{(i)} := [\mathbb{1}(z_1^{(i)} < 5), \mathbb{1}(z_2^{(i)} < 5), \mathbb{1}(z_3^{(i)} < 5), \mathbb{1}(z_4^{(i)} < 4), \mathbb{1}(z_5^{(i)} < 2), \mathbb{1}(z_6^{(i)} < 7)]$. For this task, we construct a set of 12 labels from the concept annotations by setting $y^{(i)} = \left[c_1^{(i)} c_2^{(i)} c_3^{(i)}\right]_{10}$ if $c_5^{(i)} = 1$ and $y^{(i)} = 8 + \left[c_4^{(i)} c_6^{(i)}\right]_{10}$ otherwise. As in the dSprites task defined above, we further parameterise this dataset with parameter $\lambda \in \{0, 1, \cdots, 5\}$ to control the number of random conditional dependencies we introduce at construction time. The procedure used to introduce such dependencies is the same as in $dSprites(\lambda)$. Similarly, we use the same subsampling as in $dSprites(\lambda)$ to maintain a constant-sized dataset, resulting in all parametric variants of this dataset having around $\sim 16,000$ samples.

6.5.2 Model Architectures and Training

In all of the results we report on the *TabularToy*(δ) dataset, for CBM and CCD we use a 4-layer ReLU MLP with activations $[7, 128, 64, 3]$ as the concept encoder $g(\mathbf{x})$ and a 4-layer ReLU MLP with activations $[64, 128, 64, 1]$ as label predictor $f(\hat{\mathbf{c}})$. For CW, we use the same architecture with the exception that a CW module is applied to the output of the concept encoder model. For both CBM and CCD, we train each model for 300 epochs with a batch size of 32. For CW, we train each model for 300 epochs, with CW updates occurring every 20 batches, using a batch size of 128. Finally, for the mixture hyperparameter of the joint loss in CBMs, unless specified otherwise we use a value of $\alpha = 0.1$.

For our CBM and CCD experiments in *dSprites*(λ) and *3dshapes*(λ), we use a Convolutional Neural Network (CNN) with four (Conv + BatchNorm + ReLU + MaxPool) blocks with feature maps $\{8, 16, 32, 64\}$ followed by a three fully-connected layers with activations $\{64, 64, k\}$ for the concept encoder model $g(\mathbf{x})$ (with k being the number of ground truth concepts in the dataset). Furthermore, for the label predictor model $f(\hat{\mathbf{c}})$ we use a simple 4-layer ReLU MLP with activations $\{k, 64, 64, L\}$ (with L being the number of output labels in the task). For CW's concept encoder $g(\mathbf{x})$ we use a CNN with three (Conv + BatchNorm + ReLU + MaxPool) blocks with feature maps

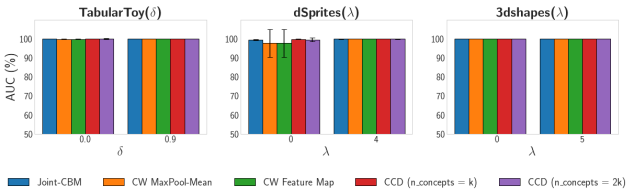


Figure 9: Downstream task predictive AUC for all datasets using original pre-trained models prior to bottleneck construction. Note that because DGL methods have no downstream task supervision in their training pipelines, we do not include those methods.

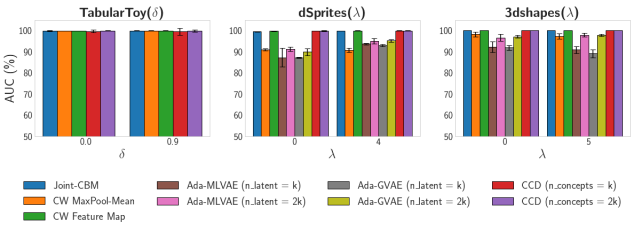


Figure 10: Downstream task predictive AUC for all datasets using the overall set of learnt concepts to predict the task labels. Note that this plot faithfully replicates the downstream task predictive AUC of methods that received direct task supervision in their training (shown in Figure 9).

$\{8, 16, 32\}$ followed by a (Conv + CW) block with 64 feature maps. For the label predictor, we use a model composed of a MaxPool layer followed by five ReLU fully connected layers with activations $\{64, 64, 64, 64, L\}$. All models evaluated for CBM and CCD are trained for 100 epochs using a batch size of 32. In contrast, CW models are trained for 100 epochs using a batch size of 256 and an CW module update step every 20 batches. Finally a value of $\alpha = 10$ is used during joint CBM training.

For evaluating Ada-GVAE and Ada-MLVAE, we use the same architecture as in CBM's and CCD's concept predictor for the encoder and the same architecture as in (Locatello et al., 2020b) for the decoder. The decoder consists of two ReLU fully connected layers with activations $\{256, 512\}$ followed by four ReLU deconvolutional layers with feature maps $\{64, 32, 32, \textit{input_feature_maps}\}$. All DGL models are trained for 100 epochs using a batch size of 32 with a dataset consisting of $\frac{2N}{3}$ pairs of images that share at least one factor of variation (with N being the number of samples in the original dataset). As in other methods, we train all DGL models using a default Adam optimizer (Kingma & Ba, 2014), with learning rate 10^{-3}.

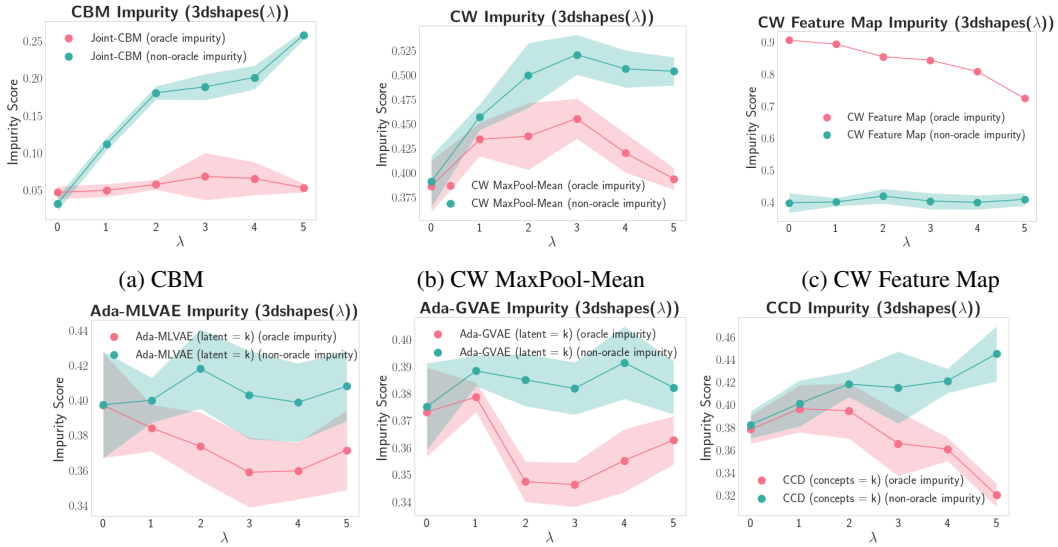
When evaluating CCD, we use a threshold of $\beta = 0.0$ for computing concepts scores and the same regulariser parameters $\lambda_1 = 0.1$, $\lambda_2 = 0.1$, $\epsilon = 10^{-5}$ as in Yeh et al. (2020)'s released code for their work in (Yeh et al., 2020). Finally, all models, across all tasks, are trained for 100 epochs and a batch size of 32 using a default Adam optimizer (Kingma & Ba, 2014), with learning rate 10^{-3}.

6.6 Downstream Task Predictive AUC

Figure 9 shows downstream task predictive AUC for all datasets using raw inputs, in absence of any dependencies as well as the maximum dependencies between concepts. If concept representations from various methods are good surrogate to input, they need to recover the same predictive performance. Figure 10 confirms that this holds to a good degree by looking at the task AUC of a simple ReLU MLP with hidden layers $\{64, 64\}$ trained to predict the corresponding task labels using the concept representations generated by each method.

6.7 Impurity Scores (Oracle and Non-oracle)

Figure 11 confirms that, for 3dShapes(λ), across all models the non-oracle impurity misleadingly shows more impurity that often increases with the dependence of concepts.



(a) CBM (b) CW MaxPool-Mean (c) CW Feature Map

(d) Ada-MLVAE (e) Ada-GVAE (f) CCD

Figure 11: Impurity scores (both oracle and non-oracle) in *3dshapes*(λ).