Interactive Disentanglement: Learning Concepts by Interacting with their Prototype Representations

Supplementary Materials

1 Hyperparameters and model details

1.1 Interactive Concept Swapping Networks

In our experiments, the prototype slots were initialized randomly from a truncated Gaussian distribution with mean $\mu = 0$, variance $\sigma^2 = 0.5$, minimum $a = -1$, and maximum $b = 1$. The encoder used in our experiments was a convolutional neural network with residual connections and ReLU activations. Each read-out encoder is a linear layer with LeakyReLU activations. Lastly, the decoder architecture was again a neural network with transposed convolutions and also here residual layers. In the standard ECR experiments with iCSNs, $J = 3$, $Z = 512$, $Q = 128$, $K = 6$ for each $j \in [1, \ldots, J]$. $N = 128$ and $\tau$ was decreased every 1000 epochs with steps $[0.5, 0.1, 0.01, 0.001, 0.0001, 0.00001]$ over 8000 epochs in total. Notably, group normalization as proposed by Wu et al. [2] was applied after extracting the concept encodings via the read-out encoders (performed in collectedReadOutEncoders in Alg. 1).

Pseudo code can be found in Alg. 1 and Alg. 2.

1.2 Baseline models

For the experiments with Cat-VAE, the softmax temperature was set to $\tau = 0.1$ and each categorical distribution had $k = 6$ categories. The number of latent variables was set to 3 for Cat-VAE, $\beta$-VAE, Ada-VAE, and VAE runs. For all baselines, the encoder and decoder consisted of a convolutional and transposed convolutional network. In all experiments, $\beta = 4$, except for Ada-VAE, where $\beta = 1$, as recommended by Locatello et al. [1]. All baseline models were trained for 2000 epochs.

1.3 Linear probing

The linear models for probing the latent representations of the different model configurations were a decision tree and logistic regression model. The max depth of the decision tree was set to 8. The logistic regression model was run with parameters $C = 0.316$ and the maximum number of iterations at 1000. Both the decision tree and logistic regression model were trained with a fixed random seed.

2 Details on simulated interactions

The simulated user interactions were performed via an $L_2$ regulatory loss term on the latent codes $y$.

In case a user tells an iCSN not to use a specific prototype slot of the superordinate concept $j$ and slot $k$, this loss corresponds to: $MSE(y_{j \cdot k + k}, 0)$, where $y_{j \cdot k + k}$ corresponds to the value of $y$ at position $j \cdot k + k$ and 0 being a vector of length $N$.

When a user provides a subset of examples with corresponding desired prototype slot IDs the loss term corresponds to: $MSE(y_{j \cdot k + k}^{\text{subset}}, 1)$ with 1 of length equal to the number of samples in $\text{subset}$. The subset of examples in our simulated interactions were identified via the ground truth labels, e.g., for identifying the subset of images containing a pentagon. The interactions for learning a novel basic concept followed the same procedure.

To allow the model to update is latent space via interactions we increased $\tau = 0.00001$ back to $\tau = 0.0001$.

References


Algorithm 1: Interactive Concept Swapping Network – pair images forward pass

Input: Image pair \( x \in \mathbb{R}^D, x' \in \mathbb{R}^D \), known share IDs \( v \).

Output: Image reconstructions \( \hat{x} \in \mathbb{R}^D, \hat{x}' \in \mathbb{R}^D \), and latent codes \( y \in [0, 1]^{J \times K}, y' \in [0, 1]^{J \times K} \).

1 // Forward pass through initial encoder. \( z \in \mathbb{R}^Z \)
2 \( z \leftarrow f(x) \)
3 \( z' \leftarrow f(x') \)
4 // Forward pass through \( J \) read-out encoders. \( \phi \in \mathbb{R}^{J \times Q} \)
5 \( \phi \leftarrow \text{collectedReadOutEncoders}(z) \)
6 \( \phi' \leftarrow \text{collectedReadOutEncoders}(z') \)
7 // Compute the distance of each concept encoding to all prototype slots of its corresponding category \( j \).
8 \( y \leftarrow \text{computeProtoDistance}(\phi, v) \)
9 \( y' \leftarrow \text{computeProtoDistance}(\phi', v) \)
10 // Reconstruct the images from the prototype distance codes.
11 \( \hat{y} \leftarrow g(y) \)
12 \( \hat{y}' \leftarrow g(y') \)

Algorithm 2: computeProtoDistance

Input: Concept encodings \( \phi \in \mathbb{R}^{J \times Q}, \phi' \in \mathbb{R}^{J \times Q} \)

Given: Set of prototype slot codebooks \( \Theta := [P_1, ..., P_J] \in \mathbb{R}^{J \times Q \times K} \), softmax temperature \( \tau \), and share IDs \( v \).

Output: Latent codes \( y \in [0, 1]^{J \times K}, y' \in [0, 1]^{J \times K} \)

1 // For every superordinate concept category
2 for \( j \leftarrow 0 \) to \( J - 1 \) do
3 // Dot-product between concept encoding and all prototype slots from codebook \( P_j \).
4 \( s_j \leftarrow \text{softmaxDotProduct}(\phi_j, P_j) \)
5 \( s'_j \leftarrow \text{softmaxDotProduct}(\phi'_j, P_j) \)
6 // Compute normalizing weighted softmax.
7 \( \Pi_j \leftarrow \text{softmaxNormTau}(s_j, \tau) \)
8 \( \Pi'_j \leftarrow \text{softmaxNormTau}(s'_j, \tau) \)
9 end for
10 // Swap the distance codes at the position corresponding to the shared IDs.
11 \( y \leftarrow [\Pi_1, ..., \Pi'_v, ..., \Pi_J] \)
12 \( y' \leftarrow [\Pi'_1, ..., \Pi_v, ..., \Pi'_J] \)