Supplementary Materials of
Learning Bottleneck Concepts in Image Classification

Contents

1. Unveiling Learned Concepts

Figure 1a shows the activations of all 20 concepts for digit `0` to `9`, learned from MNIST [4]. We can see that each digit only has a few concepts activated, *e.g.*, digit `7` has `Cpt.3`, `Cpt.8`, `Cpt.11`, `Cpt.12`, and `Cpt.15`. We also show the top-10 activated samples for each concept (*i.e.*, for concept κ, the samples with the highest ten t_κ's in the training set) in Figure 1b. It can be observed that some concepts are hardly activated. For instance, `Cpt.1` has no significant highlights, suggesting smaller t_κ.

Figure 1c provides the reconstruction results by our concept decoder when a certain concept is deactivated by setting corresponding t_κ being zero (reconstructed images with significant visual changes are marked in red). We can see that digit `7` turns into digit `9` when `Cpt.3` is deactivated. Figure 1d shows that this change happens consistently for all samples of digit `7`.

In addition, as our classifier is a single fully-connected (FC) layer, we can easily obtain the contribution of concept κ to class ω as $I_{\omega\kappa} = t_\kappa z_{\omega\kappa}$, where $z_{\omega\kappa}$ is the (ω, κ)-th element of the learnable matrix Z of the FC layer in Eq. (13) in the main paper. Figure 1e gives the importance of each concept for the digit `7` shown in Figure 1a. We can see that `Cpt.3` and `Cpt.8` are among the most decisive concepts.

Figure 2 shows the attention map for each concept for the input image (the left-most image) and $z_{\omega\kappa}$'s for $\omega =$ `yellow headed black bird`. We see that the classification of `yellow headed black bird` is mainly based on `Cpt.2` and `Cpt.16`, which look to represent the breast and head, respectively. Figure 3 show 10 example images with the attention map for each concept learned from a 50-class subset of CUB200 [12], where the 10 images are of the highest t_κ. We can see that the attended regions of most concepts are consistent among its top-10 activated samples, and most concepts look to represent meaningful patterns. For example, `Cpt.7` focuses on the leg, and `Cpt.8` focuses on wings.

The concepts learned on the ImageNet dataset [3] are shown in Figure 6. We can see that, for the given sample of `Goldfish` (Figure 4a), the two most important concepts are `Cpt.2` and `Cpt.9`. These two concepts cover semantically consistent regions in the training samples (according to Figure 4b) and look to represent dorsal fins and a near-gill region, respectively.

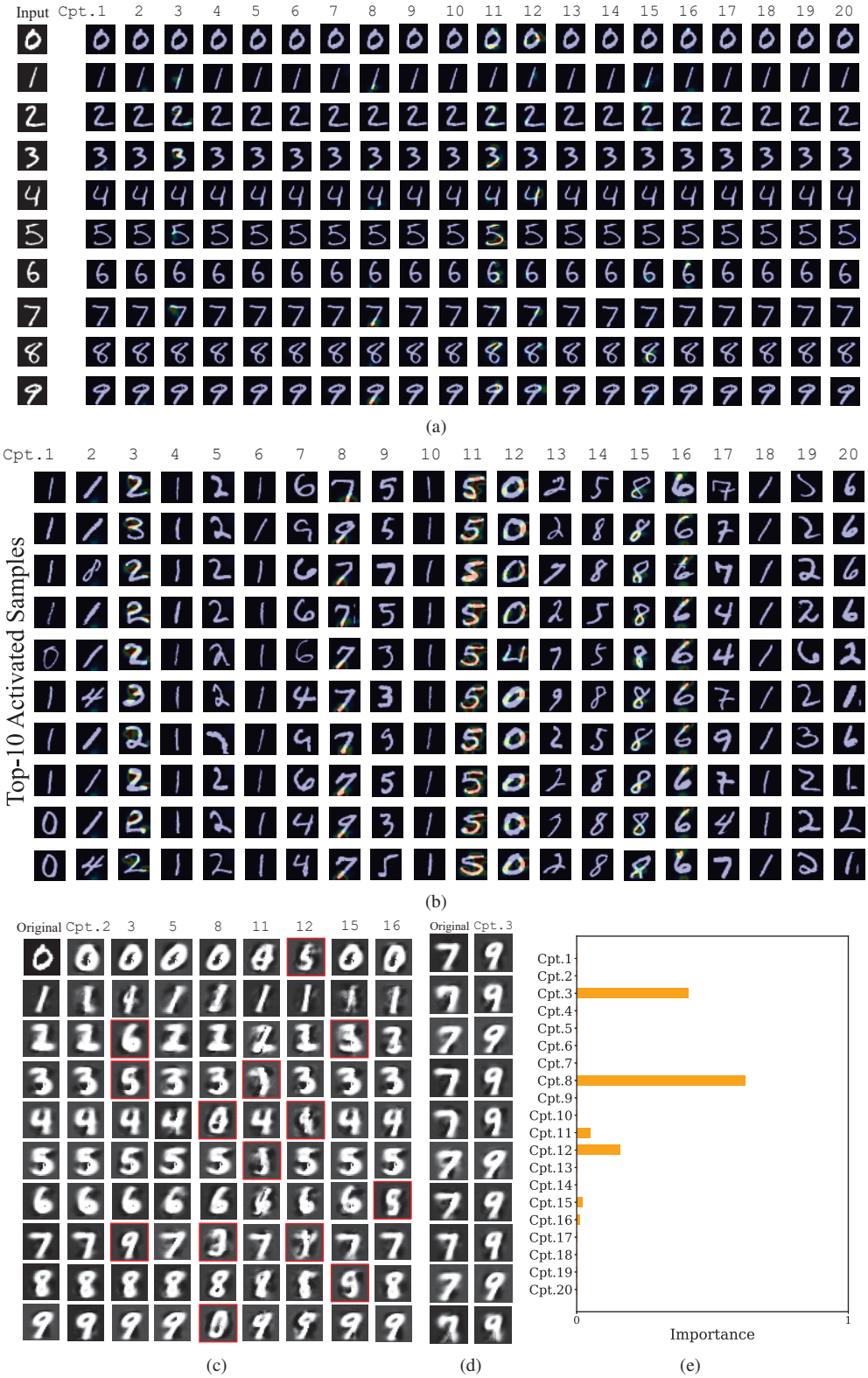


Figure 1. (a) Attention map for each of 20 concepts extracted for the input (left-most) image of each digit. (b) Top-10 activated samples for each concept. (c) Image reconstruction with one concept deactivated. (d) Image reconstruction for different samples of digit `7` with deactivating `Cpt.3`. (e) Concept importance for digit `7`.

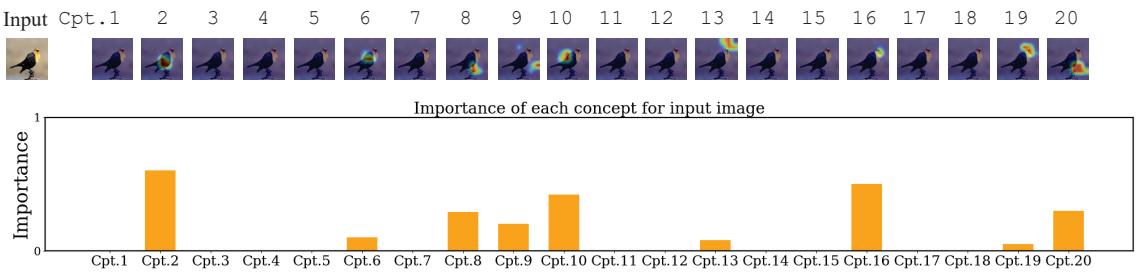


Figure 2. Concept activations for a sample of `yellow headed black bird`.

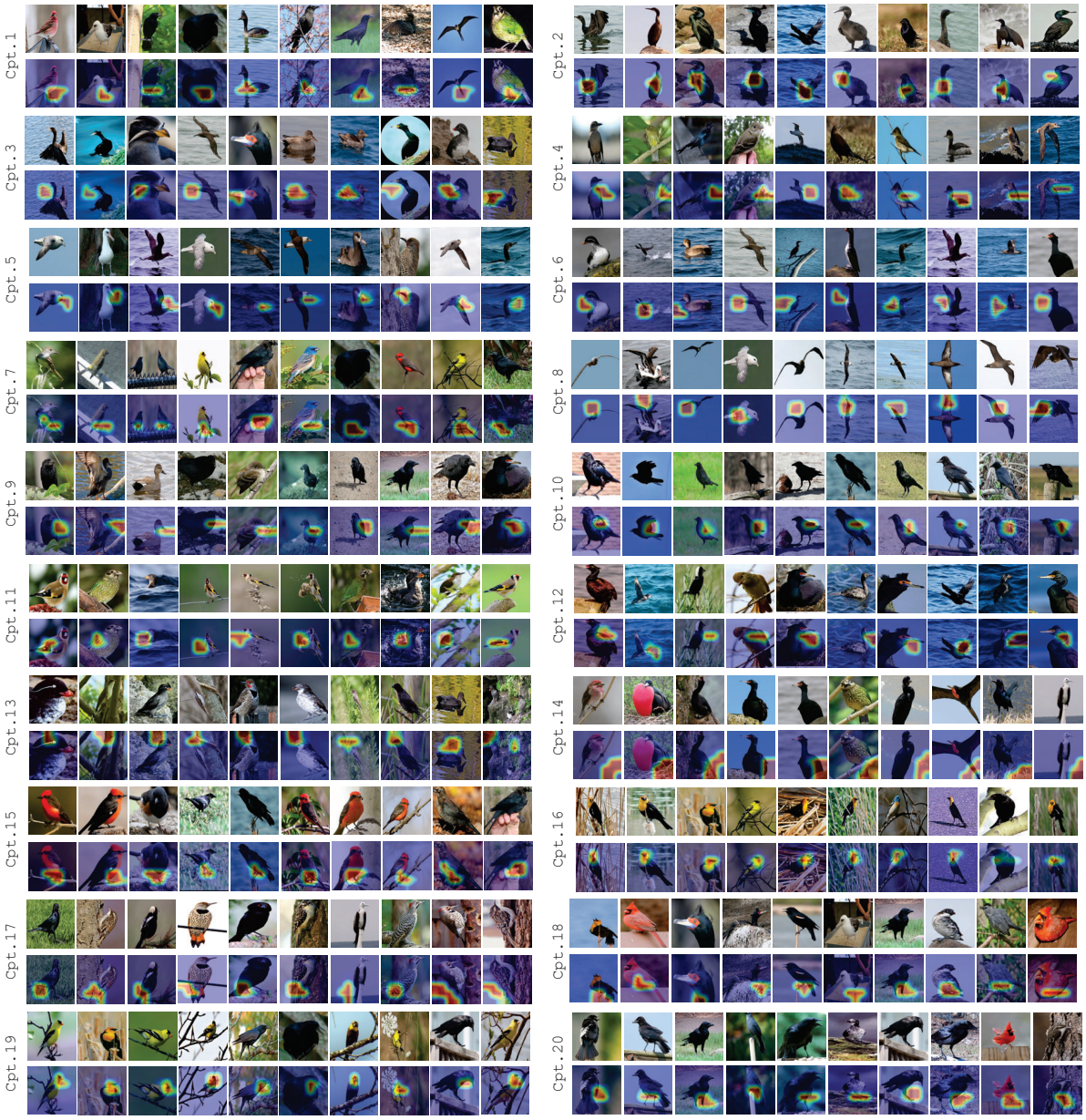


Figure 3. Concepts learned from CUB200, represented by top-10 activated samples.

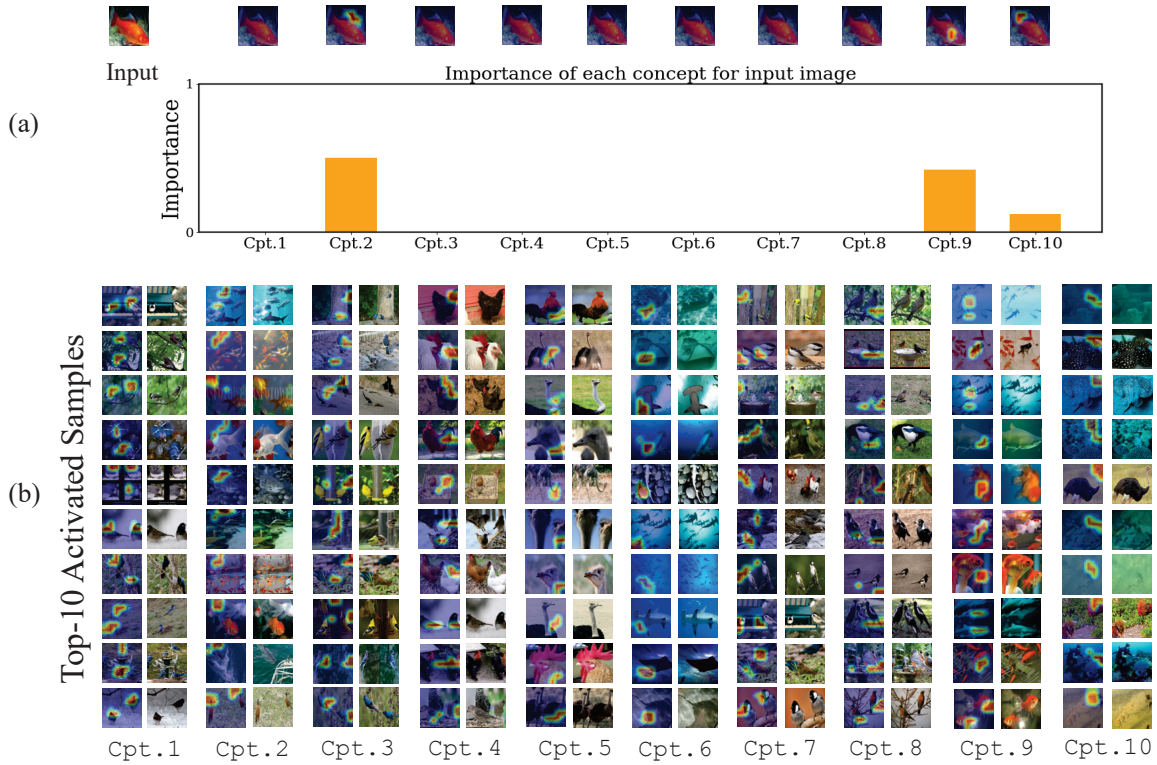


Figure 4. (a) Concept activations for a sample of `Goldfish` and the importance of each concept. (b) Concepts learned from ImageNet ($n = 20$ and $k = 10$).

2. Details of Experiments Settings

2.1. Normalization Function ϕ

The normalization ϕ determines the spatial distribution of each concept, which may depend on the target domain. For example, images for the handwritten digit recognition dataset are typically in black and white, and only the shape formed by strokes matters. In this case, concepts are less likely to overlap with each other spatially. Meanwhile, natural images have colors, textures, and shapes; any (combination) of them can be a concept. Thus, concepts possibly coincide at the same spatial position.

Let $a'_k = Q(c_\kappa)^\top K(F')$ (appears in Eq. (1)). For domain with supposedly non-overlapping concepts (*e.g.*, MNIST), we use ϕ given by

$$\phi_\kappa(\{a'_\kappa\}) = \sigma(a'_\kappa) \odot \mathrm{softmax}_\mathrm{S}(\{a'_\kappa\}). \tag{1}$$

This normalization takes $\{a'_\kappa\}$ for all concepts as input, which slightly abuses Eq. (1) of the main paper. σ is the (element-wise) sigmoid function, and $\odot$ is the Hadamard product. $\mathrm{softmax}_\mathrm{S}(\cdot)$ is taken over all concepts at each spatial position, so different concepts are less likely to be detected at the same spatial position.

For domains with overlapping concepts (*e.g.*, CUB200 and ImageNet), we only use the sigmoid function for normalization as

$$\phi(a'_\kappa) = \sigma(a'_\kappa). \tag{2}$$

2.2. Weight $\alpha(y, y')$

Equation (5) in the main paper uses weight $\alpha(y, y')$ to mitigate the imbalance of class distribution. Among a mini-batch $\mathcal{B}$, the number C_S of pairs with the same label is far less than the number C_D of different labels. We therefore introduce a weight $\alpha(y, y')$ given by

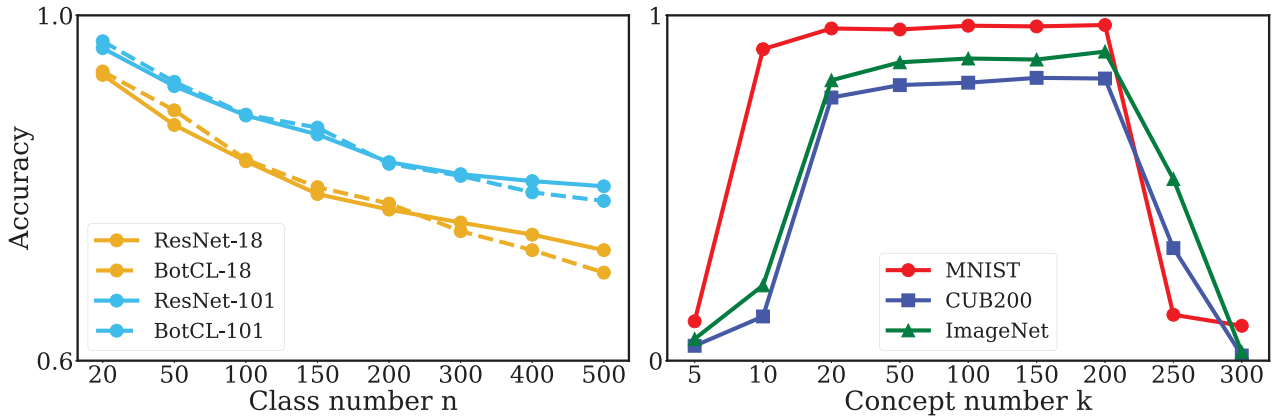
$$\alpha(y, y') = \begin{cases} C_\mathrm{D}/(C_\mathrm{S} + C_\mathrm{D}), & \text{for } y = y' \\ C_\mathrm{S}/(C_\mathrm{S} + C_\mathrm{D}), & \text{otherwise} \end{cases}. \tag{3}$$

2.3. Implementation of k-means and PCA

We use the ResNet-18 backbone to compute feature map $F \in \mathbb{R}^{d \times h \times w}$ from all images in the training set. Let $\mathcal{F}$ denote the set of all features $f_{pq} \in \mathbb{R}^d$ in F ($p = 1, \ldots, h$ and $q = 1, \ldots, w$) from all images (thus, $|\mathcal{F}| = N \times h \times w$). We apply k-means or PCA to $\mathcal{F}$. The cluster centers or the principal components are deemed as concepts.

Let $f_{pq} \in \mathbb{R}^d$ be features at the spatial position (p, q) in a new image, after necessary preprocessing[1]. We can

[1]PCA's features should be centered by subtracting the mean of $\mathcal{F}$.



(a) (b)

Figure 5. The relationships between the accuracy and the hyperparameter settings. (a) Number n of classes vs. accuracy. (b) Number k of concepts vs. accuracy.

calculate the soft-assignment $a_{\kappa pq}$ of f_{pq} to each concept c_κ. For k-means, we used

$$a_{\kappa pq} = e^{-\|f_{pq} - c_\kappa\|}. \tag{4}$$

For PCA, we adopt the absolute value of the cosine similarity, given by

$$a_{\kappa pq} = \text{abs}(\text{sim}(f_{pq}, c_\kappa)), \tag{5}$$

where $\text{sim}(\cdot,\cdot)$ is the cosine similarity and $\text{abs}(\cdot)$ give the absolute value.

We aggregate $a_{\kappa pq}$ for all spatial positions to form attention map $a_\kappa \in \mathbb{R}^l$. Similarly to BotCL, we summarize the presence of each concept into concept activation t_κ by reducing the spatial dimension of a_κ as

$$t_\kappa = \tanh\left(\sum_{pq} a_{\kappa pq}\right). \tag{6}$$

Also, the classifier is learned from the concept activations computed for all images in the training set.

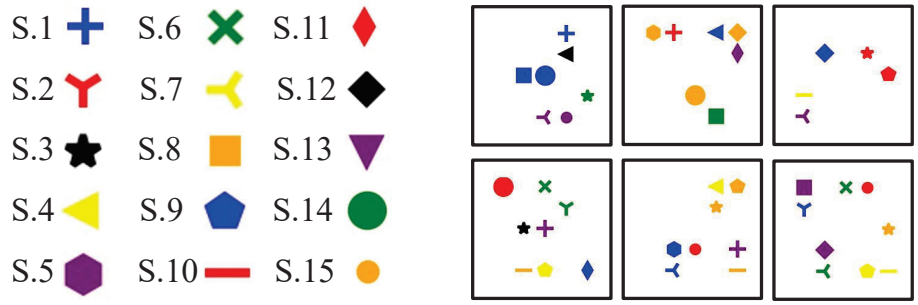
2.4. Numbers of Classes and Concepts

In Figure 5a, we evaluate the classification performance of BotCL on subsets of ImageNet with a different number n of classes while the number k of concepts is fixed at 50). BotCL has a competitive performance when n is less than 200, compared to the ResNet baseline. However, BotCL suffers from a performance drop when n is larger than 200, which means BotCL is more suitable for small- and middle-sized tasks.

This performance drop may be relieved by increasing k, as indicated in Figure 5b, where we give the relationship between the number k of concepts and the classification accuracy (with n fixed at 10 for MNIST; 50 for CUB200 and ImageNet).

On the one hand, a large k (when $k \leq 200$) can help improve the performance. The best performance for MNIST, CUB200, and ImageNet is achieved when $k = 100$, $k = 150$, and $k = 100$, respectively. This implies that k should be tuned for each dataset to achieve the best classification accuracy. However, training fails when $k \geq 300$. This is a drawback of BotCL.

On the other hand, k is directly related to the granularity of the learned concepts. That is, a larger k tends to learn finer-grained concepts, while a smaller k leads to coarse-grained ones. Therefore, the choice of k should be decided by jointly considering the actual needs of accuracy as well as the concept granularity.



(a) (b)

Figure 6. Generation of the Synthetic dataset. (a) Defined shapes from `S.1` to `S.15`, where (`S.1` to `S.5`) are the shapes-of-interest, while (`S.6` to `S.15`) are noises. (b) Data samples.

Table 1. Selected combinations of shapes-of-interest for the Synthetic dataset. "$\sim$" denotes NOT, "xor" denotes exclusive OR, "$+$" denotes OR, and "$\cdot$" denotes AND. For example, ω_1 presents in an image when the image does not contain both `S.1` and `S.3` or contains `S.4`.

Label	Definition
ω_1	$\sim$ (`S.1` $\cdot$ `S.3`) $+$ `S.4`
ω_2	`S.2` $+$ `S.3` $+$ `S.5`
ω_3	`S.2` $\cdot$ `S.3` $+$ `S.4` $\cdot$ `S.5`
ω_4	`S.2` xor `S.3`
ω_5	`S.2` $+$ `S.5`
ω_6	$\sim$ (`S.1` $+$ `S.4`) $+$ `S.5`
ω_7	(`S.2` $\cdot$ `S.3`) xor `S.5`
ω_8	`S.1` $\cdot$ `S.5` $+$ `S.2`
ω_9	`S.3`
ω_{10}	(`S.1` $\cdot$ `S.2`) xor `S.4`
ω_{11}	$\sim$ (`S.3` $+$ `S.5`)
ω_{12}	`S.1` $+$ `S.4` $+$ `S.5`
ω_{13}	`S.2` xor `S.3`
ω_{14}	$\sim$ (`S.1` $\cdot$ `S.5` $+$ `S.4`)
ω_{15}	`S.4` xor `S.5`

3. Details of the Synthetic Dataset

3.1. Generation

For evaluating the performance of concept discovery, we regenerate the Synthetic dataset using the official code from ConceptSHAP [13] (as the Synthetic dataset is not directly provided). As shown in Figure 6a, there are 15 different shapes (from `S.1` to `S.15`) in this dataset. The first 5 shapes (`S.1` to `S.5`) are selected as the shapes-of-interest, and the other 10 shapes are noises. As shown Table 1, 15 different combinations of the shapes-of-interest form 15 classes. The color of the shapes is randomly picked from 'green', 'red', 'blue', 'black', 'orange', 'purple', and 'yellow'. The positions of the shapes are constrained not to overlap each other. For this, we divide an image into a 7×7 grid (which coincides ResNet's grid corresponding to F) and place a single shape in a block. We show some samples in Figure 6b.

3.2. Quantitative Metrics

We denote a set of N_{E} test images as $\mathcal{X} = \{x_i | i = 1, \ldots, N_{\mathrm{E}}\}$ and a set of k learned concepts as $\mathcal{C} = \{\kappa | \kappa = 1, \ldots, k\}$. For each test sample $x \in \mathcal{X}$, we denote the ground-truth position of each shapes-of-interest `S.`j as s_j, which is a set of pixels inside the block (in the original image size). Meanwhile, we also define the area of each concept. For BotCL, k-means, and PCA, the spatial position of each concept is given by a_κ. We apply thresholding to a_κ to spot the concept. We denote the set of pixels whose attention value is larger than the threshold $\beta = 0.2$ by $\bar{a}_\kappa$. For ACE [5], $\bar{a}_\kappa$ includes all pixels in the super-pixels corresponding to the concept.

We first define $h_{j\kappa}$ for shape `S.`j and concept κ, which represents if κ overlaps `S.`j, as

$$h_{j\kappa} = \begin{cases} 1, & |s_j \cap a_\kappa| / |s_j| > \gamma \\ 0, & \text{otherwise} \end{cases} . \tag{7}$$

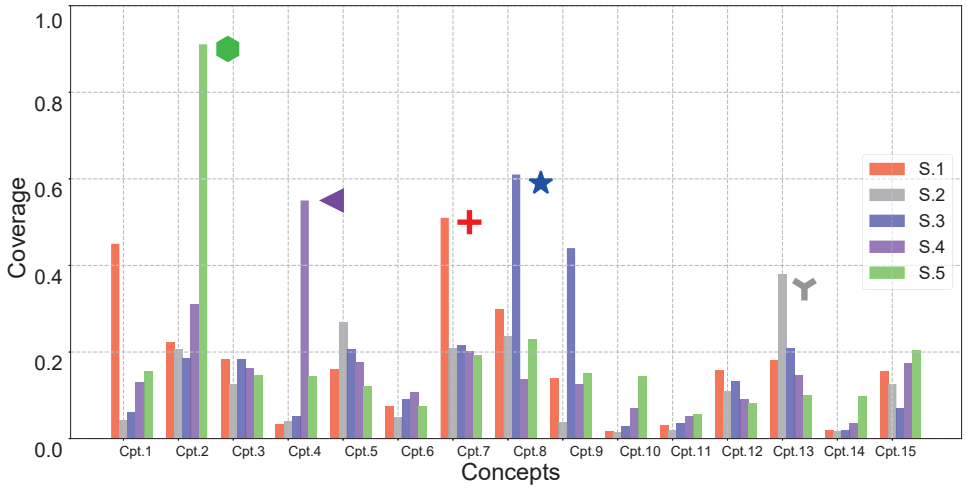
where $\cap$ is the intersection, and γ is a predefined threshold ($\gamma = 0.9$ in our setting). Note that we do not use IoU, as a single concept can cover multiple shapes. For example, one of the noise shapes (`S.6`–`S.15`) can be covered by a concept that also covers one of shapes-of-interest when the noise shape co-occurs with the shape-of-interest. In this case, the area of the concept is large, but this does not necessarily mean the discovered concept is inferior as the noise shapes are irrelevant to the target classification task. We thus design another metric named Purity to evaluate the practical purity of the concept, which is detailed later in this section.

The coverage of s by concept κ is then given by

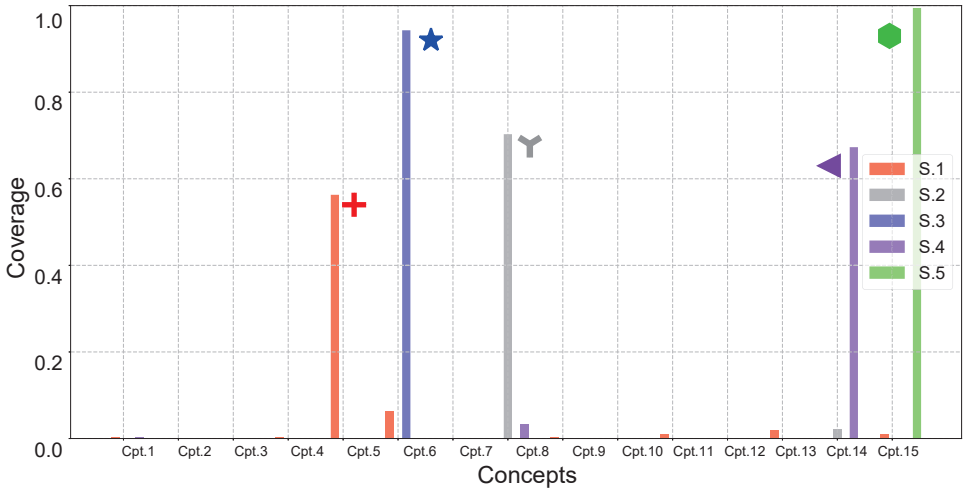
$$\text{Coverage}_{s\kappa} = \mathbb{E}[h_{s\kappa}], \tag{8}$$

which is computed over all images in $\mathcal{S}$ who contain s.

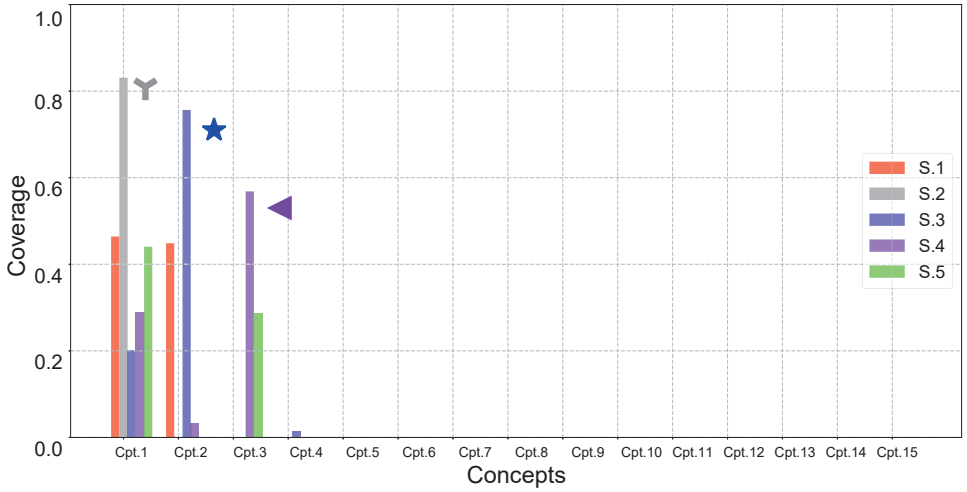
Similarly to [13], we associate each of the shapes-of-interest to one of the concepts for evaluation. Let $\mathcal{A}$ denote a set of pairs of a shape-of-interest and a concept, *i.e.*,



(a) ACE



(b) k-means



(c) PCA

Figure 7. Coverage$_{s_K}$ (the concept associated with each of the five shapes is marked).

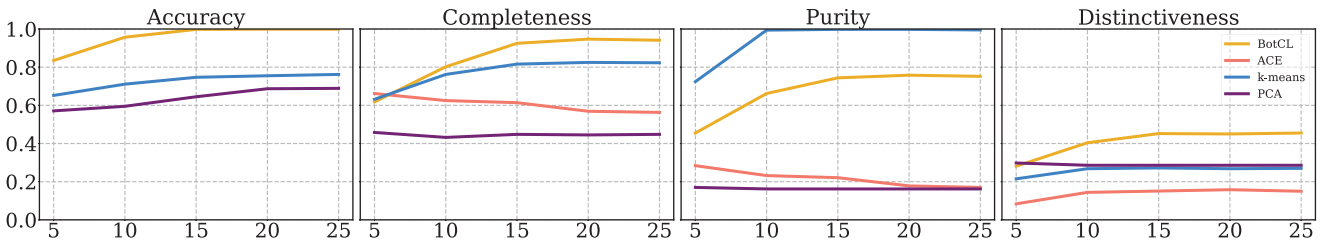


Figure 8. The impact of concept number k on BotCL, ACE, k-means, and PCA to classification accuracy, Completeness, Purity, ad Distinctiveness. Note that the classification accuracy of ACE is not shown because ACE is a post-hoc method and does not do classification by itself.

$\mathcal{A} = \{(\texttt{S}.j, \kappa_j) | j = 1, \ldots, 5\}$, where $\kappa_j \in \mathcal{C}$. We can find optimal $\mathcal{A}$ by

$$\mathcal{A}^* = \text{argmax}_{\mathcal{A}} \sum_{(s,\kappa)\in\mathcal{A}} \text{Coverage}_{s\kappa}. \tag{9}$$

Note that in this maximization, only concepts in $\mathcal{A}$ are the variables and the shapes are fixed.

Based on this association, three metrics are defined to evaluate the concept discovery performance.

- **Completeness**: The most important quality of a concept is whether it has the ability to capture the associated shape completely. This can be given by

$$\text{Completeness} = \frac{1}{|\mathcal{A}^*|} \sum_{(s,\kappa)\in\mathcal{A}^*} \text{Coverage}_{s\kappa} \tag{10}$$

- **Purity**: We also expect one learned concept to be pure; that is, a concept should only cover the associated shape but not the other shapes-of-interest. Thus, we define Purity as

$$\text{Purity} = \frac{1}{|\mathcal{A}^*|} \sum_{(s,\kappa)\in\mathcal{A}^*} \frac{\text{Coverage}_{s\kappa}}{\sum_{s'} \text{Coverage}_{s'\kappa}}, \tag{11}$$

  where the summation in the denominator is computed over all shapes-in-interest.

- **Distinctiveness**: We designed BotCL so that the discovered concepts are distinctive. That is, any pair of concepts should cover different sets of shapes. We thus define distinctiveness as

$$\text{Distinctiveness} = \frac{1}{5|\mathcal{O}|} \sum_{(\kappa,\kappa')\in\mathcal{O}} \sum_{s} |\text{Coverage}_{s\kappa} - \text{Coverage}_{s'\kappa'}|, \tag{12}$$

  where $\mathcal{O}$ is the set of all possible pairs of concepts in $\mathcal{A}^*$ and the second summation is computed over all shapes-in-interest.

3.3. Coverage of ACE, k-means, and PCA

Figure 7 shows Coverage of ACE [5], k-means, and PCA (with $k = 15$). We can observe for ACE that, although some concepts tend to be dominated by one shape (*e.g.*, `Cpt.1` captures `S.5`), most of the concepts are less discriminate. For k-means, one concept captures only one shape, which leads to high Purity. However, the completeness is not as good as BotCL (refer to Figure 6b of the main paper). For example, Coverage `Cpt.5` over `S.1` is less than 0.6. In addition, PCA does not extract enough meaningful concepts.

3.4. Impact of Number k of Concepts

As shown in Figure 8, we can observe that BotCL outperforms others regardless of k in all metrics except Purity. When $0 \leq k \leq 15$, all metrics mostly improve with k. However, a larger k harms Completeness and Purity of ACE and the Distinctiveness of PCA. When $k > 15$, there are no obvious changes for all methods on all metrics. Interestingly, k-means achieves the best Purity for any k. This means that features sufficiently discriminate different shapes. However, its performance over other metrics is mostly much lower than BotCL's.

4. Details on User Study

4.1. Design of user study

Designing a user study for evaluating the interpretability of unsupervised concepts is not trivial. One straightforward way can be to ask multiple participants to write a description for each concept by reviewing *e.g.*, the top-10 activated samples, but this approach poses an extra challenge in comparing free-form descriptions. Therefore, we decided to provide a vocabulary for each dataset so that the participants could choose some terms from it.

Table 2 shows our predefined vocabularies for MNIST and CUB200. For MNIST, we set the number k of concepts to 20, but only 8 of them are activated, and the others are never activated as shown in Figure 1b. Therefore, we only show the most activated images of these 8 concepts to the participants. The vocabulary consists of two groups, *position* and *shape*. These groups are combinatorial; the participants choose one from each to describe the concept. We found that some concepts cover two different elements of the digits, so we allow the participants to specify two pairs of position and shape. For example, a participant may choose *upper* and *a horizontal line* as well as *lower* and *a (part of) curve* for `Cpt.11` (refer to Figure 1b, as it involves two highlighted regions. When no consistent concept can be found in the provided samples, participants can choose *None of them*. For CUB200, all 20 concepts learned from a subset with $n = 50$ classes are presented. The vocabulary is defined based on the terms related to birds, falling into five groups (i) *Body Part*, (ii) *Color*, (iii) *Texture*, (iv) *Action*, and (v) *Background*. Each group requires to choose one term. Otherwise, a participant can choose *None of them* when no consistent concept can be found. We provide the screenshot of our user interface in Figure 9 and 10.

4.2. Metrics

We designed four metrics to evaluate learned concepts based on the user study:

- **Concept discovery rate** (CDR): This metric is the ratio of participants who can successfully find a meaningful concept in given samples, *i.e.*, the ratio of participants who selected terms other than *None of them*. This metric directly indicates how human-understandable the learned concepts at a conscious level.

- **Concept consistency** (CC): This metric involves the consistency of responses of a pair of participants for one concept, measuring inter-participant differences in the perception of a concept. Let R_{gi} denote participant i's response on group g, which is one of the terms in the group g. CC is formulated as :

$$\text{CC} = \sum_{g \in \mathcal{G}} w_g r_g \frac{1}{|\mathcal{P}|} \sum_{(i,j) \in \mathcal{P}} \mathbb{I}(R_{gi}, R_{gj}), \tag{13}$$

  where $\mathbb{I}$ is the indicator function that gives 1 when $R_{gi} = R_{gj}$, and 0 otherwise. $\mathcal{G}$ is the set of all groups, and $\mathcal{P}$ is the set of all possible pairs of participants. For MNIST, we expand the groups by making all possible combinations of positions and shapes because it is more natural to see the *position* group as modifiers. Therefore, for MNIST, $|\mathcal{G}| = 1$ and this single group contains $3 \times 8 = 24$ terms. We introduce w_g to compensate for the imbalance among the number of times, one term of each group is selected so that a group that is used many times can contribute more to the final score. w_g is the ratio of times in which one term in group g is selected overall responses. r_g is a discount factor for *None of Them*. Let η be the number of all pairs of participants and η'_g the number of pairs whose responses for group g are both non-*None of Them*. We define the discount factor as $r_g = 1 - \eta'_g/\eta$.

- **Mutual information between concepts** (MIC): This metric measures the similarity of the response distribution over all possible pairs of concepts. Letting H denote the concatenation of histograms H_g for all group g and H' is the same concatenated histogram, but for a different concept, it can be formulated as follows:

$$\text{MIC} = \text{MI}(H, H'), \tag{14}$$

  where MI gives the mutual information between H and H'. Note that for MIC, the statistics (the mean and standard deviation) are computed over all possible pairs of concepts, whereas for the other three metrics, they are computed over all concepts.

For comparison, we conducted an extra round of user study with manually labeled concepts and random concepts. To be consistent with BotCL's setting, we used the same number of concepts (8 for MNIST and 20 for CUB200) as well as the number of participants (20 for MNIST and 30 for CUB200). For manually labeled concepts, we picked out a (combination of) terms from our vocabulary to make a concept and selected 10 images that contained the concept. We then manually annotated the region corresponding to the concept in each image. This renders a certain cap for each metric. For random concepts, we randomly selected 10 samples for each concept and randomly generated highlights for each sample; therefore, there barely be a consistent concept within the samples. Figures 13–16 show some examples of manually labeled and random concepts for both MNIST and CUB200.

Table 2. Vocabulary used in the user study.

Dataset	Group	Vocabulary
MNIST	Position (3)	upper, middle, lower
	Shape (8)	the end of a slanted vertical line, the end of a vertical line, a (part of) curve, a (part of) right-open curve, a circle, a white-black-white pattern, a horizontal line, the edge around a curve/line
CUB200	Body Part (9)	head, wing, leg, beak, crawl, breast, tail, neck, back
	Color (10)	red, grey, beige, black, yellow, brown, white, blue, green, colorful
	Texture (2)	striped, spotted
	Action (4)	flying, swimming, climbing, perching
	Background (5)	sea, tree, sky, grass, land

We show the distributions of participants' answers in Figure 11 and 12. For MNIST, we can find that most concepts are recognized to be meaningful. The participants tend to choose the same term for one concept, *e.g.*, the option for `Cpt.8` mostly described by *lower* and *an end of a slanted vertical line*. However, we also observe that `Cpt.5` cannot be identified by most of the participants, as its highlighted regions are too weak and hardly noticeable as shown in Figure 1b. For CUB200, we show the distribution of each group in the first five columns, and the last column shows the number of participants who selected *None of them*. We find that most of the learned concepts in CUB200 are meaningful, as the number of *None of them* is small for most concepts. Participants' responses are mostly distributed in Body Part (especially *Wing* and *Leg*), *Color* (*Black*), and *Action* (*Perching*).

From this user study, we would conclude that the concepts learned by BotCL are recognizable, individually consistent, and mutually distinct for humans, comparable with manually labeled concepts, which means that BotCL can potentially apply to a wide range of applications that require interpretability.

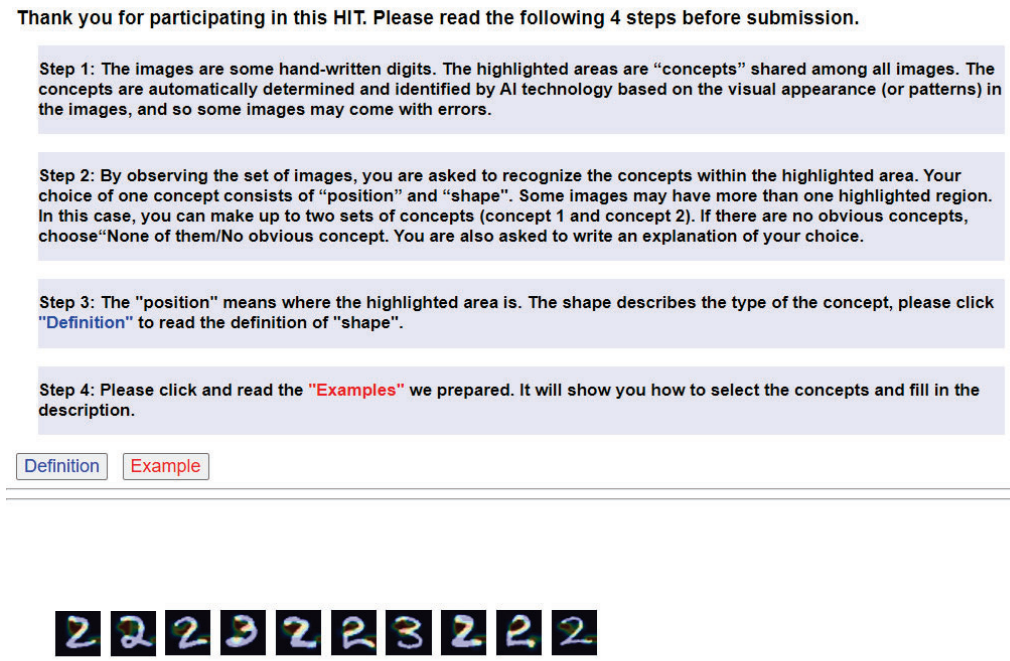


Figure 9. User interface of the user study for MNIST.

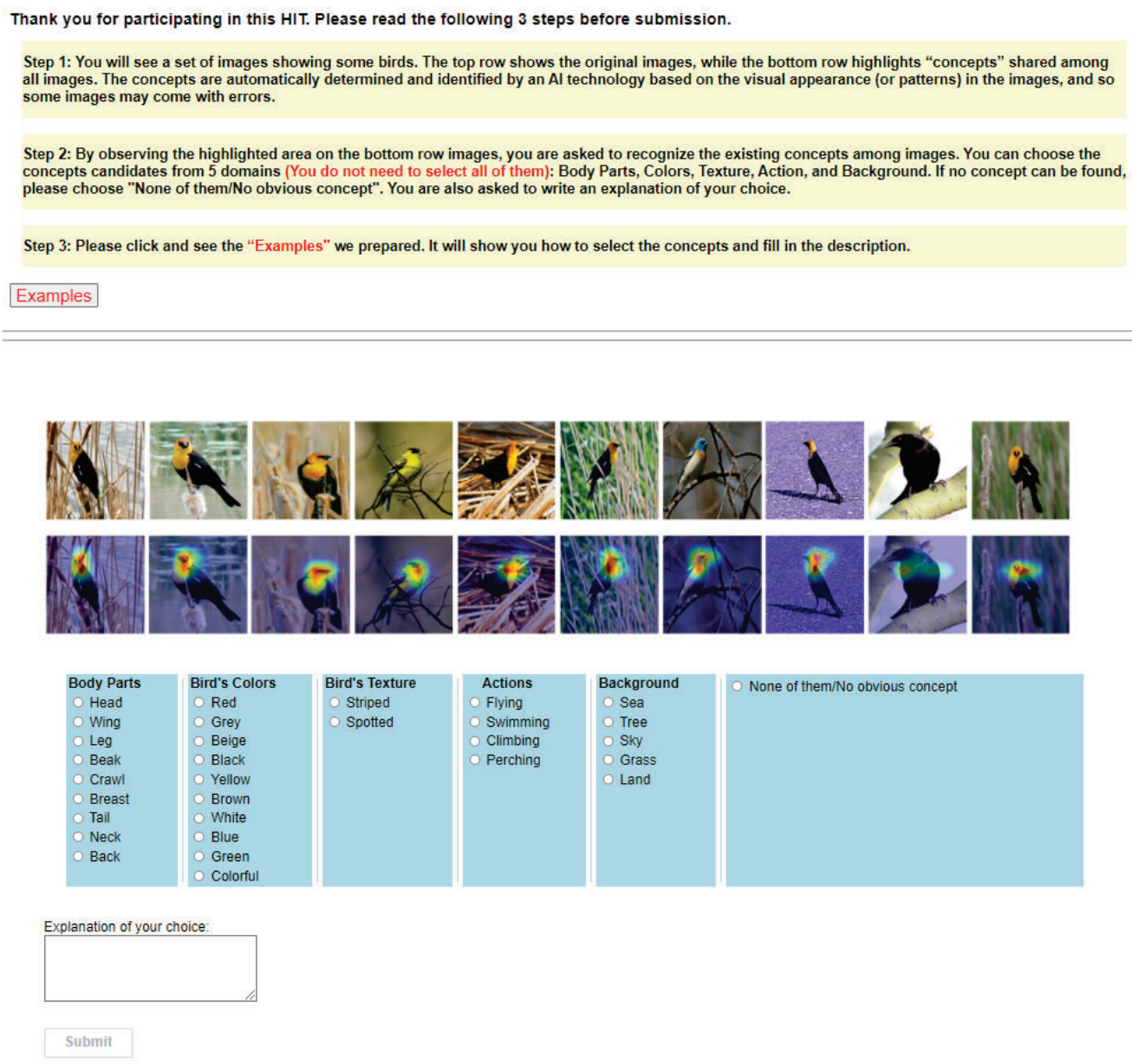


Figure 10. User interface of the user study for CUB200.

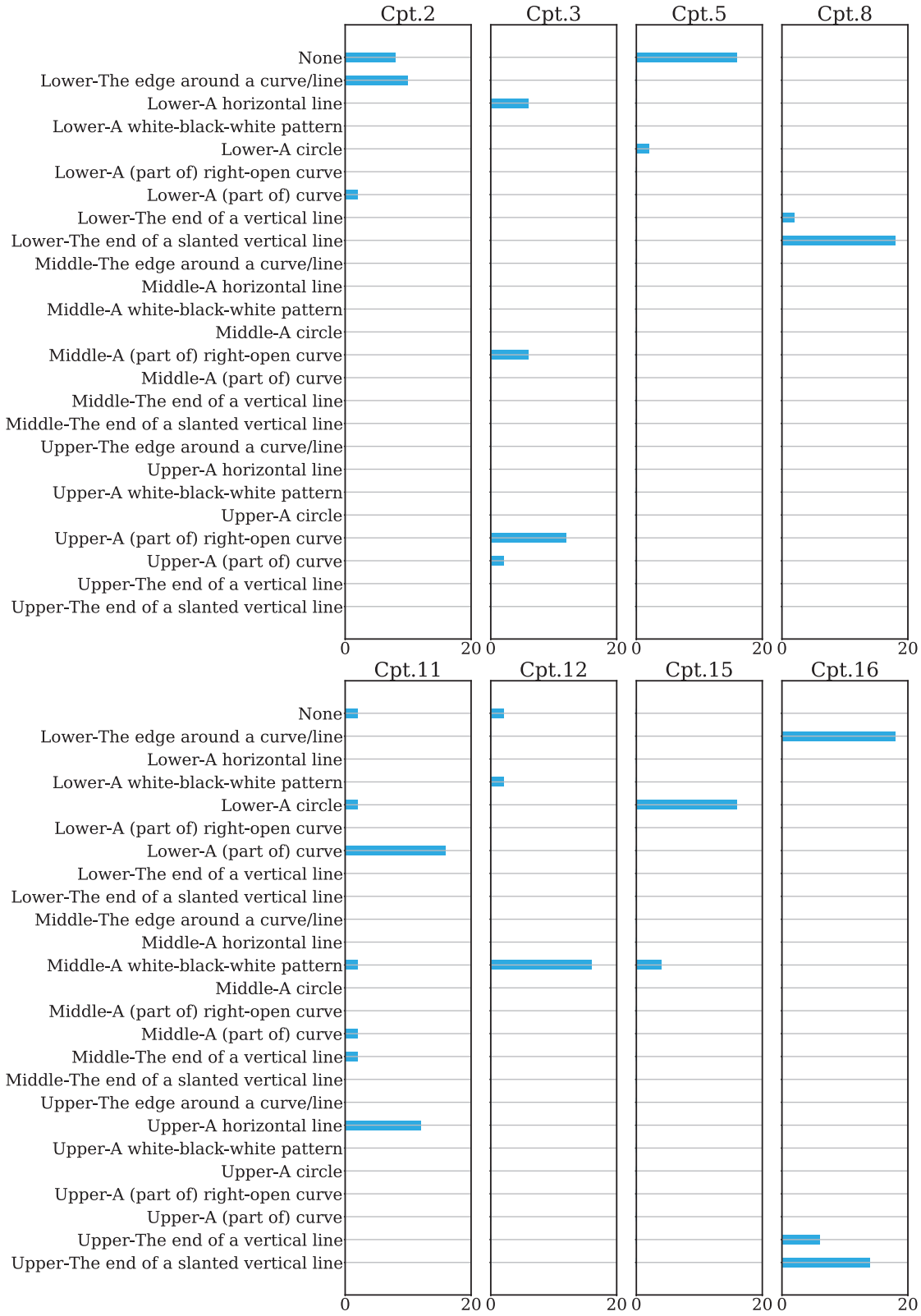


Figure 11. BotCL's distribution of responses for MNIST.

Figure 12. BotCL's distribution of responses for CUB200.

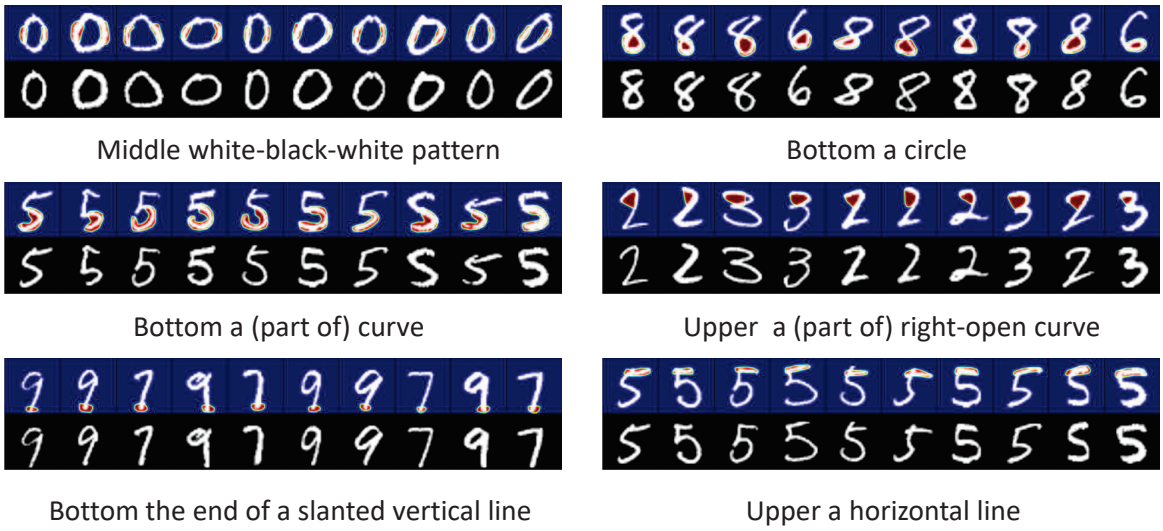


Figure 13. Examples of manually labeled concepts for MNIST.

Figure 14. Examples of random concepts for MNIST.

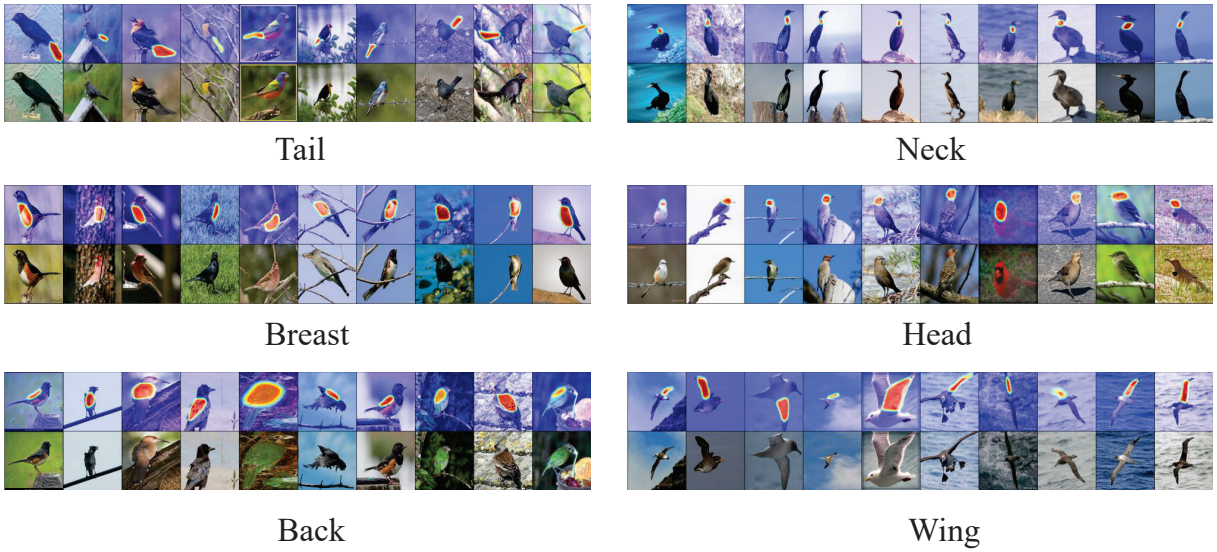


Figure 15. Examples of manually labeled concepts for CUB200.

Figure 16. Examples of random concepts for CUB200.

5. Comparison to existing XAI methods

BotCL aims at learning concepts, which is completely different from per-pixel importance-based XAI methods. Therefore, the explainability scores tailored for these XAI methods are not the main concern of this paper. Yet, comparing BotCL with major XAI methods gives strong evidence of its explainability. For this comparison, the attention of each concept can be merged into an overall explanation $\bar{a}$ by the weighted sum as

$$\bar{a} = \frac{1}{k} \sum_{\kappa} a_{\kappa} z_{\omega\kappa}, \tag{15}$$

where ω is the ground-truth class. We adopt four evaluation metrics, including Insertion area under curve (IAUC) and deletion area under curve (DAUC) are the metrics designed in [7], Stability [1], and Infidelity [13].

IAUC is calculated by gradually adding pixels (in the order of importance) to a blank image and seeing how the prediction confidence evolves. The prediction confidence should rise quickly if the pixel-adding process is guided by an explanation that well understands the model and thus can point out the most important pixels. In contrast, **DAUC** is calculated by gradually removing pixels (in the order of importance) from the original image. Similarly, the prediction confidence should drop quickly if the pixel-removing process is guided by an explanation.

Stability is quantified by Lipschitz estimation to measure how stable the explanation method performs when the input is perturbed with minor noises. It can be formulated as follows:

$$\text{Stability} = \frac{\|\mathcal{E}_{\gamma}(x) - \mathcal{E}_{\gamma}(x')\|_2}{\|x - x'\|_2}, \tag{16}$$

where x is the original input and x' is the perturbed input. γ is a model (*i.e.*, a composition of feature extractor Ψ and classifier f, and $\mathcal{E}_{\gamma}$ is the function to generate an explanation of model γ. Adding minor white noise to the input image should not have a significant impact on the prediction result. However, the explanation may change a lot if the method is instability.

Infidelity measures the consistency between input perturbations and consequent significant explanation changes. It is formulated as follows:

$$\text{Infidelity} = \mathbb{E}_{I \sim \mu_I}[(I^{\top}\mathcal{E}_{\gamma}(x) - (\gamma(x) - \gamma(x - I)))^2], \tag{17}$$

where I is a significant perturbation to the input with one probability measure μ_I, and the variables (*i.e.*, I and $\mathcal{E}_{\gamma}$) are vectorized if necessary. The paper [13] provides multiple options for μ_I, and we chose $\mu_I = \mathcal{N}(0, \sigma^2)$.

In addition, the explainability of the existing XAI methods is evaluated with the baseline ResNet [6] model, which uses a single FC as the classifier, while our results are obtained on BotCL, which uses the same ResNet model (without the FC classifier) as the backbone. In Table 3, we can see that BotCL achieves the best scores in stability and infidelity, and is among the best for IAUC/DAUC. BotCL is slightly worse than the best results in IAUC is that BotCL requires the activations of enough number concepts to lead to the correct classification, for which more pixels are necessary. Similarly, BotCL works well even if some concepts are not activated. More pixels need to be masked to change the output, which implies BotCL's robustness.

Table 3. Evaluation of BotCL and existing XAI methods using explainability metrics.

	CUB200				ImageNet			
Methods	Stability ↓	Infidelity ↓	IAUC ↑	DAUC ↓	Stability ↓	Infidelity ↓	IAUC ↑	DAUC ↓
LIME [8]	0.175	0.150	0.664	0.133	0.211	0.398	0.624	0.154
CAM [14]	0.170	0.138	0.695	0.114	0.208	0.372	0.678	0.135
GradCAM [9]	0.155	0.142	0.712	0.110	0.180	0.358	0.682	0.130
GradCAM++ [2]	0.168	0.135	**0.731**	**0.099**	0.188	0.360	0.687	0.121
Score-CAM [11]	0.160	0.122	0.725	0.102	0.176	0.355	**0.697**	**0.118**
SS-CAM [10]	0.166	0.130	0.698	0.109	0.191	0.377	0.675	0.133
BotCL	**0.102**	**0.051**	0.718	0.105	**0.125**	**0.341**	0.680	0.131

References

[1] David Alvarez-Melis and Tommi S Jaakkola. Towards robust interpretability with self-explaining neural networks. *NeurIPS*, 2018.

[2] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *WACV*, pages 839–847, 2018.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

[4] Li Deng. The mnist database of handwritten digit images for machine learning research. *Signal Processing Magazine*, 29(6):141–142, 2012.

[5] Amirata Ghorbani, James Wexler, James Zou, and Been Kim. Towards automatic concept-based explanations. *NeurIPS*, 2019.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[7] Vitali Petsiuk, Abir Das, and Kate Saenko. RISE: Randomized input sampling for explanation of black-box models. *BMVC*, 2018.

[8] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " Why should i trust you?" Explaining the predictions of any classifier. In *ACM SIGKDD*, pages 1135–1144, 2016.

[9] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *CVPR*, pages 618–626, 2017.

[10] Haofan Wang, Rakshit Naidu, Joy Michael, and Soumya Snigdha Kundu. SS-CAM: Smoothed Score-CAM for sharper visual feature localization. *arXiv preprint arXiv:2006.14255*, 2020.

[11] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *CVPR workshops*, pages 24–25, 2020.

[12] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-UCSD birds 200. 2010.

[13] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. In *NeurIPS*, volume 33, pages 20554–20565, 2020.

[14] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, pages 2921–2929, 2016.