A SHARCS IMPLEMENTATION DETAILS

A.1 DIFFERENT CONFIGURATION OF SHARCS

End-to-end It is possible to train all SHARCS components simultaneously, allowing a joint optimisation of the task and the concepts found. Therefore, it is also possible to include the loss of the local tasks in Equation 3. However, to use local supervision, we need to implement inside the model n local label predictor function $f_1, \ldots, f_n \in \mathbb{N}$, one for each modality $i = 1, \ldots, n$. The local label predictor function $f_i : C_i \rightarrow Y_i$ maps the local concepts from the i-th modality to the downstream local task space $Y \subseteq R^{l_i}$, where l_i is the number of classes of the local task of the modality i. Therefore the objective function to minimise is the following:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{s}) = \mathcal{T}(\mathbf{y}, \hat{\mathbf{y}}) + \frac{\lambda}{|M|} \sum_{(i,q) \in M \subseteq \binom{\{1,\ldots,n\}}{2}} \big|\big|\mathbf{s}_i - \mathbf{s}_q\big|\big|_2 + \sum_{i=1}^{n} \beta_i \mathcal{T}_i(\mathbf{y_i}, \hat{\mathbf{y}}_\mathbf{i}) \tag{7}$$

where $\beta_i \in \mathbb{R}$ is a hyperparameter that controls the strengths of the local loss $\mathcal{T}_i$.

Sequential The training process of this method is split in two parts. In the first one, a model similar to Concept Multimodal is trained. Therefore, unimodal models $g_1, \ldots, g_n$ are utilised to compute local concepts, which are concatenated and passed through the label predictor function to solve the downstream task. This part of the entire architecture is trained first, using an objective function equals to $\mathcal{T}$, solving the task using local concepts. Then, the concept encoders functions $g_1, \ldots, g_n$ are frozen. In the second part of the training, local concepts are projected into the shared space by $h_1, \ldots, h_n$, concatenated and used by f to make the final prediction. At this point, the standard loss described in Equation 3 is applied.

Local pre-training In this approach, SHARCS' single modality components $g_1, \ldots, g_n$ are trained first, using the same local label predictor functions $f_1, \ldots, f_n$ described in the end-to-end approach to make a prediction. Each is trained using their specific local loss $\mathcal{T}_i$. Then, the concept encoders functions $g_1, \ldots, g_n$ are frozen, while the other SHARCS' modules are employed and trained using the standard objective function described in Equation 3.

A.2 CONCEPT FINDING ON GRAPH

Although our solution is model agnostic, it is important to treat every modality properly. Therefore, we slightly modify the concept encoder function when it is composed of a Graph Neural Network. Specifically, we applied a modified version of the Concept Encoder Module (CEM)(Magister et al., 2022). In this case, the concept encoder function g_i is composed of a Graph Neural Network $\phi_i : X_i \rightarrow H_i$, a Gumbel Softmax (Jang et al., 2017) to find the "node concepts", an add pooling over the nodes of the graph, a batch scaling function and a sigmoid Function. Therefore to find $\mathbf{c}_{im}$, where i is a graph modality, the equation becomes the following:

$$\mathbf{t}_{im} = \phi_i(\mathbf{x}_{im}) \qquad \mathbf{n}_{im} = \sum_{d \in \mathbf{x}_{im}} \sigma(\mathbf{t}_{imd})) \tag{8}$$

$$\mathbf{c}_{im} = \Big(1 + \exp\Big(-\Big(\mathbf{n}_{im} \underset{j \in B_{im}}{\circledast} \mathbf{n}_{ij}\Big)\Big)\Big)^{-1} \tag{9}$$

where ϕ_i represents the Graph Neural Network applied to the modality i, which outputs the representation of each node d of graph m in the modality i, σ is the Gumbel Softmax, and $\mathbf{n}$ represents the sum over the node concept of the graph m. Therefore, in our solution, the graph concept is related to the occurrences of each node concept.

The issue with CEM is that when it aggregates node concepts, there is no one-to-one mapping between a set of node concepts and graph concepts. This could lead to giving the wrong concept to a graph. Figure 4 shows an example of a situation where two different graphs end up with the same concepts.

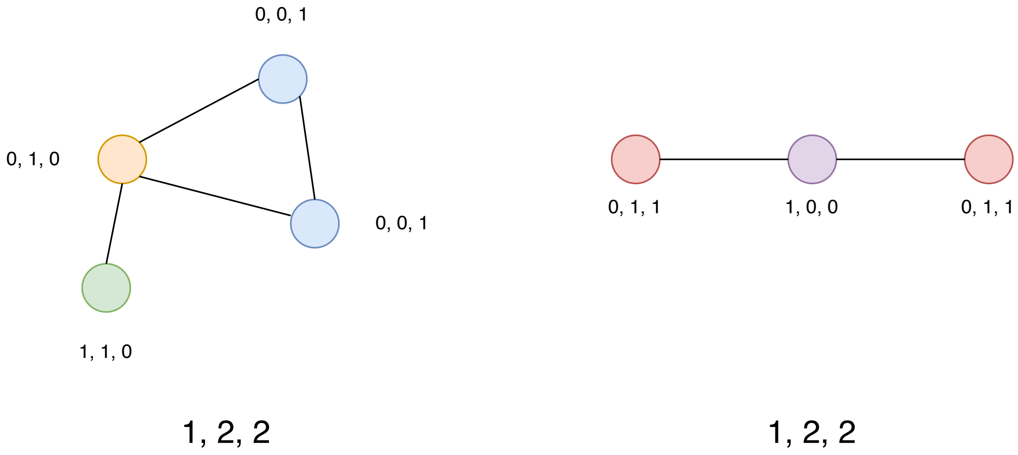


Figure 4: An example of two different graphs with a different set of node concepts described with the same graph concept.

A.3 CODE, LICENCES AND RESOURCES

Libraries For our experiments, we implemented all baselines and methods in Python 3.9 and relied upon open-source libraries such as PyTorch 2.0 (Paszke et al., 2019) (BSD license), Pytorch Geometric 2.3 (Fey & Lenssen, 2019) (MIT license) and Sklearn 1.2 (Pedregosa et al., 2011) (BSD license). In addition, we used Matplotlib (Hunter, 2007) 3.7 (BSD license) to produce the plots shown in this paper and Dtreeviz[1] 2.2 (MIT license) to produce the tree visualisations. We will publicly release the code to reproduce all the experiments under an MIT license.

Resources We run all the experiments on a cluster with 2x AMD EPYC 7763 64-Core Processor 1.8GHz, 1000 GiB RAM, and 4x NVIDIA A100-SXM-80GB GPUs. We estimate that all the experiments require approximately 50 GPU hours to be completed.

B DATASET DETAILS

We design the experiments to understand the potentiality of the proposed solution, described in Section 2. Specifically, We create a synthetic dataset that can validate all the contributions of our method, design two other tasks using existing datasets and use an existing dataset and task to test it in less constrained situations. Each dataset is split into the train (80%) and test set (20%).

B.1 XOR-AND-XOR

We design a synthetic dataset (XOR-AND-XOR) that contains two modalities: tabular data and graphs. The first contains 6 random bits, but only the first two are meaningful. The second modality contains one of 4 kinds of graphs: (i) 10 nodes which are not connected; (ii) 4 nodes connected in a circle and 6 not; (iii) 6 nodes connected in a circle and 4 not; (iv) 4 nodes connected in a circle, 6 other nodes connected in another circle and the two connected by an edge. They also have a few random edges, and the initial nodes' feature is its betweenness centrality. Each of these graphs can be associated with a combination of the two significant bits of the table, as Figure 5 shows.

In this dataset, there is a local task and a global task. The local one is intra-modality and is the XOR operator between the two meaningful bits or between the above-explained translation from graphs to bits. On the other hand, the global task corresponds to the AND operator between the result of the local tasks. The global task cannot be solved using just one modality, so both pieces of information are needed to classify each entry correctly. However, on this dataset, we do not make supervision on the local task available, letting the model understand it.

---

[1] `https://github.com/parrt/dtreeviz`

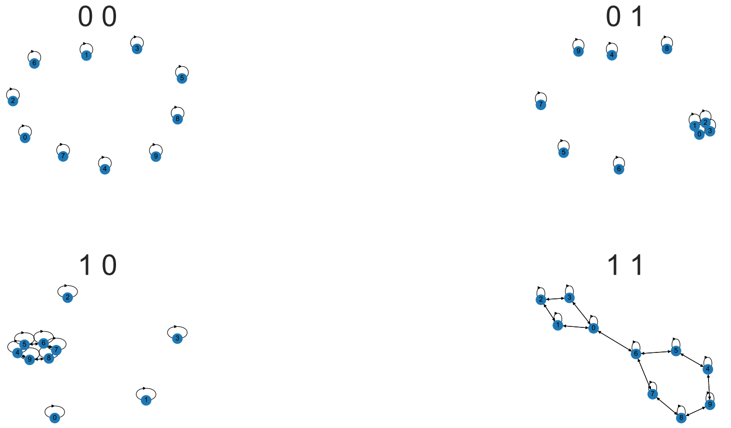
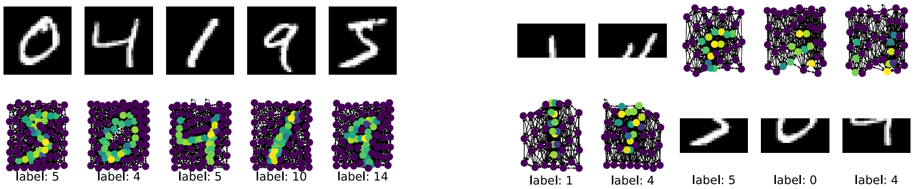


Figure 5: Examples of the conversion from the four main families of graphs to the meaningful bits of the tabular data in the XOR-AND-XOR dataset. In the dataset, they have some additional random edges.



(a)

(b)

Figure 6: (a) Examples from the MNIST+Superpixels dataset. The shown label is related to the task, which is the sum of the two digits. (b) Examples from the HalfMNIST dataset. The shown label is related to the task, which is the digit represented by joining both parts. Each half can be represented with one of the two modalities.

The entire dataset contains 1000 samples for each modality, the translations from one modality to the other for both modalities and the labels relative to the global task.

B.2 MNIST + SUPERPIXELS

The second dataset (MNIST+Superpixels) consists in predicting the sum of two digits described in two different ways. One is in the shape of an MNIST image (Deng, 2012), and the other is represented as a superpixel graph of another MNIST image (Monti et al., 2016). Here the local task is correctly classifying the single digit, while the global is predicting the sum of the two. Figure 6a shows five samples from this dataset, including the global label.

In this task, local supervision is available. Therefore, the dataset contains 60000 couple of digits, both described as a graph and as an image (during training, we use the graph of digit 1 and the image of digit 2), the labels for the local tasks and the ones for the global task.

B.3 HALF-MNIST

The third experiment uses another dataset containing the same modalities as the previous one (MNIST and MNIST Superpixels) but differently. In this case, the task is to predict the single digit, but one half is in the shape of an image, and the other half is described as a superpixel graph. It is important

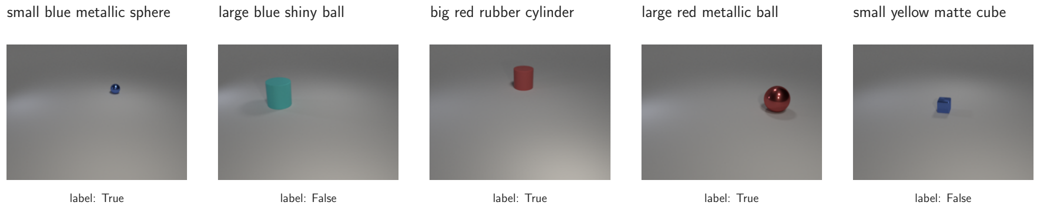


Figure 7: Examples from the CLEVR dataset, where there is a text caption and an image of an object. The label is True if the caption correctly describes the image, otherwise is False.

to say that some of the upper halves are images, some are graphs, and the same last for the bottom halves. Figure 6b shows five samples from this dataset.

Here, the global task is the same as the local one, but it is possible to use more information from a different modality to solve it. Therefore, this dataset contains 60000 digits described as graphs and images (only half image and half graph are used during training) and labels for the global task.

B.4 CLEVR

The last dataset used is a version of CLEVR we generate using the official repository. [2] Our version is inspired by Saqur & Narasimhan (2020), where each image contains one object and the relative caption can match or not the image. Specifically, in our case, every caption contains four attributes used to describe the scene in the image. They are the shape (sphere, cylinder, cube), the size (big, small), the colour (green, yellow, gray, red, purple, cyan, blue, brown) and the material (matte, metallic) of the object. The task is to predict if the caption correctly describes the image. If the label is equal to True, we consider that a connection between the two modalities, as it means that the two modalities represent the same object. Figure 7 shows five samples taken from this dataset, the top row represents the captions, while the bottom is about the images.

In this situation, there is no local task. Therefore, this dataset contains 8320 couple of captions and images, their translation in the other modality and the labels.

C MODELS DETAILS

In this section, we describe in detail the configuration of SHARCS used in each experiment. Then, we add only the missing or different information needed to build the other models used, as most of the details are in common between our solutions and baselines.

In general, single modality models used only the DL model inside of the respective g_i, with (or without) a sigmoid function, if it is a concept-based (concept-less) solution. Simple Multimodal and Relative representation solutions employ the DL models inside g_i and the label predictor f, while Concept Multimodal also uses batch scaling and the sigmoid inside g_i.

C.1 XOR-AND-XOR

On this task, we trained SHARCS with the end-to-end configuration, as we do not have local supervision. It is composed of two g_i concept encoder functions, one for each modality. To handle the

---

[2] https://github.com/facebookresearch/clevr-dataset-gen

graph modality, the DL model inside of g_1 is composed of 5 layers of Graph Convolutional Networks (Kipf & Welling, 2016) with LeakyReLU as the activation function. The input size is 1 as described in Appendix B.1, the hidden size of all the intermediate layers is 30, while the output dimension of g_1 is 7. On the other hand, a simple 2-layer MLP with a ReLU as the activation function is the DL model of g_2, which takes tabular data as input. The input size is 8, the hidden size is 30, and the output dimension is equal to 7. SHARCS uses Batch Normalisation as batch scaling and Sigmoid to compute concepts, but on the graph modality follows the approach described in Appendix A.2. The second set of concept encoders h_1 and h_2 are 2-layer MLPs with a ReLU as the activation function, with an input dimension of 8, as well as the hidden and output size. Finally, the label prediction function f is a 2-layer MLP with a ReLU as the activation function, with an input dimension of 16, a hidden size of 10 and an output dimension equals to the number of classes, which is 2.

An additional detail for single modality models is their label prediction function f_i, one for each modality, which is a 2-layers MLPs with a ReLU as the activation function, with an input dimension of 8, a hidden size of 10 and an output dimension of 2.

In terms of learning process, we used a Binary Cross Entropy Loss (BCELoss) with Logits (which incorporates a sigmoid layer before computing the BCELoss) as $\mathcal{T}$, a λ equals to 0.1, and at every iteration, we took 10% of randomly draw samples to compute the distance. Other hyperparameters used to train the models are the Batch Size used (64), the number of epochs (150) and the Learning Rate used by an Adam optimizer (0.001). However, we train the unimodality models of Relative representation models for 150 epochs and its label predictor function for other 150 epochs.

C.2 MNIST+SUPERPIXELS AND HALFMNIST

On MNIST+Superpixels and HalfMNIST, we used an almost identical setup. We trained SHARCS with the local pre-training configuration, as we have local supervision. It is composed of two g_i concept encoder functions, one for each modality. To handle the graph modality, the DL model inside of g_1 is composed of 2 layers of SplineCNN (Fey et al., 2018) with ELU as the activation function, similar to the SplineCNN model described in the original paper. Therefore, a max pooling operator based on the Graclus method (Dhillon et al., 2007) is applied after every layer. The input size is 1, the hidden size of all the intermediate layers is 32, and the output dimension of g_1 is 12. On the other hand, a Convolutional Neural Network is the DL model of g_2. It is composed of the following layers: a Convolutional Layer (input channel=1, output channel=16, kernel size=5, padding=2, stride=1), a ReLU, a MaxPool with a kernel size of 2, a Convolutional Layer (input channel=16, output channel=16, kernel size=5, padding=2, stride=1), a ReLU, a MaxPool with a kernel size of 2, then the output is flattened and taken as input from a 2-layer MLP with a ReLU as the activation function, with a hidden dimension of 64 and output size of 12. Moreover, SHARCS uses Batch Normalisation as batch scaling and sigmoid to compute concepts, but on the graph modality follows the approach described in Appendix A.2. The second set of concept encoders h_1 and h_2 are 2-layer MLPs with a ReLU as the activation function, with an input dimension of 12, as well as the output size and a hidden size of 64. Finally, the label prediction function f is a 2-layer MLP with a ReLU as the activation function, with an input dimension of 24, a hidden size of 128 and an output dimension equals to the number of classes, which is 19 for MNIST+Superpixels and 10 for HalfMNIST. As we apply the local pre-training configuration, in the first part of the training, we used some local label predictor function f_i, one for each modality. They are 2-layer MLPs with a ReLU as the activation function, with an input dimension of 12, a hidden size of 64 and an output dimension equals to the number of classes of the local task, which is 10 for both datasets. Other unimodal baselines also use these local label predictor functions.

Regarding the learning process, we used a BCELoss with Logits both with local and global tasks, a λ equals to 0.1, and at every iteration, we took 10% of randomly drawn samples to compute the distance. Other hyperparameters used to train the models are the Batch Size used (64), the number of epochs used to pretrain the unimodal models (15) and the additional epochs used to train the second part of SHARCS (15). The learning rate used by the Adam optimiser is equal to 0.01 for the Graph Neural Network and 0.001 for all the other layers of the model. However, we train the unimodality models of Relative representation models for 15 epochs and its label predictor function for other 20 epochs.

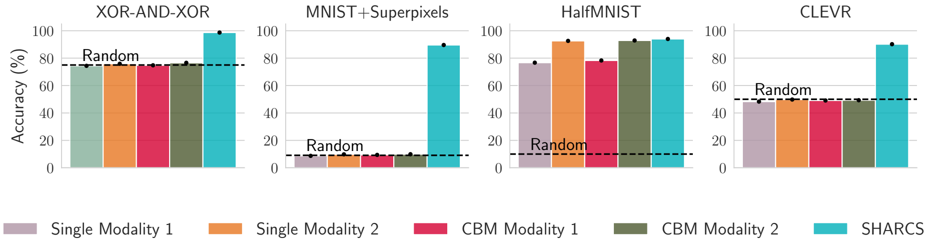


Figure 8: Accuracy of unimodal models and SHARCS on all datasets. SHARCS outperforms all the other models on all tasks.

C.3 CLEVR

On this task, we trained SHARCS with the sequential configuration, as we do not have local supervision and want to discover local concepts that are not influenced by the other modality. It is composed of two g_i concept encoder functions, one for each modality. To handle the image modality, the DL model inside of g_1 is a pretreated ResNet18 (He et al., 2015), followed by a Dense layer that reduced the output size of the ResNet to 24. On the other hand, a simple 2-layer MLP with a ReLU as the activation function is the DL model of g_2, which takes the TF-IDF representation of the caption received as input. The input size is 22, the hidden size is 48, and the output dimension is equal to 24. SHARCS uses Batch Normalisation as batch scaling and sigmoid to compute concepts. The second set of concept encoders h_1 and h_2 are 2-layer MLPs with a ReLU as the activation function, with an input dimension of 24, as well as the hidden and output size. Finally, the label prediction function f is a 2-layer MLP with a ReLU as the activation function, with an input dimension of 48, a hidden size of 10 and an output dimension equals to the number of classes, which is 2.

An additional detail for single modality models is their label prediction function f_i, one for each modality, which is a 2-layers MLPs with a ReLU as the activation function, with an input dimension of 24, a hidden size of 24 and an output dimension of 2.

In terms of learning process, we used a BCELoss with Logits, a λ equals to 0.1, and at every iteration, we took the samples with the label equals to True out of 20% of randomly drawn samples to compute the distance. Other hyperparameters used to train the models are the Batch Size used (64), the number of epochs used by all models and in the first part of the training of SHARCS (30), the additional epochs used in the second part of the training of SHARCS (20) and the Learning Rate used by an Adam optimizer (0.001). In addition, we train the unimodality models of Relative representation models for 30 epochs and its label predictor function for other 20 epochs.

D ADDITIONAL RESULTS

This section includes additional results and consideration of the experiments presented in Section 3.

Broader Impacts We do not believe this approach can have a direct harmful impact when applied in AI systems. On the contrary, it can positively influence the development of models for safety-critical domains, such as healthcare.

Detailed results of experiments

Figure 8 shows the performance of unimodal models compared to SHARCS in all tasks. It is clear how 3 out of the 4 datasets we designed are not solvable by unimodal models, proving our design choice. Furthermore, Table 4 shows the Accuracy for all the models trained and the Completeness Score for the multimodal interpretable models. It gives more detailed results and compares together all the trained models. On the other hand, Table 5, shows the result of an analysis we performed on CLEVR, where we checked for each model which characteristics of the retrieved sam- ple matched with the ones of the object used as the source.

Interpretability We present the visual results for each dataset to give a better idea of the performance of our solution. We show the retrieved examples per modality in each dataset, the learnt shared

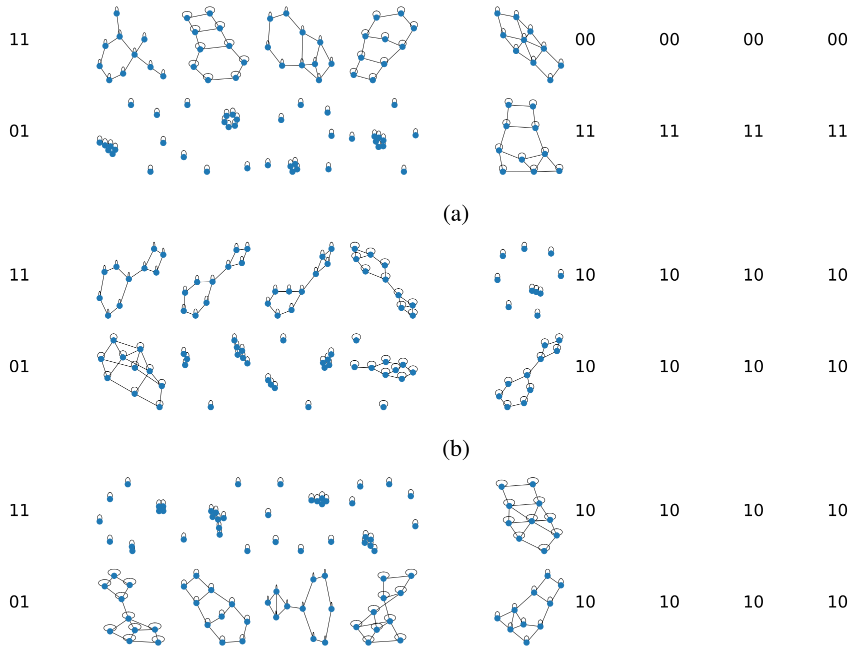
Table 4: Accuracy (%) and Completeness Score (%) of SHARCS compared to non-interpretable unimodal models (Simple Modality 1 and Simple Modality 2), non-interpretable multimodal models (Simple Multimodal and Relative representation), interpretable unimodal models (CBM Modality 1 and CBM Modality 2) and interpretable multimodal baselines (Concept Multimodal). Generally, SHARCS achieves better (or comparable) performance than the other baselines, producing better and more compact concepts.

Model	**XOR-AND-XOR**		**MNIST+SuperP.**		**HalfMNIST**		**CLEVR**	
	Acc.	Compl.	Acc.	Compl.	Acc.	Compl.	Acc.	Compl.
Mod 1	74.4 ± 0.7	-	8.7 ± 0.1	-	76.7 ± 0.2	-	48.3 ± 0.3	-
Mod 2	75.9 ± 1.4	-	9.8 ± 0.1	-	92.6 ± 0.2	-	49.8 ± 0.1	-
CBM 1	74.8 ± 0.0	-	9.4 ± 0.1	-	78.3 ± 0.1	-	49.1 ± 0.5	-
CBM 2	76.6 ± 1.3	-	9.9 ± 0.2	-	92.9 ± 0.1	-	49.3 ± 0.4	-
Simple	99.3 ± 0.5	-	86.6 ± 3.0	-	94.2 ± 0.2	-	59.5 ± 9.5	-
Concept	99.0 ± 0.8	96.2 ± 1.2	88.2 ± 0.1	78.9 ± 1.4	93.9 ± 0.0	91.3 ± 0.1	90.1 ± 1.0	$\mathbf{82.3} \pm 1.2$
Relative	$\mathbf{99.5} \pm 0.3$	-	80.4 ± 0.2	-	$\mathbf{95.6} \pm 0.1$	-	48.7 ± 0.5	-
SHARCS	98.7 ± 0.5	$\mathbf{98.0} \pm 1.2$	$\mathbf{89.6} \pm 0.1$	$\mathbf{88.7} \pm 0.2$	94.0 ± 0.1	$\mathbf{92.6} \pm 0.3$	$\mathbf{90.2} \pm 0.2$	81.5 ± 1.1

Table 5: Accuracy (%) of Relative representation, Concept Multimodal and SHARCS in retrieving a specific characteristic in a modality using the other. SHARCS attains higher figures than other models on every characteristic.

Model	**Modality**	**Shape**	**Size**	**Material**	**Color**	**Mean**
Concept	Text	31.7 ± 2.3	46.0 ± 3.0	52.1 ± 0.2	16.2 ± 3.7	36.5 ± 0.6
	Image	30.0 ± 0.2	45.4 ± 5.3	51.3 ± 2.6	10.8 ± 1.3	34.3 ± 0.6
Relative	Text	29.9 ± 1.0	50.5 ± 0.7	50.0 ± 0.6	13.3 ± 1.2	35.9 ± 0.3
	Image	33.0 ± 1.0	49.6 ± 0.2	49.0 ± 1.0	11.1 ± 0.7	35.6 ± 0.2
SHARCS	Text	$\mathbf{56.8} \pm 1.4$	$\mathbf{63.6} \pm 3.5$	$\mathbf{53.9} \pm 1.8$	$\mathbf{30.2} \pm 5.6$	$\mathbf{51.1} \pm 2.0$
	Image	$\mathbf{51.4} \pm 2.4$	$\mathbf{61.5} \pm 1.7$	$\mathbf{53.4} \pm 2.6$	$\mathbf{27.5} \pm 4.0$	$\mathbf{48.5} \pm 1.9$

space and the decision tree. The results presented here for the same dataset included in Section 3 are run with a different random seed to show how solid the performance of SHARCS is. Figure 9 shows the retrieved examples by SHARCS, Relative Representation and Concept Multimodal in the XOR-AND-XOR dataset. In particular, in Figure 9a, it is interesting to see how SHARCS retrieves tabular data that are not constrained to be closer but have the same semantic meaning for the local task, which is False in the XOR operator. Figure 10 illustrates the same experiments with MNIST+Superpixels and Figure 11 with HalfMNIST. Finally, Figure 12 shows the retrieval capability of these models on the CLEVR dataset. In all these experiments, it can be seen that the quality of the retrieved examples is higher than the others, where the Relative Representation is not always accurate and Concept Multimodal resembles random retrieval. The second set of images visually confronts the shared space learnt by SHARCS and Concept Multimodal. For this purpose, we visualise the tSNE representation of the shared concepts for SHARCS and the local concepts for Concept Multimodal. Figure 13 shows these shared spaces for the MNIST+Superpixels dataset, Figure 14 for HalfMNIST and Figure 15 for CLEVR. It is clear how the concept representation learnt by SHARCS for one modality overlaps with that for the other, especially when considering semantically similar examples from different modalities that are closer in the space representation. All these results are expected by design since we force the model to produce the shared space with these properties. Finally, part of the decision trees used to compute the completeness score is visualised. At every split, it shows the concept that is considered to make the decision, and it can be active (right branch) or non-active (left branch). If the node is not the roof, it also shows three samples with the highest (concept active) or the lowest (concept non-active) value for the concepts of the previous split, among the ones that respect all the previous split conditions. Each leaf shows the class distribution of the samples that it represents, in addition to the most characteristic samples. Moreover, the root of the tree uses the most influential concept for the classification task, as it is by definition the one that brings the highest Information Gain, and the same is applicable to the following splits. For example, Figure 16 shows the decision tree used in the XOR-AND-XOR dataset. Specifically, you can see that if Concept 11 is active, the prediction is always Class 0 (False). As you can see, if a sample has Concept 11 active, it means that it has both tabular significant digits equal to 1, which implies that the local XOR



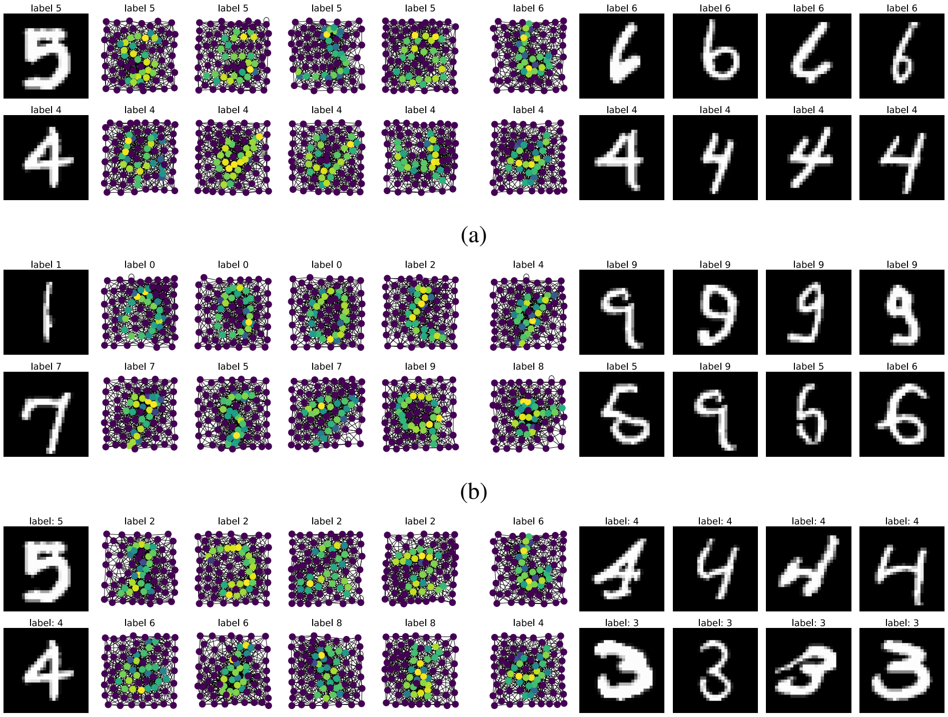
(a)

(b)

(c)

Figure 9: Retrieval examples obtained by (a) SHARCS, (b) Relative representation, and (c) Concept Multimodal on the XOR-AND-XOR dataset. The top two rows are samples of retrieved graphs using tabular data, while the bottom two are retrieved tabular entries using graph samples.

operation is False and as a consequence the global AND operation is False, no matter what is the other modality. Furthermore, the following split is focused on Concept 4, which is curiously the corresponding concept in the shared space of the graph modality for Concept 11 (7 is the number of concepts per modality, so 11 - 7 = 4). This split represents the same underlying idea of the previous one but for the graph modality. If the concept is active, it means that the graph is connected (False in the local XOR operation). Therefore, it shows also how the concepts from one modality are related and translated into the other, confirming that the concept shared space created is meaningful. Finally, Figure 17 shows part of the first three layers of the Decision tree used in CLEVR.

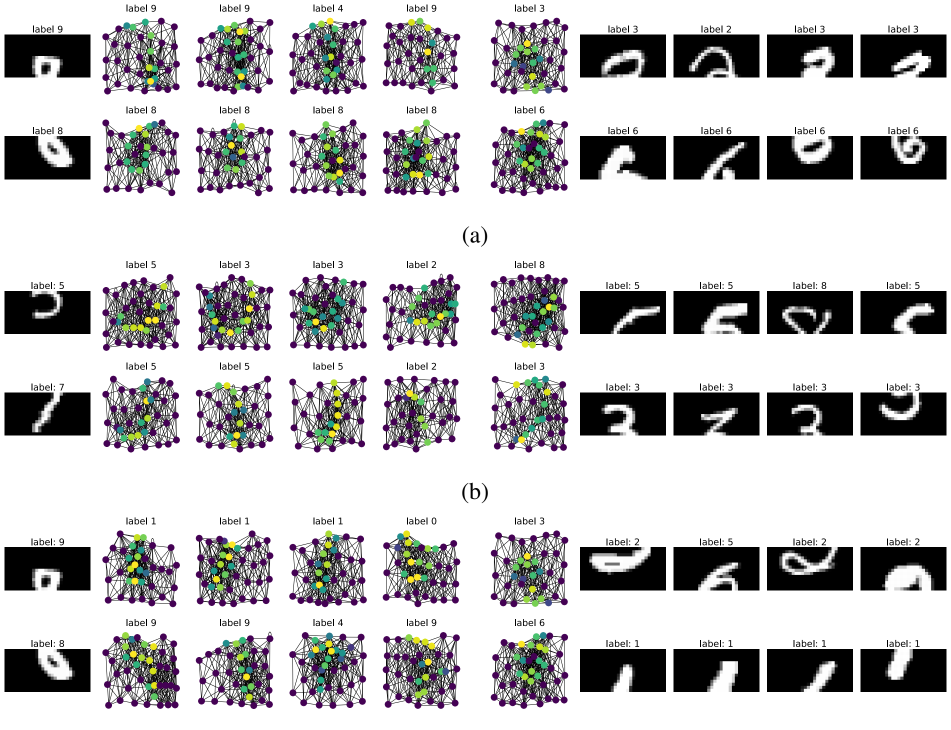


(a)

(b)

(c)

Figure 10: Retrieval examples obtained by (a) SHARCS, (b) Relative representation, and (c) Concept Multimodal on the MNIST+Superpixels dataset. The top two rows are samples of retrieved graphs using images, while the bottom two are retrieved images using graph samples.

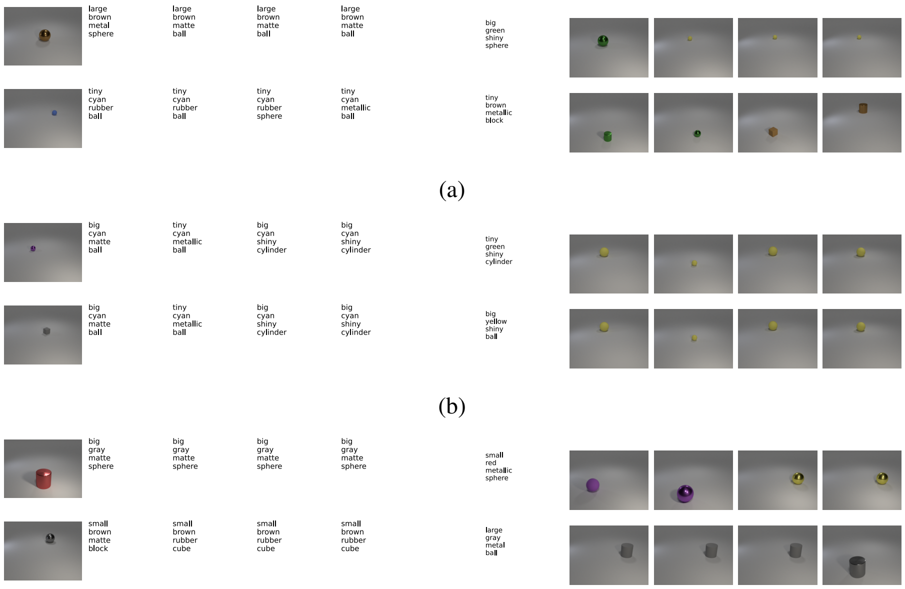


(a)

(b)

(c)

Figure 11: Retrieval examples obtained by (a) SHARCS, (b) Relative representation, and (c) Concept Multimodal on the MNIST+Superpixels dataset. The top two rows are samples of retrieved graphs using images, while the bottom two are retrieved images using graph samples.

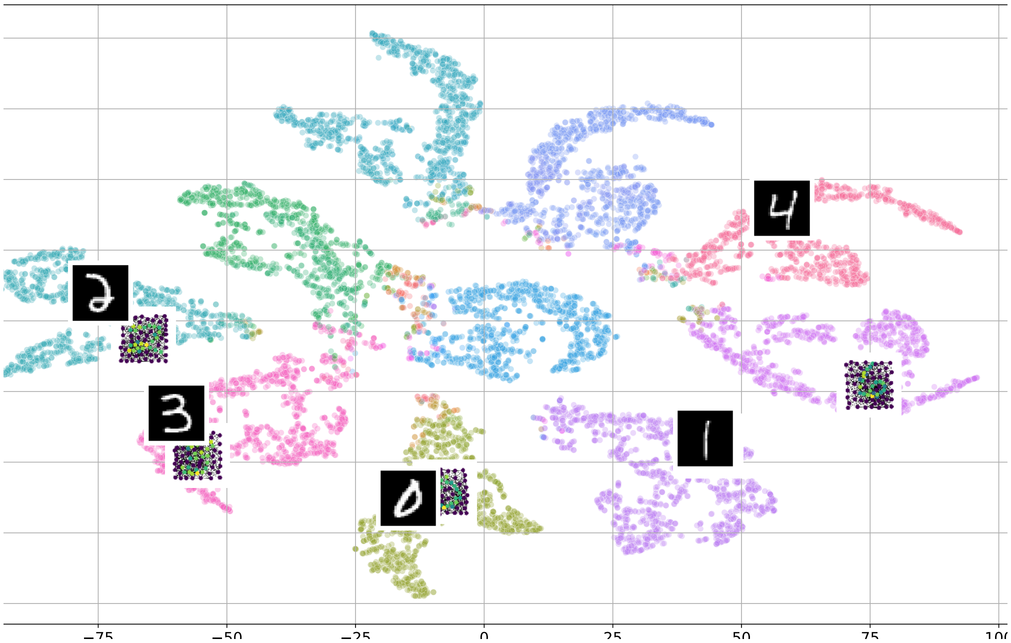


(a)

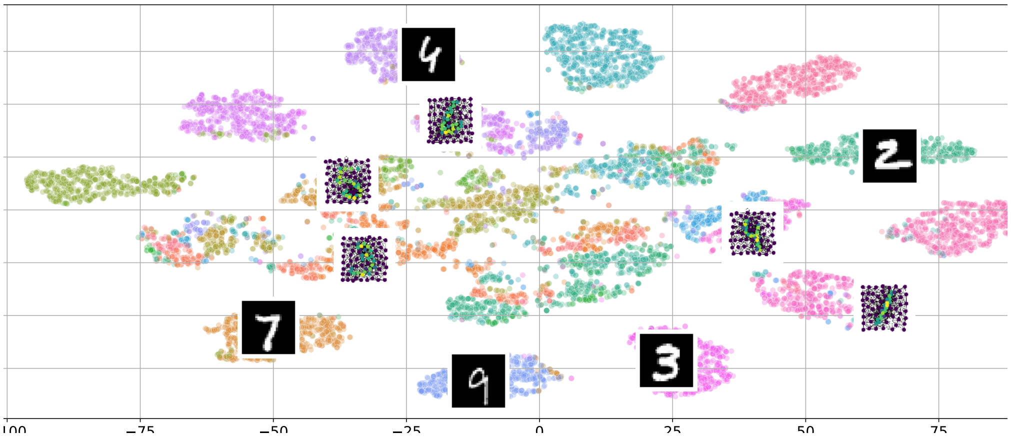
(b)

(c)

Figure 12: Retrieval examples obtained by (a) SHARCS, (b) Relative representation, and (c) Concept Multimodal on the CLEVR dataset. The top two rows are samples of retrieved text using images, while the bottom two are retrieved images using graph samples.

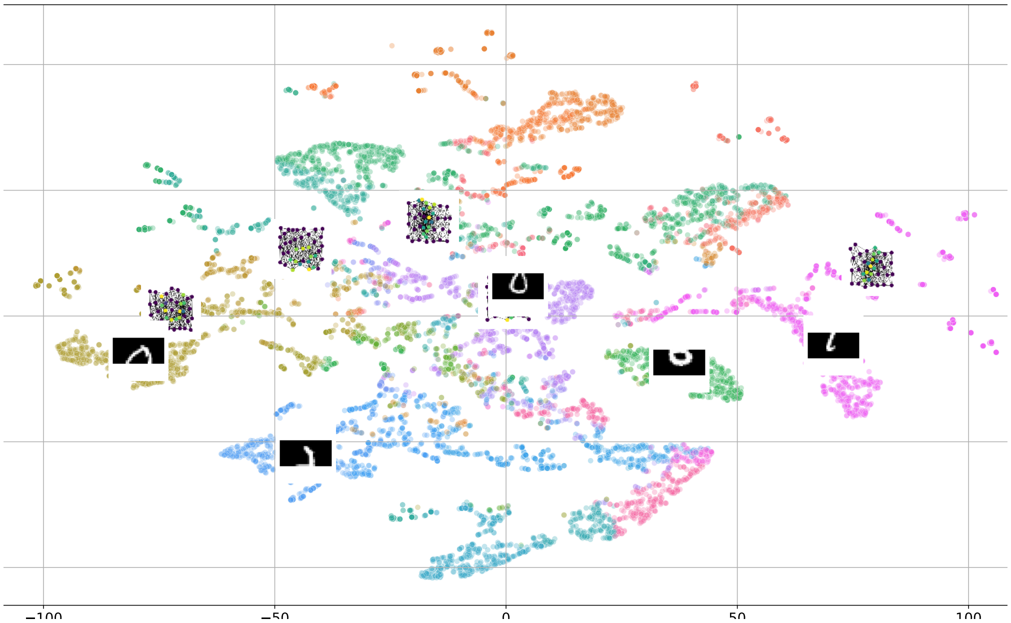


(a)

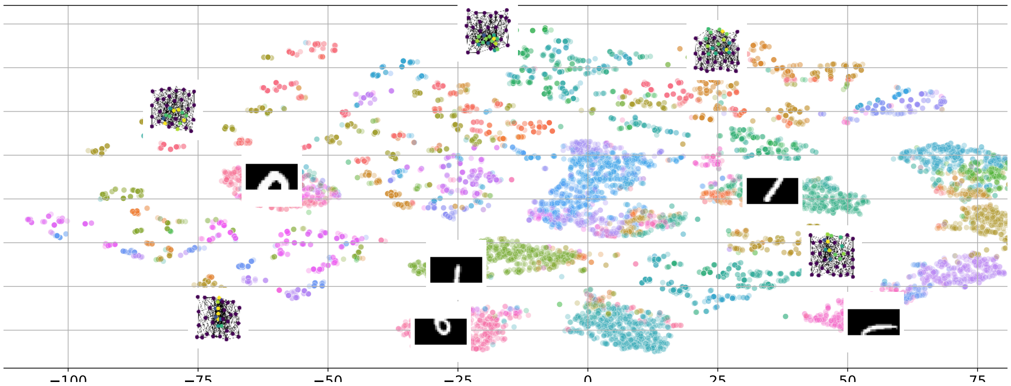


(b)

Figure 13: tSNE plot of the concept space. The images represent the centroid of the top-5 common concepts per modality in the MNIST+Superpixels dataset (a) SHARCS (b) Concept Multimodal

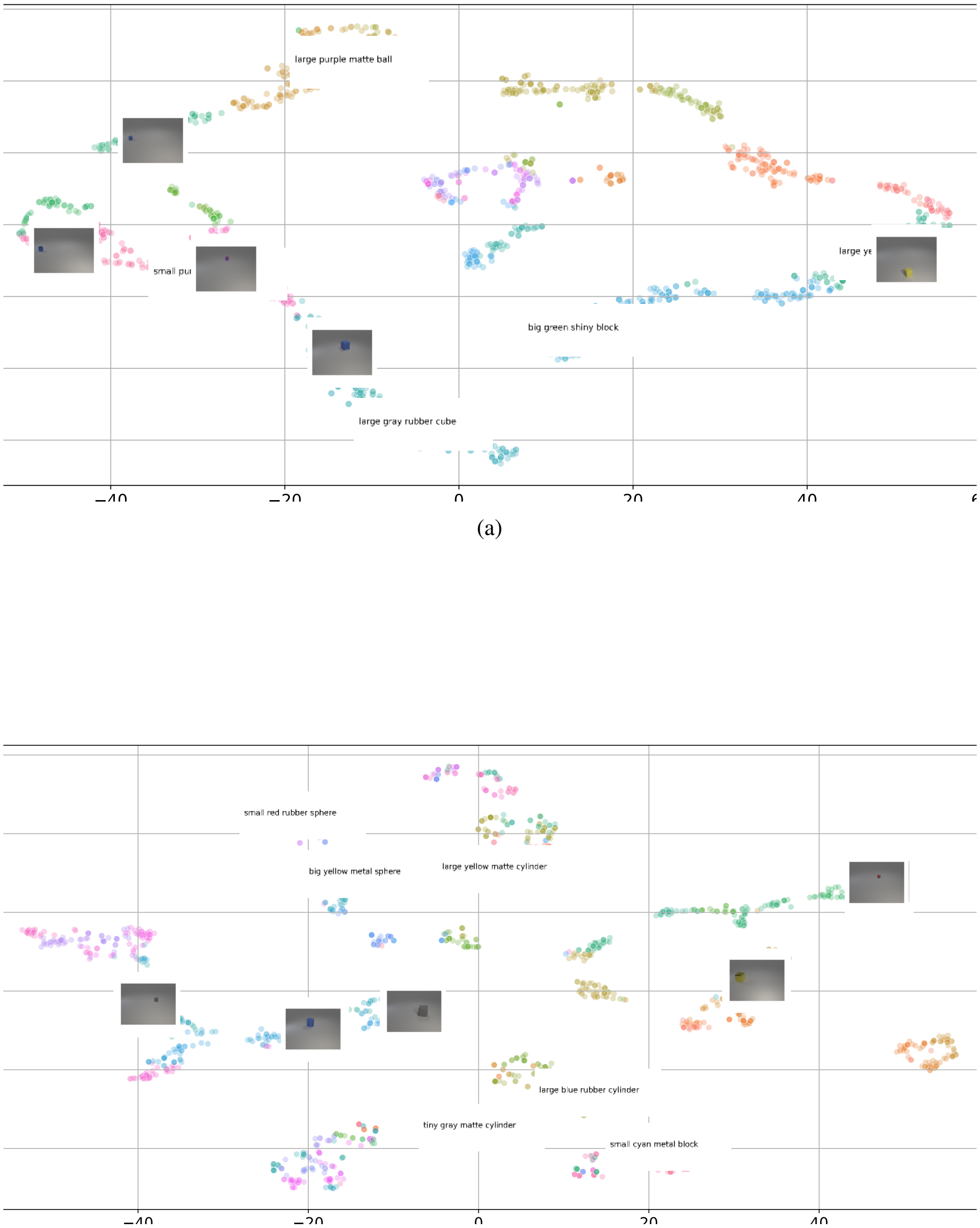


(a)



(b)

Figure 14: tSNE plot of the concept space. The images represent the centroid of the top-5 common concepts per modality in the HalfMNIST dataset (a) SHARCS (b) Concept Multimodal



(a)

(b)

Figure 15: tSNE plot of the concept space. The images represent the centroid of the top-5 common concepts per modality in the CLEVR dataset (a) SHARCS (b) Concept Multimodal

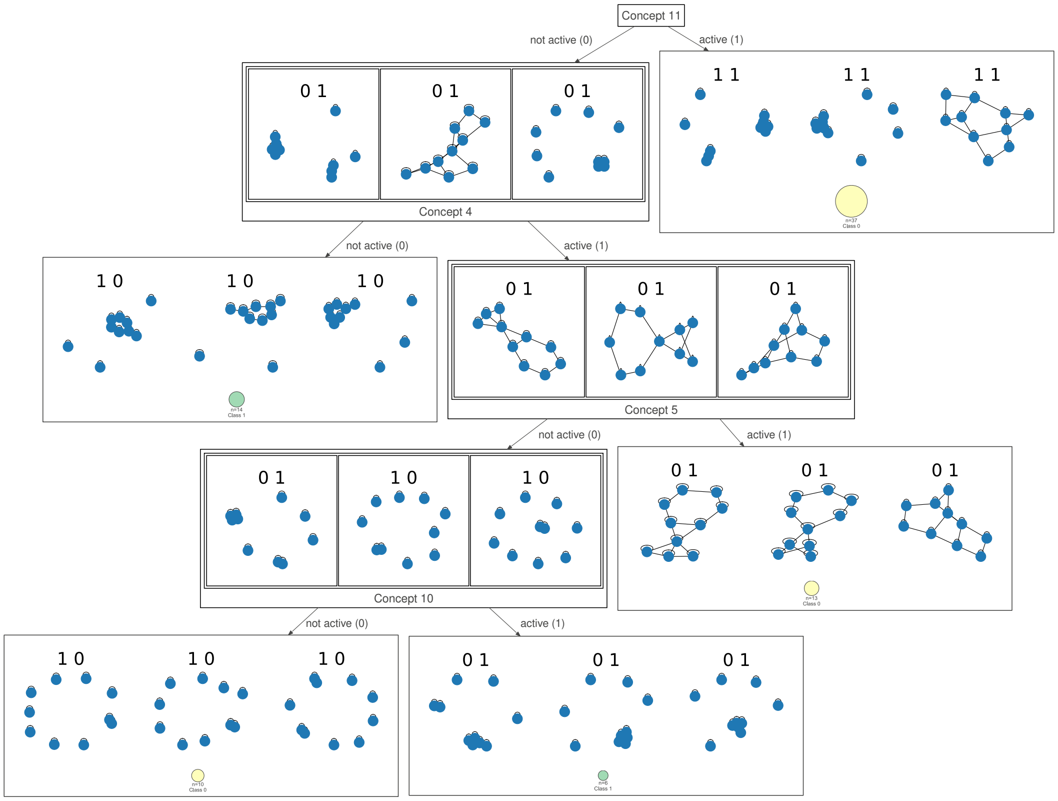


Figure 16: Decision tree visualisation of SHARCS concepts on the XOR-AND-XOR dataset. Every split shows the combined concept closer to the cluster's centroid lower and greater than the splitting criteria. In addition, each leaf shows the class distribution of the samples that it represents.

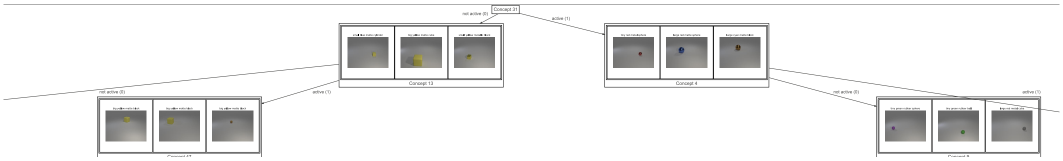


Figure 17: Visaulisation of part of the first 3 layers of a Decision tree trained on SHARCS concepts on the CLEVR dataset. Every split shows the combined concept closer to the cluster's centroid lower and greater than the splitting criteria. In addition, each leaf shows the class distribution of the samples that it represents.