Search for Concepts: Discovering Visual Concepts Using Direct Optimization

Pradyumna Reddy[1]
https://preddy5.github.io/
Paul Guerrero[2]
http://paulguerrero.net/
Niloy J. Mitra[1, 2]
http://www0.cs.ucl.ac.uk/staff/n.mitra/index.html

[1] University College London
[2] Adobe Research London

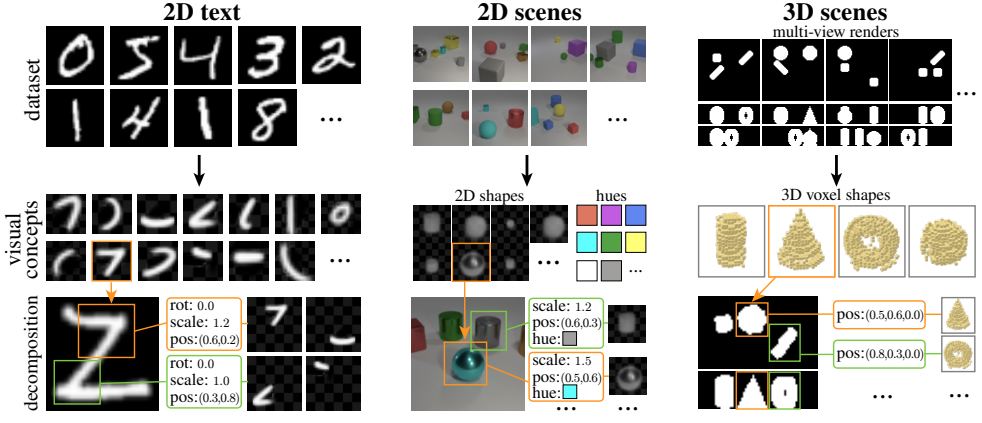

Figure 1: We present a search-and-learn paradigm that starts from an unlabeled dataset and a known image formation model, and learns *visual concepts* in the form of a dictionary of base elements along with their placement parameters to best explain the input dataset. Here we show results on the MNIST dataset, Clevr renderings, and a 3D sprite dataset.

Abstract

Finding an unsupervised decomposition of an image into individual objects is a key step to leverage compositionality and to perform symbolic reasoning. Traditionally, this problem is solved using amortized inference, which does not generalize beyond the scope of the training data, may sometimes miss correct decompositions, and requires large amounts of training data. We propose finding a decomposition using direct, unamortized optimization, via a combination of a gradient-based optimization for differentiable object properties and global search for non-differentiable properties. We show that using direct optimization is more generalizable, misses fewer correct decompositions, and typically requires less data than methods based on amortized inference. This highlights a weakness of the current prevalent practice of using amortized inference that can potentially be improved by integrating more direct optimization elements.

1 Introduction

Reconstructing an input signal as a *composition* of different meaningful parts is a long standing goal in data analysis. The ability to decompose a signal into meaningful parts not only results in an interpretable abstraction, but also improves sampling efficiency and generalization of learning-based algorithms. Notable classical unsupervised methods for part/parameter decomposition include Principal Component Analysis (PCA), Independent Component Analysis (ICA), Dictionary Learning, Matching pursuits. In computer vision, the output of these methods are regularly used for classification, denoising, texture propagation, etc.

In the context of images, amortised optimization with neural networks is currently the unquestioned practice in self-supervised decomposition [2, 15, 28, 40]. Amortised optimization is fast and has the potential to avoid local minima, but can be inexact and is known to struggle with more complex settings. For this reason, several well-known works like AlphaZero, AlphaGo, and AlphaFold mix *direct search* with amortised optimization.

We pose the question *if direct optimization can also benefit unsupervised scene decomposition*. In this paper, we learn unsupervised *visual concepts* from data using a direct search approach instead of amortized inference. By visual concepts, we refer to a small dictionary of (unknown) parameterized objects, that are acted upon by parameterized transformations (e.g., translation, rotation, hue change), resulting in transformed instances of the visual concepts called *elements*, which are rendered into a final image using a given image formation model. Given access to a sufficiently large dataset, we demonstrate that interpretable visual concepts naturally emerge as they allow efficient explanation of diverse datasets.

While the direct search problem for visual concepts is computationally ill-behaved, we show that splitting the problem into subtasks not only results in computationally efficient problems but also provides, as empirically observed, near optimal solutions. Particularly, we alternate between solving for the dictionary of visual concepts and their parameterized placement across any given image collection. We show that this approach has several advantages: using an optimization to perform the decomposition, instead of a single forward pass in a network, (i) allows finding solutions that the network missed and improves decomposition performance and (ii) often requires less training data than amortized inference and produces (iii) fully interpretable decompositions where elements can be edited by the user. For example, in Figure 1, our method extracts strokes from MNIST digits, 2D objects from Clevr images, and 3D objects from a multi-view dataset of 3D scene renders.

We evaluate on multiple data modalities, report favorable results against different SOTA methods on multiple existing datasets, and extract interpretable elements on datasets without texture cues where deep learning methods like Slot Attention suffer. Additionally, we show that our method improves generalization performance over supervised methods.

2 Related Work

Supervised methods. The well-studied problems of instance detection and semantic segmentation are common examples of supervised decomposition approaches. Due to the large body of literature, we only discuss some representative examples. Methods like Segnet [1] and many others [4, 47] have tackled the problem of semantic segmentation, decomposing an image into a set of non-overlapping masks, each labelled with a semantic category. Instance detection methods [11, 12, 36, 37] decompose an image into a set of bounding boxes, where each box contains a semantic object, while others [5, 25, 29] go further by detect-

ing relationship edges between objects, producing an entire scene graph. Mask-RCNN [17] proposed an architecture to perform both instance detection and the segmentation using a single network. Recently, there has been a rise of architectures that that explore set generation methods [3, 24, 27, 46] for decomposition. However, as the name indicates, supervised methods require access to different volumes of annotated data for supervision, and often fail to generalize to unseen data, beyond the scope of the distribution available for supervision.

Unsupervised methods. Prior to the rise of deep learning, methods [20] have been proposed to model an input signal as a composition of epitomes, which contain information about shape and appearance of objects in an input image, and further research also tried to represent objects and scenes as hierarchical graphs composed of primitives and their relationships [6, 48, 49, 50].

Several methods [10, 13, 18, 23, 28, 38] try to perform decomposition as routing in an embedding space. The decomposition performance of these methods is sensitive to the input data distribution and may completely fail on some common cases, as we show in Section 6. Recently, a method to decompose a 3D scene into multiple 3D objects was proposed [42]. However, the method is domain specific to 3D data. In a related line of research [2, 9, 14, 15], methods naturally encourage decomposing the input image into desirable sets of objects during learning. However, these methods are currently out-performed in most tasks by embeddings-space routing methods such as Slot Attention [28], and extending these methods to other domains is not straight forward. A differentiable decomposition method was recently proposed [35], however, extensive information about the content of the decomposed elements is needed as input.

Inspired by the use of compositionality in traditional computer graphics pipelines, recent generative methods for 3D scenes encourage object-centric representations, using 3D priors [8, 32, 33, 34, 43]. However, such ideas are yet to be extended beyond the generative setting. Decomposition is also discussed in more general AI-focused contexts [16]. Most recently, DTI-Sprites[31], Marionette [40] use a neural network to estimate a decomposition into a set of learned sprites, however the reliance on differential sampling and soft occlusion introduces local minima and undesirable artifacts.

3 Overview

Our goal is an unsupervised decomposition of an RGBA image I into a set of elements $E_1, \ldots, E_n$ that approximate I when combined using a given image formation function and where each element is an instance of a *visual concept*. A visual concept is an (unknown) object or pattern that commonly occurs in a dataset of images $\mathcal{I}$, such as Tetris blocks in a dataset of Tetris scenes, characters in a dataset of text images, or individual strokes in a dataset of hand-drawn characters. Figure 2 shows an overview of our approach.

An *element* E_i is a transformed instance of a visual concept. We use a parametric function e to create each element $E_i = e_{\mathcal{V}}(\theta_i)$, where $\mathcal{V}$ is a sparse dictionary of *visual concepts* extracted from an image dataset $\mathcal{I}$, and $\theta_i = (\tau_i, \phi_i)$ is a set of *per-element parameters*: τ_i is an index that describes which visual concept from $\mathcal{V}$ element E_i is an instance of, and ϕ_i are domain-specific transformation parameters, such as translations, rotations and scaling. The reconstructed image $\tilde{I} = h(E_1, \ldots, E_n)$ is computed from the elements with an image formation function h, which we assume to be given and fixed. Details on the parametric image and element representations are given in Section 4.

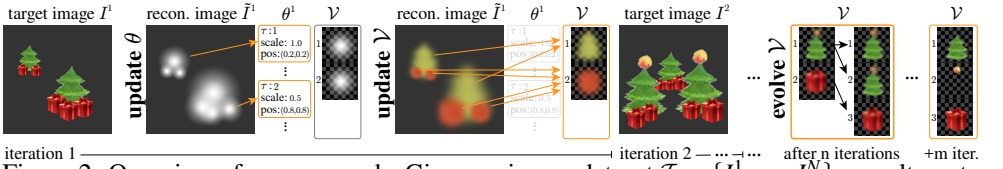

Figure 2: Overview of our approach. Given an image dataset $\mathcal{I} = \{I^1, \ldots, I^N\}$, we alternate between updating element parameters θ (such the choice of visual concept τ, its position, scale, etc.) and the visual concepts $\mathcal{V}$. We iterate over multiple target images I^k before evolving $\mathcal{V}$ by cloning concepts that are used often and have a large error. Another m iterations specialize the cloned concepts to better reconstruct the target images. The result of this iterative procedure is a library of visual concepts that can be used to efficiently decompose any image that makes use of similar visual concepts.

We learn visual concepts $\mathcal{V}$ by optimizing both $\mathcal{V}$ and the element parameters θ_i to reconstruct an image dataset $\mathcal{I}$. The dictionary $\mathcal{V}$ is shared between all images in $\mathcal{I}$, while θ_i has different values for each element. While jointly optimizing $\mathcal{V}$ and θ_i jointly is hard, we find that that optimizing one given the other is tractable. Thus, we alternate between optimizing $\mathcal{V}$ and θ_i. At test time, we keep the visual concepts fixed and only optimize for the element parameters that best reconstruct a given image. The optimization is described in Section 5.

In Sections 4 and 5, we first describe our approach with 2D alpha compositing as image formation function, and 3D voxel compositing is described in the supplement.

4 Parametric Elements and Images Formation

We approximate an image I with a set of parametric elements $\tilde{I} = h(E_1, \ldots, E_n)$, where each element is an instance of a visual concept.

Visual concepts. The dictionary of visual concepts $\mathcal{V} = (V_1, \ldots, V_m)$ defines a list of visual building blocks that can be transformed and arranged to reconstruct each image in a dataset $\mathcal{I}$. A visual concept V_j is defined as a small RGBA image patch of a user-specified size The size of the patch determines the maximum size of a visual concept. Depending on the application, we either set the number m of visual concepts manually, or learn the number while optimizing the dictionary. Section 5 provides more details on the optimization.

Parametric elements. Each element $E_i = e_{\mathcal{V}}(\theta_i)$ is a transformed visual concept. The parameters $\theta_i = (\tau_i, \phi_i)$ determine which visual concept is used with $\tau_i \in [1, m]$, and how the visual concept is transformed with the parameters ϕ_i: $E_i = e_{\mathcal{V}}(\theta_i) := T(V_{\tau_i}, \phi_i)$, where $T(V, \phi_i)$ transforms a visual concept V according to the parameters ϕ_i and re-samples it on the image pixel grid (samples that fall outside the area of the visual concept have zero alpha and do not contribute to the final image). The type of transformations performed depend on the application, and may include translations, rotations and scaling. See Section 5 for details.

Image formation function. The reconstructed image $\tilde{I}$ is an alpha-composite of the individual elements:

$$\tilde{I} = h(E_1, \ldots, E_n) := \sum_{i=1}^{n} E_i \prod_{j=1}^{i-1} (1 - E_j^{\mathrm{A}}) \tag{1}$$

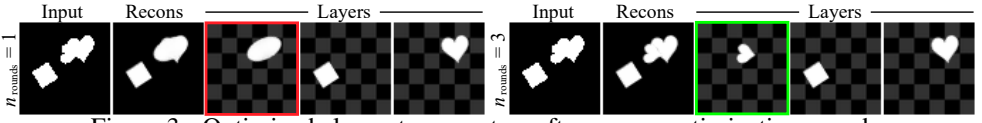

Figure 3: Optimized element parameters after n_{rounds} optimization rounds.

where A is the alpha channel and channel products are element-wise (with broadcasting to avoid a cluttered notation). We set the maximum number of elements n manually (n is between 4 and 45 in our experiments, depending on the dataset). Note that we can also use fewer elements than the maximum since the transformation T can place elements outside the image canvas, where they do not contribute to the image.

5 Optimizing Visual Concepts

We train our dictionary of visual concepts to reconstruct a large image dataset $\mathcal{I}$ as accurately as possible:

$$\arg\min_{\mathcal{V},\Theta} \mathcal{E}(\mathcal{V},\Theta) := \sum_{k=1}^{|\mathcal{I}|} \|I^k - \tilde{I}^k\|_2^2 \quad \text{with } \tilde{I}^k = h(e_{\mathcal{V}}(\theta_1^k),\ldots,e_{\mathcal{V}}(\theta_n^k)), \tag{2}$$

where $\mathcal{E}$ is the L_2 reconstruction error, θ_i^k denotes the parameters of element i in image k, and $\Theta = \{\theta_i^k\}$ is the set of element parameters in all the images of $\mathcal{I}$. Optimizing over $\mathcal{V}$ and Θ jointly is infeasible since the search space is not well-behaved. It is high-dimensional, and contains both local minima and discrete dimensions, such as those corresponding to the visual concept selection parameters τ_i. However, since the element parameters θ_i^k of different images k appear in separate linear terms, they can be independently optimized *given* $\mathcal{V}$. This motivates a search strategy that iterates over images I_k, and alternates between updating the visual concepts $\mathcal{V}$ and the element parameters θ_i^k.

5.1 Updating Element Parameters

While the element parameters of different images can be optimized independently, the optimal parameters of different elements in the same image depend on each other due to the alpha-composite. One possible approach to optimize all element parameters in an image given the visual concepts $\mathcal{V}$ is to use differentiable compositing [35]. However, we show that even a simpler greedy approach gives us good results. We initialize all elements to be empty and optimize the parameters of one element at a time, starting at θ_1. The optimum of θ_1 is likely to be the least dependent on the other elements, since it corresponds to the top-most element that is not occluded by other elements. We perform n_{rounds} rounds of this per-element optimization (typically $n_{\text{rounds}} = 3$ in our experiments). In Figure 3, we compare $n_{\text{rounds}} = 1$ versus $n_{\text{rounds}} = 3$.

The parameters in a single element determine the choice of visual concept in an element and its transformation. Gradient descent is not well suited for finding the element parameters, due to discontinuous parameters and local minima, but the dimensionality of the parameters is relatively small, between 2 and 4 in our applications. This allows us to perform a grid search in parameter space (see the supplementary material for grid resolutions). From the element parameters we typically use, the objective values are most sensitive to the translation parameters. A small translation can misalign a visual concept with the target image and

cause a large change in the objective, requiring us to use a relatively high grid resolution. Fortunately, we can speed up the search over the translation parameters considerably by approximating the original objective with a normalized correlation and formulating the grid search over translations as a convolution, which can be performed efficiently with existing libraries. Details on our convolution-based grid search are given in the supplement.

Element shuffling. We shuffle the order of elements to improve convergence and avoid local minima encountered due to our greedy per-element optimization. After optimizing all elements in an image, we move each element to the front position in turn, effectively changing the occlusion order. After each swap, we check the objective score and keep the swap only if it improves the objective.

5.2 Updating Visual Concepts

The dictionary of visual concepts $\mathcal{V}$ is shared across all images in the dataset $\mathcal{I}$. A parametric element $E_i = T(V_{\tau_i}, \phi_i)$ is differentiable w.r.t. the visual concept $V_{\tau_i} \in \mathcal{V}$, thus we can update $\mathcal{V}$ using stochastic gradient descent. After updating all element parameters of a given image $I \in \mathcal{I}$, we jointly update the visual concepts used in all elements of the image by taking one gradient descent step with the following objective: $\arg\min_{\mathcal{V}} \|I - h(e_{\mathcal{V}}(\theta_1), \ldots, e_{\mathcal{V}}(\theta_n))\|_2^2$, while keeping the element parameters θ_i fixed. To restrict the value domain of visual concepts V_j to the range $[0,1]$, we avoid functions that have vanishing gradients and use an approach inspired by periodic activation functions [39]: $V_j = \sin(30V'_j) * 0.5 + 0.5$, and we optimize over V'_j.

Evolving visual concepts. In many practical applications, we may not know the optimal number of visual concepts in advance. Choosing too many concepts may result in less semantically meaningful concepts, while too few concepts prevent us from reconstructing all images. We can learn the number of concepts along with the concepts using an evolution-inspired strategy. We start with a small number of visual concepts, and every n_{ev} epochs, we check how well each concept performs (n_{ev} is between 1 and 3 in our experiments, depending on dataset size). We replace concepts that incur a large reconstruction error and occur frequently with two identical child concepts. In the next epoch, these twin concepts will be used in different contexts, and will specialize to different patterns or objects in the images. Concepts that occur too infrequently, are removed from our dictionary. This results in a tree of visual concepts that is grown during optimization. The supplement describes thresholds for removing and splitting concepts and an illustration of the visual concept tree.

Composite visual concepts. A visual concept may appear in several discrete variations in the image dataset. For example, each Tetris block in the Tetris dataset may appear in one of 6 different hues. To avoid having to represent each combination of hue and block shape as separate visual concept, we could add a hue parameter to our element parameters. However, that would not give us explicit information about the discrete set of hues that appear in the dataset. Instead, we can split our library of visual concepts into two parts: $\mathcal{V}^s$ captures the discrete set of shapes in the dataset, and $\mathcal{V}^h$ captures the discrete set of hues as a dictionary of 3-tuples. The visual concept selector $\tau = (\tau^s, \tau^h)$ in the element parameters is then a tuple of indices, one index into the shapes and one into the hues, and the transformation function $T(V^s_{\tau^s}, V^h_{\tau^h}, \phi)$ combines shape $V^s \in \mathcal{V}^s$ with hue $V^h \in \mathcal{V}^h$ through multiplication. When using

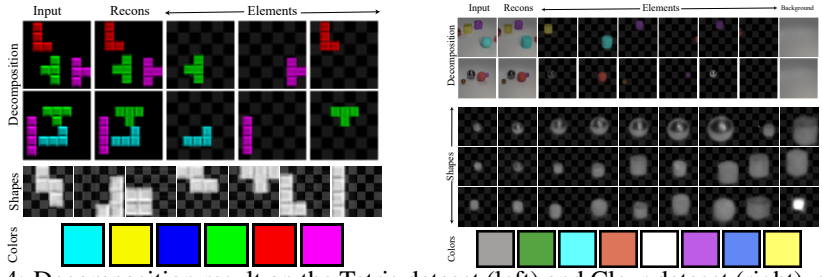

Figure 4: Decomposition result on the Tetris dataset (left) and Clevr dataset (right), showing composite concepts consisting of both learned shapes and learned hues. Since we do not include mirroring or scaling in our image formation model, mirrored objects and objects at different depths are learned as separate concepts.

these composite visual concepts, both shape and hue dictionaries are updated in the visual concept update step.

6 Results and Discussion

We demonstrate our method's performance on three tasks: (i) *unsupervised object segmentation*, where our unsupervised decomposition is used to segment an image that has a known ground truth segmentation, (ii) *cross-dataset reconstruction*, where we test the generalization performance of our method by training our visual concepts on one dataset and using them to decompose an image from a second dataset, and (iii) *3D scene reconstruction*, where we learn 3D concepts and reconstruct 3D scenes from multiple 2D views.

6.1 Object Segmentation

In this experiment, we measure the quality of our decompositions by comparing the segmentation induced by a decomposition to a known ground truth.

Datasets We test on four decomposition datasets: Tetrominoes, Multi-dSprites [30], Multi-dSprites adversarial, and Clevr6 [19]. Please refer to S.3 for more details on datasets, optimization parameters and other hyperparameters.

Baselines We compare our results with the state-of-the-art in unsupervised decomposition: Iodine [15], Slot Attention [28], DTI-Sprites [31] and Marionette [40], which use trained neural networks to perform the decomposition. Note that these baselines do *not* create an explicit dictionary of visual concepts except Marionette. While DTI-Sprites does create a dictionary, but individual slots entangle multiple different concepts, and the trained network is needed to disentangle them at inference time. Further, Iodine and Slot Attention do not generate explicit element parameters.

Results For all the datasets, we start with 3 visual concepts and evolve more concepts as needed. Table 1 shows quantitative comparisons, measuring the segementation performance with the Adjusted Rand Index (ARI) [44]. Our method achieves the best performance in the two variants of the M-dSprites dataset, with SlotAttention failing on the ad-

Table 1: **Segmentation performance.** Decomposition accuracy measured by the ARI metric. Second best values are underlined, while (*) uses the same element transformations and number of concepts as our method. See Section 6.1 for details.

	Tetrominoes	dSprites Color.	dSprites Bin.	dSprites Adv.	CLEVR6
IODINE	99.2 ± 0.4	76.7 ± 5.6	64.8 ± 17.2	-	98.8 ± 0.0
Slot Attention	99.5 ± 0.2	91.3 ± 0.3	69.4 ± 0.9	12.7 ± 1.1	**98.8 ± 0.3**
DTI-Sprites	**99.6 ± 0.2**	**92.5 ± 0.3**	$75.5^{*} \pm 0.4$	$75.3^{*} \pm 0.4$	97.2 ± 0.2
Ours	99.5 ± 0.1	90.6 ± 0.8	**85.1 ± 0.7**	**76.4 ± 2.4**	64.6 ± 0.8

Table 2: **Visual concept error.** Average L_1 distances between each library concept and the nearest learned concept.

	dSprites Bin	dSprites Adv
Slot Att.	0.0312	0.0497
DTI-Sprites	0.0133	0.0219
Ours	**0.0033**	**0.0051**

Table 3: **Cross-dataset reconstruction.** MSE reconstruction loss on EMNIST letters for methods trained on MNIST digits.

	MNIST(Train)	EMNIST(Test)
Slot Att.	0.0048	0.0560
DTI-Sprites	0.0065	0.0202
Ours(128)	0.0114	0.0169
Ours(512)	0.0090	0.0140

versarial version. In Tetrominoes, the performance of all methods is near optimal. In the Clevr6 dataset, the lighting, reflections, and perspective projection effects violate our assumptions about the image formation model, which we assume to be alpha-blending. Nevertheless, we include Clevr6 to show that our method gracefully fails if our assumptions about the image formation do not hold. Table S2 shows that the visual concepts learned by our method are closer to the ground truth concepts than for existing methods, that is, our method finds the dictionary of objects that scenes are composed of more accurately.

Figures 4 and S7 show example decompositions on each dataset. We provide the full dictionaries of visual concepts extracted from each dataset in the supplemental. DTI-Sprites is most related to our method. Table 1 shows our competitive segmentation performance, but Table S2 and Figure 7 reveal that the concepts it learns each may entangle multiple ground truth visual concepts, especially when using lower concept numbers. When reconstructing a scene, their image formation model needs to disentangle these concepts. Thus, it does often not correctly identify the concepts in a dataset.

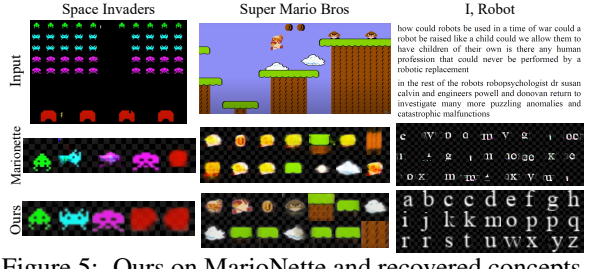

Figure 5: Ours on MarioNette and recovered concepts.

In Fig 5 we should comparision between our method and Marionette, our method requires less data to extract concepts, only 4 and 6 frames for Space Invaders and Super Mario Bros, respectively, compared to several thousands of frames required by MarioNette.

6.2 Cross-dataset Reconstruction

To quantify generalization performance, we train our algorithm and the baselines on the MNIST [26] dataset, which contains hand-written digits, and test their reconstruction performance on EMNIST [7] dataset, which also contains hand-written letters. Table 3 shows a quantitative comparison between Slot Attention, DTI-Sprites and two versions of our method

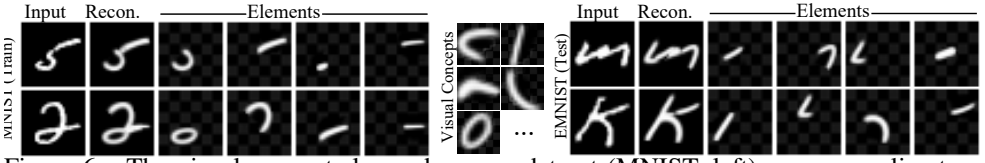

Figure 6: The visual concepts learned on one dataset (MNIST, left), can generalize to a second dataset (EMNIST, right), without further training.

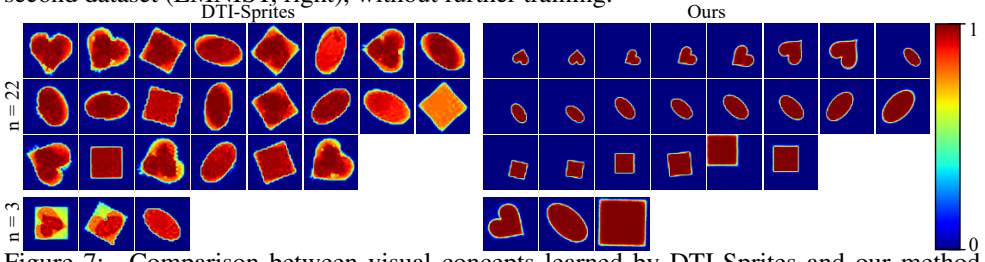

Figure 7: Comparison between visual concepts learned by DTI-Sprites and our method on the M-dSprites Bin. dataset with different dictionary sizes n. The heatmap shows pixel intensities. DTI-sprites learns entangled concepts that correspond to multiple ground truths.

using the MSE reconstruction loss, with the visual dictionary size capped to 128 or 512. Figure 6 shows an example decomposition. Since we do not rely on learned priors in addition to our dictionary, our inference pipeline shows significantly better generalization performance than the baselines. Note that in this experiment, DTI-Sprites is allowed translation, rotation and scaling of elements, while our method only uses translations.

Limitations. Our pipeline has two main limitations. First, the computational cost of the element parameter search. We plan to optimize the search operation using techniques like a coarse-to-fine search in future work. Second, we currently search for exact repetitions of objects to learn our concepts. Accounting for deformations/variations by incorporating a more general parametric deformation model, for example by using a neural network with the help of differentiable rasterizers, will be a valuable next step towards a more general model.

7 Conclusion and Future Work

We presented a general method to learn visual concepts from data, both for images and shapes, without explicit supervision or learned priors. Our main idea is posing the search for visual concepts as a direct optimization, which can be solved efficiently when splitting the task into alternating dictionary finding and parameter optimization steps. Using direct optimization, instead of a network-based approach, improves the quality of the resulting visual concepts and additionally reveals parameters such as hue, position, and scale that are not available to most network-based approaches. In the future, we would like to extend our approach to a fully generative model. One approach would be to learn a distribution over the element parameters. When combined with the learned concepts, we could sample the element parameter distributions to produce new images with the given image formation function. This opens up new avenues for parametric generative models, blurring the line between neuro-symbolic and image-based generative models. We believe that ultimately the right direction for a decomposition is a hybrid between network-based and search-based methods.

References

[1] Vijay Badrinarayanan, Ankur Handa, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *arXiv preprint arXiv:1505.07293*, 2015.

[2] Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[5] Vincent S Chen, Paroma Varma, Ranjay Krishna, Michael Bernstein, Christopher Re, and Li Fei-Fei. Scene graph prediction with limited labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2580–2590, 2019.

[6] Yuanhao Chen, Long Zhu, Chenxi Lin, Hongjiang Zhang, and Alan L Yuille. Rapid inference on a novel and/or graph for object detection, segmentation and parsing. *Advances in neural information processing systems*, 20:289–296, 2007.

[7] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

[8] Sébastien Ehrhardt, Oliver Groth, Aron Monszpart, Martin Engelcke, Ingmar Posner, Niloy J. Mitra, and Andrea Vedaldi. RELATE: Physically plausible multi-object scene synthesis using structured latent spaces. *NeurIPS*, 2020.

[9] Martin Engelcke, Adam R Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. *arXiv preprint arXiv:1907.13052*, 2019.

[10] Martin Engelcke, Oiwi Parker Jones, and Ingmar Posner. Genesis-v2: Inferring unordered object representations without iterative refinement. *arXiv preprint arXiv:2104.09958*, 2021.

[11] RJCS Girshick. Fast r-cnn. arxiv 2015. *arXiv preprint arXiv:1504.08083*, 2015.

[12] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[13] Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. *arXiv preprint arXiv:1909.10893*, 2019.

[14] Klaus Greff, Sjoerd Van Steenkiste, and Jürgen Schmidhuber. Neural expectation maximization. *arXiv preprint arXiv:1708.03498*, 2017.

[15] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pages 2424–2433. PMLR, 2019.

[16] Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[18] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with em routing. In *International conference on learning representations*, 2018.

[19] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.

[20] Nebojsa Jojic, Brendan J Frey, and Anitha Kannan. Epitomic analysis of appearance and shape. In *Computer Vision, IEEE International Conference on*, volume 2, pages 34–34. IEEE Computer Society, 2003.

[21] Artan Kaso. Computation of the normalized cross-correlation by fast fourier transform. *PLOS ONE*, 13(9):1–16, 09 2018. doi: 10.1371/journal.pone.0203434. URL https://doi.org/10.1371/journal.pone.0203434.

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

[23] Adam R Kosiorek, Sara Sabour, Yee Whye Teh, and Geoffrey E Hinton. Stacked capsule autoencoders. *arXiv preprint arXiv:1906.06818*, 2019.

[24] Adam R Kosiorek, Hyunjik Kim, and Danilo J Rezende. Conditional set generation with transformers. *arXiv preprint arXiv:2006.16841*, 2020.

[25] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *arXiv preprint arXiv:1602.07332*, 2016.

[26] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[27] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pages 3744–3753. PMLR, 2019.

[28] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *arXiv preprint arXiv:2006.15055*, 2020.

[29] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *European conference on computer vision*, pages 852–869. Springer, 2016.

[30] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

[31] Tom Monnier, Elliot Vincent, Jean Ponce, and Mathieu Aubry. Unsupervised layered image decomposition into object prototypes. *arXiv preprint arXiv:2104.14575*, 2021.

[32] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7588–7597, 2019.

[33] Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *arXiv preprint arXiv:2002.08988*, 2020.

[34] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[35] Pradyumna Reddy, Paul Guerrero, Matt Fisher, Wilmot Li, and Niloy J Mitra. Discovering pattern structure using differentiable compositing. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020.

[36] J Redmon, S Divvala, R Girshick, and A Farhadi. You only look once: Unified, real-time object detection. arxiv 2015. *arXiv preprint arXiv:1506.02640*, 2015.

[37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

[38] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. *arXiv preprint arXiv:1710.09829*, 2017.

[39] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020.

[40] Dmitriy Smirnov, Michael Gharbi, Matthew Fisher, Vitor Guizilini, Alexei A. Efros, and Justin Solomon. MarioNette: Self-supervised sprite learning. *Conference on Neural Information Processing Systems*, 2021.

[41] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332, 2012.

[42] Karl Stelzner, Kristian Kersting, and Adam R Kosiorek. Decomposing 3d scenes into objects via unsupervised volume segmentation. *arXiv preprint arXiv:2104.01148*, 2021.

[43] Sjoerd van Steenkiste, Karol Kurach, Jürgen Schmidhuber, and Sylvain Gelly. Investigating object compositionality in generative adversarial networks. *Neural Networks*, 130:309–325, 2020.

[44] Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, 11:2837–2854, 2010.

[45] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[46] Yan Zhang, Jonathon Hare, and Adam Prugel-Bennett. Deep set prediction networks. *Advances in Neural Information Processing Systems*, 32:3212–3222, 2019.

[47] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[48] Long Zhu, Yuanhao Chen, Antonio Torralba, William Freeman, and Alan Yuille. Part and appearance sharing: Recursive compositional models for multi-view. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1919–1926. IEEE, 2010.

[49] Long Leo Zhu, Chenxi Lin, Haoda Huang, Yuanhao Chen, and Alan Yuille. Unsupervised structure learning: Hierarchical recursive composition, suspicious coincidence and competitive exclusion. In *European Conference on Computer Vision*, pages 759–773. Springer, 2008.

[50] Song-Chun Zhu and David Mumford. *A stochastic grammar of images*. Now Publishers Inc, 2007.