Inductive Biases for Object-Centric Representations in the Presence of Complex Textures

Samuele Papa[1], Ole Winther[2,3], and Andrea Dittadi[2,4]

[1] POP-AART Lab, University of Amsterdam & The Netherlands Cancer Institute
s.papa@uva.nl
[2] Technical University of Denmark
[3] University of Copenhagen & Copenhagen University Hospital
[4] Max Planck Institute for Intelligent Systems, Tübingen, Germany

Abstract. Understanding which inductive biases could be helpful for the unsupervised learning of object-centric representations of natural scenes is challenging. In this paper, we systematically investigate the performance of two models on datasets where neural style transfer was used to obtain objects with complex textures while still retaining ground-truth annotations. We find that by using a single module to reconstruct both the shape and visual appearance of each object, the model learns more useful representations and achieves better object separation. In addition, we observe that adjusting the latent space size is insufficient to improve segmentation performance. Finally, the downstream usefulness of the representations is significantly more strongly correlated with segmentation quality than with reconstruction accuracy.

Keywords: representation learning, object-centric learning, structured representations, realistic dataset, deep learning

1 Introduction

A core motivation for object-centric learning is that humans interpret complex environments such as natural scenes as the composition of distinct interacting objects. Evidence for this claim can be found in cognitive psychology and neuroscience [33, 34, 37, 6]. Current object-centric learning approaches try to merge the advantages of connectionist and symbolic methods by representing each object with a distinct vector [17]. The problem of object separation becomes central for unsupervised methods that can only use the data itself to lean how to isolate objects. Several methods have been proposed to provide inductive biases to achieve this objective [5, 12, 27, 11, 24]. However, they are typically tested on simple datasets where objects show little variability in their texture, often being monochromatic. This characteristic may allow models to successfully separate objects by relying on low-level characteristics, such as color [16], over more desirable high-level ones, such as shape.

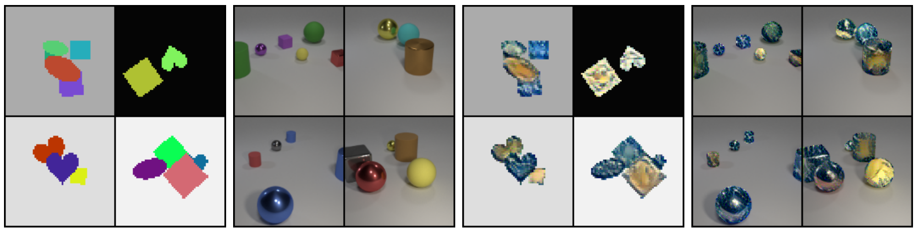


Fig. 1: Left: samples from the original datasets. Right: samples from the same datasets with neural style transfer.

Research in the direction of natural objects is still scarce [22, 13, 24], as such datasets often do not provide exhaustive knowledge of the factors of variation, which are very rich in natural scenes. In this context, unsupervised methods struggle to learn object-centric representations, and the reason for this remains unexplained [16, Section 5].

In this paper, we conduct a systematic experimental study on the inductive biases necessary to learn object-centric representations when objects have complex textures. To obtain significant and interpretable results, we focus on static images and use *neural style transfer* [15] to apply complex textures to the objects of the Multi-dSprites [21] and CLEVR [20] datasets, as shown in Fig. 1. This makes the task significantly more challenging while still preserving all the advantages of a procedurally generated dataset, and avoiding the above-mentioned pitfalls of natural datasets.

We investigate MONet [5] and Slot Attention [27], two popular slot-based autoencoder models that learn to represent objects separately and in a common format. The latter obtains object representations by applying Slot Attention to a convolutional embedding of the input, and then decodes each representation into shape and visual appearance via a *single* component. In contrast, MONet reconstructs them with *two* components: a recurrent attention network that segments the input, and a variational autoencoder (VAE) [23, 31] that separately learns a representation for each object by learning to reconstruct its shape and visual appearance. Unlike in Slot Attention, the shape reconstructed by the VAE is *not* used to reconstruct the final image; instead the shape from the recurrent attention network is used. To still have shape information in the latent representation of the VAE, the training loss includes a KL divergence between the mask predicted by the VAE and the one predicted by the attention network. Therefore shape and visual appearance are, in effect, *separate* unless the KL divergence provides a strong enough signal.

For this study, we posit two desiderata for object-centric models, adapted from [8]:

Desideratum 1. *Object separation and reconstruction.* The models should have the ability to accurately separate and reconstruct the objects in the input, even those with complex textures. For the models considered here, this means that they should correctly segment the objects and reproduce their properties in the reconstruction, including their texture.

Desideratum 2. *Object representation.* The models should capture and represent the fundamental properties of each object present in the input. When ground-truth properties are available for the objects, this can be evaluated via a downstream prediction task.

We summarize our **main findings** on the two models analyzed as follows:

1. Models that better balance the importance of both shape and visual appearance of the objects seem to be less prone to what we call *hyper-segmentation* (see Section 3). We show how this can be achieved with an architecture that uses a single module to obtain both shape and visual appearance of each object. When this is not the case, it becomes significantly more challenging for a model to correctly separate objects and learn useful representations.
2. Hyper-segmentation of the input leads to the inability of the model to obtain useful representations. Separation is a strong indicator of representation quality.
3. The *representation bottleneck* is not sufficient to regulate a model's ability to segment the input. Tuning other hyperparameters such as encoder and decoder capacity appears to be necessary.

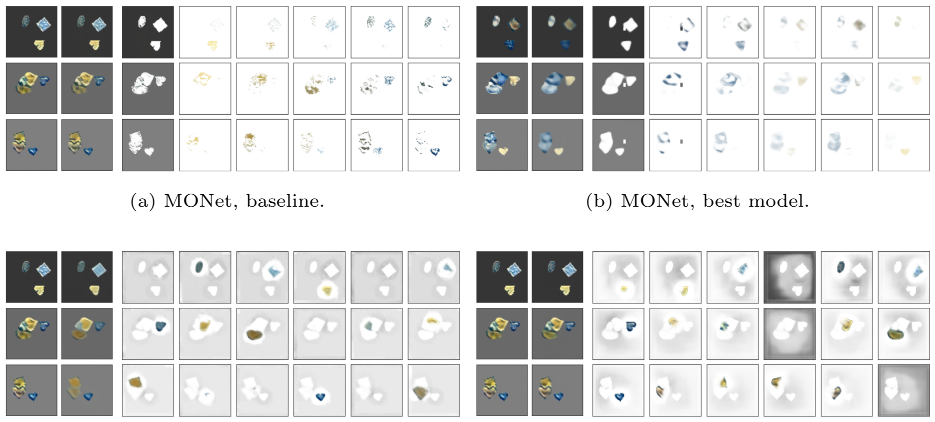
2 Methods

Datasets and models. Similarly to [8], we use neural style transfer [15] to increase the complexity of the texture of the objects in the Multi-dSprites and CLEVR datasets (see Appendix B for details). This allows for textures that have high variability but are still correlated with the shape of the object, as opposed to preset patterns as done in [16] and [22] or completely random ones. We apply neural style transfer to the entire image and then select the objects using the ground-truth segmentation masks (see Fig. 1). Keeping the background simple allows for a more straightforward interpretation of the models' performance. As object-centric models, we consider MONet [5] and Slot Attention [27], chosen because they implement very distinct mechanisms to solve the the problem of separation (see Section 1).

Evaluation. Following the two desiderata in Section 1, we separately focus on the *separation*, *reconstruction*, and *representation* performance of the models:

- *Separation* is measured by the Adjusted Rand Index (ARI) [18], which quantifies the similarity between two partitions of a set.
- *Reconstruction* is measured using the Mean Squared Error (MSE) between input and reconstructed images.
- *Representation* is measured by the performance of a simple downstream model trained to predict the properties of each object using only the object representations as inputs. Following previous literature [27, 8], we match ground-truth objects with object representations such that the overall loss is minimized.

Performance studies. The *baseline* performance of the models on the style transfer datasets is established using the hyperparameters from the literature. We then vary parameters and architectures to improve performance. In MONet, we reduce the number of skip-connections of the U-Net in the attention module, we change the latent space size, the number of channels in the encoder and decoder of the VAE, and the β and σ parameters in the training objective. In Slot Attention, we increase the number of layers and channels in both encoder and decoder and increase the size of the latent space. For both models, we investigate how the latent space size alone affects performance. We use multiple random seeds to account for variability in performance when feasible (see Appendix E for further details).



(a) MONet, baseline. (b) MONet, best model.

(c) Slot Attention, baseline. (d) Slot Attention, best model.

Fig. 2: Reconstruction and separation performance on Multi-dSprites (from the validation set). From left to right in each subfigure: original input, final reconstruction, and product of the reconstruction and mask for each of the six slots. The improved architecture for Slot Attention splits objects less often. MONet still fails to separate objects correctly although it blurs the reconstructions. See Fig. 13 (Appendix G) for similar results on CLEVR.

3 Experiments

Architectural biases. Qualitatively, the MONet baseline (Fig. 2a) segments primarily according to color, resulting in each slot encoding fragments of multiple objects that share the same color. We call this behavior *hyper-segmentation*. On the other hand, the Slot Attention baseline (Fig. 2c) produces blurred reconstructions and avoids hyper-segmentation. Here, some objects are still split across more than one slot but, unlike in MONet, we do not observe multiple objects that are far apart in the scene being (partially) modeled by the same slot. We observe this quantitatively in Fig. 4 (top): compared to the Slot Attention baseline, the MONet baseline has a significantly worse ARI score but a considerably better MSE. Fig. 13 in Appendix G shows similar results on CLEVR.

These observations can guide our search for better model parameters. Slot Attention is blurring away the small details of the texture and focuses on the shape to separate them. MONet, instead, achieves good reconstructions but does so by using the attention module to select pixels that share the same color, as opposed to entire objects, while the VAE simply reconstructs plain colors (see Appendix G for more details). Therefore, for MONet we attempt to sacrifice some reconstruction quality in exchange for better object separation. For Slot Attention, we investigate whether improving the reconstructions negatively affects object separation. We refer to Appendix E for further details and results on the hyperparameter search for MONet and Slot Attention.

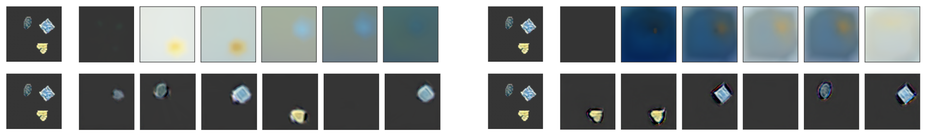


Fig. 3: Qualitative results for the reconstruction of the visual appearance of the objects on Multi-dSprites (from the validation set). In each of the 4 groups, we show the input (leftmost image) and the reconstruction of the visual appearance from each of the six slots, before masking. Top row: MONet; bottom row: Slot Attention; left: baselines; right: models with best hyperparameters. MONet primarily reconstructs plain colors with little to no regard to objects, while Slot Attention consistently separates objects even in the presence of textures. In Fig. 13 (Appendix G), we show similar results on CLEVR. See also Fig. 12 for results on additional images from Multi-dSprites.

When qualitatively looking at the results obtained by the hyperparameter search performed for MONet, we notice a consistent inability of the "component VAE" to capture different characteristics other than simple colors (see Fig. 3 and Fig. 13), even when strongly penalizing the VAE for reconstructing shapes that are inconsistent with the masks computed by the recurrent attention network (which are directly used for reconstruction). We now discuss the results obtained using the combination of hyperparameters that achieves the best performance, called *best model*. In Fig. 2b, we see that MONet still hyper-segments even though the reconstructions are now blurred. For Slot Attention (Fig. 2d), we observe that the quality of reconstructions has improved, and it more often represents an entire object in a single slot. Although the ARI for MONet has also improved, the separation problem is still far from solved, while Slot Attention shows a significant improvement both in terms of ARI and MSE (see two uppermost plots in Fig. 4). Note how, for Slot Attention, the ARI is significantly lower in Multi-dSprites, when compared to CLEVR. The likely reason is that, when a significant portion of an object is occluded by another, the visible shape is being altered significantly and the edges of objects are not clear. Therefore, two explanations of the same scene can still be reasonable while not corresponding to the ground truth. This extreme overlap never occurs in CLEVR.

Overall, even when MONet sacrifices reconstruction quality and blurs away the details, hyper-segmentation is still present as evidenced by our qualitative and quantitative analyses. This suggests that the separation problem in MONet may not simply be caused by the training objective, but rather by its architectural biases. Indeed, improving the reconstruction performance of Slot Attention has, instead, yielded both better separation and more detailed reconstructions, suggesting that generating shape and appearance using a single module is a more favorable inductive bias for learning representations of objects with complex textures.

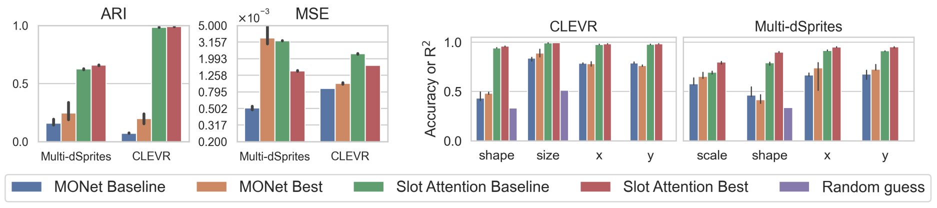
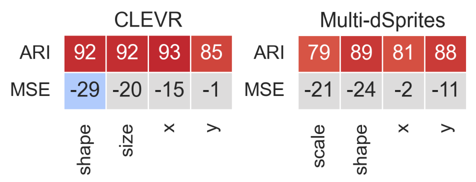


Fig. 4: Median performance of the different seeds trained for each of the indicated models (error bars denote 95% confidence intervals). Top: ARI (↑) and MSE (↓) for each dataset and model. Bottom: performance of the downstream model on each feature of the objects. Accuracy is used for categorical features and R^2 for numerical features. A random guess baseline is shown in purple.



CLEVR	shape	size	x	y
ARI	92	92	93	85
MSE	-29	-20	-15	-1

Multi-dSprites	scale	shape	x	y
ARI	79	89	81	88
MSE	-21	-24	-2	-11

Fig. 5: Rank correlation of the ARI and MSE scores with downstream property prediction performance. Correlations are computed over *all* the models trained with that dataset in our study.

Representation performance. To understand the interplay between separation and learned representations, we explore the performance on a downstream property prediction task that was trained using the object representations as inputs (see Appendix C.1 for details). From Fig. 4, we observe how MONet fails to capture some of the properties in the representations and consistently performs worse than Slot Attention, for both the baseline and the improved versions. This suggests that, as highlighted in [8], a model that is not capable of correctly separating objects will also fail to accurately represent them. The trend is also clear from Fig. 5, which shows that a higher ARI score strongly correlates with an increased performance of the downstream model on all object properties. The correlation with MSE is much weaker, which highlights how *strong visual reconstruction performance is not the ultimate indicator for good object representations*. This result does not contradict previous findings [8] as here the properties we expect the downstream model to predict have little to do with the texture of the object and, therefore, the model can have poorer reconstructions while still obtaining useful representations.

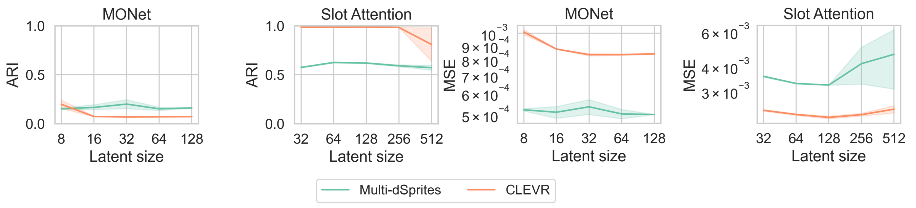


Fig. 6: The ARI (↑) and MSE (↓) show no significant change across latent space sizes, supporting that the *representation bottleneck* (see main text) is not a sufficient inductive bias for object separation. Training instability can be seen in Slot Attention with latent size above 128. Two seeds for each latent size are used. Shown are mean (line) and 95% confidence interval (shaded area).

Representation bottleneck. Here, we investigate how the size of the *representation bottleneck* (see Appendix A) affects performance. In Fig. 6, we observe that the MSE improves slightly and quickly plateaus for both MONet and Slot Attention when the latent space size increases. However, the ARI does not significantly change even with increased latent size. The increase in latent space size arguably increases the model's capacity, but it does not prove to be enough to improve the separation and reconstruction performance significantly.

4 Related work

Learning representations that reflect the underlying structure of data is believed to be useful for downstream learning and systematic generalization [4, 25, 17]. While many recent empirical studies have investigated the usefulness of disentangled representations and the inductive biases involved in learning them [26, 36, 35, 29, 9, 10], analogous experimental studies in the context of object-centric representations are scarce. The study by [11] presents an investigation into inductive biases for object separation, focusing on one model and traditional synthetic datasets. In this work, we study hyper-segmentation on datasets where objects have complex textures. [22] recently proposed ClevrTex, a dataset that introduces challenging textures to scenes from CLEVR [20]. The experiments reported show that, without any tuning, some models fail to segment complex scenes by focusing on colors. The authors, similarly to what we highlighted in this work, state that "ignoring confounding aspects of the scene rather than representing them might aid in the overall task [of segmentation]." In our work, we investigate the mechanism behind the ability of some models to ignore superfluous details and more successfully segment the image, proposing a useful inductive bias to achieve better object representations.

Recently, works that propose new object-centric learning methods also include evaluations on more complex datasets. [16] train IODINE on Textured MNIST and ImageNet, and observe that the model separates the image primarily according to color when the input is complex. GENESIS-V2 [13] was trained on Sketchy and APC, two real-world robot manipulation datasets. However, the authors do not explore the mechanism behind the performance of the models they tested, and do not attempt to optimize the architectures. In the video domain, [24] include evaluations on a dataset with complex textures, training their model to predict optical flow rather than a reconstruction of the input.

5 Conclusions

In this paper, we have investigated which inductive biases may be most useful for slot-based unsupervised models to obtain good object-centric representations of scenes where objects have complex textures. We found that using a single module to reconstruct both shape and visual appearance of objects naturally balances the importance of these two aspects in the generation process, thereby avoiding hyper-segmentation and achieving a better compromise between precise texture reconstructions and correct object segmentation. Therefore, our recommendation is that models should have separation as an integral part of the representation process. Additionally, we showed that separation strongly correlates with the quality of the representations, while reconstruction accuracy does not: this justifies sacrificing some reconstruction quality. Finally, we observed that the representation bottleneck is not a sufficient inductive bias, as increasing the latent space size can be counterproductive unless the model is already separating the input correctly.

Although the models considered in our study have been shown strong performance on this type of data, it would be interesting to explore if the same conclusions hold for other models that approach the problem in a similar way, such as GENESIS, IODINE, and GENESIS-V2, as well as methods based on spatial slots, such as SPAIR or SPACE. Another interesting avenue for future work is to extend our study to more complex downstream tasks involving abstract reasoning, e.g., in a neuro-symbolic system, where symbol manipulation can be performed either within a connectionist framework [14, 32, 3] or by purely symbolic methods [1, 28, 7]. Finally, it would be relevant to validate our conclusions on additional datasets, and to introduce objects with varying texture complexity, as this could require different model capacities to achieve separation [11].

References

1. Asai, M., Fukunaga, A.: Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In: Proceedings of the aaai conference on artificial intelligence. vol. 32 (2018)
2. Bachlechner, T., Majumder, B.P., Mao, H., Cottrell, G., McAuley, J.: ReZero is all you need: Fast convergence at large depth. In: Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence. pp. 1352–1361. PMLR (Dec 2021)
3. Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., Pascanu, R.: Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261 (2018)
4. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(8), 1798–1828 (2013)
5. Burgess, C.P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., Lerchner, A.: Monet: Unsupervised scene decomposition and representation. arXiv preprint arXiv:1901.11390 (2019)
6. Dehaene, S.: How We Learn: Why Brains Learn Better than Any Machine ... for Now. Viking, New York (2020)
7. Dittadi, A., Drachmann, F.K., Bolander, T.: Planning from pixels in atari with learned symbolic representations. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 4941–4949 (2021)
8. Dittadi, A., Papa, S., De Vita, M., Schölkopf, B., Winther, O., Locatello, F.: Generalization and robustness implications in object-centric learning. arXiv preprint arXiv:2107.00637 (2021)
9. Dittadi, A., Träuble, F., Locatello, F., Wüthrich, M., Agrawal, V., Winther, O., Bauer, S., Schölkopf, B.: On the transfer of disentangled representations in realistic settings. In: International Conference on Learning Representations (2021)
10. Dittadi, A., Träuble, F., Wuthrich, M., Widmaier, F., Gehler, P.V., Winther, O., Locatello, F., Bachem, O., Schölkopf, B., Bauer, S.: The role of pretrained representations for the OOD generalization of RL agents. In: International Conference on Learning Representations (2022)
11. Engelcke, M., Jones, O.P., Posner, I.: Reconstruction bottlenecks in object-centric generative models. arXiv preprint arXiv:2007.06245 (2020)
12. Engelcke, M., Kosiorek, A.R., Jones, O.P., Posner, I.: GENESIS: Generative Scene Inference and Sampling with Object-Centric Latent Representations. In: ICLR 2020 (2020)
13. Engelcke, M., Parker Jones, O., Posner, I.: GENESIS-V2: Inferring unordered object representations without iterative refinement. arXiv preprint arXiv:2104.09958 (2021)
14. Evans, R., Grefenstette, E.: Learning explanatory rules from noisy data. Journal of Artificial Intelligence Research **61**, 1–64 (2018)
15. Gatys, L., Ecker, A., Bethge, M.: A Neural Algorithm of Artistic Style. Journal of Vision **16**(12), 326–326 (Sep 2016). https://doi.org/10.1167/16.12.326
16. Greff, K., Kaufman, R.L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., Lerchner, A.: Multi-object representation learning with iterative

variational inference. In: International Conference on Machine Learning. pp. 2424–2433. PMLR (2019)
17. Greff, K., van Steenkiste, S., Schmidhuber, J.: On the binding problem in artificial neural networks. arXiv preprint arXiv:2012.05208 (2020)
18. Hubert, L., Arabi, P.: Comparing partitions. Journal of Classification **2**(1), 193–218 (Dec 1985). https://doi.org/10.1007/BF01908075
19. Jacq, A.: Neural Transfer Using PyTorch — PyTorch Tutorials 1.10.1+cu102 documentation. https://pytorch.org/tutorials/advanced/neural_style_tutorial.html (2021)
20. Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., Girshick, R.: Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2901–2910 (2017)
21. Kabra, R., Burgess, C., Matthey, L., Kaufman, R.L., Greff, K., Reynolds, M., Lerchner, A.: Multi-object datasets (2019)
22. Karazija, L., Laina, I., Rupprecht, C.: Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. In: Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2) (2021)
23. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. In: International Conference on Learning Representations (2014)
24. Kipf, T., Elsayed, G.F., Mahendran, A., Stone, A., Sabour, S., Heigold, G., Jonschkowski, R., Dosovitskiy, A., Greff, K.: Conditional object-centric learning from video. arXiv preprint arXiv:2111.12594 (2021)
25. Lake, B.M., Ullman, T.D., Tenenbaum, J.B., Gershman, S.J.: Building machines that learn and think like people. Behavioral and Brain Sciences **40** (2017)
26. Locatello, F., Bauer, S., Lucic, M., Gelly, S., Schölkopf, B., Bachem, O.: Challenging common assumptions in the unsupervised learning of disentangled representations. In: International Conference on Machine Learning (2019)
27. Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., Kipf, T.: Object-centric learning with slot attention. arXiv preprint arXiv:2006.15055 (2020)
28. Mao, J., Gan, C., Kohli, P., Tenenbaum, J.B., Wu, J.: The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In: International Conference on Learning Representations (2019)
29. Montero, M.L., Ludwig, C.J., Costa, R.P., Malhotra, G., Bowers, J.: The role of disentanglement in generalisation. In: International Conference on Learning Representations (2021)
30. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019)
31. Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. arXiv preprint arXiv:1401.4082 (2014)
32. Smolensky, P.: Tensor product variable binding and the representation of symbolic structures in connectionist systems. Artificial intelligence **46**(1-2), 159–216 (1990)
33. Spelke, E.S.: Principles of object perception. Cognitive Science **14**, 29–56 (1990). https://doi.org/10.1016/0364-0213(90)90025-R

34. Téglás, E., Vul, E., Girotto, V., Gonzalez, M., Tenenbaum, J.B., Bonatti, L.L.: Pure Reasoning in 12-Month-Old Infants as Probabilistic Inference. Science **332**, 1054–1059 (May 2011). https://doi.org/10.1126/science.1196404
35. Träuble, F., Creager, E., Kilbertus, N., Locatello, F., Dittadi, A., Goyal, A., Schölkopf, B., Bauer, S.: On disentangled representations learned from correlated data. In: International Conference on Machine Learning. pp. 10401–10412. PMLR (2021)
36. Van Steenkiste, S., Locatello, F., Schmidhuber, J., Bachem, O.: Are disentangled representations helpful for abstract visual reasoning? arXiv preprint arXiv:1905.12506 (2019)
37. Wagemans, J.: The Oxford handbook of perceptual organization. Oxford Library of Psychology (2015)

A Remarks on notation

The term *representation bottleneck* should not be confused with the *reconstruction bottleneck* introduced by [11]. The representation bottleneck refers to the small size of the latent space, while the reconstruction bottleneck refers to how easy it is for the model to reconstruct the data. In [11], the reconstruction bottleneck is posited to be the reason behind the models' inability to separate objects into different slots.

Often, in the paper, we refer to *object-centric representations* and *slots* as synonyms, although this is only true for slot-based models.

The term *hyper-segmentation* refers to when a model splits the input into different slots with little to no regard to high-level characteristics of the input, such as the shapes of objects, and instead just uses low-level characteristics, primarily color. This often results in slots that consist of small clusters of pixels with similar color, which means that several objects can be partially represented in the same slot and at the same time each object may be partially represented in multiple slots. This phenomenon is distinct from *over-segmentation* [12], where multiple slots may reconstruct a single object but no slot reconstructs (parts of) multiple objects. Examples are shown in the main text of the paper (see Fig. 2b and Fig. 2b), where MONet is hyper-segmenting the input, while Slot Attention is sometimes over-segmenting it.

B Datasets

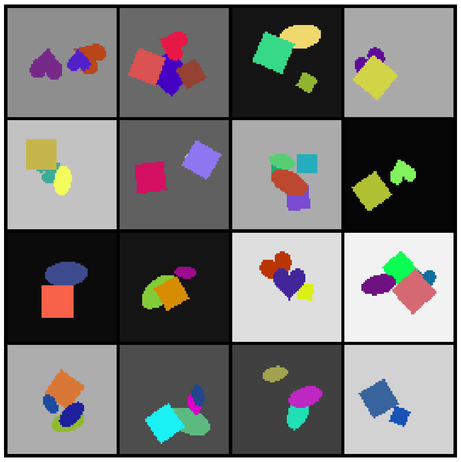
The original versions of both datasets are taken from [21].

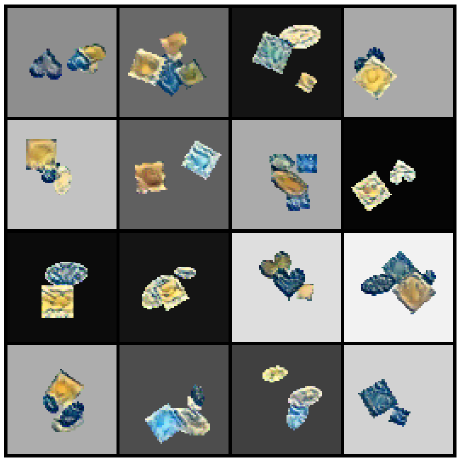
CLEVR. The CLEVR dataset consists of 3D objects placed on a gray background at different distances from the camera. Overlap between objects is kept to a minimum. There are spheres, cylinders, and cubes of eight different colors. The objects can be metallic of opaque. There is a big and small variant of each object and they can be placed in several different orientations. We use the variant of the dataset that has no more than 6 objects in it, as was done in previous object-centric learning research. The total number of samples in the training dataset is 49483, and we leave 2000 samples for validation and 2000 samples for testing.

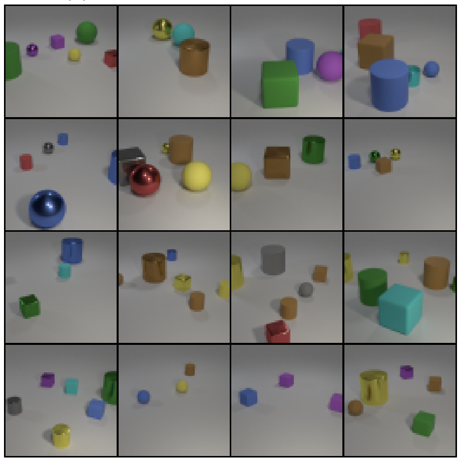
Multi-dSprites. The Multi-dSprites dataset places several 2D objects on a grayscale background. The objects can be a square, an ellipse, or a heart. They can have any RGB color, orientation, and different levels of overlap. Here, we use 90000 samples for the training, 5000 for validation, and 5000 for testing.

Neural Style Transfer. Neural Style Transfer was applied in its most basic form [19] except for a few additions to make running it on several datasets easier. We opted to use *The Starry Night* by Dutch painter *Vincent Van Gogh* as a reference style image (we used the photo from Wikimedia Commons, which is in the public domain). We experimented with several parameters, and we noticed a lot of variability between runs and a more pleasant result from the most basic implementation of the algorithm.

The final version of the datasets was obtained by first applying neural style transfer to each image (optimization happens on an image-to-image basis). This results in the entire scene having the style of the reference image. After obtaining the neural style transfer version of the image, we applied the original segmentation masks of the objects to obtain an image where only the foreground objects have a complex texture, while the background remains the original one.


(a) Samples original Multi-dSprites.


(b) Samples style transfer Multi-dSprites.


(c) Samples original CLEVR.

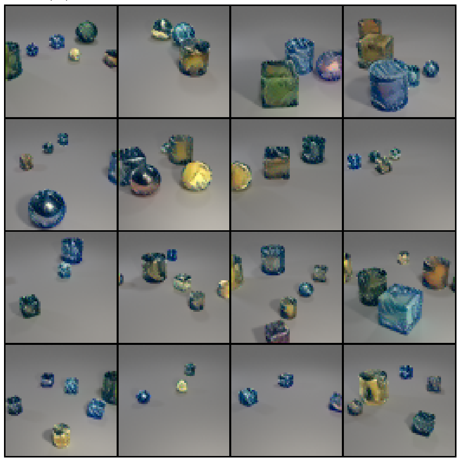

(d) Samples style transfer CLEVR.

Fig. 7: Samples from the original and neural style transfer datasets.

C Evaluation

C.1 Downstream feature prediction task

The setup for the feature prediction task is the same as the one used in [8]. The models used are a simple linear model and an MLP with one hidden layer having 256 neurons and enough outputs to predict all of the features of an object. The input to the model is the object representation of a single object and the output is the predicted features for that object. Let r be the representation of an object, M the model, $\hat{y} = M(r)$ is the output of the model, and y is the target vector such that $y_{i_k:i_{k+1}}$ is the kth feature of the object, a vector of dimension $(i_{k+1} - i_k + 1)$. We use the MSE loss for numerical features and the cross-entropy loss for categorical ones.

Now, it is important to note that, in order to correctly train the model, the representation r needs to be matched with the target vector y of the object that r is representing. However, this is very challenging, as the models can represent any object in any of the slots. Therefore, following [8], we adopted two different strategies to match slots with objects. The first is called *loss matching*: The loss for each pair of slot and object is computed, resulting in a loss matrix L, where $L_{i,j}$ is the loss between the predicted features from the jth slot and the target features from the ith object in the scene. Then, the Hungarian matching algorithm is used to find the pairs of slots-objects that minimize the sum of the loss. The second approach is called *mask matching*: The masks predicted by the models and the ground masks are matched, to find the pairs that have the smallest difference. By using loss matching, the assumption is that the initial errors that are inevitable (because the downstream model has not been trained yet) will eventually disappear. When using mask matching, this problem disappears, however, we rely on the ability of the models to generate masks that closely match the ground truth, which is not the case for models that are hyper-segmenting the input, as is often the case in our study.

C.2 ARI and MSE

We use the standard definitions of Adjusted Rand Index (ARI) and Mean Square Error (MSE).

The ARI, as the name suggests, is the Adjusted version of the Rand Index. The Rand Index is defined as follows. Given a set of n elements S and two partitions A and B of this set, the Rand Index looks at the number of pairs of elements from S and what set they belong to in the two partition. The definition is the following:

$$\mathrm{R} := \frac{m_{11} + m_{00}}{m_{11} + m_{00} + m_{10} + m_{01}},$$

where m_{11} is the number of pairs that are in the same set in both A and B, m_{00} the number of pairs that are in different sets in both partitions, m_{10} the number of pairs that are in the same set in A but different sets in B, and m_{01} counts how many pairs are in different sets in A but the same set in B.

Then, the ARI normalizes and corrects the bias from the Rand Index based on a given null hypothesis [18], such that an ARI of 0 results from completely random partitions of the set, while 1 when the two partitions coincide.

It is important to highlight that the ARI is dependent on the size of the partitions and the number of elements present in each set of the partition, which are both assumed to not change across the entire dataset, which is not always true for the current application.

The MSE is computed between each channel in each pixel of the image, following the traditional formula.

D Implementation of the models

The models were re-implemented in PyTorch [30] and run on NVIDIA A100 or NVIDIA TitanRTX GPUs. The total approximate training time to reproduce this study is 300 GPU days.

E Hyperparameter searches

E.1 Baselines

The baselines were obtained by training the models on the two datasets with 3 different seeds. The parameters are taken from the original papers, but for MONet we use different values for the foreground and background sigma, as suggested by [16]. We stopped the training for all runs in our study, even the ones described later, to 500k steps.

E.2 Improving MONet

Starting from the baseline results, we first explored the hyperparameter space manually, to develop an intuition regarding the effect of each hyperparameter on the performance.

We then performed a hyperparameter search for MONet. We ran a full search, resulting in 36 runs. Because of the high number of runs, we decided to use a single seed. The parameters are listed in Table 1. Those that are not listed were kept unchanged. All combinations of parameters are tested, but `foreground sigma` and `background sigma` have been changed in pairs, so that when the foreground sigma is 0.05, the background sigma is 0.03 and when foreground sigma is 0.5, background sigma is 0.3 to keep consistent weights of the reconstruction loss.

Some analysis on the results of these models can be seen in Appendix E.2, where we can see how the parameters have little to no impact on the overall performance of the model. What proved to be most effective was reducing the number of skip connections in the U-Net and using a small sigma for the loss function. However, these results are not very conclusive, as a small number of skip connections is actually just increasing the ARI slightly by reconstructing bigger patches of objects in the slots and not actually separating them correctly.

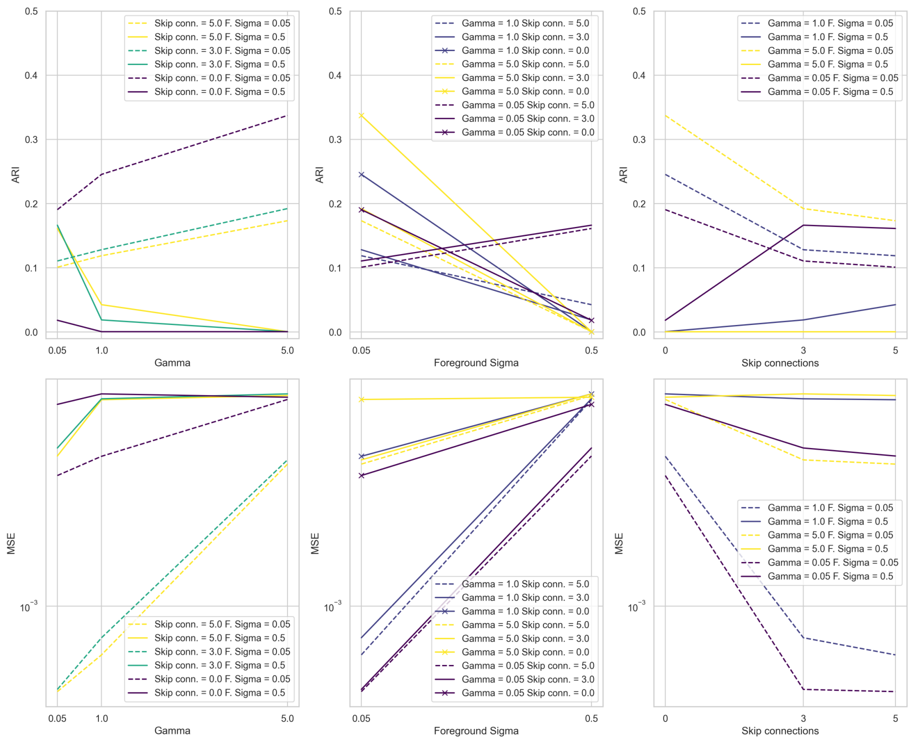


Fig. 8: Results (top row: ARI; bottom row: MSE) from the hyperparameter search for MONet. Although increasing gamma or foreground sigma in the loss function successfully deteriorates reconstruction performance (MSE), they are not sufficient to improve the ARI (in fact, the increase of foreground sigma actually decreases the ARI). A smaller number of skip connections also achieves the desired higher MSE, which corresponds to higher ARI only for small values of sigma. Often, having big values for gamma and sigma results in trends opposing the desired ones in terms of ARI.

Table 1: Hyperparameter search for MONet.

Parameter	**Value(s)**
`foreground sigma`	0.05, 0.5
`background sigma`	0.03, 0.3
`gamma`	0.05, 1, 5
`latent size`	64
`latent space MLP size`	128
`decoder input channels`	66
`number of skip connections in U-Net`	0, 3, 5
`dataset`	CLEVR, Multi-dSprites

E.3 Representation bottleneck study

The representation bottleneck study was done by changing the latent space of both MONet and Slot Attention with 2 seeds and without changing any other parameter, resulting in 32 runs. The latent sizes tested are shown in Table 2. The findings are summarized in the main text of the paper.

Table 2: Latent space sizes tested in the study for each of the two models.

MONet	**Slot Attention**
8	32
16	64
32	128
64	256
128	512

E.4 Improving Slot Attention

We tried to increase the size of the encoder and decoder architecture as much as possible, while being reasonable regarding training time and GPU memory. We tested several architectures, with the objective of improving the overall reconstruction quality by reducing the blurriness. We quickly realised that we needed a very deep architecture, therefore, we opted to use residual layers. The final architecture managed to achieve the best results when averaged over 3 different seeds. Each layer is a stack of two convolutional layers, with Leaky ReLU activation functions, a skip connection and we also employ the re-zero strategy [2].

We increased the latent size to 512, used upscaling in the encoder and downscaling in the decoder. We fixed the broadcast size of the decoder to 32. We used a stack of 16 residual blocks. The architecture of the encoder is described in Table 3, and the decoder is symmetrical (starting from 256 channels going down and instead of downscaling we have upscaling). To map from the input number of channels to the desired ones we use an additional convolutional layer, the same for the output channels. We did not experiment with the number of iterations that the Slot Attention Module performs, but it would be interesting to understand whether this parameter is very influential in natural scenes.

Table 3: Final encoder used for the Slot Attention model that obtained the best results in terms of both ARI and MSE. Residual Blocks always have 2 convolutions and use ReZero [2], two downscaling operations are used for the CLEVR dataset, while one for Multi-dSprites. The decoder is perfectly simmetrical to this structure.

Name	**Number of channels**	**Activation/ Comment**
Residual Block	64	Leaky ReLU
Residual Block	64	Leaky ReLU
Residual Block	64	Leaky ReLU
Residual Block	64	Leaky ReLU
Downscaling		Only for CLEVR
Residual Block	64	Leaky ReLU
Residual Block	64	Leaky ReLU
Residual Block	64	Leaky ReLU
Residual Block	64	Leaky ReLU
Downscaling		
Residual Block	128	Leaky ReLU
Residual Block	128	Leaky ReLU
Residual Block	128	Leaky ReLU
Residual Block	128	Leaky ReLU
Residual Block	256	Leaky ReLU
Residual Block	256	Leaky ReLU
Residual Block	256	Leaky ReLU
Residual Block	256	Leaky ReLU

F Additional results

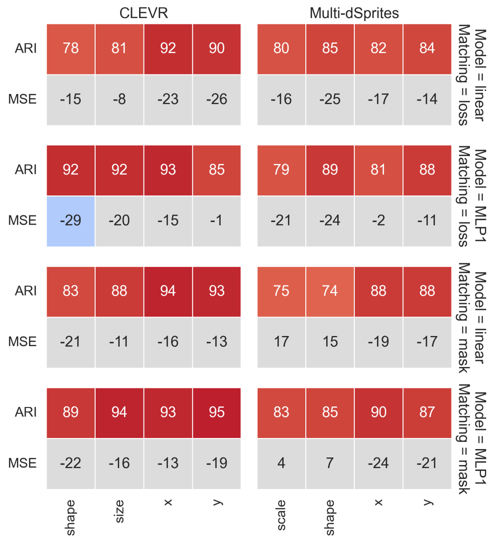


Fig. 9: Pearson's correlation coefficient for all runs, grouped by dataset and showing the different combinations of matching and downstream model. The correlation between downstream model's performance and the ARI (↑) and MSE (↓) metrics shows that ARI is a strongest indicator of good representation quality when the object-centric models are being trained on data with complex texture. Difference in correlation between different matching methods and downstream models is again to be attributed to the poor mask generation quality, which makes mask matching very challenging.

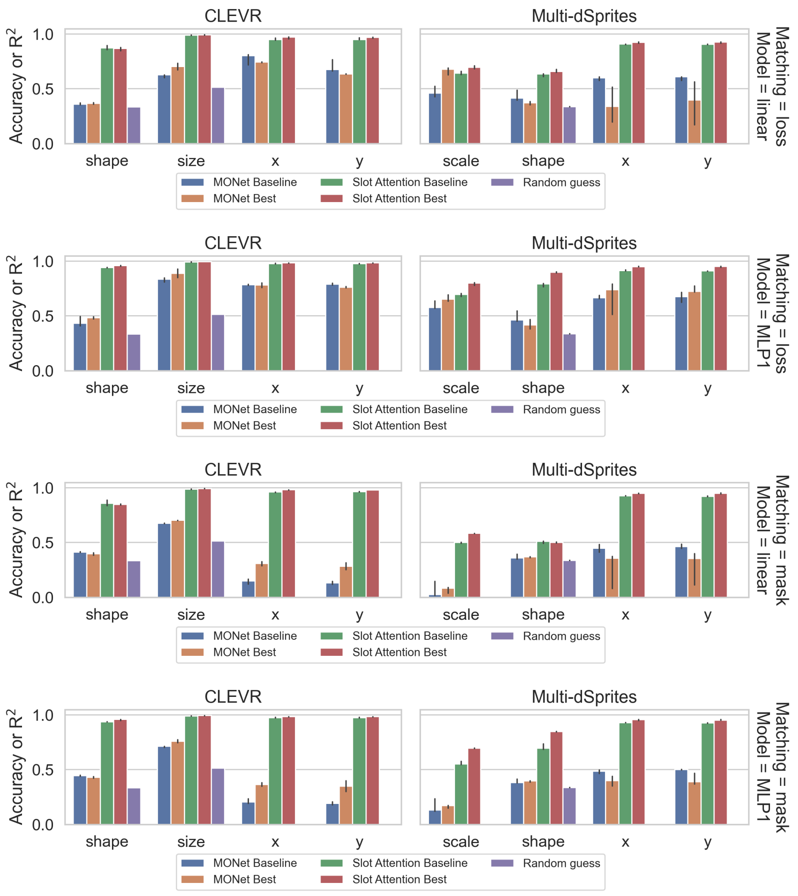


Fig. 10: Comparison of the downstream performance for all combinations of slot-object matching and model type (results on from the test set, downstream models trained on the validation set). We notice how accuracy (↑) and R^2 (↑) both increase significantly for both MONet models when using loss matching compared to mask matching (especially for numerical features). This is expected, as the masks generated by MONet suffer substantially from hypersegmentation, which makes mask-matching a very unstable way to pair slots with the correct object. Instead, Slot Attention manages to generate more accurate masks, which results in more consistent performance between mask and loss matching methods.

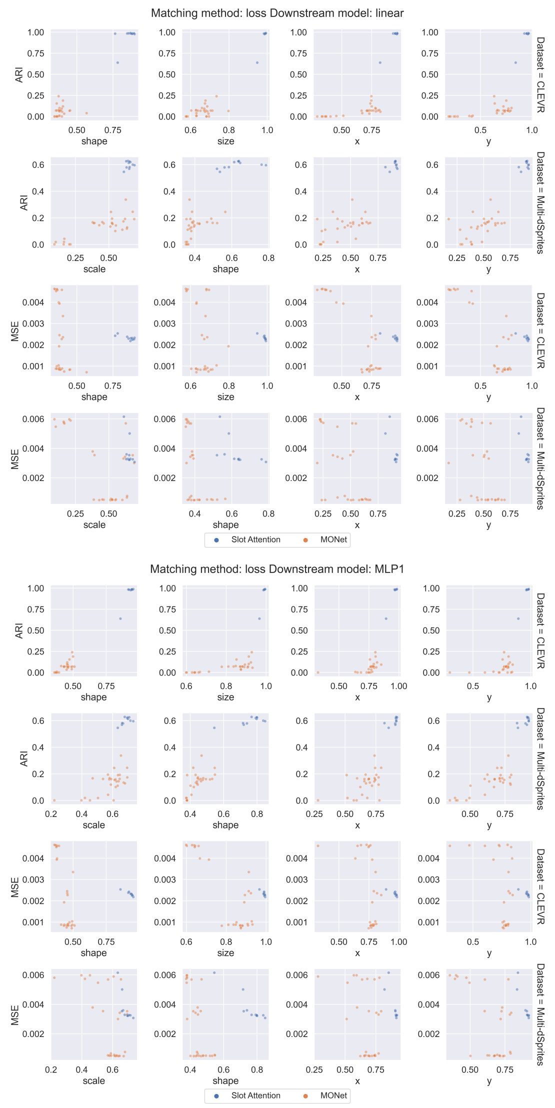
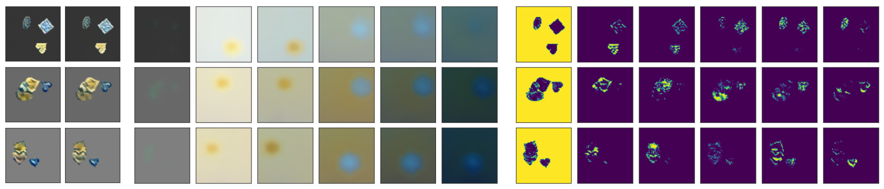
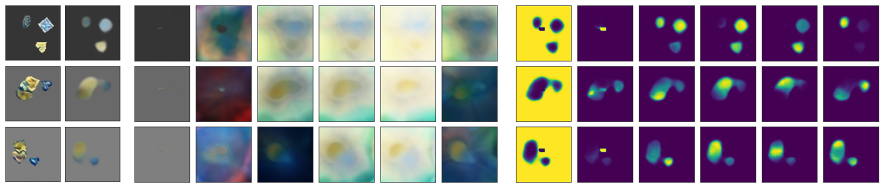
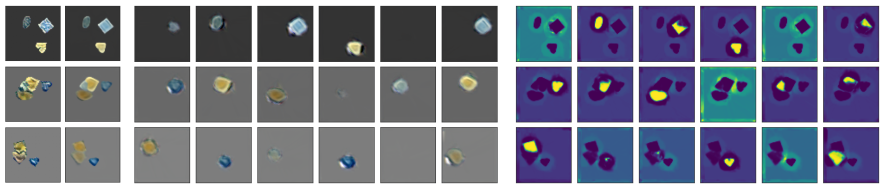


Fig. 11: Scatter plots to inspect correlation between downstream performance, and ARI (↑) and MSE (↓). The color shows the different models, which clearly display distinct patterns. A visual inspection shows that there is very little correlation between downstream performance and MSE. Only loss matching is shown here.

G Qualitative results


(a) Baseline MONet


(b) Best result MONet


(c) Baseline Slot Attention

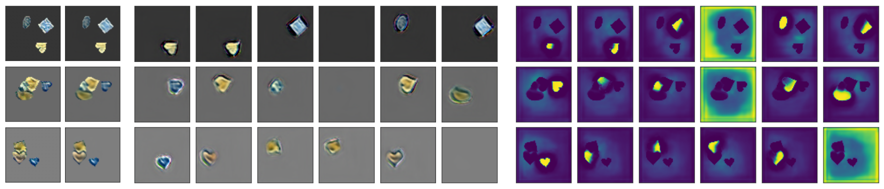
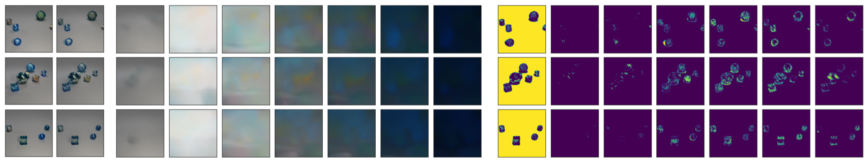
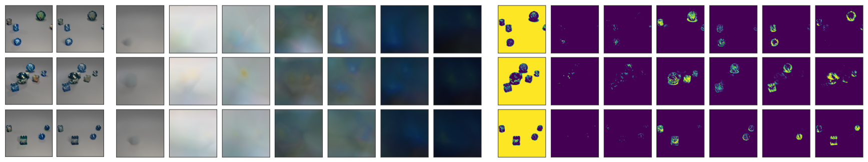
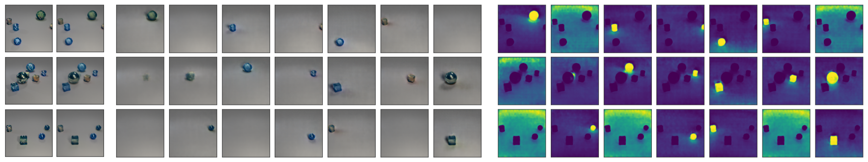

(d) Best result Slot Attention

Fig. 12: Qualitative results for the separation performance of the models in the comparative study on *Multi-dSprites*. From left to right in all subfigures: (top) input, final reconstruction, reconstruction for each of the six slots (no predicted mask is applied here, only the visual appearance part of the reconstruction is shown), (bottom) mask for each of the six slots. Here the difference between the two versions of Slot Attention is even more noticeable, and we can see how MONet is blurring the masks. However, MONet never manages to reconstruct the correct visual appearance, even when a more accurate shape of the objects is being predicted by the attention module. Balancing visual appearance and shape is much more challenging in MONet.


(a) Baseline MONet


(b) Best result MONet


(c) Baseline Slot Attention

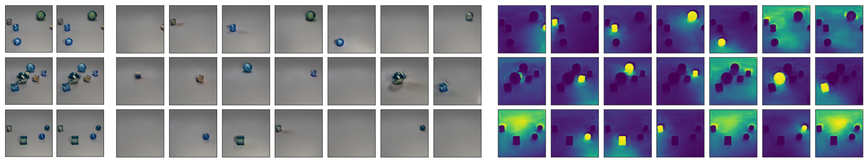

(d) Best result Slot Attention

Fig. 13: Qualitative results for the separation performance of the models in the comparative study on *CLEVR*. From left to right in all subfigures: (top) input, final reconstruction, reconstruction for each of the six slots (no predicted mask is applied here, only the visual appearance part of the reconstruction is shown), (bottom) mask for each of the six slots. The masks on the improved Slot Attention start to include more of the background for each object. In both baseline and best result, Slot Attention isolates each object in a distinct slot, rarely over-segmenting the input, a stark difference when comparing to Multi-dSprites. For MONet, it manages to perform better in CLEVR than Multi-dSprites, however, the best result is still hypersegmenting the input and not blurring it. Overall, MONet cannot reconstruct the visual appearance using the VAE, and leaves all the heavy lifting to the attention module.