CS 679: Project Report

Isaac Joffe, Benjamin Colussi *21138179, 20469173*

1 Background

Multi-Object Network (`MONet`), as described in "MONet: Unsupervised Scene Decomposition and Representation" [1], is an unsupervised generative architecture that decomposes scenes consisting of basic shapes in simple environments into semantically-meaningful components. The unsupervised learning paradigm enables generalization without the need for labelled training data, all while maintaining strong performance.

`MONet` is composed of two neural network (NN) modules: an attention network and a variational autoencoder (VAE) [2]. The attention network auto-regressively generates segmentation masks corresponding to objects in the scene. The VAE then uses these masks to represent and reconstruct each segmented object.

The attention network draws 'attention' to specific image pixels. Given the image and a recurrently-tracked 'scope' representing what image pixels remain to be 'explained' by the VAE, this NN produces a mask representing the next set of image pixels for the VAE to 'explain' (i.e., represent and reconstruct). Distinct masks are recurrently produced for a pre-defined number of attention 'slots', K, that each—ideally—represent an individual element of the image. The initial scope is the entire image, but the mask generated at each iteration is deducted from the next scope; in the final step, the mask is set manually such that all the masks sum to one (i.e., the whole image is completely 'explained'). A U-Net [3] with deep convolutional neural networks (CNNs), skip connections, and a non-skip multi-layer perceptron (MLP) accomplishes this.

The VAE extracts relevant image and mask features using a deep CNN encoder to produce a low-dimensional latent representation (specifically, a 16-dimensional Gaussian distribution). This representation is then sampled, spatially broadcasted [4], and passed through a deep CNN decoder that reconstructs the masked region of the image as well as the mask itself. This VAE bottleneck and decoding scheme facilitates disentanglement while enabling compact object representations.

`MONet` is trained end-to-end using a three-term loss function. The first term is the standard VAE decoder negative log likelihood loss, weighted by mask and summed across slots. This pushes the VAE to accurately reconstruct the components representing each masked region of the image and, thus, disentangle objects in the scene. This is because each masked region—and only the masked region—must be accurately reconstructed through the VAE bottleneck, so the masks should correspond to distinct objects that are best reconstructed separately. The second term is the standard VAE Kullback–Leibler divergence (KLD) loss, summed across slots. This term limits the model to a condensed latent space matching a Gaussian distribution and, thus, ensures that the reconstructed masks and components are uncomplicated. The third term is the KLD between the attention mask distribution and the reconstructed mask distribution. This term drives the VAE to accurately reconstruct the generated masks and, thus, causes the attention network to generate masks that are simple enough to reconstruct through the VAE bottleneck.

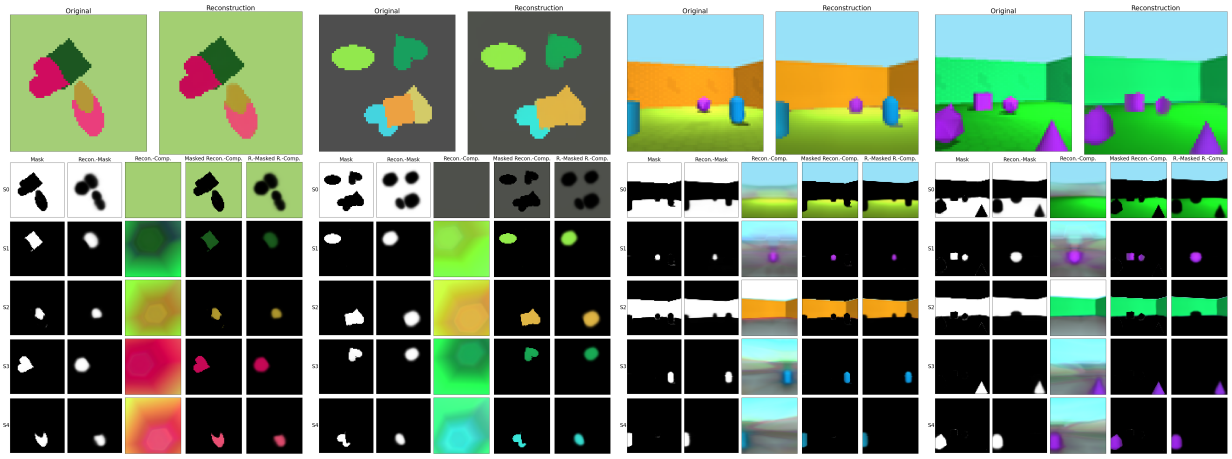
The main claims of the paper (to be verified) are as follows:

1. `MONet` is a state-of-the-art unsupervised model for scene segmentation.
2. `MONet` generates a disentangled representation of each constituent object in a common latent substrate.
3. `MONet` handles two- and three-dimensional scenes with occlusion and variable object counts, and generalizes to novel scene compositions.

2 Reimplementation

The main task of the project was reimplementing `MONet` from scratch. This involved developing the VAE and attention network, as well as combining them to produce `MONet`. The authors did not publish any code alongside the original paper, so the following implementation details were inferred:

- Several relatively inconsequential particulars were assumed. For example, the convolutional layers of the VAE encoder and decoder used a bias term, the convolutional layers of the VAE encoder and attention down- and up-sampling paths used padding, and the VAE MLP used no activation functions.
- The depth of the feature maps throughout the U-Net-based attention network were assumed to be exponentially-growing and -decaying powers of two.



(a) `MultiDSprites`: ID. (b) `MultiDSprites`: OOD. (c) `ObjectsRoom`: ID. (d) `ObjectsRoom`: OOD.

Figure 1: Strong selected example outputs of `MONet`. The image (top left) is recurrently decomposed into $K = 5$ slots (rows) defined by distinct masks generated by the attention network (first column). These masks are then reconstructed by the VAE (second column) alongside the masked image region (third column). These reconstructed components are then masked by the generated (fourth column) or reconstructed (fifth column) masks; the combination of the former represents the complete reconstruction of the entire scene (top right).

- The `LogSigmoid` activation function was used to translate the attention network output logits into the mask and updated scope instead of the `LogSoftmax` activation function; the use of `Softmax` in this way is impossible because when the operation needs to be performed (i.e., at each recurrent step of the model), the dimension to normalize across (all recurrent steps) is necessarily incomplete.

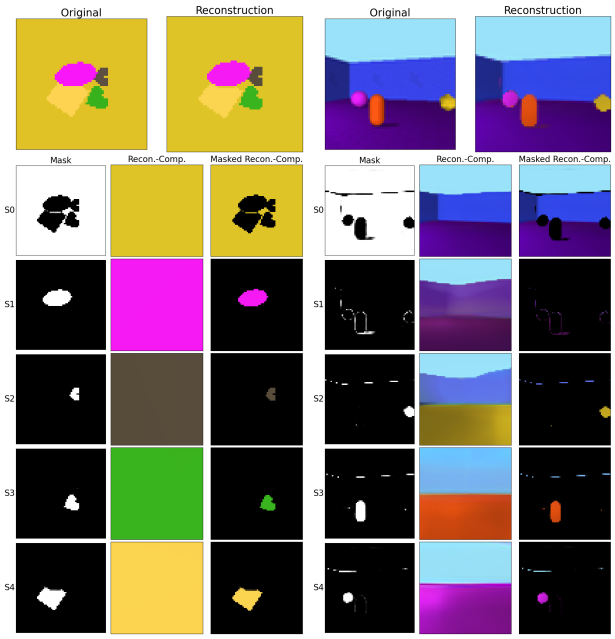
Generally, these points were small and our assumptions were founded in NN best practices. Given that our results closely aligned with those of the original paper, these assumptions are likely to be reasonable. `MONet` is a large ($1,207,921$ trainable parameters) recurrent NN, demanding significant computational resources to train. Even with WatGPU acceleration and several optimizations (e.g., NVIDIA cuDNN, automatic mixed-precision training, and many others), fully training `MONet` took on the order of 24 hours.

The two datasets chosen for demonstration were `MultiDSprites` and `ObjectsRoom` [5, 6]. These datasets exactly fit the task at hand: each consist of random collections of abstract objects (i.e., without grounded real-world correlates) forming two- and three-dimensional scenes. Each contain instances of occlusion and variable object counts, as well as in-distribution (ID) and out-of-distribution (OOD) testing data to gauge model generalization. Both of these datasets were explored in the original paper; they represent industry-standard testbeds that are simple enough to be feasibly implemented given our computational constraints.

Quantifying the performance of `MONet` is difficult because of the nature of the unsupervised learning paradigm (and the challenges inherent to assigning ground-truth labels in object-centric learning); as such, the original paper presents no numerical results and relies only on qualitative interpretation of outputs. To verify that our reimplementation replicates the original paper, we generate analogous output visualizations (see Fig. 1) and show that they generally mirror the results presented in the original paper. Additionally, to compare our own versions of `MONet`, we define a quantitative metric of model performance: the mean-squared-error (MSE) between the image and the model's complete reconstruction (see Table 1 in the Appendix). Overall, our results are strong and support the findings of the original paper.

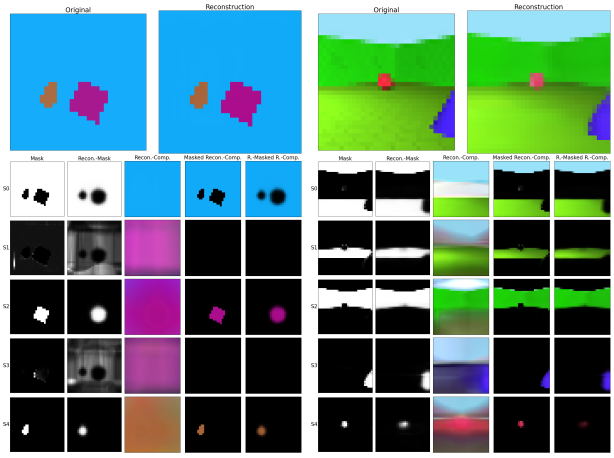
3 Extensions

We developed two extensions to our reimplementation of `MONet`: `NoRMaskMONet`, a version of the model where the VAE does not reconstruct the attention masks, and `MiniMONet`, a miniaturized version of the model designed for smaller images. Additionally, we explored applying `MONet` to a new domain: the Abstract and Reasoning Corpus for Artificial General Intelligence (`ARC-AGI`) [7].



(a) `MultiDSprites`: ID. (b) `ObjectsRoom`: ID.

Figure 2: Strong selected example outputs of `NoRMaskMONet`. Visualizations follow the same format as Fig. 1, but without the two columns corresponding to the reconstructed masks (which are no longer generated).



(a) `MultiDSprites`: ID. (b) `ObjectsRoom`: ID.

Figure 3: Strong selected example outputs of `MiniMONet`. Visualizations follow the same format as Fig. 1, but all images are lower-resolution (32×32 pixels instead of 64×64 pixels).

3.1 Version: NoRMaskMONet

Because the VAE generates the reconstructed attention masks from a simple latent code—a 16-dimensional Gaussian distribution—their quality is generally poor. These reconstructed masks are not used in any user-facing capacity (the attention-generated masks are used for output visualization), and the few shown in the original paper share this blurred appearance. Thus, we removed the reconstruction of masks by the VAE.

This extension involved modifying the VAE to reconstruct only the masked portion of the image, and no longer the mask itself. This produced a slightly smaller, faster, simpler version of `MONet` that still accomplished the same task: constructing a set of disentangled object representations (with segmentation masks) to accurately decompose (and reconstruct) a scene. Eliminating this feature rendered the third term of the loss function purposeless, so it was removed. This caused the generated masks to be more complex because the removed term of the loss function indirectly motivated the attention network to generate masks that were easy to reconstruct through the VAE bottleneck (i.e., simple masks). By removing this objective, the model learned to generate complex masks without a prohibitive penalty; this behaviour was especially apparent on the `ObjectsRoom` dataset, which contains more complicated scenes than the `MultiDSprites` dataset. Without this regularization-like training objective, the model focused purely on generating optimal component masks and reconstructions, causing it to obtain the best scores (see Table 1 in the Appendix).

Overall, the results of this version of the model were strong (see Fig. 2). These results are consistent with those of the original paper; `NoRMaskMONet` maintains strong performance, and our adjustment alters outputs in a manner consistent with `MONet`'s theoretical basis.

3.2 Version: MiniMONet

`MONet` requires substantial computational resources, so we endeavoured to create a lightweight model that could produce similar results for lower-resolution images. `MiniMONet` was designed to work on 32×32-pixel images, causing the model to have fewer learnable parameters. We experimented with removing certain layers

from the NNs as well as reducing the dimensionality of certain components, leading to further model size decreases. The final version of `MiniMONet` had 313,553 parameters (about one-quarter the size of `MONet`).

The original paper's authors use a 6-block-high U-Net to implement the attention network of `MONet` when working with 128×128-pixel images, and a 5-block-high U-Net when working with 64×64-pixel images. Following this pattern, `MiniMONet` was implemented using a 4-block-high U-Net. This construction pattern results in the same dimensionality of the deepest feature maps for all image resolutions, but with fewer feature maps. This makes the non-skip fully-connected MLP require fewer input and output nodes. Additionally, `MiniMONet`'s VAE uses one less layer in both the encoder and decoder, and fewer channels between each convolutional layer. To train `MiniMONet`, the `MultiDSprites` and `ObjectsRoom` datasets were scaled down using nearest-neighbour interpolation. This scaling strategy generally produced the cleanest separation of the unique elements of a scene, which is necessary to fairly assess the model's performance relative to `MONet`.

During the construction of `MiniMONet`, we attempted to reduce the latent dimensionality of the VAE from 16 dimensions to 8. This model struggled to accurately disentangle and reconstruct the elements of a scene, especially objects occupying fewer pixels. This suggests that the latent representation's dimensionality is integral to model success; objects within smaller images are still the same objects, so the same representational complexity is required. Our final version of `MiniMONet` used a 16-dimensional latent Gaussian distribution.

Overall, the results of this version of the model were weaker than those of `MONet` and `NoRMaskMONet`, but still strong (see Fig. 3). `MiniMONet` performs similarly to `MONet` when scenes are composed of few elements, but struggles to disentangle every unique element when there are many (see Table 1 in the Appendix). This suggests that the overall architecture of `MONet` may not be scalable down to little computational resources. Again, these results are consistent with those of the original paper; `MiniMONet` maintains some level of strong performance, and our adjustment alters outputs in a manner consistent with `MONet`'s theoretical basis.

3.3 Application: ARC-AGI

Our original motivation for choosing to reimplement `MONet` was its potential use in solving the `ARC-AGI` problem [7]. Although the model may perform a useful preprocessing step—extracting object-centric representations of the abstract building blocks that compose a scene—for solving `ARC-AGI`, `ARC-AGI` grids present many practical challenges to `MONet`. For example, their format (grids have variable sizes ranging from 1×1 to 30×30, complicating model interfacing), composition (some grids are not well-suited to an object-centric representation), and amount (there are only 7,000 grids in total, while the other datasets used contain 100,000 samples). While these issues can be partially mitigated (e.g., scaling grids up to size with nearest-neighbour interpolation, ignoring irrelevant examples, training in conjunction with other datasets), experimentation exposed several deeper weaknesses of the model for this application. For example, `MONet` is missing required inductive biases (e.g., interpolation of occluded objects and the notion of object persistence) and is overly colour-sensitive (e.g., it fails to differentiate nearby objects of identical colour and recognize multi-coloured objects). As such, results on `ARC-AGI` grids were poor (see Fig. 4 in the Appendix). In future research on `ARC-AGI`, other perceptual models better-suited to satisfying `ARC-AGI`-specific desiderata will be investigated.

4 Conclusion

Our results verify the aforementioned main claims of the paper as follows:

1. `MONet` is indeed a powerful unsupervised model for scene segmentation; while the state-of-the-art is rapidly evolving and difficult to define given the nature of the unsupervised learning paradigm, our results were strong, and no auxiliary information beyond the image itself is required by the model.
2. `MONet` indeed generates a disentangled representation of each constituent object in a common latent substrate; objects are generally separated into slots, each represented by a Gaussian distribution.
3. `MONet` indeed handles two- and three-dimensional scenes with occlusion and variable object counts, and generalizes to novel scene compositions; `MultiDSprites` contains two- and `ObjectsRoom` contains three-dimensional scenes, both datasets contain correctly-handled cases of occlusion and variable object counts, and performance on withheld and out-of-distribution testing data remains strong.

In summary, our project was a success. We reimplemented `MONet` [1], qualitatively replicating the original paper's results and verifying its claims. Additionally, we extended our reimplementation in multiple ways, producing generally strong results consistent with the claims and results of the original paper.

References

[1] C. P. Burgess, L. Matthey, N. Watters, R. Kabra, I. Higgins, M. Botvinick, and A. Lerchner, "MONet: Unsupervised Scene Decomposition and Representation," 2019. [Online]. Available: https://arxiv.org/abs/1901.11390

[2] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," 2022. [Online]. Available: https://arxiv.org/abs/1312.6114

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[4] N. Watters, L. Matthey, C. P. Burgess, and A. Lerchner, "Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes," 2019. [Online]. Available: https://arxiv.org/abs/1901.07017

[5] R. Kabra, C. Burgess, L. Matthey, R. L. Kaufman, K. Greff, M. Reynolds, and A. Lerchner, "Multi-Object Datasets," https://github.com/deepmind/multi-object-datasets/, 2019.

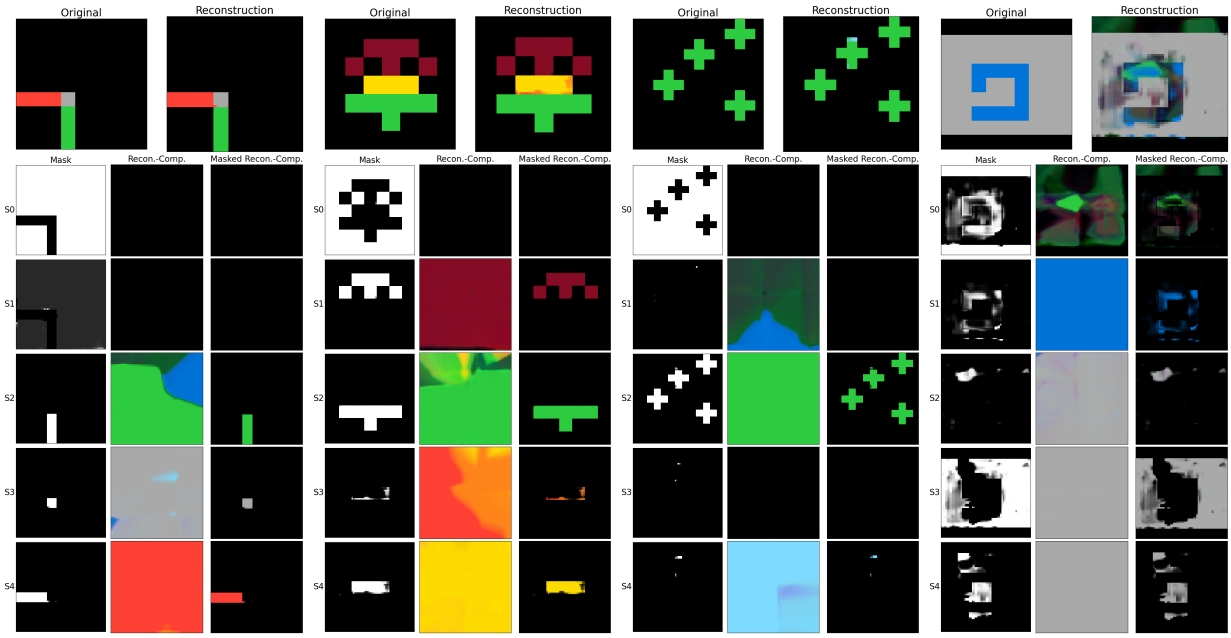
[6] ——, "Multi-Object Datasets," https://github.com/JohannesTheo/multi_object_datasets_torch/, 2019.

[7] F. Chollet, "On the Measure of Intelligence," 2019. [Online]. Available: https://arxiv.org/abs/1911.01547

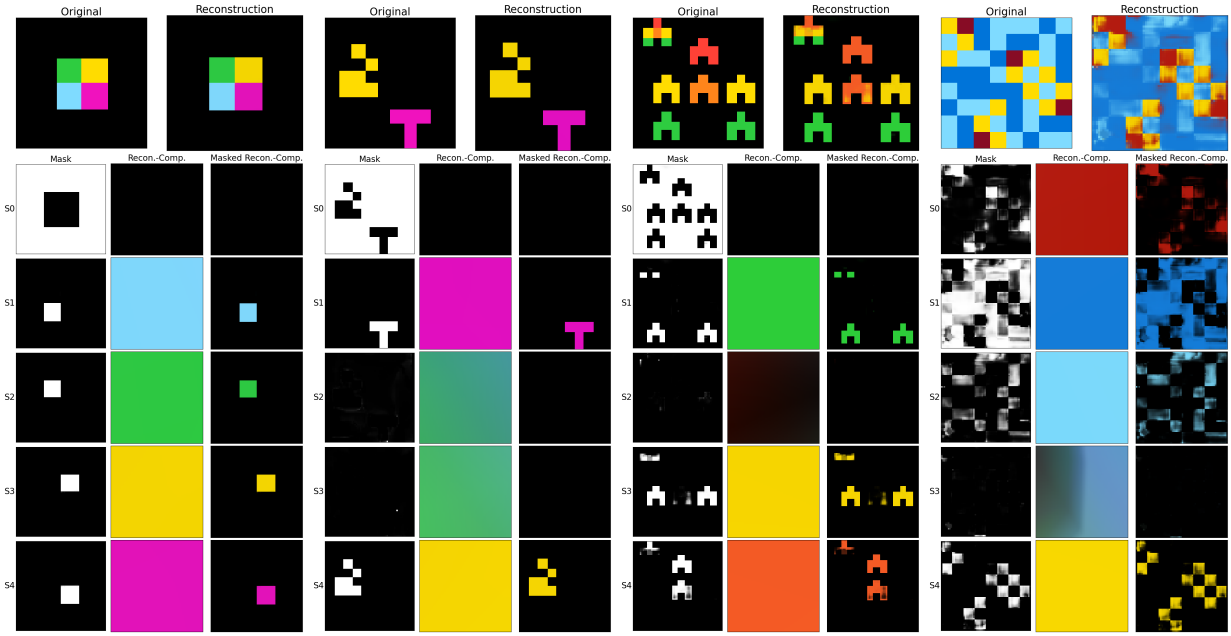
Appendix

Table 1: Performance comparison of all versions of our `MONet` models on all versions of the applicable testing datasets. Reconstruction scores were computed as the average pixel-wise mean-squared error (MSE) between the input image and the model's corresponding overall reconstruction, aggregated across each image in the testing dataset. Each ID testing dataset comprises $5,000$ withheld samples generated by the same underlying distribution used to generate the training dataset used to train the model, while the OOD testing datasets contain $5,000$ (`MultiDSprites`) or 922 (`ObjectsRoom`) samples of data generated by a similar, but novel, underlying distribution ("Grayscale" images have a random gray background and sometimes contain an additional object, "Empty Room" images contain no objects, "Identical Colour" images contain only objects of the exact same colour, and "Six Objects" images contain exactly six objects). All datasets and trained models used in this experiment are available in the `datasets/` and `models/` directories, respectively, of our submission repository (note that this table was generated by our `demo.ipynb` demonstration notebook, also included therein). Two-hundred random sample outputs generated by each trained model are available in the `results/` directory of our submission repository.

Model	Testing Dataset	Reconstruction Score (MSE)
`MONet_MultiDSprites`	`MultiDSprites`: ID	0.00042
	`MultiDSprites`: OOD: Grayscale	0.00078
`MONet_ObjectsRoom`	`ObjectsRoom`: ID	0.00119
	`ObjectsRoom`: OOD: Empty Room	0.00046
	`ObjectsRoom`: OOD: Identical Colour	0.00733
	`ObjectsRoom`: OOD: Six Objects	0.00728
`MONet_NoRMask_MultiDSprites`	`MultiDSprites`: ID	0.00013
	`MultiDSprites`: OOD: Grayscale	0.00041
`MONet_NoRMask_ObjectsRoom`	`ObjectsRoom`: ID	0.00098
	`ObjectsRoom`: OOD: Empty Room	0.00032
	`ObjectsRoom`: OOD: Identical Colour	0.00753
	`ObjectsRoom`: OOD: Six Objects	0.00398
`MONet_Mini_MultiDSprites`	`MultiDSprites`: ID	0.00071
	`MultiDSprites`: OOD: Grayscale	0.00133
`MONet_Mini_ObjectsRoom`	`ObjectsRoom`: ID	0.00225
	`ObjectsRoom`: OOD: Empty Room	0.00057
	`ObjectsRoom`: OOD: Identical Colour	0.00852
	`ObjectsRoom`: OOD: Six Objects	0.00867



(a) ARC-AGI: ID. (b) ARC-AGI: ID. (c) ARC-AGI: ID. (d) ARC-AGI: ID.



(e) ARC-AGI: ID. (f) ARC-AGI: ID. (g) ARC-AGI: ID. (h) ARC-AGI: ID.

Figure 4: Selected example outputs of two special MONet models on ARC-AGI grids. NoRMaskMONet models were used because this architecture generally achieved the strongest performance (see Table 1), especially on MultiDSprites data, which is qualitatively similar to ARC-AGI data. As such, visualizations follow the same format as Fig. 2. Subfigures (a)–(d) are the outputs of a model trained only on our custom ARC-AGI dataset, while subfigures (e)–(h) are the outputs of a model trained on the grayscale MultiDSprites dataset in conjunction with our custom ARC-AGI dataset. Many grids were decomposed well (e.g., subfigures (a), (b), (e), and (f)), but others were not, suggesting fundamental shortcomings of MONet (e.g., subfigures (c) and (g) show that objects of the same colour are not properly separated) and issues in its application to this domain (subfigures (d) and (h) show generally low-quality outputs).