# BINDING VIA RECONSTRUCTION CLUSTERING

**Klaus Greff, Rupesh Kumar Srivastava & Jürgen Schmidhuber**
The Swiss AI lab IDSIA, USI-SUPSI
Lugano, Switzerland
{klaus,rupesh,juergen}@idsia.ch

## ABSTRACT

Disentangled distributed representations of data are desirable for machine learning, since they are more expressive and can generalize from fewer examples. However, for complex data, the distributed representations of multiple objects present in the same input can interfere and lead to ambiguities, which is commonly referred to as the *binding problem*. We argue for the importance of the binding problem to the field of representation learning, and develop a probabilistic framework that explicitly models inputs as a composition of multiple objects. We propose an algorithm that uses a denoising autoencoder to dynamically bind features together in multi-object inputs through an Expectation-Maximization-like clustering process. The effectiveness of this method is demonstrated on artificially generated datasets of binary images, showing that it can even generalize to bind together new objects never seen by the autoencoder during training.

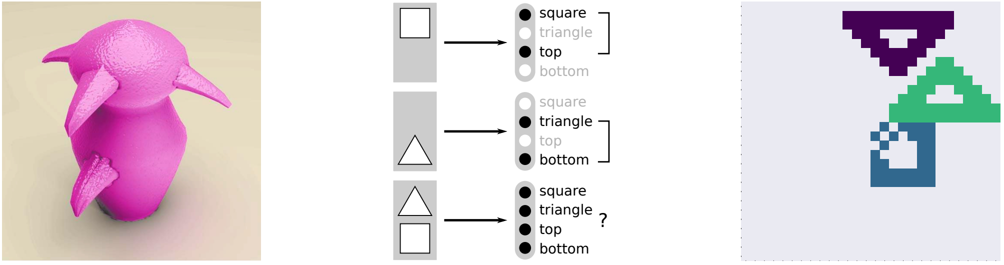
## 1 THE BINDING PROBLEM

Two important properties of good representations are that they are *distributed* and *disentangled*. Distributed representations (Hinton, 1984) are far more expressive than local ones, requiring exponentially fewer features to capture the same space. Complementary to that, disentangling (Barlow et al., 1989; Schmidhuber, 1992; Bengio et al., 2007) requires the factors of variation in the data to be separated into different independent features. This concept is closely related to invariance and eases further processing because many properties, that we might be interested in, are invariant under a wide variety of transformations (Bengio et al., 2013a). Unfortunately distributed representations can interfere and lead to ambiguities when multiple objects are to be represented at the same time.

The *binding problem* refers to these ambiguities that can arise from the superposition of multiple distributed representations. This problem has been debated quite extensively in the neuroscience and psychology communities perhaps starting with Milner (1974) and von der Malsburg (1981), but its existence can be traced back at least to a description by Rosenblatt (1961). It is classically demonstrated with a system required to identify an input as either square(□) or triangle(△) and to decide whether it is at the top(↑) or at the bottom(↓). It represents every object as a distributed representation with two active disentangled features (see Figure 1b). The binding problem arises when the system is presented with two objects at the same time: In this scenario, all four features become active and from the representation alone it cannot be determined whether the input contains a square on top and a triangle at the bottom or vice versa.

One way the system can circumvent this problem is through the use of a *local* representation, with one feature for each combination of shape and position: $\triangle_\uparrow, \triangle_\downarrow, \square_\uparrow, \square_\downarrow$. Sadly the size of such a purely local-representation scales exponentially with the number of factors to represent. In contrast, *distributed representations* (Hinton, 1984) are much more expressive and can generalize better through the reuse of features. The former system could, for example, correctly represent the position for a new object such as a circle by the already available position features (↑↓).

Generalization of internal representations is a crucial capability of any intelligent system, and one that still sets humans apart from current machine learning systems. Consider Figure 1a, an example from studies in psychology: chances are this is the first time you see a *Greeble* (Gauthier & Tarr, 1997). Nevertheless, you are capable of describing its shape, texture, and color. Moreover, you can easily segment it and tell it apart from the background, without having seen any other Greeble before.



(a) (b) (c)

Figure 1: (a) A Greeble. (b) Example demonstrating the binding problem (c) An illustration of intra-object predictability. The missing pixels from the square can be predicted using other pixels constituting the box, but not from pixels constituting other objects.

It has long been argued that such generalization capabilities are a result of the use of distributed representations in the human brain.

Despite its importance to the neuroscience community, binding has received relatively little attention in representation learning. Two important reasons for this are:

Firstly, most pattern recognition tasks and benchmarks are set up to avoid the binding problem. Many popular visual pattern recognition datasets consist of images that contain only one object at a time. Similarly, speech recognition mostly considers recordings of just one speaker talking and little background noise. In these settings, a machine learning algorithm can assume that there is only a single prominent object of interest, reducing the binding problem to the problem of ignoring irrelevant details. When tackling more challenging problems such as image caption generation, scene parsing segmentation, or the cocktail party problem, the deficiencies of popular methods become more apparent and restrictive.

Secondly, the recent increase in processing power due to the use of Graphics Processing Units made it feasible to mitigate the binding problem using localized binding in the form of convolutions (Riesenhuber & Poggio, 1999). Convolutional Networks (Fukushima, 1979; Le Cun et al., 1990) use feature detectors with limited receptive fields (filters) replicated over the whole input to represent its inputs. Therefore the resulting features of spatially separated objects do not interact: they are invariant to changes outside their field of view. On the other hand, they do not disentangle the location from the detected pattern, which comes at the cost of having to compute the same feature replicated many times over the image. While this is reasonable for low-level features like edges, it seems wasteful to replicate specialized high-level features such as dog-faces.

In this paper, we develop an unsupervised method that *dynamically* binds features of different objects together. This is in contrast to local representations which by nature *statically* bind several input features together (a feature for $\square_{\uparrow}$ permanently binds the concepts $\square$ and $\uparrow$ together). It explicitly models inputs as a composition of multiple entities and recovers these "objects" using the notion of mutual predictability. This is achieved through a clustering process which utilizes a denoising autoencoder (DAE; Behnke, 2001; Vincent et al., 2008) to iteratively reconstruct an input. In the future, such a mechanism could help to effectively use distributed representations in multi-object settings without being impaired by the ambiguities due to superposition. Alternative approaches to the binding problem proposed in the literature are discussed in Section 5.

## 2 Reconstruction Clustering

This section describes *Reconstruction Clustering* (RC), a formal framework for tackling the binding problem as a clustering problem. For ease of explanation we will refer to inputs as images and the individual dimensions of an input as pixels, though the framework is not restricted to visual inputs. It is based upon two insights: Firstly, if the image was segmented into its constituent objects, there would be no binding problem. Secondly, the intuitive notion of an object can be formalized as a group of mutually predictive pixels. The proposed method therefore iteratively clusters pixels based on how well they predict each other.

### 2.1 Images as Compositions

The first central idea behind RC is to model images as being composed of several independent objects with each pixel belonging to one of them. Unlike in classic segmentation where each pixel is assigned to a predefined class, the goal here is to simply segregate different objects. In doing so we avoid all ambiguities that might arise from a superposition of their representations. Of course, the information about which objects are present and which pixels they consist of is unknown in practical applications. So for each image, the aim is to infer both the object representations and the corresponding pixel assignments.

Similar to Le Roux et al. (2011), we introduce a binary latent vector $\mathbf{z_i}$ for each pixel $x_i$ that specifies which of the $K$ objects it belongs to. Therefore, $\mathbf{z_i} = (z_{i1}, z_{i2}, \ldots, z_{iK}) \in \{0,1\}^K$ with the constraint that $\sum_{k=1}^{K} z_{ik} = 1$. Let $N \in \mathbb{N}$ denote the number of pixels in the image $\mathbf{x} = \{x_1, \ldots, x_N\}$. Then we define the prior over $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ as:

$$P(\mathbf{Z}|\boldsymbol{\pi}) = \prod_{i=1}^{N} P(\mathbf{z}_i|\boldsymbol{\pi}) = \prod_{i=1}^{N}\prod_{k=1}^{K} \pi_k^{z_{ik}}, \tag{1}$$

where the $\mathbf{z}_i$'s are assumed to be independent given $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_K\}$. The assumed probabilistic structure is shown in Figure 2a. We assume $\boldsymbol{\pi}$ to be uniformly distributed for simplicity, but its estimation can be incorporated into the algorithm if required (see Appendix).

### 2.2 Objects

So far, we have used the word *object* to describe a group of pixels that somehow "belong together". The second central idea of RC is to concretize that notion using mutual predictability of the pixels. Intuitively, knowing about some pixel values that belong to an object helps in predicting the others. An example can be seen in Figure 1c where the corrupted pixels in the bottom left corner of the square could be reconstructed from knowledge about the rest of the square, but not from any of the triangles. So we define an object as a group of pixels that help in predicting each other, but do not carry information about pixels outside of that group.

Predictability, as we use it here, is derived from the structure of the underlying data-distribution. Knowledge about this structure is also precisely what is needed in order to remove corruption from an image. Based on this insight, we propose to use a denoising autoencoder (DAE) to measure predictability.

### 2.3 Denoising Autoencoder

Let $f$ be the encoder and $g$ be the decoder of a DAE, such that $\boldsymbol{\theta} = f(\mathbf{x})$ is the encoded representation of input $\mathbf{x}$ and $\boldsymbol{\mu} = g(\boldsymbol{\theta})$ is the decoded output. The DAE is trained to remove corruption from images of single objects and thus learns a local model of the data generating distribution (Vincent et al., 2008; Bengio et al., 2013b). After training the same DAE is used for each of the clusters to get predictions $\mu_{ik}$ from cluster $k$ for pixel $i$, where the object in cluster $k$ is represented by $\boldsymbol{\theta}_k$:

$$P(\mathbf{x}|\boldsymbol{\theta}_k) = \prod_{i=1}^{N} P(x_i|\boldsymbol{\theta}_k) = \prod_{i=1}^{N} P(x_i|\mu_{ik}) \tag{2}$$

Here $x_i$'s are assumed to be independent given $\boldsymbol{\mu}$. Combining this with the latent variables we get:

$$P(\mathbf{x}|\mathbf{Z}, \theta) = \prod_{i=1}^{N}\prod_{k=1}^{K} P(x_i|\mu_{ik})^{z_{ik}} \tag{3}$$

### 2.4 Clustering

We can now outline a clustering algorithm that estimates the object identities and the corresponding pixel assignments. Formally, we seek to maximize the complete data log-likelihood:

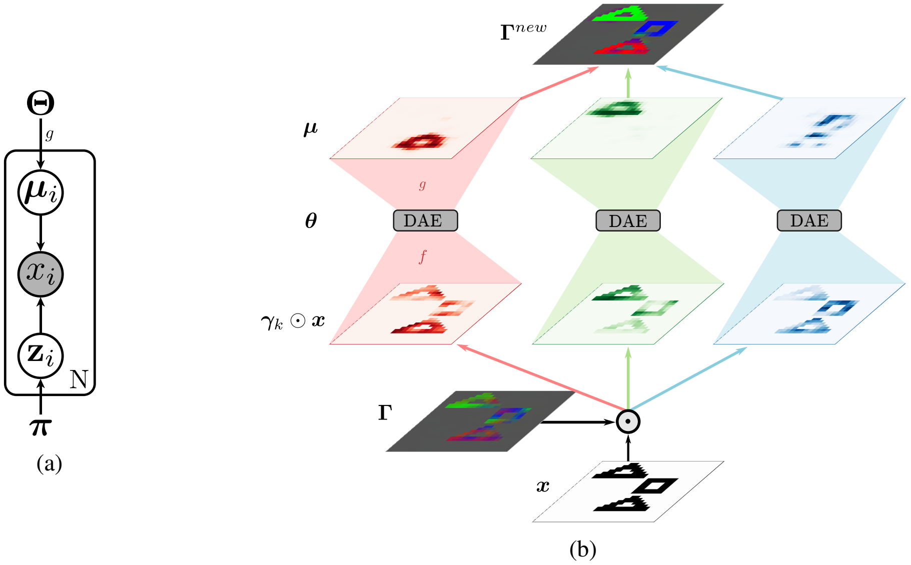


Figure 2: (a) The assumed probabilistic structure. (b) A schematic illustration of one iteration of the RC algorithm.

$$\log P(\mathbf{x}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\pi}) = \sum_{i=1}^{N}\sum_{k=1}^{K} z_{ik}(\log P(x_i|\mu_i) + \log \pi_k) \tag{4}$$

This can be done in an iterative procedure where we start by randomly initializing the latent cluster assignments $\mathbf{Z}$ and then alternating between the following two steps:

1. Apply the autoencoder to the each of the $K$ images that are assigned to the clusters to get a new estimate of the $K$ object representations. (R-step)
2. Re-assign the pixels to the clusters according to their reconstruction accuracy. (E-step)

### 2.4.1 Reconstruction Step

The R-step applies the encoder to generate a new object representation from each of the $K$ partial images that are assigned to each cluster. We call this representation of a partial image since the encoder only gets to see as much of each pixel of the original image as has been soft-assigned to the current cluster. The DAE then denoises the "corruption" caused by the cluster assignments. The R-Step is thus given by the following formula, where $\odot$ denotes point-wise multiplication:

$$\boldsymbol{\theta}_k = f(\boldsymbol{\gamma}_k \odot \mathbf{x}), \tag{5}$$

Unfortunately this step can not be guaranteed to increase the expected log-likelihood, because only in expectation does the DAE map from regions of low likelihood to regions of higher likelihood. Moreover, this property only holds for the whole image and not for all subsets of pixels. Thus, convergence can't be proven and RC is not an Expectation Maximization algorithm (Dempster et al., 1977). Nevertheless, empirical results show that convergence does occur reliably (Section 4.2).

### 2.4.2 Estimation Step

In the E-step, for each pixel $x_i$ the posterior $\gamma_{ik}$ of $\mathbf{Z}$, given the data and the predictions $\boldsymbol{\mu}_i = \{g(\boldsymbol{\theta}_1)_i, \ldots, g(\boldsymbol{\theta}_K)_i\}$ of the autoencoders based on the object representations, is

$$\gamma_{ik} = P(z_{ik} = 1|x_i, \boldsymbol{\mu_i}, \boldsymbol{\pi}) = \frac{P(x_i|z_{ik} = 1, \boldsymbol{\mu_i})P(z_{ik} = 1|\boldsymbol{\pi})}{P(x_i|\boldsymbol{\mu_i}, \boldsymbol{\pi})}. \tag{6}$$

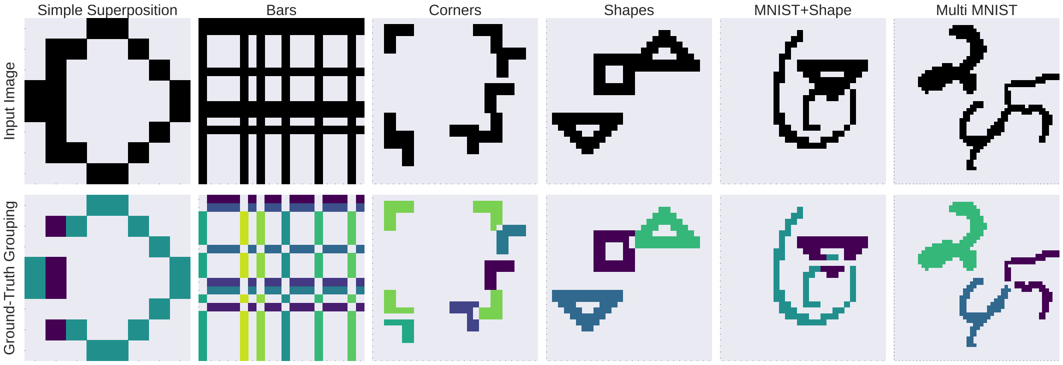


Figure 3: One example from each of the six datasets. The input images are shown on the top row with the corresponding ground-truth grouping below.

In this paper, we assume the pixels $\mathbf{x}$ to be binary and the predictions of the network $\mu$ to correspond to the mean of a binomial distribution. Then the following performs a soft-assignment of the pixels to the $K$ different clusters:

$$\gamma_{ik} = \frac{\mu_{ik}^{x_i}(1-\mu_{ik})^{1-x_i}\pi_k}{\sum_{j=1}^{K}\mu_{ij}^{x_i}(1-\mu_{ij})^{1-x_i}\pi_j}. \tag{7}$$

## 3 Experiments

We evaluated RC on a series of artificially generated datasets consisting of binary images of varying complexity. For each dataset, a DAE was trained to remove salt&pepper noise on images with single objects. The autoencoders used were fully-connected feed-forward neural networks with a single hidden layer and sigmoid output units. A random search was used to select appropriate hyperparameters (see Appendix for details). The best DAE obtained for each dataset was used for reconstruction clustering on 1000 test images containing multiple objects, and the binding performance was evaluated based on groud-truth object identities. All the code for this paper (including the creation of the datasets and figures) is available online on GitHub.com/Qwlouse/Binding.

### 3.1 Datasets

Representative examples from the datasets are shown in Figure 3.

**Simple Superposition** A collection of simple pixel patterns two of which are superimposed. Taken from Rao et al. (2008). This is a simple dataset with no translations, but significant overlap between patterns.

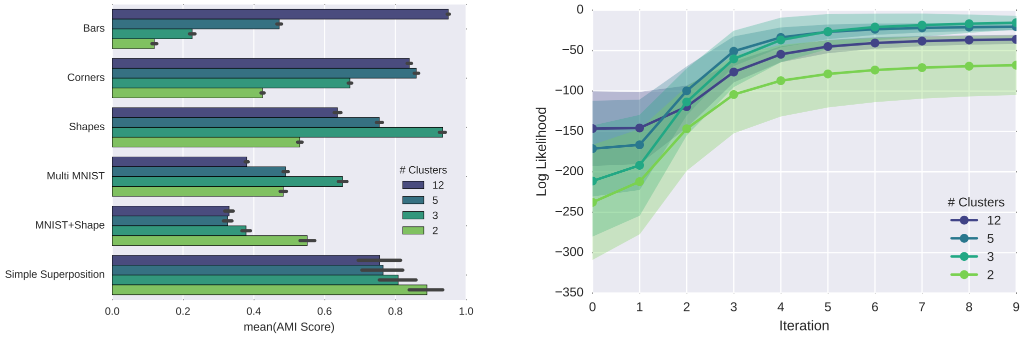
**Shapes** Taken from Reichert & Serre (2013). Three shapes ($\square, \triangle, \bigtriangledown$) are randomly placed in an image (possibly with overlap). This dataset tests binding of shapes under translation invariance and varying overlap.

**Bars** Introduced by Földiak (1990) to demonstrate unsupervised learning of independent components of an image. We use the variant from Reichert & Serre (2013) which employs 6 horizontal, and 6 vertical lines placed in random positions in the image.

**Corners** This dataset consists of 8 corner shapes placed in random orientations and positions, such that 4 of them align to form a square. It was introduced by Reichert & Serre (2013) to demonstrate that spatial connected-ness is not a requirement for binding.

**MNIST+Shape** Another dataset from Reichert & Serre (2013), which combines a random shape from the shapes dataset with a single MNIST digit. This dataset is useful to investigate binding multiple types of objects.

**Multi-MNIST** Three random MNIST digits are randomly placed in a 48×48 image. It provides a more challenging setup with multiple complex objects.



(a) Overall Scores

(b) Convergence

Figure 4: **Left:** Mean AMI score over 1000 test samples for all datasets and various number of clusters $K$. **Right:** Convergence of the log-likelihood on the *shapes* dataset for different numbers of clusters, showing test set mean (line) and standard deviation (shaded) over the test set.

### 3.2 Evaluation

Since the data is generated, a ground-truth segmentation for each image is available. For the binding task, all pixels corresponding to the same object should be clustered together. We evaluated performance by measuring the Adjusted Mutual Information (AMI; Vinh et al., 2010) between the true segmentation and the result of the binding, to which we refer to as the *score*. This score measures how well two cluster assignments agree and takes a value of 1 when they are equivalent, and 0 when their agreement corresponds to that expected by chance. Only pixels that unambiguously belong to one object were counted, ignoring background pixels and regions where multiple objects overlap.

## 4 Results

### 4.1 Scores

Figure 4a shows the mean scores obtained using RC for each dataset averaged over 100 runs. Scores obtained with different choices of the number of clusters $K$. Results are consistent across runs, hence the standard deviations are very low and barely visible. The optimal number of clusters is two for *Simple Superposition* and *MNIST+Shape*, three for *Multi MNIST* and *Shapes*, five for *Corners*, and 12 for *Bars*. Scores are higher than 0.5 for all datasets and higher than 0.8 for four out of the six datasets demonstrating the ability of RC to successfully bind objects together.

### 4.2 Convergence

Figure 4b shows the convergence of the mean log-likelihood over RC iterations on the *shapes* dataset. Convergence is quick, typically within 5-10 iterations, depending on the chosen number of clusters $K$ and the dataset (not shown). As expected, the final likelihood is highest when the number of clusters equals the number of objects in the shapes dataset (3), matching the results from Figure 4a. The likelihood is much lower for $k = 2$ than for $k = 3$ and drops again slightly if we choose $k = 5$. The likelihood for $k = 12$ is significantly lower. In some cases the correct choice of $k$ did not result in the highest likelihood, but in general this correspondence appeared to hold. If the number of objects is unknown, this trend can be used to determine the correct number of clusters.

### 4.3 Qualitative Analysis

Figure 5 shows a few example RC runs of on the shapes dataset for qualitative evaluation. The initial cluster assignments are random, therefore all observed structure is due to the clustering process. The final clustering corresponds well to the ground truth even for cases with significant overlap. Again, it is notable that RC converges quickly (within 5 iterations).

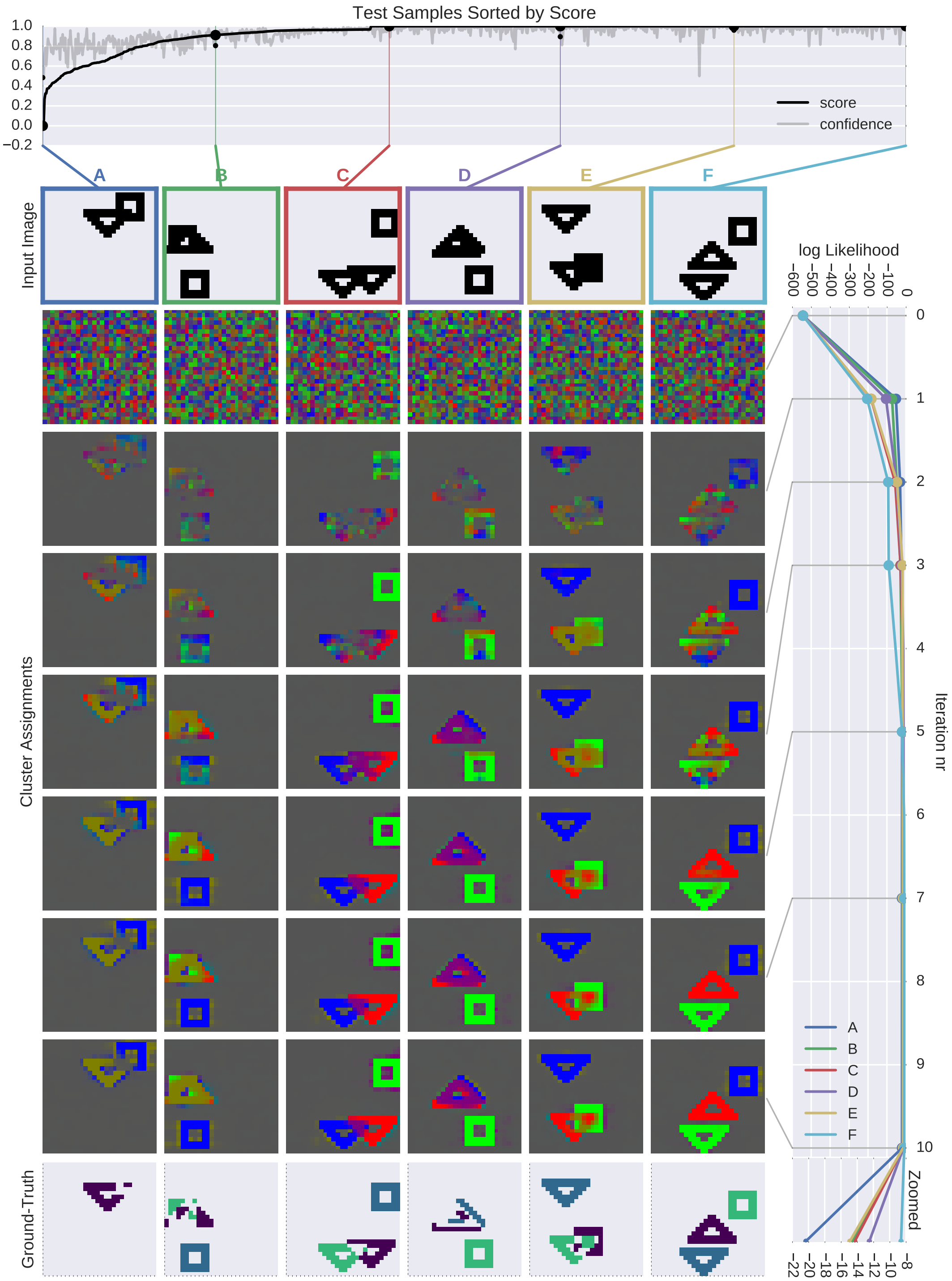
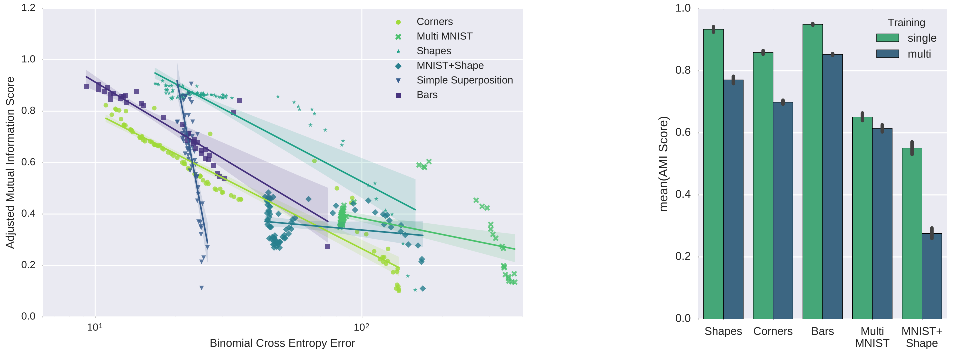


Figure 5: The top plot shows the score and confidence for each of the 1000 test images from the shapes dataset, sorted by score. The confidence is the average value of $\max_k \gamma_{ik}$ for each evaluated pixel (non-background, non-overlap). The central part of the figure shows six examples (columns) along with the cluster assignments (indicated by different colors) over RC iterations. The corresponding ground-truth is shown at the bottom. The right vertical plot shows the log-likelihood over the RC iterations corresponding to the displayed cluster assignments. Similar plots for the other datasets are included in the Appendix.



(a) Loss Vs Score

(b) Single vs. Multi-object DAE training

Figure 6: **Left:** Relationship between the DAE loss and the AMI score. All networks have 250 hidden units and were trained with random learning rates and initializations. A few networks that failed to train were removed from the plot for better visualization. **Right:** RC scores obtained when training DAEs on multi-object images vs. single object images.

### 4.4 Loss vs Score

RC utilizes autoencoders trained with the denoising objective for binding. Therefore, it is instructive to examine the relationship between denoising performance and the final RC binding score. For this purpose, we trained 100 DAEs with the same architecture on each dataset with random learning rates and initializations, and then performed RC using each of them. Figure 6a shows the relationship between the denoising loss and binding score for each dataset. It can be observed that lower loss correlates positively with higher score for all datasets, indicating that denoising is a suitable surrogate training objective. We added a regression line to indicate that relation for each dataset, even though for *MNIST+Shape* and *Multi MNIST* it doesn't look even remotely linear. Instead, the individual points are approximately arranged on a curve. This suggests that there is a direct but complex interplay between the denoising performance and the score.

### 4.5 Training on Multiple Objects

So far the DAEs were trained on single-object images, then used to bind objects in multi-object images. In general it is desirable to not *require* single-object images for training, and be able to directly use any image without this restriction. This would remove the last bit of supervision and make RC a truly unsupervised method.

Why should this work at all? On the surface it seems that RC would depend on the DAE to prefer single objects in order to work correctly. However, even if each cluster tries to reconstruct every object, there will be small asymmetries due to the difference in inputs they see. Since no object carries any information about the shape and position of another object in our datasets, this will lead to differences in prediction quality of the objects. The resulting difference in reconstruction quality will then be amplified by RC and can still lead to a segregation of the objects.

To test this scenario, we performed a new random search to tune DAE hyperparameters for the case of multi-object training. Similar to the single-object case, we then used the best obtained DAEs to perform RC on test examples. We found that with soft-assignments to the clusters, the differences were too small and would even out over several iterations, leading to uniform cluster assignments. By changing the E-step to hard (K-Means-like) assignments, we were able to amplify these changes enough to make the whole procedure work. Figure 6b shows that DAEs trained on multi-object images can indeed be used for binding via RC with hard assignments, although they lead to lower scores in comparison. Further discussion and examples for this case can be found in Appendix C.

### 4.6 Generalization to a new domain

A central intuition behind our approach to binding is that the low-level structures learned by the model will generalize to new and unseen configurations. Evaluation on unseen test sets demonstrated

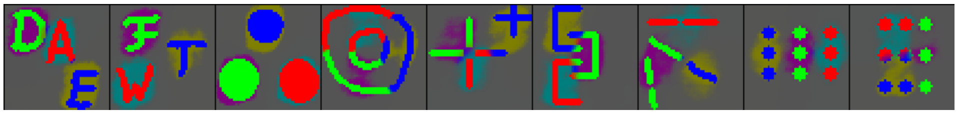


Figure 7: Binding novel objects via RC. The DAE used was trained on the *Multi MNIST* dataset.

this to be true, but we can take it one step further. We can test what happens when we confront our method with novel objects that the auto-encoders have not been trained on.

We ran RC on several images with non-digits using a DAE trained on the *Multi-MNIST* dataset. Figure 7 shows that RC "correctly" binds letters and circles together. We also show images for which the resulting binding differs from our expectation. It appears that the network has mainly learned to bind based on spatial proximity with a slight bias towards vertical proximity. This can be expected since that it has only seen digits of roughly the same size so far, and because the used autoencoder is very limited. Nevertheless, it is very interesting that a fully-connected network which is permutation invariant learns the preference for spatial proximity entirely from data. It is reasonable to speculate that it in the future it may be possible to recover other *Gestalt Principles* such as continuity and similarity with a similar procedure.

## 5 Relationship to other Methods

The binding problem and its possible solutions are a long standing debate in the neuroscience literature (see e.g. Milner (1974); von der Malsburg (1981); Gray (1999); Treisman (1999); Di Lollo (2012)). A major thread of work on binding has been inspired by the temporal correlation theory (von der Malsburg, 1981), based on utilizing synchronous oscillations to bind neuronal activities together. von der Malsburg (1995) provides an overview of these ideas. Recently, these ideas were implemented using complex valued activations in neural networks to jointly encode firing rate and phase (Rao et al., 2008; Reichert & Serre, 2013). Such binding mechanisms are close to their biological inspiration, clustering only implicitly through synchronization. In contrast, RC is based on a mathematical framework which explicitly incorporates binding.

Mechanisms for tackling the binding problem which do not require temporal synchronization have also been proposed (e.g. O'Reilly et al., 2003). O'reilly & Busby (2002) argued that the intuitive explanation of the binding problem from Figure 1b only applies if the distributed features themselves are local codes. They suggested that neural networks can avoid the binding problem using coarse-coded representations. Various feature representation types including coarse-coding and their limitations were described by Hinton (1984).

In principle, Recurrent Neural Networks (RNNs; e.g. Robinson & Fallside, 1987; Werbos, 1988) can solve the binding problem by learning a mechanism to avoid it. Psychologists (Di Lollo, 2012) and machine learning researchers (Weng et al., 2006) alike have suggested feedback as a mechanism to do binding. An RNN may utilize an implicit or explicit attention mechanism to selectively process different parts of the input (Schmidhuber & Huber, 1991; Mnih et al., 2014; Bahdanau et al., 2014). In this context, explicit binding via RC can be seen as a technique of paying attention to multiple objects at once, instead of focusing on them sequentially.

The core ideas of RC are similar to Masked Restricted Boltzmann Machines (MRBM) introduced by Le Roux et al. (2011). They model an image as being composed of multiple layers (clusters) and also add a corresponding latent variable for each pixel. Similar to the DAEs used for RC each layer is modelled by a separate RBM all of which share weights. Le Roux et al. (2011) further parameterize the model for the shape of masks (cluster assignments) and explicitly model occlusion. The main difference is that for RC we use DAEs together with a simple clustering mechanism instead of RBMs to perform inference. This simplifies training and seems to speed up convergence.[1]

[1] Since the datasets used for evaluation differ this comparison isn't very informative. That being said, Le Roux et al. (2011) report results for 5000 steps of Gibbs sampling, while our model typically converges within 10 iterations. This speedup is also in line with the observations of Reichert & Serre (2013), whose model also builds upon RBMs and typically takes 100 steps to converge.

Structurally, RC resembles a model introduced by Weng et al. (2006). They too split the image representation into competing feature layers (objects). Inter-object predictability is modelled by lateral connections that represent compatibility between features. These connections are trained from labelled samples using Hebbian learning. After training, the clustering for an input is obtained by finding a fixed point of the energy function defined over the layers using the Gauss-Seidel method.

In some aspects, RC is also similar to segmentation algorithms. The main difference is that RC learns the segmentation from the data in a largely unsupervised manner. In this sense, it is more similar to *superpixel* methods (see e.g. Achanta et al. (2012) for an overview). However, these methods impose a handcrafted similarity measure over pixels or pixel regions, whereas RC learns a non-linear similarity measure from the data, parameterized by a DAE.

## 6 Conclusion and Future Work

We introduced the Reconstruction Clustering framework to explicitly model data as a composition of objects, where the notion of object-ness is defined by mutual predictability. Compared to many previous solutions to the binding problem, this framework is mathematically rigorous, integrates well with current representation learning methods, and is effective for a variety of binary image datasets. While a typical representation learning method (such as a denoising autoencoder) learns a *static binding* of features, Reconstruction Clustering utilizes it to iteratively perform *dynamic binding* for every input example by introducing interaction between the statically bound features extracted by the autoencoder. In particular, this interaction enables dynamic binding of feature combinations never seen before by the autoencoder.

This paper lays the groundwork for many concrete lines of future exploration. The treatment of real-valued inputs is an important next step to extension RC towards natural data. Also the use of more powerful autoencoders will be key. Integrating RC with the training of the DAE should help to deal with multiple objects in the training data. Since the method is general, we expect to apply it to other domains such as audio data (binding different speaker voices together) or medical data (binding various related symptoms of disease together). A particularly interesting direction for future work is to show that Gestalt principles are a natural result of such a representation learning approach.

### Acknowledgments

We thank Jan Koutník, Sjoerd van Steenkiste, Boyan Beronov and Julian Zilly for helpful discussions and comments. This project was funded by the EU project NASCENCE (FP7-ICT-317662).

## References

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2274–2282, 2012. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6205760. bibtex: achanta2012.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. URL http://arxiv.org/abs/1409.0473. bibtex: bahdanau_neural_2014.

Horace B. Barlow, Tej P. Kaushal, and Graeme J. Mitchison. Finding minimum entropy codes. *Neural Computation*, 1(3):412–423, 1989. URL http://www.mitpressjournals.org/doi/abs/10.1162/neco.1989.1.3.412.

Sven Behnke. Learning iterative image reconstruction in the Neural Abstraction Pyramid. *International Journal of Computational Intelligence and Applications*, 1(04):427–438, 2001. URL http://www.worldscientific.com/doi/pdf/10.1142/S1469026801000342.

Yoshua Bengio, Yann LeCun, and others. Scaling learning algorithms towards AI. *Large-scale kernel machines*, 34(5), 2007. URL http://www.iro.umontreal.ca/~lisa/bib/pub_subject/language/pointeurs/bengio+lecun-chapter2007.pdf.

Yoshua Bengio, Aaron Courville, and Pierre Vincent. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1798–1828, 2013a. bibtex: bengio2013a.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, pp. 899–907, 2013b. bibtex: bengio2013.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977. bibtex: dempster1977.

Vincent Di Lollo. The feature-binding problem is an ill-posed problem. *Trends in Cognitive Sciences*, 16(6):317–321, 2012. URL http://www.sciencedirect.com/science/article/pii/S1364661312000988. bibtex: dilollo2012.

Peter Földiak. Forming sparse representations by local anti-Hebbian learning. *Biological cybernetics*, 64(2):165–170, 1990. URL http://link.springer.com/article/10.1007/BF02331346. bibtex: foldiak1990.

K. Fukushima. Neural network model for a mechanism of pattern recognition unaffected by shift in position - Neocognitron. *Trans. IECE*, J62-A(10):658–665, 1979.

Isabel Gauthier and Michael J. Tarr. Becoming a "Greeble" expert: Exploring mechanisms for face recognition. *Vision Research*, 37(12):1673–1682, June 1997. ISSN 0042-6989. doi: 10.1016/S0042-6989(96)00286-6. URL http://www.sciencedirect.com/science/article/pii/S0042698996002866.

Charles M. Gray. The temporal correlation hypothesis of visual feature integration: Still alive and well. *Neuron*, 24(1):31–47, September 1999. ISSN 0896-6273. doi: 10.1016/S0896-6273(00)80820-X. URL http://www.sciencedirect.com/science/article/pii/S089662730080820X. bibtex: Gray1999.

Geoffrey E. Hinton. Distributed representations. 1984. URL http://repository.cmu.edu/cgi/viewcontent.cgi?article=2841&context=compsci. bibtex: hinton1984.

B. Boser Le Cun, John S. Denker, D. Henderson, Richard E. Howard, W. Hubbard, and Lawrence D. Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*. Citeseer, 1990. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.5076. bibtex: lecun_handwritten_1990.

Nicolas Le Roux, Nicolas Heess, Jamie Shotton, and John Winn. Learning a generative model of images by factoring appearance and shape. *Neural Computation*, 23(3):593–650, 2011.

Peter M. Milner. A model for visual shape recognition. *Psychological review*, 81(6):521, 1974. URL http://psycnet.apa.org/journals/rev/81/6/521/. bibtex: milner1974.

Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pp. 2204–2212, 2014. URL http://papers.nips.cc/paper/5542-recurrent-models-of-visual-attention. bibtex: mnih_recurrent_2014.

Randall C. O'reilly and Richard S. Busby. Generalizable relational binding from coarse-coded distributed representations. *Advances in neural information processing systems*, 1:75–82, 2002. URL https://grey.colorado.edu/mediawiki/sites/CompCogNeuro/images/e/e6/OReillyBusby02.pdf. bibtex: oreilly2002.

Randall C. O'Reilly, Richard S. Busby, and Rodolfo Soto. Three forms of binding and their neural substrates: Alternatives to temporal synchrony. *The unity of consciousness: Binding, integration, and dissociation*, pp. 168–192, 2003. URL http://chemistry47.com/PDFs/Cognition/Neuronal%20Synchrony/Three%20Forms%20of%20Binding%20and%20their%20Neural%20Substrates%20Alternatives%20to%20Temporal%20Synchrony.pdf. bibtex: oreilly2003.

Ravishankar A. Rao, Guillermo Cecchi, Charles C. Peck, and James R. Kozloski. Unsupervised segmentation with dynamical units. *Neural Networks, IEEE Transactions on*, 19(1):168–182, 2008. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4359215. bibtex: rao2008.

David P. Reichert and Thomas Serre. Neuronal synchrony in Complex-Valued deep networks. *arXiv:1312.6115 [cs, q-bio, stat]*, December 2013. URL http://arxiv.org/abs/1312.6115. arXiv: 1312.6115 bibtex: Reichert2013.

Maximilian Riesenhuber and Tomaso Poggio. Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019–1025, 1999. URL http://www.nature.com/neuro/journal/v2/n11/abs/nn1199_1019.html.

A. J. Robinson and F. Fallside. The utility driven dynamic error propagation network. Technical Report CUED/F-INFENG/TR.1, Cambridge University Engineering Department, 1987.

Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, DTIC Document, 1961. bibtex: rosenblatt1961.

Juergen Schmidhuber and Rudolf Huber. Learning to generate artificial fovea trajectories for target detection. *International Journal of Neural Systems*, 2(01n02):125–134, 1991. URL http://www.worldscientific.com/doi/abs/10.1142/S012906579100011X. bibtex: schmidhuber_learning_1991.

Jürgen Schmidhuber. Learning factorial codes by predictability minimization. *Neural Computation*, 4 (6):863–879, 1992. URL http://www.mitpressjournals.org/doi/abs/10.1162/neco.1992.4.6.863. bibtex: schmidhuber1992.

Anne Treisman. Solutions to the binding problem: Progress through controversy and convergence. *Neuron*, 24(1):105–125, September 1999. ISSN 0896-6273. doi: 10.1016/S0896-6273(00)80826-0. URL http://www.sciencedirect.com/science/article/pii/S0896627300808260. bibtex: Treisman1999.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103. ACM, 2008. URL http://dl.acm.org/citation.cfm?id=1390294. bibtex: vincent2008.

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. URL http://dl.acm.org/citation.cfm?id=1953024. bibtex: vinh2010.

Christoph von der Malsburg. The correlation theory of brain function. 1981. bibtex: vondermalsburg1981.

Christoph von der Malsburg. Binding in models of perception and brain function. *Current opinion in neurobiology*, 5(4):520–526, 1995. URL http://www.sciencedirect.com/science/article/pii/095943889580014X. bibtex: vondermalsburg1995.

Shijie Weng, Jochen Jakob Steil, and Helge Ritter. Learning lateral interactions for feature binding and sensory segmentation from prototypic basis interactions. *Neural Networks, IEEE Transactions on*, 17(4):843–862, 2006. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1650242.

P. J. Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, 1, 1988.

## A Reconstruction Clustering Derivation

This section contains a more detailed derivation of Reconstruction Clustering (RC) for binary inputs. It follows the notation and derivation of an Expectation Maximization (EM) algorithm wherever possible. Only for the M-step does RC deviate from EM.

Consider $N$ binary random variables (one for each pixel) that are distributed according to a mixture of $K$ Bernoulli distributions with means $\boldsymbol{\mu}_i = (\mu_{i1}, \mu_{i2}, \dots, \mu_{iK})$ and mixing coefficients $\boldsymbol{\pi} = (\pi_1, \pi_2, \dots, \pi_K)$ that sum to one $\sum_{k=1}^{K} \pi_k = 1$. Under this model the data likelihood given the parameters is given by:

$$P(x_i|\boldsymbol{\mu}_i, \boldsymbol{\pi}) = \sum_{k=1}^{K} \pi_k \mu_{ik}^{x_i}(1-\mu_{ik})^{1-x_i} \tag{8}$$

By defining $\mathbf{x} = (x_1, x_2, \dots, x_N)$ and $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_N)$ and assuming independence of the $x_i$'s given $\boldsymbol{\mu}$ and $\boldsymbol{\pi}$ (but not identical distribution)[2] we get the (incomplete) log likelihood function for this model:

$$\log P(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\pi}) = \sum_{i=1}^{N} \log P(x_i|\boldsymbol{\mu}_i, \boldsymbol{\pi}) \tag{9}$$

Let us now introduce an explicit binary latent variable $\mathbf{z_i} = (z_{i1}, z_{i2}, \dots, z_{iK}) \in \{0,1\}^K$ with $\sum_{k=1}^{K} z_{ik} = 1$ associated with each $x_i$. Let the prior distribution be:

$$P(\mathbf{Z}|\boldsymbol{\pi}) = \prod_{i=1}^{N} P(\mathbf{z}_i|\boldsymbol{\pi}) = \prod_{i=1}^{N}\prod_{k=1}^{K} \pi_k^{z_{ik}}, \tag{10}$$

where we set $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_N)$ and assume $\mathbf{z}_i$'s to be independent given $\boldsymbol{\pi}$. With that we define the conditional distribution of $x_i$ given the latent variables as:

$$\begin{align} P(x_i|\mathbf{z}_i, \boldsymbol{\mu}_i) &= \prod_{k=1}^{K} P(x_i|\mu_{ik})^{z_{ik}} \tag{11}\\ &= \prod_{k=1}^{K} (\mu_{ik}^{x_i}(1-\mu_{ik})^{1-x_i})^{z_{ik}} \tag{12}\end{align}$$

If we marginalize Equation 12 over all choices of $\mathbf{z}_i$ we recover Equation 8:

$$\begin{align} \sum_{\mathbf{z}} P(x_i|\mathbf{z}, \boldsymbol{\mu}_i, \boldsymbol{\pi})P(\mathbf{z}|\boldsymbol{\pi}) &= \sum_{\mathbf{z}} \prod_{k=1}^{K} (\mu_{ik}^{x_i}(1-\mu_{ik})^{1-x_i})^{z_k} \pi_k^{z_k} \tag{13}\\ &= \sum_{k=1}^{K} \pi_k \mu_{ik}^{x_i}(1-\mu_{ik})^{1-x_i} \tag{14}\\ &= P(x_i|\boldsymbol{\mu}_i, \boldsymbol{\pi}) \tag{15}\end{align}$$

The second line is obtained by realizing that the $\sum_{\mathbf{z}}$ sums over exactly $K$ terms, each corresponding to a $\mathbf{z}$ with one $z_k = 1$ and all other entries equal to zero. So we can replace this sum by $\sum_{k=1}^{K}$. The product over the entries of $\mathbf{z}$ then vanishes except for the term corrsponding to the $k$-th entry.

[2] This assumption means that we assume the hidden representation of the DAE to capture the structure in the image well.

Using the same conditional independence assumption from before we can thus write the data distribution given all the latent variables as follows:

$$P(\mathbf{x}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\pi}) = \prod_{i=1}^{N} P(x_i|\mathbf{z}_i, \boldsymbol{\mu}_i, \boldsymbol{\pi}) \tag{16}$$

And by using Bayes rule and assuming that $\mathbf{Z}$ is independent of $\boldsymbol{\mu}$:

$$\begin{aligned} P(\mathbf{x}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\pi}) &= P(\mathbf{x}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\pi}) P(\mathbf{Z}|\boldsymbol{\pi}) && (17) \\ &= \prod_{i=1}^{N} \prod_{k=1}^{K} (\mu_{ik}^{x_i}(1-\mu_{ik})^{1-x_i}\pi_k)^{z_{ik}} && (18) \end{aligned}$$

If we set $\boldsymbol{\theta} = \{\boldsymbol{\mu}, \boldsymbol{\pi}\}$[3] the complete-data log likelihood becomes:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}|\mathbf{x}, \mathbf{Z}) &= \log P(\mathbf{x}, \mathbf{Z}|\boldsymbol{\mu}, \boldsymbol{\pi}) && (19) \\ &= \sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \left[x_i \log \mu_{ik} + (1-x_i)\log(1-\mu_{ik}) + \log \pi_k\right] && (20) \end{aligned}$$

To maximize $\mathcal{L}$ with respect to $\boldsymbol{\theta}$ and $\mathbf{Z}$ we follow the same idea as the EM algorithm: Based on the observation that if we knew the values of either of these two, optimizing the other would be feasible. So we divide the optimization problem into two steps where we pretend to know either $\boldsymbol{\theta}$ (E-step) or $\mathbf{Z}$ (M-step).

In the E-Step we assume to know $\boldsymbol{\theta}$ and calculate the posterior probability of $z_{ik} = 1$ for each datapoint calling it $\gamma_{ik}$: (We assume the $z_{ik}$ to be independent of $x_j$ for $i \neq j$)

$$\begin{aligned} \gamma_{ik} = P(z_{ik} = 1|x_i, \boldsymbol{\mu_i}, \boldsymbol{\pi}) &= \frac{P(x_i|z_{ik} = 1, \boldsymbol{\mu_i}) P(z_{ik} = 1|\boldsymbol{\pi})}{P(x_i|\boldsymbol{\mu_i}, \boldsymbol{\pi})} && (21) \\ &= \frac{\mu_{ik}^{x_i}(1-\mu_{ik})^{1-x_i}\pi_k}{\sum_{j=1}^{K} \mu_{ij}^{x_i}(1-\mu_{ij})^{1-x_i}\pi_j} && (22) \end{aligned}$$

Next we calculate the $\mathcal{Q}$ value used in EM which is defined as the expectation of the complete data log-likelihood $\mathcal{L}$ with respect to the posterior of $\mathbf{Z}$ given the data and the old parameters $\boldsymbol{\theta}^{old}$:

$$\begin{aligned} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) &= E_{\mathbf{Z}}[\log P(\mathbf{x}, \mathbf{Z}|\boldsymbol{\theta})|\mathbf{x}, \boldsymbol{\theta}^{old}] && (23) \\ &= \sum_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{x}, \boldsymbol{\theta}^{old}) \log P(\mathbf{x}, \mathbf{Z}|\boldsymbol{\theta}) && (24) \\ &= \sum_{i=1}^{N} \sum_{k=1}^{K} \gamma_{ik} \left[x_i \log \mu_{ik} + (1-x_i)\log(1-\mu_{ik}) + \log \pi_k\right] && (25) \end{aligned}$$

In the M-step of EM we aim to maximize $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{old})$ over all choices of $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^{new} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) \tag{26}$$

[3] Here we deviate slightly from the notation in the paper.

Using a Lagrange multiplier to enforce $\sum_{k=1}^{K} \pi_k = 1$ we find:

$$\pi_k^{new} = \frac{\sum_{i=1}^{N} \gamma_{ik}}{N} \tag{27}$$

But when maximizing wrt. $\boldsymbol{\mu}$ we see that the maximum is trivially obtained by setting $\mu_{ik} = x_i$ for all $k$. This is due to the fact that the problem is actually ill-posed in the sense that we have $K$ parameters to fit for each datapoint. So there are infinitely many solutions which achieve the optimal log likelihood of the data of 0.

At this point we introduce an autoencoder with encoder $f$ and decoder $g$ to restrict the capacity of our model by forcing $\mu$ to be:

$$\boldsymbol{\mu}_{\cdot k} = g(f(\boldsymbol{\gamma}_k \odot \mathbf{x})) \tag{28}$$

We use this reconstruction step (Equation 28) instead of an actual maximization step, thus deviating from the EM formulation.

# B Training Details

All experiments have been performed with the brainstorm library and were organized and logged using sacred. The code for this paper can be found on GitHub.

## B.1 Training Denoising Autoncoders

- simple feed forward fully connected NNs
- with sigmoid output layer
- loss is Binomial Cross Entropy Error
- trained with SGD
- minibatch size 100
- salt& pepper noise
- early stopped when validation BinomialCEE doesn't decrease for more than 10 epochs

## B.2 Random Search

There are several hyperparameters to be chosen for the denoising autoencoders. To find good values we performed a random search with 100 runs for each dataset. For each run we randomly sampled from the following parameters:

- learning rate log-uniform from $[10^{-3}, 1]$
- Amount of Salt& Pepper Noise from $[0.0, 0.1, \ldots, 0.9]$
- hidden layer size from $[100, 250, 500, 1000]$
- hidden layer activation function from [rel, sigmoid, tanh]

The best network configurations found by that search can be found in Table 1.

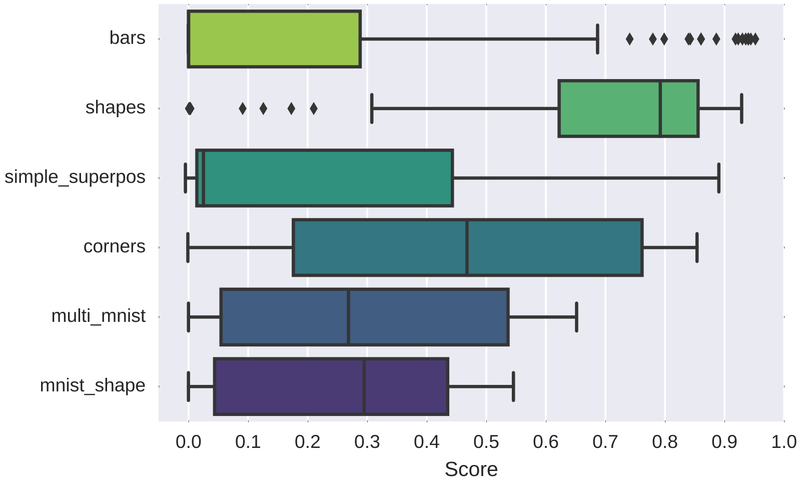


Figure 8: Summary of the scores achieved during the random search

## B.3 Random Search for Training with Multiple Objects

For training with multiple objects we do an equivalent random search for hyperparameters. The only difference is the training data and that for determining the final score we use K-means-like (hard) cluster assignments in RC. Note also that we didn't include the Simple Superposition dataset, since we only have 120 images with multiple objects available, and no separate test set.

| Dataset | learning rate | # hidden units | activation | salt&pepper | score |
|---|---|---|---|---|---|
| bars | 0.768015 | 100 | ReL | 0.0 | 0.951809 |
| corners | 0.001920 | 100 | ReL | 0.0 | 0.853866 |
| multi_mnist | 0.011362 | 1000 | ReL | 0.6 | 0.651657 |
| mnist_shape | 0.031685 | 250 | sigmoid | 0.6 | 0.545559 |
| shapes | 0.083147 | 500 | tanh | 0.4 | 0.928792 |
| simple_superpos | 0.366627 | 100 | ReL | 0.1 | 0.890472 |

Table 1: Configuration of the best network for each dataset as found by the random search.

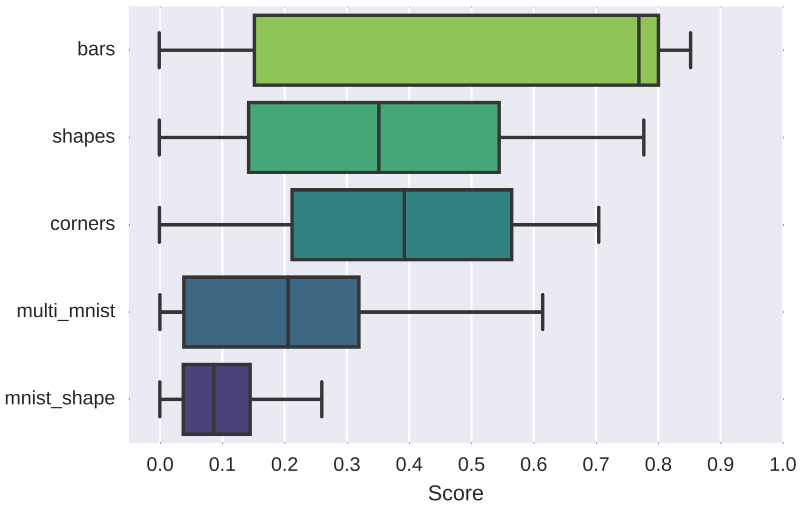


Figure 9: Summary of the scores achieved during the random search for training with multiple objects

| Dataset | learning rate | # hidden units | activation | salt&pepper | score |
|---|---|---|---|---|---|
| bars | 0.012192 | 100 | sigmoid | 0.8 | 0.851777 |
| corners | 0.026035 | 100 | ReL | 0.7 | 0.704285 |
| mnist_shape | 0.033200 | 1000 | ReL | 0.6 | 0.259646 |
| multi_mnist | 0.001786 | 250 | sigmoid | 0.9 | 0.614277 |
| shapes | 0.049402 | 100 | sigmoid | 0.9 | 0.776656 |

Table 2: Configuration of the best network trained on *multiple objects* for each dataset as found by the random search.

# C Multi Object Training

When training the DAEs on images with multiple objects, it is less obvious why running RC should lead to a segregation of the objects. It seems that the autoencoder should always try to reconstruct the whole image including all the objects. And if we run normal (soft) RC we in fact see that after a few iterations each pixel is equally represented by each cluster.

By switching to hard cluster assignments we eliminate this stable state, and force the clusters to compete more for the pixels. Together with the fact that in our datasets objects don't carry any information about other objects this leads to a stronger amplification of the initial differences in reconstruction quality. In Figure 10 this process can be seen on the shapes dataset. Note that the hard RC converges even faster, but generally leads to worse performance.

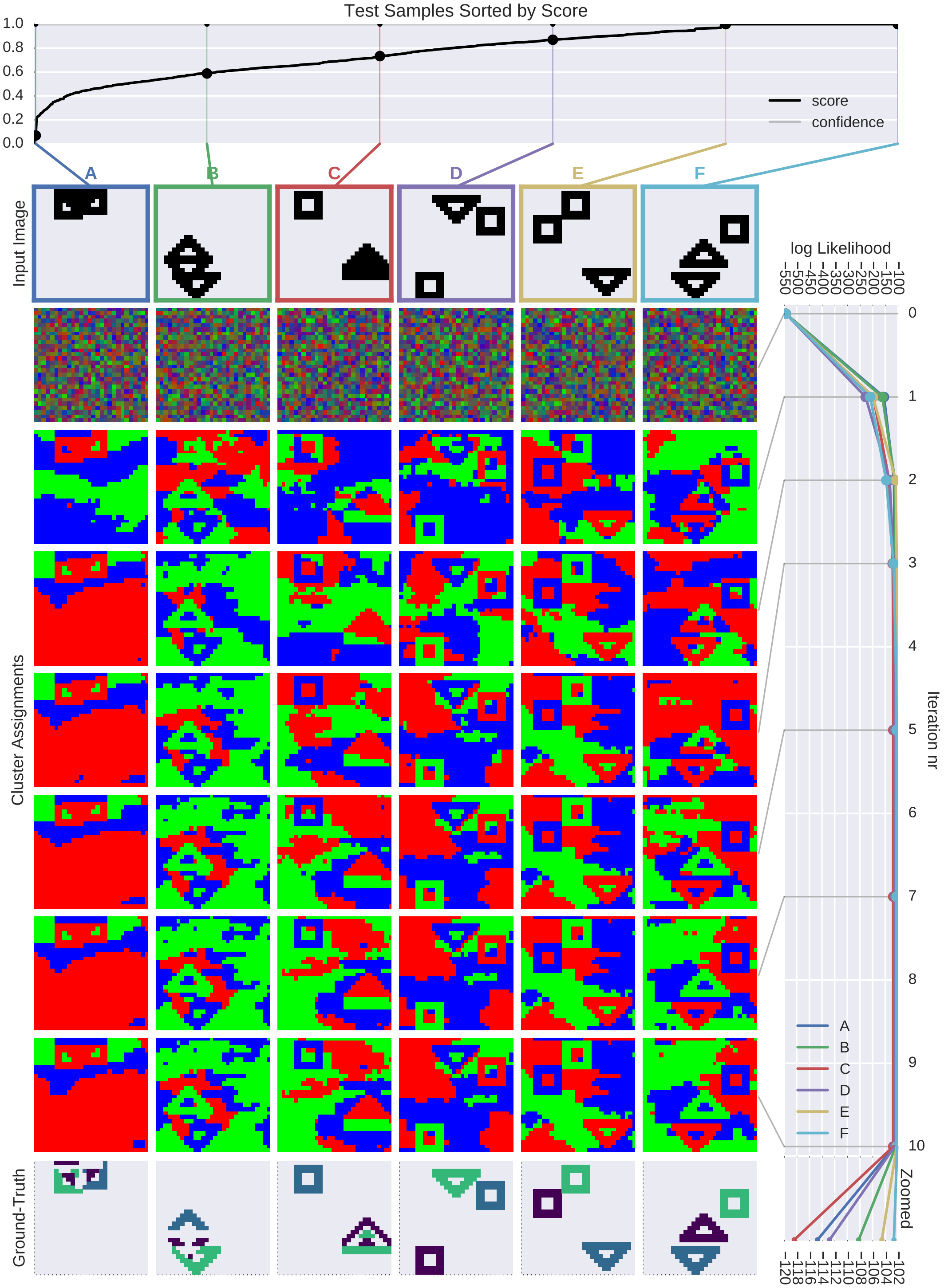


Figure 10: Example iterations of RC when using hard assignments and a DAE that has been trained only on images with multiple objects.

# D ADDITIONAL FIGURES

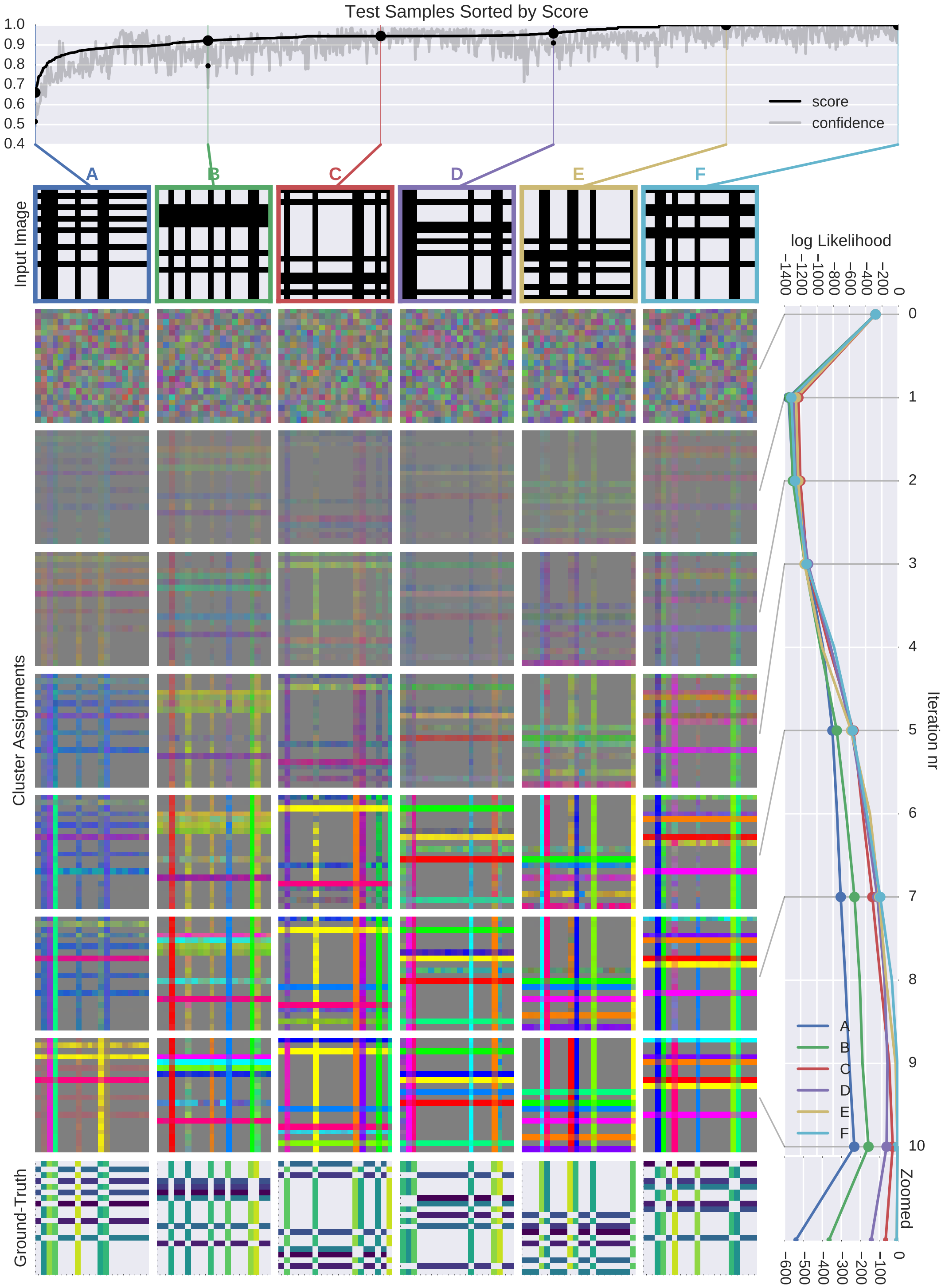


Figure 11

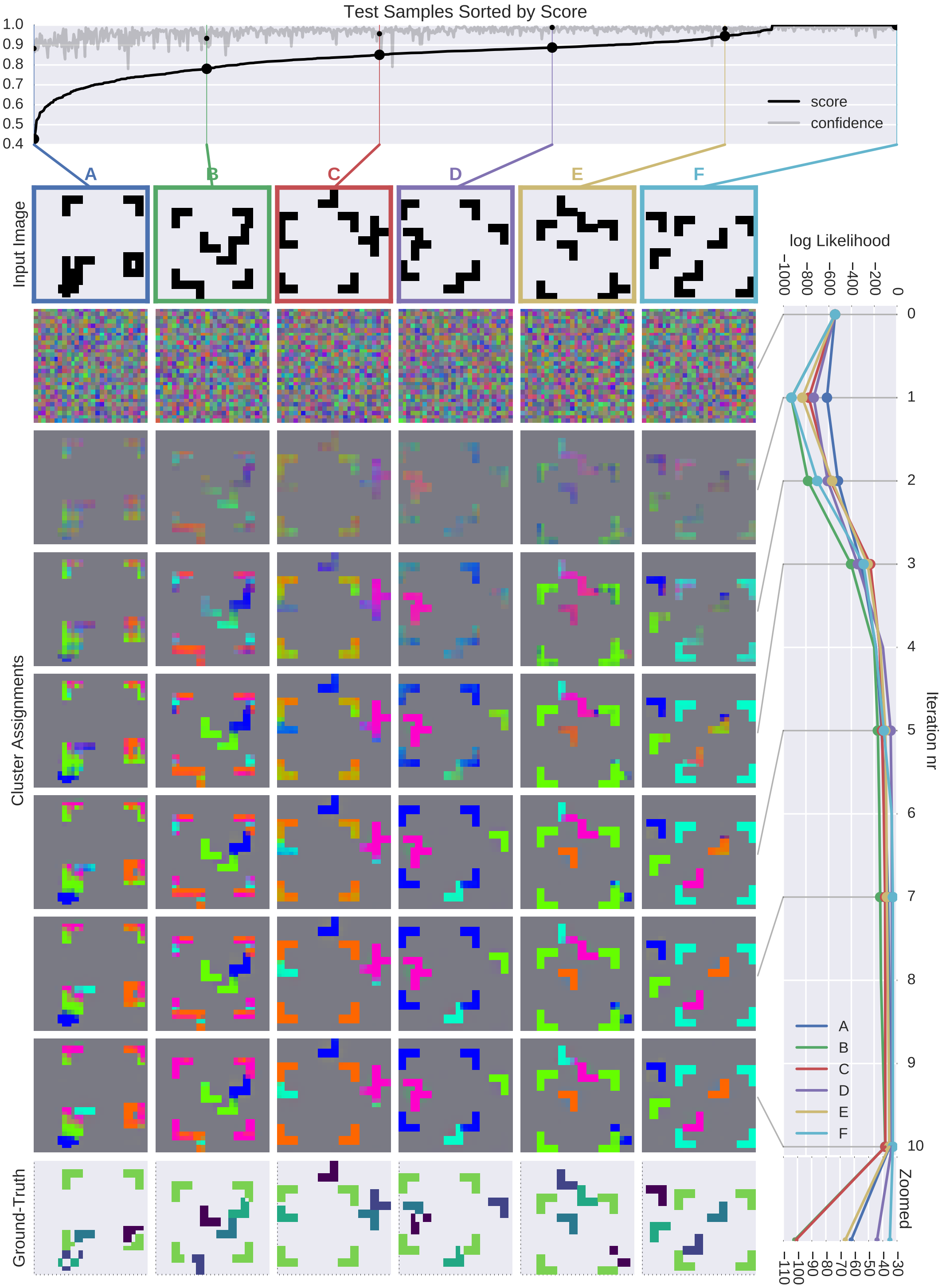

Test Samples Sorted by Score
1.0
0.9
0.8
0.7
0.6
0.5
0.4
score
confidence
A
B
C
D
E
F
Input Image
Cluster Assignments
Ground-Truth
log Likelihood
−1000
−800
−600
−400
−200
0
0
1
2
3
4
5
6
7
8
9
10
Iteration nr
Zoomed
−110
−100
−90
−80
−70
−60
−50
−40
−30

Figure 12

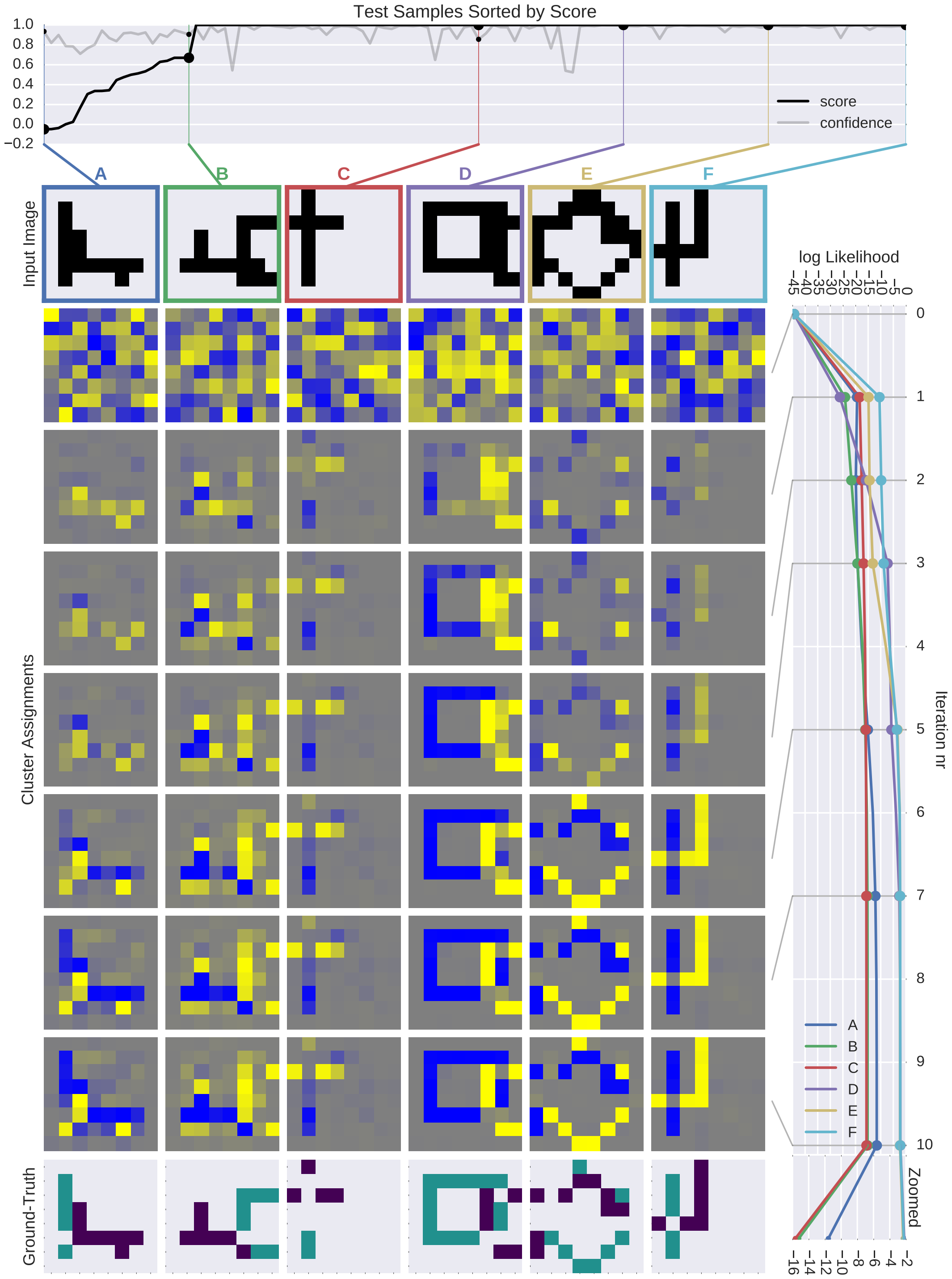

Test Samples Sorted by Score
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
score
confidence
A
B
C
D
E
F
Input Image
Cluster Assignments
Ground-Truth
log Likelihood
0
−5
−10
−15
−20
−25
−30
−35
−40
−45
Iteration nr
0
1
2
3
4
5
6
7
8
9
10
Zoomed
−2
−4
−6
−8
−10
−12
−14
−16

Figure 13

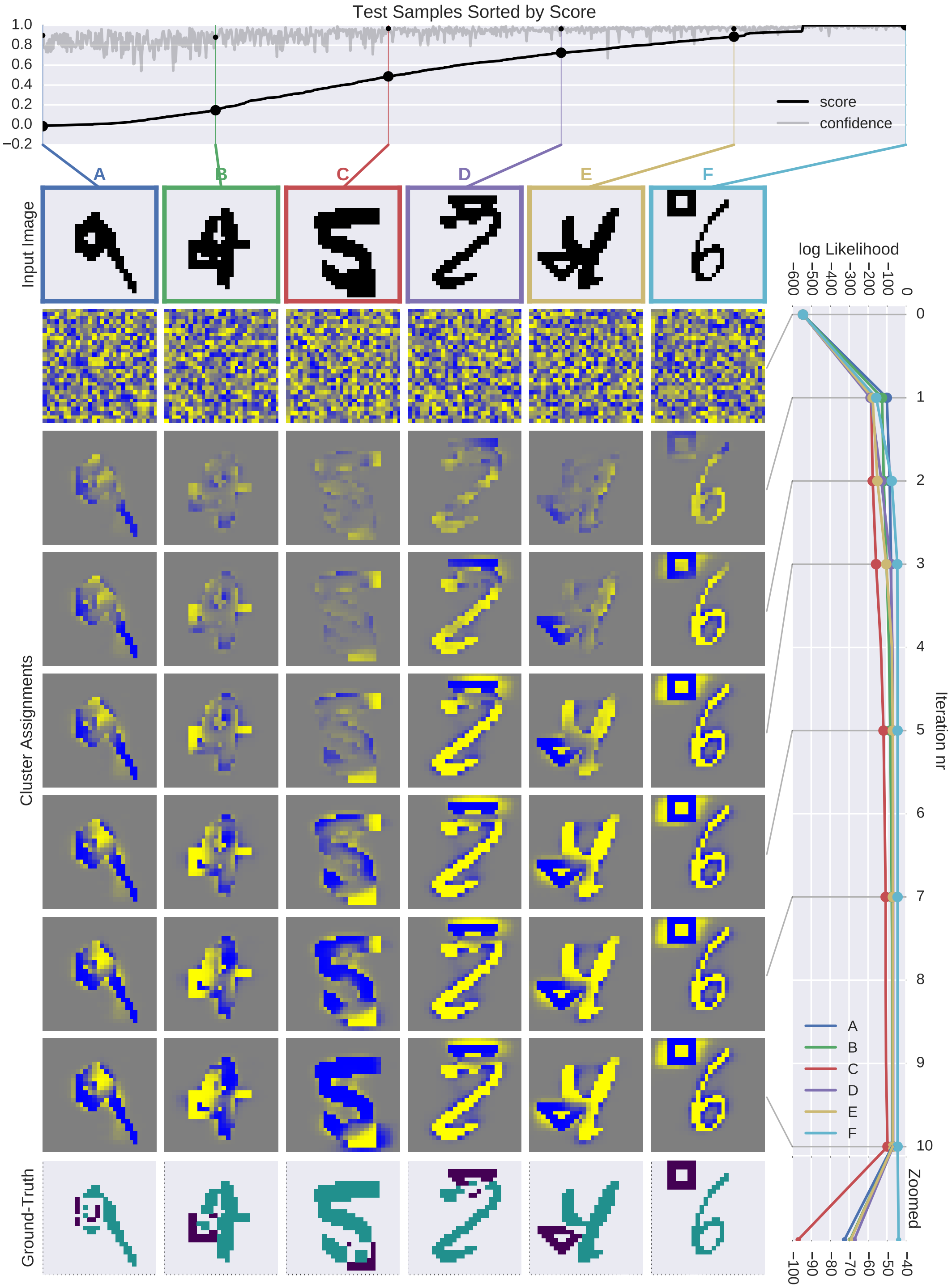

Test Samples Sorted by Score
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
score
confidence
A
B
C
D
E
F
Input Image
Cluster Assignments
Ground-Truth
log Likelihood
−600
−500
−400
−300
−200
−100
0
Iteration nr
0
1
2
3
4
5
6
7
8
9
10
Zoomed
−100
−90
−80
−70
−60
−50
−40

Figure 14

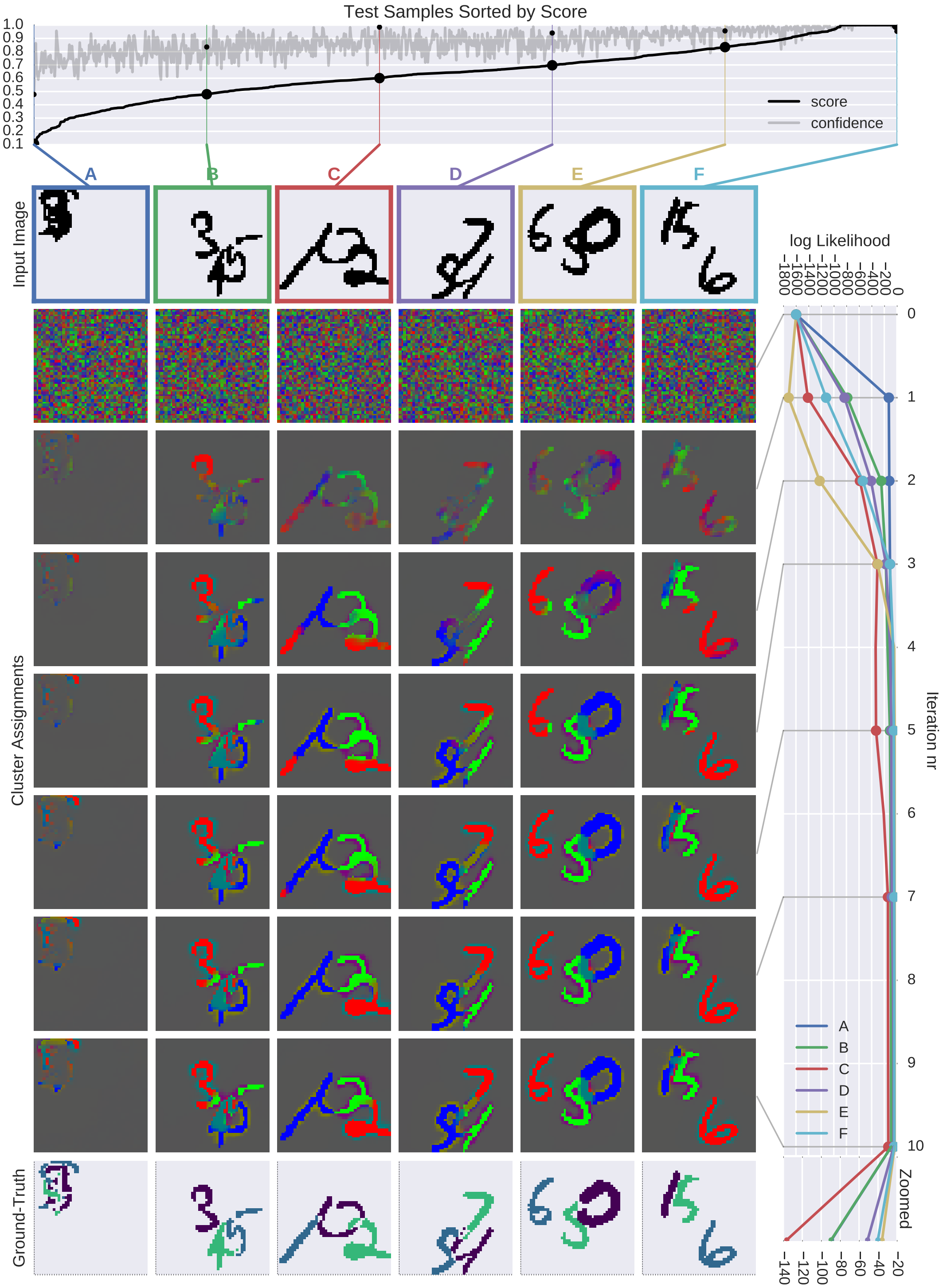

Test Samples Sorted by Score
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
score
confidence
A
B
C
D
E
F
Input Image
Cluster Assignments
Ground-Truth
log Likelihood
0
−200
−400
−600
−800
−1000
−1200
−1400
−1600
−1800
Iteration nr
0
1
2
3
4
5
6
7
8
9
10
A
B
C
D
E
F
Zoomed
−20
−40
−60
−80
−100
−120
−140

Figure 15