Capsule Networks as Recurrent Models of Grouping and Segmentation

Adrien Doerig[a,†,*], Lynn Schmittwilken[a,†], Bilge Sayim[b,c], Mauro Manassi[d] & Michael H. Herzog[a]

[a] Laboratory of Psychophysics, Brain Mind Institute, EPFL, Lausanne, 1015, Switzerland

[b] Institute of Psychology, University of Bern, 3012 Bern, Switzerland

[c] Univ. Lille, CNRS, UMR 9193- SCALab- Sciences Cognitives et Sciences Affectives, F-59000 Lille, France

[d] Department of Psychology, University of Aberdeen, Aberdeen, Scotland, UK

† Equal contributions

* Corresponding author: adrien.doerig@gmail.com

Keywords: Vision, Neural Networks, Capsule Networks, Crowding, Global Shape Processing, Recurrent Processing

Abstract

Classically, visual processing is described as a cascade of local feedforward computations. Feedforward Convolutional Neural Networks (ffCNNs) have shown how powerful such models can be. Previously, using visual crowding as a well-controlled challenge, we showed that no classic model of vision, including ffCNNs, can explain human global shape processing (1). Here, we show that Capsule Neural Networks (CapsNets; 2), combining ffCNNs with a grouping and segmentation mechanism, solve this challenge. We also show that ffCNNs and standard recurrent networks do not, suggesting that the grouping and segmentation capabilities of CapsNets are crucial. Furthermore, we provide psychophysical evidence that grouping and segmentation is implemented recurrently in humans, and show that CapsNets reproduce these results well. We discuss why recurrence seems needed to implement grouping and segmentation efficiently. Together, we provide mutually reinforcing psychophysical and computational evidence that a recurrent grouping and segmentation process is essential to understand the visual system and create better models that harness global shape computations.

Author Summary

Feedforward Convolutional Neural Networks (ffCNNs) have revolutionized computer vision and are deeply transforming neuroscience. However, ffCNNs only roughly mimic human vision. There is a rapidly expanding literature investigating differences between humans and ffCNNs. Several findings suggest that, unlike humans, ffCNNs rely mostly on local visual features. Furthermore, ffCNNs lack recurrent connections, which abound in the brain. Here, we use visual crowding, a well-known psychophysical phenomenon, to investigate recurrent computations in global shape processing. Previously, we showed that no model based on the classic feedforward framework of vision, including ffCNNs, can explain global effects in crowding. Here, we show that Capsule Networks (CapsNets), combining ffCNNs with recurrent grouping and segmentation, solve this challenge. Lateral and top-down recurrent connections do not, suggesting that grouping and segmentation are crucial for human-like global computations. Based on these results, we hypothesize that one computational function of recurrence is to efficiently implement grouping and segmentation. We provide psychophysical evidence that, indeed, recurrent processes implement grouping and segmentation in humans. CapsNets reproduce these results too. Together, we provide mutually reinforcing computational and psychophysical evidence that a recurrent grouping and segmentation process is essential to understand the visual system and create better models that harness global shape computations.

Introduction

The visual system is often seen as a hierarchy of local feedforward computations (3), going back to the seminal work of Hubel and Wiesel (4). Low-level neurons detect basic features, such as edges. Higher-level neurons pool the outputs from the lower-level neurons to detect higher-level features such as corners, shapes, and ultimately objects. Feedforward Convolutional Neural Networks (ffCNNs) embody this classic framework of vision and have shown how powerful it can be (e.g., 5–8). However, despite their amazing success, ffCNNs only roughly mimic human vision. For example, they lack the abundant recurrent processing of humans (9, 10), perform differently than humans in crucial psychophysical tasks (1, 11), and can be easily misled (12–14). An important point of discussion concerns global visual processing. It was suggested that ffCNNs may focus mainly on local, texture-like features, while humans harness global shape level computations (1, 14–18; but see 19). For example, it was shown that changing local features, such as the texture or the edges of an object, can

lead ffCNNs to misclassify it (14, 15). Humans, in contrast, can still easily classify the object based on its global shape.

There are no widely accepted diagnostic tools to specifically characterize global computations in neural networks. Models are usually compared either on computer vision benchmarks, such as ImageNet (20) or with neural responses in the visual system (21, 22). One drawback with these approaches is that the datasets are hard to control. Psychophysical results can be used to fill this gap and create well-controlled challenges for visual models, tailored to target specific aspects of vision (23). Here, we use visual crowding to specifically target global shape computations in humans and machines.

In crowding, objects that are easy to identify in isolation appear as jumbled and indistinct when clutter is added (1, 24–29). For example, a vernier target is presented, i.e., two vertical lines separated by a horizontal offset (Figure 1a). When the vernier is presented alone, observers easily discriminate the offset direction. When a flanking square surrounds the target, performance drops, i.e., there is strong crowding (30, 31). Surprisingly, *adding* more flanking squares *reduces* crowding strongly, depending on the configuration (Figure 1b; 29). Hence, the *global* configuration of visual elements across large regions of the visual field influences perception of the small vernier target. This global *un*crowding effect occurs for a wide range of stimuli in vision, including foveal and peripheral vision, audition, and haptics (32–38). The ubiquity of (un)crowding in perception is not surprising since elements are rarely seen in isolation. Hence, any perceptual system needs to cope with crowding, i.e., isolating important information from clutter.

We have shown previously that these global effects of crowding *cannot* be explained by models based on the classic framework of vision, including ffCNNs (1, 18, 39). Here, we propose a new framework to understand these global computations. We show that Capsule Neural Networks (CapsNets; 2), augmenting ffCNNs with a recurrent grouping and segmentation process, can explain these complex global (un)crowding results in a natural manner. Two processing regimes can occur in CapsNets: a fast feedforward pass able to quickly process information, and a time-consuming recurrent regime to compute in-depth global grouping and segmentation. We will show that the human visual system indeed harnesses recurrent processing for efficient grouping and segmentation, and that CapsNets naturally explain these results. Together, our results suggest that a time-consuming recurrent grouping and segmentation process is crucial for global shape-level computations in both humans and artificial neural networks.

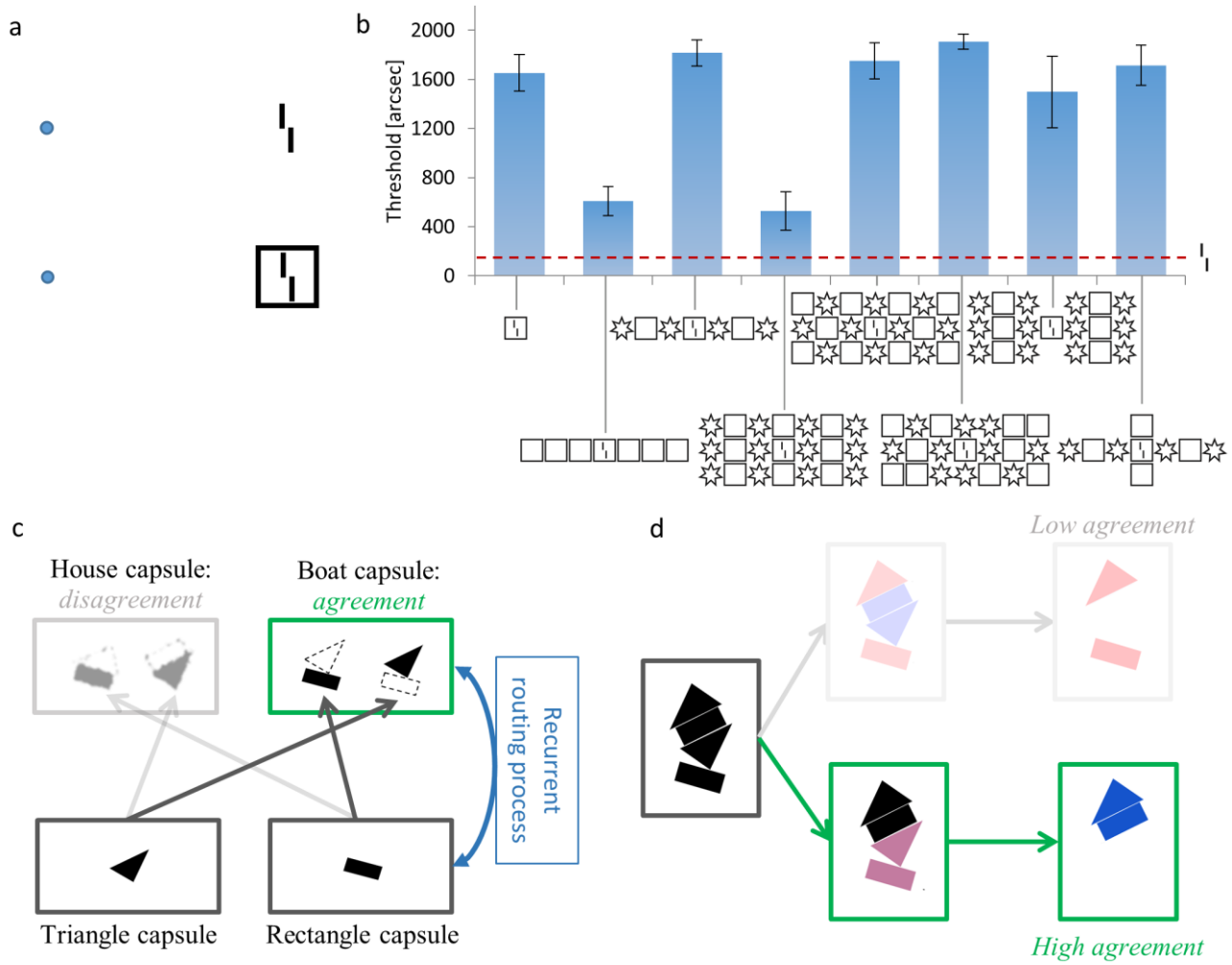


Figure 1: a. Crowding: Perception of visual elements deteriorates in clutter, an effect called crowding. In this example, a vernier (two vertical bars with a horizontal offset) becomes harder to perceive when a square flanker is added (fixate on the blue dots). **b. Uncrowding:** A vernier is presented in the visual periphery. The offset direction is easily reported (dashed red line; the y-axis shows the threshold, i.e., the minimal offset size at which observers can report the offset direction with 75% accuracy). When a square flanker surrounds the vernier, performance deteriorates- a classic crowding effect. When more squares are added, performance recovers (uncrowding). Critically, the uncrowding effect depends on the global stimulus configuration. For example, if some squares are replaced by stars, performance deteriorates again (3rd bar; 25). **c. Routing by agreement in CapsNets:** Information propagates between layers of capsules through a recurrent routing process aiming to maximize agreement between capsules. Each capsule is a group of neurons whose activity vector represents the pose (such as position, orientation, etc.) of the feature it detects. In this toy example, lower-level capsules detect simple shapes such as triangles and rectangles. In the next layer, capsules have learnt combinations of these shapes. Here, the triangle capsule detects a tilted triangle and the rectangle capsule detects a tilted rectangle. Each of these capsules predicts what is represented at the next layer. For example, the triangle capsule predicts an upside-down house or a tilted boat, while the rectangle capsule predicts a tilted house or a tilted boat. The recurrent routing by agreement process routes information between the layers so that agreement is maximized. In this case, capsules agree about the titled boat, but disagree about the house orientation. Hence, the routing by agreement suppresses activity in the house capsule and boosts activity in the boat capsule. **d. Grouping and segmentation in CapsNets:** This recurrent routing by

agreement process endows CapsNets with natural grouping and segmentation capabilities. Here, an ambiguous stimulus, which can be seen either as an upside-down house (top) or a house on a boat (bottom), is presented. The upside-down house interpretation leaves parts of the image unexplained and this causes disagreement. Hence, the routing by agreement will select the latter interpretations because it is the best explanation of the input and therefore maximizes agreement. Thereby, the house and boat are each grouped as an object and segmented into the corresponding higher-level capsules.

Results

Experiment 1: Crowding and Uncrowding Naturally Occur in CapsNets

In CapsNets, early convolutional layers extract basic visual features. Recurrent processing combines these features into groups and segments objects by a process called *routing by agreement*[1]. The entire network is trained end-to-end through backpropagation. *Capsules* are groups of neurons representing visual features and are crucial for the routing by agreement process. Low-level capsules iteratively predict the activity of high-level capsules in a recurrent loop. If the predictions agree, the corresponding high-level capsule is activated. For example, if a capsule responds to a triangle above a rectangle detected by another capsule, they agree that the higher-level object should be a house and, therefore, the corresponding high-level capsule is activated (Figure 1c). This process allows CapsNets to group and segment objects (Figure 1d).

We trained CapsNets with two convolutional layers followed by two capsule layers to recognize greyscale images of vernier targets and groups of identical shapes (see Methods). During training, either a vernier or a group of identical shapes was presented. The network had to simultaneously classify the shape type, the number of shapes in the group, and the vernier offset direction. Importantly, verniers and shapes were never presented together during training, i.e., there were no (un)crowding stimuli during training.

When combining verniers and shapes after training, both crowding and uncrowding occurred (Figure 2a): presenting the vernier target within a single flanker deteriorated vernier offset discrimination (crowding), and adding more identical flankers recovered performance (uncrowding). Adding configurations of alternating different flankers did not recover the network's performance, similarly to human vision. Small changes in the network hyperparameters, loss terms or stimulus characteristics do not affect these results (supplementary material). As a control condition, we checked that when the

[1] In most implementations of CapsNets, including ours and (2), the iterative routing by agreement process is not explicitly implemented as a "standard" recurrent neural network processing sequences of inputs online. Instead, there is an iterative algorithmic loop (see (2) for the algorithm), which is equivalent to recurrent processing.

vernier target is presented outside the flanker configuration, rather than inside, there was no performance drop (supplementary material). Hence, the performance drop in crowded conditions was not merely to the simultaneous presence of the target and flanking shape in the stimulus.

Reconstructing the input image based on the network's output (see Methods) shows that (un)crowding occurs through grouping and segmentation (figure 2b). Crowding occurs when the target and flankers cannot be segmented and are therefore routed to the same capsule. In this case, they interfere because a single capsule cannot represent well two objects simultaneously due to limited neural resources. This mechanism is similar to pooling: information about the target is pooled with information about the flankers, leading to poorer representations. However, if the flankers are segmented away and represented in a different capsule, the target is released from the flankers' deleterious effects and *un*crowding occurs (Figure 2c). This segmentation can only happen if the network has learnt to group the flankers into a single higher-level object represented in a different capsule than the vernier target. Segmentation is facilitated when more flankers are added because more low-level capsules agree about the presence of the flanker group.

Alternating configurations of different flankers, as in the third configuration of Figure 1b, usually do not lead to uncrowding (29). In some rare cases, the network produced uncrowding with such configurations (stimuli h, u ,v & J; Figure 2). Reconstructions show that in these cases the network simply could not differentiate between different shapes of the flankers (e.g. between circles and hexagons), which therefore formed a group for the network and were segmented away from the target (Figure 2b). This further reinforces the notion that grouping and segmentation differentiate crowding from uncrowding: whenever the network reaches the conclusion that flankers form a group, segmentation is facilitated. When this happens, the vernier and flankers are represented in different capsules, leading to good performance.

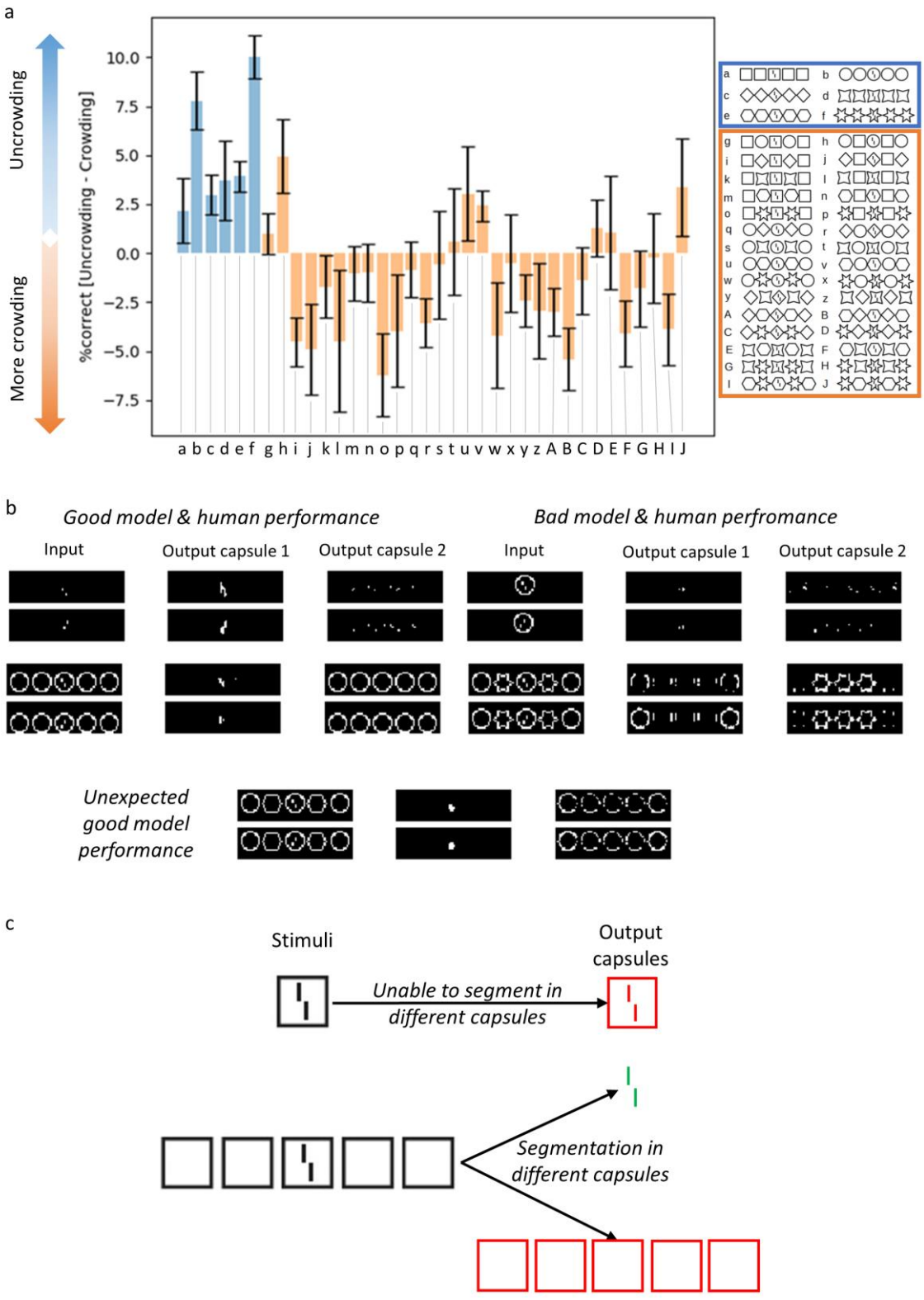


Figure 2: a. CapsNets explain both crowding and uncrowding: The x-axis shows the various stimuli. We used 6 different flanker shape types and tested all configurations with 5 identical or alternating shapes (e.g., 5 squares, 5 circles, circle-square-circle-square-circle, etc; see Methods). Performance is shown on the y-axis as the % correct for each stimulus *minus* the % correct with only the central single flanker. For example, in column *a*, vernier offset direction is easier to read out with 5 square flankers than with 1 square flanker, as expected. Error bars are the standard error over 10 network trainings (we used 10 networks to match the typical number of observers in human experiments; 29, 40). The blue bars

represent configurations for which *un*crowding is expected (blue bars larger than 0.0 are in accordance with the human data) and orange bars represent configurations for which crowding is expected (orange bars smaller than or around 0.0 are in accordance with the human data). **b. Reconstructions:** We reconstructed the input image based on the output capsules' activities (see Methods). The reconstructions based on the two most activated capsules are shown. When the vernier is presented alone (top left), the reconstructions are good. When a single flanker is added (top right), the vernier reconstruction deteriorates (crowding) because the vernier is not well segmented from the flanker. When identical flankers are added (bottom left), the vernier reconstruction recovers, i.e., it is well segmented from the flankers (uncrowding). With different flankers (bottom right), the vernier is not represented at all in the two winning capsules (crowding). Interestingly, when the network produces "unexpected" uncrowding (i.e., the network shows uncrowding contrary to humans; bottom left), the reconstructions strongly resemble the case of "normal" uncrowding (compare middle and bottom left panels). In this case, the network was unable to notice the difference between circles and hexagons, and treated both stimuli in the same way. **c. Segmentation and (un)crowding in CapsNets:** If CapsNets can segment the vernier target away from the flankers during the recurrent routing by agreement process, uncrowding occurs. Segmentation is difficult when a single flanker surrounds the target because capsules disagree about what is shown at this location. In the case of configurations that the network has learned to group, many primary capsules agree about the presence of a group of shapes, which can therefore easily be segmented away from the vernier target.

In previous work, we have shown that pretrained ffCNNs (including an ffCNN biased towards global shape processing; 14) cannot explain uncrowding (18). Currently, CapsNets cannot be trained on large-scale tasks such as ImageNet because routing by agreement is computationally too expensive. Therefore, here, we took a different approach. As explained above, we trained our CapsNets to recognize groups of shapes and verniers and asked how they would generalize from shapes presented in isolation to crowded shapes. To make sure that CapsNets explain global (un)crowding thanks to their grouping and segmentation *architecture* and not merely due to this different *training* regime, we conducted three further experiments. We investigated how performance changes when the capsule layers are replaced by other architectures, keeping the number of neurons constant.

First, we replaced the capsules by a fully connected feedforward layer, yielding a classic ffCNN with three convolutional layers and a fully connected layer. We trained and tested this architecture exactly in the same way as the CapsNets, i.e., with the same stimuli, the same loss function, etc. The results clearly show that there is no uncrowding (Figure 3a): ffCNNs do not reproduce human-like global computations with this procedure.

Second, we added lateral recurrent connections to the fully connected layer of the previous ffCNN, yielding a network with three convolutional layers followed by a fully connected recurrent layer. We used the same number of recurrent iterations as for the routing by agreement in the CapsNets. Again,

we trained and tested this architecture exactly like we trained and tested the CapsNets. There is no uncrowding with this architecture either (Figure 3b).

Lastly, we added top-down connections feeding back from the final fully connected layer of the previous ffCNN to the layer below, yielding a network with three convolutional layers followed by a fully connected layer that fed back into the layer below (again with the same number of recurrent iterations as iterations of routing by agreement in the CapsNets). Again, after training and testing this architecture in the same way as the other networks, we found no uncrowding (Figure 3c). The absence of uncrowding in feedforward ffCNNs and ffCNNs with added lateral or top-down connections suggests that the *architecture* of CapsNets, and not our training regime, explains why (un)crowding is reproduced. Furthermore, recurrence by itself is not sufficient to produce (un)crowding. The grouping and segmentation performed by routing by agreement seems crucial.

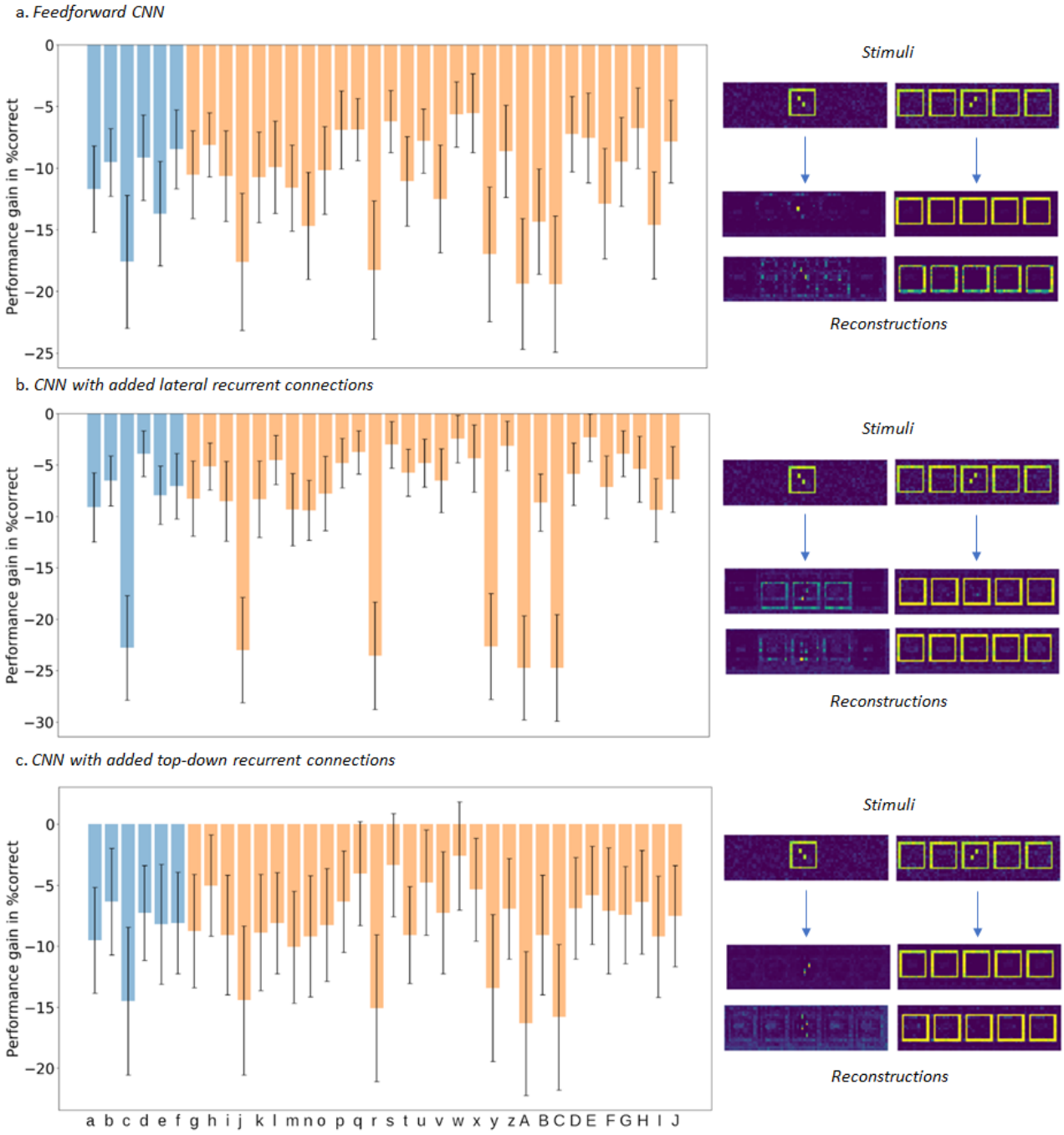


Figure 3: Other network architectures do not explain uncrowding. To verify that the ability of CapsNets to explain uncrowding is due to their architecture and not merely to the way they are trained, we replaced the recurrent routing by agreement processing by three different alternative architectures: a feedforward fully connected layer (yielding a classic ffCNN, a), a fully connected layer with lateral recurrent connections (b) and a fully connected layer with top-down recurrent connections to the layer below (c). The plots on the left show the model's performance in the same way as figure 2a (the x-axes represent (un)crowding stimuli, positive values on the y-axes show uncrowding). None of these architectures can produce uncrowding (compare with the CapsNet results in figure 2a). On the right, reconstructions are shown. For all of these networks, the vernier can be reconstructed with a single flanker but not when there are five flankers, showing that adding further flanker increases crowding, in contrast to humans where adding flankers rescues perception of the vernier (uncrowding).

Experiment 2: The role of recurrent processing

As mentioned, processing in CapsNets starts with a feedforward sweep followed by recurrent routing by agreement to refine grouping and segmentation. We hypothesize that humans may use recurrent processing to efficiently implement grouping and segmentation. To test this hypothesis, we psychophysically investigated the temporal dynamics of (un)crowding. We show that uncrowding is mediated by a time-consuming *recurrent* process in humans. When the target groups with the flankers, crowding occurs immediately. In contrast, when the target and flankers form separate groups, time-consuming recurrent computations are required to segment the flanker from the target. We successfully model these results with CapsNets.

First, we performed a psychophysical crowding experiment with a vernier target flanked by either two lines or two cuboids (see Methods; Figure 4). The stimuli were displayed for varying durations from 20 to 640ms and five observers reported the vernier offset direction. For short stimulus durations, crowding occurred for both flanker types, i.e., thresholds increased for both the lines and cuboids conditions compared to the vernier alone condition (lines: p = 0.0017, cuboids: p = 0.0013, 2-tailed one-sample t-tests).

We quantified how performance changed with increasing stimulus duration by fitting a line $y = ax + b$ to the data for each subject, and comparing the mean slope a across subjects with 0 in one-sample 2-tailed t-tests. The performance on the lines condition did not significantly change with increasing stimulus duration (p = 0.057). These results are in accordance with previous results which show that crowding varies very little with stimulus duration (41; but see 42, 43). With the flanking cuboids we found a different pattern of results: performance dramatically improves with stimulus duration (p = 0.0007). This improvement cannot be explained by local mechanisms, such as lateral inhibition (30, 44) or pooling (45–47) since the inner flanking vertical lines are the same in the lines and cuboids. Hence, according to a local approach we should expect no difference in thresholds between the two flanking conditions.

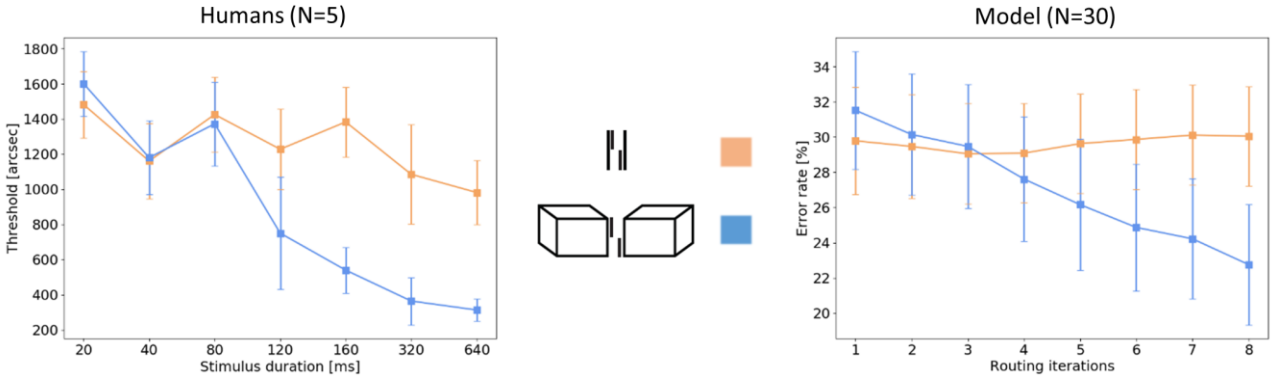


Figure 4: Temporal dynamics of uncrowding: *Left: Human data*. For cuboid flankers, strong crowding occurs up to 100ms of stimulus presentation, and then uncrowding gradually occurs for longer durations (i.e., performance improves; blue). The x-axis shows different stimulus durations and the y-axis shows the corresponding threhsolds (i.e., lower values indicate better performance). Error bars indicate standard error. Uncrowding does not occur with single line flankers, even for long stimulus durations (orange). We hypothesize that the cuboids are segmented from the vernier target through time-consuming recurrent processing (the line flankers are grouped with the target and cannot be segmented at all). *Right: Model data.* CapsNets can explain these results by varying the number of recurrent routing by agreement iterations. The x-axis shows different numbers of routing iterations during testing and the y-axis shows the corresponding error rates (i.e., lower values indicate better performance). Error bars indicate standard deviation across 30 trained networks (see Methods). Similarly to humans, both lines and cuboids lead to crowding with few routing by agreement iterations. Performance increases with routing iterations only for the cuboids. This suggests that recurrent processing helps to compute and segment the complex cuboids, but the lines are immediately strongly grouped with the vernier and can never be segmented. Hence, they do not benefit from the recurrent segmentation process.

Crucially, uncrowding occurred for the cuboid flankers only when stimulus durations were sufficiently long (Figure 4). In contrast, the effect of the line flankers does not change over time. We propose that these results reflect the time-consuming recurrent computations needed to segment the cuboid flankers away from the target. Performance does not improve with the line flankers, because they are too strongly grouped with the vernier target, so recurrent processing cannot segment them away.

We trained CapsNets with the same architecture as in experiment 1 to discriminate vernier offsets, and to recognize lines, cuboids and scrambled cuboids (see Methods; the scrambled cuboids were included only to prevent the network from classifying lines vs. cuboids simply based on the number of pixels in the image). As in experiment 1, during training, each training sample contained one of the shape types, and the network had to classify which shape type was present and to discriminate the vernier offset direction. We used 8 routing by agreement iterations during training. As in experiment

1, verniers and flankers were never presented together during training (i.e., there were no (un)crowding stimuli).

After training, we tested the networks on (un)crowding stimuli, changing the number recurrent routing by agreement iterations from one (leading to a purely feedforward regime) to 8 iterations (a highly recurrent regime; Figure 3). We found that CapsNets naturally explain the human results. Using the same statistical analysis as for humans, we found that with more iterations, the cuboids are better segmented from the target, and performance improves ($p = 0.003$). On the other hand, the effect of the line flankers does not change over time ($p = 0.64$). These results were not affected by small changes in network hyperparameters or loss terms (supplementary material). We did not compare these results with the ffCNN and recurrent networks used in experiment 1, because these networks produced no uncrowding at all.

These findings are explained by the recurrent routing by agreement process. With cuboids, capsules across an extended spatial region need to agree about the presence of a cuboid, which is then segmented into its own capsule. This complex process requires several recurrent iterations of the routing by agreement process. On the other hand, the lines are immediately strongly grouped with the vernier, so further iterations of routing by agreement do not achieve successful segmentation and, hence, cannot improve performance.

Discussion

Our results provide strong evidence that time-consuming recurrent grouping and segmentation is crucial for shape-level computations in both humans and artificial neural networks. We used (un)crowding as a psychophysical probe to investigate how the brain flexibly forms object representations. These results specifically target global, shape-level and time-consuming recurrent computations and constitute a well-controlled and difficult challenge for neural networks.

It is well known that humans can solve a number of visual tasks very quickly, presumably in a single feedforward pass of neural activity (48). ffCNNs are good models of this kind of visual processing (21, 22, 49). However, many studies have shown that neural activities are not determined by the feedforward sweep alone, and recurrent activity affords a distinct processing regime to perform more in-depth time-consuming computations (9, 10, 50–53). Similarly, CapsNets naturally include both a fast feedforward and a time-consuming recurrent regime. When a single routing by agreement iteration is used, CapsNets are rapid feedforward networks that can accomplish many tasks, such as

vernier discrimination or recognizing simple shape types (e.g. circles vs. squares). With more routing iterations, a recurrent processing regime arises and complex global shape effects emerge, such as segmenting the cuboids in experiment 2. We showed how the transition from feedforward to recurrent processing in CapsNets explains psychophysical results about temporal dynamics of (un)crowding.

Recurrent activity offers several advantages. First, although feedforward networks can in principle implement any function (54), recurrent networks can implement certain functions more efficiently. Flexible grouping and segmentation is exactly the kind of function that may benefit from recurrent computations (see also Seijdel et al., under review). For example, to determine which local elements should be grouped into a global object, it helps to compute this global object first. This information can then be fed back to influence how each local element is processed. For example, to model (un)crowding, it helps to compute the global configuration of flankers first to determine how to process the vernier. Should it be grouped with the flankers (crowding) or not (uncrowding)? In CapsNets, the first feedforward sweep of activity provides an initial guess about which global objects are present (e.g., large cuboids). At this stage, as shown in experiment 2, information about the vernier interferes with information about the cuboids (crowding). Then, recurrent processing routes information relative to cuboids and the vernier to different capsules (uncrowding). Without recurrence, in contrast, it is difficult to rescue the vernier information once it has been crowded.

Second, although any network architecture can implement any computation in principle (given enough neurons), they differ in the way they *generalize* to previously unseen stimuli. Hence, recurrent grouping and segmentation architectures influence what is learned from training data. Here, we have shown that only CapsNets, but not ffCNN or ffCNNs augmented with recurrent lateral or top-down connections, produce uncrowding when trained identically to recognize groups of shapes and verniers. In general, ffCNNs tend to generalize poorly (review: 55). Using different architectures to improve how current systems generalize is a promising avenue of research. In this respect, we have shown that CapsNets generalize more similarly to humans than ffCNNs and standard recurrent networks in the context of global (un)crowding.

One limitation in our experiments is that we explicitly taught the CapsNets which configurations to group together by selecting which groups of shapes were present during training (e.g., only groups of identical shapes in experiment 1). Effectively, this gave the network adequate priors to produce un-crowding with the appropriate configurations (i.e., only identical, but not different flankers). Hence, our results show that, given adequate priors, CapsNets explain uncrowding. We have shown that

ffCNNs and lateral or top-down recurrent connections do *not* produce uncrowding, *even* when they are trained identically on groups of identical shapes and showed learning on the training data comparable to the CapsNets (furthermore, we showed previously that pretrained ffCNNs who are often used as general models of vision do not show uncrowding either; 18). This shows that merely training networks on groups of identical shapes is not sufficient to explain uncrowding. It is the recurrent segmentation in CapsNets that is crucial. Humans do not start from zero and therefore do not need to be trained in order to perform crowding tasks. The human brain is shaped through evolution and learning to group elements in a useful way to solve the tasks it faces. As mentioned, (un)crowding can be seen as a probe into this grouping strategy. Hence, we expect that training CapsNets on more naturalistic tasks such as ImageNet may lead to grouping strategies similar to humans and may therefore naturally equip the networks with priors that explain (un)crowding results. At the moment, however, CapsNets have not been trained on such difficult tasks because the routing by agreement algorithm is computationally too expensive.

Recurrent networks are harder to train than feedforward systems, which explains the dominance of the latter during these early days of deep learning. However, despite this hurdle, recurrent networks are emerging to address the limitations of ffCNNs as models of the visual system (10, 50, 52, 53, 56, 57). Although there is consensus that recurrence is important for brain computations, it is currently unclear which functions exactly are implemented recurrently, and how they are implemented. Our results suggest that one important role of recurrence is shape-level computations through grouping and segmentation. We had previously suggested another recurrent segmentation network, hardwired to explain uncrowding (58). However, CapsNets, bringing together recurrent grouping and segmentation with the power of deep learning, are much more flexible and can be trained to solve any task. Linsley et al. (53) proposed another recurrent deep neural network for grouping and segmentation, and there are other possibilities too (59, 60). We do not suggest that CapsNets are the only implementation of grouping and segmentation. We only suggest that grouping and segmentation is important and further work is needed to show how the brain implements it.

In conclusion, our results provide mutually reinforcing modelling and psychophysical evidence that time-consuming, recurrent grouping and segmentation plays a crucial role for global shape computations in humans and machines.

Methods

The code to reproduce all our results will be available with the journal version of this contribution.

All models were implemented in Python 3.6, using the high-level estimator API of Tensorflow 1.10.0. Computations were run on a GPU (NVIDIA GeForce GTX 1070). We used the same basic network architecture in all experiments (Figure 5a). We implemented early feature extraction by using three convolutional layers without padding, each followed by an ELU non-linearity. We used dropout (61) after the first and second convolutional layers. The outputs of the last convolution were reshaped into *m* primary capsule types outputting *n*-dimensional activation vectors. The number of output capsule types was equal to the number of different shapes used as input. The network was trained end-to-end through backpropagation. For training, we used an Adam optimizer with a batch size of 48 and a learning rate of 0.0004. To this learning rate, we applied cosine decays with warm restarts (62).

This choice of network architecture was motivated by the following rationale (Figure 5b). After training, ideally, primary capsules detect the individual shapes present in the input image, and output capsules group and segment these shapes through recurrent routing by agreement. The network can only group shapes together if it was taught during training that these shapes should form a group. To match this rationale, we set the primary capsules' receptive field sizes to roughly the size of one shape, and we set the number of output capsules equal to the number of shape types.

Inputs were grayscale images (Figure 5c&d). We added random Gaussian noise with mean $\mu = 0$ and standard deviation randomly drawn from a uniform distribution $\sigma \sim \mathcal{U}(0.00,0.02)$. The contrast was varied either by first adding a random value between -0.1 and 0.1 to all pixel values and then multiplying them with a random value drawn from a uniform distribution $\mathcal{U}(0.6, 1.2)$, or vice versa. The pixel values were then clipped between 0 and 1.

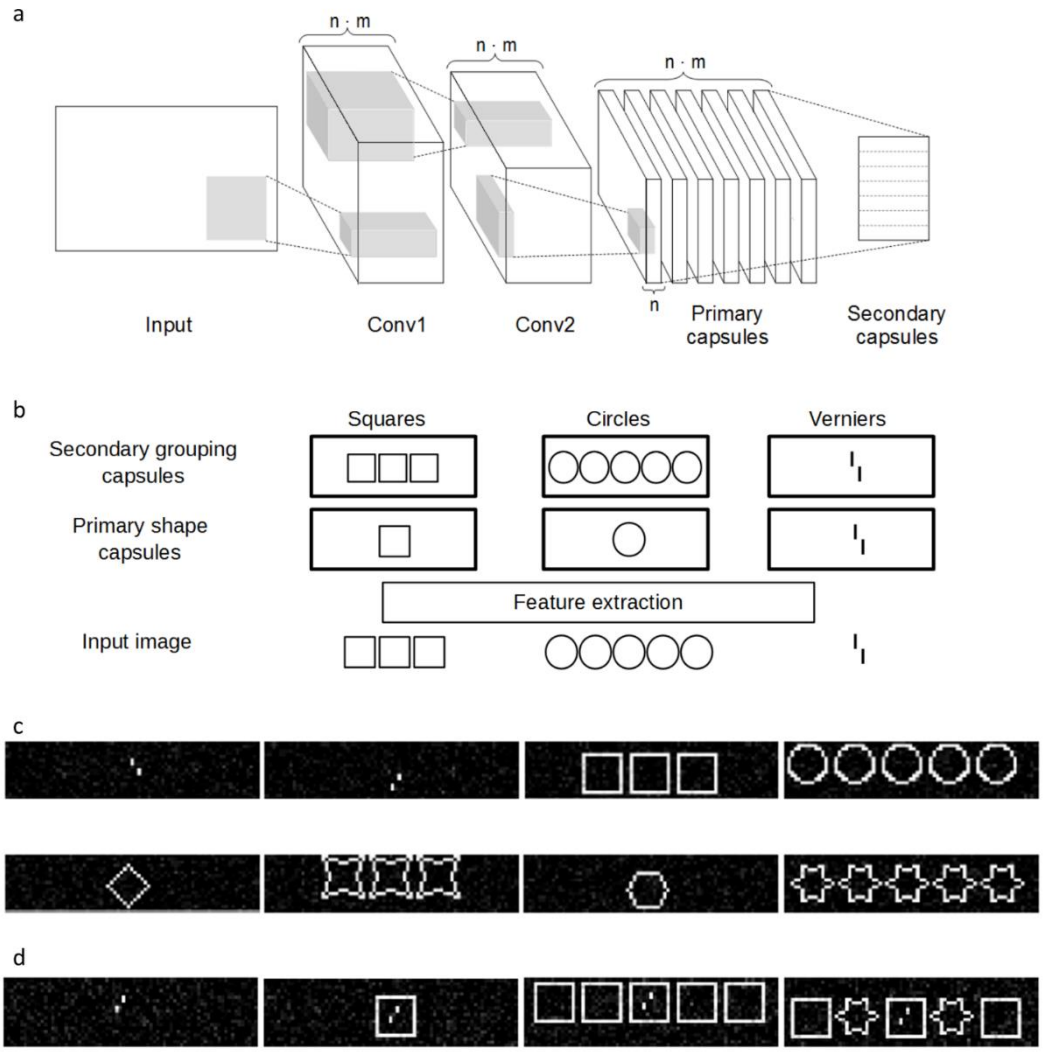


Figure 5: a. Network architecture: We used capsule networks with three convolutional layers whose last outputs was reshaped into the primary capsule layer with m primary capsule types and n primary capsule dimensions. In this example, the number of primary and output capsules types is seven to match the seven shape types we used in experiment 1 (see caption c), but the number depended on the experiment. The primary and output capsule layers communicate via routing-by-agreement. **b. Ideal representations:** After training, the primary capsules detect single shapes of different types at different locations. In this example, there are squares, circles and verniers. By routing the outputs of the primary capsules to the corresponding output capsules, the output capsules group these shapes in groups of one, three or five, based on the number of shapes detected by the primary capsules. If the left stimulus with three squares is presented, the primary square capsules detect squares at three different locations. Through routing by agreement, the output squares capsule groups these three squares. If the middle stimulus with five circles is presented, the primary circle capsules detect circles at five different locations. Through routing by agreement, the output circles capsule represents a group of five circles after routing. Lastly, if a vernier is presented (right stimulus), it is detected by primary capsules and is represented in the vernier output capsule. **c. Training stimuli for experiment 1:** All shapes were shown randomly in groups of one, three or five, except verniers who were always presented alone. **d. Testing stimuli for experiment 1:** Example stimuli for the four test conditions: In the vernier-alone condition (*left*), we expected the network to perform well on the vernier discrimination task. In crowding conditions (*middle-left*), we expected a deterioration of the vernier discrimination as in classical crowding. In uncrowding conditions with many identical flankers (*middle-right*), we expected a recovery of the vernier discrimination.

In no-uncrowding conditions with different flanker types (*right*), we expected crowding. After training, the network has learnt about groups of identical shapes and verniers, but has never encountered these (un)crowding stimuli.

Experiment 1:

Modelling

Human data for experiment 1 is based on (26). We trained CapsNets with the above architecture to solve a vernier offset discrimination task and classify groups of identical shapes. The training dataset included vernier stimuli and six different shape types (Figure 4c). Shapes were presented in groups of one, three or five shapes of the same type. The group was centered in the middle of the image, with a jitter of 2 pixels along the x-axis and 6 pixels along the y-axis.

The loss function included a term for shape type classification, a term for vernier offset discrimination, a term for the number of shapes in the image, and a term for reconstructing the input based on the network output (see equations 1-5). Each loss term was scaled so that none of the terms dominated the others. For the shape type classification loss, we implemented the same margin loss as in (2). This loss enables the detection of multiple objects in the same image. For the vernier offset loss, we used a small decoder to determine vernier offset directions based on the activity of the vernier output capsule. The decoder was composed of a single dense hidden layer followed by a ReLU-nonlinearity and a dense readout layer of two nodes corresponding to the labels left and right. The vernier offset loss was computed as the softmax cross entropy between the decoder output and the one-hot-encoded vernier offset labels. The loss term for the number of shapes in the image was implemented similarly, but the output layer comprised three nodes representing the labels one, three or five shape repetitions. For the reconstruction loss, we trained a decoder with two fully-connected hidden layers (h1: 512 units, h2: 1024 units) each followed by ELU nonlinearities to reconstruct the input image. The reconstruction loss was then calculated as the squared difference between the pixel values of the input image and the reconstructed image. The total loss is given by the following formulas:

$$L_{total} = \alpha_{shape\ type} L_{shape\ type} + \alpha_{vernier\ offset} L_{vernier\ offset} + \alpha_{shape\ repetitions} L_{shape\ repetitions} + \alpha_{reconstruction} L_{reconstruction} \quad (1)$$

$$L_{shape\ type} = \sum_k T_k \max(0,\ (m^+ - \|v_k\|)^2) + \lambda(1 - T_k) \max(0,\ (\|v_k\| - m^-)^2) \quad (2)$$

$$L_{vernier\ offset} = Crossentropy(vernier\ labels, vernier\ decoder\ output) \quad (3)$$

$$L_{shape\ repetitions} = Crossentropy(shape\ repetitions\ labels, shape\ repetitions\ decoder\ output) \quad (4)$$

$$L_{reconstruction} = \sum_{i,j} (input(i,j) - reconstruction(i,j))^2 \quad (5)$$

Where the α are real numbers scaling each loss term, T_k = 1 if shape class k is present, $\|v_k\|$ is the norm of output capsule k, and m^+, m^- and λ are parameters of the margin loss with the same values as described in (2).

After training, we tested vernier discrimination performance on (un)crowding stimuli (figure 4d), and obtained input reconstructions. We trained 10 different networks and averaged their performance. Before this experiment, the network had never seen crowding nor uncrowding stimuli, but it knew about groups of shapes and about the vernier discrimination task. Therefore, the network could not trivially learn when to (un)crowd by overfitting on the training dataset. This situation is similar for humans: they know about shapes and verniers, but their visual system has never been trained on (un)crowding stimuli.

To check that CapsNets explain uncrowding because of the grouping and segmentation capabilities offered by routing by agreement and not merely because of the way they are trained, we replaced the capsule layers by other architectures (a feedforward fully connected layer, a fully connected layer with lateral recurrent connections and a fully connected layer with top-down recurrent connections to the layer below; see Results). All these networks had the same number of neurons as our CapsNets, and we used the same number of recurrent iterations as the number of routing by agreement used for the CapsNets. The networks were trained and tested in exactly the same way, with the same losses and datasets. The only difference is that CapsNets represent different classes in different capsules, so we could decode information directly from specific capsules (for example, we could decode vernier offsets specifically from the vernier capsule, or reconstruct squares specifically from the squares capsule). The other networks do not offer this possibility, because different classes are not represented in different known groups of neurons. Therefore, we decoded vernier offsets, reconstructions, the number of shapes and the shape type from the entire last layer of the network rather than from specific capsules. This difference did not limit the networks' performance, since these architectures performed well during training. Hence, the fact that they do not produce uncrowding is not explained by training limitations, but rather by the fact that they *generalize* to novel inputs differently than CapsNets.

Experiment 2:

Psychophysical experiment:

Observers

For experiment 2, we collected human psychophysical data. Participants were paid students of the Ecole Polytechnique Fédérale de Lausanne (EPFL). All had normal or corrected-to-normal vision, with a visual acuity of 1.0 (corresponding to 20/20) or better in at least one eye, measured with the Freiburg Visual Acuity Test. Observers were told that they could quit the experiment at any time they wished. Five observers (two females) performed the experiment.

Apparatus and stimuli

Stimuli were presented on a HP-1332A XY-display equipped with a P11 phosphor and controlled by a PC via a custom-made 16-bit DA interface. Background luminance of the screen was below 1 cd/m^2. Luminance of stimuli was 80 cd/m^2. Luminance measurements were performed using a Minolta Luminance meter LS-100. The experimental room was dimly illuminated (0.5 lx). Viewing distance was 75 cm.

We determined vernier offset discrimination thresholds for different flanker configurations. The vernier target consisted of two lines that were randomly offset either to the left or right. Observers indicated the offset direction. Stimulus consisted of two vertical 40' (arcmin) long lines separated by a vertical gap of 4' and presented at an eccentricity of 5° to the right of a fixation cross (6' diameter). Eccentricity refers to the center of the target location. Flanker configurations were centered on the vernier stimulus and were symmetrical in the horizontal dimension. Observers were presented two flanker configurations. In the lines configuration, the vernier was flanked by two vertical lines (84') at 40' from the vernier. In the cuboids configuration, perspective cuboids were presented to the left and to the right of the vernier (width = 58', angle of oblique lines = 135◦, length = 23.33'). Cuboids contained the lines from the Lines condition as their centermost edge.

Procedure

Observers were instructed to fixate a fixation cross during the trial. After each response, the screen remained blank for a maximum period of 3 s during which the observer was required to make a response on vernier offset discrimination by pressing one of two push buttons. The screen was blank for 500 ms between response and the next trial.

An adaptive staircase procedure (PEST; 63) was used to determine the vernier offset for which observers reached 75% correct responses. Thresholds were determined after fitting a cumulative Gaussian to the data using probit and likelihood analyses. In order to avoid extremely large vernier offsets,

we restricted the PEST procedure to not exceed 33.3' i.e. twice the starting value of 16.66'. Each condition was presented in separate blocks of 80 trials. All conditions were measured twice (i.e., 160 trials) and randomized individually for each observer. To compensate for possible learning effects, the order of conditions was reversed after each condition had been measured once. Auditory feedback was provided after incorrect or omitted responses.

Modelling:

To model the results of experiment 2, we trained our CapsNets to solve a vernier offset discrimination task and classify verniers, cuboids, scrambled cuboids and lines. The training dataset included vernier stimuli and one of three different shape types (lines, cuboids, scrambled cuboids). The scrambled cuboids were included to make the task harder, and to prevent the network from classifying cuboids simply based on the number of pixels in the image. The line stimuli were randomly presented in a group of 2, 4, 6 or 8. Both, cuboids and shuffled cuboids were always presented in groups of two facing one another. The distance between these shapes was varied randomly between one and six pixels. The loss function was very similar to experiment 1, but without the loss term for shape repetitions, since there were no repetitions (each term is the same as in eqs. 1-5):

$$L_{total} = \alpha_{shape\ type} L_{shape\ type} + \alpha_{vernier\ offset} L_{vernier\ offset} + \alpha_{reconstruction} L_{reconstruction} \quad (6)$$

After training, we tested the network's vernier discrimination performance on (un)crowding stimuli (verniers surrounded by either lines, cuboids or scrambled cuboids), while varying the number of recurrent routing by agreement iterations. We trained the same network 50 times and averaged performance over these trained networks, excluding 21 networks for which vernier discrimination performance with *both* line and cuboid flankers was at ceiling (>=95%) or floor (<=55%). This exclusion criterion is used for cleaner results and does *not* impact the crucial result showing that uncrowding occurs with increasing routing iterations only with cuboid, but not with line flankers. The effect still occurs when all 50 networks are included in the analysis, but the fact that certain networks are at floor or ceiling is misleading. Before this experiment, the network had never seen (un)crowding stimuli, but it knew about cuboids, scrambled cuboids and about the vernier discrimination task. Therefore, the network could not trivially learn when to (un)crowd by overfitting on the training dataset.

Acknowledgements

Adrien Doerig was supported by the Swiss National Science Foundation grant n.176153 "Basics of visual processing: from elements to figures".

Bibliography

1. A. Doerig, *et al.*, Beyond Bouma's window: How to explain global aspects of crowding? *PLOS Computational Biology* **15**, e1006580 (2019).

2. S. Sabour, N. Frosst, G. E. Hinton, Dynamic routing between capsules in *Advances in Neural Information Processing Systems*, (2017), pp. 3856–3866.

3. J. J. DiCarlo, D. Zoccolan, N. C. Rust, How Does the Brain Solve Visual Object Recognition? *Neuron* **73**, 415–434 (2012).

4. D. H. Hubel, T. N. Wiesel, Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology* **160**, 106–154 (1962).

5. A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks in *Advances in Neural Information Processing Systems*, (2012), pp. 1097–1105.

6. S. A. Eslami, *et al.*, Neural scene representation and rendering. *Science* **360**, 1204–1210 (2018).

7. L. Gatys, A. S. Ecker, M. Bethge, "Texture Synthesis Using Convolutional Neural Networks" in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, R. Garnett, Eds. (Curran Associates, Inc., 2015), pp. 262–270.

8. T. Karras, S. Laine, T. Aila, A style-based generator architecture for generative adversarial networks. *arXiv preprint arXiv:1812.04948* (2018).

9. V. A. Lamme, P. R. Roelfsema, The distinct modes of vision offered by feedforward and recurrent processing. *Trends in neurosciences* **23**, 571–579 (2000).

10. T. C. Kietzmann, *et al.*, Recurrence required to capture the dynamic computations of the human ventral visual stream. *arXiv preprint arXiv:1903.05946* (2019).

11. C. M. Funke, *et al.*, Comparing the ability of humans and DNNs to recognise closed contours in cluttered images in *18th Annual Meeting of the Vision Sciences Society (VSS 2018)*, (2018), p. 213.

12. J. Su, D. V. Vargas, K. Sakurai, One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation* (2019).

13. C. Szegedy, *et al.*, Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

14. R. Geirhos, *et al.*, ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231* (2018).

15. N. Baker, H. Lu, G. Erlikhman, P. J. Kellman, Deep convolutional networks do not classify based on global object shape. *PLoS computational biology* **14**, e1006613 (2018).

16. W. Brendel, M. Bethge, Approximating CNNs with Bag-of-local-Features models works surprisingly well on ImageNet. *arXiv preprint arXiv:1904.00760* (2019).

17. T. Kim, W. Bair, A. Pasupathy, Neural coding for shape and texture in macaque area V4. *Journal of Neuroscience* **39**, 4760–4774 (2019).

18. A. Doerig, A. Bornet, O. H. Choung, M. H. Herzog, Crowding Reveals Fundamental Differences in Local vs. Global Processing in Humans and Machines. *bioRxiv*, 744268 (2019).

19. K. Hermann, S. Kornblith, Exploring CNN Inductive Biases: Shape vs. Texture. *NeurIPS Workshop on Shared Visual Representations in Human & Machine Intelligence* (2019).

20. J. Deng, *et al.*, Imagenet: A large-scale hierarchical image database in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, (Ieee, 2009), pp. 248–255.

21. S.-M. Khaligh-Razavi, N. Kriegeskorte, Deep supervised, but not unsupervised, models may explain IT cortical representation. *PLoS computational biology* **10**, e1003915 (2014).

22. D. L. Yamins, *et al.*, Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences* **111**, 8619–8624 (2014).

23. B. RichardWebster, S. Anthony, W. Scheirer, Psyphy: A psychophysics driven evaluation framework for visual recognition. *IEEE transactions on pattern analysis and machine intelligence* (2018).

24. D. M. Levi, Crowding—An essential bottleneck for object recognition: A mini-review. *Vision Research* **48**, 635–654 (2008).

25. D. Whitney, D. M. Levi, Visual crowding: a fundamental limit on conscious perception and object recognition. *Trends in Cognitive Sciences* **15**, 160–168 (2011).

26. H. Bouma, Visual interference in the parafoveal recognition of initial and final letters of words. *Vision Research* **13**, 767–782 (1973).

27. D. G. Pelli, Crowding: a cortical constraint on object recognition. *Current Opinion in Neurobiology* **18**, 445–451 (2008).

28. M. Manassi, D. Whitney, Multi-level Crowding and the Paradox of Object Recognition in Clutter. *Current Biology* **28**, R127–R133 (2018).

29. M. Manassi, S. Lonchampt, A. Clarke, M. H. Herzog, What crowding can tell us about object representations. *Journal of Vision* **16**, 35–35 (2016).

30. G. Westheimer, G. Hauske, Temporal and spatial interference with vernier acuity. *Vision research* **15**, 1137–1141 (1975).

31. D. M. Levi, S. A. Klein, A. P. Aitsebaomo, Vernier acuity, crowding and cortical magnification. *Vision research* **25**, 963–977 (1985).

32. D. Oberfeld, P. Stahn, Sequential grouping modulates the effect of non-simultaneous masking on auditory intensity resolution. *PloS one* **7**, e48054 (2012).

33. K. E. Overvliet, B. Sayim, Perceptual grouping determines haptic contextual modulation. *Vision Research* **126**, 52–58 (2016).

34. T. P. Saarela, B. Sayim, G. Westheimer, M. H. Herzog, Global stimulus configuration modulates crowding. *Journal of Vision* **9**, 5–5 (2009).

35. M. H. Herzog, M. Fahle, Effects of grouping in contextual modulation. *Nature* **415**, 433 (2002).

36. B. Sayim, G. Westheimer, M. H. Herzog, Gestalt factors modulate basic spatial vision. *Psychological Science* **21**, 641–644 (2010).

37. T. P. Saarela, G. Westheimer, M. H. Herzog, The effect of spacing regularity on visual crowding. *Journal of Vision* **10**, 17–17 (2010).

38. M. Manassi, B. Sayim, M. H. Herzog, Grouping, pooling, and when bigger is better in visual crowding. *Journal of Vision* **12**, 13–13 (2012).

39. M. V. Pachai, A. C. Doerig, M. H. Herzog, How best to unify crowding? *Current Biology* **26**, R352–R353 (2016).

40. M. Manassi, B. Sayim, M. H. Herzog, When crowding of crowding leads to uncrowding. *Journal of Vision* **13**, 10–10 (2013).

41. J. M. Wallace, M. K. Chiu, A. S. Nandy, B. S. Tjan, Crowding during restricted and free viewing. *Vision Research* **84**, 50–59 (2013).

42. S. P. Tripathy, P. Cavanagh, H. E. Bedell, Large crowding zones in peripheral vision for briefly presented stimuli. *Journal of Vision* **14**, 11–11 (2014).

43. E. A. Styles, D. A. Allport, Perceptual integration of identity, location and colour. *Psychological Research* **48**, 189–200 (1986).

44. Z. Li, Visual segmentation by contextual influences via intra-cortical interactions in the primary visual cortex. *Network: computation in neural systems* **10**, 187–212 (1999).

45. L. Parkes, J. Lund, A. Angelucci, J. A. Solomon, M. Morgan, Compulsory averaging of crowded orientation signals in human vision. *Nature neuroscience* **4**, 739 (2001).

46. D. G. Pelli, M. Palomares, N. J. Majaj, Crowding is unlike ordinary masking: Distinguishing feature integration from detection. *Journal of Vision* **4**, 12–12 (2004).

47. R. Rosenholtz, D. Yu, S. Keshvari, Challenges to pooling models of crowding: Implications for visual mechanisms. *Journal of vision* **19** (2019).

48. S. Thorpe, D. Fize, C. Marlot, Speed of processing in the human visual system. *nature* **381**, 520 (1996).

49. T. C. Kietzmann, P. McClure, N. Kriegeskorte, Deep neural networks in computational neuroscience. *bioRxiv*, 133504 (2018).

50. J. Kim, D. Linsley, K. Thakkar, T. Serre, Disentangling neural mechanisms for perceptual grouping. *arXiv preprint arXiv:1906.01558* (2019).

51. H. Tang, *et al.*, Recurrent computations for visual pattern completion. *Proceedings of the National Academy of Sciences* **115**, 8835–8840 (2018).

52. C. J. Spoerer, T. C. Kietzmann, N. Kriegeskorte, Recurrent networks can recycle neural resources to flexibly trade speed for accuracy in visual recognition. *bioRxiv*, 677237 (2019).

53. D. Linsley, J. Kim, T. Serre, Sample-efficient image segmentation through recurrence. *arXiv:1811.11356 [cs]* (2018) (June 27, 2019).

54. K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators. *Neural networks* **2**, 359–366 (1989).

55. T. Serre, Deep learning: the good, the bad, and the ugly. *Annual Review of Vision Science* **5**, 399–426 (2019).

56. C. J. Spoerer, P. McClure, N. Kriegeskorte, Recurrent convolutional neural networks: a better model of biological object recognition. *Frontiers in psychology* **8**, 1551 (2017).

57. K. Kar, J. Kubilius, K. Schmidt, E. B. Issa, J. J. DiCarlo, Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior. *Nature neuroscience* **22**, 974 (2019).

58. G. Francis, M. Manassi, M. H. Herzog, Neural dynamics of grouping and segmentation explain properties of visual crowding. *Psychological review* **124**, 483 (2017).

59. O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, (Springer, 2015), pp. 234–241.

60. R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, K. He, *Detectron* (2018).

61. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* **15**, 1929–1958 (2014).

62. I. Loshchilov, F. Hutter, Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983* (2016).

63. M. Taylor, C. D. Creelman, PEST: Efficient estimates on probability functions. *The Journal of the Acoustical Society of America* **41**, 782–787 (1967).

Supplementary Material

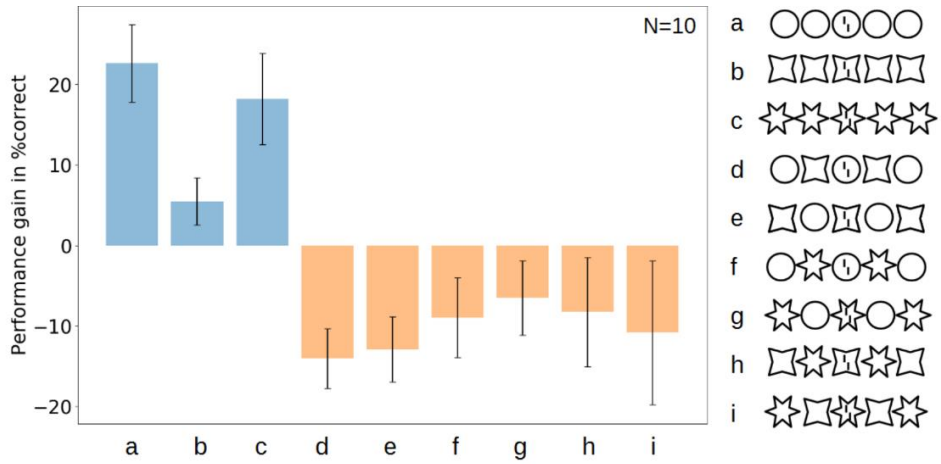
Experiment 1

Results are robust against stimuli and hyperparameters changes

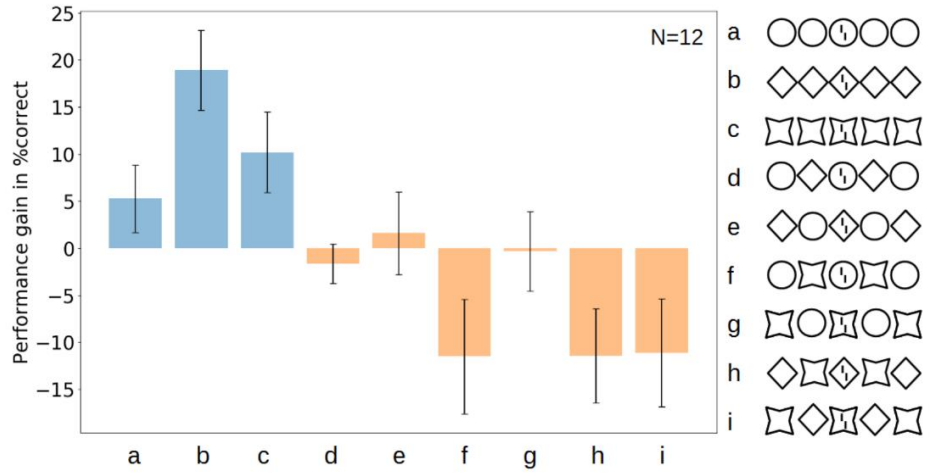
To avoid cherrypicking our hyperparameters, we ran several networks with different hyperparameter sets, and show that our results are robust with respect to these changes.

The results of experiment 1 remain qualitatively similar for different image sizes and network hyperparameters. Below is a selection of results using different sets of hyperparameters. In all these cases, both crowding and uncrowding occur, similarly to the results shown in Figure 2.

a)

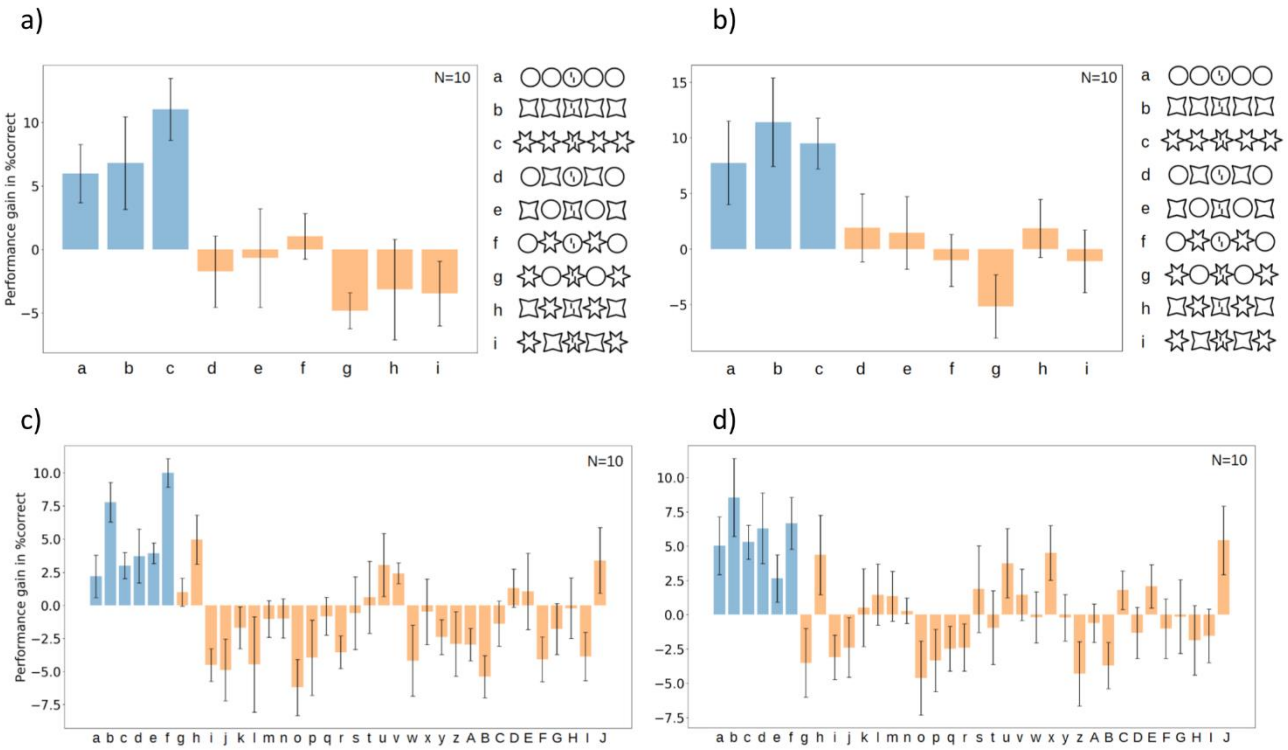


b)



Network ID	(a)	(b)
Primary capsule types	4	4
Primary capsule dims	1	1
Secondary capsule types	4	4
Secondary capsule dims	3	4
Routing iterations	5	3
Training steps	4000	4000
Learning rate decay steps	100	400
Shape repetition loss	x	x
Taining noise σ range	[0.00, 0.02]	[0.00, 0.00]
Test noise σ range	[0.02, 0.02]	[0.01, 0.01]

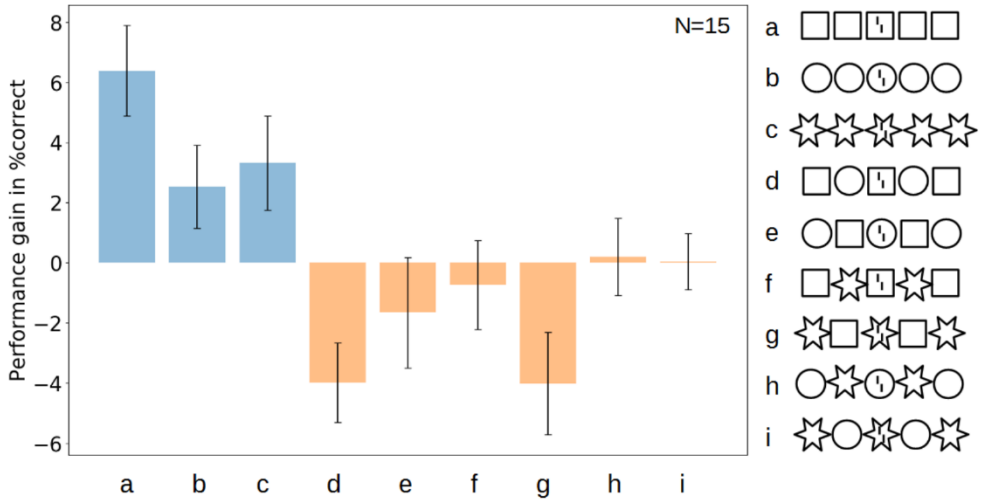
Supplementary Figure 1: Results for 16x72 pixel images. Both crowding and uncrowding occur similarly to the results in figure 2. Plotting conventions are the same as in figure 2. Main hyperparameters are summarized at the bottom. With these small images, we often encountered ceiling effects. We trained 20 networks and dropped those that were at ceiling (i.e., we dropped networks that were at 100% performance for all conditions).



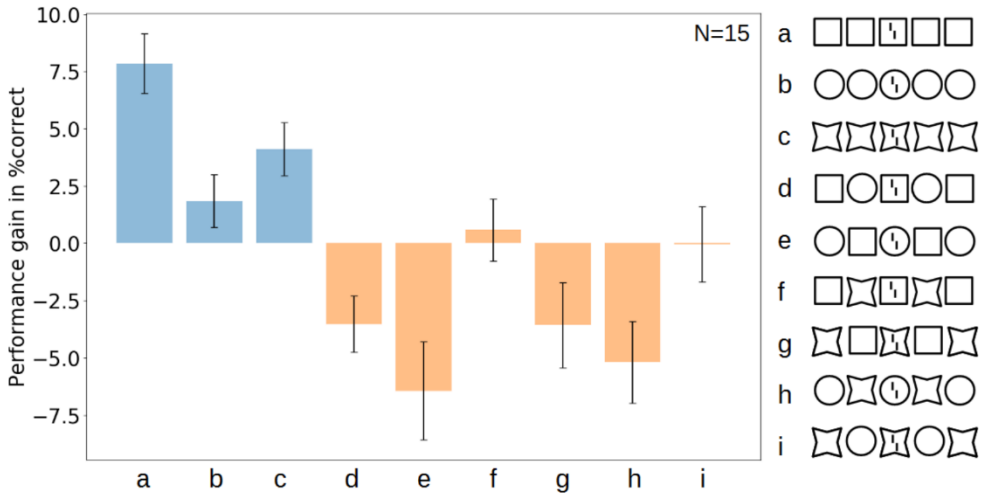
Network ID	(a)	(b)	(c)	(d)
Primary capsule types	4	4	7	7
Primary capsule dims	1	1	2	2
Secondary capsule types	4	4	7	7
Secondary capsule dims	4	4	8	10
Routing iterations	3	5	3	3
Training steps	8000	6000	2500	5000
Shape repetition loss	x	x	x	x
Location loss			x	x
Reconstruction loss			x	x
Gaussian training noise	[0.00, 0.05]	[0.00, 0.00]	[0.02, 0.04]	[0.02, 0.04]
Gaussian test noise	[0.05, 0.05]	[0.05, 0.05]	[0.04, 0.06]	[0.04, 0.06]

Supplementary Figure 2: 20x72 pixel images. Both crowding and uncrowding occur similarly to the results in figure 2. Plotting conventions are the same as in figure 2. Main hyperparameters are summarized at the bottom. Stimuli not shown for panels b&c, for clarity.

a)



b)



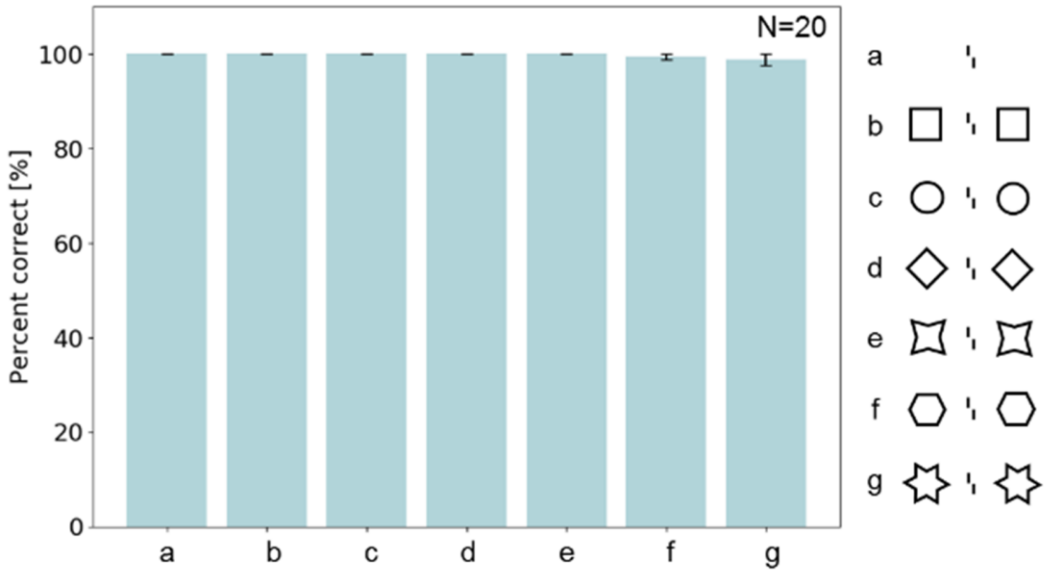
Network ID	(a)	(b)
Primary capsule types	20	20
Primary capsule dims	1	1
Secondary capsule types	4	4
Secondary capsule dims	12	12
Routing iterations	4	5
Training steps	3000	3000
First decay steps	500	500
Shape repetition loss	x	x
Gaussian training noise	[0.00, 0.00]	[0.00, 0.02]
Gaussian test noise	[0.01, 0.01]	[0.01, 0.01]

Supplementary Figure 3: 30x72 pixel images. Both crowding and uncrowding occur similarly to the results in figure 2. Plotting conventions are the same as in figure 2. Main hyperparameters are summarized at the bottom.

Performance deterioration is due to crowding

As a control to check that performance dropped because of crowding and not merely because of the simultaneous presentation of a vernier target and another shape, we measured performance when the vernier was presented outside, rather than inside, flanking shapes. Performance does not drop in

this case, compared to when the vernier is presented alone. This suggests that performance drops because of crowding in the networks.



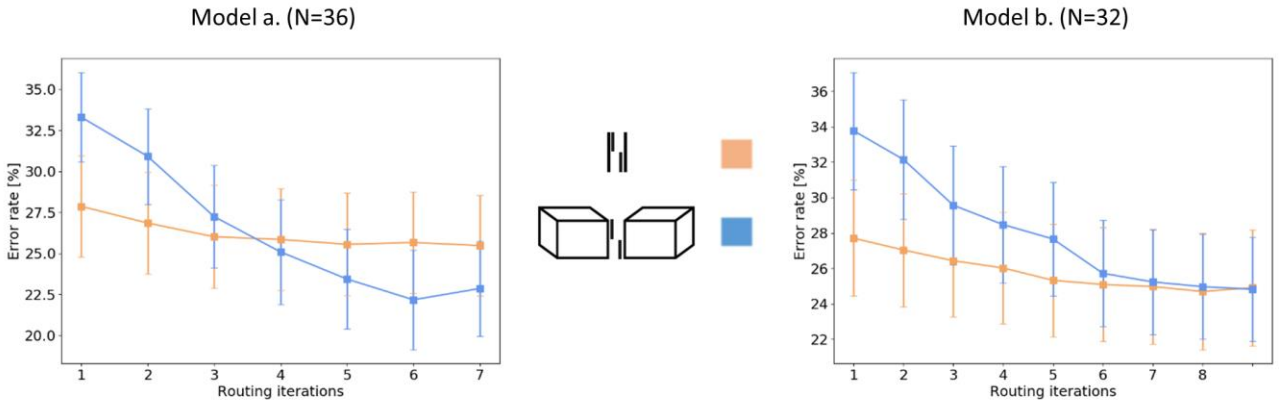
Supplementary Figure 4: Performance deterioration is due to crowding. The x-axis shows different conditions shown on the right, the y-axis shows vernier offset discrimination percent correct. Vernier accuracy does not decrease when the vernier is presented outside flanking shapes compared to the vernier alone condition.

Experiment 2

Results are robust against stimuli and hyperparameters changes

To avoid cherrypicking our hyperparameters, we ran several networks with different hyperparameter sets, and show that our results are robust with respect to these changes.

The results of experiment 2 remain qualitatively similar for different network hyperparameters. Below is a selection of results using different sets of hyperparameters. In both these cases, performance on the cuboids condition, but not the lines condition, drastically improves with the number of recurrent routing by agreement iterations (network a: lines: p = 0.041 vs. cuboids p = .0.0005, network b: lines: 0.11 vs. cuboids p=0.006). In network a, the lines show a marginally significant improvement, but the p-value is 100 times smaller than for the cuboids.



Model	a	b
Primary capsule types	4	4
Primary capsule dims	1	1
Secondary capsule types	4	4
Secondary capsule dims	4	3
Training routing iterations	7	9
Training steps	4000	4500
Gaussian training noise	[0.00, 0.02]	[0.00, 0.02]
Gaussian testing noise	[0.00, 0.05]	[0.00, 0.05]

Supplementary Figure 5: Experiment 2 results are reproduced with different network hyperparameters. The x-axis shows different numbers of routing iterations during testing and the y-axis shows the corresponding error rates (i.e., lower values indicate better performance). Error bars indicate standard deviation across N trained networks (see Methods). Performance increases drastically with recurrent routing iterations only for the cuboids condition, and not for the lines condition. A difference with the results shown in figure 3 is that performance with cuboids flankers is worse than performance with line flankers at early iterations. This may be explained by the far greater amount of pixels in cuboids than lines, increasing the interference between the cuboids and the vernier until the cuboids are segmented away. As the results exhibited in Figure 3 show, this effect can be mitigated through adequate hyperparameter choice. However, in this experiment, we focused on demonstrating that only the cuboids benefit from additional routing iterations, and this result is very stable across hyperparameter changes.