Recurrent networks can recycle neural resources to flexibly trade speed for accuracy in visual recognition

Courtney J Spoerer[1*], Tim C Kietzmann[1], Nikolaus Kriegeskorte[2]

1 Medical Research Council Cognition and Brain Sciences Unit, University of Cambridge, Cambridge, United Kingdom
2 Department of Psychology, Department of Neuroscience, Department of Electrical Engineering, Zuckerman Mind Brain Behavior Institute, Columbia University, New York, NY, USA

* courtney.spoerer@mrc-cbu.cam.ac.uk

Abstract

Deep feedforward neural network models of vision dominate in both computational neuroscience and engineering. However, the primate visual system contains abundant recurrent connections. Recurrent signal flow enables recycling of limited computational resources over time, and so might boost the performance of a physically finite brain. In particular, recurrence could improve performance in vision tasks. Here we find that recurrent convolutional networks outperform feedforward convolutional networks matched in their number of parameters in large-scale visual recognition tasks. Moreover, recurrent networks can trade off accuracy for speed, balancing the cost of error against the cost of a delayed response (and the cost of greater energy consumption). We terminate recurrent computation once the output probability distribution has concentrated beyond a predefined entropy threshold. Trained by backpropagation through time, recurrent convolutional networks resemble the primate visual system in terms of their speed-accuracy trade-off behaviour. Moreover, their learned lateral connectivity patterns are consistent with those observed in primate early visual cortex. These results suggest that recurrent models are preferable to feedforward models of vision, both in terms of their performance at vision tasks and their ability to explain biological vision.

Author summary

Deep neural networks (DNNs) provide the best current models of biological vision and achieve the highest performance in computer vision. Although originally inspired by the primate brain, these models are still missing important functional elements of their biological counterparts. One biological feature typically absent from models for visual object recognition is the ability to recycle limited neural resources by processing information recurrently. We report that including connections that let information flow in cycles can improve performance, even as the total number of connections is held constant. Recurrent processing also enabled DNNs to behave more flexibly and trade off speed for accuracy. Similar to the primate brain, the networks can compute longer to boost accuracy for objects that are more difficult to recognise. This work shows how a known feature of the primate brain contributes to its computational function and suggests that taking inspiration from biology can help us further improve artificial vision systems.

Introduction

Neural networks have a long history as models of biological vision [1–3] and the recent success of deep neural networks (DNNs) in computer vision has led to a renewed interest in neural network models within neuroscience [4–6]. Contemporary deep neural networks not only perform better in machine learning challenges but also provide better predictions of neural and behavioural data than previous, shallower models [7–11].

While deep neural networks have provided better models of biological vision, there are significant discrepancies between models and brains in terms of both computational mechanisms and recognition behaviour. In terms of recognition behaviour, networks and primates do show similar patterns of image classifications at the level of object categories, but their behaviour diverges when the comparison is made at the level of individual images [12]. Moreover, it has been shown that DNNs heavily rely on texture in image classification, whereas humans more strongly rely on larger-scale shape information [13].

In terms of computational mechanisms, DNNs diverge from biology in that they are typically rate-coded rather than spiking, feedforward rather than recurrent, and trained using backpropagation on millions of labelled images. While some degree of abstraction is necessary when modelling complex systems such as the brain, it is important to understand which features of biology are essential to the computations as reflected in task performance [6].

One area that has received particular interest within machine learning and neuroscience has been the lack of recurrence in deep neural networks for object recognition. Although core object recognition has typically been viewed as a feedforward process in primates [14], it is known from neuroanatomy that the visual system is highly recurrent [15–17]. Functional evidence also indicates that recurrent computations are utilised during object recognition [18–25].

Recent work has focused on introducing recurrence into the framework of convolutional neural networks for processing static images, with a particular focus on object recognition [26–30]. These recurrent convolutional neural networks (RCNNs) are better able to explain neural and behavioural data than their feedforward counterparts [24, 25, 29, 31, 32]. Additionally, recurrence brings performance benefits in object recognition tasks, with recurrent networks outperforming feedforward networks of similar complexity (typically measured by the number of parameters) [26–29]. Performance gains have previously been shown for small-scale tasks [26–28] or using specialised forms of recurrence [29]. An important open question, which we address here, is whether simple recurrent extensions of the convolutional framework can bring performance gains on large-scale recognition tasks when the number of parameters is matched to feedforward control models.

Beyond the number of parameters, we must consider the computational cost of recognition. A recurrent network might outperform a feedforward network with a similar number of parameters, but require more computation (and time) to arrive at an accurate answer. If we look to how the brain performs object recognition, we see a more flexible mechanism: Extensive recurrent computations are not always required. For some images, fast feedforward computations are sufficient [24]. This aligns with our current understanding of biological decision-making, where evidence about a decision is accumulated until a threshold is reached and a decision made [33]. If the network converges on a decision in the initial feedforward sweep, then recurrent computation is not required. Using threshold-based decision making might allow RCNNs to save time and energy by only running for the number of time steps required for a given level of confidence.

A further benefit of threshold-based decisions is the ability to implement speed-accuracy trade-offs (SATs), another feature of biological object recognition [34].

In engineering, this has been implemented using a range of separate neural network models of varying scale (e.g. [35]). However, a threshold-based mechanism would allow a range of SATs to be implemented by a single RCNN without any need for additional training. This appears advantageous for both biological and artificial object recognition, which similarly face limitations of memory, time, and energy.

To better understand the role of recurrent computations in artificial and biological visual systems, we explore how recurrent DNNs that trade off speed and accuracy compare to feedforward control models in terms of performance, and how their learned recurrent connectivity and behaviour compares to primate brains. We train these networks on the ImageNet Large Scale Visual Recognition Challenge (referred to as *ImageNet* for brevity) [36], and a more ecologically valid recognition task called *ecoset* [37]. We look to see whether recurrence brings performance gains in these tasks and integrate threshold-based decision making in RCNNs, varying the threshold to control the SAT [34]. Finally, we look to see whether the computations performed in RCNNs capture properties of biological visual systems by testing whether the dynamics of RCNNs predict human object recognition behaviour and by comparing the learned lateral connectivity of RCNNs to connectivity in primate early visual cortex.

Results

We trained a range of deep convolutional neural networks on two large-scale visual object-recognition tasks, ImageNet [36] and ecoset [37]. The networks trained included a feedforward network, referred to as B (for bottom-up only), and a recurrent network, referred to as BL, with bottom-up and lateral recurrent connections (recurrent connections within a layer). We focus our investigation on lateral connections, which constitute a form of recurrence that is ubiquitous in biological visual systems and proved more powerful than top-down recurrent connections on simple tasks in earlier work [28].

The recurrent networks are implemented by unrolling the computational graph of the recurrent network for a finite number of time steps (see Methods). The model is trained to produce a readout at each time step, which predicts the category of the object present in the image.

As the addition of recurrent connections adds more parameters to the models, we use three larger feedforward architectures that are approximately matched in the number of parameters (Fig. 1) as control models. The first of these architectures (referred to as B-K) uses larger kernel sizes. This has the benefit of having the same number of units in each layer as B and only changes the number of incoming connections for each unit. However, increasing the kernel size may be an unconventional way to spend additional parameters in a feedforward network. We therefore also included control models with a larger number of features (referred to as B-F) in each layer. These models have a larger number of units than B, but keep the number of layers fixed. Finally, we trained a deeper feedforward network (referred to as B-D), approximately matching the number of parameters to BL by doubling the number of layers. Increasing the number of layers is, arguably, the most common and effective way to make a feedforward network larger and more powerful.

Recurrent networks outperform parameter-matched feedforward models

We compared the performance of recurrent networks and the feedforward networks, including the parameter-matched controls on both tasks. For the recurrent networks, BL, we defined the prediction of the model as the average of the category readout across

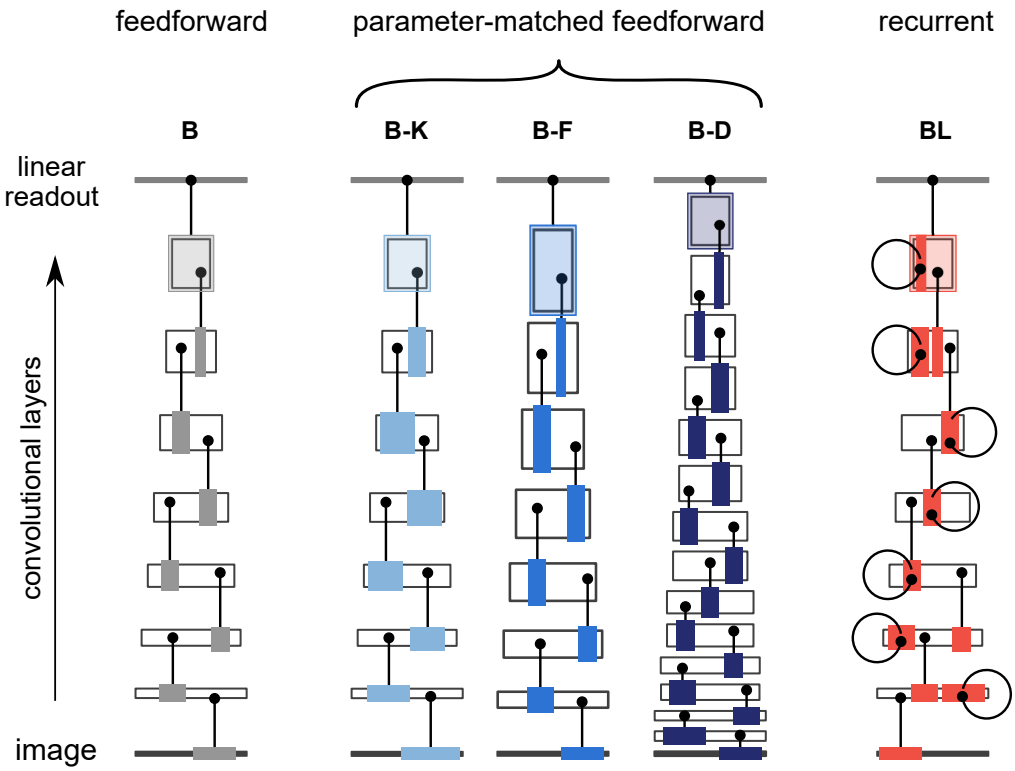

Fig 1. A schematic representation of the networks trained. White boxes represent convolutional layers, the width represents spatial dimensions of the convolutional layers and height represents the number feature maps. Example units are shown with coloured regions representing the extent of the layer acting as input to the unit. The areas represented in these diagrams are illustrative and are not drawn to scale.

all time steps, referred to as the cumulative readout. The cumulative readout tends to produce the best results (see Methods).

The recurrent models performed best, outperforming both the baseline feedforward model, B, and the parameter-matched controls, on both data sets (Fig. 2B). BL showed a performance benefit of over 1.5 percentage points relative to the best feedforward model, B-D, on both tasks (Table 1).

Table 1. Accuracies on held-out data and number of parameters for each model

models	ImageNet	ecoset	parameters
B (baseline)	58.42%	64.25%	11.0 million
B-K (larger kernels)	56.46%	62.81%	39.8 million
B-F (more feature maps)	60.34%	66.54%	40.0 million
B-D (deeper network)	62.68%	68.36%	28.9 million
BL (recurrent)	**64.37%**	**69.98%**	28.9 million

The number of parameters are calculated for ImageNet models, ecoset models have slightly fewer parameters due to fewer categories in the final readout layer.

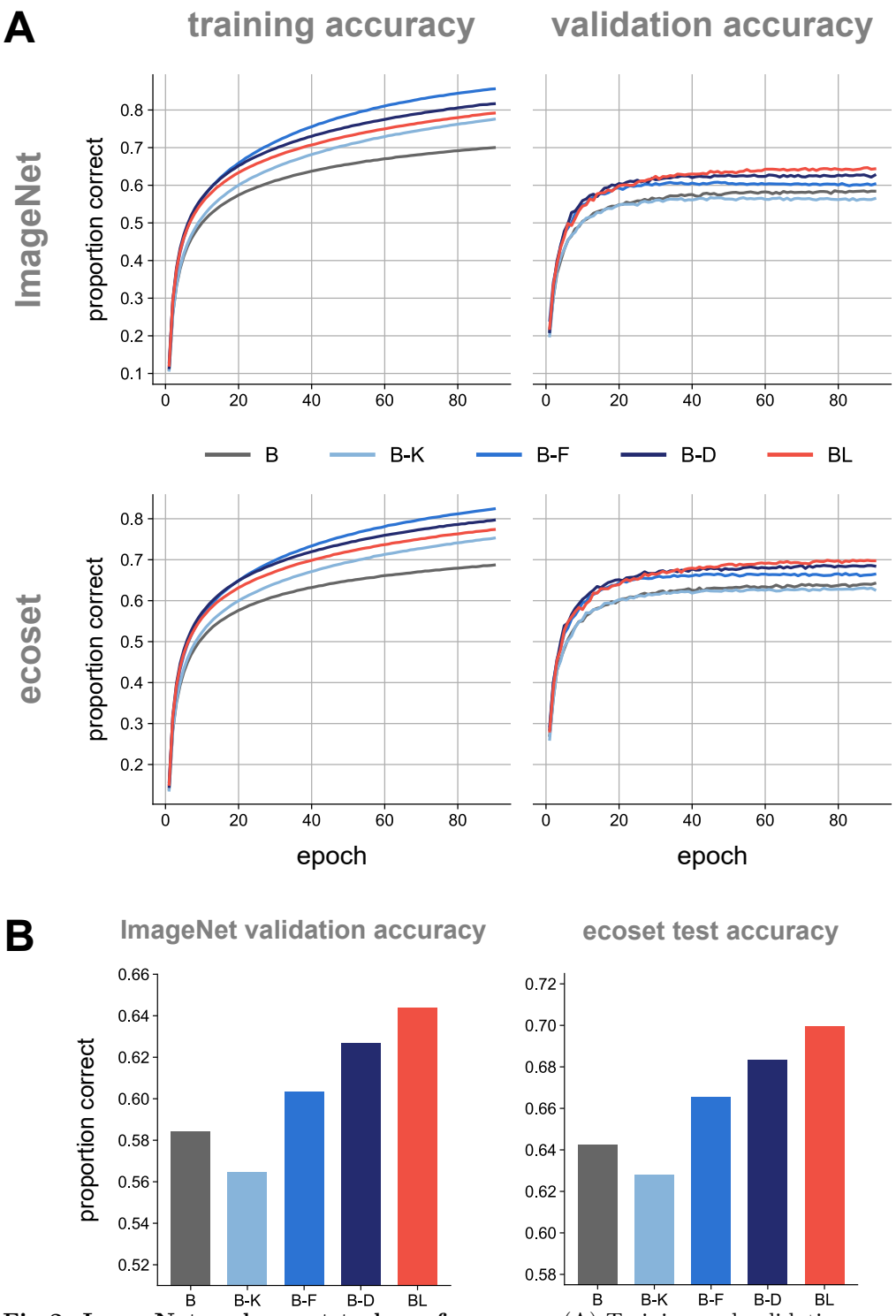

Fig 2. ImageNet and ecoset task performance. (A) Training and validation accuracies across training for all networks. (B) Performance of networks on held-out data using the fully-trained networks.

Both B-D (deeper network) and B-F (more feature maps) outperformed the baseline model, B. B-K has a worse test accuracy than the baseline model but a higher training

accuracy (Fig. 2A). This suggests that using additional parameters to increase the kernel size in our models leads to overfitting rather than a generalisable increase in performance.

Pairwise McNemar tests [38,39] showed all differences in model performance to be significant ($p \leq 0.05$). Bonferroni correction was used to correct for multiple comparisons by controlling the family-wise error rate at less than or equal to 0.05.

Single recurrent models span speed-accuracy trade-offs of multiple feedforward models

We compared the computational efficiency of feedforward and recurrent networks by measuring the accuracy as a function of the number of floating-point operations (Fig. 3). The number of floating-point operations of a model reflects the energy cost, which might be related to the metabolic cost in a biological system. A feedforward model has a fixed computational cost, whereas a recurrent model can flexibly terminate computations when confidence passes a threshold, trading off accuracy for speed.

In the context of a particular recurrent model, the computational cost is proportional to the number of time steps that the model runs for and thus to the reaction time. When interpreted as models of brains, our recurrent models therefore make predictions about speed-accuracy trade-offs. Note that reaction time and computational cost may diverge when comparing architectures that employ parallel processing to different degrees (trading off speed for fewer units). However, the trade-off between parallel physical resources (connections and units) and time is beyond the scope of this paper. We focus on comparisons between models matched in their numbers of parameters, where computational cost is proportional to reaction time.

For the recurrent models, we used cumulative readouts with entropy thresholding. The network runs until the entropy of its cumulative readout falls below a predefined threshold. The final cumulative readout is then taken as the network's prediction. This effectively takes an internal estimate of the networks' confidence in the decision and terminates once a desired confidence level is reached. Entropy thresholding has the benefit of being economical, as it uses the minimum number of time steps to reach the required level of confidence for an image. Moreover, it closely corresponds to theories of biological decision making, where evidence is accumulated until it reaches a bound [33].

A recurrent model may choose to compute longer for harder images. The number of time steps required to pass the entropy threshold varies across the test set. For a given entropy threshold, we define the computational cost for a recurrent model as the average across the test set of the number of operations used. We plot the accuracy of the model as a function of the computational cost (Fig. 3). For a given recurrent model, the resulting plot reflects the speed-accuracy trade-off, because the reaction time is proportional to the computational cost. Feedforward models are represented by single points because their computational cost and reaction time are constant across images.

When comparing the recurrent models to feedforward models we see a remarkable correspondence between the two classes of architecture (Fig. 3): The accuracy of the recurrent models as a function of the computational cost passes through the points describing the feedforward control models. This means that the different architectures yield the same accuracy for a given computational budget. However, the computational costs and accuracies of the feedforward models are fixed, whereas the recurrent models can be left to compute longer so as to achieve higher accuracies.

To inferentially compare the performance of the feedforward and recurrent networks at matched computational cost, we consider the performance of the recurrent networks at a single entropy threshold. We select the threshold that minimises the absolute difference between the average number of operations for the recurrent network and the

number of operations for the feedforward network. McNemar tests were again used to compare the performance of the networks.

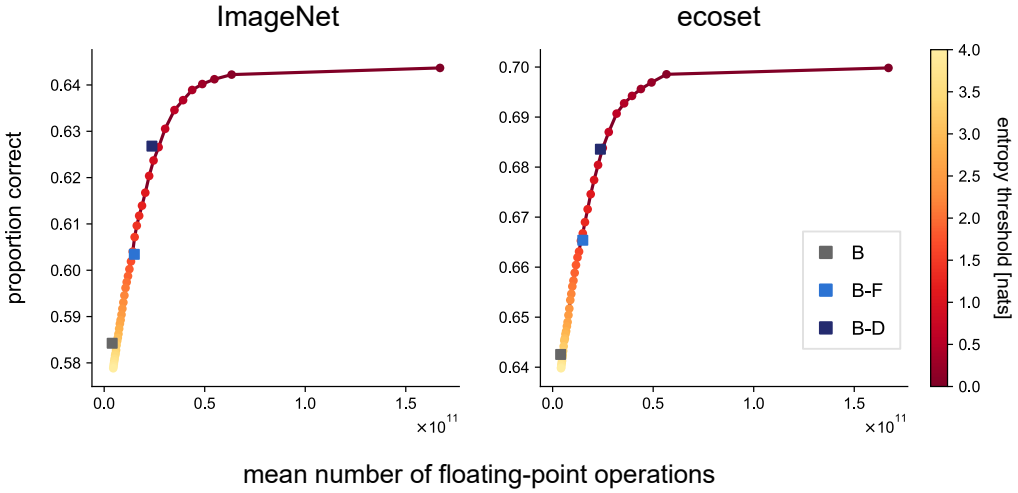

Fig 3. Relationship between computational cost and performance for feedforward and recurrent models. The recurrent models are assessed using a range of entropy thresholds, with the computational cost corresponding to the mean number of floating-point operations used across the test set to reach the given entropy threshold. The computational cost for feedforward models is the number of floating-point operations in a single pass through the model. In all cases, performance is assessed based on held-out data.

Across both datasets only one significant difference in performance was found between recurrent and feedforward models. This difference was the between B and BL in ImageNet, which achieved 58.42% and 57.71%, respectively, a difference of 0.70% ($p < 0.001$). This comparison matches a pass through B to the initial feedforward pass through BL. BL appears to slightly compromise its performance on the initial feedforward pass to support later gains through recurrence. All other differences between BL and feedforward networks were even smaller and not significant, ranging between -0.37% and +0.32%, relative to the performance of BL. B-K was excluded from this analysis because it had worse performance than the baseline feedforward model (possibly due to overfitting).

These results suggest that recurrent models perform similarly to feedforward models when matching the number of floating-point operations. This is surprising given that recurrent networks operate under the additional constraint of having to use their weights across multiple time steps, which does not apply to feedforward networks. We may have expected the operations learned by recurrent networks to be less specialised and less efficient with regards to performance achieved at a given computational cost. Instead, we found that the computational efficiency of recurrent and feedforward networks are well matched. The graceful degradation of performance of recurrent models when the computational cost is limited may depend on training with a loss function that rewards rapid convergence to an accurate output (see Methods).

Overall our results suggest that we can use a single recurrent network to span the space of SATs covered by multiple feedforward networks. Furthermore, using the same network we can achieve a higher performance than all of the parameter-matched feedforward networks by running more recurrent computations.

Network reaction times predict human recognition uncertainty

Recurrent connections endow a model with temporal dynamics. If the recurrent computations in a model match those of the human brain during object recognition, then model behaviour should be predictive of human behaviour. For example, images that require the model to perform more extended recurrent computations for accurate recognition should be more challenging also for humans.

To test this hypothesis we used data from an object categorisation task where humans had to categorise 1,500 greyscale images as animate or inanimate [40]. For each image we calculated the proportion of trials in which the image was classified correctly across human participants. Some images were more consistently recognised by humans (whether accurate or inaccurate) than others. Our goal was to quantify the extent to which images more consistently recognised by humans were more rapidly recognised by the models.

We computed a decision uncertainty index D based on the proportion correct, PC, across humans. D was defined as $0.5 - |0.5 - PC|$. This metric is largest when humans are most inconsistent in their decision making (if $PC = 0.5$ then $D = 0.5$), and it is smallest when all decisions across trials are the same (if $PC = 1.0$ or $PC = 0.0$ then $D = 0.0$).

We fitted ImageNet and ecoset models to these human data and tested the fitted models using cross-validation across images. Network reaction times were extracted by training an additional readout for the animacy discrimination task and fitting an entropy threshold to maximise the correlation with human uncertainty (see Methods). We then tested the fitted models by predicting human uncertainty for different images in crossvalidation (using Spearman correlation to measure prediction accuracy). As a control, we ran the fitting procedure using a network with randomly initialised weights.

Model predictions could rely on category mean decision uncertainty to explain the human data. To exclude this possibility we shuffled the images within each category before fitting the entropy thresholds and recomputing the network reaction times. This shuffling procedure was repeated 100 times.

Results show that reaction times obtained from both ImageNet and ecoset trained networks significantly predicted human decision uncertainty. Furthermore, both trained networks predicted human decision uncertainty better than a randomly initialised network that was fitted using the same procedure (two-tailed paired permutation test, $p < 0.01$) and when images were shuffled within categories (Fig. 4). There was no significant difference between the correlation obtained for the ecoset- and ImageNet-trained networks (two-tailed paired permutation test, $p = 0.40$). Overall, images for which our recurrent networks took longer to converge were less consistently recognised by humans.

Learned recurrent connectivity resembles that of primary visual cortex

To understand the types of computations being performed by recurrent networks and how they relate to our understanding of biological vision, we conducted an exploratory analysis of the learned recurrent connectivity. We focus on the recurrent connectivity in the first layer of the network. This has the benefit that weight templates are easier to interpret in lower than in higher layers of networks. In addition, recurrent processing in biological vision is arguably best understood in lower-level visual areas, which correspond to early model layers.

Because the number of recurrent lateral connections in the model's first layer is large (over 450,000 connections), we use a technique similar to that of Linsley et al. [30] to constrain the analysis. We use principal components analysis (PCA) to decompose the

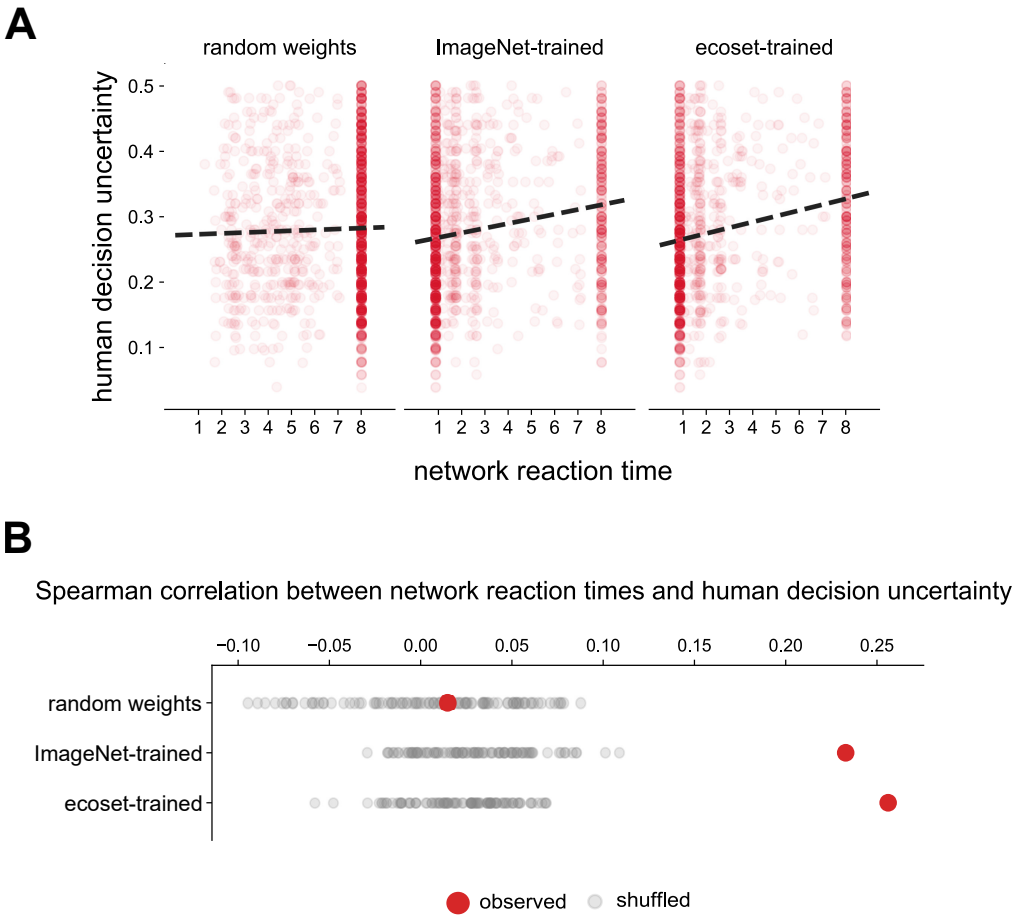

Fig 4. Model reaction times are longer for images that humans are uncertain about. (A) Scatter plot of network reaction times against network decision uncertainty. For each network, a sigmoid animacy readout was trained to maximise accuracy and an entropy threshold fitted so that network reaction times best predicted human uncertainty ratings. Results shown are for images not used in fitting the models or the entropy threshold (cross-validation). (B) Spearman correlations between network reaction times and human decision uncertainty (red) alongside correlations obtained when images were randomly shuffled within categories before fitting network reaction times (grey).

lateral-weight templates into orthogonal components (see Methods). We then explore these lateral-weight components, and the bottom-up features they connect, to compare the lateral connectivity with that of primary visual cortex (Fig. 5).

We focus on the first five principal components of the lateral-weight templates of BL, trained on ImageNet. These weight components capture approximately 43% of the variance across all recurrent weights in the first layer of the ImageNet trained network (Fig. 5). All five components are interpretable: inhibition/excitation (component 1), vertical antagonism (component 2), centre-surround antagonism (component 3), horizontal antagonism (component 4), and perpendicular antagonism (component 5).

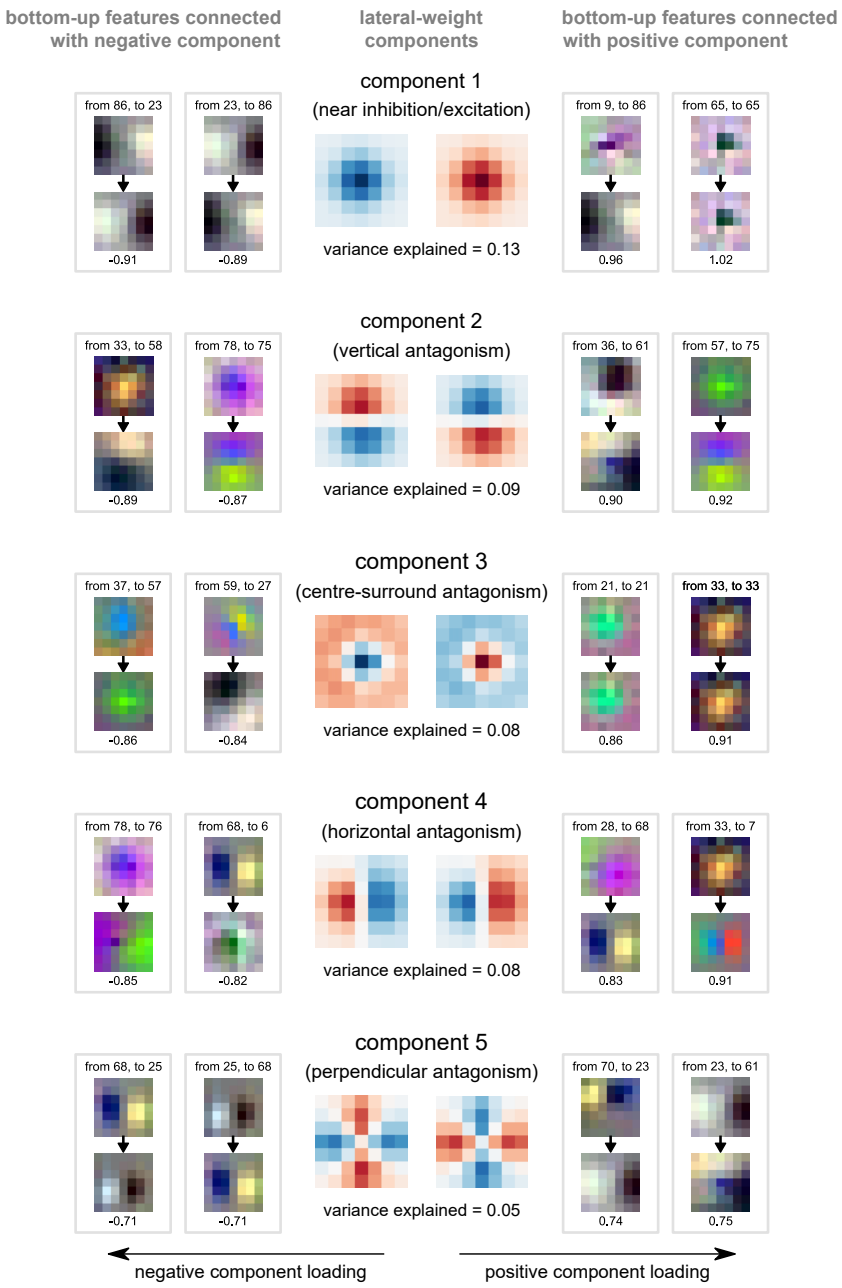

Fig 5. Lateral-weight components for layer 1 of an RCNN trained on ImageNet. Every feature is laterally connected to each other feature via a local lateral-weight pattern. We used principal component analysis to summarise the lateral weight patterns. The top five lateral-weight principal components are shown in both their positive (centre right) and negative forms (centre left). Blue shading corresponds to negative values and red to positive. The proportion of variance explained is given beneath each lateral-weight component. Bottom-up feature maps connected by lateral weights with the strongest positive (right) and negative loadings (left) on the weight component are shown alongside. Arrows between bottom-up features indicate the direction of the connection and the loading is given underneath each pair of bottom-up features.

Local inhibition/excitation

The lateral-weight component explaining the most variance in the network corresponds to local inhibition and excitation. Near inhibitory connections could be used to generate sparse representations, similar to visual cortex [41].

To further understand how inhibitory connectivity relates to the properties of bottom-up features, we correlated the bottom-up weight templates of features connected by lateral-weights with strong negative loadings on the first component (defined as the lowest percentile of loadings on the component). We found a median correlation of -0.16 between bottom-up features with local inhibitory recurrent connections. This value significantly differed from zero (Wilcoxon signed-rank test, $p < 0.001$), suggesting that dissimilar features inhibit each other in the network, possibly increasing the sparsity of the representation.

Centre-surround antagonism

Centre-surround antagonism is a well-studied feature of biological vision and is most often seen in the context of near excitation and far inhibition. In these arrangements, a unit will be excited if a preferred stimulus is detected in the centre and suppressed if the preferred stimulus appears in the surround.

In the lateral-weights of the network, we see centre-surround antagonism in both the classical arrangement of near excitation and far inhibition and the non-classical arrangement of near inhibition and far excitation (Fig. 5, component 3). However, features connected with non-classical centre-surround connectivity (highest percentile of loadings on component 3) had a median negative correlation of -0.04, which significantly differed from zero (Wilcoxon signed-rank test, $p = 0.003$). Non-classical centre-surround connectivity in the network, thus, could still lead to reduced responses if a preferred stimulus is detected in the surround, like classic centre-surround connectivity, but due to reduced excitation rather than increased inhibition.

Cardinal antagonism

Vertical and horizontal antagonism are also observed in the network (Fig. 5, component 2 and component 4). We collectively refer to vertical and horizontal antagonistic weight templates as cardinal antagonism. This type of interaction leads to excitation if a feature is detected to one side of a unit and leads to inhibition if that same feature is detected on the opposite side. This type of asymmetry could be useful for developing border ownership cells [42], which have varying levels of response, depending on which side of an edge corresponds to an object or background surface.

A unit that detects an edge between two surfaces could show properties of border ownership if it receives recurrent input carrying information about the spatial extent of the two surfaces meeting at the edge. We see examples of this type of connectivity in the network. For instance, feature 76 is sensitive to purple-green edges and it receives input from feature 78, which prefers diffuse purple features (Fig. 5, component 4). The recurrent connectivity between them is cardinally antagonistic such that the unit detecting the purple-green edge is only excited if a diffuse purple feature is detected on the purple side of the edge.

Perpendicular antagonism

Perpendicular antagonism is observed in this network where there are excitatory recurrent connections along one orientation and inhibitory recurrent connections along the orthogonal orientation (in both directions). This type of connectivity is consistent with association fields that could support contour integration [43].

Studying the feature maps that most heavily load on these components, we find that feature maps that detect gradients in similar orientations with edges in phase have collinear inhibition and orthogonal excitation (Fig. 5, component 5). In comparison, we see collinear excitation and orthogonal inhibition when feature maps are detecting gradients that have similar orientations but opposite phases.

Collinear excitation may be expected between features detecting gradients in similar directions because the presence of such features is consistent with a continuous contour. However, collinear inhibition is consistent with end-stopping behaviour observed in complex cells of visual cortex [44]. In this case, cells were observed that have suppressed firing rates if edges extend beyond the classical receptive field of the cell.

Overall, the patterns of connectivity learned by recurrent convolutional networks appear to be consistent with what is known about the connectivity of primary visual cortex.

Discussion

Our results show that recurrent architectures can outperform parameter-matched feedforward controls on a naturalistic visual recognition task. In addition to superior performance, recurrent networks more closely resemble biological visual systems in both structure and function. Structurally, biological visual systems exhibit ample recurrent signal flow. Functionally, they exhibit greater robustness and flexibility than current feedforward neural network models.

An important functional feature of our recurrent model is the flexibility to trade off speed and accuracy, which the model shares with biological visual systems. A single recurrent network can span the space of speed-accuracy trade-offs covered by multiple feedforward models. One might have expected that there is a significant cost to the added flexibility of recurrent computation. Among the models considered here, however, we find only marginal costs to performance of recurrent models when the computational budget is matched.

Recurrent models not only have the functional benefit of flexible speed-accuracy trading, shared with human vision, but they also predicted human behaviour: their reaction times were longer for images less consistently recognised by humans.

The performance of recurrent models, relative to feedforward, is consistent with previous work using small-scale machine learning tasks [26, 28]. However, it contrasts with more recent results suggesting that specialised recurrent architectures, in the form of reciprocally gated cells, are required for recurrent networks to outperform their feedforward counterparts in naturalistic visual recognition tasks [29]. One potential explanation of these diverging results is the scale of the feedforward control models relative to the recurrent networks. In the experiments described here, the recurrent networks had approximately 72-100% of the parameters of the feedforward control models. In comparison, the baseline recurrent models "Vanilla RNN" (similar to BL) had approximately 39% and 45% of the parameters of the feedforward control models ("FF Deeper" and "FF Wider", respectively) in [29]. While reciprocally gated cells clearly produce better task performance, this difference in the number of parameters could explain why our recurrent convolutional networks (without the addition of gating) were able to outperform the parameter-matched feedforward models. It also highlights the difficulty of defining appropriate feedforward control models. Here, we take the approach of matching the number of parameters in feedforward and recurrent models. We also consider the performance of the networks at matched computational costs.

We showed additional practical benefits for recurrent networks by borrowing two ideas from the literature on biological decision making: threshold-based decision making [33] and speed-accuracy trade-offs [34]. First, using a fixed posterior-entropy

threshold, networks were able take longer to recognise more difficult images. Second, by varying the posterior-entropy threshold, networks could change their required confidence, trading off accuracy for speed. These behaviours enable economical object recognition, only spending the time (and energy) required by the given task or situation.

This type of flexible behaviour is useful in biological and artificial object recognition, where both time and computational resources are often limited. RCNNs for vision may be useful in artificial intelligence technologies, particularly those operating under resource constraints (e.g. [35, 45, 46]).

Our finding that RCNNs predicted human uncertainty for individual images suggests an interesting direction for future models of biological decision making. RCNNs could provide a unified basis for predicting image-specific distributions of errors and reaction times. This would complement previous work on recurrent processing in the decision-making literature.

Recurrent processing in human decision-making is typically viewed as the accumulation of independent noisy samples of some underlying variable. This leads to a stochastic drift toward a decision bound, depending on the noise of the sample [33]. In real-world perceptual decisions, however, evidence may vary across time due to non-random processes. Beyond evidence accumulation, recurrent processing might lead to different decisions being favoured at different points in time. This could lead to more exotic predictions that are not easily generated by drift diffusion models (such as class A being favoured early in the trial, class B being preferred in the middle and class A being preferred again at the end).

In addition, our exploratory analysis of recurrent connectivity in the network shows evidence that RCNNs may learn recurrent computations resembling those in biological vision. There is evidence of centre-surround computations as well as connectivity that could help to support properties such as sparse representations [41], border ownership [42], contour integration [43], and end-stopping [44]. These analyses of recurrent connectivity offer a promising starting point for understanding recurrent computations in artificial visual systems and should be followed up by a detailed analysis of activity patterns in the models.

The observed lateral connections in our networks trained for object recognition also show a resemblance to the lateral connections of networks trained for contour integration tasks [30]. Given the different nature of these tasks, the similarity in lateral connectivity is surprising. This leads to the interesting hypothesis that there might be a subset of lateral computations that are useful across a range of visual tasks, at least in low-level visual areas. This would be consistent with the fact that a large range of objectives can be optimised to obtain simple-cell like features in feedforward templates that are observed in low-level visual areas. Such objectives include image classification performance [47], predictive coding [48], temporal stability [49], and sparsity [41].

In general, the work described here adds to a growing body of research on RCNNs as models of object recognition [25–29, 31, 32]. These models provide us with a white box, a vision system that can be observed from input to behavioural response. Understanding how these models perform object recognition might reveal the role of recurrent processing in biological vision.

Methods

Deep neural network implementation

All deep neural networks in these experiments were implemented using TensorFlow [50]. The different architectures used are specified in detail in Table 2.

Artificial recurrent neural networks are typically implemented with feedforward

Table 2. Specification of network architectures

Model	B	B-K	B-F	B-D	BL
Block 1	F = 96, K = 7	F = 96, K = 11	F = 192, K = 7	F = 96, K = 7 F = 96, K = 7	(F = 96, K = 7) × 2
Pool 1	2 × 2 max pooling				
Block 2	F = 128, K = 5	F = 128, K = 7	F = 256, K = 5	F = 128, K = 5 F = 128, K = 5	(F = 128, K = 5) × 2
Pool 2	2 × 2 max pooling				
Block 3	F = 192, K = 3	F = 192, K = 5	F = 384, K = 3	F = 192, K = 3 F = 192, K = 3	(F = 192, K = 3) × 2
Pool 3	2 × 2 max pooling				
Block 4	F = 256, K = 3	F = 256, K = 5	F = 512, K = 3	F = 256, K = 3 F = 256, K = 3	(F = 256, K = 3) × 2
Pool 4	2 × 2 max pooling				
Block 5	F = 512, K = 3	F = 512, K = 5	F = 1024, K = 3	F = 512, K = 3 F = 512, K = 3	(F = 512, K = 3) × 2
Pool 5	2 × 2 max pooling				
Block 6	F = 1024, K = 3	F = 1024, K = 5	F = 2048, K = 3	F = 1024, K = 3 F = 1024, K = 3	(F = 1024, K = 3) × 2
Pool 6	2 × 2 max pooling				
Block 7	F = 2048, K = 1	F = 2048, K = 3	F = 4096, K = 1	F = 2048, K = 1 F = 2048, K = 1	(F = 2048, K = 1) × 2
Readout	global average pooling 565 or 1000 category readout				
Parameters	**11.0 million**	**39.8 million**	**40.0 million**	**28.9 million**	**28.9 million**

Each row in the table represents a convolutional layer. F specifies the number of feature maps in the layer and K represents the height and width dimension of the convolutional kernel. For BL, "(...) × 2" indicates that the same size convolutional kernel is applied twice, once to the bottom-up input (from the layer below)and once to the lateral input (from the same layer). All convolutions are applied with 1 × 1 stride and all max pooling is applied with 2 × 2 stride. The number of parameters are calculated for ImageNet models, ecoset models have slightly fewer parameters for the readout due to smaller number of categories in ecoset.

connections taking no time and recurrent connections taking a single time step, we refer to this as "engineering" time. In comparison, all connections in biological neural networks should incur some form of time delay. A more biologically realistic implementation of a recurrent network may have every form of connection taking a single time step [25, 29]. However, these two implementations produce equivalent computations in BL networks if we do not consider computations that either: (1) occur prior to the first feedforward sweep, or (2) cannot reach the readout before the final time step is reached (Fig. 6). As such, we use "engineering" time for recurrent networks in these experiments. Therefore, time in recurrent networks is defined as the number of

complete feedforward sweeps that have occurred.

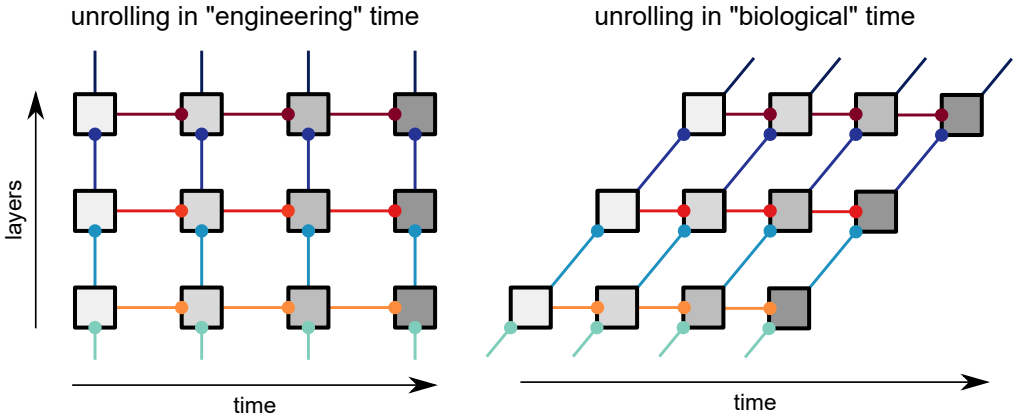

Fig 6. Network unrolling through time. Unrolling is shown for engineering time (left) and biological time (right). Each box represents a layer and the shading corresponds to its label in engineering time. Connections with the same colour represent shared parameters.

We define the output from a standard convolutional layer at layer n on time step t as

$$H_{t,n} = F(W_n^b * H_{t,n-1} + b_n) \quad (1)$$

Where W_n^b are the bottom-up convolutional weights for the layer and b_n are the biases. The convolution operation is represented as $*$. All operations applied after the convolution are represented by the function F. These operations include batch-normalisation [51] and rectified linear units in that order.

For a recurrent BL layer, the output is defined as

$$H_{t,n} = F(W_n^b * H_{t,n-1} + W_n^l * H_{t-1,n} + b_n) \quad (2)$$

Where W_n^l are the lateral recurrent weights.

For the recurrent networks, batch-normalisation is applied independently across time. Whilst this means that the networks are not truly recurrent due to unique normalisation parameters at each time step, this does not affect arguments related to parametric efficiency, as the numbers of parameters added by batch-normalisation at each time-step are negligible compared to the overall scale of the network. Approximately, 60,000 parameters are added across time due to batch-normalisation compared to 28.9 million parameters for the network as a whole.

In addition, we tested whether the use of independent batch-normalisation across time confers an additional performance advantage to recurrent networks by training B-D and BL on ImageNet without batch-normalisation. In this case, networks were trained using the same procedure but for only 25 epochs to prevent overfitting (as the removal of batch-normalisation reduces stochasticity in training). B-D and BL achieved a validation accuracy of 52.5% and 58.6%, respectively. This suggests that independent batch-normalisation across time does not explain the performance difference between feedforward and recurrent networks and even has a more beneficial effect for feedforward networks than recurrent networks (approximately 10 percentage point increase for B-D compared to a 6 percentage point increase for BL).

Before passing the images to the network, a number of pre-processing steps were applied. First, a crop was taken from the image, which was resized to 128×128 pixels. During testing and validation, a centre crop was taken from the image. During training, a random crop was taken covering at least one third of the image area. Further data

augmentation was also applied in training, this included random left-right flips, and small distortions to the brightness, saturation and contrast of the image. Finally, the pixel values in the image were scaled from the range [0, 1] to be in the range [-1, 1].

The networks were trained for a total of 90 epochs with a batch size of 100. The cross-entropy between the softmax of the network category readout and the labels was used as the training loss. For recurrent networks, we calculate the cross-entropy on each time step and average this across time. Adam [52] was used for optimisation with a learning rate of 0.005 and epsilon parameter 0.1. L2-regularisation was applied throughout training with a coefficient of 10^{-6}.

The code for models and weights for pre-trained networks are made available at github.com/cjspoerer/rcnn-sat.

Defining accuracy in recurrent networks

As recurrent networks are unrolled across time, they have readouts at multiple time steps. This means that we must map from many readouts for a single image to one prediction. This leads to some ambiguity about how to produce predictions from recurrent networks for object recognition. Therefore, we conducted initial analyses to determine how to generate predictions from recurrent networks in the experiments described here.

One decision is how to select the time step to readout from the network, which we refer to as the network's reaction time. A fixed time step could be chosen. For example, the readout could always be taken at the final time step that the recurrent model runs until. We refer to this as time-based accuracy.

Alternatively, we could select the readout to use based on when the model reaches some threshold. For example, the prediction is taken from the network once a certain level of confidence is reached. This confidence level could be defined by the entropy of the readout distribution where a lower entropy corresponds to a higher confidence. If the required confidence level is never reached then the final time step is selected as the reaction time. This is referred to as threshold-based accuracy. It should be noted that threshold-based accuracy can be implemented in recurrent networks using dynamic computational graphs that only execute up to the desired threshold. However, for our analyses we simply measure the time that it takes for the network to achieve a given level of entropy.

Once the decision time has been selected, we need to decide how to reduce the readout distribution across time. One method is to generate the prediction based solely on the readout at the network reaction time. We refer to this as the instantaneous readout. A second method is to generate the prediction from the cumulative readout up to the decision time, allowing the network's predictions to be explicitly aggregated across time.

These different methods were compared using held-out data (Fig. 7). For ecoset the held-out data corresponds to the test set and for ImageNet this corresponds to the validation set, as the test set is not publicly available.

For time-based methods, we see that the accuracy of the readout tends to increase across time. However, there is some drop-off in performance at later time steps if the instantaneous readout is used. One explanation for this pattern is that, by training the network to produce a readout at each time step, the network is encouraged to produce accurate predictions more quickly at the cost of higher accuracy at later time steps.

If a cumulative readout is used then accuracy improves more steadily across time, which is consistent with the smoothing effects expected from a cumulative readout. However, cumulative readouts produce a higher overall level of accuracy than instantaneous readouts. This suggests there is some benefit of accumulating evidence

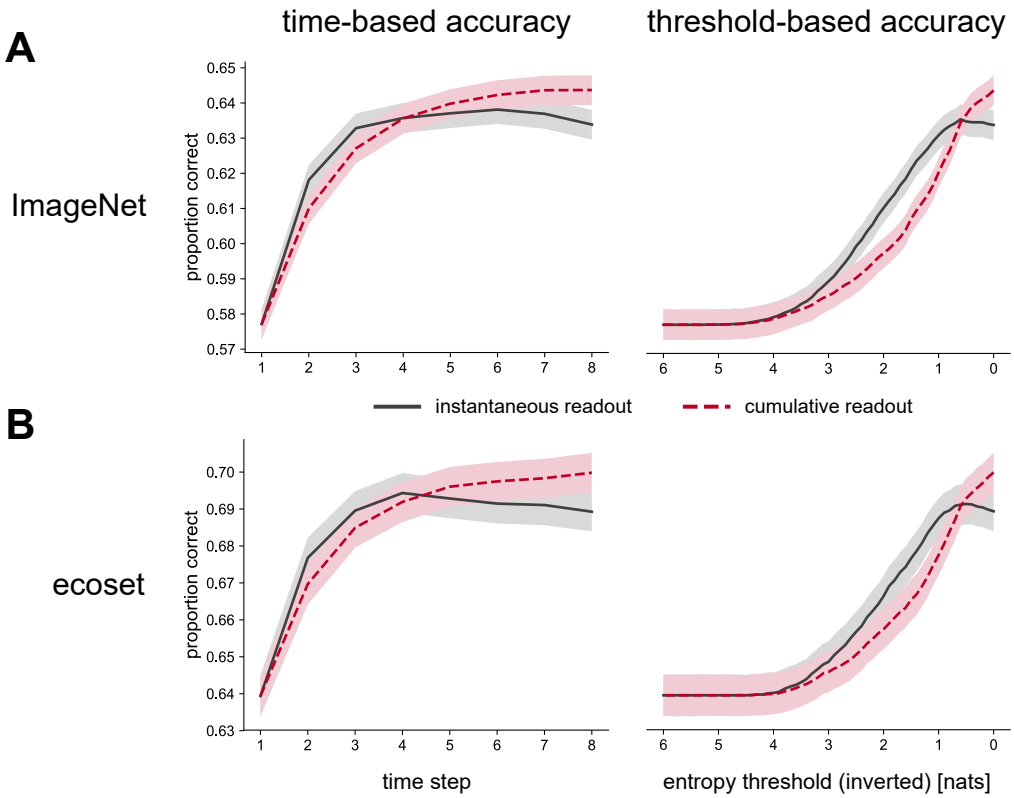

Fig 7. Task performance using varied definitions of predictions for recurrent models. Accuracies are given for models trained on (A) ImageNet and (B) ecoset using both time-based (left) and threshold-based (right) methods. Accuracies obtained from instantaneous readouts are shown with solid lines and results from cumulative readouts are shown with dashed lines. Shaded areas represent 95% confidence intervals obtained through bootstrap resampling.

across time for the performance of the network, even though the predictions themselves are not independent across time.

Similar results are seen when threshold-based accuracies are used. This reflects the fact that decreasing the entropy threshold will naturally lead to later time steps being increasingly utilised. Threshold-based accuracies also show a decrease in accuracy for instantaneous readouts at the lowest entropy levels. This is again due to worse performance at later time steps but also highlights an assumption of threshold-based accuracies that letting the network run for longer, to obtain higher confidence levels, will generate better predictions.

As a result of these analyses, all reported accuracies for recurrent networks refer to predictions based on cumulative readouts as these tend to produce the best performance.

Fitting network reaction times to human decision uncertainty

A cross-validated procedure was used to fit RCNNs to human decision uncertainty data from Eberhardt et al. [40]. This data consists of human animacy judgements for 1,500 different images. A total of at least 50 unique responses were recorded for each image.

Firstly, images were split into training and tests sets, 10-fold cross-validation was used such that there were 1350 training images and 150 testing images in each fold. Using the training images, a fully-connected layer was trained to produce a readout, y_t,

predicting the label at each time step $t \in \{1, ..., 8\}$. The readout was defined as follows

$$y_t = \sigma(\alpha y_{t-1} + W H_{t,N} + b) \quad (3)$$

Where $H_{t,N}$ are the flattened activations from the final convolutional layer at each time step, α is a recurrent parameter that allows evidence to be accumulated across time, W are the weights for the linear readout, b are biases and σ is the sigmoid non-linearity. The initial readout state y_0 was defined to neutral, such that $y_0 = 0.5$.

The readout was optimised using batch gradient descent with Adam. The learning rate was set to 0.001 and the readout was trained for 1000 iterations.

The readout for each of the images, y_t, was then upsampled by linearly interpolating across between all timesteps, excluding the initial state y_0. This increased the fidelity of the network readout from from the 8 original time steps to 800 samples.

Entropy thresholds were used to extract reaction times for each image using the linearly interpolated readout. The entropy threshold was set using grid search to maximise the correlation between network reaction times and human decision uncertainty for the training set. Using the fitted readout and thresholds, reaction times were extracted for the testing data. This procedure was repeated using 10-fold cross-validation such that a reaction time was obtained for each image after fitting to independent data.

As a control we also extracted reaction times when individual images where randomly shuffled within the same category and train/test split. After every shuffle, the cross-validated threshold fitting procedure was rerun and reaction times were extracted for each image. This shuffling procedure was repeated 100 times for each trained network.

Extracting lateral-weight components

We analyse the lateral connectivity of the network by decomposing the lateral weights in the network into lateral-weight components. To do this, we focus of the 7×7 weight templates that connect each of the feature maps within the first layer of the network. There are 96^2 weight templates in total connecting every feature map to each other in both directions (including self-connections from a feature map to itself). We focus on the first layer of the network as the corresponding bottom-up weights are easier to interpret and recurrence is arguably best understood in early regions of the visual system (corresponding to early layers of the network).

Firstly, the weight templates are normalised such that the vector of the flattened weight template has unit length. After normalisation, the lateral weights are processed using principal components analysis (PCA) where each weight template is considered as an individual sample. The first five components resulting from the PCA are used as the lateral-weight components for the analysis.

Acknowledgments

We thank Sven Eberhardt, Jonah Cader and Thomas Serre for sharing their behavioural data. This project has received funding from the European Union's Horizon 2020 Programme for Research and Innovation under the Specific Grant Agreement No. 720270 and 785907 (Human Brain Project SGA1 and SGA2), the NVIDIA GPU Grant Program, and the German Science Foundation (DFG grant 'DynaVision').

References

1. Wallis G, Rolls ET. Invariant face and object recognition in the visual system. Progress in neurobiology. 1997;51(2):167–194.

2. Riesenhuber M, Poggio T. Hierarchical models of object recognition in cortex. Nature neuroscience. 1999;2:1019–1025.

3. Serre T, Wolf L, Bileschi S, Riesenhuber M, Poggio T. Robust object recognition with cortex-like mechanisms. IEEE Transactions on Pattern Analysis & Machine Intelligence. 2007;29(3):411–426.

4. Kriegeskorte N. Deep Neural Networks: A New Framework for Modeling Biological Vision and Brain Information Processing. Annual Review of Vision Science. 2015;1(1):417–446. doi:10.1146/annurev-vision-082114-035447.

5. Yamins DL, DiCarlo JJ. Using goal-driven deep learning models to understand sensory cortex. Nature neuroscience. 2016;19:356–365.

6. Kietzmann TC, McClure P, Kriegeskorte N. Deep Neural Networks in Computational Neuroscience; 2019. Available from: `http://oxfordre.com/neuroscience/view/10.1093/acrefore/9780190264086.001.0001/acrefore-9780190264086-e-46`.

7. Cadieu CF, Hong H, Yamins DLK, Pinto N, Ardila D, Solomon EA, et al. Deep Neural Networks Rival the Representation of Primate IT Cortex for Core Visual Object Recognition. PLOS Computational Biology. 2014;10(12):1–18. doi:10.1371/journal.pcbi.1003963.

8. Khaligh-Razavi SM, Kriegeskorte N. Deep Supervised, but Not Unsupervised, Models May Explain IT Cortical Representation. PLOS Computational Biology. 2014;10(11):1–29. doi:10.1371/journal.pcbi.1003915.

9. Yamins DLK, Hong H, Cadieu CF, Solomon EA, Seibert D, DiCarlo JJ. Performance-optimized hierarchical models predict neural responses in higher visual cortex. Proceedings of the National Academy of Sciences. 2014;111(23):8619–8624. doi:10.1073/pnas.1403112111.

10. Güçlü U, van Gerven MA. Deep Neural Networks Reveal a Gradient in the Complexity of Neural Representations across the Ventral Stream. Journal of Neuroscience. 2015;35(27):10005–10014. doi:10.1523/JNEUROSCI.5023-14.2015.

11. Rajalingham R, Schmidt K, DiCarlo JJ. Comparison of Object Recognition Behavior in Human and Monkey. Journal of Neuroscience. 2015;35(35):12127–12136. doi:10.1523/JNEUROSCI.0573-15.2015.

12. Rajalingham R, Issa EB, Bashivan P, Kar K, Schmidt K, DiCarlo JJ. Large-Scale, High-Resolution Comparison of the Core Visual Object Recognition Behavior of Humans, Monkeys, and State-of-the-Art Deep Artificial Neural Networks. Journal of Neuroscience. 2018;38(33):7255–7269. doi:10.1523/JNEUROSCI.0388-18.2018.

13. Geirhos R, Rubisch P, Michaelis C, Bethge M, Wichmann FA, Brendel W. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In: International Conference on Learning Representations; 2019.Available from: `https://openreview.net/forum?id=Bygh9j09KX`.

14. DiCarlo JJ, Zoccolan D, Rust NC. How Does the Brain Solve Visual Object Recognition? Neuron. 2012;73(3):415–434. doi:https://doi.org/10.1016/j.neuron.2012.01.010.

15. Felleman DJ, Van Essen DC. Distributed Hierarchical Processing in the Primate Cerebral Cortex. Cerebral Cortex. 1991;1(1):1–47.

16. Sporns O, Zwi JD. The small world of the cerebral cortex. Neuroinformatics. 2004;2:145–162.

17. Markov NT, Vezoli J, Chameau P, Falchier A, Quilodran R, Huissoud C, et al. Anatomy of hierarchy: feedforward and feedback pathways in macaque visual cortex. Journal of Comparative Neurology. 2014;522(1):225–259.

18. Sugase Y, Yamane S, Ueno S, Kawano K. Global and fine information coded by single neurons in the temporal visual cortex. Nature. 1999;400(6747):869–873.

19. Brincat SL, Connor CE. Dynamic Shape Synthesis in Posterior Inferotemporal Cortex. Neuron. 2006;49(1):17–24. doi:https://doi.org/10.1016/j.neuron.2005.11.026.

20. Freiwald WA, Tsao DY. Functional Compartmentalization and Viewpoint Generalization Within the Macaque Face-Processing System. Science. 2010;330(6005):845–851. doi:10.1126/science.1194908.

21. Clarke A, Devereux BJ, Randall B, Tyler LK. Predicting the Time Course of Individual Objects with MEG. Cerebral Cortex. 2015;25(10):3602–3612. doi:10.1093/cercor/bhu203.

22. Kietzmann TC, Ehinger BV, Porada D, Engel AK, König P. Extensive training leads to temporal and spatial shifts of cortical activity underlying visual category selectivity. NeuroImage. 2016;134:22–34. doi:https://doi.org/10.1016/j.neuroimage.2016.03.066.

23. Kietzmann TC, Gert AL, Tong F, König P. Representational Dynamics of Facial Viewpoint Encoding. Journal of Cognitive Neuroscience. 2017;29(4):637–651. doi:10.1162/jocn_a_01070.

24. Kar K, Kubilius J, Schmidt KM, Issa EB, DiCarlo JJ. Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior. Nature Neuroscience. 2019;.

25. Kietzmann TC, Spoerer CJ, Sörensen L, Cichy RM, Hauk O, Kriegeskorte N. Recurrence required to capture the dynamic computations of the human ventral visual stream. arXiv preprint arXiv:190305946. 2019;.

26. Liang M, Hu X. Recurrent convolutional neural network for object recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA, USA; 2015. p. 3367–3375.

27. Liao Q, Poggio T. Bridging the gaps between residual learning, recurrent neural networks and visual cortex. arXiv preprint arXiv:160403640. 2016;.

28. Spoerer CJ, McClure P, Kriegeskorte N. Recurrent Convolutional Neural Networks: A Better Model of Biological Object Recognition. Frontiers in Psychology. 2017;8:1551. doi:10.3389/fpsyg.2017.01551.

29. Nayebi A, Bear D, Kubilius J, Kar K, Ganguli S, Sussillo D, et al. Task-Driven convolutional recurrent models of the visual system. In: Advances in Neural Information Processing Systems; 2018. p. 5290–5301.

30. Linsley D, Kim J, Veerabadran V, Windolf C, Serre T. Learning long-range spatial dependencies with horizontal gated recurrent units. In: Advances in Neural Information Processing Systems; 2018. p. 152–164.

31. Kubilius J, Schrimpf M, Nayebi A, Bear D, Yamins DL, DiCarlo JJ. CORnet: Modeling the Neural Mechanisms of Core Object Recognition. bioRxiv. 2018;.

32. Tang H, Schrimpf M, Lotter W, Moerman C, Paredes A, Ortega Caro J, et al. Recurrent computations for visual pattern completion. Proceedings of the National Academy of Sciences. 2018;115(35):8835–8840. doi:10.1073/pnas.1719397115.

33. Gold JI, Shadlen MN. The Neural Basis of Decision Making. Annual Review of Neuroscience. 2007;30(1):535–574. doi:10.1146/annurev.neuro.29.051605.113038.

34. Wickelgren WA. Speed-accuracy tradeoff and information processing dynamics. Acta Psychologica. 1977;41(1):67–85. doi:https://doi.org/10.1016/0001-6918(77)90012-9.

35. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:170404861. 2017;.

36. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, et al. Imagenet large scale visual recognition challenge. International Journal of Computer Vision. 2015;115(3):211–252.

37. Mehrer J, Kietzmann TC, Kriegeskorte N. Deep neural networks trained on ecologically relevant categories better explain human IT. In: Conference on Cognitive Computational Neuroscience. New York, NY, USA; 2017.Available from: `https://www2.securecms.com/CCNeuro/docs-0/5927d79368ed3feb338a2577.pdf`.

38. McNemar Q. Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika. 1947;12(2):153–157.

39. Dietterich TG. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. Neural Computation. 1998;10(7):1895–1923. doi:10.1162/089976698300017197.

40. Eberhardt S, Cader JG, Serre T. How Deep is the Feature Analysis underlying Rapid Visual Categorization? In: Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R, editors. Advances in Neural Information Processing Systems 29. Curran Associates, Inc.; 2016. p. 1100–1108.

41. Olshausen BA, Field DJ. Sparse coding of sensory inputs. Current Opinion in Neurobiology. 2004;14(4):481 – 487. doi:https://doi.org/10.1016/j.conb.2004.07.007.

42. Zhou H, Friedman HS, von der Heydt R. Coding of Border Ownership in Monkey Visual Cortex. Journal of Neuroscience. 2000;20(17):6594–6611. doi:10.1523/JNEUROSCI.20-17-06594.2000.

43. Field DJ, Hayes A, Hess RF. Contour integration by the human visual system: Evidence for a local "association field". Vision Research. 1993;33(2):173–193. doi:https://doi.org/10.1016/0042-6989(93)90156-Q.

44. Hubel DH. Exploration of the primary visual cortex, 1955–78. Nature. 1982;299:515–524.

45. Iandola FN, Han S, Moskewicz MW, Ashraf K, Dally WJ, Keutzer K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and¡ 0.5 MB model size. arXiv preprint arXiv:160207360. 2016;.

46. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC. Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018. p. 4510–4520.

47. Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ, editors. Advances in Neural Information Processing Systems 25. South Lake Tahoe, NV, USA: Curran Associates, Inc.; 2012. p. 1097–1105.

48. Rao RP, Ballard DH. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. Nature neuroscience. 1999;2:79–87.

49. Wiskott L, Sejnowski TJ. Slow Feature Analysis: Unsupervised Learning of Invariances. Neural Computation. 2002;14(4):715–770. doi:10.1162/089976602317318938.

50. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, et al. TensorFlow: A System for Large-Scale Machine Learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). Savannah, GA: USENIX Association; 2016. p. 265–283. Available from: `https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi`.

51. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:150203167. 2015;.

52. Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980. 2014;.