Recurrent convolutional neural networks: a better model of biological object recognition under occlusion

Courtney J. Spoerer[1], Patrick McClure[1], and Nikolaus Kriegeskorte[1]

[1]Medical Research Council Cognition and Brain Sciences Unit, Cambridge, CB2 7EF, United Kingdom

April 24, 2017

Feedforward neural networks provide the dominant model of how the brain performs visual object recognition. However, these networks lack the lateral and feedback connections, and the resulting recurrent neuronal dynamics, of the ventral visual pathway in the human and nonhuman primate brain. Here we investigate recurrent convolutional neural networks with bottom-up (B), lateral (L), and top-down (T) connections. Combining these types of connections yields four architectures (B, BT, BL, and BLT), which we systematically test and compare. We hypothesized that recurrent dynamics might improve recognition performance in the challenging scenario of partial occlusion. We introduce two novel occluded object recognition tasks to test the efficacy of the models, *digit clutter* (where multiple target digits occlude one another) and *digit debris* (where target digits are occluded by digit fragments). We find that recurrent neural networks outperform feedforward control models (approximately matched in parametric complexity) at recognising objects, both in the absence of occlusion and in all occlusion conditions. Recurrent neural networks are not only more neurobiologically plausible in their architecture; their dynamics also afford superior task performance. This work shows that computer vision can benefit from using recurrent convolutional architectures and suggests that the ubiquitous recurrent connections in biological brains are essential for task performance.

1 Introduction

The primate visual system is highly efficient at object recognition, requiring only brief presentations of the stimulus to perform the task (Potter, 1976; Thorpe et al., 1996; Keysers et al., 2001). Within 150ms of stimulus onset, neurons in inferior temporal cortex (IT) encode object information in a form that is robust to transformations in scale and position (Hung et al., 2005; Isik et al.,

2014), and is predictive of human behavioural responses (Majaj et al., 2015).

This rapid processsing lends support to the idea that invariant object recognition can be explained through a feedforward process (DiCarlo et al., 2012), a claim that has been supported by the recent successes of feedforward neural networks in computer vision (e.g. Krizhevsky et al., 2012) and the usefulness of these networks as models of primate visual processing (Riesenhuber and Poggio, 1999; Serre et al., 2007; Wallis and Rolls, 1997; Yamins et al., 2013, 2014; Khaligh-Razavi and Kriegeskorte, 2014; Güçlü and van Gerven, 2015).

The success of feedforward models of visual object recognition has resulted in feedback processing being underexplored in this domain. However, both anatomical and functional evidence seems to suggest that feedback connections play a role in object recognition. For instance, it is well known that the ventral visual pathway contains similar densities of feedforward and feedback connections (Felleman and Van Essen, 1991; Sporns and Zwi, 2004; Markov et al., 2014), and functional evidence from primate and human electrophysiology experiments show that processing of object information unfolds over time, beyond what would be interpreted as feedforward processing (Sugase et al., 1999; Brincat and Connor, 2006; Freiwald and Tsao, 2010; Carlson et al., 2013; Cichy et al., 2014; Clarke et al., 2015). Some reports of robust object representations, normally attributed to feedforward processing (Isik et al., 2014; Majaj et al., 2015), occur within temporal delays that are consistent with fast local recurrent processing (Wyatte et al., 2014). This suggests that we need to move beyond the standard feedforward model if we are to gain a complete understanding of visual object recognition within the brain.

Fast local recurrent processing is temporally dissociable from attentional effects in frontal and parietal areas, and is thought to be particularly important in recognition of the degraded objects (for a review see Wyatte et al., 2014). In particular, object recognition in the presence of occlusion is thought to engage recurrent processing. This is supported by the finding that recognition under these conditions produces delayed behavioural and neural responses, and recognition can be disrupted by masking, which is thought to interfere with recurrent processing (Johnson and Olshausen, 2005; Tang et al., 2014; Wyatte et al., 2012). Furthermore, scene information can be decoded from areas of early visual cortex that correspond to occluded regions of the visual field (Smith and Muckli, 2010) further supporting the claim that feedback processing is engaged when there is occlusion in the visual input.

Occluded object recognition has been investigated using neural network models in previous work, which found an important role for feedback connections when stimuli were partially occluded (O'Reilly et al., 2013). However, the type of occlusion used in these simulations, and previous experimental work, has involved fading out or deleting parts of images (Wyatte et al., 2012; Smith and Muckli, 2010; Tang et al., 2014). This does not correspond well to vision in natural

environments where occlusion is generated by objects occluding one another. Moreover, deleting parts of objects, as opposed to occluding them, leads to poorer accuracies and differences in early event-related potentials (ERPs) that could indicate different effects on local recurrent processing (Johnson and Olshausen, 2005). Therefore, it is important to investigate the effects of actual object occlusion in neural networks to complement prior work on deletion.

To test the effects of occlusion, we developed a new generative model for occlusion stimuli. The images contain parameterised, computer-generated digits in randomly jittered positions (optionally, the size and orientation can also be randomly varied). The code for generating these images is made available at `https://github.com/cjspoerer/digitclutter`. The task is to correctly identify these digits. Different forms of occlusion are added to these images, including occlusion from non-targets and other targets present in the image, we refer to these as *digit debris* and *digit clutter*, respectively. The first form of occlusion, digit debris, simulates situations where targets are occluded by other objects that are task irrelevant. The second case, digit clutter, simulates occlusion where the objective is to account for the occlusion without suppressing the occluder, which is itself a target. This stimulus set has a number of benefits. Firstly, the underlying task is relatively simple to solve, which allows us to study the effects of occlusion and recurrence with small-scale neural networks. Therefore, any challenges to the network will only result from the introduction of occlusion. Additionally, as the stimuli are procedurally generated, they can be produced in large quantities, which enables the training of the networks.

Recurrent processing is sometimes thought of as cleaning up noise, where occlusion is a special case of noise. A simple case of noise is additive Gaussian noise, but recurrence is unlikely to show benefits in these conditions. Consider the case of detecting simple visual features show no variation, e.g. edges of different orientations. An optimal linear filter can be learnt for detecting these features. This linear filter would remain optimal under independent, additive Gaussian noise, as the expected value of the input and output will remain the same under repeated presentations. Whilst this result does not exactly hold for the case of non-linear filters that are normally used in neural networks, we might expect similar results. Therefore, we would expect no specific benefit of recurrence in the presence of additive Gaussian noise. If this is true, we can infer that the role of recurrence is not for performing object recognition in noisy conditions, generally.

In this work, we investigate the effect of occlusion on object recognition by convolutional neural networks. We extend the idea of the convolutional architecture to networks with bottom-up (B), lateral (L), and top-down (T) connections in a similar fashion to previous work (Liang and Hu, 2015; Liao and Poggio, 2016). These connections roughly correspond to processing information from lower and higher regions in the ventral visual hierarchy (bottom-up and top-down connections), and processing information from within a region (lateral connections). We choose to use the convolutional architecture as it is a parameter efficient method for building large

neural networks that can perform real-world tasks (LeCun et al., 2015). It is directly inspired by biology, with restricted receptive fields and feature detectors that replicate across the visual field (Hubel and Wiesel, 1968) and advances based on this architecture haveproduced useful models for visual neuroscience (Kriegeskorte, 2015). By using convolutional neural networks as the basis of our models, we aim to maximise the transfer of knowledge from these more biologically motivated experiments to applications in computer vision, and by using recurrent connections, we hope that our models will contribute to a better understanding of recurrent connections in biological vision whilst maintaining the benefits of scalability from convolutional architectures.

To test whether recurrent neural networks perform better than feedforward networks at occluded object recognition, we trained and tested a range of networks to perform a digit recognition task under varying levels of occlusion. Any difference in performance reflects the degree to which networks learn the underlying task of recognising the target digits, and handle the occlusion when recognising the digit. To differentiate between these two cases we also look at how well networks trained on occluded object recognition generalise to object recognition without occlusion. We also test whether recurrence shows a general advantage when dealing with noisy inputs by measuring object recognition performance with and without the presence of additive Gaussian noise. Finally, we study whether any benefit of recurrence extends to occluded object recognition where the occluder is also a target, the networks are tested on multiple digit recognition tasks where the targets overlap.

2 Material and Methods

2.1 Generative model for stimuli

To investigate the effect of occlusion in object recognition, we opt to use a task that could be solved trivially without the presence of occlusion, computer generated digit recognition. Each digit uses the same font, colour, and size. The only variable is the position of the digit, which is drawn from a uniform random distribution. This means, the only invariance problem that needs to be solved is translation invariance, which is effectively built into the convolutional networks we use.

We generate occlusion using two methods, by scattering debris across the image, digit debris, and by presenting overlapping digits within a scene, which the network has to simultaneously recognise, digit clutter.

For digit debris, we obtain debris from fragments of each of the possible targets, taking random crops from randomly selected digits. Each of these fragments are then added to a mask that

is overlaid on the target digit (Figure 1). As a result, the visual features of non-target objects, that the network has to ignore, are present in the scene. These conditions mean that summing the overall visual features present for each digit becomes a less reliable strategy for inferring the target digit. This is in contrast to deletion where there is only a removal of features that belong to the target.

Figure 1: The process for generating stimuli for digit debris. First the target digit is generated. Random crops of all possible targets are taken to create a mask of debris, which is applied to the target as an occluder.

However, within natural visual scenes, occlusion is generated by other whole objects. These objects might also be of interest to the observer. In this scenario, simply ignoring the occluding objects would not make sense. In digit clutter, these cases are simulated by generating images with multiple digits that are sequentially placed in an image, where their positions are also drawn from a uniform random distribution. This generates a series of digits that overlap, producing a relative depth order. The task of the network is then to recognise all digits that are present.

Design of these images was performed at high resolution (512×512 pixels) and, for computational simplicity, the images were resized to a low resolution (32×32 pixels) when presented to the network.

In these experiments we use stimulus sets, that vary in either the number of digits in a scene - three digits, four digits, or five digits - or the number of fragments that make up the debris - 10 fragments (light debris), 30 fragments (moderate debris), or 50 fragments (heavy debris). Examples from these stimulus sets are shown in Figure 2. This allows us to measure how the performance of the networks differ across these task types and levels of occlusion.

For each of these image sets, we randomly generated a training set of 100,000 images and a validation set of 10,000 images, which were used for the determining the hyperparameters and learning regime. All analyses where performed on an independent test set of 10,000 images.

All images underwent pixel-wise normalisation prior to being passed to the network. The mean and standard deviations used for the normalisation were always calculated from the training data.

To test the hypothesis that the benefit of recurrence is not simply for cleaning up noise, we also test the network on object recognition where the input has additive Gaussian noise. To prevent ceiling performance, we use the MNIST handwritten digit recognition data set. The MNIST data set contains 60,000 images in total that are divided into a training set of 50,000 images, a validation set of 5,000 images, and a testing set of 10,000 images. We add Gaussian noise to these images after normalisation, which allows an easy interpretation in terms of signal to noise ratio.

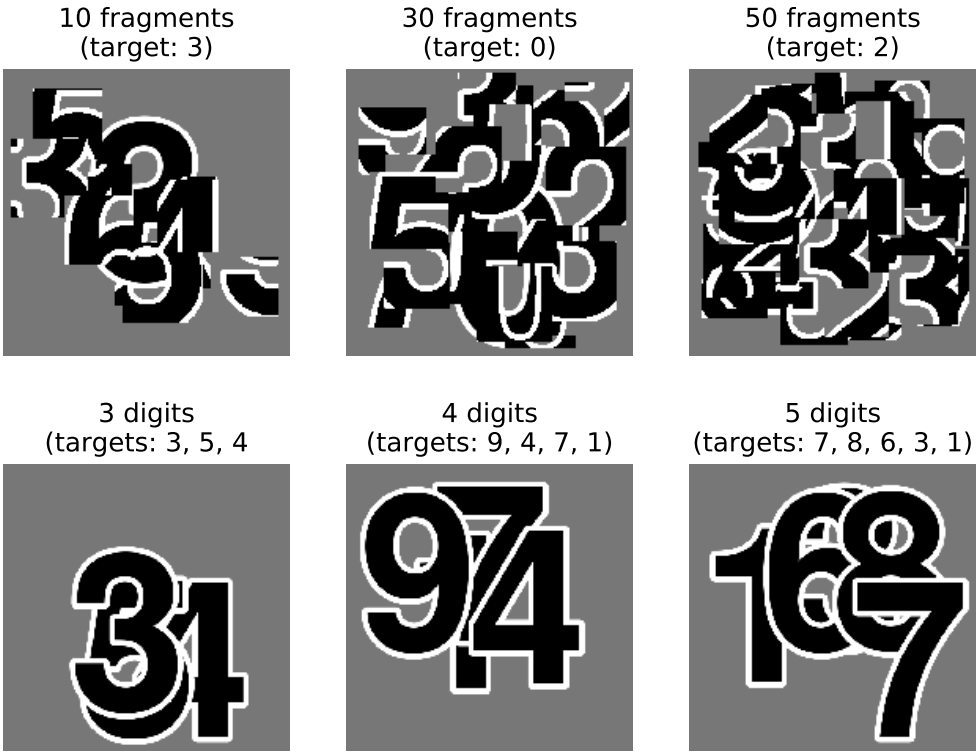

Figure 2: High resolution examples from the stimulus sets used in these experiments. The top row shows digit debris stimuli for each of the three conditions tested here, with 10, 30, and 50 fragments. The bottom row shows digit clutter stimuli with 3, 4, and 5 digits.

2.2 Models

In these experiments we use a range of convolutional neural networks (for an introduction to this architecture, see Goodfellow et al., 2016). These networks can be categorised by the particular combination of bottom-up, lateral, and top-down connections that are present. As it does not make sense to construct the networks without bottom-up connections (as information from the input cannot reach higher layers), we are left with four possible architectures with the following connections, bottom-up only (B), bottom-up and top-down (BT), bottom-up and lateral (BL), and bottom-up, lateral and top-down (BLT). Each of these architectures are illustrated schematically in Figure 3.

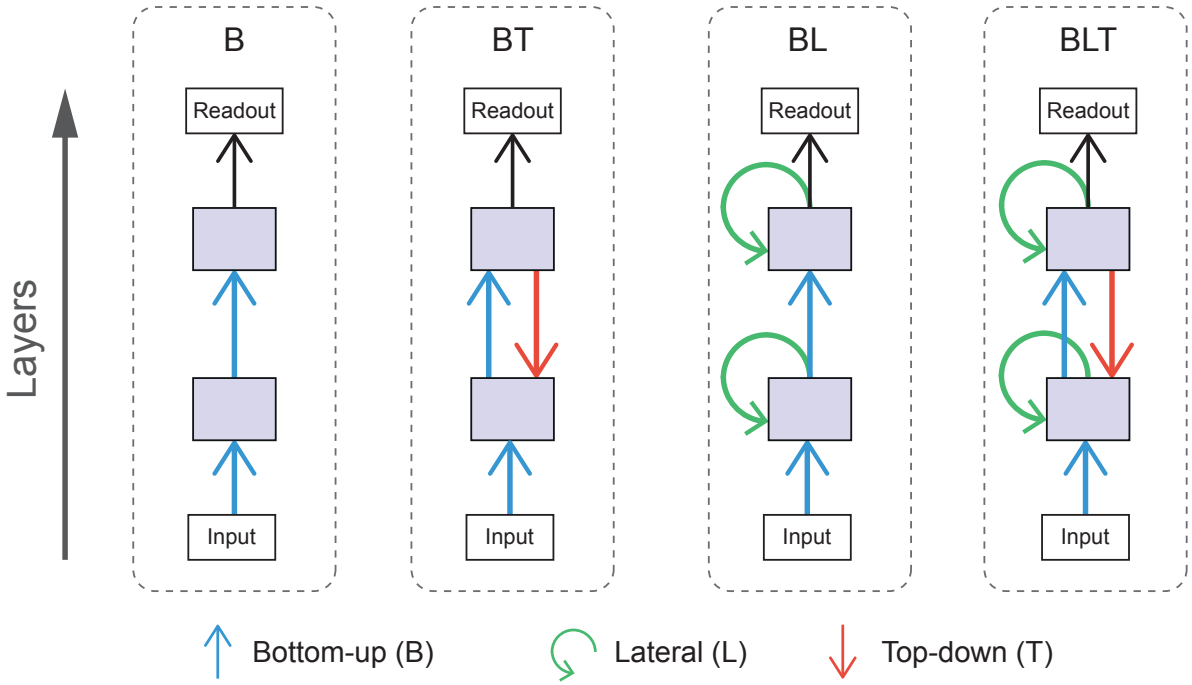

Figure 3: Schematic diagrams for each of the architectures used. Arrows indicate bottom-up (blue), lateral (green), and top-down (red) convolutions.

Adding top-down or lateral connections to feedforward models introduces cycles into the graphical structure of the network. The presence of cycles in these networks allow recurrent computations to take place, introducing internally generated temporal dynamics to the models. In comparison, temporal dynamics of feedforward networks can only be driven by changes in the input. The effect of recurrent connections can be seen through the unrolling of the computational graph across time steps. In these experiments, we run our models for four time steps and the resulting graph for BLT is illustrated in Figure 4.

As the recurrent networks (BT, BL, and BLT) have additional connections compared to purely

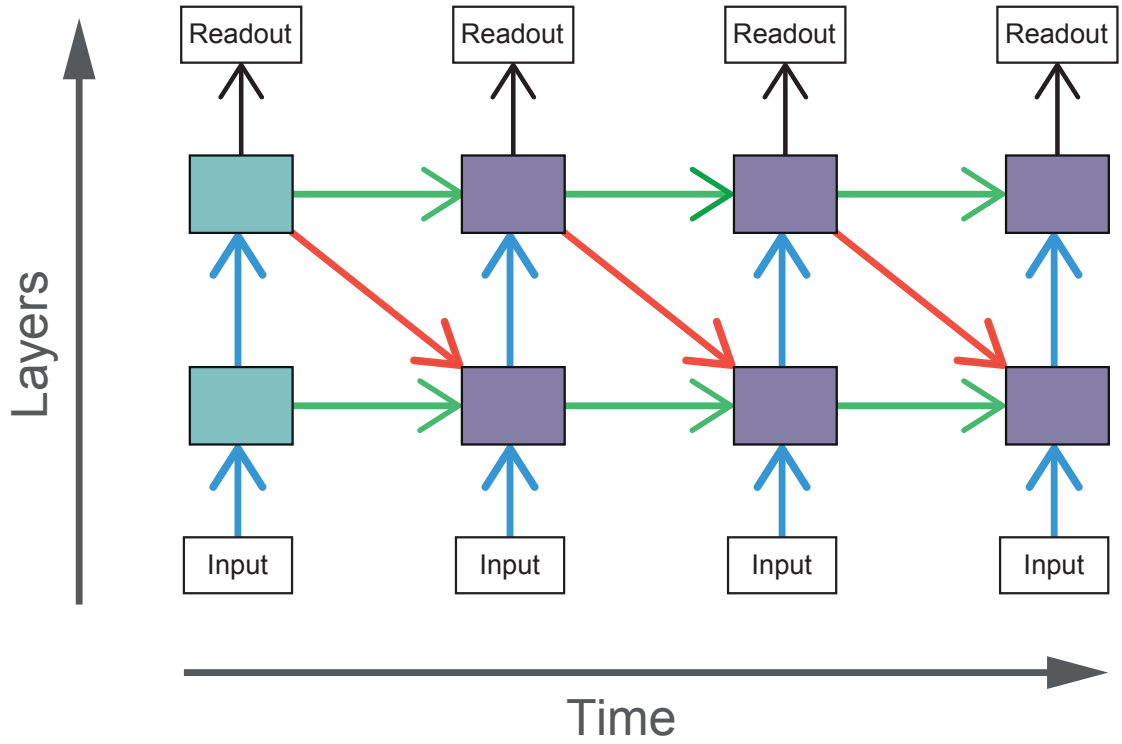

Figure 4: The computational graph of BLT unrolled over four time steps. The shaded boxes indicate hidden layers that receive purely feedforward input (blue) and those that receive both feedforward and recurrent input (purple).

feedforward networks (B), they also have a larger number of free parameters (Table 1). To control for this difference, we test two variants of B that have a more similar number parameters to the recurrent networks. The first control increases the number of features that can be learned by the bottom-up connections and the second control increases the size of individual features (known as the kernel size). These are referred to as B-F and B-K, respectively. Conceptually, B-K is a more appropriate control compared to B-F, as it effectively increases the number of connections that each unit has, holding everything else constant. In comparison, B-F increases the number of units within a layer, altering the layers representational power, in addition to changing the number of parameters. However, B-F is more closely parameter matched to some of the recurrent models, which motivates the inclusion of B-F in our experiments.

2.2.1 Architecture overview

All of the models tested consist of two hidden recurrent convolutional layers (described in Section 2.2.2) followed by a readout layer (described in Section 2.2.3). Bottom-up and lateral connections are implemented as stantdard convolutional layers with a 1×1 stride. The feedforward inputs between the hidden layers go through a max pooling operation, with a 2×2 stride and a 2×2

Model	Kernel size	No. Features	No. parameters	No. units
B	3×3	32	9,920	40,970
B-F	3×3	64	38,272	81,930
B-K	5×5	32	26,816	40,970
BT	3×3	32	19,168	40,970
BL	3×3	32	28,416	40,970
BLT	3×3	32	37,568	40,970

Table 1: Brief descriptions of the models used in these experiments including the number of learnable parameters and the number of units in each model.

kernel. This has the effect of reducing the height and width of a layer by a factor of two. As a result, top-down connections cannot be implemented by standard convolution, as the size of the top-down input from the second hidden layer would not match the size of the first hidden layer. To increase the size of the top-down input, we use transposed convolution (also known as deconvolution Zeiler et al., 2010) with an output stride of 2×2. This deconvolution increases the size of the top-down input so that it matches the size of the first hidden layer.

As feedforward networks do not have any internal dynamics and the stimuli are static, feedforward networks only run for one time step. Each of the recurrent networks are run for four time steps. This is implemented as a computational graph unrolled over time (Figure 4), where the weights for particular connections are shared across each time step. The input is also replicated at each time point.

To train the network, error is backpropagated through time for each time point (Section 2.2.4), which means that the network is trained to converge as soon as possible, rather than at the final time step. However, when measuring the accuracy, we use the predictions at the final time step as this generally produces the highest accuracy.

2.2.2 Recurrent convolutional layers

The key component of these models is the recurrent convolutional layer (RCL). The inputs to these layers are denoted by $\boldsymbol{h}_{(\tau,m,i,j)}$, which represents the vectorised input from a patch centred on location i, j, in layer m, computed at time step τ. We define $\boldsymbol{h}_{(\tau,0,i,j)}$ as the input image to the network.

For B, the lack of recurrent connections reduces RCLs to a standard convolutional layer where the pre-activation at time step τ for a unit in layer m, in feature map k, in position i, j is defined

as

$$z_{\tau,m,i,j,k} = (\boldsymbol{w}^b_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau,m-1,i,j)} + b_{m,k} \tag{1}$$

where $\tau = 0$ (as B only runs for a single time step) the convolutional kernel for bottom-up connections is given in vectorised format by $\boldsymbol{w}^b_{m,k}$ and the bias for feature map k in layer m is given by $b_{m,k}$.

In BL, lateral inputs are added to the pre-activation, giving

$$z_{\tau,m,i,j,k} = (\boldsymbol{w}^b_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau,m-1,i,j)} + (\boldsymbol{w}^l_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau-1,m,i,j)} + b_{m,k} \tag{2}$$

The term for lateral inputs $(\boldsymbol{w}^l_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau-1,m,i,j)}$ uses the same indexing conventions as the bottom-up inputs in Equation 1, where $\boldsymbol{w}^l_{m,k}$ is the lateral convolutional kernel in vectorised format. As the lateral input is dependent on outputs computed on the timestep $\tau - 1$, they are undefined for the first time step, when $\tau = 0$. Therefore, when $\tau = 0$ we set recurrent inputs to be a vector of zeros. This rule applies for all recurrent input, including top-down inputs.

In BT, we add top-down inputs to the pre-activation instead of lateral inputs. This gives

$$z_{\tau,m,i,j,k} = (\boldsymbol{w}^b_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau,m-1,i,j)} + (\boldsymbol{w}^t_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau-1,m+1,i,j)} + b_{m,k} \tag{3}$$

Where the top-down term is $(\boldsymbol{w}^t_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau-1,m+1,i,j)}$, and $\boldsymbol{w}^t_{m,k}$ is the top-down convolutional kernel in vectorised format. In our models, top-down connections can only be received from other hidden layers. As a result, top-down inputs are only given when $m = 1$ and otherwise they are set to a vector of zeros. The rule for top-down inputs also applies to top-down inputs in BLT.

Finally, we can add both lateral and top-down inputs to the pre-activation, which generates the layers we use in BLT

$$z_{\tau,m,i,j,k} = (\boldsymbol{w}^b_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau,m-1,i,j)} + (\boldsymbol{w}^l_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau-1,m,i,j)} + (\boldsymbol{w}^t_{m,k})^\mathsf{T}\ \boldsymbol{h}_{(\tau-1,m+1,i,j)} + b_{m,k} \tag{4}$$

The output, $h_{\tau,m,i,j,k}$, is calculated using the same operations for all layers. The pre-activation $z_{\tau,m,i,j,k}$ is passed through a layer of rectified linear units (ReLUs), and local response normalization (Krizhevsky et al., 2012).

ReLUs are defined as

$$\sigma_z(z_{\tau,m,i,j,k}) = \max(\{0, z_{\tau,m,i,j,k}\}) \tag{5}$$

and local response normalisation is defined for input $x_{\tau,m,i,j,k}$ as

$$\omega(x_{\tau,m,i,j,k}) = x_{\tau,m,i,j,k} \left(c + \alpha \sum_{k'=\max(0,k-n/2)}^{\min(n-1,k+n/2)} x_{\tau,m,i,j,k'} \right)^{-\beta} \tag{6}$$

For local response normalisation, we use $n = 5$, $c = 1$, $\alpha = 10^{-4}$, and $\beta = 0.5$ throughout.

The output of layer l at time step t is then given by

$$h_{\tau,m,i,j,k} = \omega(\sigma_z(z_{\tau,m,i,j,k})) \tag{7}$$

2.2.3 Readout layer

In the final layer of each time step, a readout is calculated for each class. This is performed in three steps. The first stage is a global max pooling layer, which returns the maximum output value for each feature map. The output of the global max pooling layer is then used as input to a fully connected layer with 10 output units. These outputs are passed through a sigmoind non-linearity, $\sigma_y(x)$, defined as

$$\sigma_y(x) = \frac{1}{1 + e^{-x}} \tag{8}$$

This has the effect of bounding the output between 0 and 1. The response of each of these outputs can be interpreted as the probability that each target is present or not.

2.2.4 Learning

At each time step, the networks give an output from the readout layer, which we deonte $\hat{\boldsymbol{y}}_t$, where we interpret each output as the probability that a particular target is present or not.

In training, the objective is to match this output to a ground truth $\boldsymbol{y}$, which uses binary encoding such that its elements y_i are defined as

$$y_i = \begin{cases} 1 & \text{if } i \in \boldsymbol{y'} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Where $\boldsymbol{y'}$ is the list of target digits present.

We used cross-entropy to calculate the error between $\hat{\boldsymbol{y}}_t$ and $\boldsymbol{y}$, which is summed across all time steps

$$E(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_{t=0}^{T} \sum_{i=0}^{N} y_i \cdot \log \hat{y}_{t,i} + (1 - y_i) \cdot \log(1 - \hat{y}_{t,i}) \tag{10}$$

L2-regularisation is included, with a coefficient of $\lambda = 0.0005$, making the overall loss function

$$\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = E(\hat{\boldsymbol{y}}, \boldsymbol{y}) + \lambda ||\boldsymbol{w}||_2^2 \tag{11}$$

Where $\boldsymbol{w}$ the vector of all trainable parameters in the model.

This loss function was then used to train the networks by changing the parameters at the end of each mini-batch of 100 images according to the momentum update rule

$$\boldsymbol{v}_{n+1} = \mu \boldsymbol{v}_n - \varepsilon \frac{\partial \mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y})}{\partial \boldsymbol{w}} \tag{12}$$

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n + \boldsymbol{v}_{n+1} \tag{13}$$

Where n is the iteration index, μ is the momentum coefficient, and ε is the learning rate. We use $\mu = 0.9$ for all models and set ε by the following weight decay rule

$$\varepsilon_n = \eta \delta^{\frac{e}{d}} \tag{14}$$

Where η is the initial learning rate, δ is the decay rate, e is the epoch (a whole iteration through all training images), and d is the decay step. In our experiments we use $\eta = 0.1$, $\delta = 0.1$, and $d = 40$. All networks were trained for 100 epochs.

2.3 Analysing model image set interactions

To analyse how the performance of feedforward and recurrent networks change depending on the image set being used, we test for an interaction effect on the error rates between model type and image set. As the distributions are non-normal, we use a permutation test to look for this interaction effect.

First, we calculate the error rates in one image set, A, and a second image set, B (e.g. light debris is image set A and moderate debris is image set B), averaged across all feedforward models and all recurrent models. We denote A_{FF} and B_{FF} as the average error for feedforward models for image sets A and B, respectively, and A_R and B_R as the average error for recurrent models, also for image sets A and B.

The average change in error between these too images is calculated separately for feedforward models ($\Delta E_{FF} = B_{FF} - A_{FF}$) and recurrent models ($\Delta E_R = B_R - A_R$). We then take the difference between these two averages $D = \Delta E_{FF} - \Delta E_R$ as our test statistic. This statistic has an expected value of $D = 0$ if the change in error across image sets is the same for both feedforward and recurrent models. To construct a null distribution centred on zero, the correct and incorrect predictions are permuted across all models for each individual image (i.e. within an image set, correct and incorrect predictions are permuted across all models). D is calculated for these permuted values to form the null distribution. The p-value is then obtained by making a two-tailed comparison between the observed value for D and the null distribution.

3 Results

3.1 Recognition of digits under debris

3.1.1 Learning to recognise digits occluded by debris

Networks were trained and tested to recognise digits under debris to test for a particular benefit of recurrence when recognising objects under structured occlusion. We used three image sets containing different levels of debris, 10 fragments (light debris), 30 fragments (moderate debris), and 50 fragments (heavy debris). For every model, the error rate was found to increase as the level of debris in the image increased (Figure 5)

Pairwise McNemar tests are used to compare the performance of networks on the same image set, the false discovery rate (FDR) was controlled at 0.05 for all pairwise tests. Under light and moderate debris, all but one of the pairwise differences were found to be significant ($p < 0.05$,

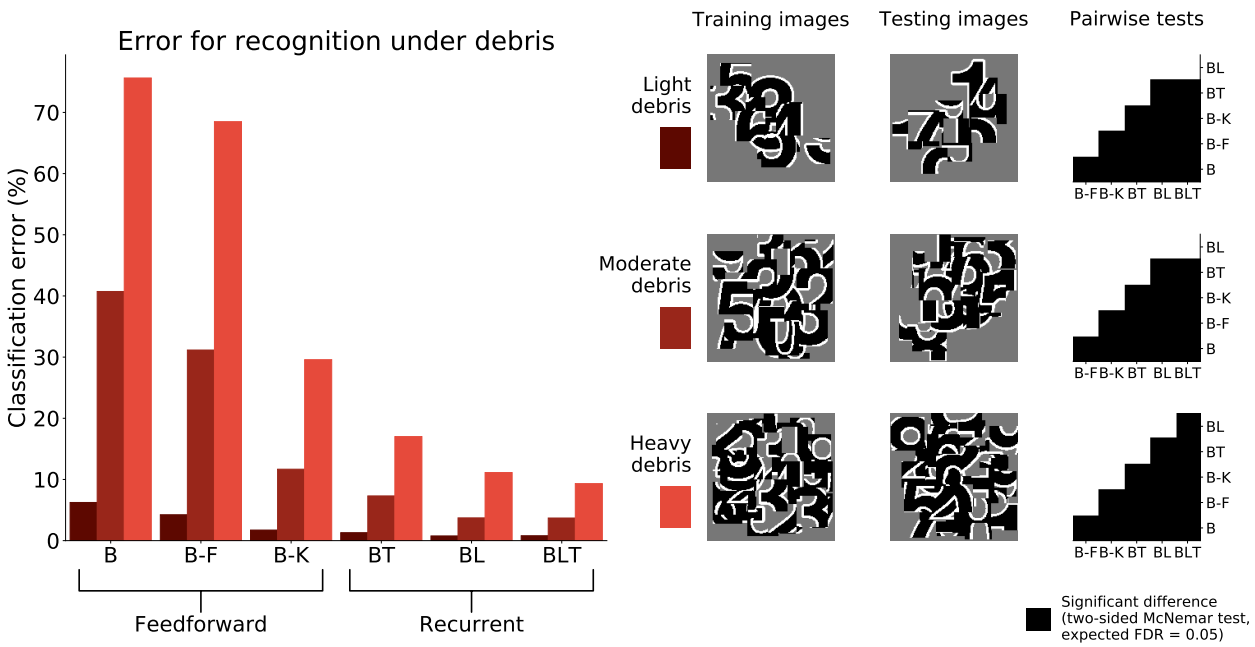

Figure 5: Classification error for all models on single digit detection under varying levels of debris. Examples of the images used to train and test the networks are also shown. Matrices to the right indicate significant results of pairwise McNemar tests. Comparisons are across models and within image sets. Black boxes indicate significant differences at p $<$ 0.05 when controlling the expected false discovery rate at 0.05.

expected FDR $=$ 0.05). with no significant difference between BL or BLT for light ($\chi^2(1, N = 10,000) = 0.04$, $p = 0.835$) and moderate debris ($\chi^2(1, N = 10,000) = 0.00$, $p = 0.960$). The recurrent models always performed best, with BL and BLT outperforming BT. Of the feedforward models, B-K was the best performing. The error rates for each of the models are shown in Table 2.

Under heavy debris all pairwise differences were significant ($p < 0.05$, expected FRD $=$ 0.05) including the difference between BL and BLT, which was not significant under light and moderate debris, with BLT outperforming BL.

The interaction effect between image set and recurrence is significant when comparing light debris

Image set	B	B-F	B-K	BT	BL	BLT
Light debris	6.24%	4.23%	1.73%	1.30%	0.77%	0.80%
Moderate debris	40.73%	31.16%	11.68%	7.31%	3.72%	3.70%
Heavy debris	75.63%	68.49%	29.58%	17.01%	11.13%	9.32%

Table 2: Classification error for all of the models on single digit detection with varying levels of debris.

Image set	B	B-F	B-K	BT	BL	BLT
Tested on debris	75.63%	68.49%	29.58%	17.01%	11.13%	9.32%
Tested without debris	79.37%	69.88%	0.34%	3.10%	2.11%	0.55%

Table 3: Classification error for all of the models on single digit detection with varying levels of debris.

(image set A) to moderate debris (image set B), $\Delta E_{FF} = 23.79$, $\Delta E_R = 3.95$, $D = 19.84$, $p < 0.05$, and when comparing error rates for moderate debris (image set A) and heavy debris (image set B), $\Delta E_{FF} = 30.04$, $\Delta E_R = 7.58$, $D = 22.47$, $p < 0.05$. Both of these results suggest a much larger increase in error for feedforward networks as the amount of debris increases, in comparison to recurrent networks.

3.1.2 Learning to recognise unoccluded digits when trained with occlusion

To test if the networks learn a good model of the digit when trained to recognise the digit under debris, we test the performance of networks when recognising unoccluded digits.

When networks were trained to recognise digits under heavy debris, and tested to recognise unoccluded digits, we found all pairwise differences to be significant ($p < 0.05$, expected FDR = 0.05, Figure 6). The best performing network was B-K, followed by recurrent networks. B and B-F performed much worse than all of the other networks (Table 3).

3.2 Recognition of multiple digits

To examine the ability of the networks to handle occlussion when the occluder is not a distractor, the networks were tested on their ability to recognise multiple overlapping digits.

When recognising three digits simultaneously, recurrent networks generally outperformed feedforward networks (Figure 7), with the exception of BT and B-K where no significant difference was found ($\chi^2(1, N = 30,000) = 3.53$, $p = 0.06$). All other differences were found to be significant ($p < 0.05$, expected FDR = 0.05). A similar pattern is found when recognising both four and five digits simultaneously. However, in both four and five digit tasks, all pairwise differences were found to be significant, with B-K outperforming BT (Figure 7).

For every model, the error rate increases as the number of digits increase (Table 4). An interaction effect was found between image set and recurrence when comparing recognising three digits (image set A) and four digits simultaneously (image set B), $\Delta E_{FF} = 6.11$, $\Delta E_R = 5.08$, $D = 1.04$,

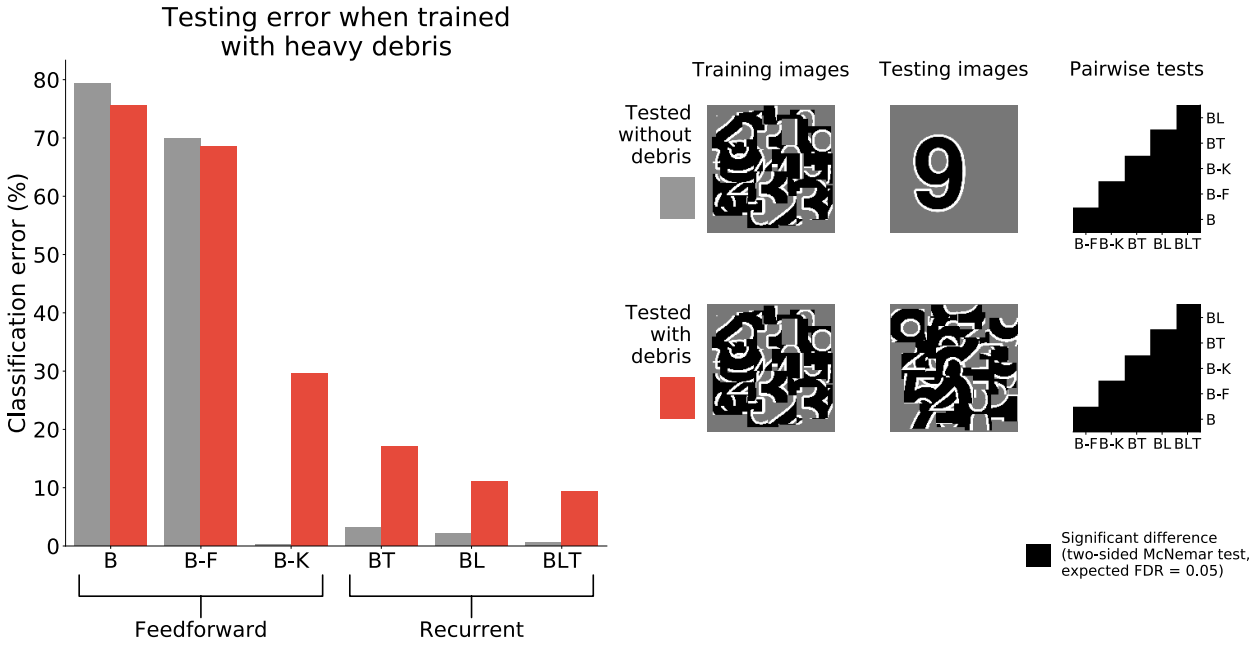

Figure 6: Classification error for all models trained under heavy debris conditions and testewd with or without debris. Examples of the images used to train and test the networks are also shown. Matrices to the right indicate significant results of pairwise McNemar tests. Comparisons are across models and within image sets. Black boxes indicate significant differences at $p < 0.05$ when controlling the expected false discovery rate at 0.05.

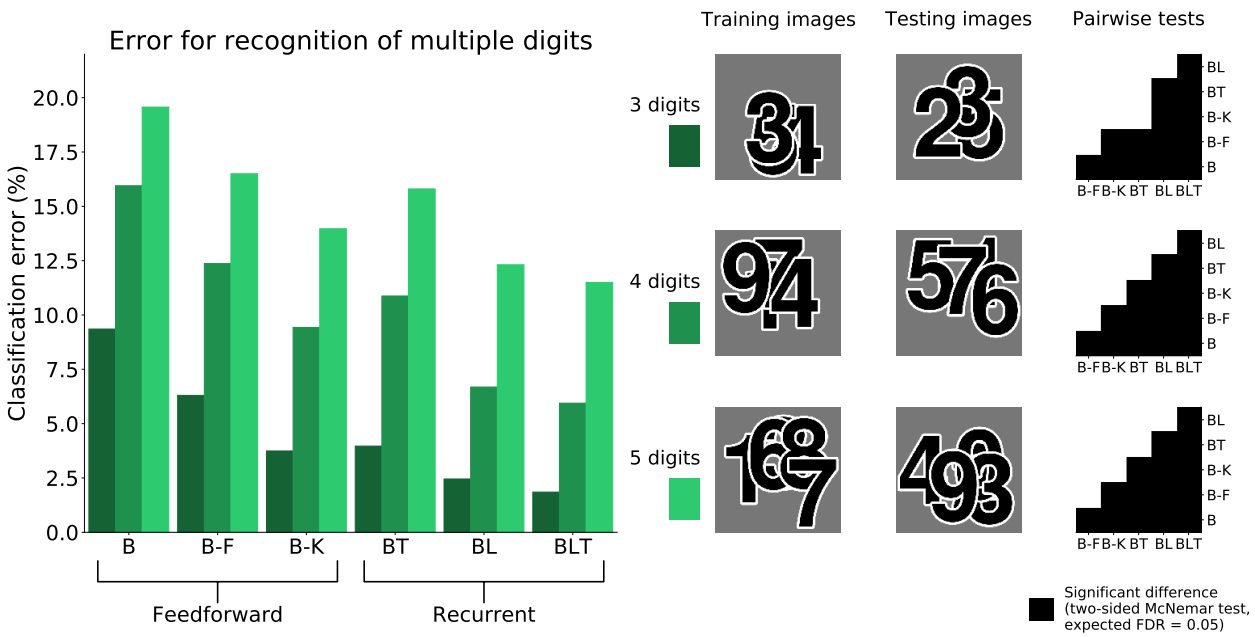

Figure 7: Classification error for all models on multiple digit detection with varying number so of digits. Examples of the images used to train and test the networks are also shown. Matrices to the right indicate significant results of pairwise McNemar tests. Comparisons are across models and within image sets. Black boxes indicate significant differences at $p < 0.05$ when controlling the expected false discovery rate at 0.05.

Image set	B	B-F	B-K	BT	BL	BLT
3 digits	9.35%	6.30%	3.74%	3.97%	2.45%	1.85%
4 digits	15.95%	12.37%	9.43%	10.88%	6.69%	5.94%
5 digits	19.57%	16.50%	13.97%	15.80%	12.31%	11.50%

Table 4: Classification error for all of the models multiple digit recognition with varying numbers of targets.

$p < 0.05$. This suggests a greater increase in error for feedforward networks compared to recurrent networks as the number of digits increases from three to four. An interaction effect was also found when comparing recognising four digits (image set A) and five digits (image set B) simultaneously, $\Delta E_{FF} = 4.10$, $\Delta E_R = 5.37$, $D = -1.27$, $p < 0.05$. This shows the opposite relationship, with recurrent networks showing a larger increase in error than feedforward networks.

3.3 MNIST with Gaussian noise

To test the hypothesis that the benefit of recurrence does not extend to dealing with noise in general, we test the performance of the networks on MNIST with unstructured additive Gaussian noise.

For MNIST with and without Gaussian noise, all recurrent networks performed significantly better than the feedforward models, and all pairwise comparisons between feedforward models were found to be significant ($p < 0.05$, expected FDR = 0.05). However, no significant difference was found within recurrent models (Figure 8). These relationships were found for both MNIST with and without Gaussian noise.

When testing for interactions between the presence of recurrence and image sets, we found that D was not significantly different from zero when comparing models trained and tested on MNIST without Gaussian noise, image set A, and without Gaussian noise, image set B ($\Delta E_{FF} = -0.05$, $\Delta E_R = -0.05$, $D = 0.00$, $p = 0.97$). This suggests that a change in image set has the same effect on error rates for both recurrent and feedforward models, with a slight decrease in error when training and testing on MNIST with Gaussian noise.

4 Discussion

We found support for the hypothesis that recurrence helps when recognising objects under occlusion. This benefit does not appear to extend to unstructured noise in general. The benefit of

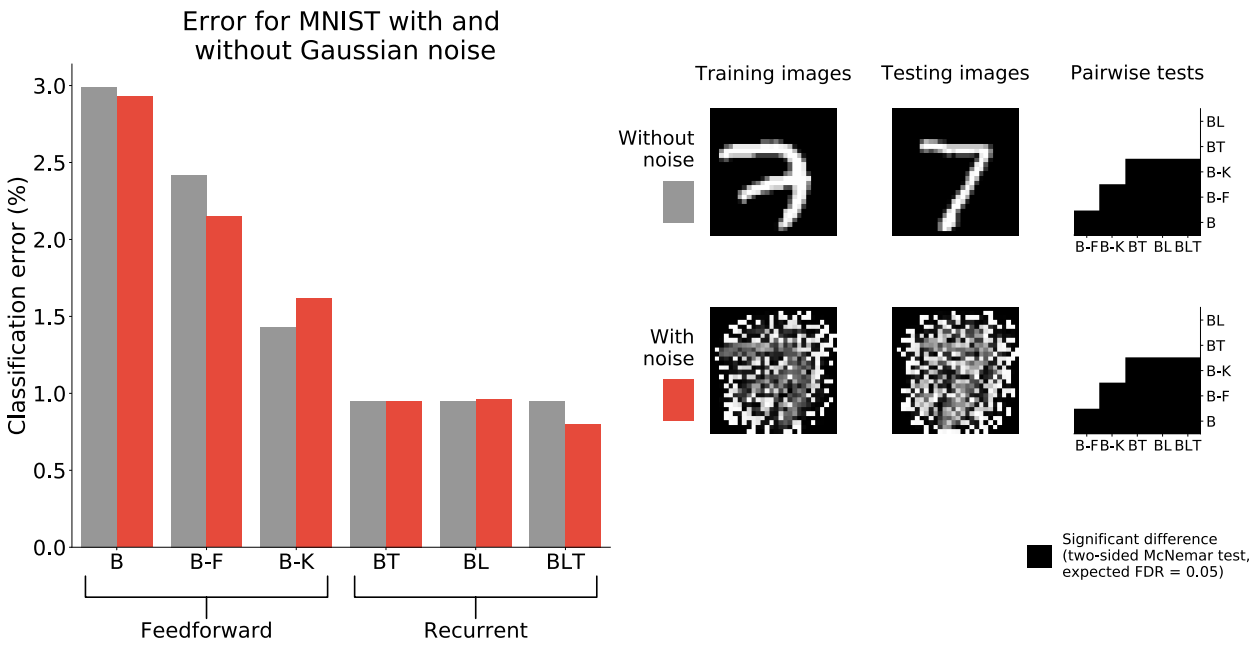

Figure 8: Classification error for all models on recognition in MNIST with and without Gaussian noise. Examples of the images used to train and test the networks are also shown. Matrices to the right indicate significant results of pairwise McNemar tests. Comparisons are across models and within image sets. Black boxes indicate significant differences at $p < 0.05$ when controlling the expected false discovery rate at 0.05.

recurrence for occluded object recognition appears to be particularly strong in the case of occlusion generated by a non-target, with much larger increases in error observed for feedforward networks compared to recurrent networks when the amount of occlusion is increased. In the multiple digit recognition tasks, where the occlusion is generated by other targets, the best performing networks are still recurrent. However, the difference in performance is not as large and recurrent networks do not exclusively outperform feedforward networks, with B-K outperforming BT.

In these experiments, BLT is the best performing network across all tasks, but it performs at the same level as BL when recognising digits under light and moderate levels of debris. This suggests that at lower levels of occlusion, feedforward and lateral connections are sufficient for performing the task. However, top-down connections become beneficial when the task involves recognising digits under heavier levels of debris.

Of the feedforward models, B-K is always the best performing and outperforms recurrent models in some tasks, in the case of BT. One potential explanation is that B-K encorporates some of the benefits of recurrence by having a larger receptive field. This is because recurrence increases the effective receptive field of a unit by receiving input from neighbouring units. This may also explain why BT is the worst performing recurrent model (and outperformed by B-K) as it does not have lateral connnections that more directly integrate information from neighbouring units,

but information has to go through a higher layer first in order to achieve this.

We find evidence to suggest that feedforward networks have particular difficulty recognising objects under occlusion generated by debris, and not just learning the task of recognising digits when trained with heavily occluded objects. This is given by the finding that feedforward networks (specifically B-K) can perform very well at recognising the digit without occlusion, when trained to recognise digits under occlusion. This suggests they have learnt a good model of the underlying task (Section 3.1.2). However, B-K performs worse than recurrent models when recognising the target under occlusion. This indicates that the B-K has difficulty recognising the digit under occlusion rather than a problem with learning to perform the task whereas, recurrent networks have much less difficulty of performing the task of recognising the target under occlusion. This further supports the idea that recurrent processing helps in occluded object recognition.

Recurrent networks also outperformed the parameter matched controls on object recognition tasks where no occlusion was present (Section 3.3). This is consistent with previous work that has shown that recurrent networks, similar in architecture to the BL networks used here, perform strongly compared to other feedforward models with larger numbers of parameters (Liang and Hu, 2015). Therefore, some level of recurrence may be beneficial in standard object recognition, and idea that is supported by neural evidence that shows object information unfolding over time, even without the presence of occlusion (Sugase et al., 1999; Brincat and Connor, 2006; Freiwald and Tsao, 2010; Carlson et al., 2013; Cichy et al., 2014; Clarke et al., 2015).

This work suggests that networks with recurrent connections generally show performance gains relative to feedforward models when performing occluded object recognition. However, it does not indicate which of these models best describe human object recognition. Future comparisons to neural or behavioural data will be needed to test the efficacy of these models. For example, as these models are recurrent and unfold over time, they can be used to predict human recognition dynamics for the same stimuli, such as reaction time distributions and the order that digits are reported, in the multiple digit recognition tasks.

Furthermore, we can study whether the activation patterns of these networks predict neural dynamics of object recognition. This is similar to previous work that has attempted to explain neural dynamics of representations using individual layers of deep feedforward networks (Cichy et al., 2016), but by using the recurrent models we can directly relate temporal dynamics in the model to temporal dynamics in the brain. For instance, in tasks with multiple targets (such as those in Section 3.2) we can look at the target representations over recurrent iterations and layers in the model, and compare this to the spatiotemporal dynamics of multiple object representations in neural data. Testing these models against this experimental data will allow us to better understand the importance of lateral and top-down connections, in these models, for explaining

neural data.

In addition, whilst we know that the adding recurrent connections leads to performance gains in these models, we do not know the exact function of these recurrent connections. For instance, in the case of occlusion, the recurrent connections might complete some of the missing information from occluded regions of the input image, which would be consistent with experimental evidence in cases where parts of the image have been deleted (O'Reilly et al., 2013; Smith and Muckli, 2010). Alternatively, as our occluders contain visual features that could be potentially misleading, recurrent connections may have more of an effect of suppressing the networks representation of the occluders. If these networks are to be useful models of visual processing, then it is important that future work attempts to understand the underlying processes taking place.

It could be argued that BLT performs the best due to the larger number of parameters it can learn. However, we know that the performance of these networks is not only explained by the number of learnable parameters, as B-F has the largest number of parameters of the models tested (Table 1) and performs poorly in all tasks relative to the recurrent models. Finding exactly parameter matched controls for these models that are conceptually sound is difficult. As discussed earlier (Section 2.2), altering the kernel size of the feedforward models is the best control, but this provides a relatively coarse-grained way to match the number of parameters. Altering the number of learnt features allows more fine-tuned controls for the number of parameters, but this also changes the number of units in the network, which is undesirable. We believe that the models used here represent a good compromise between exact parameter matching and the number of units in each model.

This research suggests that recurrent convolutional neural networks can outperform their feedforward counterparts in the task of occluded object recognition, building on previous work showing a benefit of recurrent connections in non-convolutional networks where parts of target objects are deleted (O'Reilly et al., 2013). This work represents initial steps for using recurrent convolutional neural networks as models of visual object recognition. Scaling up these networks and training them on large sets of natural images (e.g. Russakovsky et al., 2015) will also be important for developing models that mirror processing in the visual system more closely. Future work with these networks will allow us to capture temporal aspects of visual object recognition that are currently neglected in most models, whilst incorporating the important spatial aspects that have been established by prior work (DiCarlo et al., 2012). Modelling these temporal properties will lead to a more complete understanding of visual object recognition in the brain.

Acknowledgements

This research was funded by the UK Medical Research Council (Programme MC-A060- 5PR20), by a European Research Council Starting Grant (ERC-2010-StG 261352), and by the Human Brain Project (EU grant 604102 Context-sensitive multisensory object recognition: a deep network model constrained by multi-level, multi-species data).

References

Brincat, S. L. and Connor, C. E. (2006). Dynamic shape synthesis in posterior inferotemporal cortex. *Neuron*, 49(1):17–24.

Carlson, T., Tovar, D. A., Alink, A., and Kriegeskorte, N. (2013). Representational dynamics of object vision: the first 1000 ms. *Journal of vision*, 13(10):1–19.

Cichy, R. M., Khosla, A., Pantazis, D., Torralba, A., and Oliva, A. (2016). Comparison of deep neural networks to spatio-temporal cortical dynamics of human visual object recognition reveals hierarchical correspondence. *Scientific reports*, 6.

Cichy, R. M., Pantazis, D., and Oliva, A. (2014). Resolving human object recognition in space and time. *Nature neuroscience*, 17(3):455–462.

Clarke, A., Devereux, B. J., Randall, B., and Tyler, L. K. (2015). Predicting the time course of individual objects with meg. *Cerebral Cortex*, 25:3602–3612.

DiCarlo, J. J., Zoccolan, D., and Rust, N. C. (2012). How does the brain solve visual object recognition? *Neuron*, 73(3):415–434.

Felleman, D. J. and Van Essen, D. C. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cerebral cortex*, 1(1):1–47.

Freiwald, W. A. and Tsao, D. Y. (2010). Functional compartmentalization and viewpoint generalization within the macaque face-processing system. *Science*, 330(6005):845–851.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Convolutional networks. In *Deep Learning*, chapter 9, pages 330 – 372. MIT Press. `http://www.deeplearningbook.org`.

Güçlü, U. and van Gerven, M. A. (2015). Deep neural networks reveal a gradient in the complexity of neural representations across the ventral stream. *Journal of Neuroscience*, 35(27):10005–10014.

Hubel, D. H. and Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243.

Hung, C. P., Kreiman, G., Poggio, T., and DiCarlo, J. J. (2005). Fast readout of object identity from macaque inferior temporal cortex. *Science*, 310(5749):863–866.

Isik, L., Meyers, E. M., Leibo, J. Z., and Poggio, T. (2014). The dynamics of invariant object recognition in the human visual system. *Journal of neurophysiology*, 111(1):91–102.

Johnson, J. S. and Olshausen, B. A. (2005). The recognition of partially visible natural objects in the presence and absence of their occluders. *Vision research*, 45(25):3262–3276.

Keysers, C., Xiao, D.-K., Földiák, P., and Perrett, D. I. (2001). The speed of sight. *Journal of Cognitive Neuroscience*, 13(1):90–101.

Khaligh-Razavi, S.-M. and Kriegeskorte, N. (2014). Deep supervised, but not unsupervised, models may explain it cortical representation. *PLoS Comput Biol*, 10(11):e1003915.

Kriegeskorte, N. (2015). Deep neural networks: a new framework for modeling biological vision and brain information processing. *Annual Review of Vision Science*, 1:417–446.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Liang, M. and Hu, X. (2015). Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3367–3375.

Liao, Q. and Poggio, T. (2016). Bridging the gaps between residual learning, recurrent neural networks and visual cortex. *arXiv preprint arXiv:1604.03640*.

Majaj, N. J., Hong, H., Solomon, E. A., and DiCarlo, J. J. (2015). Simple learned weighted sums of inferior temporal neuronal firing rates accurately predict human core object recognition performance. *Journal of Neuroscience*, 35(39):13402–13418.

Markov, N. T., Vezoli, J., Chameau, P., Falchier, A., Quilodran, R., Huissoud, C., Lamy, C., Misery, P., Giroud, P., Ullman, S., et al. (2014). Anatomy of hierarchy: feedforward and feedback pathways in macaque visual cortex. *Journal of Comparative Neurology*, 522(1):225–259.

O'Reilly, R. C., Wyatte, D., Herd, S., Mingus, B., and Jilk, D. J. (2013). Recurrent processing during object recognition. *Frontiers in psychology*, 4:124.

Potter, M. C. (1976). Short-term conceptual memory for pictures. *Journal of Experimental Psychology: Human Learning and Memory*, 2(5):509–522.

Riesenhuber, M. and Poggio, T. (1999). Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019–1025.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Serre, T., Oliva, A., and Poggio, T. (2007). A feedforward architecture accounts for rapid categorization. *Proceedings of the national academy of sciences*, 104(15):6424–6429.

Smith, F. W. and Muckli, L. (2010). Nonstimulated early visual areas carry information about surrounding context. *Proceedings of the National Academy of Sciences*, 107(46):20099–20103.

Sporns, O. and Zwi, J. D. (2004). The small world of the cerebral cortex. *Neuroinformatics*, 2(2):145–162.

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

Tang, H., Buia, C., Madhavan, R., Crone, N. E., Madsen, J. R., Anderson, W. S., and Kreiman, G. (2014). Spatiotemporal dynamics underlying object completion in human ventral visual cortex. *Neuron*, 83(3):736–748.

Thorpe, S., Fize, D., and Marlot, C. (1996). Speed of processing in the human visual system. *Nature*, 381(6582):520–522.

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

Wyatte, D., Curran, T., and O'Reilly, R. (2012). The limits of feedforward vision: Recurrent processing promotes robust object recognition when objects are degraded. *Journal of Cognitive Neuroscience*, 24(11):2248–2261.

Wyatte, D., Jilk, D. J., and O'Reilly, R. C. (2014). Early recurrent feedback facilitates visual object recognition under challenging conditions. *Frontiers in Psychology*, 5:674.

Yamins, D. L., Hong, H., Cadieu, C., and DiCarlo, J. J. (2013). Hierarchical modular optimization of convolutional networks achieves representations similar to macaque it and human ventral stream. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3093–3101. Curran Associates, Inc.

Yamins, D. L., Hong, H., Cadieu, C. F., Solomon, E. A., Seibert, D., and DiCarlo, J. J. (2014). Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624.

Zeiler, M. D., Krishnan, D., Taylor, G. W., and Fergus, R. (2010). Deconvolutional networks. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2528–2535. IEEE.